

Efficient characterization, calibration, and correction of quantum information processors

by

John Paul Marceaux

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Applied Science and Technology

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor K. Birgitta Whaley, Co-chair

Dr. Kevin C. Young, Co-chair

Professor Hartmut Häffner

Professor Alp Sipahigil

Spring 2025

Efficient characterization, calibration, and correction of quantum information processors

Copyright 2025
by
John Paul Marceaux

Abstract

Efficient characterization, calibration, and correction of quantum information processors

by

John Paul Marceaux

Doctor of Philosophy in Applied Science and Technology

University of California, Berkeley

Professor K. Birgitta Whaley, Co-chair

Dr. Kevin C. Young, Co-chair

The key challenge to useful quantum information processing is the deleterious effects of noise on intricate and fragile quantum states that are extraordinarily sensitive to environmental variations. The problem of quantum noise will be surmounted by quantum computer engineering. I propose a three step approach to the engineering problem 1) understand the noise and its nature, 2) remove all controllable sources of errors, and 3) correct for uncontrollable sources of error, ie., characterize, calibrate, and correct which are the three C's in the title of this thesis and form the three core parts of this work. In order to produce useful techniques, I have made great efforts to design an approach that is *efficient* in both classical and quantum computational resources.

Characterization concerns the definition and estimation of quantum noise models. A complete tomographic theory, such as that provided by gate set tomography, is theoretically useful, but efficiency constraints motivate reduced protocols such as phase estimation. Calibration deals with techniques to control a quantum processor and minimize its operational error. For this purpose, I adapt the celebrated “linear, quadratic, Gaussian” (LQG) control of classical theory in the context of quantum calibration for the first time. LQG is the *optimal* control algorithm when model assumptions are met. Correction removes any remaining error, which is usually incoherent sources that cannot be corrected for with optimal classical control. I introduce a novel metric of code performance that can be used to identify optimal codes for efficient correction of errors under non-uniform noise distributions. The techniques presented in this work address foundational issues in quantum computer engineering.

Dedicated to the memory of my grandparents:
John Clifford (Bill) “paw-paw” Marceaux
Marie Mazie “maw-maw” Miller-Marceaux
Billy Neal “pops” Barnett,
Jane Patricia “shug” Coco-Barnett

Contents

Contents	ii
List of Figures	v
List of Tables	xii
I Introduction	1
1 Context of the work	2
1.1 Quantum information processing devices	2
1.2 Gate-based quantum computation	5
1.3 Prominent architectures	6
1.4 Potential use cases of quantum computers	8
2 Mathematical foundations	9
2.1 Compositions of Hilbert spaces	9
2.2 Lie groups	10
2.3 Lie algebras	11
2.4 Important discrete group	12
2.5 Technical elements from classical information theory	13
II Characterization	18
3 Quantum noise models	19
3.1 Quantum information dynamics	19
3.2 Unitary models and coherent errors	22
3.3 Quantum channels and incoherent errors	24
3.4 Pauli error models	32
3.5 Depolarizing models	36
3.6 Non-Markovian models	37

4	Gate set tomography	39
4.1	Gate set models	40
4.2	Linear GST	42
4.3	Long Sequence GST	44
4.4	Error generators	46
4.5	Gate set gauges	47
4.6	Outstanding problems	52
5	Phase estimation for calibration	56
5.1	The basic primitive	57
5.2	Controlled-Z RPE	62
5.3	Qutrit RPE	64
5.4	Decoupled phase estimation	69
5.5	Twirled robust phase estimation	72
III	Calibration	74
6	Filtering and system estimation	75
6.1	Bayesian filtering	77
6.2	Linear Kalman filtering	78
6.3	Non-linear filtering	83
6.4	Kalman filters and gate set tomography	87
6.5	Numerical verification	90
6.6	Implementation considerations	92
7	Calibration and stabilization	98
7.1	Classical system models of quantum devices	99
7.2	Principles of control law design	104
7.3	Classical observers of quantum devices	107
7.4	Linear-quadratic regulators	109
7.5	Dual control estimation	111
7.6	Calibration algorithm	113
7.7	Numerical implementation	116
7.8	Discussion	125
IV	Correction	128
8	Topics in quantum error correction	129
8.1	Stabilizer codes	130
8.2	Tableau representation of a stabilizer code	137
8.3	Codespace entropy calculus	138

8.4	Decoding	140
8.5	Channel representations of quantum codes	144
8.6	Fault tolerance	148
9	Entropic measures of code performance	151
9.1	Entropy as a measure of code performance	152
9.2	The codespace entropy theorem	153
9.3	Applications	156
9.4	Comparison with coherent information	158
9.5	Numerical studies	160
9.6	Future directions	165
V	Conclusion	169
10	Conclusion	170
10.1	Summary	170
10.2	Future directions	172
	Bibliography	174

List of Figures

1.1	<i>Hierarchy of quantum computer engineering problems.</i> My work in this thesis concerns the problems in processor characterization, error correction, and at their intersection in code co-design.	4
1.2	<i>An example of a random quantum circuit.</i> Single-qubit operations are represented as boxes that appear on the wire corresponding to the qubit and labeled according to their identity. Two-qubit operations appear like vertical links that connect the wires of two qubits. The temporal execution of the circuit proceeds from left to right. In this example, we start with initial state preparations in the all 0's state, denote by the kets $ 0\rangle$ on the left, and we measure in the computational basis, denoted by the meters on the right.	6
2.1	<i>Typical sets in the space of all sets.</i> The product set $\mathcal{Z}^n \times \mathcal{Y}^n$ defines the space of all possible pairings. The marginally typical sets $A_{Z,\varepsilon}^{(n)}$ and $A_{Y,\varepsilon}^{(n)}$ contain the sequences $z^{(n)}$ and $y^{(n)}$ that typically occur with non-vanishing probability, and the jointly typical set $B_\varepsilon^{(n)}$ contains the pairings that typically occur. Observe the nested structure of the sets.	16
3.1	<i>Relationships between quantum channel representations.</i> The four canonical representations of quantum channels are depicted with the associated translation action between representations. Each translation action is represented by a particular line type (solid, dotted, dashed) The Pauli transfer matrix representation should be understood as a particular choice of basis for the Liouville representation. See also [WBC11].	27
3.2	<i>Hierarchical classification of quantum noise channels.</i> A general CPTP noise channel can be separated into unitary and non-unitary dynamics. The non-unitary dynamics correspond to decoherence, which can be further classified into a number of sub-categories.	30

- 4.1 *Tomographic Reconstruction.* The structure of the circuits required for state, process, and measurement tomography are shown. Each of these protocols reconstructs an initially-unknown quantum operation (a state ρ , process G , or POVM M) by combining that operation into simple circuits with a set of *known* complementary operations. The known operations form a reference frame for estimation of the unknown operation’s matrix elements. If the reference frame is *informationally complete*, then all matrix elements of the unknown operation can be learned. We denote reference frame operations with “primed” symbols (ρ' and M') to indicate that they are *effective* (rather than *native*) state preparations and measurements, usually implemented by applying gate operations after or before a native state preparation or measurement. These techniques are limited by systematic errors stemming from imperfect *a priori* knowledge of the ρ' and M' . Figure and caption reproduced with permission from [Nie+21]. 40
- 4.2 *Linear Gate Set Tomography.* Structures of the two types of circuits required by the LGST algorithm. **Upper panel:** each native gate, G_k , is sandwiched between the elements of informationally complete sets of *effective state preparations*, $\{\rho'_i\}$, and of *effective measurements*, $\{M'_j\}$. These are the same circuits that QPT requires to characterize G_k . (a) Shows these circuits in their simplest form, with each informationally complete set displayed as a unit. (b) Depicts the common case when the set of effective preparations (measurements) is implemented by following (preceding) a single native state preparation (measurement) operation with a *fiducial circuit* F_f (H_h). (c) Exemplifies that the fiducial circuits are composed of native gates G_i and gives the circuit entirely in terms of native operations. **Lower panel:** because LGST does not assume knowledge of ρ'_i and M'_j , it requires circuits that sandwich nothing between pairs of fiducials in order to be self-calibrating. The circuit diagrams in (d), (e), and (f) parallel those in (a), (b), and (c). LGST also requires the circuits that perform state (measurement) tomography on ρ (M), but these are not explicitly shown. They are similar to (d) – (f) (replacing ρ' with ρ or M' with M), and are actually included as a subset of these circuits when the gate set contains only a single native state preparation (measurement) and one of the preparation (measurement) fiducial circuits is the empty (do-nothing) circuit. (Figure and caption reproduced with permission from [Nie+21].) 54

4.3	<i>Long Sequence Gate Set Tomography.</i> The structure of circuits in the standard LSGST experiment design, shown in increasing detail. (a) Each GST circuit consists of an effective state preparation ρ' followed by a <i>germ</i> circuit g repeated p times, followed by an effective measurement $M' = \{E'_i\}$. (b) Effective preparations are often implemented by a native state preparation ρ followed by a <i>preparation fiducial circuit</i> F , and similarly effective measurements are often implemented by <i>measurement fiducial circuit</i> H followed by a native measurement M . (c) Writing the fiducials and germ in terms of the native gate operations reveals how the native operations of a gate set compose to form a GST circuit. (Figure and caption reproduced with permission from [Nie+21].)	55
5.1	<i>Metrics before and after calibration.</i> Panel (a) shows the during-gate Hamiltonian error rates and panel (b) shows the angle error. The Hamiltonian error was measured with GST and indicate that significant resonant and cross-resonant terms persist in the drive Hamiltonian after calibration, which motivates future work to efficiently estimate and calibrate such small non-commuting terms. The angle error was measured with RPE, and the error bars correspond to the standard root-mean-square (RMS) error regions of an RPE estimate ($\pi/(2d)$ for RPE of depth d). While RPE does estimate remaining errors in the local IZ- and ZI-phases after calibration, the magnitude of the errors falls within the RMS bounds of RPE.	64
6.1	<i>Kalman filtering on quantum circuit outcomes.</i> Closed-loop control techniques requires online filters that can update estimates in real time at rates comparable to circuit execution. The linear and extended Kalman filter can be adapted for the purpose of streaming quantum tomography estimation. This chapter focuses on identification of model parameters, and § 7 introduces control laws (the dashed gray triangle in the bottom right) that use active feedback.	76
6.2	<i>Illustration of Gaussian updates used in a Kalman filter.</i> A prior Gaussian, here represented in a 2D vector space, is propagated through the model into a predicted observation. The prediction is compared with the observation, and application of Bayes rule results in a posterior distribution.	79
6.3	<i>Flowchart of Kalman filtered GST.</i> Diagram shows the flow of information between sub-components in a Kalman GST algorithm.	87

6.4	<i>Influence of the prior distribution on Bayesian updates.</i> The top two plots show the prior and likelihood for a short and a long circuit for a simplified 1-parameter model, and the bottom two plots show the resulting posterior when calculated via Bayes rule. In the case of the short circuit, the wide circuit prior moves closer to the true value than the narrow circuit prior. In the case of the long circuit, the wide prior produces a multi-modal distribution when multiplied with the likelihood, which violates the assumptions of the Kalman filter, and the narrow prior results in a unimodal distribution that can be well approximated as a Gaussian. This example illustrates that, in order to assume Gaussian priors and Gaussian likelihoods needed for Kalman filtering, the length of the circuit should be selected such that the likelihood is unimodal on the principle support of the prior.	96
6.5	<i>Numerical performance of streaming GST.</i> Plots (a)-(d) compare the convergence rates of the Kalman filter's point estimate with batched MLE point estimates under a metric of mean square error (MSE) and mean absolute error (MAE) between the point estimate and the parameters of the data generating model. The x -coordinates of the gray lines correspond to batches of germs of fixed power, and the batched MLE point estimates are calculated based on the observations from all data up-to and including the current batch. These plots also compare the evolution in the filters' MSE and MAE with their expected evolution given by $\text{Tr}(P_k)$ and $\text{Tr}(\sqrt{P_k})$ respectively. MSE is the natural metric for a Kalman filter since Kalman filters minimize the square of the expected error in the estimate, but MAE is a stronger performance metric that is more sensitive to small differences in parameters. Plots (e) and (f) display error in the estimate of particular Hamiltonian parameters that correspond to the types of errors we expect could be reduced with improved calibration. The dotted lines denote the "true" parameters that were used to generate the data.	97
7.1	<i>Generic calibration loop structure.</i> Classical controls drive the operation of a quantum device. Specific probe circuits are measured and processed with an estimator to infer deviations from the target (ideal) operation of the processor. A classical feedback law updates the controls to optimize performance and minimize device errors.	99
7.2	<i>Calibration as an input-output problem.</i> An input configuration θ maps to an error state x through the device map Ξ . The configuration θ represents all classical control parameters used to drive the quantum device, e.g. pulse amplitudes, frequencies, timings, etc... The device map represents the physical realization of the computation with given configuration. The error state represents the deviations between the ideal operation and the actual implementation.	100

7.3	<i>Linearization of the input-output map.</i> A sequence of configurations are produced via control updates u . If u is small, then the new error state x_{k+1} is well approximated by a linear function of the old error state x_k , the configuration update u_k , and the control map B that depends on the device map Ξ	100
7.4	<i>Dual control estimation for Ramsey experiments with batched regression.</i> A grid of detunings was selected and each point was evaluated with Ramsey RPE experiments. The estimated phases clearly exhibit linear dynamics with detunings in this range. The slope was estimated on Q1 to be 0.1257(2) and on Q2 to be 0.1269(2). The idle gate was taken at 20 ns, so the expected slope is 40π ns ≈ 0.1257 , which exhibits good agreement with measured values.	120
7.5	<i>Dual control estimation for X-gate over-rotation experiments with batched regression.</i> A grid of amplitude offsets (in device units) was selected and each point was evaluated with X-gate RPE experiments. The estimated phases clearly exhibit linear dynamics with amplitude changes in this range. The slope was estimated on Q1 to be 4.3262(2) and on Q2 to be 5.470(9). These results indicate that qubit-drive coupling rates differ for different qubits in a superconducting device.	121
7.6	<i>Histogram of samples collected from 1-parameter RPE observations.</i> 10,000 samples were collected from a gate set with coherent idle error of 1.329×10^{-3} (chosen arbitrary) on one gate and uniform depolarization on the other gates (see main text). An RPE experiment was simulated up to a maximum depth of 2^9 with 50 shots per circuit. The distribution is well approximated by a Gaussian. The standard deviation was calculated to be 6.8×10^{-4} , which should be compared to $\pi/(2^9\sqrt{50}) \approx 8.7 \times 10^{-4}$	122
7.7	<i>Performance of LQR regulation in two regimes.</i> An LQR experiment is simulated in pyGSTi using a known control model (with no estimate errors). The coherent parameter being regulated is subject to a drift process of covariance 10^{-4} per step. An under-damped LQR control law assumes a large control cost, which is not responsive enough to regulate drift. Choosing a lower control cost yields better performance. In the limit of infinite cost on controls, one recovers free evolution under the drift process.	123
7.8	<i>Evolution of the RMSE over 1000 random trajectories with and without Kalman filtering</i>	124
7.9	<i>Dual control estimation for CZ gate phases with batched regression.</i> This data was taken on a Transmon device and verifies that a linear model holds in the parameter regime considered.	125
7.10	<i>Average performance of direct regulation.</i> 100 random trajectories were simulated with a drift covariance of 10^{-6} uniform on the three phases. The RPE experiments were taken up to max depth 2^9 with 50 shots per circuit. The gate set had uniform depolarizing noise at 10^{-3} on each gate and no other out-of-model errors. . . .	126

7.11	<i>RMSE evolution over 100 random trajectories.</i> The RMSE in each time step calculated on the data of Fig. 7.10. The specific control model used in simulation had a slightly larger ZI effect from drive amplitude variations, which explains the higher noise floor. This higher noise floor illustrates how stochastic noise can couple across multiple parameters.	126
8.1	<i>Cycles of QEC code operation.</i> Operation of a QEC code can be modelled as a sequence of discrete cycles. An initial encoding cycle prepares the logical 0 state. Syndrome extract measures stabilizers and performs the subspace projection. Decoding infers the logical effect from syndrome information. The logical operation perform transformations on the encoded logical dof. The cycles are iterated over. Sometimes, it is useful to perform multiple rounds of syndrome extraction before logical operation. In the memory paradigm, no logical operation is performed, and syndrome extraction is only interleaved with decoding.	130
8.2	<i>Four models for classical simulation of QEC codes.</i>	131
8.3	<i>Visualization of the code distance in QEC.</i> Logical states $ \bar{0}\rangle$ and $ \bar{1}\rangle$ are separated by Hamming distance 5 in this example. We see that an error of weight greater than 2 will map one logical state into the radius of the other logical state, which will cause the decoder to map the resulting state to the other logical state, implying a logical error. However, any error that is not a logical operation (i.e. maps one logical state to another) will be detectable.	133
8.4	<i>Codespace partitioning of Hilbert space.</i> An initial separable state (in the codespace) is mapped through an encoding unitary B to a state in the physical basis. In the physical basis the state is highly entangled across the register. In the codespace basis, the state is separable. The encoder can be understood as a change of basis, and it is usually a Clifford operation.	138
8.5	<i>Measurement based stabilizer encoding.</i> Measurement of the stabilizers projects onto a definite syndrome state y . Decoding the syndrome y returns a correction that maps to $\rho \otimes 0\rangle\langle 0 $. Measurement of logical-Z operators projects into a definite logical state $ z\rangle\langle z \otimes 0\rangle\langle 0 $ and logical-X operators can be applied to map to $ 0\rangle\langle 0 \otimes 0\rangle\langle 0 $	150
9.1	<i>Entropic measures in the 5-qubit code under depolarizing noise.</i> Exact codespace error distributions were calculated for varying rate of depolarization error on a grid of 100 equally spaced depolarization rates. CCE $H(Z Y)$, mutual information $I(Z;Y)$, and logical codespace entropy $H(Z)$ were calculated at each point. Additionally, the single-qubit entropy of the separable noise channel and the 5-qubit entropy of the total noise channel were calculated. The logical entropy peaks at 2 bits, corresponding to the four types of Pauli errors that can occur in a single-qubit logical state. The mutual informaiton grows rapidly then decays more slowly after peaking at some fixed depolarization rate. CCE grows like a sigmoid function before peaking at the maximum logical entropy.	163

9.2	<i>Entropic measures in the surface code under depolarizing noise.</i> This plot should be compared with Fig. 9.1. Remark on the similarities in the structure of the entropy measures.	163
9.3	<i>Entropy difference between the surface codes and 5-qubit code.</i> Plot shows the surface code rates (Fig. 9.2) minus the 5-qubit rates (Fig. 9.1). See in-text for discussion.	164
9.4	<i>Contour plot of the CCE as a function of X- or Z-bias.</i> A simple biased noise model was used to generate a 2-dimensional grid of conditional entropy estimates. The contour plot displays the characteristics of CCE in terms of the biases. Observe that for small biases in one axis, the CCE is small even for relatively large values in the other axis.	164
9.5	<i>Distribution of empirical estimates for the five-qubit code under fixed depolarization.</i> Using a fixed depolarizing noise strength of 0.11 (about the value that conditional entropy crosses single-qubit entropy), a number of different samples (in the list $[10^3, 10^4, 10^5]$) were collected from the composite noise-syndrome channel of the 5-qubit code and empirical entropy estimates were created using. This process was repeated for varying number of samples to create a histogram of the estimate distribution.	165

List of Tables

1.1	Summary of key topics and my contributions in this work.	3
3.1	Summary of the quantum noise models defined in this work	19
5.1	Single-parameter feed-forward RPE estimation.	60
5.2	Eigenstates and eigenphases for the model CZ operation, Eq. 5.22. The II phase was set to 0 using $U(1)$ phase freedom.	63
5.3	The state preparations and measurement (SPAM) pairs used in phase identification for the CZ gate. Each SPAM pair surrounds logarithmic spaced application of the CZ unitary $\mathbf{U}_{CZ}^{2^k}$. For a given state preparation there are two measurements that estimate the cosine and sine parts of a complex phase listed in the last column.	63
6.1	Linear Kalman filter algorithm.	82
6.2	Extended Kalman filter algorithm.	85
6.3	Sigma-point choices. Some popular choices of sigma-point definitions with references.	86
6.4	Kalman GST algorithm.	88
7.1	Formulas for linear quadratic regulation. See also [A J08]. Here, Q is the state cost, and R is the control cost.	110
7.2	Summary of the three algorithms presented in this chapter.	116
7.3	Phase-based linear-quadratic regularization without smoothing.	117
7.4	Phase-based regularization with Kalman filtering. When $k = 0$, the observation update step $\hat{x}_k \leftarrow \dots$ should be replaced with $\hat{x}_0 \leftarrow y_0$ as there is no previous control or estimate.	118
7.5	Phase-based regularization with on-line dual estimation. When $k = 0$, the observation update step $\hat{x}_k \leftarrow \dots$ should be replaced with $\hat{x}_0 \leftarrow y_0$ as there is no previous control or estimate.	119

Acknowledgments

This work was made possible by the mentorship and guidance of Kevin Young and K. Birgitta Whaley. Kevin’s unfailing support and Birgitta’s keen wisdom made my PhD experience one of joy and wonder. I thank the other members of my committee who also volunteered their time to serve as qualifying examiners. I also thank Murphy Y. Niu for helping me to grow as a scientist. Next, I owe much gratitude to the people in the Quantum Performance Laboratory at Sandia National Labs. In particular, I thank Corey Ostrove, Kenny Rudinger, Stefan Seritan, Timothy Proctor, and Robin Blume-Kohout for guidance, technical support, and many fond memories. I also am indebted to other graduate students in the Whaley group, including Nam Nguyen, Jordan Hines, Tyler Kharazi, and Oskar Leimkuhler. I also thank Samyak Surti for insightful research discussions. I would not have made it to this moment without the mentorship and wisdom of A. Ravi P. Rau who helped me discover the joy of physics. In graduate school, I benefited from and am thankful for the teaching of Robert Littlejohn, Thomas Courtade, and Venkat Anantharam. Other particularly impactful professors in my undergraduate were Shane Stadler, Gabriela González, and Kenneth Schafer. A long list of teachers helped me develop in my younger years, and I am particularly thankful to Tina Fink, LindaGail Schumaker and Claudia Brunot. Finally, I am thankful for the unfailing support of my family, my parents Greg and Nina, my aunts Jenny and Dawn, and my siblings Holden, Quinn, and Mazie.

Disclosure on the use of AI: large language models (LLMs), including ChatGPT, Gemini, and Github Co-Pilot, were used to develop code products related to this work, to write Python code to graph data in Matplotlib plots, and to accomplish a few tedious L^AT_EX tasks such as generating TikZ code for quantum circuit diagrams. At moments during my brainstorming process, I consulted LLMs to review existing theory. However, the core perspective of this work represents my original, independent work. No English text in this work was generated artificially, with the exception of the references.

Part I

Introduction

Chapter 1

Context of the work

This thesis was written in 2024-2025 and the research was developed over the 4 years preceding. The decade of the 2020s has been an exciting time for the field of quantum computing. We saw the first demonstrations of nonclassical computation [Aru+19], of quantum error correction [Rei+24; Goo25], and small algorithmic performance. An optimist might think that the first clear demonstrations of useful quantum computation are on the horizon. My thesis is designed to provide techniques to support that monumental scientific step and to ensure stable systems in the age of quantum utility. My work particularly addresses the two levels of quantum noise characterization and quantum error correction in the hierarchy of quantum engineering theory shown in Fig. 1.1. This chapter provides the context for my work by summarizing some models of quantum computing, prominent architectures, and the state of quantum algorithms. A summary of the contributions of each chapter may be found in Tab. 1.1

1.1 Quantum information processing devices

The field of quantum information processing began germination in the later half of the 1900's. A good historical starting date could be Feynman's lecture *There's Plenty of Room at the Bottom* [Fey60], and it would be accurate to attribute Feynman to having been one of the first to recognize the possibility of quantum computation. Of course, scientists are but the product of their environment, and the integration of quantum computing in the fabric of modern science follows in a centuries old tradition, all the way back to Democritus [Aar13] and before. A quantum information processing device will only be one step along the natural course of scientific progress.

In 2024, the field of quantum computing has primarily focused on one particular model of quantum computation: the gate-based model of computation, described in the next section. However, gate-based quantum computation is by no means the only viable model of quantum information processing, and less-but-still significant efforts have considered alternatives, including measurement-based and adiabatic computation, summarized in the following

Chapter	Contribution
Chapter 2	I recall a number of useful mathematical facts and properties that are exploited in later chapters. This is introductory material.
Chapter 3	I define and discuss various models of quantum noise. Little of this is novel, though the synthesis is done at a high level of abstraction and in a novel way. These models are applied throughout the rest of this thesis.
Chapter 4	I review gate set tomography theory, based on some of my contributions to the review article [Has+24]. Little of this is my work. The only new topic introduced in this chapter is on invariant frames for gate sets. The ideas and techniques discussed in this chapter are foundational for the rest of the work.
Chapter 5	I review existing and introduce new ancilla-free approaches to quantum phase estimation for processor calibration. I discuss joint work [Rud+25] that applied phase estimation for multi-qubit gate calibration. I derive a two-qutrit phase estimation protocol suitable for gate calibration. I derive two new group-theoretic approaches for multi-phase decoupling.
Chapter 6	I derive a Kalman filter estimator for gate set tomography based on my publication [MY23]. I synthesize many developed ideas in classical control theory and apply them to quantum processor characterization. I review existing theory.
Chapter 7	I derive a drift regulation protocol for quantum processors based on efficient techniques in classical control theory. I present numeric studies and experimental results. I discuss implementation considerations. I synthesize a number of classical ideas in control theory to novel application in quantum processor control.
Chapter 8	I review foundational QEC theory to support the derivations of §9. I introduce a notion of entropy rates in the codespace of QEC codes. I discuss the decoding problem. I derive channel representations of quantum codes. I briefly mention the notion of fault tolerance.
Chapter 9	I prove a theorem that upper bounds the best achievable error rate of a stabilizer code (in a code capacity model) under a given noise distribution and <i>any</i> decoder. I present numeric studies, discuss application, compare to existing theory, and outline future prospects.

Table 1.1: Summary of key topics and my contributions in this work.

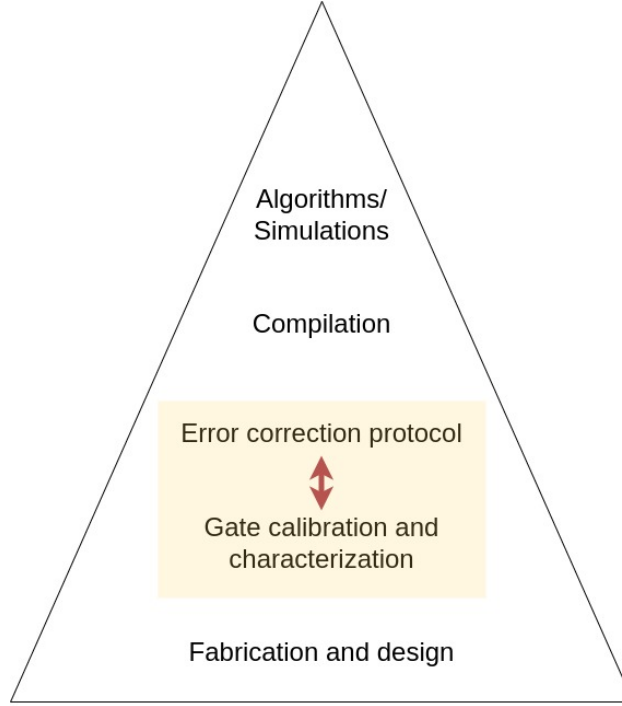


Figure 1.1: *Hierarchy of quantum computer engineering problems.* My work in this thesis concerns the problems in processor characterization, error correction, and at their intersection in code co-design.

paragraphs. It is usually the case that a particular model of computation is linked to a particular plan for physical implementation. Nevertheless, if one assumes a *universal* model of quantum computation, then a particular representation of an algorithm in one model of quantum computation can be mapped to another. From this perspective, the job of the quantum engineer is to develop both a useful physical model of quantum computation and a practical abstract model that can be used by the quantum algorithm designer. The network model [Deu89], which is essentially the same as the gate-based model, seems most suitable for general algorithm design due to its intuitive graphical representation.

Measurement-based quantum computing [Bri+09] is a universal alternative to gate-based computation and is theoretically interesting because it illustrates some of the essential nature of entanglement. A measurement-based computation starts with some initial, highly entangled resource state and performs a series of measurements that actually implement a desired algorithm. The results of the algorithm are obtained by processing the outcomes of the measurements. The measurements may be made in an entangled basis, or remarkably, the measurements can be unentangled single-qubit measurements, as in one-way quantum computation [RB01]. The one-way quantum computer provides a useful theoretical model for understanding how entanglement is used as a resource in quantum information processing.

Measurement-based quantum computing is a primary physical model for photonic quantum information processing [SP19] and has also been proposed for atomic lattices [Bri+09].

Adiabatic quantum computing [AL16] is another example of an alternative universal model of quantum computation that is theoretically interesting because it highlights the role of infinitesimal quantum dynamics in useful algorithms. The adiabatic quantum computer begins with the definition of two Hamiltonians: an initial Hamiltonian H_0 whose ground state is easy to prepare and a final Hamiltonian H_f whose ground state encodes the solution to a computational problem. The adiabatic theorem of quantum mechanics [Tos13] states that if the system is prepared in the ground state of H_0 and then subjected to a time-dependent Hamiltonian $H(t) = (1 - t)H_0 + tH_f$ that *smoothly*¹ interpolates between the initial and final Hamiltonians at $t = 0$ and $t = 1$, then the instantaneous state at all times during the transition will remain an eigen-state of $H(t)$. The adiabatic theorem is used to ensure that at $t = 1$ one will have prepared the ground state of the difficult Hamiltonian H_f and thus can solve the computational problem by making measurements of the final state. Adiabatic quantum computation is thought to be a useful model for optimization problems, especially with the quantum annealing algorithm [DC05] and has a unique proposed physical implementation [Shi+14].

1.2 Gate-based quantum computation

The most prominent physical model of quantum computation at the time of this thesis is the gate-based model, which is essentially² equivalent to a “circuit-based” model. In the gate-based model, quantum algorithms are encoded into a sequence of gates that implement discrete sub-components of the computation. The gate-based model is conveniently described in terms of a circuit diagrams, which are intuitive graphic representations analogous to classical logical diagrams [Deu89]. Fig. 1.2 shows an example of a quantum circuit diagram.

In the gate-based model of quantum computation, it is usually assumed that the computation begins in the all 0’s computational basis state, which is usually defined to be the ground state of the device. Furthermore, it is usually assumed that the measurements are made in the computational basis, which defines the z -axis of the Pauli algebra. The choice of computational basis is made based on the physics of a device and should be selected to be whatever makes device operation easiest. State preparations in other bases can be made by beginning in the all 0’s state and performing a sequence of gates to rotate into another basis. Similarly, measurements can be made in other bases by preceding a measurement in the computational basis with a dual³ sequence that rotates into the desired measurement basis.

¹The notion of smoothness is related to the speed of the transition and can be quantified by the eigen-spectra of the initial and final Hamiltonians.

²It could be said that a circuit-based model is a gate-based model that is written entirely in terms of the native operations, i.e., a transpiled gate-based circuit.

³Here, dual denotes the fact that a measurement fiducial sequence should be properly understood as the dual (Hermitian conjugate) of the sequence that prepares the state that is measured by the corresponding

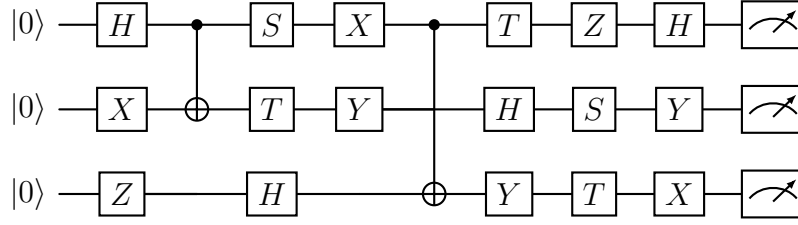


Figure 1.2: *An example of a random quantum circuit.* Single-qubit operations are represented as boxes that appear on the wire corresponding to the qubit and labeled according to their identity. Two-qubit operations appear like vertical links that connect the wires of two qubits. The temporal execution of the circuit proceeds from left to right. In this example, we start with initial state preparations in the all 0's state, denote by the kets $|0\rangle$ on the left, and we measure in the computational basis, denoted by the meters on the right.

A gate-based model of quantum computation is universal if it is composed of a set of gates that can efficiently approximate any unitary [Bar+95]. There are a variety of choices for a universal quantum gate set. From the perspective of algorithm design, it is often useful to work with a set of gates that are different than the actual native gate set used to implement the computation. Certain gate sets have different properties that may make it easier to identify algorithmic utility, while other gate sets better reflect the native physics of the device. The problem of quantum compilation is then to transpile a high-level algorithm into a native gate set. Then, the transpiled native circuit definition is finally compiled into classical machine code that drives the waveforms of the quantum computation and measures the results.

1.3 Prominent architectures

At the time of publication of this thesis, there are two leading platforms for quantum computation: superconducting circuits and ion trap devices. Other promising platforms are also emerging, including photonic, neutral atom, and spin-based silicon devices. These platforms are each promising on their own, and much could be said. For my purposes, it is sufficient to observe that all the promising platforms, excluding the measurement based photonic quantum computer, are readily described with the gate-based quantum computer model. Other more exotic hardware, such as adiabatic devices, seem more useful for particular problems rather than as universal quantum information processors and are not the focus of my work.

outcome in the final computational basis measurement.

Superconducting architecture

Superconducting circuits are one of the most well studied platforms proposed for quantum computing. At the time of this thesis, the most common superconducting circuit architecture is the Transmon device [Koc+07]. A newer alternative modality is the Fluxonium device [Bao+21], which seems to offer better fidelities at the cost longer gate speeds. A very interesting new proposed platform is the Kerr-Cat qubit [Gri+19], which is a noise-biased qubit that has an exponential suppression of bit-flip noise at the cost of a linear increase in phase-flip noise⁴. Each of these platforms could be discussed in great detail, and there are many other possible superconducting architectures, including other exotic sounding words. One of the remarkable features of quantum circuit electrodynamics is that it can be used to define almost limitless possible superconducting architectures. Again here, I emphasize that all the superconducting circuit architectures thus proposed are well described by a gate-based model of quantum computation, so the techniques in this thesis should be readily applicable. Indeed, superconducting Transmon circuits are used in all the experimental demonstrations of this work.

One of the biggest advantages of superconducting circuits is their low gate time. Transmon devices offer single-qubit gate times of about 20 ns and two-qubit gate times of around 200-800 ns. These numbers can be compared with the speed of the best classical processors today, which are in the range of 2 GHz or 0.5 ns per operation. Superconducting circuits are also appealing due to the fact that they can be controlled with RF drives, which are well established technology.

Despite their appeal, superconducting circuits also have significant drawbacks. Superconducting circuits are mesoscopic quantum objects; what represents a single qubit is in fact a collection of many atoms with a radius somewhere around 0.1 mm. Whether or not qubits this large can be used to fabricate a sufficiently large device for useful quantum computation is somewhat unclear. However, there is a more interesting and fundamental reason as to why larger qubits may be worse: mesoscopic quantum systems are extraordinarily sensitive to fabrication defects. A few misplaced atoms in the fabrication of a superconducting circuit can parasitically couple to the coherent quantum state of the mesoscopic object and induce decoherence. Quantum defects, named “two-level systems” or TLS in the literature, can be detected in high-precision characterization, and much research has been done into understanding the nature of defects in quantum condensed matter systems, with results proving rather difficult to obtain. TLS remain one of the most fundamental obstacles to superconducting quantum processes.

⁴The reason people think this is a good trade-off is that phase-flip noise can be mitigated with well developed dynamical decoupling techniques and devices that are dominantly bit-flip noise can be more readily corrected with efficient quantum repetition codes. More generally, biased noise sometimes offer better QEC thresholds, see Part III of this thesis.

Ion-trap architecture

At the time of this thesis, the other leading contender for a large-scale quantum computing platform is the ion trap quantum computer. In this paradigm, ions are trapped on fabricated substrate and quantum information processing is performed on the energy levels of the ions. There are a number of different ion species being investigated, and popular platforms are based on ion species such as Calcium, Barium, Beryllium, and Ytterbium. The physical quantum information processing is done in two prominent approaches: hyperfine qubit, which exploits two almost-degenerate states in the ground state, and the optical qubit, which exploits a state in the ground state and a metastable state.

One of the biggest appeals of ion traps is their extremely high gate fidelities. At the time of this thesis, the leading single-qubit gate infidelity is a remarkable 10^{-7} [Smi+24] and the leading two-qubit gate infidelity is around $10^{-3} - 10^{-4}$. These numbers are multiple orders of magnitude lower than the infidelities of superconducting qubits (about 10^{-3} for single-qubit gates and approaching 10^{-2} for two-qubit gates). Whereas superconducting qubits are mesoscopic quantum objects that can be readily coupled with their environment, ions are microscopic quantum systems that seem to exhibit less environment coupling.

However, ion traps also come with associated difficulties. One of the primary downsides of ion trap quantum computers is their comparably lower gate speed. A gate in an ion-trap can take times in the order of tens or hundreds of microseconds. This is orders of magnitude larger than superconducting circuit gate times. However, it should be said that at the time of this thesis, it is unclear that the long gate time outweighs the high fidelities of ion trap quantum computers. Indeed, it would seem that the proper performance metric is the ratio between the gate time and error rate, which is the relevant metric for quantum error correction requirements.

1.4 Potential use cases of quantum computers

At the time of the thesis, there are many open questions about the first potential use cases of quantum computers for useful computation. Despite rapid progress in hardware development, identification of near-term algorithms on these devices have proven difficult. Long-term prospects remain very promising, from implementation of Shor’s factoring algorithm and cryptographic applications to quantum chemistry and material science applications. Quantum machine learning is also a promising newly emerging field that seeks to use quantum computers to speed up or reduce the energy consumption of current machine learning techniques. As was the case with classical computers, it is my opinion that the use cases will become more clear once the hardware has been developed.

Chapter 2

Mathematical foundations

2.1 Compositions of Hilbert spaces

In this thesis I will invoke the concept of a Hilbert space of a quantum register [Wat18], which is a fundamental notion in quantum information science. Essentially, the Hilbert space of a register is the space in which all the possible states live. Formally, a Hilbert space $\mathcal{H}$ of an n -qubit quantum register¹ is a complex vector space equipped with an inner product that is also a complete metric space [Rud86].

The direct sum, written as $\oplus$, operates on a pair of Hilbert spaces and produces a new Hilbert space $\mathcal{H}_1 \oplus \mathcal{H}_2$ of dimension $2^{n_1} + 2^{n_2}$ that consists of all pairs of states in $\mathcal{H}_1$ and $\mathcal{H}_2$. One can understand the direct sum as a “stacking” operation that concatenates two vectors. The inner product structure of $\mathcal{H}_1 \oplus \mathcal{H}_2$ is defined as the sum of the inner product in the component Hilbert spaces, as

$$\langle (\psi_1, \psi_2), (\phi_1, \phi_2) \rangle_{\mathcal{H}_1 \oplus \mathcal{H}_2} = \langle \psi_1, \phi_1 \rangle_{\mathcal{H}_1} + \langle \psi_2, \phi_2 \rangle_{\mathcal{H}_2}, \quad (2.1)$$

where $\psi_1, \phi_1 \in \mathcal{H}_1$ and $\psi_2, \phi_2 \in \mathcal{H}_2$. More generally, one can write a k -fold direct sum of Hilbert spaces $\bigoplus_{j=1}^k \mathcal{H}_j$ that consists of all possible k -tuples of states in each component space. The inner product of $\bigoplus_{j=1}^k \mathcal{H}_j$ is the sum of the inner products in the component spaces.

The tensor product, written as $\otimes$, operates on a pair of Hilbert spaces and produces a new Hilbert space $\mathcal{H}_1 \otimes \mathcal{H}_2$ of dimension $2^{n_1+n_2}$ with the defining inner product

$$\langle \psi_1 \otimes \psi_2, \phi_1 \otimes \phi_2 \rangle_{\mathcal{H}_1 \otimes \mathcal{H}_2} = \langle \psi_1, \phi_1 \rangle_{\mathcal{H}_1} \langle \psi_2, \phi_2 \rangle_{\mathcal{H}_2}, \quad (2.2)$$

More generally, one can define k -fold direct products with the inner product structure being the product of the component spaces.

¹Hilbert spaces can be defined either for pure- or mixed-states, and I will not spend time developing the mathematical subtleties of the difference – suffice to say that a pure-state Hilbert space contains state vectors and a mixed-state space contains vectorized density matrices.

In quantum mechanics, tensor products of Hilbert spaces are the fundamental operation used to represent combinations of different systems or particles. In quantum information, the tensor product describes the computational space associated with a collection of qubits. The direct sum corresponds to the combined state of two subspaces *that do not interact*; the Hilbert space of non-interacting particles corresponds to the direct sum of the composite systems.

2.2 Lie groups

Lie groups [Hal16; Hay16; Bin12] provide a basic mathematical mechanism to represent the unitary operations that a quantum computer is designed to implement, and Lie groups can also be used in analysis of general quantum channels – with some caveats. To understand the concept of a Lie group, start with the definition of a group: a **group** is a collection of elements with a binary product that has following properties: associativity (i.e., $(ab)c = a(bc)$), existence of the identity (there exists an element $\mathbb{1}$ such that $g\mathbb{1} = g$ for all elements g), and existence of the inverse (for each g , there exists g^{-1} such that $gg^{-1} = \mathbb{1}$). A *Lie group* is a distinguished group that is also a *differentiable manifold*. To understand that, start with the concept of a manifold; a **n -dimensional manifold** is a continuous group whose elements are defined by a parametrization with n numbers that vary continuously. A finite **differentiable manifold** is a n -dimensional manifold that is also differentiable. The definition of differentiable can be stated with great precision, but basically it is the property that ensures that a tangent line exists at every point on the manifold. You can imagine some amorphous blob in n -dimensional space; the surface of the blob is the manifold, and differentiability ensures that the surface does not have discontinuities or sharp spikes where tangent lines cannot be uniquely defined.

While Lie groups are defined very generally, our interest in quantum information processing means that we exclusively consider *matrix Lie groups*. To define that, first begin with the most general group of matrices; the **general linear group** over a field $\mathbb{K}$ with n elements, denoted $\text{GL}(n; \mathbb{K})$, is the collection of all $n \times n$ invertible matrices with entries in $\mathbb{K}$. From its definition, $\text{GL}(n; \mathbb{K})$ is a group. In our case, the field $\mathbb{K}$ will be exclusively either the reals $\mathbb{R}$ or the complex numbers $\mathbb{C}$. A **matrix Lie group** is then any closed² subgroup of $\text{GL}(n, \mathbb{K})$. The closure property essentially ensures that there are no “holes” or ill-defined points in the group, analogous to the concept of differentiability.

Lie groups represent compositions of quantum operations in a gate-based model of quantum computing. Imagine that each distinct gate in the computer is labeled with an identifier ξ . The actual physical operations result from compiling and performing the waveform associated to ξ , and each physical operation can be associated with a mathematical representation

²*closure* is a highly technical definition; a matrix group G is **closed** if every convergent sequence of matrices $(A_m)_m$ in G has the property that $A \in G$ or A is not invertible.

G_ξ . Now, it should be the case that the collection G_ξ with the binary product of matrix multiplication forms a representation of a Lie group³.

Vector spaces associated with Lie groups

The elements of a Lie group are matrices contained in the general linear group. These matrices are, in fact, themselves operators that act on a vector space. In our case, this vector space is the state space of a quantum computer, while the group elements are the gates. Similarly, one can define the dual vector space; this the space of vectors that act with left-multiplication on elements of the Lie group. The dual vector space of a Lie group should be associated with measurement.

2.3 Lie algebras

By taking special care to define our operations as elements of a matrix Lie group, we are immediately able to exploit an extremely useful and intuitive framework for dealing with infinitesimal transformations on the group; this is the *Lie algebra* associated with the Lie group. Being able to deal with infinitesimal transformations with such precise a language as Lie's gives us the ability to make strong statements about high-precision tomography and derive useful metrology algorithms for parameter estimation.

Let us start with an informal introduction to the concept of a Lie algebra. Recall that an element of an n -dimensional Lie group is explicitly parameterized by an n -dimensional vector x . Now, write a group element as $G(x)$. If we agree that $x = 0$ is the identity operation, then any element of the group that is sufficiently close to the identity can be expressed as

$$G(x) = \exp(ix_a \mathbf{X}_a), \quad (2.3)$$

where Einstein summation notation⁴ is employed. Here, $\mathbf{X}_a$ is a matrix, and I bold it to denote the fact that it is contained in the *Lie algebra*. x_a is simply the a th component of the vector x .

In quantum mechanics, the conventional sign notation is that ordinary time evolution proceeds with a negative in the exponential, as

$$U(t) = \exp(-i\mathbf{H}t) \quad (2.4)$$

³The requirement that a gate set be a representation of a Lie group is one of convenience. We can define gate sets that do not represent Lie groups. An example is a rotation by any real number z such that z/π is irrational. If the gate can only be performed an integer number of times, then nz will never be an integer multiple of 2π and hence there will be no inverse in the gate set (there will be generally be approximate inverses, which might be sufficient). However, if our gate set does form a representation of a Lie group, then it can be understood with a useful and generic theory.

⁴In this notation, repeated indices are summed, i.e., $x_a X_a$ is equivalent to $\sum_a x_a X_a$.

where $\mathbf{H}$ is an Hermitian Hamiltonian that generates unitary dynamics. On the other hand, mathematicians often employ notation

$$U(t) = \exp(\mathbf{G}t) \quad (2.5)$$

where $\mathbf{G}$ is an anti-Hermitian generator of the dynamics.

Lie algebra short hand

Throughout this work, I employ a very useful short hand notation to define elements in a Lie group under the exponential map. Here, I write e^x to denote the exponential of a vector x via the Lie group mapping

$$e^x \equiv \exp(-i\mathbf{H}_a x_a) \quad (2.6)$$

where Einstein summation is employed. This should be understood as the exponential of a x whose elements contain all the information about the resulting group element. Given x , the group element is fixed by a choice of basis $\mathbf{H}_a$.

Adjoint representations

In discussing Lie groups and Lie algebras, it is immediately useful to define adjoint representations. The adjoint representation provides an equivalent representation of the group that often appears in analysis in quantum information. There are two adjoint representations, a “little” adjoint associated with the algebra and a “big” adjoint associated with the group.

The algebra adjoint of X acting on Y is defined as the commutator between X and Y

$$\text{ad}_X(Y) \equiv [X, Y]. \quad (2.7)$$

While it may appear at first glance that the adjoint is just another way to write the commutator, it becomes useful when evaluated explicitly as a matrix in a basis

$$[\text{ad}_X]_{i,j} = \langle Y_i, [X, Y_j] \rangle. \quad (2.8)$$

Here the matrix representation of the adjoint describes how commutation acts on an element decomposed in a basis and is extensively used in representation theory.

The group adjoint is defined

$$\text{Ad}_X(Y) = XYX^{-1} \quad (2.9)$$

and can also be developed in an explicit matrix representation. The group adjoint appears extensively in this work.

2.4 Important discrete group

The two most important discrete groups in quantum information theory are the Pauli and Clifford groups, which I review here.

Pauli Group

The Pauli group is of fundamental importance in quantum information. Pauli operators define a basis for the space of unitary operators on n -qubits. In the case of one qubit, the Pauli group is generated by three-plus-one elements: the X, Y, and Z type Paulis:

$$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \mathbf{Y} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}; \mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (2.10)$$

and the identity operation

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.11)$$

The entire **1-qubit Pauli group** $\mathcal{P}$ is composed of all unique products of its generators that is the elements of $\{1, -1, i, -i\} \times \{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$. The **projective 1-qubit Pauli group** $\mathcal{P}^*$ is the full group modulo the phases, i.e., $\{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$. n -qubit Pauli groups $\mathcal{P}^n$ are formed from tensor products of the 1-qubit Pauli group.

Clifford groups

The Clifford group is another important discrete group that appears extensively in quantum information theory. The Clifford group is defined as the normalizer of the Pauli group. That is to say, the Clifford group is the set

$$\{C : CPC^{-1} \in \mathcal{P}^n \quad \forall P \in \mathcal{P}^n\}, \quad (2.12)$$

ie. Clifford conjugation maps a Pauli operator to a Pauli operator. The Clifford group is widely used in randomized benchmarking techniques, as discussed below.

The Clifford group is also useful because it permits efficient classical simulation of quantum computers [Got98]. In this formalism, one observes that a convenient representation of quantum information can be developed in terms of *stabilizers*⁵ of quantum states. The dynamics of the circuit can be described by tableaus that record how stabilizers transform under circuit operation. If all operations in a circuit are Clifford operations, then an initial set of Pauli stabilizers will be mapped to new Pauli stabilizers, ie. the stabilizers will be Pauli operators at all points in the circuit. This permits efficient classical simulation. The Clifford tableau framework or CHP formalism is widely used across quantum information and particularly in quantum error correction.

2.5 Technical elements from classical information theory

I now collect some technical elements from classical information theory that will be useful in the proof of the codespace entropy theorem and its applications.

⁵Stabilizers are operators that leave quantum states invariant

Classical channel A classical channel is a stochastic matrix⁶ W on input set X and output set Y :

$$W[y, x] = \Pr(X = x | Y = y). \quad (2.13)$$

$W[y, x]$ is the probability of receiving y having sent x . Generally, we should think of X as the register that computation is done on and Y as the register after having been subjected to an error.

A **code** for channel $W : X \rightarrow Y$ is a pair of mappings (f, φ) that encodes a vector space and attempts to recover the vector space after having been subjected to the (classical) noise channel. The encoding map f maps elements in the vector space $\mathcal{V}$ to elements in the channel input set

$$f : \mathcal{V} \rightarrow X \quad (2.14)$$

The **decoding map** φ maps elements in the channel output set back to elements in the vector space

$$\varphi : Y \rightarrow \mathcal{V}'. \quad (2.15)$$

In general, we could assume $\mathcal{V} \neq \mathcal{V}'$, but here we will always assume they are the same $\mathcal{V} = \mathcal{V}'$.

A code may be interpreted as a classical channel

$$T(v'|v) \equiv \Pr(\varphi^{-1}(v') | f(v)) = W[f(v), \varphi^{-1}(v')], \quad (2.16)$$

i.e., $T(v'|v)$ is the probability of receiving v' having sent v . In the theory of error correcting code design, we are interested in maximizing $T(v|v)$ and minimizing $T(v' \neq v|v)$.

The **probability of erroneous transmission** of message v is

$$\epsilon(v) = 1 - T(v|v). \quad (2.17)$$

The **maximum probability of error** is

$$\epsilon \equiv \max_{v \in \mathcal{V}} \epsilon(v). \quad (2.18)$$

If we assume a probability distribution on vectors v , then the **expected probability of error** is

$$\mathbb{E}[\epsilon] = \sum_{v \in \mathcal{V}} \Pr(v) \epsilon(v). \quad (2.19)$$

Typical sets . Consider a random variable $X \in \mathcal{X}$ with distribution $\Pr(X)$. It may be argued that in the limit of large n the probability of a particular sequence $x^{(n)} = (x_1, x_2, \dots, x_n)$ of iid. samples from X converges to

$$\Pr(X_1 = x_1) \Pr(X_2 = x_2) \dots \Pr(X_n = x_n) \rightarrow 2^{-nH(X)} \quad (2.20)$$

⁶A **stochastic matrix** is matrix with rows that are probability distributions.

where ⁷

$$H(X) = - \sum_x \Pr(X = x) \log \Pr(X = x). \quad (2.21)$$

Through standard arguments [TT12], it follows that there exists a typical set $A_\epsilon^{(n)}$ of sequences $x^{(n)}$ such that:

1. If $x^{(n)} \in A_\epsilon^{(n)}$ then $\Pr(x^{(n)}) \approx 2^{-nH(X)}$,
2. $\Pr(x^{(n)} \in A_\epsilon^{(n)}) \geq 1 - \epsilon$ for n sufficiently large
3. $|A_\epsilon^{(n)}| \approx 2^{nH(X)}$.

The remarkable observation of typical sets is that sampling many times from a probability distribution results in a set of typical sequences that are generally *polynomial* in size, while the set of all possible outputs is *combinatorial* in size. Hence, in the limit of a large number of samples n , only a vanishingly small fraction of events of all possible events actually occur.

Jointly typical sets . The notion of a jointly typical set generalizes the notion of a typical set to a pair of random variables. Consider a sequence of pairings

$$z^{(n)} \times y^{(n)} = (z_1, y_1), (z_2, y_2), \dots, (z_n, y_n) \quad (2.22)$$

generated iid. according to a joint probability distribution $P(Z, Y)$. The joint sequence is $z^{(n)} \times y^{(n)}$ is jointly typical if the subsequences $z^{(n)}, y^{(n)}$ are typical according to marginals and the joint sequence is typical according to the joint distribution. In this way, the jointly typical set $B_\epsilon^{(n)}$ is defined

$$B_\epsilon^{(n)} = \left\{ ((z_1, y_1), (z_2, y_2), \dots, (z_m, y_m)) : \left| -\frac{1}{n} \log \Pr(z^{(n)}) - H(Z) \right| \leq \epsilon, \right. \quad (2.23)$$

$$\left. \left| -\frac{1}{n} \log \Pr(y^{(n)}) - H(Y) \right| \leq \epsilon, \left| -\frac{1}{n} \log \Pr(z^{(n)}, y^{(n)}) - H(Z, Y) \right| \leq \epsilon \right\}.$$

It is useful to observe that the jointly typical set is a subset of all possible sets and is also a subset of the product set of the typical marginals sets

$$B_\epsilon^{(n)} \subseteq A_{Z, \epsilon}^{(n)} \times A_{Y, \epsilon}^{(n)} \subseteq \mathcal{Z}^n \times \mathcal{Y}^n. \quad (2.24)$$

which yields a nice geometric picture Fig. 2.1.

⁷Here, and throughout this thesis, I leave the base of the logarithm unspecified, as is typical in information theory. All logarithms should be assumed to be base 2, though one can work in a different base and change the base of the exponents, eg. $2^{H(X)}$.

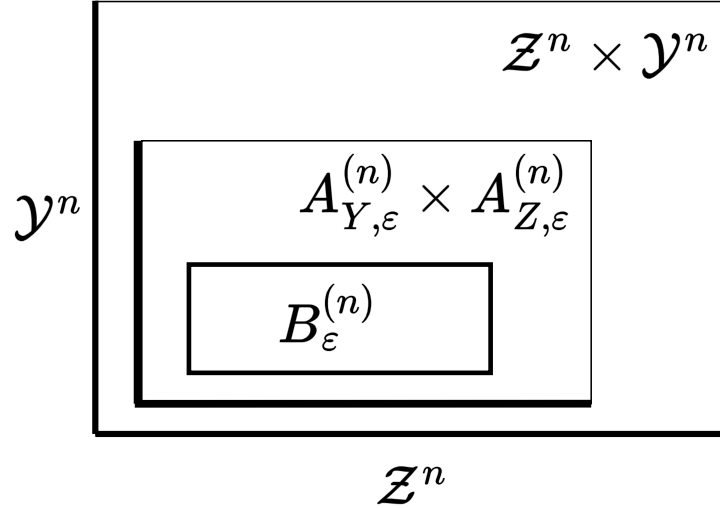


Figure 2.1: *Typical sets in the space of all sets.* The product set $\mathcal{Z}^n \times \mathcal{Y}^n$ defines the space of all possible pairings. The marginally typical sets $A_{Z,\epsilon}^{(n)}$ and $A_{Y,\epsilon}^{(n)}$ contain the sequences $z^{(n)}$ and $y^{(n)}$ that typically occur with non-vanishing probability, and the jointly typical set $B_\epsilon^{(n)}$ contains the pairings that typically occur. Observe the nested structure of the sets.

Asymptotic equipartition (AEP). The notion of asymptotic equipartition (AEP) is the observation that all elements in a typical set occur with approximately equal probability in the large n limit. This is already clear from the definition: there are $2^{nH(X)}$ elements in the typical set, each with probability $2^{-nH(X)}$. AEP is extremely useful when reasoning about typical sets and behavior in the large n limit.

Jointly typical decoding.

In Shannon's theorem, a key proof technique is the jointly typical decoder, which is also used in the proof of the CCE theorem in this work (§9.2). Given $y^{(n)}$, the decoding problem is to select $\hat{z}^{(n)}(y^{(n)})$ that is equal to the true $z^{(n)}$. The jointly typical decoder assumes knowledge of $P(Z, Y)$ and selects a sequence $\hat{z}^{(n)}$ if

- The input is marginally typical; $y^{(n)} \in A_{Y,\epsilon}^{(n)}$
- The input sequence is jointly typical with a unique output; $w^{(n)} \times y^{(n)} \notin B_\epsilon^{(n)} \quad \forall w^{(n)} \neq \hat{z}^{(n)}$

$$\hat{z}^{(n)}$$

If the observed sequence $y^{(n)}$ satisfies these two criteria then the correction is successful. Otherwise, the decoder reports a failure. It is interesting to observe that the jointly typical decoder always either succeeds or self-reports a failure. In practice, one would never use a jointly typical decoder due to efficiency constraints, but it is an extremely useful theoretical tool for understanding large n behavior.

Conditionally typical sets. Given a marginally typical sequence $y^{(n)}$ generated from $\Pr(Z, Y)$, the conditionally typical set may be defined [Wil13] as the set of all sequences $z^{(n)}$ that typically occur with $y^{(n)}$. Based on the observation⁸

$$\Pr(z_1|y_1) \Pr(z_1|y_1) \dots \Pr(z_1|y_1) \rightarrow 2^{-nH(Z|Y)} \quad (2.30)$$

define the set

$$C_\varepsilon^{Z^n|y^{(n)}} = \left\{ z^{(n)} : \left| -\frac{1}{n} \log \Pr(z^{(n)}|y^{(n)}) - H(Z|Y) \right| \leq \varepsilon \right\} \quad (2.31)$$

to be the conditionally typical set. The conditionally typical set contains all the sequences that typically occur with a given sequence. It is remarkable to note that the size of the conditionally typical set does not depend on the particular sequence $y^{(n)}$ upon which $z^{(n)}$ is conditioned. The size of the conditionally typical set may be interpreted as a sort of volume in the space of all output sequences, and it turns out that for sufficiently large n , each typical input sequence $y^{(n)}$ is jointly typical with approximately the same number of output sequences $z^{(n)}$.

⁸A proof of Eq. 2.30 may be given as follows: starting from

$$P(x_1, x_2, \dots, x_n | y_1, y_2, \dots, y_n) = \frac{P(x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n)}{P(y_1, y_2, \dots, y_n)} \quad (2.25)$$

invoke separability (that the samples are iid.) to write

$$P(x_1|y_1)P(x_2|y_2)\dots P(x_n|y_n) = \frac{P(x_1, y_1)P(x_2, y_2)\dots P(x_n, y_n)}{P(y_1)P(y_2)\dots P(y_n)}. \quad (2.26)$$

Then invoke convergence to typical and jointly typical sequences to write

$$P(x_1|y_1)P(x_2|y_2)\dots P(x_n|y_n) \rightarrow 2^{-nH(X,Y)} 2^{nH(Y)}. \quad (2.27)$$

Finally, using

$$H(X|Y) = H(X, Y) - H(Y) \quad (2.28)$$

one finds

$$P(x_1|y_1)P(x_2|y_2)\dots P(x_n|y_n) \rightarrow 2^{-nH(X|Y)}. \quad (2.29)$$

Part II

Characterization

Chapter 3

Quantum noise models

The core role of characterization in quantum computer engineering is to identify a particular noise model of a quantum information processor. That noise model provides some approximate¹ representation of the device’s true operation. Noise models can be constructed in increasing levels of generality, from simple coherent error models, to quantum channels, to quantum process tensors [Pol+18; Whi+21]. A particularly useful noise model for quantum error correction is the Pauli noise channel. I start my discussion by reviewing the generic model of quantum dynamics in a gate-based quantum processor in the language of Lie group theory. Then, I define coherent (unitary) models, incoherent (CPTP matrices) models, and finally Pauli channel models. A summary of the noise models developed in this chapter is provided in Tab. 3.1.

3.1 Quantum information dynamics

The standard gate-based model of quantum computation is discussed at length in numerous introductory textbooks [NC02; Aar13]. In the model, one has three essential components:

¹In fact, any useful noise model must be an approximate representation. While one can strive to provide a most general representation of the physical quantum dynamics, such a representation would be of no practical use because it would be impossible to efficiently do the required classical post-processing to infer the model parameters from experiment.

Noise model	Dynamical equation
Coherent error	$\rho \mapsto U\rho U^\dagger$
Quantum channel	$\rho \mapsto \sum_j K_j \rho K_j^\dagger$
Pauli noise	$\rho \mapsto \sum_j p_j \mathbf{P}_j \rho \mathbf{P}_j$
Uniform depolarizing noise	$\rho \mapsto (1 - p)\rho + p \frac{\mathbb{1}}{d}$

Table 3.1: Summary of the quantum noise models defined in this work

the states, the measurements, and the operations. Actually, this all fits very nicely in terms of Lie group theory: the states are vectors in the Lie group, the operations are the elements of the Lie group, and the measurements are associated with dual vectors. I develop these notions in the following.

Markovian dynamics

When discussing quantum dynamics, an important distinction must be made between Markovian and non-Markovian dynamics. This distinction is still somewhat controversial at the time of this thesis, and various definitions appear in the literature. In my treatment, I will assert that Markovian dynamics are dynamics in which the action of a gate depends only on its identity and on the state upon which it acts. In other words, a Markovian dynamical process is one in which each operation can be represented as a CPTP map that can be defined via the Lindblad equation. Another useful working definition could be that a Markovian quantum processor has a gate set with a static Lie group. Circuits that execute in the presence of changing Lie group dynamics then are non-Markovian because the gates depend on the external context of the drift process.

States

The term “quantum state” is used to describe both the physical state of a quantum system and a mathematical model of that system. In their mathematical form, quantum states are represented as arrays of complex numbers. One can think of quantum states as generalizations of probability distributions. Quantum states are defined either in a state vector or a density matrix, developed below, which are levels of abstraction that describe different approximations of noisy quantum dynamics.

State vectors A state vector representation deals with pure states. A pure state is one that contains no classical uncertainty, ie., has von Neumann entropy 0. Pure states are written as the kets of bra-ket notation, eg., $|\psi\rangle$. A pure state is represented by a vector of complex numbers whose modulus square is unity, as $|\langle\psi|\psi\rangle|^2 = 1$. In the language of Lie groups, pure states are the normalized vectors in the vector space that the Lie group $SU(N)$ acts upon.

Density matrices A density matrix representation deals with mixed quantum states. A mixed quantum state is one that contains classical uncertainty, ie., has von Neumann entropy greater than 0. Density matrices are written as complex matrices with trace equal to 1. A density matrix should be thought of as a probability distribution on a set of pure states, as

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|. \quad (3.1)$$

Another representation for density matrices is the “super-operator” or Hilbert-Schmidt representation, which essentially amounts to a vectorization of the density matrix. In this fashion, one writes $|\rho\rangle\rangle$ to denote a “super-ket” that is a vectorized density matrix. Super-kets can be thought of as vectors in a vector space associated to CPTP maps.

Operations

Quantum operations map quantum states to new quantum states. As with quantum states, quantum operations can be defined either in a state vector or density matrix formalism. In the state vector formalism, quantum operations are modeled as unitary matrices acting on pure states. In the density matrix formalism, quantum operations are modeled as quantum channels. Both the unitary and quantum channel models are elaborated on below.

In a Markovian model, an operation’s action is totally determined by its identity and the state upon which it acts. In a non-Markovian model, the action of an operation is determined by additional context. This additional context can include things like previously applied operations, time, or a changing environment. Most of this work considers Markovian models and very simple models of non-Markovian dynamics.

In Lie’s language, quantum operations are the elements of a group, and the states are the vectors upon which they act. In the case of unitary dynamics, this is an exact analogy because the set of unitary matrices forms a Lie group $SU(N)$. However, in the case of density matrices, the set of quantum channels only forms a semi-group [RH11; BP02] because decoherence has no physical inverse. Nevertheless, it is often useful to reason about quantum channels using ideas from Lie theory, as discussed in §3.3.

Measurements

Quantum measurements are perhaps the most difficult concept of the three components of quantum dynamics. The measurement problem of quantum mechanics [Sch05; Leg05] indicates that the foundations of quantum theory are still somewhat uncertain – as are the outcomes of quantum measurements. Nevertheless, there is a well defined mathematical formalism that deals with quantum measurements in generality: the notion of positive operator-valued measures, developed below.

A **positive operator-valued measure** (POVM) is, in its most general form, a decomposition $\{E_\mu\}_{\mu \in \Sigma}$ of the identity matrix in terms of positive semi-definite matrices. Each element E_μ corresponds to an outcome labeled by characters μ in an alphabet Σ [Wat18]. When a mixed state ρ or a pure state $|\psi\rangle$ is subjected to a measurement represented by a POVM, then the outcome μ is obtained with probability $\text{Tr}(E_\mu \rho)$ or $\text{Tr}(E_\mu |\psi\rangle\langle\psi|)$. In this way, a POVM induces a probability distribution or measure structure on measurement outcomes.

When a POVM consists exclusively of projection operators, it is called a **projection valued measurement** (PVM). In the gated-based model of quantum computing, it is usual to assume PVMs that are performed in the standard computational basis, which I will refer

to as computational basis measurements or just as measurements. Computational basis measurements on n qubits are defined by the decomposition

$$\sum_{\mu \in \text{range}(2^n)} |\mu\rangle\langle\mu| = \mathbb{1}. \quad (3.2)$$

where each $|\mu\rangle\langle\mu|$ is a rank-1 projector.

PVMs can be associated with dual vectors in representations of quantum gates. To see this, observe that one can write $\text{Tr}(|\psi\rangle\langle\psi| |\mu\rangle\langle\mu|) \equiv |\langle\mu|\psi\rangle|^2$. Here, $\langle\mu|$ is an element of the dual vector space of the Lie group $\text{SU}(N)$. Furthermore, one can work with “super-bras” and write $\text{Tr}(\rho |\mu\rangle\langle\mu|) \equiv \langle\langle\mu|\rho\rangle\rangle$. Here $\langle\langle\mu|$ is a dual vector in the Lie group of extended CPTP maps.

There is another useful identification of PVMs in terms of the character theory of groups (see the last paragraph of §2.3.1 of [Hay16]). If one associates E_μ with the subspace $\mathcal{H}_\mu$ of the total Hilbert space of state vectors $\bigoplus_\mu \mathcal{H}_\mu$, and one further associates ρ with an element g of a group, as $\rho = f(g)$, with the condition that f be a completely reducible representation, then the probability to obtain outcome μ after application of the PVM is $\text{Tr}(f(g)E_\mu)$ that is, in fact, the character of $\chi_{f_\mu}(g)$ of the irreducible representation f_μ on the subspace $\mathcal{H}_\mu$. In this way, characters have meaning as measurement probabilities.

3.2 Unitary models and coherent errors

Coherent errors correspond to persistent, systematic errors in the operation of a quantum gate. Coherent errors preserve the entropy of the state upon which they act, which may be taken as the defining property of a coherent process. In the case of quantum information processing devices that exploit fragile interference effects, even the smallest coherent errors can have disastrous effects on long computation. Coherent errors build up with repeated application and can convert a useful interference effect into useless diffusion. Coherent errors should be removed by precise calibration of classical control parameters that implement the waveforms that drive a quantum computation, which is the focus of §III.

To model coherent errors, the *unitary representation* of quantum gates is most appropriate. A small coherent error may be uniquely defined by a post-gate residual unitary error, so that the actual implementation of target unitary $\tilde{U}$ is

$$U = R\tilde{U} \quad (3.3)$$

where $U \equiv U\tilde{U}^{-1}$ represents residual error in the gate. The goal of calibration is to set $R = \mathbb{1}$.

Infinitesimal dynamics and Hamiltonians

The usual theory of quantum mechanics [SN20; Sha12] discusses coherent processes in terms of Hamiltonians, which are generators of unitary dynamics. As quantum engineers, we tend

to think in terms of unitary operators, which correspond to time-integrated Hamiltonian dynamics. Understanding the connection between infinitesimal Hamiltonian dynamics and resulting unitary processes is useful to evaluate the physical origins of noise.

The Schrödinger-Liouville-von Neumann (SLN) equation [SN20] models the time-dependent evolution of a quantum state under a coherent Hamiltonian $\mathbf{H}$ that generates infinitesimal dynamics

$$\frac{\partial \rho}{\partial t} = -i[\mathbf{H}, \rho] \quad (3.4)$$

for state ρ and Hamiltonian $\mathbf{H}$ under the assumption that $\hbar = 1$. In this way, the state at some time is defined

$$\rho(t) = U(t)\rho(0)U(t)^\dagger \quad (3.5)$$

where $\rho(0)$ is an initial state and $U(t)$ is the time evolution operator

$$U = \exp_+ \left[-i \int_0^t d\tau \mathbf{H}(\tau) \right] \quad (3.6)$$

where $\mathbf{H}(\tau)$ is a time-dependent Hamiltonian and $\exp_+$ is the time-ordered exponential series (i.e. a Dyson series) [SN20]

$$\exp_+ \left[-i \int_0^t d\tau \mathbf{H}(\tau) \right] \equiv \mathbb{1} + \sum_{n=1}^{\infty} (-i)^n \int_{t_0}^t dt_1 \int_{t_0}^{t_1} dt_2 \dots \int_{t_0}^{t_{n-1}} dt_n \mathbf{H}(t_1) \mathbf{H}(t_2) \dots \mathbf{H}(t_n). \quad (3.7)$$

In the case of constant Hamiltonian evolution, $\mathbf{H}(t) = \mathbf{H}_0$ for all t , then

$$U(t) = \exp(-i\mathbf{H}_0 t). \quad (3.8)$$

In the case of commutative Hamiltonian evolution, $[\mathbf{H}(t_1), \mathbf{H}(t_2)] = 0 \quad \forall t_1, t_2$, then

$$U = \exp \left[-i \int_0^t d\tau \mathbf{H}(\tau) \right]. \quad (3.9)$$

i.e., no time-ordering is required. However, if the driving Hamiltonian dynamics are non-commutative, then the time evolution operator can only be solved for with the Dyson series, which is a type of Volterra series².

Lie algebra of coherent errors

Lie algebras provide a very useful model of coherent error. There are two essential ways to define a unitary representation of a coherent error: the during-gate and the post-gate error representation. In case of unitary coherent error, both the during-gate and post-gate

²Volterra filters

representations are equivalent³. Errors in the during-gate representation occur inside the exponential of the target operation, as

$$U_{\text{during}}(x) = \exp(-i(h_a + x_a)\mathbf{X}_a) \quad (3.10)$$

or, in our short-hand:

$$U_{\text{during}}(x) = e^{h+x} \quad (3.11)$$

where h represents the target operation and x represents a coherent error. Both x and h are vectors in the Lie algebra $\mathfrak{su}(N)$. x is generally assumed small enough that quadratic terms in the BCH expansion can be ignored, whereas h is allow to be large. During-gate representations are perhaps more intuitive when one thinks about integrating a sequence of infinitesimal operations that generates a large operation.

The post-gate representation separates the target operation from the noise, as

$$U_{\text{post}}(x) = e^x e^h. \quad (3.12)$$

The post-gate representation is nice because it allows us to separate out residual errors from the error from the target operations. The adjoint representations of a Lie group and algebra can be exploited to analyze post-gate errors and derive useful properties, as in twirled RPE, §5.5.

3.3 Quantum channels and incoherent errors

Whereas coherent errors do not increase the entropy of a state, incoherent errors generally introduce additional classical uncertainty into a quantum state and hence increase entropy. Unitary representations are not sufficient to describe general incoherent noise effects. Instead, we require the notion of a quantum channel, which provides a more general model of quantum state evolution. The Pauli transfer matrix (PTM) is a particularly convenient representation. The effects of incoherent noise are best understood with the theory of open quantum systems [RH11; Lid19; BP02], in which incoherent quantum dynamics can be derived from unitary dynamics acting on a larger Hilbert space. A quantum master equation generalizes the SLN Equation 3.4 to incoherent dynamics with the addition of stochastic “jump operators.” As was the case for unitaries, it is useful to consider the Lie algebra of small incoherent noise processes; however, we will see that some peculiarities accompany this representation.

Quantum channel representations

Quantum channels [Wat18; NC02] provide a general description of quantum dynamics that can describe both coherent and incoherent errors. A quantum channel describes the most

³This will not be the case for incoherent gate error, for which the during-gate error is not universal. In the unitary case, we can use the fact that the Lie algebra $\mathfrak{su}(N)$ covers its associated group $\text{SU}(N)$ to argue that given $e^x U$ for x in the Lie algebra, there always exists an x' in the Lie algebra such that $e^{x'} = e^x U$. See §3.3 for the CPTP case.

general Markovian process that maps a physical state to a physical state. Valid quantum states are density matrices with real and non-negative eigenvalues that sum to one, so it follows that a quantum channel should preserve the trace of a density matrix (to ensure that the eigenvalues of the resulting map sum to 1) and also be positive (to ensure that it does not produce states with negative eigenvalues). In fact, it is necessary to impose that the channel be completely positive⁴. Complete positivity ensures that when the channel acts on one part of an entangled state in a larger Hilbert space, the resulting state is a physical state. Finally, to reflect linearity in quantum mechanics, quantum channels are generalised assumed to be convex linear, which is the property that

$$\Phi\left(\sum_j p_j \rho_j\right) = \sum_j p_j \Phi(\rho_j). \quad (3.13)$$

There are four essential representations of quantum channels: the linear operator (standard) representation, the Kraus (operator-sum) representation, the Choi matrix (state) representation, the Steinspring dilation (system-environment) representation [Wat18], which I briefly summarize below. The four representations are related through isomorphisms, depicted in Fig. 3.1.

The **standard representation** of a quantum channel writes operations as linear operators that map vectorized density matrices to vectorized density matrices

$$G(\Phi) : |\rho\rangle\rangle \rightarrow |\rho'\rangle\rangle. \quad (3.14)$$

The probability of measurement μ after application of the channel Φ on an input state ρ is evaluated as a product of a row vector, a matrix, and a column vector

$$\Pr(\mu|\rho, \Phi) = \langle\langle E_\mu | G(\Phi) |\rho\rangle\rangle. \quad (3.15)$$

The standard representation is used in this work unless otherwise stated. It provides the most convenient representation for numerical evaluation.

The **operator-sum representation** of a quantum channel writes channel operations as a linear combination of maps on density matrices:

$$\mathcal{K}^\Phi[\rho] = \sum_j K_j \rho K_j^\dagger \quad (3.16)$$

for a set of Kraus operators $\bigcup_j \{K_j\}$ that satisfy $\sum_j K_j^\dagger K_j = \mathbb{1}$. In the operator-sum representation, a probability is evaluated

$$\Pr(\mu|\rho, \Phi) = \text{Tr}[E_\mu \mathcal{K}^\Phi[\rho]]. \quad (3.17)$$

⁴A completely positive map Φ is such that $\mathbb{1}_n \otimes \Phi$ is positive for all integers $n > 0$. It suffices to ensure that this condition is satisfied for $n = \dim(\Phi)$.

The operator-sum representation is useful because it ensures complete positivity. However, the Kraus operators of a channel are not unique, and the probabilities of the channel are invariant under any unitary change of basis of the Kraus operators.

The **state representation** of a quantum channel writes the channel in terms of its action on one-half of a maximally entangled state

$$J(\Phi) = \Phi \otimes \mathbb{1} \left[\frac{1}{2^n} \sum_{i,j=1}^{2^n} |i, i\rangle\langle j, j| \right]. \quad (3.18)$$

In this way, the state representation writes a valid quantum state for every valid quantum channel. In this representation a measurement probability may be evaluated

$$\Pr(\mu|\rho, \Phi) = \text{Tr}[(E_\mu \otimes \mathbb{1})J(\Phi)]. \quad (3.19)$$

The Choi matrix $J(\Phi)$ can be used to easily verify complete positivity⁵, and its eigenvectors correspond to vectorized Kraus operators.

Finally, the **system-environment representation** of a quantum channel writes the channel as the partial trace of unitary dynamics in a larger Hilbert space

$$\Phi[\rho] = \text{Tr}_{\mathcal{H}'} [U^\Phi(\rho \otimes |0\rangle\langle 0|)(U^\Phi)^\dagger] \quad (3.20)$$

for some environment Hilbert space $\mathcal{H}'$ and system-environment unitary U^Φ that defines the representation. Similarly to the Kraus representation, a probability in this representation may be evaluated

$$\Pr(\mu|\rho, \Phi) = \text{Tr}[E_\mu \Phi[\rho]]. \quad (3.21)$$

The existence a system-environment representation, as promised by the Steinspring dilation theorem [Wat18], is of fundamental importance and motivates the understanding of incoherent dynamics as results from coupling with an environment.

Pauli transfer matrix

The standard representation of a quantum channel writes gates as matrices that act on vectorized density matrices. One is free to choose any valid basis and write the states, operations, and measurements in that basis. A particular choice of basis turns out to be convenient, resulting in a *Pauli transfer matrix* (PTM). PTMs have real components and come equipped with a natural block structure that reveals group theoretic properties. A useful introduction to PTMs can be found in [Nie+21; Gre15]. [HBG24] discusses PTMs as a change of basis, and [RMM22] describes use in partial tomography.

To introduce the notion of a PTM, start by expanding a density matrix in the Pauli basis

$$\Phi[\rho] = \Phi \left[\frac{1}{d} \mathbb{1} + \frac{1}{d} \sum_{i=1}^{4^n} c_i \mathbf{P}_i \right] \quad (3.22)$$

⁵When the Choi matrix is positive, the associated channel is completely positive.

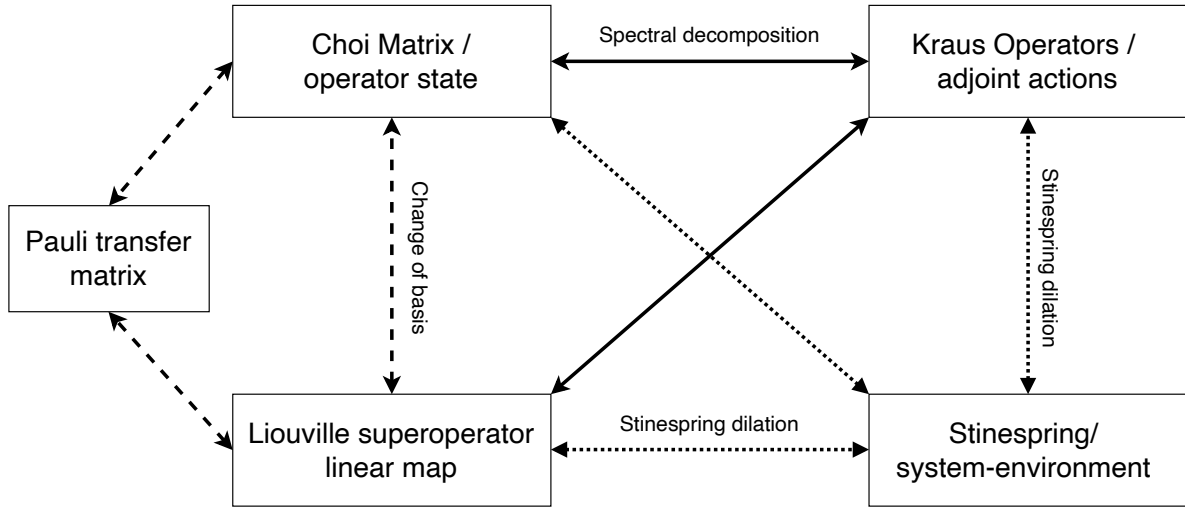


Figure 3.1: *Relationships between quantum channel representations.* The four canonical representations of quantum channels are depicted with the associated translation action between representations. Each translation action is represented by a particular line type (solid, dotted, dashed) The Pauli transfer matrix representation should be understood as a particular choice of basis for the Liouville representation. See also [WBC11].

with real coefficients c_i . It follows from linearity that

$$\Phi[\rho] = \frac{1}{d} \Phi[\mathbb{1}] + \frac{1}{d} \sum_{i=1}^{4^n} c_i \Phi[\mathbf{P}_i]. \quad (3.23)$$

We see that the action of a channel on a density matrix can be characterized completely by its action on the Pauli basis $\mathcal{P}^n$. This fact motivates us to define a Pauli transfer matrix G for channel Φ in terms of the “transfer” of weight between basis elements:

$$[G]_{ij} = \frac{1}{d} \text{Tr}[\mathbf{P}_i \Phi[\mathbf{P}_j]]. \quad (3.24)$$

It is unusual, but completely valid, to calculate the action of a channel on a Pauli operator as implied by $\Phi[\mathbf{P}]$. The resulting PTM G acts on vectorized density matrices in the Pauli basis

$$|\rho\rangle\rangle = \begin{bmatrix} 1 \\ c_1 \\ \vdots \\ c_{4^n} \end{bmatrix} \quad (3.25)$$

so that $G |\rho\rangle\rangle \cong \Phi[\rho]$.

Representation of an n -qubit CPTP quantum channel as a Pauli transfer matrix results in the block structure

$$G = \begin{bmatrix} 1 & \vec{0}^T \\ \vec{v} & R \end{bmatrix}. \quad (3.26)$$

$\vec{v}$ is a real column vector of length $4^n - 1$ that represents a translation, and R is a real square matrix that can be understood to represent a combination of dilation and rotation. For unital channels⁶ $\vec{v} = 0$. A unitary channel has an orthogonal R and $\vec{v} = 0$.

Trace preservation enforces the row vector of zeros $\vec{0}^T$ in the top-right block and the 1 in the top-left corner, see⁷. Completely positivity does not manifest as a simple constraint on the form of the PTM and instead is enforced by inequalities between components of $\vec{v}$ and R .

The block structure of a PTM, Eq. 3.26, indicates that a general n -qubit PTM (and hence a quantum channel) might be understood as an element of the general affine group $\text{GA}(4^n)$. However, not all valid quantum channels have an invertible R , and even if R is invertible, the inverse of a PTM may not be CP, which spoils a formal identification. Nevertheless, if the incoherent part of the process is small and non-destructive, then we are free to imagine an extension of the set of CPTP processes to the general affine group. We can recover a CPTP process from a TP process by finding the nearest CPTP process to the TP process under a suitable operator norm, e.g., the usual Frobenius distance. We shall soon see §3.3 that, under sufficient conditions, the matrix R can be further decomposed into a “Hamiltonian” and a “stochastic” part that correspond to coherent and incoherent dynamics.

Lindblad equation

Just as the SLN Equation 3.4 connects infinitesimal dynamics to unitary operations, the Gorini-Kossakowski-Sudarshan-Lindblad (GKSL) (or just Lindblad) equation [BP02; Man19] connects infinitesimal dynamics to quantum channels. The Lindblad equation may be derived from a number of different approaches [Lid19], which are not discussed here. Operationally, the Lindblad equation may be understood as an augmentation of the SNL equation with additional stochastic jump operators

$$\frac{\partial \rho}{\partial t} = -i[\mathbf{H}, \rho] + \sum_{n,m} \Lambda_{nm} \left(\mathbf{A}_n \rho \mathbf{A}_m^\dagger - \frac{1}{2} \{ \mathbf{A}_m^\dagger \mathbf{A}_n, \rho \} \right) \quad (3.27)$$

where $\bigcup_m \{\mathbf{A}_m\}$ are arbitrary operators, $\{\cdot, \cdot\}$ denotes anti-commutation, and Λ is a positive semi-definite matrix, which ensures CPTP dynamics. While one can use any number of operators written in any basis, an N -dimensional system can be completely describe by an operator basis with $N^2 - 1$ elements.

⁶A **unital channel** is one that maps the identity to the identity, i.e., $\Phi(\mathbb{1}) = \mathbb{1}$

⁷The trace of a vectorized density matrix $|\rho\rangle\rangle$ is

Actually, the general form of the Lindblad equation can be further reduced by diagonalization of Λ with a unitary change of basis that results in a form with a single summation

$$\frac{\partial \rho}{\partial t} = -i[\mathbf{H}, \rho] + \sum_j \gamma_j \left(\mathbf{L}_j \rho \mathbf{L}_j^\dagger - \frac{1}{2} \{ \mathbf{L}_j^\dagger \mathbf{L}_j, \rho \} \right). \quad (3.28)$$

In this form, γ_j is a stochastic jump rate and $\mathbf{L}_j$ is the corresponding stochastic jump operator.

The Lindblad equation is a general Markovian master equation and a cornerstone of open quantum system theory [RH11; Lid19; BP02]. One should understand the Lindblad equation as the generator of quantum channels. Given a set of jump rates, operators, and a driving Hamiltonian, the Lindblad equation may be integrated to obtain a quantum channel representation.

Lie algebra of non-destructive quantum channels

Constructing a Lie algebraic description of quantum channels, analogous to the unitary Lie algebras of coherent errors §3.2, provides us with a convenient algebraic framework to describe error process. In particular, such a description allows us to define a working hierarchical classification of noise channels, see Fig. 3.2, supporting physically meaningful intuition and statements about measured noise processes.

In developing an algebraic classification for general Markovian noise, one encounters a subtle mathematical issue. Unlike coherent errors, which always have a well-defined Lie algebra, incoherent errors may not always be invertible. Quantum channels form a semi-group [Ali02], as they are associated, have an identity element, but are not always invertible. However, if we impose that the channel be non-destructive, i.e., have a linear operator representation with non-zero eigenvalues, then the matrix representation can be inverted. The inverse may not necessarily be a valid channel, as it may not be CP, but is mathematically well defined, so it can be used to define a Lie group and associated Lie algebra. A simple way to do this is to restrict to small errors [Blu+22]. Alternatively, a polar decomposition of quantum channels [CAE19] also support general group-theoretic analysis. Here, I take a slightly different approach and consider the Lie algebra of the Lindblad equation itself.

To begin, write the Lindblad equation

$$\frac{\partial}{\partial t} \rho = \mathcal{H}[\rho] + \mathcal{D}[\rho]. \quad (3.29)$$

or in a linear-operator representation

$$\frac{\partial}{\partial t} |\rho\rangle\rangle = \mathbf{H} |\rho\rangle\rangle + \mathbf{D} |\rho\rangle\rangle. \quad (3.30)$$

It follows that for all $t > 0$,

$$|\rho(t)\rangle\rangle = \exp((\mathbf{H} + \mathbf{D})t) |\rho(0)\rangle\rangle. \quad (3.31)$$

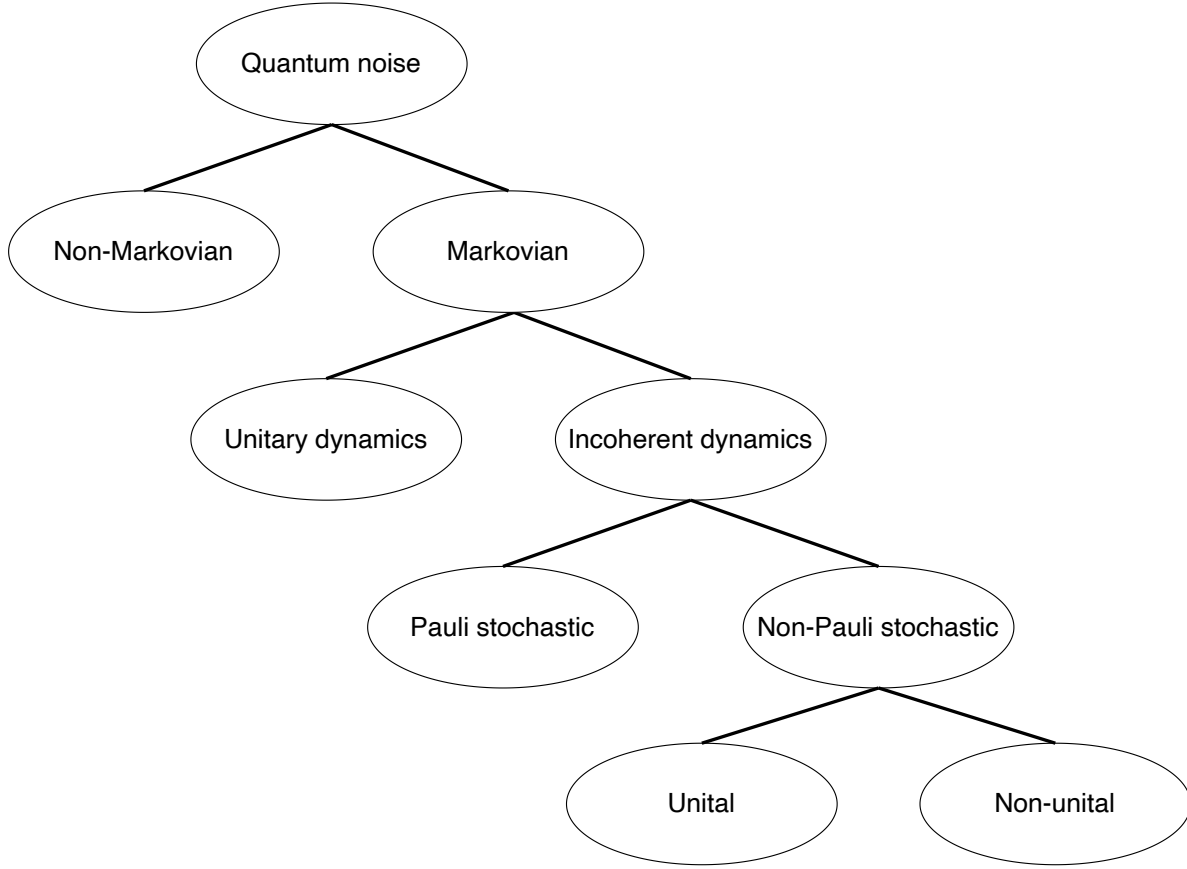


Figure 3.2: *Hierarchical classification of quantum noise channels.* A general CPTP noise channel can be separated into unitary and non-unitary dynamics. The non-unitary dynamics correspond to decoherence, which can be further classified into a number of sub-categories.

Hence, $\mathbf{H} + \mathbf{D}$ forms a Lie algebra. $\mathbf{H}$ is the unitary part and $\mathbf{D}$ is the decoherence, providing the first level of the classification of Fig. 3.2.

$\mathbf{H}$ is an element of $\mathfrak{su}(N)$ and the generator of the unitary part of dynamics (recall the discussion of the SLN Equation 3.4). In fact, $\mathfrak{su}(N)$ is closed under commutation, so it follows that $\mathbf{H}$ generates a unitary sub-algebra. Terms in $\mathbf{H}$ may be written in the Pauli basis as

$$\sum_j \theta_j (\mathbf{P}_j^T \otimes \mathbb{1} - \mathbb{1} \otimes \mathbf{P}_j). \quad (3.32)$$

Now, consider the decoherence operator $\mathcal{D}$. A general decomposition (Eq. 3.27) may be written in the Pauli basis

$$\frac{\partial \rho}{\partial t} = -i[\mathbf{H}, \rho] + \sum_{n,m} [D]_{nm} \left(\mathbf{P}_n \rho \mathbf{P}_m^\dagger - \frac{1}{2} \{ \mathbf{P}_m^\dagger \mathbf{P}_n, \rho \} \right) \quad (3.33)$$

for a positive semi-definite D . The Pauli stochastic terms (see next section) correspond to the diagonal part of the decoherence. These are generated by basis elements of the form

$$\mathbf{P}_j^T \otimes \mathbf{P}_j - \mathbb{1} \otimes \mathbb{1}. \quad (3.34)$$

It follows that the non-diagonal parts generate the non-Pauli stochastic terms, providing the 2nd division in the classification of Fig. 3.2.

Cholesky decomposition of decoherence

A nice parametrization of the decoherence part of the Lindblad equation may be developed by observing that the positive semi-definite constraint on D in Eq. 3.33 implies that this matrix *has the form of a covariance matrix*. This remarkable fact motivates an interpretation of quantum noise processes as a sort of “Gaussian random variable” where the mean is determined by the unitary part and the covariance determines the decoherence. This observation also provides another useful parametrization of decoherence in terms of Cholesky factors of D .

Observe that, because D is positive semi-definite, it may be written in the form

$$D = LAL^T \quad (3.35)$$

where L is a lower-unit-triangular matrix with 1's on the diagonal and unconstrained lower triangle

$$\Sigma = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ l_{2,1} & 1 & 0 & \dots & 0 \\ l_{3,1} & l_{3,2} & 1 & \dots & 0 \\ \vdots & & & \ddots & \vdots \\ l_{N,1} & l_{N,2} & l_{N,3} & \dots & 1 \end{bmatrix} \quad (3.36)$$

and A is a diagonal matrix $A = \text{diag}(a_1, a_2, \dots, a_N)$. In this way, A contains the eigenvalues of the original “covariance” matrix D , which is analogous to techniques in principle component analysis. The LAL^T decomposition de-correlates the noise process so that the elements of A quantify variances and elements of L quantify correlations in the noise. It is useful to observe that

$$[D]_{i,j} = \sum_k l_{i,k} l_{j,k} a_k \quad (3.37)$$

Non-unital processes

The final division in the noise hierarchy Fig. 3.2 is between unital and non-unital noise processes. The non-unital part is contained in the decoherence $\mathbf{D}$ and it is clearly not contained in the Pauli-stochastic sub-algebra. The non-unital components become clear when the decoherence is written in the Pauli transfer representation

$$\mathbf{D} = \begin{bmatrix} 1 & \vec{0}^T \\ \vec{v} & S \end{bmatrix}.$$

In this case, $\vec{v}$ transfers weight from Pauli coefficients to the identity, which is non-unital. The action of S excludes the identity, so it is unital. In this way, $\vec{v}$ corresponds to the non-unital part.

Consider the general affine algebra $\text{ga}(N)$ that generates non-destructive PTMs:

$$L_D = \begin{bmatrix} 0 & \vec{0}^T \\ \vec{z} & Z \end{bmatrix}. \quad (3.38)$$

Here

$$\vec{z} = \sum_{k>0} z_k |k\rangle\rangle \langle\langle 0| \quad (3.39)$$

and

$$Z = \sum_{i,j>0} Z_{i,j} |i\rangle\rangle \langle\langle j|. \quad (3.40)$$

Observe

$$\left[|k\rangle\rangle \langle\langle 0|, |k'\rangle\rangle \langle\langle 0| \right] = 0 \quad (3.41)$$

thus

$$[\vec{z}, \vec{z}^\dagger] = 0. \quad (3.42)$$

Also

$$\left[|k\rangle\rangle \langle\langle 0|, |i\rangle\rangle \langle\langle j| \right] = -\delta_{i,k} |i\rangle\rangle \langle\langle 0| \quad (3.43)$$

thus

$$[\vec{z}, Z] = \vec{z}. \quad (3.44)$$

Finally,

$$\left[|i\rangle\rangle \langle\langle j|, |j'\rangle\rangle \langle\langle i'| \right] = \delta_{j,j'} |i\rangle\rangle \langle\langle i'| - \delta_{i,i'} |j'\rangle\rangle \langle\langle j| \quad (3.45)$$

thus

$$[Z, Z] = Z. \quad (3.46)$$

These observations imply: the non-unital sub-algebra of the Lindblad equation is a nilpotent ideal, and the unital sub-algebra is closed.

3.4 Pauli error models

General incoherent quantum noise can be largely unstructured, with complicated correlated-stochastic and affine effects. A useful approximation that captures a large class of important noise effects is the Pauli noise channel. Indeed, Pauli noise channels are widely studied and importantly appear as a key element of efficient quantum error correcting code design, see §8. Furthermore, techniques have been developed to “twirl” general noise channels and convert them to a Pauli noise channel, and other sophisticated techniques have been developed to efficiently estimate k -local Pauli noise channels. For these reasons, Pauli noise channels and their representations deserve special attention.

The Pauli noise channel

The action of a Pauli noise channel on a density matrix ρ is described by a classical distribution on Pauli matrices. It is defined

$$\mathcal{E}[\rho] = \sum_a p_a P_a \rho P_a \quad (3.47)$$

where P_a is a Pauli matrix on n qubits and a spans the range 4^n (including the identity indexed at 0). We see that Pauli noise can be understood as a classical distribution on Pauli matrices. Pauli noise corresponds to the diagonal elements of the Pauli transfer representation of a quantum channel.

Twirling to a Pauli noise channel

The notion of randomized compiling and Pauli noise twirling⁸ can be used to tailor general quantum noise channels into Pauli noise channels. A good initial reference for the theory is [Min22]; the technique abounds in the literature e.g. [GZ13; WE16].

The form of a general G -twirl of an operator Λ over a group G is

$$\mathcal{T}_G(\Lambda) \equiv \sum_{g \in G} g \Lambda g^{-1}. \quad (3.48)$$

By conjugating a general CPTP noise channel with unitary Pauli gates, one can project onto a Pauli noise channel. The ability to twirl a general channel into a Pauli noise channel is one of the reasons that the Pauli channel is a fundamental and useful object in quantum computer engineering.

There are a number of ways to approach Pauli twirling, and a simple approach follows from the observation that the only elements in the Lindblad algebra (§ 3.3) that commute with all P are Pauli-stochastic terms. The proof of this proceeds in two steps: first show that all Pauli-stochastic terms commute with the operation of Pauli conjugation, then show that all other terms do not commute with Pauli conjugation. For the first step, write the super-operation representation of Pauli conjugation

$$P \rho P \mapsto (P^T \otimes P) \vec{\rho}.$$

Then, recalling that the form of a Pauli stochastic errors associated to Pauli Q is $Q^T \otimes Q - \mathbb{1} \otimes \mathbb{1}$. Now, observe that

$$\begin{aligned} [P^T \otimes P, Q^T \otimes Q] &= (QP)^T \otimes PQ - (PQ)^T \otimes QP \\ &= ((-1)^{Q \odot P} - (-1)^{\delta[P, Q]})(PQ)^T \otimes PQ. \end{aligned}$$

⁸I have interpreted Pauli noise twirling as the theory behind the experimental protocol of randomized compiling, but someone else said they are actually the same thing with different names to avoid a patent infringement.

where $\bar{\delta}$, as throughout this thesis, is a binary indicator of commutation (0 if operands commute else 1). Because $\bar{\delta}$ is symmetric, we have

$$[P^T \otimes P, Q^T \otimes Q - \mathbb{1} \otimes \mathbb{1}] = 0. \quad (3.49)$$

I leave the proof that the other terms do not commute as an exercise to the reader.

A more straightforward (and more technical) proof is to observe that $PQ\rho Q'P = QP\rho PQ'$ for given $Q, Q' \in \mathcal{P}$ and *all* $P \in \mathcal{P}$ iff $Q = Q'$, and then apply Schur's theorem to infer that the twirled representation will be block diagonal with a single trivial irrep. and a list of multiplicity-free irreps. with dimension 1 (see Character theory of Pauli noise below). The total number of irreps. is equal to the dimension of a Pauli transfer matrix representation. Hence, twirling over the Pauli group projects general noise processes into a diagonal process matrix with elements that correspond to decay along Pauli axes.

It is useful to note that one does not need to always twirl over the entire Pauli group with 4^n elements. In fact, it usually suffices to twirl over random samples drawn i.i.d. from the Pauli group. There are also analytic techniques, such as those discussed in [CB19], that can be used to find minimal Pauli designs to twirl over. Our discussion here illustrates a useful principle of characterization wherein gate sequences are summed over [Fan+24].

Representations of Pauli noise channels

The probability representation of Pauli noise is as a probability distribution on adjoint actions of the input state by Pauli operators, Eq. 3.47. This representation is equivalent to a Kraus channel with operators $\bigcup_j \{\sqrt{p_j} P_j\}$. To find the Pauli transfer representation, one writes

$$\begin{aligned} \Lambda_{jj} &= \frac{1}{d} \text{Tr} \left[P_j \left(\sum_k p_k P_k P_j P_k \right) \right] \\ &= \frac{1}{d} \sum_k p_k \text{Tr} [P_j P_k P_j P_k] \\ &= \sum_k (-1)^{\bar{\delta}[P_k, P_j]} p_k \end{aligned} \quad (3.50)$$

where $\bar{\delta}[a, b]$ is (as throughout this thesis) a binary indicator of commutation, and I used that $P_j P_k = (-1)^{\bar{\delta}[P_j, P_k]} P_k P_j$ for Paulis.

Another useful representation of stochastic Pauli errors is linked to the notion of a stabilizer tableau [Got98], discussed in §8 and common in QEC literature. I will call this the tableau representation of Pauli noise. In this representation, Pauli noise is again written as a distribution on Pauli operators, but now the action is represented in a stabilizer tableau with $2n$ binary variables. This representation is used in a simulation in which each gate is represented as a Clifford operator acting on stabilizer states in a tableau, and after each gate – or at specific locations in a circuit – a random Pauli tableau is sampled and applied. This representation of Pauli noise is very useful for QEC code design.

Estimating Pauli noise

Efficient techniques have been designed to estimate Pauli noise [Che+23; Che+24]. These results are rather mature at the time of writing and have been generalized to k -local Pauli noise learning [RF23] that can be done efficiently. Essentially, these approaches involve considering diagonal noise channels, applying sequences of randomly selected Pauli gates and particular Clifford gates, and measuring in particular bases. The random Pauli gates perform Pauli noise twirling and the Clifford gates serve to distinguish a subset of stochastic Pauli decays that are then picked up by the basis measurements. In the presence of SPAM errors, post-processing success probabilities reveals decays like

$$p(m) = A\lambda^m + B \quad (3.51)$$

where A, B are SPAM parameters and λ is an eigenvalue of the diagonal PTM. Curve fitting reveals λ .

In multi-qubit Pauli noise learning, there is a gauge problem [Che+24] that has actually been resolved nicely. One consequence of the Pauli noise learnability theorem is that learnable Pauli noise parameter reveals nice single-exponential decays as above iff. the noise that parameter is learnable. For given gate sets, there will exist non-learnable, gauge parameters that should be excluded in estimated models.

Character theory of Pauli noise

A useful approach to understanding the action of a twirl over the Pauli group is via a *character theory* of quantum channels [Hel+18; CRW20]. Essentially, this approach makes the observation that a G -twirl

$$\hat{\Lambda}_G = \sum_{g \in G} g \Lambda g^{-1} \quad (3.52)$$

may be represented as a quantum channel (usually a PTM) as

$$\hat{\Lambda}_G \cong \bigoplus_{\mu \in M} Q_\mu \oplus \mathbb{1}_\mu \quad (3.53)$$

where Q_μ is a matrix of shape $n_\mu \times n_\mu$, n_μ is the multiplicity of the irrep. μ , and $\mathbb{1}_\mu$ is an identity.

In the case of Pauli groups, the group action in the PTM representation is multiplicity free, and the twirl decomposes into a direct sum of 1-dimensional irreps. with character functions

$$\chi_\mu(P) = (-1)^{\langle \mu, P \rangle}, \quad (3.54)$$

where $\langle \mu, P \rangle$ is zero if the Paulis commute and else 1.

3.5 Depolarizing models

Finally, the simplest stochastic noise model is the depolarizing model. In this model, the state is replaced with some probability by the maximally mixed state

$$D_p[\rho] = p\rho + (1-p)\frac{\mathbb{1}}{d}. \quad (3.55)$$

While a poor model when applied naively, the depolarizing model can be enforced like Pauli stochastic noise. Furthermore, depolarizing noise has a fundamental role in QEC due to its simple structure and the fact that being able to correct for depolarizing noise also implies the ability to correct for a wide class of errors.

The depolarizing channel is particularly nice because it composes simply. Consider a sequence of depolarizing channels. The resulting channel has a depolarizing probability that is the product of the individual probabilities

$$\bigcirc_k D_{p_k} = D_{\prod_k p_k}. \quad (3.56)$$

This fact is often used to estimate depolarized error by fitting exponential decays; the success probability of measuring a basis element a after preparing a is

$$\langle\langle a | \bigcirc_k D_{p_k} | a \rangle\rangle = A + B\lambda^k \quad (3.57)$$

where A, B account for SPAM error.

Clifford twirling

As I have said, depolarizing noise can be enforced by a twirling over the Clifford group $\mathcal{C}$. One way to show this is to consider representations of Clifford twirling in the Pauli transfer representation. The PTM M that represents a Clifford operation c is defined

$$[M(c)]_{ab} = \frac{1}{d} \text{Tr}[P_a c P_b c^\dagger]. \quad (3.58)$$

Recall (§ 2) that the action of a Clifford operation on the Pauli group may be represented as a signed permutation matrix with determinant ± 1 . It follows that a representation of a Clifford as a PTM has the form

$$\begin{bmatrix} 1 & \vec{0}^T \\ \vec{0} & S \end{bmatrix} \quad (3.59)$$

where S is a signed permutation matrix. It then follows (see below) that twirling over the Clifford group projects a general channel in the PTM representation

$$\Lambda = \begin{bmatrix} 1 & \vec{0}^T \\ \vec{v} & V \end{bmatrix} \quad (3.60)$$

into a diagonal channel

$$\hat{\Lambda}_c = \begin{bmatrix} 1 & \vec{0}^T \\ 0 & \lambda \mathbb{1} \end{bmatrix} \quad (3.61)$$

where λ is a scalar depolarizing rate and $\mathbb{1}$ is the identity matrix. The depolarizing rate is equal to the average of the diagonal elements

$$\lambda = \frac{1}{d} \sum_j \Lambda_{jj}. \quad (3.62)$$

A formal derivation of this fact follows from an application of Schur's lemma. One can observe that representations of the Clifford group as PTMs have a single trivial irrep. and a single non-trivial irrep. of dimension equal to the dimension of the PTM minus 1. The non-trivial irrep. corresponds to depolarizing decay. This fact can also be derived by observing that the Clifford group twirl is a unitary 2-design.

These ideas can be illustrated by considering Clifford twirls on a single qubit. In this case, the PTMs have dimension 4 and there are 24 elements in the single-qubit Clifford group. The group elements represent the symmetry group S_4 , the symmetry group of the cube (without reflections) and of positive, signed permutations on four elements. Each Clifford group element is represented by a PTM with a block diagonal structure of a single trivial 1 in the top left and a block that is a signed permutation. The signed permutations can be no further diagonalized. The trivial block is the trivial irrep. and the permutation block is the standard representation of S_4 . It follows from Schur's lemma that a twirl over the Clifford block is multiplicity free with a single trivial and single non-trivial block.

3.6 Non-Markovian models

In this chapter, I have exclusively considered Markovian noise models that allowed for simple identification of a matrix representation that acts on a vector space. However, real quantum devices are known to experience drift in tomographic parameters [Niu+19], which indicates that actual devices will only be approximately described by Markovian models. In fact, drift is a key source of instability and decoherence in quantum processors, so having good models for drift is essential. The problem is that good models for drift are hard to define because non-Markovian quantum dynamics are notoriously difficult to model. As with any model design problem, striking the balance between a sufficient number of parameters and expressability is the key challenge. Because the Markovian models already have an exponentially large number of parameters, tractable non-Markovian model estimation is extraordinarily difficult.

A well developed model of non-Markovianity is that of process tensors [Whi+21], which embed each state preparation, operation, and measurement as an element in a very large spaces of tensors. Full estimation of process tensors is extraordinarily expensive for it requires measuring every configuration of all gates in all contexts. However, like gate set tomography, a complete theory provides useful insights into tractable approximations.

In this thesis, I will later §7 consider simple coherent drift models, which represent one example of a non-Markovian model as a stochastic process on Markovian models. This approach of representing non-Markovianity as stochastic parameters is promising for future work. One venue of exploration is to define stochastic processes on Lie groups [BB18]. In this way, one considers a model for stochastic dynamics of post-gate error like

$$R_{k+1}U = e^{w_k} R_k U \quad (3.63)$$

where I have used the Lie algebra short-hand. Here w is a random vector sampled from a drift distribution. In this way, one can ensure that the resulting post-gate error R_{k+1} is contained in the associated group to the drift algebra.

Stochastic drift can be defined in both unitary and CPTP formalisms. In the unitary formalism, one uses the algebra $\mathfrak{su}(N)$. In the CPTP formalism, one can use the Lindblad algebra – however, the parameters of the Lindblad algebra must be constrained to CP maps.

Chapter 4

Gate set tomography

State, process, and measurement tomography are powerful tools for diagnosing errors in a quantum processor. However, each of these protocols implicitly assumes the existence of a pre-calibrated *reference frame* (see Fig. 4.1) of perfect states and/or measurements. Errors in the operations that define such a reference frame can bias the tomographic reconstructions and lead to incorrect models for the operations under test. Gate set tomography (GST) [Nie+21; Gre15] is a family of calibration-free approaches to tomography that explicitly acknowledge that all elements of a quantum computer’s *gate set* — the native state preparations, measurements, and logic gates — are subject to errors. GST protocols are able to reconstruct self-consistent mathematical representations of a quantum computer’s native gate set and the errors afflicting it.

Around 2012, groups at IBM [Mer+13] and Sandia National Laboratories [Blu+13] independently identified the need for calibration-free characterizations of quantum operations. IBM approached this problem using a so-called “overkill” tomography protocol that utilizes all circuits of depth 3 or less and fits a gate set model with MLE. Sandia’s “linear GST” method uses similar circuits to standard QPT and fits a model with linear inversion. Variants of these early protocols are still in use to some extent, but since their introduction the family of GST protocols has evolved significantly. It now encompasses a rather broad set of experiment design and data analysis techniques for self-consistently estimating the parameters of a gate set model.

Of the variants of GST, the standard approaches include *linear GST*, mentioned above, and *long-sequence GST*, which uses long, structured quantum circuits and iterative MLE. Extensions to the standard practice have introduced approaches for characterizing larger processors, including randomized linear gate set tomography [Gu+21] and compressive gate set tomography [BRK23]. A streaming implementation of gate-set tomography [MY23] replaces the MLE estimator of long-sequence GST with an efficient Kalman filter that can process experimental results at rates comparable with circuit execution, enabling real-time characterization. Other recent work has attempted to reduce the number of circuits used in long-sequence GST [Rud+23; Ost+23] by reducing the number of fiducial state preparations and measurements that are used to probe a gate set.

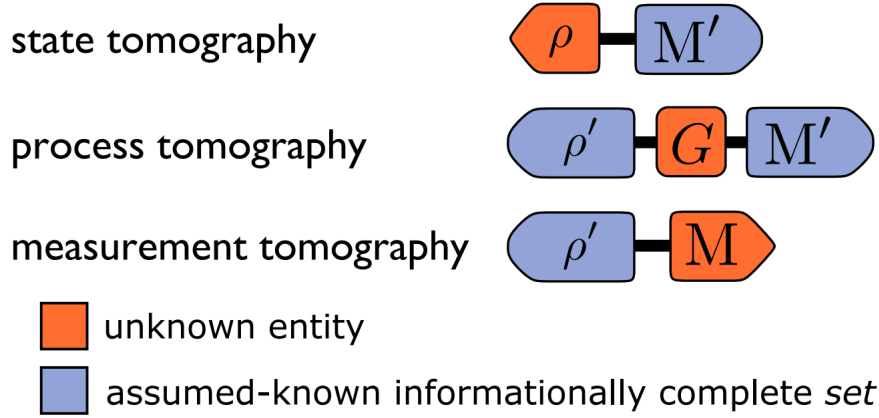


Figure 4.1: *Tomographic Reconstruction.* The structure of the circuits required for state, process, and measurement tomography are shown. Each of these protocols reconstructs an initially-unknown quantum operation (a state ρ , process G , or POVM M) by combining that operation into simple circuits with a set of *known* complementary operations. The known operations form a reference frame for estimation of the unknown operation’s matrix elements. If the reference frame is *informationally complete*, then all matrix elements of the unknown operation can be learned. We denote reference frame operations with “primed” symbols (ρ' and M') to indicate that they are *effective* (rather than *native*) state preparations and measurements, usually implemented by applying gate operations after or before a native state preparation or measurement. These techniques are limited by systematic errors stemming from imperfect *a priori* knowledge of the ρ' and M' . Figure and caption reproduced with permission from [Nie+21].

4.1 Gate set models

Gate-based quantum computers are devices that implement quantum circuits, which are sequences of instructions for applying logic operations to physical qubits. These instructions generally include a discrete set of quantum gates, as well as state preparation, and terminating (and possibly intermediate) measurements. It is convenient to construct a single mathematical object called a *gate set* that contains representations of all of the native instructions for a quantum device.

A gate set model provides a complete description of a noisy Markovian quantum computer. The standard gate set model assigns a mathematical representation to each aspect of the quantum computer:

1. State preparation: $|\rho\rangle\rangle \in \mathcal{B}(\mathcal{H})$, a 4^n -dimensional (column) vector,
2. Measurement: $\{\langle\langle E_i| \in \mathcal{B}(\mathcal{H})^*\}_{i=1}^{N_E}$, each a 4^n -dimensional dual (row) vector,

3. Logic gates: $\{G_i : \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H})\}_{i=1}^{N_g}$, each a $4^n \times 4^n$ -dimensional *process matrix*.

Here, $\mathcal{B}(\mathcal{H})$ is the space of bounded linear operators on a Hilbert space $\mathcal{H}$. The gate set model $\mathcal{W}$ is then defined as the union of the set of native gates, native state preparations, and native measurements:

$$\mathcal{W} = \{\mathcal{G}, \mathcal{S}, \mathcal{M}\}$$

where elements of $\mathcal{S}$ are the native states preparations, $\mathcal{M}$ are the native measurements, and $\mathcal{G}$ are the native gates. A gate set describes a *specific, limited* set of operations. A quantum processor may be capable of implementing other operations that are not listed in a particular gate set. A gate set $\mathcal{W}$ can only be used to describe and predict circuits built from the operations in $\mathcal{W}$.

It is standard to assume a single native state preparation in the computational zeros state $|0\rangle$ and a single native measurement in the computational basis, which reflects the capabilities of most current devices. The set of gates may be left unspecified at this point. I remark that the concept of a gate set model based on completely-positive, trace preserving (CPTP) process matrices is only an approximation, as it cannot readily take into account the effects of non-Markovian noise. However, gate set models are good approximations for devices that function pretty well already, and provide a starting point to generalize to non-Markovian effects.

In the context of gate sets, the word “gate” indicates an operation acting on the entire computer. The existing gate set formalism is not consistent with the alternative meaning of “gate” to denote an operation acting only on a subsystem (e.g., 1 or 2 qubits) of a quantum computer, which can be combined in parallel (by tensor product) with other gates on disjoint subsystems to produce a whole-computer operation called a circuit *layer* or *cycle*. In the gate set formalism, each layer (configuration of parallel gates) that can be performed should be included in the gate set as a distinct “gate.” Gate sets do not generally assume any connection or correlation between the actions of, for example, an X gate on qubit 1, an X gate on qubit 2, and parallel X gates on qubits 1 and 2. Treating each layer as an independent operation makes it possible — by comparing and contrasting different parallel combinations of gates — to study the effects of crosstalk on a device [Rud+21].

A gate set can be expressed in any representation that is convenient for the task at hand, but the most common convention is to use the Pauli transfer matrix representation and Hilbert-Schmidt space notation introduced in § 3 and represent initialization operations as superkets $|\rho\rangle\rangle$, logic gates G as transfer matrices, and POVM measurements as lists of effects $\{\langle\langle E_i|\}$. Using these representations, a general gate set is written as

$$\mathcal{G} = \left\{ \{|\rho^{(i)}\rangle\rangle\}_{i=1}^{N_\rho}, \{G_i\}_{i=1}^{N_G}, \left\{ \left\langle\langle E_i^{(m)}|\right\rangle \right\}_{m=1, i=1}^{N_M, N_E^{(m)}} \right\}. \quad (4.1)$$

For N_ρ the number of native state preparations, N_G is the number of distinct native gates, and N_M and N_E are the number of POVMs and the number of POVM effects, resp.

A *gate set model* is gate-set-valued function of some parameters — i.e., a map from a list of parameters to gate sets. A *fully-parameterized* gate set model assigns a free parameter to each matrix element of each operation in a gate set. For an n -qubit processor, a fully parameterized, trace-preserving gate set model contains $4^n - 1$ parameters per initial state, $16^n - 4^n$ parameters per logic gate, and $8^n - 4^n$ parameters per projective measurement.

Reduced models can be constructed that use fewer parameters, motivated by sparse matrix ansatz, structural properties of the processor’s Hilbert space [Blu+22], or knowledge of its low-level physics. These models have been proposed as a way to overcome the exponential growth of parameters with system size, and their construction is an area of active research. They have fewer parameters, but generally rely on assumptions, such as limited or no crosstalk, symmetries, or *ad hoc* ansatz like low-rank tensor networks. Physics-informed reduced models can have the additional advantage of more easily interpretable parameters, such as the intensity, frequency, or phase of a control field.

Finding the parameters of a gate set model (whether fully-parameterized or reduced) that fit and describe data from a particular device is the task of *gate set tomography*, discussed in detail below.

Circuits

A gate set is a model of a quantum computer that can be used to predict the measurement outcome distribution for arbitrary quantum circuits composed of elements of the gate set. For a circuit that comprises (i) preparing native state ρ , (ii) applying the sequence of operations $C = (g_1, g_2, \dots, g_L)$, and (iii) measuring the POVM $\{E_i\}$, the probability of measurement outcome i is given by Born’s rule as

$$p(i|\rho, C) = \langle \langle E_i | G_{g_L} G_{g_{L-1}} \dots G_{g_1} | \rho \rangle \rangle. \quad (4.2)$$

This can be written in the more familiar form,

$$p(i|\rho, C) = \text{Tr}[E_i \mathcal{C}(\rho)], \quad (4.3)$$

where $\mathcal{C}$ is the quantum channel of the circuit defined by the sequence of gates C .

4.2 Linear GST

Linear gate set tomography (LGST) is a self-consistent approach to simultaneous state, process, and measurement tomography that uses short quantum circuits and reconstructs a gate set model using linear inversion (see Fig. 4.2). Like process tomography, it assembles elements of the gate set to construct an informationally complete set of states and measurements, which are then used to probe the errors in elementary logic operations. Most quantum computing systems can natively prepare only a single initial state (typically $|000\dots\rangle$) and perform measurements only in a single basis (e.g., the computational basis), so a full, informationally complete set of states and measurements must be constructed from these native

operations by the application of short *fiducial* gate sequences. For instance, measurement in the $\{|+\rangle, |-\rangle\}$ basis can be performed by preceding a computational basis measurement by a Hadamard operation. In general, one could choose an informationally *overcomplete* fiducial set, but for simplicity of presentation, we assume exact informational completeness. See the Appendix of [Nie+21] for the general case.

Given an informationally complete set of fiducial states $\{|\rho'_j\rangle\rangle\}_{j=1}^{N_\rho}$ and an informationally complete set of fiducial measurement effects $\{\langle\langle E'_i|\}_{i=1}^{N_E}$, the LGST protocol prescribes a set of circuits whose output distributions provide sufficient information to estimate the parameters of a gate set model. To see how the protocol works, it is convenient to start by collecting all the fiducial states and measurement effects into matrices A and B defined as

$$A = \begin{pmatrix} \langle\langle E'_1| \\ \langle\langle E'_2| \\ \vdots \\ \langle\langle E'_N| \end{pmatrix} \quad (4.4)$$

and

$$B = \left(|\rho'_1\rangle\rangle, |\rho'_2\rangle\rangle, \dots, |\rho'_{N_\rho}\rangle\rangle \right) . \quad (4.5)$$

Now, the parameters of a gate G_k can be estimated by first estimating a matrix of probabilities $\mathbf{P}_k$ defined component-wise as

$$[\mathbf{P}_k]_{i,j} = \langle\langle E'_i| G_k |\rho'_j\rangle\rangle , \quad (4.6)$$

or as a matrix equation,

$$\mathbf{P}_k = A G_k B . \quad (4.7)$$

Measuring $\mathbf{P}_k$ is essentially standard QPT. However, because we do not assume a particular calibrated reference frame, we do not know A or B and hence cannot determine G_k from $\mathbf{P}_k$ alone.

In order to compensate for our ignorance of the reference frame, we measure an additional set of circuits that would correspond to QPT on the null operation. We arrange the results into a *Gram matrix* $\tilde{\mathbb{I}}$ defined component-wise

$$[\tilde{\mathbb{I}}]_{i,j} = \langle\langle E'_i|\rho'_j\rangle\rangle , \quad (4.8)$$

or as a matrix

$$\tilde{\mathbb{I}} = A B . \quad (4.9)$$

As long as we assume that the fiducial states and measurements are informationally complete, the Gram matrix is square and invertible: $\tilde{\mathbb{I}}^{-1} = B^{-1}A^{-1}$. It follows that multiplying both sides of Eq.4.7 by $\tilde{\mathbb{I}}^{-1}$ yields

$$\tilde{\mathbb{I}}^{-1}\mathbf{P}_k = B^{-1}G_kB , \quad (4.10)$$

or, solving for G_k ,

$$G_k = B\tilde{\mathbb{I}}^{-1}\mathbf{P}_k B^{-1} . \quad (4.11)$$

In this way, we have successfully defined G_k in terms of physically measurable quantities $\tilde{\mathbb{I}}$ and $\mathbf{P}_k$ and an *unknown gauge freedom* B . We can repeat this process for all of the gates G_k in the gate set and recover an estimate of all of them up to the same B .

Finally, to estimate the native state preparations $\{\rho^{(i)}\}_{i=1}^{N_\rho}$ and measurements $\{E_j^{(m)}\}_{m=1, j=1}^{N_M, N_E^{(m)}}$, we construct the following vectors,

$$[\mathbf{R}^{(l)}]_j = \langle \langle E'_j | \rho^{(l)} \rangle \rangle , \quad (4.12)$$

$$[\mathbf{Q}_l^{(m)}]_j = \langle \langle E_l^{(m)} | \rho'_j \rangle \rangle , \quad (4.13)$$

by measuring them experimentally on the device. Observe that we may write

$$\mathbf{R}^{(l)} = A |\rho^{(l)}\rangle\rangle , \quad (4.14)$$

$$\mathbf{Q}_l^{(m)T} = \langle \langle E_l^{(m)} | B . \quad (4.15)$$

Using the Gram matrix identity $\tilde{\mathbb{I}} = AB$ allows us to finally write all the elements of a gate set model in terms of measurable quantities and a gauge transformation B as

$$G_k = B\tilde{\mathbb{I}}^{-1}\mathbf{P}_k B^{-1} , \quad (4.16)$$

$$|\rho^{(l)}\rangle\rangle = B\tilde{\mathbb{I}}^{-1}\mathbf{R}^{(l)} , \quad (4.17)$$

$$\langle \langle E_l^{(m)} | = \mathbf{Q}_l^{(m)T} B^{-1} . \quad (4.18)$$

Any invertible matrix B defines an equally valid gauge transformation. No physical experiment can single out a “proper” gauge. The process of gauge fixing, discussed in Sec. 4.5, can permit analysis of gauge-dependent metrics, such as state or process fidelity.

In the above discussion, $\mathbf{P}_k$ is a matrix of circuit outcome probabilities that must be estimated from data. The maximum likelihood estimator for those probabilities is simply the observed frequency. If the circuit is run N times, then the finite sample fluctuations will lead to error in the estimate that scales like $\mathcal{O}(1/\sqrt{N})$. We see in Eq. 4.16 that G_k is linear in $\mathbf{P}_k$, so the error bars on the estimate of the G_k transfer matrix also scale as $1/\sqrt{N}$. This is the so-called “standard quantum limit” for parameter estimation [GLM06], and it results here from the fact that each gate is only used once per circuit (excluding any potential uses in creating the fiducials). In the next subsection, we discuss how we can go beyond this limit, using long circuits to achieve “Heisenberg” scaling in the estimation error.

4.3 Long Sequence GST

Long-sequence GST (LSGST) is an approach to self-consistent tomography of quantum gate sets that can beat the standard quantum limit [Nie+21]. To accomplish this, it requires two

adjustments to the LGST approach: i) the addition of longer quantum circuits that amplify gate errors, and ii) a new statistical estimation protocol suited to the updated experiment design.

LSGST circuits are formed in a similar fashion to LGST circuits. First, one selects an informationally complete set of state preparations and measurement fiducials, as in LGST. Where LGST uses these fiducials to probe each gate G_k in the gate set, long sequence GST uses them to probe each of an *amplificationally complete* list of “germs,” which are short sequences of native gates. The circuits run by LSGST comprise a fiducial state preparation, an L -fold repeated germ, and a fiducial measurement (see Fig. 4.3), with L typically chosen to be logarithmically-spaced integers, $1, 2, 4, 8, \dots, L_{\max}$. These many-fold repeated germs are what enable LSGST to achieve Heisenberg-limited scaling.

To see how this works, consider the repeating a single gate G many times. The resulting process can be computed from a spectral decomposition of the gate,

$$G = R \begin{pmatrix} e^{\phi_1} & & \\ & e^{\phi_2} & \\ & & \ddots \end{pmatrix} R^{-1}, \quad (4.19)$$

$$G^L = R \begin{pmatrix} e^{L\phi_1} & & \\ & e^{L\phi_2} & \\ & & \ddots \end{pmatrix} R^{-1}, \quad (4.20)$$

where $\text{diag}(e^{\phi_1}, e^{\phi_2}, \dots)$ is a diagonal matrix of (generally complex) eigenvalues and R is the change of basis matrix between the original and diagonalized bases. As we saw above, probing G with an informationally complete set of fiducials allows us to estimate its eigenvalues, and thus the complex phases ϕ_i , with uncertainty $\mathcal{O}(1/\sqrt{N})$. Similarly probing G^L allows us to estimate the *amplified* phases $L\phi_i$ with uncertainty $\mathcal{O}(1/\sqrt{N})$, giving a $\mathcal{O}(1/L\sqrt{N})$ uncertainty in the estimate of ϕ_i . This example also helps explain why GST does not exclusively use the longest sequences: knowledge of $e^{L\phi}$ is generally insufficient to reliably learn ϕ , particularly at large L , because the logarithm is multi-valued, and one must determine to which branch ϕ belongs.

In the above example, we saw that repetition has *amplified* the eigenvalues of the gate, and resulted in a much better estimate. But this process has not increased our sensitivity to *axis errors*, which only impact the matrix of eigenvectors R . To learn about axis misalignment between gates, we have to consider composite germs that consist of products of native gates. An *amplificationally complete* set of germs is one for which knowledge of their eigenvalues is sufficient to reconstruct all of the non-gauge, non-SPAM degrees of freedom in the gate set. Analyzing what errors a given germ amplifies is considerably simplified if one assumes that errors act as small perturbations to the target operation of a gate. In that case, one argues that the errors amplified by a given germ are those that commute with the target operation of the germ. It follows that each germ amplifies a subspace of gate errors, and a complete LSGST experiment design is found by searching over all possible germs until the amplified directions span the non-gauge subspace; see [Nie+21].

Fitting LSGST data is typically done using iterative maximum likelihood or least squares optimization. This approach begins by fitting a model to the shortest ($L = 1$) sequences, and then using that to seed the optimizer for the next round, which includes the $L = 1$ and $L = 2$ circuits. This procedure repeats until all circuits have been added and the optimizer has converged. This approach helps to avoid the wrong branch issue described above, and in practice is extremely robust.

To find the MLE estimate, consider depth- d quantum circuits $c = (c_d c_{d-1} \dots c_2 c_1)$ that begin with preparation of the qubits in their initial state, execute a sequence of logic operations (specified by their index in c), and terminate in measurement of all qubits. The model predicts the probability of observing outcome j to be:

$$P_j(c) = \langle \langle E_j | G_{c_N} G_{c_{N-1}} \dots G_{c_2} G_{c_1} | \rho \rangle \rangle \quad (4.21)$$

The parameters of this model are the elements of the vectors, dual vectors, and process matrices. The GST protocol estimates these parameters by first executing set of circuits $\{c^{(i)}\}$ on the quantum computer and gathering data $\{n_j^{(i)}\}$, the number of times measurement outcome j was observed when running circuit i . The parameters are chosen to maximize the likelihood function,

$$\mathcal{L} = \prod_{i,j} P_j(c^{(i)})^{n_j^{(i)}} \quad (4.22)$$

It turns out that the maximal experiment design described here is *massively* overcomplete. For instance, a maximal two-qubit long sequence GST experiment might require more than 30,000 circuits! Fortunately, many of these circuits provide redundant information so most can be excluded from the experiment design with almost no loss of estimation accuracy. Techniques such as fiducial pair reduction [Rud+23; Ost+23], can reduce the experimental overhead by more than an order of magnitude.

4.4 Error generators

The linear and long sequence GST designs above provide general experimental protocols to estimate the parameters of a gate set model. Up to this point, the model has been implicitly assumed to be a completely parameterized model on raw process matrices, density matrices, and POVM effects. However, the completely parameterized raw model is not a particularly intuitive representation; it is defined in terms of a Lie group, and the parameters have no clear physical interpretation. Instead of working with models on Lie groups, it is easier to define models in the Lie algebra, which provide a more direct description of physically meaningful quantities.

To move from a group theoretic error model parameterized in general CPTP maps to an algebraic model that support physically meaningful quantities, it is best to begin with a multiplicative error map. Define the error map $R(G_i)$ associated with gate G_i to be the product of the actual process preceded by the inverse of the target

$$R(G_i) = G_i \tilde{G}_i^{-1} \quad (4.23)$$

where $\tilde{G}_i$ is the target process. When the target process is unitary, as is almost always the case, then the inverse exists. Note the handedness of our definition, which assures that $R(G_i)\tilde{G}_i = G_i$. The error map on state preparations $R(\rho)$ may be implicitly defined by the relationship $|\rho\rangle\rangle = R(\rho)|\tilde{\rho}\rangle\rangle$, and the error map on measurements by $\langle\langle E_\mu| = \langle\langle \tilde{E}_\mu|R(E_\mu)$.

The multiplicative error map $R(G_i)$ is near the identity when the actual process is near the target operation, which means that we can directly adopt a convenient Lie algebraic formulation. As is standard in the GST literature, an *error generator* model (see § 3) parameterize a gate set in terms of small algebraic perturbations away from the target. The general error generator parametrization has the form

$$G_i = e^{\sum_j \theta_j^{(i)} L_j} \tilde{G}_i, \quad (4.24)$$

where $\tilde{G}_i$ is the ideal unitary action of gate i and L_j is a basis for the Lie semi-algebra of completely-positive, trace-preserving gate errors [Blu+22]. In this way,

$$R(G_i) = e^{\sum_j \theta_j^{(i)} L_j}. \quad (4.25)$$

In quantum computers, the gates generally well-approximate their target unitaries, so the error parameters of this representation $\{\theta_j^{(i)}\}$ are typically $\ll 1$.

In order to construct an explicit error generator model from tomographic information, we first pick a basis and write out a set elementary error generators or “features” that model different types of noise processes that may be present in a gate set. It is convenient to work in the Pauli transfer basis [Gre15], then elementary error generators may be defined as in [Blu+22] such that each elementary feature can be associated with a distinct physical noise process. In the Pauli transfer basis, a subset of basis elements describes stochastic noise, another subspace of features describe Hamiltonian (coherent) noise, and one may also include features corresponding to “correlated Pauli noise” and “active noise” [Blu+22].

In defining a processor’s error model, one may generally pick a restricted basis of features that span a subspace of possible noise processes. In this way, one may use a feature set that is only designed to capture the dominant noise processes afflicting a device. Reducing the number of features in a model is particularly useful for reducing the the complexity of the associated estimation algorithm. A convenient subspace is that spanned by Hamiltonian and stochastic noise, which is referred to as the “H+S” set of elementary error generators. This elementary decomposition captures coherent (unitary) noise and stochastic noise that shrinks the Bloch sphere along one of the cardinal axes. In practical engineering applications when theory or past experiments indicate strong effects by other noise processes not described within the H+S model, then it is simple to add in additional features to a model such that all dominant noise processes are captured by the model.

4.5 Gate set gauges

As we have seen, there are myriad models one may define to describe the errors in a quantum processor. Tomographic experiments, such as those employed in GST, provide empirical

information that may be exploited to optimize the parameters of a particular model such that the output predicts the observed dynamics. However, there exists other models with different parameters that would equally well describe the observed dynamics. For example, one is always free to rotate a quantum processor, redefining x, y, z for the gates and SPAM, and the output should be the same. It turns out that the existence of such *gauge symmetries* between models pose many challenges in practical applications, for example in streaming estimation, interpretation of performance metrics, and calibration. Resolving gauge issues has proven remarkably difficult. At the end of this section, I propose one potential route to solution that uses an elegant idea in representation theory.

Issues posed by gauge symmetry

Gauge symmetries pose problems in defining metrics for processor performance. Consider for example the channel fidelity between a noisy gate G and target (unitary) operation U

$$F(G) = \frac{1}{d^2} \text{Tr}(GU^{-1}). \quad (4.26)$$

One may observe that changing the basis of G without also changing U results in a different metric because in general

$$\frac{1}{d^2} \text{Tr}(MGM^{-1}U^{-1}) \neq \frac{1}{d^2} \text{Tr}(GU^{-1}), \quad (4.27)$$

which is the essential reason that fidelity and some other performance metrics are gauge dependent.

Gauge issues have also been observed in randomized benchmarking (RB) [MPF18]; here, the gate set metrics that one estimates based on RB decays have been shown to be gauge-dependent, and numerical simulations show that certain gate set models predict RB decays that in turn predict fidelities somewhat far from the fidelity of the data generating model [Pro+17].

Gauge symmetries also arise in Pauli noise learning [Che+23]. Here, one may observe that the data collected describes an equivalency class of possible Pauli noise models. In this case, the gauge is a discrete group of symmetries, as opposed to the continuous symmetry group of quantum gate sets. A proposed resolution [Che+23] involves identifying all learnable degrees of freedom and learning only the learnable parameters. In this one, one removes all unlearnable parameters (setting them to constants), which as I discuss below, provides an invariant frame or a gauge fixing.

Gauge issues are also particularly important in online estimation algorithms such as the Kalman filter. The Kalman filter tracks a parameter covariance estimate, which controls the magnitude of filter updates. When gauge symmetries exist, then some directions in parameter space do not affect observable outcomes, which causes convergence problems and leads to ill-defined covariances. Essentially, the Kalman filter will never learn anything about gauge directions, so some eigenvalues of the covariance matrix never decrease, preventing

convergence. One possible resolution is to employ approximately gauge-invariant models [MY23], which reduce the numbers of parameters in the model to remove gauge symmetries.

Gate set gauge freedom

A gauge transformation M maps a gate set $\mathcal{G}$ to a new gate set $\mathcal{G}'$ with a new set of parameters, but $\mathcal{G}$ and $\mathcal{G}'$ predict identical outcome probabilities for all possible circuits. This follows from the observation that the basis of a gate set model may be transformed by an arbitrary invertible $d^2 \times d^2$ matrix M without changing predictions. Acting on the gate set model with M as:

$$\langle\langle E_i^{(m)} | \mapsto \langle\langle E_i^{(m)} | M^{-1} , \quad (4.28)$$

$$|\rho\rangle\rangle \mapsto M |\rho\rangle\rangle , \quad (4.29)$$

$$G_i \mapsto M G_i M^{-1} , \quad (4.30)$$

results in a new model with equivalent predictions according, as

$$\langle\langle E_i^{(m)} | M^{-1} (M G M^{-1}) M |\rho\rangle\rangle = \langle\langle E_i^{(m)} | G |\rho\rangle\rangle . \quad (4.31)$$

Let me now define some notation. Recall that a gate set model $\mathcal{W}$ contains the states, gates, and measurements $\{\mathcal{S}, \mathcal{G}, \mathcal{M}\}$, resp. Let us imagine a global parametrization θ of the gate set model. The global parametrization θ contains all the parameters of all the individual gates, preparations, and measurements in the model. Say that each gate G_j has parameters $\theta^{(j)}$ and similarly for the SPAM. Then, the global parametrization θ may be represented as a stacked vector of local gate set parametrizations

$$\theta = (\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(N)})^T. \quad (4.32)$$

for N the total number of local gate and SPAM parameterizations. This corresponds to a direct sum of parameter vector spaces. The vector space $\mathcal{V}$ that describes the global parametrization θ is a direct sum of local vector spaces $V^{(j)}$ as

$$\mathcal{V} = V^{(1)} \oplus V^{(2)} \oplus \dots \oplus V^{(N)}. \quad (4.33)$$

A gauge transformation M can be represented as a basis transformation of the gate set model. In this way, define the gauge transformation map Φ_M that applies M as in Eq. 4.28. As a map on the global parametrization, Φ_M acts on local vector spaces

$$\Phi_M : \mathcal{V} \rightarrow \mathcal{V}. \quad (4.34)$$

It is possible to represent Φ_M as a linear operator on a global parameterization $\theta \in \mathcal{V}$. The action of Φ_M on a gate parametrization may be represented as local application of the adjoint Ad_M on each gate

$$M G(\theta^{(j)}) M^{-1} = G(\text{Ad}_M(\theta^{(j)})). \quad (4.35)$$

where Ad_M is an isomorphism of the adjoint action on the parameters after a pull through the model. Similar linear operators may be defined for the SPAM, but I will focus on the gates for now. The action of the gauge transformation on the vector (considering just gates) may be represented as a block-diagonal rotation

$$\Phi_M(\theta) = \begin{bmatrix} \text{Ad}_M & 0 & \dots & 0 \\ 0 & \text{Ad}_M & \dots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \dots & \text{Ad}_M \end{bmatrix} \quad (4.36)$$

acting on each local component of the parameterization. One should observe that, while it has a global action in a very large parameter space, the gauge transformation is defined only with local degrees of freedom, having a number of parameters equal to the size of the basis transformation matrix M .

Previously proposed solutions to gauge issues

One standard approach to mitigate performance metrics' gauge-dependence is to employ gauge-fixing [Nie+21]. This is most commonly done via “gauge optimization,” which varies over all possible gauge transformations to find a gauge that minimizes the deviation between a (noisy) gate set model and an ideal “target” model. The metric of deviation is somewhat arbitrary, and weighted Frobenius distance is commonly used for convenience. The need for gauge-fixing can be avoided by using strictly gauge-invariant metrics of error. Gauge transformations do not change the eigenvalues of a gate's transfer matrix, so any error metric that depends only on a gate's spectrum is gauge-invariant.

Some work has explored alternative model constructions that circumvent the gauge problem. For example, [Di +20] employs a representation of gate sets in terms of the probabilities of linear inversion gate set tomography. However, the parameters this model are LGST probabilities, which are not particularly convenient for calibration or physical intuition. Nevertheless, this representation completely avoids gauge issues because every parameter in the representation is explicitly gauge-invariant.

Another approach [MY23; NYB22] makes use of “first-order gauge invariant” (FOGI) parameterizations that are invariant under small gauge transformations. Essentially, a FOGI model constructs a basis for parameter combinations that are invariant at first order¹ under a gauge transformation. The QR decomposition plays a key role in constructing a convenient orthonormal basis for the invariant degrees of freedom. After identifying a FOGI model, the resulting model contains a set of gauge and non-gauge parameters. The gauge parameters control translations in gauge space, while the non-gauge parameters have actual effects on model predictions. The reader should observe how this approach fits with a common theme of reducing the number of model parameters to resolve gauge issues.

¹Here, first order means a linear approximation.

Invariant frames

One avenue for resolution of gauge issues may be found in *Cartan's moving frame technique* [Olv99]. The moving frame technique is a deep idea in representation theory [Cle17] that has practical utility in classical estimation theory [BMR06]. The moving frame technique is foundational to the method of Kalman filtering on Lie groups and symmetry preserving observers [BB18]. I believe this to be a very promising route of investigation. Here, I provide preliminary results.

The trick of invariant frames is to observe that one can eliminate gauge degrees of freedom by choosing to work in a gauge such that a certain subset of parameters are always set to constant values regardless of experimental observations. Clearly, not all the parameters can be set to constants, and identifying which parameters can *always* be set to constant values, regardless of experimental data, is one of the chief challenges in constructing an invariant frame.

Essentially, the process of invariant frame construction is an iterative gauge fixing procedure. One considers all possible gauge transformations and then iteratively isolates gauge-dependent model parameters, setting them to constant values and removing them as parameters. Then, one defines a new model and repeat with another free parameter and associated gauge symmetry. Each parameter that is removed from the model also removes a gauge degree of freedom, because now the action of that gauge transformation would change the value of the parameter that was set to a constant. The iterative process is repeated until all gauge degrees of freedom are removed.

My description of invariant frame construction here is unfortunately very brief, though well founded in existing mathematical analysis [Olv99]. At the time of this work, I have not fully resolved this approach for general CPTP models and multi-qutbit gate sets. Instead, I conclude my discussion of invariant frames by providing an example construction of an invariant frame for a single-qubit unitary gate set model.

Consider a single-qubit unitary gate set model. The invariant frame may be identified by selecting two reference gates, constructing an invariant frame with respect to the two reference gates, and then using that frame for all the other possible gates in the model. The SPAM elements do not appear in the identification of the invariant frame – though the gauge that is identified should, of course, also be applied to the SPAM.

Thus, select two reference unitary gates from the single-qubit gate set model. These can be represented as unitary rotations $R^{(1)}$ and $R^{(2)}$ with Pauli parametrization $\theta^{(1)}$ and $\theta^{(2)}$. Each gate has three free parameters, and there are three gauge parameters. To eliminate all gauge degrees of freedom, begin by diagonalizing one of the gates to provide the first reference. Then, the first parametrization, which was a general 3-element vector becomes fixed to a vector with one free component and two fixed components that may be assumed 0

$$(x, y, z) \rightarrow (x', 0, 0). \quad (4.37)$$

This fixes two gauge degrees of freedom because now any gauge transformation about an axis other than x will result in a model that does not have 0's in the locations identified

in the invariant frame. There is then one remaining gauge degree of freedom corresponding to rotations about the axis that was fixed². Now, consider the second rotation $R^{(2)}$. Use the remaining gauge degree of freedom to eliminate either of the components that was not fixed in the first reference. For example choosing to eliminate y , identify a rotation in the remaining directions of gauge symmetry $S_X(\phi)$ such that

$$S_X(\phi)G(x, y, z)S_X^{-1}(\phi) = (x', 0, z'), \quad (4.38)$$

which removes the y component from the second reference gate. This process results in a gate set that is promised to have 0's in the locations identified above. No remaining unitary gauge degrees of freedom exist.

The process above maps a general model with gauge freedom to a distinguished model in a particular invariant frame. The constants that were chosen for the gauge dof. determine the particular invariant frame. Different invariant frames may be constructed through alternative choices of constants.

It is important to realize that one does not have to go through the process of gauge fixing with every given gate set model. Once the process has been fully carried out for a particular gate set model, then one can construct one's parametrization always in the identified invariant frame, with free parameters for the values that were not set to constants. In this way, one can work in the invariant frame when doing estimation and avoid gauge issues.

Finally, observe that given two models that are not in the same invariant frame, one can go through a gauge fixing process to find a representation of both models with the same fixed reference gauge so that metrics like fidelity can be consistently compared. In this way, invariant frames may help resolve the gauge dependence of some gate set performance metrics.

4.6 Outstanding problems

I conclude this chapter's discussion of GST with a summary of open questions.

Reduced model GST. GST experiment designs are very expensive to collect. Some work has developed elegant fiducial selection techniques [Ost+23; Rud+23] that reduce the number of circuits substantially. Another approach is based in reducing the number of parameters in the model. This can be done by physically motivated models or by model reduction techniques on real data collected from actual devices. In this way, GST becomes a guiding paradigm for efficient noise model identification. Yet another approach involves adaptive phase estimation techniques § 5 in the context of GST to produce a “gate set phase tomography” that estimates coherent parameters to high precision and only obtains partial information about stochastic noise.

²It should be clear that $U_X(\theta)G(x, 0, 0)U_X^{-1}(\theta) = G(x, 0, 0)$

Efficient classical estimation. GST experiment designs are very expensive to estimate classically, as they involve a difficult likelihood optimization. It has been observed that using gauge free “FOGI models” [NYB22] reduces the estimation time by removing ridges in optimization landscapes. Other work [MY23] introduced an efficient classical Kalman filter for this purpose. Nevertheless, this problem remains daunting when scaling to very large systems and further work is needed.

Gage independent performance metrics. As we say, many common metrics of gate sets are gauge variant. Developing gauge-invariant metrics is an important outstanding question. This problem has been partially solved by considering eigenvalues of the estimated process matrices. Another approach may be to always report fidelity in a standard invariant frame.

Non-Markovian models. GST fits a Markovian model of a quantum gate set, and experimental evidence has indicated that physical processors are only approximately Markovian. Developing good non-Markovian models of quantum devices and associated estimation protocols is an important outstanding problem.

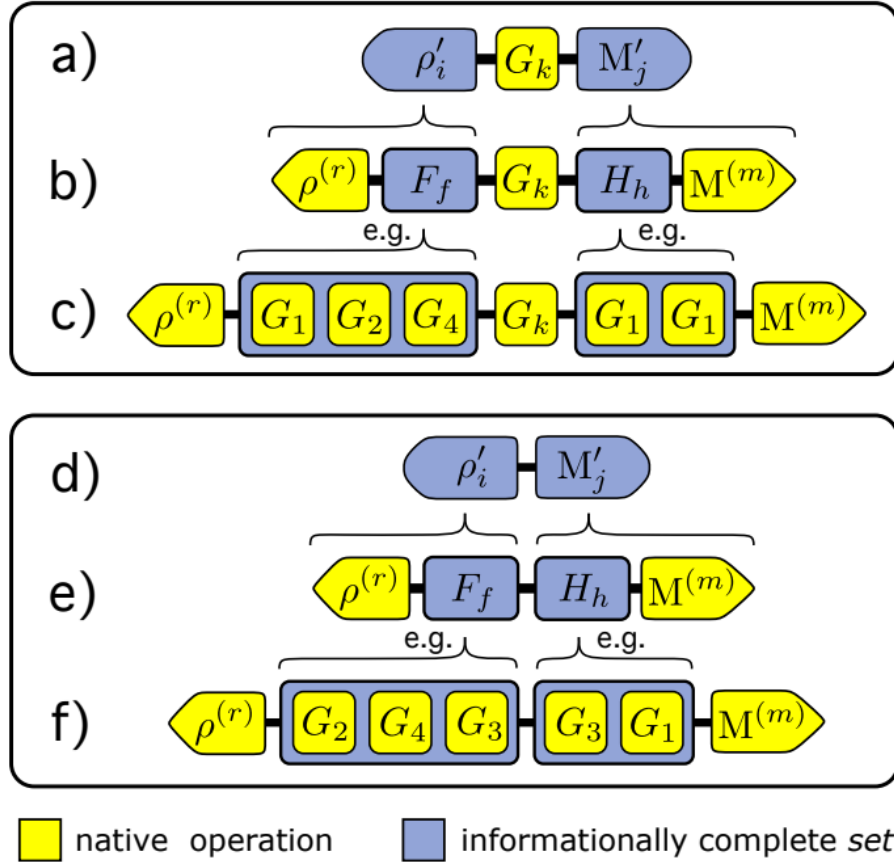


Figure 4.2: *Linear Gate Set Tomography*. Structures of the two types of circuits required by the LGST algorithm. **Upper panel:** each native gate, G_k , is sandwiched between the elements of informationally complete sets of *effective state preparations*, $\{\rho'_i\}$, and of *effective measurements*, $\{M'_j\}$. These are the same circuits that QPT requires to characterize G_k . (a) Shows these circuits in their simplest form, with each informationally complete set displayed as a unit. (b) Depicts the common case when the set of effective preparations (measurements) is implemented by following (preceding) a single native state preparation (measurement) operation with a *fiducial circuit* F_f (H_h). (c) Exemplifies that the fiducial circuits are composed of native gates G_i and gives the circuit entirely in terms of native operations. **Lower panel:** because LGST does not assume knowledge of ρ'_i and M'_j , it requires circuits that sandwich nothing between pairs of fiducials in order to be self-calibrating. The circuit diagrams in (d), (e), and (f) parallel those in (a), (b), and (c). LGST also requires the circuits that perform state (measurement) tomography on ρ (M), but these are not explicitly shown. They are similar to (d) – (f) (replacing ρ' with ρ or M' with M), and are actually included as a subset of these circuits when the gate set contains only a single native state preparation (measurement) and one of the preparation (measurement) fiducial circuits is the empty (do-nothing) circuit. (Figure and caption reproduced with permission from [Nie+21].)

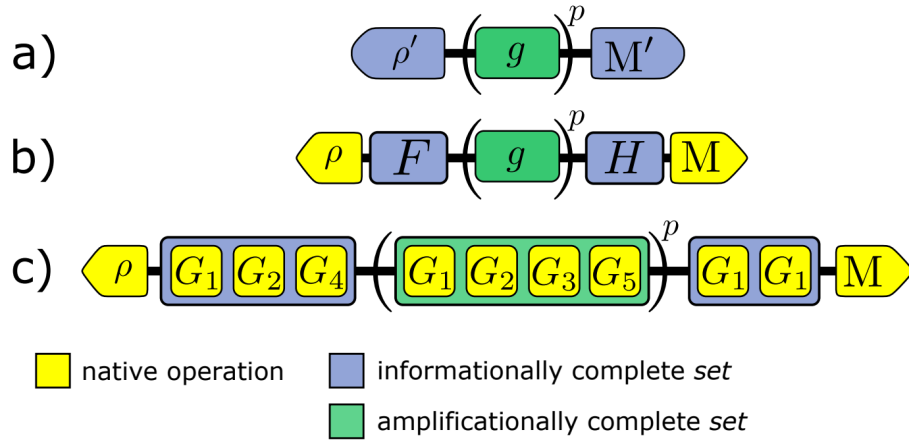


Figure 4.3: *Long Sequence Gate Set Tomography*. The structure of circuits in the standard LSGST experiment design, shown in increasing detail. (a) Each GST circuit consists of an effective state preparation ρ' followed by a *germ* circuit g repeated p times, followed by an effective measurement $M' = \{E'_i\}$. (b) Effective preparations are often implemented by a native state preparation ρ followed by a *preparation fiducial circuit* F , and similarly effective measurements are often implemented by *measurement fiducial circuit* H followed by a native measurement M . (c) Writing the fiducials and germ in terms of the native gate operations reveals how the native operations of a gate set compose to form a GST circuit. (Figure and caption reproduced with permission from [Nie+21].)

Chapter 5

Phase estimation for calibration

While gate set tomography (GST) provides a complete tomographic reconstruction of a quantum processor’s gate set, it comes with a very large overhead that is prohibitive in practical calibration and control problems. Phase estimation provides much of the benefit of GST without the large overhead. Instead of completely reconstructing a processor’s gate set, phase estimation estimates only a few parameters of interest. Robust phase estimation (RPE) is *robust* to out-of-model effects, including SPAM errors and decoherence. The efficiency and robustness of RPE makes it an attractive primitive for use in closed-loop calibration frameworks, discussed in §III of this work.

Algorithms for performing phase estimation can generally be divided into two classes: ancilla-assisted and ancilla-free. Ancilla-assisted protocols infer the phases from measurements on an ancilla register that has been entangled with conditional rotations from a data register, and these types of protocols are often used as primitives in large quantum algorithms [Mar+21]. Ancilla-free protocols achieve high-precision estimates by repeated application of a unitary operator without entanglement to additional systems, and these type of protocols are often used in metrological or tomographic applications. RPE and its variants discussed in this chapter are ancilla-free protocols. Ancilla-free protocols are most appropriate for tuning noisy quantum computers that may be moderately far from the ideal operations.

In its basic form, RPE infers relative phase differences of a unitary representation of a quantum gate. Equivalently, RPE infers the imaginary part of the eigenvalues of a process matrix. I will focus on the unitary interpretation. Because of global $U(1)$ symmetry, only relative phases are observable in quantum devices.

RPE was first introduced in [KLY15] as a generalization of the Heisenberg-limited phase estimation protocol of [Hig+07] that adds explicit robustness to incoherent noise and to errors in state preparation and measurement. RPE was originally proposed to estimate the phase (i.e., rotation angle) of a single-qubit gate, and has since been expanded to multi-qubit gates [Rus+21a] [Rud+25]. I show an extension to a multi-qutrit gate here.

In this chapter, I review the theory of single- and multi-qubit RPE, discuss the robustness of these protocols, and introduce decoupled and twirled RPE, which are new techniques that measure a subset of coherent error parameters in multi-qubit unitaries.

5.1 The basic primitive

The basic phase estimation primitive is a single-parameter parameter protocol that measures a particular relative phase in a unitary. The procedure measures a pair of circuit sequences to estimate the cosine and sine (or in-phase and quadrature) components of a complex signal that oscillates with frequency equal to a relative phase difference. This basic primitive can be applied in various multi-parameter contexts, discussed in subsequent sections.

Explicitly, consider the spectral decomposition of a general unitary model of a quantum gate

$$U = \sum_k e^{i\phi_j} |\psi_j\rangle\langle\psi_j| \quad (5.1)$$

to measures the relative phase $\varphi_{(a,b)} := \phi_b - \phi_a$, where ϕ_a, ϕ_b are eigen-phases that correspond to eigen-states $|\psi_a\rangle, |\psi_b\rangle$, prepare an in-phase superposition

$$|I_{(a,b)}\rangle = \frac{|\psi_a\rangle + |\psi_b\rangle}{\sqrt{2}} \quad (5.2)$$

and a quadrature superposition

$$|Q_{(a,b)}\rangle = \frac{|\psi_a\rangle + i|\psi_b\rangle}{\sqrt{2}}. \quad (5.3)$$

Next, define measurement probabilities

$$p^I_{(a,b)}(k) = |\langle I_{(a,b)} | U^k | I_{(a,b)} \rangle|^2 \quad (5.4)$$

and

$$p^Q_{(a,b)}(k) = |\langle I_{(a,b)} | U^k | Q_{(a,b)} \rangle|^2. \quad (5.5)$$

It follows that

$$\begin{aligned} |\langle I_{(a,b)} | U^k | I_{(a,b)} \rangle|^2 &= \frac{1}{4} |e^{i\phi_a} + e^{i\phi_b}|^2 \\ &= \frac{1 + \cos(\pm(\phi_a - \phi_b))}{2} \end{aligned} \quad (5.6)$$

where the sign $\pm$ may be chosen arbitrarily. Similarly

$$\begin{aligned} |\langle I_{(a,b)} | U^k | Q_{(a,b)} \rangle|^2 &= \frac{1}{4} |e^{i\phi_a} + ie^{i\phi_b}|^2 \\ &= \frac{2 - i(e^{i(\phi_a - \phi_b)} + e^{-i(\phi_a - \phi_b)})}{4} \\ &= \frac{1 + \sin(\phi_a - \phi_b)}{2}, \end{aligned} \quad (5.7)$$

so we see that the sign of the relative phase is fixed by the choice of quadrature.

To identify a complex exponential form, we convert to in-phase and quadrature expectations

$$\langle I_{(a,b)}(k) \rangle = 2p^I_{(a,b)}(k) - 1 \quad (5.8)$$

and

$$\langle Q_{(a,b)}(k) \rangle = 2p^Q_{(a,b)}(k) - 1. \quad (5.9)$$

Now, defining an RPE signal

$$s_{(a,b)}(k) = \langle I_{(a,b)}(k) \rangle - i \langle Q_{(a,b)}(k) \rangle \quad (5.10)$$

it follows

$$s_{(a,b)}(k) = e^{-k(\delta_{(a,b)} + i\varphi_{(a,b)})} + v_k \quad (5.11)$$

where $\varphi_{(a,b)}$ is the desired eigen-phase difference, $\delta_{(a,b)}$ is an exponential decay constant (equal to 0 for a unitary), and v_k represents shot noise as a random variable. Ignoring shot noise and decay, the logarithm of the measured signal is

$$\log(s_{(a,b)}) = -k(\delta_{(a,b)} + i\varphi_{(a,b)}) + 2\pi i K(-ik\varphi_{(a,b)}) \quad (5.12)$$

where $K(-ik\varphi_{(a,b)})$ is the *unwinding number* [Rob96]. The unwinding number represents phase wrapping in the signal, which is unobservable in experiment. To avoid estimation errors from unwinding numbers, one may sample the signal $s(k)$ for multiple depths k .

A deterministic strategy [Hig+07] to ensure that the branch cuts can be resolved is to pick depths according to a base-2 logarithmic interval, as powers of 2 (1, 2, 4, 8, 16, ...), then pick the phase at each depth that is closest to the phase at the prior generation. One constructs a sequence of estimates $\hat{\varphi}_{(a,b)}^{(j)}$ for a sequence of depths $(2^j)_j$. At each depth, the estimate is

$$\hat{\varphi}_{(a,b)}^{(j)} = \frac{1}{2^j} (\text{Im}(\log(s_{(a,b)}(2^j))) + 2\pi n^{(j)}) \quad (5.13)$$

where the log is taken on the principle branch and $n^{(j)}$ is an unwinding number that selects the proper branch of the logarithm at depth 2^j . One usually assumes that $n^{(0)} = 0$, but the story is the same for other choices. After fixing $n^{(0)}$, the unwinding number at each larger depth is taken as the solution to the minimization problem

$$\text{argmin}_{n^{(j)}} \left| \hat{\varphi}_{(a,b)}^{(j-1)} - \frac{1}{2^j} (\text{Im}(\log(s_{(a,b)}(2^j))) + 2\pi n^{(j)}) \right| \quad (5.14)$$

which, restricted to integer $n^{(j)}$ is

$$n^{(j)} = \left\lfloor \frac{2^j \hat{\varphi}_{(a,b)}^{(j-1)} - \text{Im}(\log(s_{(a,b)}(2^j)))}{2\pi} \right\rfloor. \quad (5.15)$$

Here, I have derived an RPE estimator using the complex logarithm. It is more common in the literature to see a presentation with the two argument arctangent function arctan2 .

The details are essentially the same and result from the fact that the imaginary part of the complex logarithm is equivalent to an arctangent. To see this, consider a complex number $z = x + iy$ and write a polar form of the complex logarithm

$$\log(z) = \log |z| + i \arg(z) \quad (5.16)$$

then observe that

$$\arg(z) = \arctan2(y, x). \quad (5.17)$$

The two argument arctangent uses both x and y or the real and imaginary parts to resolve sign inconsistencies that would arise from $\arctan(y/x)$.

For multi-qubit Hilbert spaces of dimension > 2 , it will generally be the case that errors in state preparation, measurement, or in the gate action itself can push state population outside the $\{|\psi_a\rangle, |\psi_b\rangle\}$ subspace. Observation of such “leakage” states will not inform the phase estimate. It may therefore be desirable to post-select the measurement outcomes when constructing an empirical estimate of the expectation values. For instance, when measuring the in-phase operator, define $N_+(k)$ as the number of times the $|I_{(a,b)}\rangle$ state is observed at depth k and $N_-(k)$ as the number of times its complement $(|\psi_a\rangle - |\psi_b\rangle)/\sqrt{2}$ is observed. The post-selected expectation value is simply:

$$\langle I_{(a,b)}(k) \rangle_p = \frac{N_+ - N_-}{N_+ + N_-}. \quad (5.18)$$

and similarly for the quadrature. Such discussion highlights the differences between single-qubit and multi-qubit phase estimation: a multi-qubit phase estimation protocol may be understood as a 2-state embedding in a higher dimensional Hilbert space. Much of the robustness analysis of RPE was derived in the context of single-qubit Hilbert spaces [KLY15], and extending this to explicit analysis of larger Hilbert spaces is an outstanding research question.

Algorithm

I now summarize the RPE program into a concise algorithm, Tab. 5.1 using consistency checks introduced in §5.1. In this feed-forward algorithm, the estimate at each prior generation is fed into consistency checks, which determine the termination condition in a on-line fashion. A deterministic algorithm that does not require real-time logic may be defined by fixing the interval beforehand and applying the consistency checks in post-processing to determine the maximally trusted generation.

Multi-parameter RPE

In single-parameter RPE, a basic protocol is called to estimate a particular relative phase $\varphi_{(a,b)}$ determined by a particular choice of eigenstates ϕ_a, ϕ_b of the target unitary operator. For single-qubit systems, this is the whole story; there is only one unique relative phase

Single-parameter RPE

Require: Unitary eigenbasis pair $(|\psi_a\rangle, |\psi_b\rangle)$ of gate U

Measure $\langle I_{(a,b)}(1) \rangle$ and $\langle Q_{(a,b)}(1) \rangle$

Estimate $\hat{\varphi}_{(a,b)}(1)$

$k = 2$

while Previous consistency check passed **do**

Measure $\langle I_{(a,b)}(k) \rangle$ and $\langle I_{(a,b)}(k) \rangle$

Estimate $\hat{\varphi}_{(a,b)}(k)$

Apply consistency check

$k \rightarrow 2k$

end while

Return $\hat{\varphi}_{(a,b)}(k)$ and uncertainty

Table 5.1: Single-parameter feed-forward RPE estimation.

for an element of $SU(2)$. However, in the multi-qubit or qudit setting, there are more phases that can be estimated, so we require multiple experiments to fully characterize the spectrum. These experiments may each individually be understood as a single-parameter RPE experiment generated with different pairs of eigenstates.

For a unitary in $SU(N)$ one may form at most $N - 1$ linearly independent relative phases. This follows from the fact that there are $N - 1$ unique eigenvalues for an element of $SU(N)$, so only $N - 1$ independent differences may be constructed. Therefore complete phase estimation of a N -dimensional unitary requires $N - 1$ RPE experiments. How to choose which $N - 1$ relative phases to measure (out of a possible $\binom{N}{2}$ choices) depends on which in-phase and quadrature states are easy to prepare and measure, which will be a function of the gate in question.

Robustness of the protocol

The eponymous *robustness* of RPE ensures that the phase estimations θ are accurate even in the presence of out-of-model errors, including SPAM errors, decoherence in the gate, and shot noise. Generally, the protocol allows for additive errors $\delta^I(k)$ and $\delta^Q(k)$ to the in-phase and quadrature probabilities $p^I(k)$ and $p^Q(k)$ of Eqs. 5.4 and 5.5

$$\hat{p}^I(k) = \frac{1 + \cos(k\varphi)}{2} + \delta^I(k) \quad (5.19)$$

and similarly for the quadrature. It was be shown¹ [Rus+21b] that if $|\delta(k)| < \sqrt{3/32}$ for all $k < k_{\max}$ for both in-phase and quadrature series, then RPE will successfully produce an estimate of the true θ with an error that is bounded according to $\pi/k_{\max}$ where $k_{\max}$ was depth of the longest probe depth.

Uncertainty

The maximum error bound $\pi/k_{\max}$ provides a rather coarse estimate of RPE’s uncertainty and does not account for the reduction in uncertainty that more shots provides. For the purposes of setting optimizer thresholds that terminate the process after a sufficient accuracy is achieved, it may be useful to use a slightly tighter bound on the expected uncertainty $\sigma(\theta)$ of an RPE estimate of parameter θ that also takes into account the number of shots:

$$\sigma(\theta) = \frac{\pi}{k_{\max}\sqrt{N}} \quad (5.20)$$

where N is the number of shots used in the deepest circuit. This error bound provides a tighter estimate on the uncertainty of RPE experiments, and I have verified the accuracy of this uncertainty estimator through numerical simulations. Theoretical arguments [Gor+19] indicate that this uncertainty bound is probably the best one can do.

Consistency checking

The error bound $\sqrt{3/32} \approx 30.6\%$ is a rather generous bound on the worst-case total-variation distance between true and empirical estimates that guarantee a reliable estimate of the true RPE phase within the worst-case error bound $\pi/l_{\max}$. However, in order for this robustness to be of practical utility, it is necessary to have an assurance that this bound is satisfied experimentally. A suite of “consistency checks” [Rus+21b] have been developed to check whether experimental estimates of in-phase and quadrature probabilities are consistent with the prior generations. This approach turns the theoretically robustness bounds into practically useful metrics: a consistency check flags the depth at which decoherence or other error sources spoil the RPE estimator. A trusted estimate is then constructed by restricting to only the consistent depths – the depths up to the one that was flagged by the check.

Inferring Pauli models

The RPE protocol identifies a list of relative phases of an experimental implementation of a target unitary. For the purposes of calibration, it is often more useful to work with an explicit model of a unitary implementation of a gate in terms of Pauli coefficients, which provide a basis that more reflects the physics of a gate’s operation. The step from relative

¹In fact, the original publication [KLY15] included a slightly coarser error bound $\sqrt{1/8}$ that was later corrected and acknowledged in an errata.

phases to Pauli coefficients involves solving a system of equations. It is useful to note that, because all the information is contained in the relative phases, it is not strictly necessary to identify a Pauli model of gate operation; cost functions for control laws could be defined directly on the relative phases themselves.

One should observe that when all the operators $\mathbf{P}_j$ in a Pauli model commute, then the eigenvalues are simple linear functions of the coefficients. In this case, the coefficients can be identified by solving a system of linear equations relating measured relative phases and coefficient values. An explicit example is provided in the next section. However, in the case of non-commutative models, the eigenvalues of the generator will generally be complicated, non-linear functions of the model coefficients. In such cases, one usually invokes a small value approximation and linearizes the non-linear system of equations, see [KLY15] for an example. An alternative approach is to employ decoupled and twirling strategies, discussed at the end of this chapter.

Cartan subalgebras

As we saw, an RPE experiment measures $N - 1$ relative phases corresponding to a noisy implementation of a target unitary in $SU(N)$. There is an elegant interpretation of these measured phases as weights of a Cartan subalgebra. Basically, this follows from the observation that RPE sequences have a twirling effect and, in the limit of small errors, only the unitary errors that commute with the target operation of the gate appear in the relative phases. In this way, each RPE sequence amplifies a commuting subalgebra. These deep connections between RPE and representation theory merit further investigation.

5.2 Controlled-Z RPE

I now demonstrate how the phase estimation protocol can be employed to estimate a simple, commutative error model of a controlled-Z (CZ) operation for use in efficient calibration loops [Rud+25]. The CZ gate has a target unitary operation $\mathbf{U}_{CZ} = \mathbf{I} \otimes |0\rangle\langle 0| + \mathbf{Z} \otimes |1\rangle\langle 1|$, or written in an equivalent exponential notation

$$\mathbf{U}_{CZ} = \exp \left(-\frac{i}{2} \left(-\frac{\pi}{2} \mathbf{I} \mathbf{I} + \frac{\pi}{2} \mathbf{P}_{IZ} + \frac{\pi}{2} \mathbf{P}_{ZI} - \frac{\pi}{2} \mathbf{P}_{ZZ} \right) \right). \quad (5.21)$$

One may assume a simple model with free parameters for the Pauli coefficients of $\mathbf{P}_{IZ}$, $\mathbf{P}_{ZI}$, and $\mathbf{P}_{ZZ}$. This model is physically motivated by the off-resonant nature of the gate implementation in a superconducting device, but it only approximately captures all possible coherent errors, highlighting a downside of simple commutative approximations. Explicitly, the model is

$$\tilde{\mathbf{U}}_{CZ} = \exp \left(-\frac{i}{2} (\theta_{IZ} \mathbf{P}_{IZ} + \theta_{ZI} \mathbf{P}_{ZI} + \theta_{ZZ} \mathbf{P}_{ZZ}) \right), \quad (5.22)$$

Eigenstate	Eigenphase
$ 00\rangle$	$\theta_{ZI} + \theta_{IZ} + \theta_{ZZ}$
$ 01\rangle$	$\theta_{ZI} - \theta_{IZ} - \theta_{ZZ}$
$ 10\rangle$	$-\theta_{ZI} + \theta_{IZ} - \theta_{ZZ}$
$ 11\rangle$	$-\theta_{ZI} - \theta_{IZ} + \theta_{ZZ}$

Table 5.2: Eigenstates and eigenphases for the model CZ operation, Eq. 5.22. The II phase was set to 0 using $U(1)$ phase freedom.

State prep.	Measurements	Relative phase
$ 0+\rangle$	$\langle 0+ $ and $\langle 0+_y $	$\phi_{00,01} = \theta_{IZ} + \theta_{ZZ}$
$ 1+\rangle$	$\langle 1+ $ and $\langle 1+_y $	$\phi_{10,11} = \theta_{IZ} - \theta_{ZZ}$
$ +1\rangle$	$\langle +1 $ and $\langle +_y1 $	$\phi_{01,11} = \theta_{ZI} - \theta_{ZZ}$

Table 5.3: The state preparations and measurement (SPAM) pairs used in phase identification for the CZ gate. Each SPAM pair surrounds logarithmic spaced application of the CZ unitary $\mathbf{U}_{CZ}^{2^k}$. For a given state preparation there are two measurements that estimate the cosine and sine parts of a complex phase listed in the last column.

where I ignore the physically unobservable coefficient on $\mathbf{P}_{II}$. θ_{IZ} and θ_{ZI} have a target value of $\pi/2$, and θ_{ZZ} has a target value of $-\pi/2$

One starts by identifying the eigenstates and eigenphases of the target unitary for the CZ gate, listed in Table 5.2. Three RPE experiments then measure three independent relative phases. Each relative phase is a particular linear combination of the model parameters. The details are given in Tab. 5.3. Running RPE analysis on the experimental data collected from the three experiments results in estimates three relative phases $\phi_{00,01}, \phi_{10,11}, \phi_{01,11}$, which are related to the model coefficients through a system of equations

$$\begin{bmatrix} \phi_{00,01} \\ \phi_{10,11} \\ \phi_{01,11} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} \theta_{IZ} \\ \theta_{ZI} \\ \theta_{ZZ} \end{bmatrix}. \quad (5.23)$$

Inversion of the linear system provides an estimate of the model parameters.

This RPE protocol was used in [Rud+25] to calibrate a physical controlled-Z gate on superconducting hardware. We collected before- and after-calibration metrics of gate performance using GST. In particular, we measured the during-gate Hamiltonian error generator rates, shown in Fig. 5.1(a). The during-gate error generator is the logarithm of the measured process matrix minus the logarithm of the target:

$$L = \ln P - \ln \tilde{P} \quad (5.24)$$

where P is the measured process matrix and $\tilde{P}$ is the target process. In this way, $P = \exp(L + \tilde{L})$ where $\tilde{L}$ is the target error generator $\ln \tilde{P}$. We observed out-of-model errors

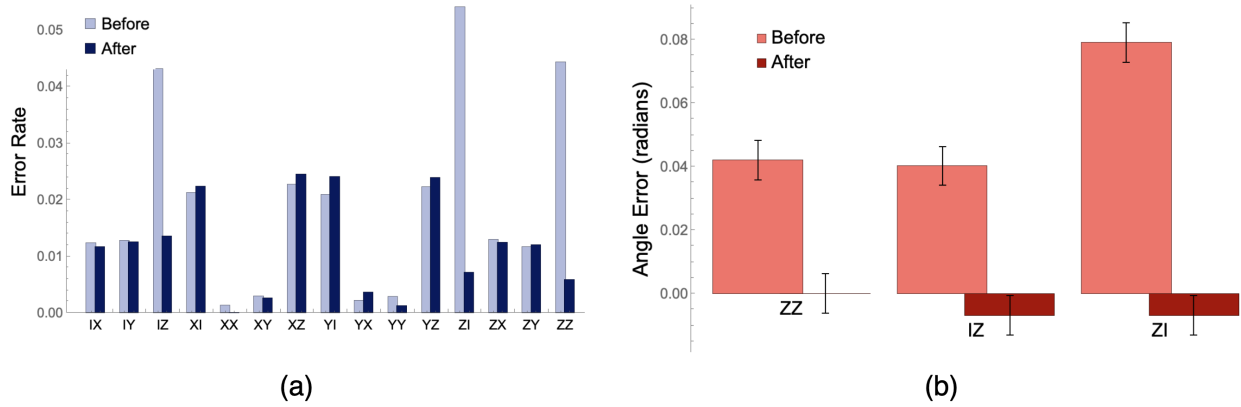


Figure 5.1: *Metrics before and after calibration.* Panel (a) shows the during-gate Hamiltonian error rates and panel (b) shows the angle error. The Hamiltonian error was measured with GST and indicate that significant resonant and cross-resonant terms persist in the drive Hamiltonian after calibration, which motivates future work to efficiently estimate and calibrate such small non-commuting terms. The angle error was measured with RPE, and the error bars correspond to the standard root-mean-square (RMS) error regions of an RPE estimate ($\pi/(2d)$ for RPE of depth d). While RPE does estimate remaining errors in the local IZ- and ZI-phases after calibration, the magnitude of the errors falls within the RMS bounds of RPE.

the persist after calibration. Because our RPE experiment does not measure these terms, they are not addressed by the current calibration and contribute to the overall diamond norm error. This fact indicates that beyond-commutative model estimation techniques are required, which motivates my derivation of decoupled and twirled RPE.

5.3 Qutrit RPE

RPE can also be deployed to estimate a generalized Pauli model in a qutrit system. Both the single- and two-qutrit gate relative phases can be estimated with a straightforward generalization of the qubit protocols, and model coefficients can be inferred and optimized. A derivation of the experiment design is provided below.

Single-qutrit models

The target single-qutrit gate set is $\{X^{01}, X^{12}, Z^{01}, Z^{12}\}$, which have matrix representations

$$X^{01} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -i & 0 \\ -i & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad X^{12} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -i \\ 0 & -i & 1 \end{bmatrix},$$

$$Z^{01}(\phi) = \begin{bmatrix} e^{\frac{i}{2}\phi} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad Z^{12}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & e^{-\frac{i}{2}\phi} \end{bmatrix}.$$

The usual Gell-man decomposition of $SU(3)$ leads us to consider a model for X^{01} with two parameters θ, γ

$$X^{01}(\theta, \gamma) = \exp\left(-\frac{i}{2}(\theta \mathbf{X}^{01} + \gamma \mathbf{\Gamma}^{01})\right)$$

where the generators are defined

$$\mathbf{X}^{01} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \mathbf{\Gamma}^{01} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{bmatrix}.$$

The model for X^{12} is then obtained by a permutation and relabeling of basis elements

$$X^{12}(\theta, \gamma) = \exp\left(-\frac{i}{2}(\theta \mathbf{X}^{12} + \gamma \mathbf{\Gamma}^{12})\right)$$

where

$$\mathbf{X}^{12} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \mathbf{\Gamma}^{12} = \begin{bmatrix} -2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Single qutrit experiment design

Measuring θ on X^{01} : To arrive at the experiment design to measure θ on X^{01} , start by observing that

$$[\mathbf{X}^{01}, \mathbf{\Gamma}^{01}] = 0, \tag{5.25}$$

which implies that these operators may be simultaneously diagonalized. The simultaneous eigenvectors of $\mathbf{X}^{01}$ and $\mathbf{\Gamma}^{01}$ are

$$|+_{01}\rangle \equiv (1/\sqrt{21}/\sqrt{2}, 0)^T \tag{5.26}$$

$$|-_{01}\rangle \equiv (1/2, -1/2, 0)^T \tag{5.27}$$

$$|2\rangle \equiv (0, 0, 1)^T. \tag{5.28}$$

Now, observe that $\mathbf{\Gamma}^{01}$ acts as the identity on $|+_{01}\rangle, |-_{01}\rangle$. Hence, one may use this pair of eigenvectors to form an RPE experiment to measure θ . Taking

$$|0\rangle = \frac{|+_{01}\rangle + |-_{01}\rangle}{\sqrt{2}} \tag{5.29}$$

observe

$$(X^{01}(\theta, \gamma))^k |0\rangle = \frac{e^{-k\frac{i}{2}(\gamma+\theta)} |_{+01}\rangle + e^{-k\frac{i}{2}(\gamma-\theta)} |_{-01}\rangle}{\sqrt{2}} \quad (5.30)$$

from which it follows

$$|\langle 0 | (X^{01}(\theta, \gamma))^k |0\rangle|^2 = \frac{1 + \cos(k\theta)}{2}. \quad (5.31)$$

In a similar fashion, define

$$|_{\rightarrow 01}\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}} \cong \frac{|_{+01}\rangle + i|_{-01}\rangle}{\sqrt{2}} \quad (5.32)$$

where ‘ $\cong$ ’ denotes equivalence under U(1) global phase. One may similarly show that

$$|\langle 0 | (X^{01}(\theta, \gamma))^k |_{\rightarrow 01}\rangle|^2 = \frac{1 + \sin(k\theta)}{2}. \quad (5.33)$$

In this way, identify in-phase and quadrature measurements as

$$p^I(k) = |\langle 0 | (X^{01}(\theta, \gamma))^k |0\rangle|^2 \quad (5.34)$$

and

$$p^Q(k) = |\langle 0 | (X^{01}(\theta, \gamma))^k |_{\rightarrow 01}\rangle|^2. \quad (5.35)$$

Measuring γ on X^{01} : To measure γ on X^{01} , start by observing that

$$Z^{01}(\pi) \mathbf{X}^{01} Z^{01}(\pi) = -\mathbf{X}^{01}. \quad (5.36)$$

Hence

$$X^{01}(\theta, \gamma) Z^{01}(\pi) X^{01}(\theta, \gamma) Z^{01}(\pi) = \exp(-i2\gamma \mathbf{\Gamma}^{01}). \quad (5.37)$$

Next, define

$$|\mu\rangle = (|_{+01}\rangle + |2\rangle)/\sqrt{2} = (1/2, 1/2, 1/\sqrt{2})^T \quad (5.38)$$

and

$$|\nu\rangle = (|_{+01}\rangle + i|2\rangle)/\sqrt{2} = (1/2, 1/2, i/\sqrt{2})^T. \quad (5.39)$$

Then,

$$|\langle \mu | (X^{01}(\theta, \gamma) Z^{01}(\pi) X^{01}(\theta, \gamma) Z^{01}(\pi))^k | \mu \rangle|^2 = \frac{1 + \cos(2\gamma)}{2} \quad (5.40)$$

and

$$|\langle \mu | (X^{01}(\theta, \gamma) Z^{01}(\pi) X^{01}(\theta, \gamma) Z^{01}(\pi))^k | \nu \rangle|^2 = \frac{1 + \sin(2\gamma)}{2}. \quad (5.41)$$

In this way, we identify in-phase and quadrature measurements as

$$p^I(k) = \langle \mu | (X^{01}(\theta, \gamma) Z^{01}(\pi) X^{01}(\theta, \gamma) Z^{01}(\pi))^k | \mu \rangle|^2 \quad (5.42)$$

and

$$p^Q(k) = |\langle \mu | (X^{01}(\theta, \gamma) Z^{01}(\pi) X^{01}(\theta, \gamma) Z^{01}(\pi))^k | \mu \rangle|^2. \quad (5.43)$$

Measuring θ and γ on X^{12} : To measure θ on X^{12} , we start by observing that this problem can be mapped to the corresponding problem on X^{01} through a permutation of the computational basis and relabeling. Explicitly, one may show that

$$|\langle 1 | (X^{12}(\theta, \gamma))^k | 1 \rangle|^2 = \frac{1 + \cos(k\theta)}{2} \quad (5.44)$$

and

$$|\langle 1 | (X^{12}(\theta, \gamma))^k | \rightarrow_{12} \rangle|^2 = \frac{1 + \sin(k\theta)}{2}. \quad (5.45)$$

Similarly, if we define $\langle \mu' | = (1/\sqrt{2}, 1/2, 1/2)$ and $\langle \nu' | = (1/\sqrt{2}, i/2, i/2)$ then

$$|\langle \mu' | (X^{12}(\theta, \gamma)Z^{12}(\pi)X^{12}(\theta, \gamma)Z^{12}(\pi))^k | \mu' \rangle|^2 = \frac{1 + \cos(2\gamma)}{2} \quad (5.46)$$

and

$$|\langle \mu' | (X^{12}(\theta, \gamma)Z^{12}(\pi)X^{12}(\theta, \gamma)Z^{12}(\pi))^k | \nu' \rangle|^2 = \frac{1 + \sin(2\gamma)}{2}. \quad (5.47)$$

Qutrit controlled-Z RPE

The qutrit CZ gate is actually is diagonal in the computational basis. Hence, the set of eigenstates is $\{|a\rangle\langle a| \otimes |b\rangle\langle b| : a, b \in \text{range}(3)\}$

We will estimate the unitary model

$$U_{CZ}(\phi_1, \dots, \phi_8) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e^{-i\phi_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{-i\phi_2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & e^{-i\phi_3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{-i\phi_4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & e^{-i\phi_5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & e^{-i\phi_6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-i\phi_7} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-i\phi_8} \end{bmatrix}, \quad (5.48)$$

which is equal to the target qutrit controlled-Z operation for $\phi_1 = \phi_2 = \phi_3 = \phi_6 = 0$, $\phi_4 = \phi_8 = -\frac{2\pi}{3}$, and $\phi_5 = \phi_7 = -\frac{4\pi}{3}$.

The in-phase measurements are

$$|\langle 0, +_{01} | U_{CZ}^k | 0, +_{01} \rangle|^2 = \frac{1 + \cos k\phi_1}{2}$$

$$|\langle 0, +_{02} | U_{CZ}^k | 0, +_{02} \rangle|^2 = \frac{1 + \cos k\phi_2}{2}$$

$$|\langle 1, +_{01} | U_{CZ}^k | 1, +_{01} \rangle|^2 = \frac{1 + \cos k(\phi_4 - \phi_3)}{2}$$

$$\begin{aligned}
|\langle 1, +_{02} | U_{CZ}^k | 1, +_{02} \rangle|^2 &= \frac{1 + \cos k(\phi_5 - \phi_3)}{2} \\
|\langle 2, +_{01} | U_{CZ}^k | 2, +_{01} \rangle|^2 &= \frac{1 + \cos k(\phi_7 - \phi_6)}{2} \\
|\langle 2, +_{02} | U_{CZ}^k | 2, +_{02} \rangle|^2 &= \frac{1 + \cos k(\phi_8 - \phi_6)}{2} \\
|\langle +_{01}, 1 | U_{CZ}^k | +_{01}, 1 \rangle|^2 &= \frac{1 + \cos k(\phi_4 - \phi_1)}{2} \\
|\langle +_{02}, 1 | U_{CZ}^k | +_{02}, 1 \rangle|^2 &= \frac{1 + \cos k(\phi_7 - \phi_1)}{2}
\end{aligned}$$

and the quadrature measurements are

$$\begin{aligned}
|\langle 0, \rightarrow_{01} | U_{CZ}^k | 0, +_{01} \rangle|^2 &= \frac{1 + \sin k\phi_1}{2} \\
|\langle 0, \rightarrow_{02} | U_{CZ}^k | 0, +_{02} \rangle|^2 &= \frac{1 + \sin k\phi_2}{2} \\
|\langle 1, \rightarrow_{01} | U_{CZ}^k | 1, +_{01} \rangle|^2 &= \frac{1 + \sin k(\phi_4 - \phi_3)}{2} \\
|\langle 1, \rightarrow_{02} | U_{CZ}^k | 1, +_{02} \rangle|^2 &= \frac{1 + \sin k(\phi_5 - \phi_3)}{2} \\
|\langle 2, \rightarrow_{01} | U_{CZ}^k | 2, +_{01} \rangle|^2 &= \frac{1 + \sin k(\phi_7 - \phi_6)}{2} \\
|\langle 2, \rightarrow_{02} | U_{CZ}^k | 2, +_{02} \rangle|^2 &= \frac{1 + \sin k(\phi_8 - \phi_6)}{2} \\
|\langle \rightarrow_{01}, 1 | U_{CZ}^k | +_{01}, 1 \rangle|^2 &= \frac{1 + \sin k(\phi_4 - \phi_1)}{2} \\
|\langle \rightarrow_{02}, 1 | U_{CZ}^k | +_{02}, 1 \rangle|^2 &= \frac{1 + \sin k(\phi_7 - \phi_1)}{2}.
\end{aligned}$$

Each series of in-phase and quadrature measurements can be correlated to recover an experimental angle estimate φ . The experimental angles are related to the model via the matrix relation

$$\vec{\varphi} = A\vec{\phi} \quad (5.49)$$

where

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \quad (5.50)$$

Given estimates of the experimental phases, we estimate the model phases according to

$$\vec{\phi} = A^{-1}\vec{\varphi}. \quad (5.51)$$

5.4 Decoupled phase estimation

In the analysis of multi-qubit RPE, it is simplest to assume a *commutative approximation* that the multi-qubit gates are generated by commuting Paulis. However, the unitary Cartan sub-algebra that a multi-qubit RPE experiment estimates is smaller than the size of the Lie algebra of coherent errors. In other words, a commutative RPE experiment design will not be able to measure all possible coherent errors on a gate. As discussed one could, in principle, attempt to linearize multi-parameter systems to infer a complete description of a set of gate. However, multi-qubit Pauli algebra is complex, which makes analytic estimators difficult to identify. An alternative approach is to assume that the single-qubit gates have very small, negligible error, and characterize all multi-qubit coherent errors by adapting ideas of dynamical decoupling (this section) and Pauli twirling (next section) to tailor the coherent noise into a form amenable to RPE.

Decoupling gates near the identity

Decoupled phase estimate (DPE) assumes that the unitary gate that is being characterized be close to the identity and then adapts ideas from dynamically decoupled gates [VLK99] to project out all but a single coherent error parameter. The decoupled unitary parameter can then be amplified and measured with an RPE experiment design. The assumption that the unitary gate be close to the identity can be satisfied for all gates whose target unitaries compile to the identity, $U^p = \mathbb{1}$ for some power p . DPE can then be applied with the simulated idle gate U^p as the atomic gate being characterized, and then total gate error can then be reconstructed using that $(e^x)^p = e^{px}$.

To define DPE sequences, begin with a general n -qubit unitary

$$U(x) = e^x \quad (5.52)$$

and assume perfect² unitary implementation of the Pauli group $\mathcal{P}^n$. x is, as usual, a vector of length 2^n in the Lie algebra and e^x is the unitary group element.

To decouple a multi-qubit Pauli index χ associated with Lie algebra parameter $[x]_\chi$ and Pauli basis element $\mathbf{P}_\chi$, decoupled RPE constructs a gate sequence of the form

$$\prod_{\gamma \in \Gamma_\chi} (P_\gamma U P_{\gamma_m}) \equiv P_{\gamma_m} U P_{\gamma_m} P_{\gamma_{m-1}} U P_{\gamma_{m-1}} \dots P_{\gamma_1} U P_{\gamma_1} \quad (5.53)$$

for P_γ unitary Pauli gates and Γ_χ a set of Pauli indexes Γ_χ that I will call the *decoupling set*. If the decoupling set is chosen such that all associated elements $\mathbf{P}_\gamma$ commute with the

²Analytic analysis of the robustness of the technique to error in single-qubit gates is left for future work.

decoupled basis element $\mathbf{P}_\chi$ for all $\gamma \in \Gamma$, then $[x]_\chi$ will remain invariant under the decoupling sequence. Furthermore, if the decoupling set is chosen such that its associated set of Pauli operators are equal to the centralizer (§2) of $\mathbf{P}_\chi$ in the Pauli group, then it will form an approximate projector Π_χ onto the basis element $\mathbf{P}_\chi$. It follows that,

$$\prod_{\gamma \in \Gamma_\chi} (P_\gamma e^x P_\gamma) \approx e^{|\Gamma_\chi| [x]_\chi}, \quad (5.54)$$

which is elaborated on below.

To analyze the effects of decoupling sequences, the adjoint representation (§2) is convenient. Each element $P_\gamma U P_\gamma$ of the decoupling sequence can be represented

$$P_\gamma e^x P_\gamma = e^{T_\gamma x} \quad (5.55)$$

where I have defined $T_\gamma \equiv \text{Ad}_{P_\gamma}$ in the Lie algebra. It follows that the entire decoupling sequence may be written

$$\prod_{\gamma \in \Gamma} (P_\gamma U P_\gamma) = \prod_{\gamma \in \Gamma} e^{T_\gamma x} \quad (5.56)$$

Iteratively applying BCH expansions (a discrete Magnus expansion) to a sequence

$$e^{T_1} e^{T_{k-1}} \dots e^{T_1}$$

results in the

$$e^{T_1} e^{T_{k-1}} \dots e^{T_1} = \exp \left(\sum_{i=1}^k T_i x + \frac{1}{2} \sum_{i=1}^k \sum_{j=i}^k [T_i x, T_j x] + \dots \right). \quad (5.57)$$

It follows that if *the parameter vector is small*, $|x| \ll 1$, then all but the linear term may be neglected, and

$$\prod_{\gamma \in \Gamma} (P_\gamma U P_\gamma) \approx e^{\sum_{\gamma \in \Gamma} T_\gamma x}. \quad (5.58)$$

Now, observe that the decoupling set forms a group; it is the adjoint representation of the centralizer of an element $\mathbf{P}_\chi$ in the group $\mathcal{P}^n$. It follows that the arithmetic average over the decoupling set forms a projector on the invariant subspace of the centralizer

$$\Pi_\chi = \frac{1}{|\Gamma_\chi|} \sum_{\gamma \in \Gamma_\chi} T_\gamma. \quad (5.59)$$

The invariant subspace of the centralizer of $\mathbf{P}_\chi$ in the Pauli group is spanned by the identity and $\mathbf{P}_\chi$ itself; hence

$$\Pi_\chi x = [x]_\chi. \quad (5.60)$$

Combining this with Eq. 5.58, I derive Eq. 5.54 reproduced here

$$\prod_{\gamma \in \Gamma_\chi} (P_\gamma e^x P_\gamma) \approx e^{|\Gamma_\chi| \Pi_\chi x}.$$

Folded decoupling

A single decoupling sequence produces an approximate projector on a single coherent parameter that neglects higher order terms in the BCH expansion. Now, one might naively try to amplify the coherent parameter by repeating the decoupling sequence over and over again. However, those higher order terms in the BCH that we neglected when we derived the approximate projector quickly add up, and the algebraic effect of naively repeated decoupling sequence quickly diverge from projectors as the sequence is repeated. Hence, in most cases

$$\left(\prod_{\gamma \in \Gamma_\chi} (P_\gamma e^x P_\gamma) \right)^k \not\approx e^{k|\Gamma_\chi|\Pi_\chi x}. \quad (5.61)$$

Instead of naive repetition, it is better to iteratively fold decoupling sequences by composing a sequence of decoupling sequence maps. Folded decoupling can continue to re-project BCH error to zero and maintain sound projective structure even at large depths.

To analyze folded decoupling, start by defining the decoupling sequence map

$$\varphi_\chi[e^x] \equiv \prod_{\gamma \in \Gamma_\chi} (P_\gamma e^x P_\gamma) \approx e^{|\Gamma_\chi|\Pi_\chi x}. \quad (5.62)$$

Next, consider a composition of decoupling sequence maps acting on a residual unitary error

$$\varphi_\chi \circ \varphi_\chi[e^x] = \prod_{\gamma' \in \Gamma_\chi} \left(P_{\gamma'} \prod_{\gamma \in \Gamma_\chi} (P_\gamma e^x P_\gamma) P_{\gamma'} \right). \quad (5.63)$$

Using the iterated BCH expansion

$$\varphi_\chi \circ \varphi_\chi[e^x] = \prod_{\gamma' \in \Gamma_\chi} \left(P_{\gamma'} \exp \left(\sum_{i=1}^k T_i x + \frac{1}{2} \sum_{i=1}^k \sum_{j=i}^k [T_i x, T_j x] + \dots \right) P_{\gamma'} \right) \quad (5.64)$$

it follows

$$\varphi_\chi \circ \varphi_\chi[e^x] \approx e^{|\Gamma_\chi|^2 \Pi_\chi x}. \quad (5.65)$$

Repeated iteration of the projection sequence yields

$$\bigcirc^k \varphi_\chi[e^x] \approx e^{|\Gamma_\chi|^k \Pi_\chi x}. \quad (5.66)$$

Detailed analysis of the BCH error is left for future work.

The expert reader familiar with traditional RPE may be concerned that the circuit length of DPE grows as $|\Gamma_\chi|^k$ and since it is almost always the case that $|\Gamma_\chi| > 2$, one may not be able to discern branch cuts for reliable estimation of $[x]_\chi$. Here again, this issue is resolved by taking the small parameter approximation. As long as $||\Gamma_\chi|^k [x]_\chi| < \pi$, then there will be no branch cut issues. This can be ensured by measuring for small k and slowly increasing k for better precision until the condition is violated. In fact, when this condition is violated, it is likely that quadratic BCH error has become significant and the decoupling sequence is no longer an approximate projector. Recall that, in calibration, we minimize $|x|$, so one only actually needs to go to the largest depth such that $[x]_\chi$ is reliably non-zero.

Experimental protocol

The theoretical analysis above leads to an experimental protocol to measure all $4^n - 1$ non-trivial Pauli coefficients of a noisy implementation of an approximate n -qubit unitary near the identity

$$e^x = \exp\left(-\frac{i}{2} \sum_i \theta_j \mathbf{P}_j\right) \quad (5.67)$$

assuming access to a set of single-qubit Paulis with negligible coherent error. The steps are as follows:

1. Choose a parameter χ to be decoupled and identify its commutant to define the decoupling set Γ_χ .
2. Choose an $+1$ and -1 eigenstate of $\mathbf{P}_\chi$. Label them $|\psi_\chi^+\rangle, |\psi_\chi^-\rangle$.
3. Identify state preparation sequences that prepare $|I_\chi\rangle = (|\psi_\chi^+\rangle + |\psi_\chi^-\rangle)/\sqrt{2}$ and $|Q_\chi\rangle = (|\psi_\chi^+\rangle + i|\psi_\chi^-\rangle)/\sqrt{2}$.
4. Measure

$$\hat{I}_\chi(k) \equiv \langle\langle I_\chi | \bigcirc^k \varphi_\chi[e^x] | I_\chi \rangle\rangle \quad (5.68)$$

and

$$\hat{Q}_\chi(k) \equiv \langle\langle I_\chi | \bigcirc^k \varphi_\chi[e^x] | Q_\chi \rangle\rangle \quad (5.69)$$

in the usual way.

5. Apply the usual phase estimation algorithm with depths that grow as $|\Gamma_\chi|^k$.

I said that one should pick a $+1$ and -1 eigenstate of the associated decoupled $\mathbf{P}_\chi$ to form the in-phase and quadrature superpositions. In general, a 2^n dimensional Pauli will have 2^{n-1} eigenstates with eigenvalue $+1$ and 2^{n-1} eigenstates with eigenvalue -1 , which follows from induction with the fact that 1-qubit Paulis have eigenvalues $(-1, +1)$ and the eigenvalues of a tensor product of two operators are multiplicative products of the eigenvalues of the factors. As in ordinary phase estimation, one has freedom in picking which eigenstates to use, and one should generally use those that are easiest to prepare, i.e., require the fewest resources.

5.5 Twirled robust phase estimation

Just as DPE borrows idea from dynamical decoupling to identify a particular coherent error parameter of a noisy multi-qubit gate, one may similarly borrow ideas from Pauli twirling (§3.4) to project out all but one coherent error parameter to be measured with phase estimation. This is the idea of twirled phase estimation (TPE). Unlike DPE, TPE makes no assumptions that the gate be close to the identity. However, this comes at a cost of having to collect shots from many different circuits to obtain reliable estimates. A complete

twirl sequence would require a number of circuits that grows combinatorially in the depth. Approximate techniques can be employed to reduce the number of circuits.

Twirled RPE employs a partial Pauli twirl over a subset of the Pauli group to convert all coherent errors except one parameter of interest into Pauli stochastic noise. The distinguished parameter that is invariant under the twirl will be amplified and can be estimated with standard RPE techniques. As we shall see, the invariant parameter is associated with the unique Pauli operator that commutes with all elements of the twirl set. The difference between twirled and decoupled RPE is that twirled RPE uses an arithmetic average over different Pauli-conjugated circuits while decoupled RPE uses a particular sequence in a static circuit design.

Twirled RPE uses a partial Pauli twirl to form a projector in the group (instead of algebra)

$$\begin{aligned} \frac{1}{|\Gamma_\chi|} \sum_{\gamma \in \Gamma_\chi} P_\gamma \Lambda P_\gamma &= \frac{1}{|\Gamma_\chi|} \sum_{\gamma \in \Gamma_\chi} \text{Ad}_{P_\gamma}(\Lambda) \\ &= \Pi_\chi(\Lambda) \end{aligned}$$

Twirled phase estimation is stronger than decoupled phase estimation, but it comes with an explicit build up stochastic noise (from averaging).

Naively applied k times, the Γ_χ -twirl of a channel Λ is

$$\begin{aligned} (\Pi_\chi(\Lambda))^k &= \left(\frac{1}{|\Gamma_\chi|} \sum_{\gamma \in \Gamma_\chi} \text{Ad}_{P_\gamma}(\Lambda) \right)^k \\ &= \frac{1}{|\Gamma_\chi|^k} \sum_{\gamma_1 \in \Gamma_\chi} \sum_{\gamma_2 \in \Gamma_\chi} \dots \sum_{\gamma_k \in \Gamma_\chi} (\text{Ad}_{P_{\gamma_1}}(\Lambda) \text{Ad}_{P_{\gamma_2}}(\Lambda) \dots \text{Ad}_{P_{\gamma_k}}(\Lambda)) \end{aligned} \tag{5.70}$$

which represents a combinatorial overhead in the number of circuits required. However, like Clifford twirling in RB, I expect that phase twirling over centralizers will also require only averaging randomly sampled sequences from the twirl set Γ_χ . This amounts to averaging many different randomly sampled decoupling sequences.

With its average over many different randomly selected sequences, centralizer twirling starts to look a lot like Pauli noise learning; some two-qubit (or n -qubit) gate is surrounded by randomly sampled layers of Pauli operators chosen uniformly from the twirl set and measured in a particular basis. The difference is that twirled RPE performs a *partial* Pauli twirl and also correlates measurements in two bases (in-phase and quadrature) rather than just a single exponential decay. Further exploration of this connection may help to resolve the question proposed in the original RPE publication [KLY15] on the relationship between RPE and RB.

Part III

Calibration

Chapter 6

Filtering and system estimation

In §II of this work, I introduced quantum error models and the GST and RPE estimation techniques, which permit identification of error parameters in a model of a quantum device. The focus of Part III of this work is calibration – that is to select optimal classical parameters that set the estimated device parameters to their target values within an uncertainty tolerance. I solve this problem in § 7 using linear-quadratic regulators (LQRs) and model-based dual control. The theory of LQRs is built on classical control theoretic models and linear system theory. In this chapter, I introduce classical system theory in the context of quantum device calibration to set up the foundation for application of LQR in quantum processor calibration.

Closed-loop calibration protocols that utilize active experimental feedback will be necessary to combat drift and maintain error rates below the threshold for fault tolerance. A basic requirement for closed-loop control is an online filter that can estimate model parameters from noisy data in a streaming fashion (i.e. incorporate one tomographic circuit at a time). However, standard tomographic estimation techniques rely on post-processing large amounts of batched data to obtain reliable estimates. As an alternative, recursive filters, such as the Kalman filter [Kal60], have a long history of performing streaming parameter estimation and are the basis of many industrial control techniques. In this chapter, I show how quantum parameter estimation, including quantum gate set tomography (GST) (§ 4) can be efficiently performed using Kalman filters to simultaneously estimate error rates and their uncertainty (see Fig. 6.1).

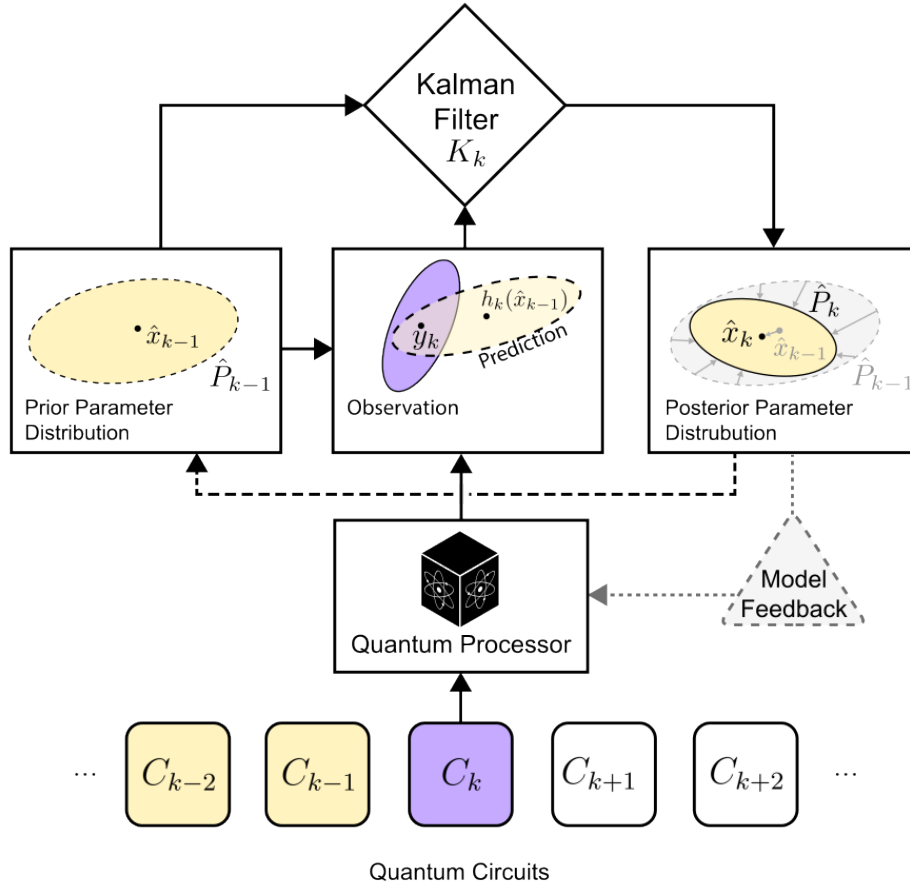


Figure 6.1: *Kalman filtering on quantum circuit outcomes.* Closed-loop control techniques requires online filters that can update estimates in real time at rates comparable to circuit execution. The linear and extended Kalman filter can be adapted for the purpose of streaming quantum tomography estimation. This chapter focuses on identification of model parameters, and § 7 introduces control laws (the dashed gray triangle in the bottom right) that use active feedback.

A conceptually simple approach to online versions of quantum parameter estimation is to adopt a Bayesian framework, where a prior probability distribution over model parameters is iteratively updated through the likelihood of each newly observed datum. However, the full Bayesian inference problem without simplification is computationally complex and ill-suited for real-time characterization within the short time-scales of quantum gate operation. When time is of the essence, one must search for simplifying model assumptions that allow for good approximate solutions to the inference problem. Such is the case with Kalman filtering, where assumptions of a) a linear model, b) Gaussian noise, and c) a Gaussian prior reduce the inference problem to a series of inexpensive matrix operations. Though gate set tomography and formally violates the these assumptions, I detail several approximations

and experiment design choices that can successfully embed GST in the framework of Kalman filtering.

The study of online estimation using Kalman filters and related techniques is a well-developed discipline with numerous introductory textbooks [GA14; The74], and a wealth of adaptations have been derived since its original formulation [Kal60] in 1960, including sigma-point (unscented) [JU97], ensemble [Eve03], and invariant [BB18] Kalman-type filters.

The outline of this chapter is as follows: I begin with a discussion of general Bayesian filtering and then introduce the linear Kalman filter. Next, I review non-linear versions of Kalman filtering, including the extended and sigma-point filters. Then, I detail how Kalman filtering applies to GST estimation and present numerical evidence of the feasibility of this approach. I conclude with a discussion of design considerations for practical use of Kalman filtering in industrial applications.

6.1 Bayesian filtering

The general problem of online filtering is conveniently cast in a Bayesian framework. In the Bayesian fashion, one defines a prior distribution $P(x|\mathcal{Y}_k)$ that quantifies the likelihood of different model parameters given all previous results $\mathcal{Y}_k = y_1, y_2, \dots, y_k$. Based on the outcome of the next experiment, one then propagates the prior to a posterior $P(x|y_{k+1}, \mathcal{Y}_k)$ distribution via Bayes rule. The posterior quantifies the likelihood given all previous and new observations, and estimates of model parameters and their uncertainties may then be drawn based on properties of the posterior distribution. The standard estimator $\hat{x}_k$ is the mean value of the distribution $P(x|\mathcal{Y}_k)$. The inference procedure may be recursively repeated each time a new observation is made.

Introducing the notion of an *innovation* is useful to connect inference to the Kalman update form. An innovation z_k quantifies the error between a new observation and the previous estimate's prediction

$$z_k = y_k - h_k(\hat{x}_{k-1}). \quad (6.1)$$

Because the model function h_k and the estimate $\hat{x}_{k-1}$ are known given $k-1$ observations, it follows that the posterior distribution is equivalent to $P(x|z_k, \mathcal{Y}_{k-1})$.

The prior and posterior are related via Bayes rule as

$$P(x|z_k, \mathcal{Y}_{k-1}) = \frac{P(z_k|x, \mathcal{Y}_{k-1}) \times P(x|\mathcal{Y}_{k-1})}{\int P(z_k, x|\mathcal{Y}_{k-1})dx}. \quad (6.2)$$

Solving the integral above yields a posterior and the whole process can be iterated each time new observations are available. However, without simplifying assumptions, the full estimation problem involves non-linear and non-Gaussian forms, so expensive numeric integration techniques that are ill suited for real time characterization are required.

Related work

Bayesian techniques have been explored for quantum state [Blu10], process [GCC16], and gate set [Di +20] tomography, but these techniques use full Bayesian inference, which is computationally expensive. The Kalman filter approach, introduced below, renders the Bayesian tomographic estimation problem tractable with modest classical resources. Other prior work [Eva+22] demonstrated a fast online gate set tomography algorithm capable of real-time characterization of quantum gate sets with a Bayesian inference technique based on a simplified Gaussian model, which shares many similarities with a Kalman filter formalism. However, the Kalman filter solves the GST estimation problem in a fashion that is theoretically more rigorous and connects to a wealth of existing techniques in classical control theory.

Particle filters

Formal application of Bayes' rule in a streaming estimation problem requires integration, which can usually only be numerically. The usual approach is to use Monte Carlo methods to simulate probability distributions and calculate posteriors [BT08]. A particularly important approach to Bayesian analysis is the particle filter, which propagates a large number of particles through a model to simulate a posterior distribution.

Particle filters are a vast topic of study [Spe16; Jou+16; K  n13] with many variations and adaptations. The design of a properly functioning particle filter is usually a difficult and subtle problem that requires balancing the number of particles, the desired classical computational complexity, and the desired estimation precision. I will not go into more technical detail and refer the reader to the literature.

Particle filters have been used in quantum noise characterization techniques [Di +20; GCC16; Fer+18] to provide Bayesian analysis of tomographic experiments. However, real-time Bayesian analysis with a particle filter is most likely infeasible due to the large classical complexity required to propagate many particles through a model and the curse of dimensionality¹. Instead, I propose to use Kalman filters to perform Bayesian analysis of quantum characterization experiments. In particular, the sigma-point Kalman filter is a promising approach, which uses particle-filter-like techniques but with a much smaller number of particles.

6.2 Linear Kalman filtering

As I have said, Bayesian inference is a natural technique for online estimation yet can incur significant computational burden without simplifying assumptions. Kalman filters overcome this computational burden by employing a linear Gaussian model (see Fig. 6.2), which ensures

¹The number of particles in a particle filter usually grows *combinatorially* with the system size. Contrast this to sigma-point Kalman filters, for which the number of particles grows linearly.

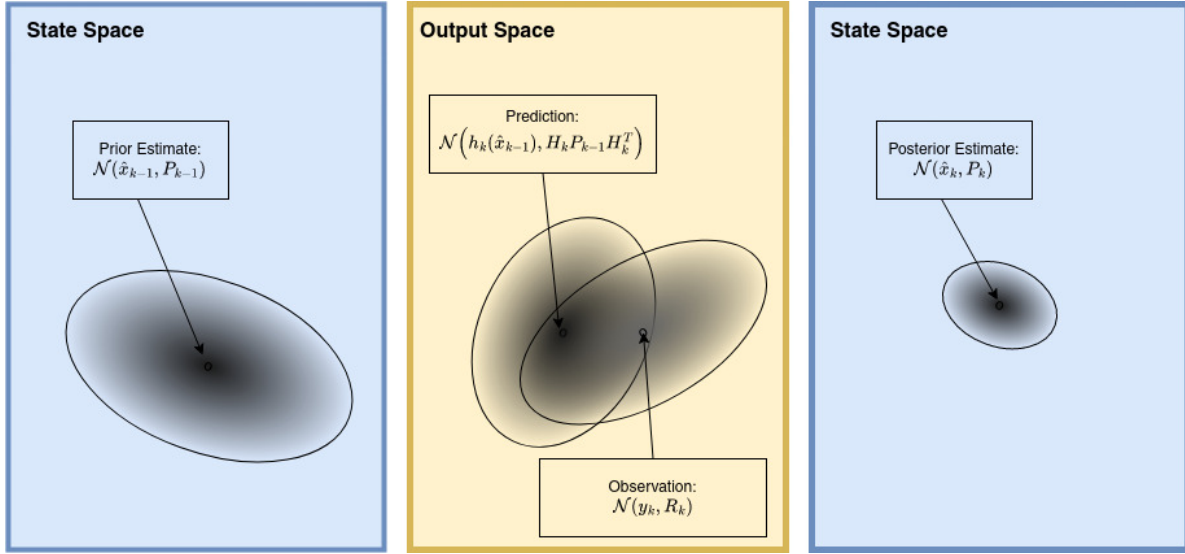


Figure 6.2: *Illustration of Gaussian updates used in a Kalman filter.* A prior Gaussian, here represented in a 2D vector space, is propagated through the model into a predicted observation. The prediction is compared with the observation, and application of Bayes rule results in a posterior distribution.

all distributions used in the estimation algorithm are Gaussian and hence can be treated analytically by tracking only a mean and covariance. Linear filtering can be used for ordinary linear tomography without amplifications. Later, I will show how non-linear Kalman filtering can be incorporated to infer model parameters from long circuits that amplify coherent parameters.

The linear Kalman filter is widely used to estimate a hidden, possibly evolving state x with linear dynamics from noisy observations y that are linear functions of the state. The system model can be cast in either continuous or discrete forms and can be adapted to include the effect of changes in control parameters. The general continuous time state evolution model employed by the linear Kalman filter is

$$\dot{x}(t) = F(t)x(t) + B(t)u(t) + w(t) \quad (6.3)$$

where $F(t)$ models the linear, possibly time-dependent dynamics of the system, $B(t)$ and $u(t)$ model the effects of controls, and $v(t)$ is a zero mean, Gaussian distributed random variable of covariance $Q(t)$ that models noise in the evolution of the state. Observations are then made of the evolving state and are modeled as

$$y(t) = H(t)x(t) + v(t) \quad (6.4)$$

where $H(t)$ is the observation or “design” matrix that models how changes in system parameters affect changes in observed outcomes, and $w(t)$ is a zero mean, Gaussian random

variable of covariance $R(t)$ that models noise in the observation that is independent of the underlying state.

Oftentimes it is conceptually simpler to work with discrete, rather than continuous, time. The passage from continuous to discrete time involves integrating the continuous time dynamics (Eq. 6.3)

$$x_{k+1} = \int_{t_k}^{t_{k+1}} (F(\tau)x(\tau) + B(\tau)u(\tau) + w(\tau)) d\tau \quad (6.5)$$

where $x_k = x(t_k)$. One then assumes that the integration interval is sufficiently small such that

$$\int_{t_k}^{t_{k+1}} (F(\tau)x(\tau) + B(\tau)u(\tau) + w(\tau)) d\tau \approx (x(t_k)F(t_k) + B(t_k)u(t_k) + w(t_k)) \Delta t \quad (6.6)$$

to arrive at

$$x_{k+1} = F_k x_k + B_k u_k + w_k \quad (6.7)$$

$$(6.8)$$

with observations

$$y_k = H_k x_k + v_k \quad (6.9)$$

where the noisy variables are still zero mean, Gaussian random variables now of discrete covariance Q_k and R_k , and $F_k = F(t_k)\Delta t$ and similarly for B_k and u_k . The passage from continuous to discrete time simply involves replacing a derivative with a finite difference. We will henceforth work in discrete time, which tends to be conceptually simpler.

A Kalman filter attempts to find an estimate $\hat{x}$ of the underlying state x by making predictions about the the next observation that will be made and updating based on the error between the prediction and actual observation. Here, as throughout this work, I denote filter estimates with a hat, e.g., $\hat{y}$ is a filter estimate for the true (infinite-shot limit) observation y . When discussing dynamic models with states that evolve in time, it is traditional to employ a double subscript notation for an estimate of the state. In this fashion, $\hat{x}_{k+1|k}$ denotes a filter's estimate of the state at time $k+1$ given all previous k observations. If one is concerned with estimating the parameters of a static model, as in the GST estimation problem, then the first subscript is not required and one can write $\hat{x}_k$ to denote the filter estimate given k observations without confusion. Kalman filters also track the covariance of the estimate, which is denoted either as $P_{k+1|k}$ or P_k depending on whether one is working with dynamic or static model parameters.

Starting with a well defined Gaussian prior distribution and assuming the linear Gaussian model above, one can then derive the Kalman update equations. First, the current estimates and the system model are used to make a prediction about the state at the next time instance

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \quad (6.10)$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \quad (6.11)$$

and then an update is made based on the current observation y_k according to

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k\hat{x}_{k|k-1}) \quad (6.12)$$

$$P_{k|k} = (I - K_k H_k)P_{k|k-1} \quad (6.13)$$

where I is the identity matrix and the Kalman gain K_k is defined as

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}. \quad (6.14)$$

The term $y_k - H_k\hat{x}_{k|k-1}$ that appears in the update equations for the mean is referred to as the “innovation” and it quantifies the error between the filter’s prediction and the observation. The Kalman update equations can be derived from multiple different starting points, including the original derivation from the perspective of minimum expected mean square error [Kal60], from using properties of jointly and conditional Gaussian random variables [MS83], or by simply multiplying a Gaussian prior with a linear Gaussian likelihood per Bayes’ rule.

Properly designed Kalman filters that satisfy the key approximations of linearity and Gaussianity promise that the expected value of the posterior error is zero,

$$\mathbb{E}[\hat{x}_{k|k} - x_k] = 0 \quad (6.15)$$

and the covariance of the posterior error is the tracked covariance $P_{k|k}$

$$P_{k|k} = \text{cov}(\hat{x}_{k|k} - x_k). \quad (6.16)$$

Similarly, the covariance of the prediction is

$$P_{k|k-1} = \text{cov}(\hat{x}_{k|k-1} - x_k). \quad (6.17)$$

I summarize linear Kalman filter recursion in Algorithm 6.1. The algorithm takes as input the priors, the ideal model’s prediction, and the observation, and then proceeds in two steps. First, a prediction is made about the noise parameters at the next time step. When the noise is assumed stationary, the dynamic linear model is the identity for all circuits, $F_k = I$, and $\hat{x}_{k|k} = \hat{x}_{k|k-1}$ such that prediction is a trivial step. After prediction, an update step is made in which an innovation and a Kalman gain are used to motivate an analog of a gradient descent algorithm. The innovation is the difference between the observed mean sample estimator and the predicted ideal model plus linear correction at the current estimate. A Kalman gain takes into account the current uncertainty in the parameters along with the uncertainty in the measurement and the model to defining a learning rate that is optimal for linear system subject to Gaussian noise. An update is made that propagates the prior mean estimate to a posterior estimate and also provides a new covariance estimate.

The central technique that connects quantum tomography with the linear Kalman filter is the construction of the design matrix [Gu+21] or the Jacobian-at-zero of the k th circuit. This quantity defines H_k that is propagated in the Kalman filter to find optimal or near-optimal estimates. Hence, the technique can be succinctly summarized as follows: define a set of model features to represent noise in a gate set, associate real parameters to each feature, and estimate the parameters using the Kalman filter with H_k given by derivatives of the circuit with respect to model features.

Input
$\hat{x}_{k-1 k-1}$ prior state estimate
$P_{k-1 k-1}$ prior covariance
y_k observation
H_k target model Jacobian
Output
$\hat{x}_{k k}$ posterior state estimate
$P_{k k}$ posterior covariance
Predict
$\hat{x}_{k k-1} = F_k \hat{x}_{k-1 k-1}$
$P_{k k-1} = P_{k-1 k-1} + Q_k$
Update
$K_k = P_{k k-1} H_k^T (H_k P_{k k-1} H_k^T + R_k)^{-1}$
$\hat{x}_{k k} \leftarrow \hat{x}_{k k-1} + K_k (y_k - H_k \hat{x}_{k k-1})$
$P_{k k} \leftarrow (I - K_k H_k) P_{k-1 k-1}$

Table 6.1: Linear Kalman filter algorithm.

Reduction to linear regression

I conclude my discussion of linear Kalman filters by demonstrating how the linear filter reduces to linear regression when under certain conditions. Observe that, when $R_k = 0$, i.e. when there is no noise in the observation, it follows that the Kalman gain is

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T)^{-1}. \quad (6.18)$$

While $(H_k P_{k|k-1} H_k^T)$ will not, in general, be invertible we can replace the inverse by the pseudo-inverse to arrive at:

$$(H_k P_{k|k-1} H_k^T)^+ = H_k P_{k|k-1} H_k^T ((H_k P_{k|k-1} H_k^T)^2)^{-1}, \quad (6.19)$$

where we have used that covariance matrices are symmetric and the fact that $A^+ = A^T (A A^T)^{-1}$ for a matrix A with linearly independent rows. Writing $K_k = H_k^+ H_k P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T)^+$ it follows that $K_k = H_k^+$ i.e., the Kalman gain is the pseudo-inverse of the design matrix of the k th circuit. We see that we can update our estimate by multiplying H_k^+ times the difference between our observed and expected measurement frequencies. If we replicate this analysis with a prior estimate of 0 and by stacking measurement outcomes and design matrices to produce a single design matrix H and observations vector y , then our best linear estimate in the absence of noise is $\hat{x} = H^+ (y - \hat{y}(0))$, where $\hat{y}(0)$ is the expected measurement outcome vector with all parameters set to zero.

Related work

My use of the Kalman filter in this chapter is distinct from quantum Kalman filters in the context of continuous weak quantum measurement [EWP17; IOY10; ASS19; Ger+03]. In the context of discrete projective measurements, the regime considered in this work, the Kalman filter has been used to estimate error bars in quantum state tomography [AS09] and I showed its applicability to gate set tomography [MY23]. The fundamental work of Ref. [GB21] indicates that many further classical estimation techniques should be applicable in this regime.

6.3 Non-linear filtering

Linear Kalman filters are optimal when the system is linear and the noise appears Gaussian [Kal60]. When the dynamics are nonlinear, then optimal filter design is a much more difficult question, and there may exist no closed form solution to an optimal filter update or the update form may be prohibitively complex. A common heuristic to deal with nonlinear filtering problems is to “extend” the Kalman filter by linearizing the system model, which produces a convenient and relatively simple filter update form. Another approach, similar to particle filtering, is to select a small number of points to propagate through a model. Both these approaches may be augmented with iteration, which attempts to improve an estimate through additional classical computation.

Extended Kalman filtering

The extended Kalman filter is based on a nonlinear model analogous to that employed in the linear Kalman filter

$$y_k = h_k(x) + v_k \quad (6.20)$$

where h_k is a nonlinear observation function that replaces the role of the design matrix H_k in the linear Kalman filter and v_k is the same observation noise as in the linear filter.

The extended Kalman filter in the static parameter regime operates almost exactly as the linear Kalman filter. The differences are that in the extended filter, we query the observation function h_k at each step, and we replace the design matrix H_k with the Jacobian of the observation function h_k with respect to model parameters x at the current estimate $\hat{x}_k$. Conceptually, these differences can be understood by replacing a linear innovation in the absence of observation noise $H_k(x - \hat{x}_{k-1})$ with a nonlinear innovation $h_k(x) - h_k(\hat{x}_{k-1})$ and then linearizing $h_k(x)$ about $\hat{x}_{k-1}$, yielding $H_k(x - \hat{x}_{k-1})$ to first order, where H_k is employed now to denote the Jacobian of h_k at the estimate $\hat{x}_{k-1}$. For a more detailed derivation of the extended Kalman update, see [Hay04].

Explicitly, in the regime of static parameter estimation without controls, the extended Kalman update equations take the form

$$\hat{x}_k = \hat{x}_{k-1} + K_k(y_k - h_k(\hat{x}_{k-1})) \quad (6.21)$$

$$P_k = (I - K_k H_k) P_{k-1} \quad (6.22)$$

where the Kalman gain K_k is defined as

$$K_k = P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1}. \quad (6.23)$$

The design matrix H_k now denotes the Jacobian of h_k at the previous estimate $\hat{x}_{k-1}$. For a complete presentation that includes a control model, see Table 6.2.

The extended Kalman filter operates very similarly to the linear Kalman filter, except the linearization is now taken at the current estimate rather than at zero in parameter space. Linearizing about the current estimate requires that we query the model during the operation of the filter and that we calculate first order variations in the model probabilities (i.e. Jacobians) at the current estimate.

Table 6.2 presents the basic form of the extended Kalman filter applied to quantum tomography. Observe that the only differences in the extended Kalman filter are that the dynamic model f_k is assumed non-linear, the observation model h_k is assumed non-linear, and the Kalman gain K_k depends on the Jacobian at the current estimate. As before, when the noise model is assumed static, then f_k is the identity function and $\hat{x}_{k|k} = \hat{x}_{k|k-1}$ for all k .

Sigma-point filtering

Like the extended Kalman filter, sigma-point Kalman filters attempt to efficiently solve streaming non-linear estimation problems using few computational resources. The sigma-point approach involves selecting a small number of nominative “sigma-points” that are similar to the particles of a particle filter. However, rather than randomly sampling a large number of particles to simulate a distribution, sigma-point filters deterministically select a small number of carefully chosen points that approximate a nearly Gaussian distribution. The number of particles required in a sigma-point filter grows linearly with the number of parameters (usually, $2n + 1$ particles are used for n parameters).

Sigma-point filters [JU97; WV00; JU04], originally called “unscented Kalman filters”², have enjoyed significant theoretical and practical attention, with a wealth of adaptations and analyses (e.g. [ZM19; KIF08; SDŠ14]). It is generally believed that sigma-point filters are superior to ordinary extended Kalman filters owing to the fact that particle propagation can capture 2nd and higher order moments [PDW10] while extended filtering based on linearization can only incorporate 1st order moments. Despite this, extended Kalman filters remain theoretically important, as Jacobian analysis is often more convenient for analytic calculations than sigma-points, which are best suited for numeric evaluation.

²Apparently, for no other reason than that their inventor thought that the extended Kalman filter smelled bad.

Input
$\hat{x}_{k-1 k-1}$ prior state estimate
$P_{k-1 k-1}$ prior covariance
y_k observation
Output
$\hat{x}_{k k}$ posterior state estimate
$P_{k k}$ posterior covariance
Predict
$\hat{x}_{k k-1} = f_k(\hat{x}_{k-1 k-1})$
$P_{k k-1} = P_{k-1 k-1} + Q_k$
Update
$H_k = \frac{\partial h_k}{\partial x} \big _{\hat{x}_{k k-1}}$
$K_k = P_{k k-1} H_k^T (H_k P_{k k-1} H_k^T + R_k)^{-1}$
$\hat{x}_{k k} \leftarrow \hat{x}_{k k-1} + K_k (y_k - h_k(\hat{x}_{k k-1}))$
$P_{k k} \leftarrow (I - K_k H_k) P_{k-1 k-1}$

Table 6.2: Extended Kalman filter algorithm.

The basic sigma-point algorithm is more complicated than the extended Kalman filter algorithm, and I refer to the [WV00; JU04] for its presentation. The central difference between the extended and sigma-point filters is that, in the sigma-point filter, the “linearization” is done via propagation of sigma-points through a model instead of calculation of a derivative. The sigma-points should be understood as a collection of vectors $\mathcal{X}$ with associated weights $\mathcal{W}$. The vectors represent points in parameter space that may be passed through the model to obtain propagated sigma points $\mathcal{Y}$ that are compared with the actual observation y . New posteriors are inferred through statistical analysis of the propagated distribution using the weights $\mathcal{W}$.

There are a variety of different sigma-point choices, summarized in Tab. 6.3, each with different trade-offs.

Iterated filtering

While the linear Kalman filter is optimal for linear systems subject to Gaussian noise, the extended and sigma-point filters do not come with any such promise of optimality. In general, the update proposed by a single iteration of a non-linear Kalman filter may not use all of the new information contained in the observation. Thus, it may be useful to iterate a non-linear Kalman filter through multiple recursions on a single circuit’s observation. By iterating

Name	# points	Reference	Comments
Julier	$2n + 1$	[JU97]	Original sigma-point definition with 1 hyper-parameter. Contains negative weights, which can cause numeric issues.
Merwe	$2n + 1$	[WV00]	Non-negative weights with multiple hyper parameters. Significant bias to the center point for some parameter choices
Cubature	$2n$	[AH09]	Equal weights on all points distributed in a “bull’s eye” pattern. No hyper parameters.

Table 6.3: Sigma-point choices. Some popular choices of sigma-point definitions with references.

multiple times on the observation of a single circuit, the filter may be able to find a new posterior estimate that is a better fit to the observed circuit outcomes.

In an iterated extended Kalman filter [HS15], the usual predict-update step is performed and then the estimate $\hat{x}_{k|k}$ is back-propagated through as a new prior another iteration. It is important that the covariance estimate $P_{k|k}$ produced by the first iteration is discarded; it is actually the prior covariance $P_{k|k-1}$ that should be used for all iterations. If we used the posterior covariance $P_{k|k}$ of the previous iteration in the next iteration, it would be as if we had observed the system exhibit the same behavior twice.

Sigma-point filters may also be iterated in a similar fashion to iterated extended Kalman filters. In this case, the sigma points are iteratively re-drawn until convergence. Recent work [ZTC22; Cha+12] has investigated a number of different approaches to iterative sigma-point filters.

The number of times to iterate over a single circuit’s observation will generally be set by a balance between computational constraints and desired performance. In practice, users should set a maximum number of iterations and an iteration similarity threshold. The iteration processes should then stop when either the maximum number of iterations has been reached or the distance between two subsequent iteration estimates is below the iteration threshold.

Iterative filtering over partial observations of a system may also be problematic. If only part of the state space is observed and that observation is used in an iterative updated, then the estimator may “travel too far” in parameter space. In general, when exploring iterative filtering techniques, it is best to batch observations so that the parameter space is completely observable or else the iterative update may diverge.

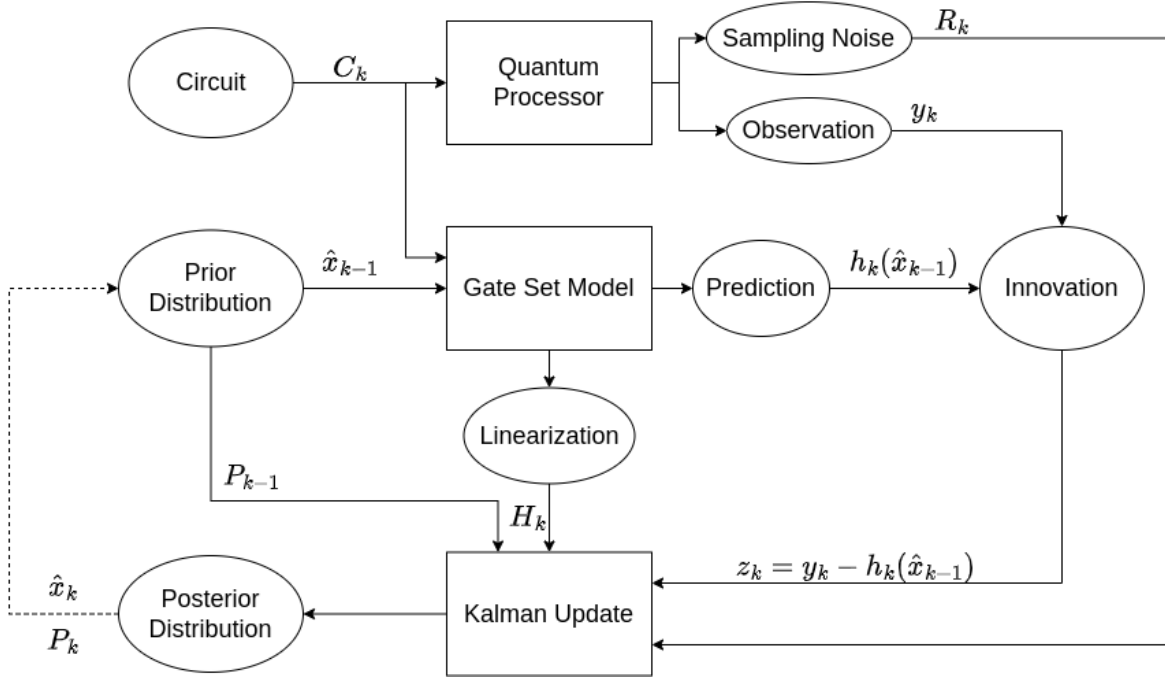


Figure 6.3: *Flowchart of Kalman filtered GST.* Diagram shows the flow of information between sub-components in a Kalman GST algorithm.

6.4 Kalman filters and gate set tomography

I now describe how Kalman filters can be used to solve the GST estimation problem, which enables streaming estimation of quantum gate set error parameters.

In applying Kalman filtering to GST estimation, there are some key assumptions and approximations that must be made: 1) the selection of an initial Gaussian prior, 2) the Gaussian approximation to the likelihood, and 3) the linearization of the observation model.

Definition of initial priors

A Kalman filter estimation routine requires initialization with an initial point estimate $\hat{x}_0$ that represents the initial guess and an initial covariance estimate P_0 that represents the estimated error in the guess. There are many choices for the initial point estimate, including starting at the target model, i.e. setting $\hat{x}_0 = 0$, or seeding the filter with an estimate derived by linear regression or MLE on some smaller set of circuits (such as those of linear GST [Nie+21]). One may also use a random Gaussian initial point centered about the target model with a predefined covariance. Other, more sophisticated techniques are also possible based on the outcome of randomized benchmarking data, such as the procedure used in [Eva+22]. In our numeric experiments, we found that estimating FOGI models is relatively

Initialization:

Gate set error model

Initial point estimate $\hat{x}_0$ Initial uncertainty P_0 **Output:**Streaming estimate $\hat{x}_k$ Streaming uncertainty P_k **Iteration:****for** Each new circuit C_k **do**Take data y_k $x_k \leftarrow x_{k-1} + K_k \cdot (y_k - h_k(\hat{x}_{k-1}))$ $P_k \leftarrow (I - K_k H_k) P_{k-1}$ **end for**

Table 6.4: Kalman GST algorithm.

robust to the choice of the initial point, and we are able to achieve good fits starting from the target model.

The Kalman filter is relatively robust to over-estimation of the initial uncertainty, so there is some freedom in choosing the initial covariance estimate P_0 . If P_0 is too small, then the filter may fail to converge, and if P_0 is too large, then the filter will explore more of parameter space early on in the estimation algorithm and thus converge more slowly. Ideally, the initial covariance should reflect the mean square error in the initial estimate.

In our examples, we set P_0 to be equal to a scalar multiple of the identity matrix. We determined the magnitude of the covariance based on the outcome of a randomized benchmarking (RB) experiment, using the RB rate r that estimates the average gate infidelity. Explicitly, we chose the initial covariance such that its trace is equal to r . In this fashion, we run a single RB experiment before we deploy streaming GST, which adds a constant overhead to the protocol. More sophisticated schemes to determine P_0 could likely be derived and is left for future work.

Gaussian likelihoods

In the context of Kalman estimation for GST, the observations are the observed frequencies of circuit outcomes. In order to employ a Kalman filter, we must describe our observations as nonlinear functions of the state, perturbed by additive Gaussian noise as in Equation 6.20. To do so we appeal to the central limit theorem [BT08]. In the limit of many circuit repetitions, M , the multinomial-distributed observations y_k will be well approximated as a

multivariate Gaussian random variable centered at the true circuit probability distribution $h_k(x)$ (the observation function evaluated at the true error parameters) with covariance

$$y_k \sim \mathcal{N}(h_k(x), R_k). \quad (6.24)$$

where

$$R_k \equiv \frac{\text{diag}(h_k(x)) - h_k(x)h_k(x)^T}{M}$$

The fact that our observations are Gaussian distributed in the limit of many circuit repetitions means that we may approximate the likelihood function for a given circuit as:

$$\Pr(y_k|h_k(x)) \propto \exp\left(-\frac{1}{2}(y_k - h_k(x))^T \cdot R_k^{-1} \cdot (y_k - h_k(x))\right).$$

where 2^n is the size of the output space and we omit the standard multivariate Gaussian normalization constant. Thus observation likelihoods are approximately Gaussian when the number of samples taken is sufficiently large. The precise number of samples that must be taken will generally depend on the number of qubits and the dimension of the output space.

In practice, we do not know a circuit's true probability distribution $h_k(x)$, so we require an estimate of the observation covariance in order to perform Kalman filtering in practice. The approach we take is to use the covariance of the conjugate Dirichlet distribution, as described in Ref. [AS09]. Given a multinomial distribution over M trials with average sample vector y , then the conjugate Dirichlet distribution whose mode is equal to y is uniquely defined as the Dirichlet distribution with pseudo-counts α equal to the observed count vector s plus the vector of all ones 1, i.e. $\alpha = s + 1$, as in Laplace's rule of succession. The resulting Dirichlet distribution will have covariance

$$R_k \approx \frac{1}{M + d + 1} \left(\frac{\text{diag}(\alpha)}{M + d} - \frac{\alpha\alpha^T}{(M + d)^2} \right) \quad (6.25)$$

where M is the number of samples and d is the dimension of the probability vector space. In this fashion, we match the covariance of our observations with the covariance of the conjugate Dirichlet distribution for the multinomial observation. It is important to note that this covariance is singular, which comes from the fact that the total counts is fixed, so we must use the pseudo-inverse in place of the usual matrix inverse.

Linearization and circuit selection constraints

Successful application of the extended Kalman filter requires accurately approximating the model observation h_k with a linear expansion in the error parameters x . The technique of model linearization expands the observation function $h_k(x)$ about the prior estimate $\hat{x}_{k-1}$:

$$h_k(x) = h_k(\hat{x}_{k-1}) + H_k[\hat{x}_{k-1}] \cdot (x - \hat{x}_{k-1}) + O(|x - \hat{x}_{k-1}|^2). \quad (6.26)$$

In order for this approximation to hold, it must be the case that higher order variations in h_k are negligible. However, the degree of nonlinearity in h_k grows as a circuit's depth increases. Our heuristic for dealing with the increasing nonlinearity of h_k is to start estimating with short circuits then feed in increasingly longer circuits as the expected estimate error shrinks. This way the observation function can be made to appear linear over the principle support of the prior, see Fig. 6.4.

Filtering on circuits in order from shortest to longest also addresses a particular kind of nonlinearity that arises in the GST circuit likelihoods. These likelihood functions can be approximately periodic in Hamiltonian error rates. This can violate the Gaussian assumption of Kalman filtering if the principle support of the prior (say the 95% confidence region) spans more than a single period of the oscillation, see Fig. 6.4. This issue also arises in robust phase estimation (RPE) [KLY15] as longer circuits provide increased accuracy but only when one can use shorter circuits to identify the principle domain of the phase. By feeding in our circuits from shortest to longest, we ensure that the priors shrink at a rate comparable to the decrease in the period of the oscillation of longer circuit likelihoods.

6.5 Numerical verification

To test and demonstrate the performance of extended filtering for online GST estimation, we have developed a Python class that iterates with the `pyGSTi` [Nie+20] package to estimate gate set model parameters in a streaming fashion. In this section, we present numerical experiments that indicate that extended Kalman filtering is a promising candidate for real-time characterization of quantum processors. We estimate the parameters of tomographically complete 1-qubit and 2-qubit FOGI noise models using both iterative extended Kalman filtering and batched MLE. We find that the Kalman filter is able to achieve estimation accuracy that compares favorably with MLE on the two simulations presented here, as well as on numerous other simulations performed with different random noise models.

The numerical experiments presented here follow the same basic steps summarized below:

1. A particular gate set is chosen.
2. A data generating model is chosen randomly.
3. An GST experiment design is computed.
4. Experimental data is simulated using the data generating model.
5. The Kalman filter is applied to simulated observations from each circuit in turn.

Our simulations are based on a 1-qubit gate set of X - and Y -rotations by $\pi/2$, and a 2-qubit gate set consisting of the same single-qubit gates and an additional controlled not (CNOT) gate from qubit 1 to 2. For convenience, we use a reduced H+S error model [Blu+22] consisting solely of Hamiltonian and Pauli-stochastic errors, and reparameterize

it using a FOGI representation. The restriction to an H+S model is not necessary, but simplifies our demonstration. We then generate a random data generating model with fixed Hamiltonian and stochastic error rates. We check that the model is completely positive and trace preserving (CPTP) and ensure that the average gate set infidelity is comparable to current devices. To highlight the ability of the Kalman filter to learn coherent errors, we also ensure that the coherent error rates contribute significantly to the infidelity. The choice of the initial covariance matrix was determined based on the outcome of a Clifford randomized benchmarking experiment [MGE11], as described above.

Circuits were selected based on standard GST practice (§ 4). In practice, we found it useful feed in circuits from a batch of fixed germ power to the Kalman filter in a random order. GST experiment designs have inherent structure, such as many circuits for which the same germ is run with different fiducials, or that include only single-qubit gates. We found this structure caused distracting artifacts in the trajectory of the point estimate. However, we found no qualitative difference in the limiting behaviour of the estimate for randomized circuit batches and structured circuit batches.

Figs. 6.5a and 6.5b display the evolution of the mean square error (MSE) in the filter model’s mean point estimate as well as in the MLE point estimate, and Figs 6.5c and 6.5d display the mean absolute error. We also plot the expected MSE and the expected MAE in the estimate, which correspond to the trace of the covariance matrix P_k and the trace of the square root of P_k , respectively. We find that the Kalman filter converges to the true model at a rate comparable to maximum likelihood estimation, and that the expected MSE and MAE evaluations are also consistent with the actual evolution. Our results indicate that Kalman filtering can achieve similar performance to batched MLE estimate.

To further illustrate the potential utility of our estimation algorithm for calibrations, we plot the evolution of a filter’s estimate of specific Hamiltonian gate errors over time in Figs. 6.5e and 6.5f. In the case of single qubit gates, we plot on-axis over-rotation errors, e.g., for a $\pi/2$ rotation about X , the corresponding over-rotation error is an additional ϵ rotation about X . In the case of the CNOT gate, we plot the over-rotation of the ZX Hamiltonian term. These results indicate that our filter is able to accurately estimate coherent, gate-specific errors, which are the types of errors that can be fixed with improved calibration and control. In particular, our technique also provides real-time uncertainty estimates of the errors, which will be useful in deriving real time calibration methods.

An unexpected advantage of our approach is that that filter was able to process circuit outcome data at a relatively fast rate. We could process about 2-25 1-qubit circuits per second and 2-5 2-qubit circuits per second on a Dell xps laptop with an i5-8250U×8 processor. The processing rate goes down as the length of the circuit increases because of the increased complexity in recomputing Jacobians for longer circuits. However, we expect that the implementation efficiency could be significantly improved by exploiting the structure of GST circuits.

6.6 Implementation considerations

In this section, we address some practical details about the implementation. Specifically, we address the selection of the prior distribution, introduce a scheme to quickly approximate the Kalman gain by calculating higher order variations in model parameters, and define an unscented Kalman filter for GST estimation.

Selection of priors

The linear and extended algorithms require a prior estimate of the model parameters $\hat{x}_{0|-1}$ and covariance matrices $P_{0|-1}$, R_k , and Q_k . The prior estimate and covariances are seen as design parameters whose definition determine the performance of a filter. These design parameters should generally be tuned to maximize experimental filter performance. Prior information and randomized benchmarking results can motivate the definition of model parameters.

The initial state estimate $\hat{x}_{0|-1}$ and covariance $P_{0|-1}$ can both be estimated by randomized benchmarking (RB) results. An RB experiment yields an RB parameter r , which is related [Pro+17; MPF18] to the average gate set infidelity. It can be argued that the RB parameter r upper bounds stochastic error parameters and that $\sqrt{r}$ upper bounds coherent error parameters. One can then reasonably assign the standard deviation of the prior covariance equal to some scalar multiple of the bounds provided by an RB experiment. Hence, a covariance

$$P_{0|-1} = d \times (rI_{\text{stochastic}}) \oplus (\sqrt{r}I_{\text{coherent}}) \quad (6.27)$$

is a reasonable prior, where d is a multiplicative constant that can be tuned to define different confidence intervals for the true model about the ideal model. In practice, it may be difficult to separate the stochastic parameters from the coherent parameters, and it may be more convenient to assign $P_{0|-1}$ equal to a multiple of $\sqrt{r}$ times an identity matrix of suitable dimension. In general, it is better to overestimate initial uncertainty.

While an all-zeros prior mean, representing the idea that the best prior model is the ideal model, will generally provide a good initial estimate of model parameters when the noise is small, other priors may also be motivated by randomized benchmarking. We may use the experimental outcomes of randomized benchmarking experiments to select a prior with a fit determined either by MLE or regression. Such fitted prior estimates will often be closer to the true noise model than the all-zeros prior and may provide better performance.

Selecting Q and R matrices

The observation covariance R_k captures uncertainty in the measurement. The nature of quantum sampling actually yields well-defined observation covariances that quantify the noise in the estimate of the mean frequency estimated by a finite number of circuit samples. We discussed how sampling error yields a Gaussian noise profile and how the Dirichlet covariance provides a robust estimate of the measurement covariance. Setting R_k equal

to the estimated Dirichlet covariance of the observation is a reasonable choice. However, it may sometimes be useful to add additional “stabilizing noise” to the system to achieve optimal performance. The exact amount of noise to add depends on the system and can be determined by experimental tuning or through methods discussed below.

The model noise covariance matrix Q_k models either stochastic fluctuations in the noise model or model fit error, which can be a result of noise processes that are present that do not fit the chosen error model. Q_k is the most difficult design parameter to provide a prior estimate for. If the system is known to be static, then it may suffice to set Q_k to an all zeros matrix, indicating that the model is perfect and the noise processes do not change. In general, both the process and measurement covariances Q_k and R_k should be tuned using real experimental data to achieve optimal filter performance.

In realistic application of Kalman filters, more attention shall necessarily be paid to the selection of design covariances R_k and Q_k , and the guidelines presented here are only first steps. It will generally be useful to add “stabilizing noise” to the observation and model covariance, which can aide the filter when the current estimate is far from the true estimate. When the current estimate is not near the true estimate, then stabilizing noise can prevent the filter from under accounting for parameter uncertainty and converge too rapidly.

Dealing with the singularity in observation noise

The Dirichlet covariance matrix that we use to model observation noise R_k is singular. This singularity means that we must employ a more expensive pseudo-inverse in our estimation routine and that useful matrix factorizations, such as the Cholesky decomposition, cannot be applied. We see two potential amendments to our method so as to deal only with invertible matrices: 1) project out the singularity and consider only the invertible part of the covariance matrix, or 2) base our covariance estimate for the observation on a Poisson rather than Dirichlet distribution. Details of the first approach may be found in [AS09], where the explicit form of the required projection operators is provided. The second approach is inspired by a Poisson Kalman filter that was derived to estimate disease transmission rates [Ebe+20]. The Poisson Kalman filter replaces our definition of R_k in Equation 6.25 with the form

$$R_k = \frac{\text{diag}(\alpha)}{(M + d)^2} \quad (6.28)$$

where, again, the pseudo-counts vector α is the observed counts plus a vector of all 1’s, M is the number of samples, and d is the dimension of the output space. As a diagonal positive matrix, the Poisson covariance estimate is clearly invertible. The key difference between the Dirichlet and the Poisson covariance estimates is that the Poisson covariance does not subtract the dyad $\alpha\alpha^T$, which is the cause of the singularity in the Dirichlet covariance. The Poisson form assumes that the samples in a batch are uncorrelated with one another, while the Dirichlet covariance includes information about correlations that arise due to the fixed shot count. We tested the Poisson form of the covariance in simulation and found little

practical difference in the convergence rates of the point estimates of the two different filters, but more work is needed to fully understand the impact of this change.

Square root Kalman filter

Finally, we note that in deploying our methods in practice, one may prefer to work with a square root Kalman filter formulation instead of directly using the covariance matrix updates. Square root Kalman filters benefit from a quadratic improvement in the maximum precision, which was actually instrumental to the early deployment of Kalman filters in the Apollo guidance systems [GA10]. Square root Kalman filters operate directly on a covariance matrix's Cholesky decomposition, which is better conditioned and accordingly affords a higher estimation precision when memory resources are limited. One obstacle in implementing a square root Kalman filter in this context is that the Dirichlet covariance is singular (due to the fact that the counts sum to a constant number), so one cannot directly apply a Cholesky decomposition. Techniques to deal with this singularity were already provided in the previous paragraph.

Sigma point Kalman filter

I have additionally investigated the sigma point (or unscented) Kalman filter, and found that it achieves comparable performance to the extended Kalman filter. I used the FilterPy package [Lab20] to implement the filter and perform the analysis. Given that sigma-point filters are generally superior in numeric applications, I believe that industrial applications should focus on the sigma point filter as the most promising estimation technique.

Fast extended filter

The recursion of the extended filter requires greater computational complexity than the linear filter because of the need to calculate a model derivative (Jacobian) H_k at each time instance and the need to query the model. We can greatly speed up the run-time of the extended Kalman filter by pre-calculating higher order variations in the ideal model and then using these higher order variations to approximate the model and its variations at other points in parameter space. For example, by calculating the 2nd order variations of a circuit's probability distribution about the ideal model, we can approximate the 1st order variations in a circuit's distribution at another point near the ideal model. Symbolically

$$\left(\frac{\partial p_k}{\partial x_i}\right)_{\hat{x}} \approx \left(\frac{\partial p_k}{\partial x_i}\right)_0 + \sum_j \left(\frac{\partial^2 p_k}{\partial x_j \partial x_i}\right)_0 \hat{x}_j \quad (6.29)$$

or using J for the Jacobian and A for the Hessian

$$J(\hat{x}) \approx J(0) + \hat{x} \cdot A(0). \quad (6.30)$$

Higher order variations about the ideal model can also be used to approximate a circuit's probability distribution when evaluated at another point in model space. For example, knowing the Jacobian and Hessian of a circuit about the ideal model and also the probability of a circuit's outcome under the ideal model, we can approximate the circuit probability at another point $\hat{x}$ in parameter space as

$$p_k(\hat{x}) \approx p_k(0) + J(0) \cdot \hat{x} + \hat{x} \cdot A(0) \cdot \hat{x}. \quad (6.31)$$

There is a memory speed trade-off between the fast and the exact filtering algorithms. While the fast filter's actual run-time is faster than the exact filter's run-time, the fast filter requires significant prior computation and memory to calculate and store the model variation tensors. The required memory to store second order variations grows quadratically in the number of model parameters. When the available memory is not enough to match the memory requirement of the model variation tensors, then we propose to use singular value decomposition (SVD) to compress the model design Hessians. In this fashion, one may calculate the SVD of each Hessian of the initial model and compress the corresponding matrices by keeping only the largest m singular values. The required memory then scales as m times the number of model parameters. Such compression schemes will be useful when deploying the fast filter on low-level control hardware, where the memory resources are limited.

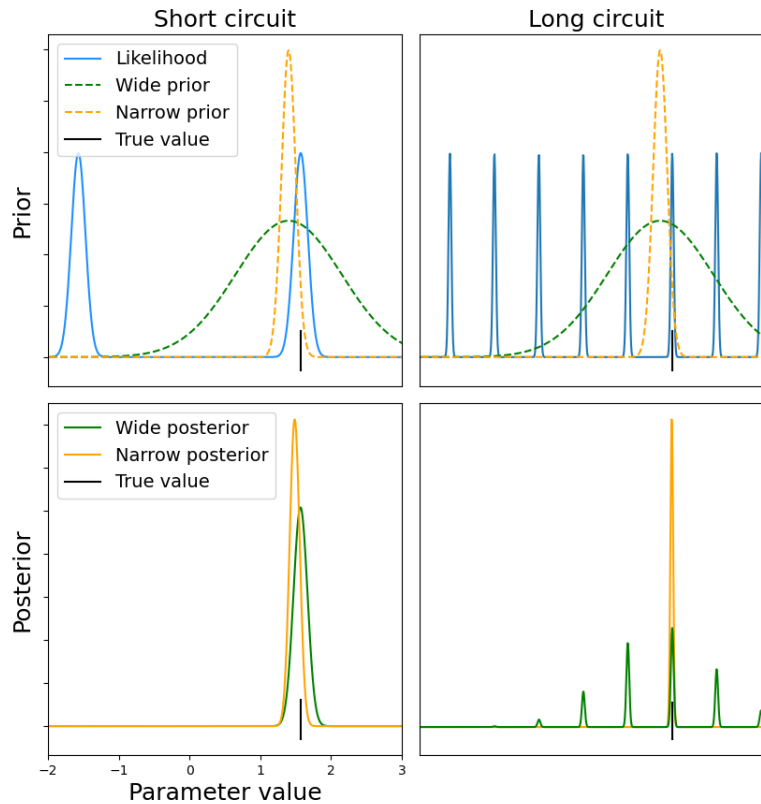


Figure 6.4: *Influence of the prior distribution on Bayesian updates.* The top two plots show the prior and likelihood for a short and a long circuit for a simplified 1-parameter model, and the bottom two plots show the resulting posterior when calculated via Bayes rule. In the case of the short circuit, the wide circuit prior moves closer to the true value than the narrow circuit prior. In the case of the long circuit, the wide prior produces a multi-modal distribution when multiplied with the likelihood, which violates the assumptions of the Kalman filter, and the narrow prior results in a unimodal distribution that can be well approximated as a Gaussian. This example illustrates that, in order to assume Gaussian priors and Gaussian likelihoods needed for Kalman filtering, the length of the circuit should be selected such that the likelihood is unimodal on the principle support of the prior.

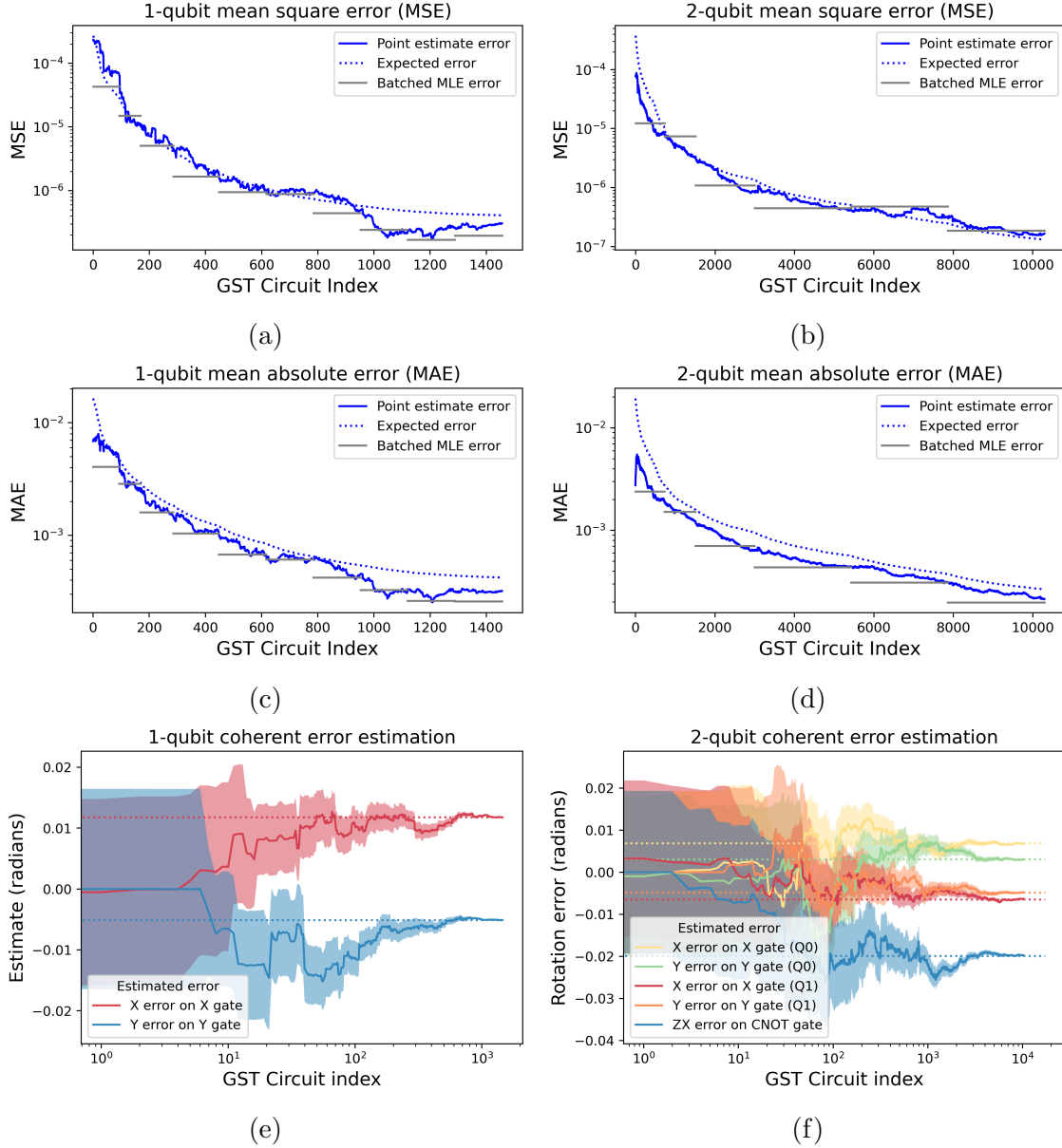


Figure 6.5: *Numerical performance of streaming GST.* Plots (a)-(d) compare the convergence rates of the Kalman filter’s point estimate with batched MLE point estimates under a metric of mean square error (MSE) and mean absolute error (MAE) between the point estimate and the parameters of the data generating model. The x -coordinates of the gray lines correspond to batches of germs of fixed power, and the batched MLE point estimates are calculated based on the observations from all data up-to and including the current batch. These plots also compare the evolution in the filters’ MSE and MAE with their expected evolution given by $\text{Tr}(P_k)$ and $\text{Tr}(\sqrt{P_k})$ respectively. MSE is the natural metric for a Kalman filter since Kalman filters minimize the square of the expected error in the estimate, but MAE is a stronger performance metric that is more sensitive to small differences in parameters. Plots (e) and (f) display error in the estimate of particular Hamiltonian parameters that correspond to the types of errors we expect could be reduced with improved calibration. The dotted lines denote the “true” parameters that were used to generate the data.

Chapter 7

Calibration and stabilization

Quantum processors are expected to implement a set of quantum gates whose actual action approximates to a target unitary action with extremely high fidelity over long time periods. A central adversary in maintaining high fidelity operation is then drift in gate operation. Coherent drift processes manifests as small changes in the qubit environment or classical control system that build up over time and result in sub-optimal gate implementation. Maintaining gate operation in the presence of drift becomes more difficult as the system scales. For this reason, identifying efficient and optimal techniques to combat coherent system drift will become essential to large-scale processor engineering.

This chapter derives one approach to drift mitigation based on a classical controller with feedback from quantum measurements. Here, a control law regulates the classical parameters, or “configurations,” that implement gate operation to maintain performance metrics. I specifically imagine the controller to act on information obtained from phase estimation algorithms §5, but I strive to provide a theory that encompasses more general situations such as partial tomographic feedback (for example using techniques from §6). The controller minimizes a regularized squared error between the estimated phases and the target phases. Additions of estimate smoothers provide additional robustness to noise.

A key technique employed in this chapter is the assumption of *linearity*, which permits derivation of efficient and optimal algorithms. In general, linear models do not hold for all points and all updates in configuration space. However, it is often the case that there exists a range of parameters (near a point of stability) where small changes affect the error state approximately linearly. I have obtained experimental evidence this is true for quantum devices, which motivates the use of linear models.

The calibration approach I introduce here essentially implements a classical “linear-quadratic Gaussian” (LQG) control law that combines a Kalman filter system smoother with a linear-quadratic regulator (LQR). The general operation consists of an estimate and update step:

1. Estimate the system’s state up to known uncertainty
2. Update the configuration to account for error.

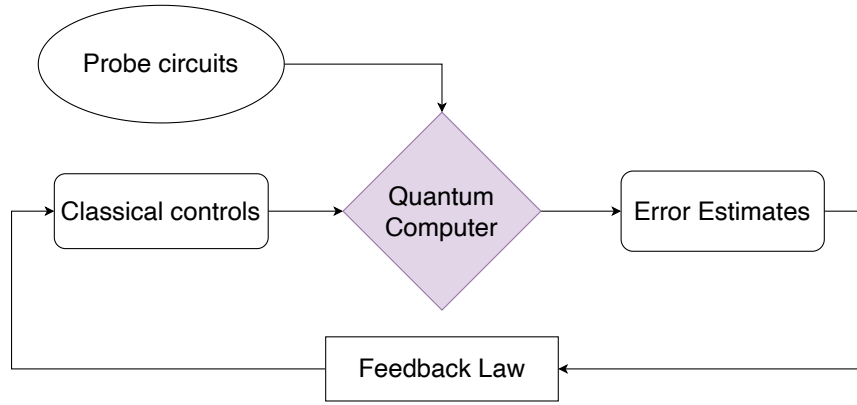


Figure 7.1: *Generic calibration loop structure.* Classical controls drive the operation of a quantum device. Specific probe circuits are measured and processed with an estimator to infer deviations from the target (ideal) operation of the processor. A classical feedback law updates the controls to optimize performance and minimize device errors.

These steps are repeated over long time periods to regulate performance. The estimation is done with RPE techniques, possible combined with a Kalman filter smoother on the estimates. The updates are calculated per the LQR formalism.

The techniques introduced are analogous to “servo” control. This follows from a comparison to classical servo motor control techniques that regulate the speed or position of a mechanical operation based on sensor feedback. In our context with quantum processor calibration, the quantum operations are regulated based on feedback from probe circuits.

This chapter introduces necessary background and then derives new regulation algorithms for quantum devices with feedback from estimated phases. I begin with a discussion of classical system models of quantum devices and a review general principles of classical control. Next, I derive quantum analogues of classical observers, review the theory of LQRs, and discuss dual control techniques. Finally, I present efficient regulation algorithms for quantum devices.

7.1 Classical system models of quantum devices

To develop the quantum calibration problem as a classical control problem, one should start by formulating the device calibration as a classical system problem. The problem can then be solved with existing classical control techniques. As we saw in § 6, quantum tomography can generally be cast as a system estimation problem with non-linear observations. In this chapter, I will mostly consider the system model to be a subset of phases that are estimated with RPE. The important thing is that a tomographic or phase estimation protocol outputs a set of error parameters x that have some target values we seek to regulate. Below,

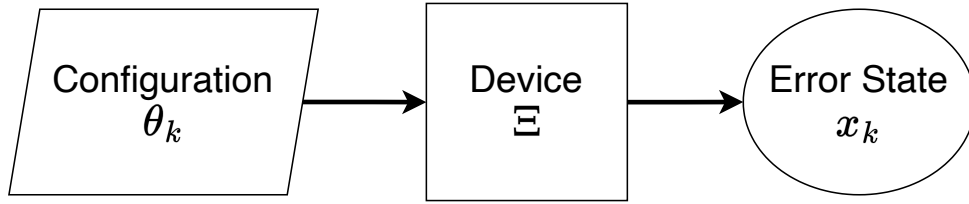


Figure 7.2: *Calibration as an input-output problem.* An input configuration θ maps to an error state x through the device map Ξ . The configuration θ represents all classical control parameters used to drive the quantum device, e.g. pulse amplitudes, frequencies, timings, etc... The device map represents the physical realization of the computation with given configuration. The error state represents the deviations between the ideal operation and the actual implementation.

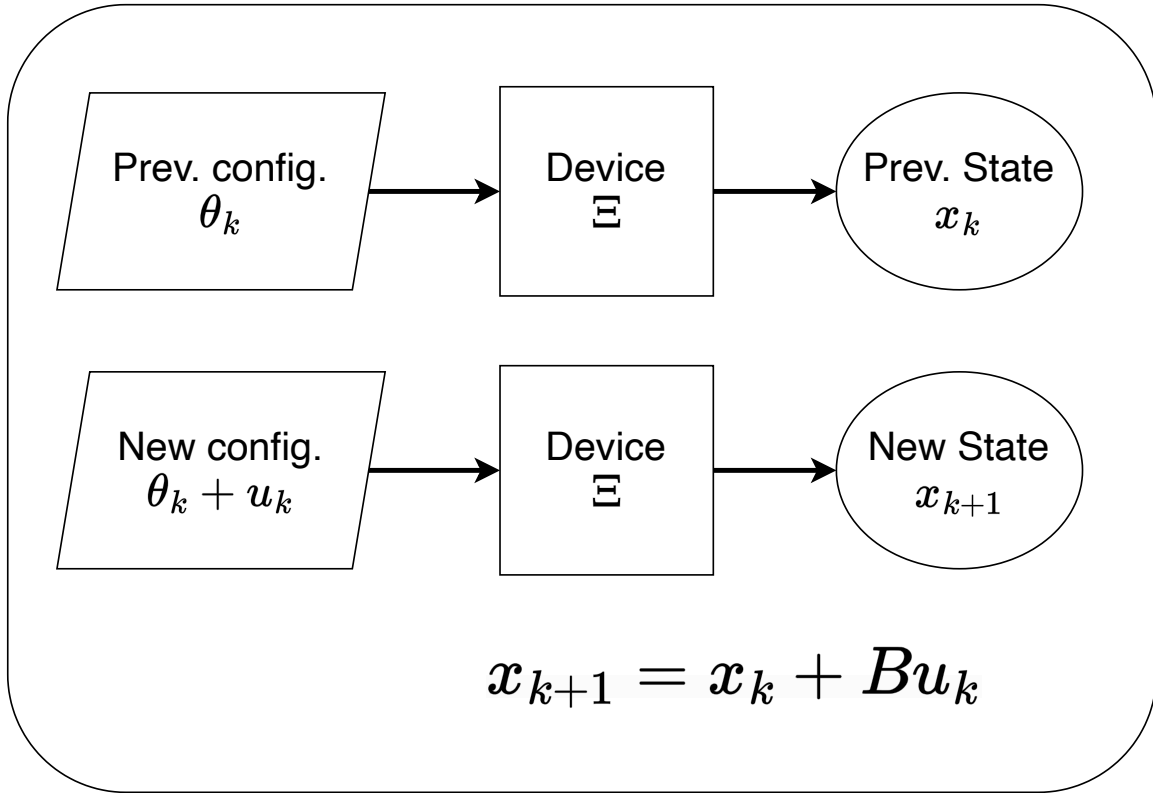


Figure 7.3: *Linearization of the input-output map.* A sequence of configurations are produced via control updates u . If u is small, then the new error state x_{k+1} is well approximated by a linear function of the old error state x_k , the configuration update u_k , and the control map B that depends on the device map Ξ .

I provide an abstract formulation of quantum devices as classical system models.

Control model of quantum devices

The general quantum device calibration (see Fig. 7.1) problem may be formulated as an input-output map Ξ between configuration parameters θ and device error parameters x

$$\Xi(\theta) = x. \quad (7.1)$$

The configuration θ represents all the classical information that is supplied to generate the drive pulses for the gates (e.g. numbers in a YAML file) and the device parameters x correspond to estimated parameters extracted from partial tomographic experiments.

A particular configuration θ contains things like drive frequency, envelope information, drive amplitude, drive phase, etc... for each gate in the gate set as well as readout frequency, amplitude, phase, etc... At runtime, the configuration is compiled into pulse waveforms to implement the gates of a particular circuit. The device map Ξ should be understood as the map from a particular configuration to physical drives with error parameters x . I write estimates of errors with a hat, as in $\hat{x}$. I will subscript the parameters, e.g. θ_k and x_k , to denote a series (usually temporal) of error states and configurations. As done throughout this thesis, when referring to a particular element of a vector, I wrap it with square brackets, e.g., $[x_k]_i$ is the i th element of the k th vector in the series.

I will usually use x to refer to coherent error parameters, but much of this discussion can be applied to general error models with parameters x . In this way, the map from configuration to errors Ξ should be understood as being specified by both the device and the model.

While the general input-output map Ξ is useful from a theoretical viewpoint, the relationship between raw configuration settings θ and device parameters x is highly non-linear, which means that deriving an optimal control law in this setting would be extremely difficult (and probably impossible). Instead of working directly with the configuration map Ξ , we *linearize* the control model by considering small, perturbative changes to the configuration. In this way, one defines the control update u_k to a prior configuration θ_k that results in a new configuration

$$\theta_{k+1} = \theta_k + u_k. \quad (7.2)$$

The corresponding effect on the error state is

$$\Xi(\theta_{k+1}) = \Xi(\theta_k + u_k). \quad (7.3)$$

Now, Taylor expanding Ξ about $u = 0$,

$$\Xi(\theta_k + u_k) = \Xi(\theta_k) + \sum_j \left. \frac{\partial \Xi}{\partial [u]_j} \right|_0 [u_k]_j + O^2(u_k) \quad (7.4)$$

and assuming u_k is sufficiently small such that higher order terms can be discarded, one obtains the relation

$$\Xi(\theta_{k+1}) = \Xi(\theta_k) + \sum_j \frac{\partial \Xi}{\partial [u]_j} \Big|_0 [u_k]_j. \quad (7.5)$$

Writing $Bu_k = \sum_j \frac{\partial \Xi}{\partial [u]_j} \Big|_0 [u_k]_j$ and recalling $\Xi(\theta_k) = x_k$, we finally obtain

$$x_{k+1} = x_k + Bu_k, \quad (7.6)$$

which is the usual linear, time-invariant control model.

Time-variant control models

It is also possible for the control model itself to be time varying. In this case, I will write the input-output model with a subscript, Ξ_k . Through a similar sequence of steps as above, we write

$$x_{k+1} = \Xi_k(\theta_k + u_k), \quad (7.7)$$

expand, and replace Bu_k with

$$\sum_j \frac{\partial \Xi_k}{\partial [u]_j} \Big|_0 [u_k]_j = B_k u_k \quad (7.8)$$

to obtains the linear, time-variant control model

$$x_{k+1} = x_k + B_k u_k. \quad (7.9)$$

Stochastic drift

The control models derived so far model the input-output only include deterministic effects. Essentially, these models make the assumption that the error state at one moment in time is completely determined by the configuration file. In practice, real devices drift, so the error state will change over time for fixed controls. The drift may actually come from a variety of sources, each with accompanying classical system model. The two most important sources of drift are: 1) in the control drives and 2) in the control model (i.e, changes in system-drive couplings). A stochastic model can deal with drift and permits effective bookkeeping of the uncertainties, which can be used to derive optimal estimators (like a Kalman filter).

To begin, let us first consider the generic classical control model for drift w in an error state x subject to controls u

$$x_{k+1} = Ax_k + Bu_k + w_k \quad (7.10)$$

where A represents deterministic drift in the error state and w_k is a random variable representing stochastic drift. Throughout this work, I assume $A = \mathbb{1}$. A is usually used to model

velocity and acceleration in control problems, which seem to have no quantum calibration analogue. However, it is possible that future work may find utility in including a non-trivial A , for example in noisy algorithm design. The nature of the stochastic drift w_k depends highly on the physical model. It is typical to assume that w_k is a zero-mean Gaussian random variable, which permits straightforward Bayesian analysis, but this is only a coarse –or even poor– approximation in many circumstances.

Now, let us restrict to the case of drift in the control drives, which implies that our configuration update equation includes a stochastic term

$$\theta_{k+1} = \theta_k + u_k + s_k. \quad (7.11)$$

For random variable s_k . The resulting error state model is

$$x_{k+1} = x_k + B(u_k + s_k). \quad (7.12)$$

In this model, the error state accrues error according to drift in the control parameters. This model may be used to account for situations where the control parameters are fluctuating. For example, one may have a drive whose amplitude is changing over time due to noise in the classical device.

It is also possible to have drift in the control model itself. Such a situation might arise when system-drive coupling changes over time. For example, in superconducting qubits, changes in the TLS structure may result in different qubit-drive couplings¹, which should be appropriately modeled as changes to the control model rather than the configuration parameters. In this case, the appropriate system model is defined by the pair of equations

$$x_{k+1} = x_k + B_k u_k. \quad (7.13)$$

$$B_{k+1} = e^w B_k$$

where the second equations models the drift as stochastic dynamics on a Lie group [BB18] (since B is a matrix). Here, I have used the short-hand notation of §2, and w should be understood as a stochastic parameter vector in a Lie algebra that generates a small action in the group.

Observation models

A situation that commonly arises in experiments is stochastic noise that does not accrue over time. For example, shot noise in experiments should not be modeled as the sort of integrated stochastic processes of the previous few equations because its covariance does not grow with time. Instead, such a situation would be better modeled as a stochastic perturbation to each measurement that is made of the system. This sort of stochastic noise is best modeled at the estimation level, rather than the dynamical level, and is discussed here.

¹In this situation, one would keep amplitude constant, but that constant amplitude would have different effects over time.

Given a control model, the final ingredient in a classical system model is the observation model and system estimation technique. In a linear system, the error parameters x_k are assumed to be related to the measured phases y_k according to the linear model

$$y_k = Cx_k + v_k \quad (7.14)$$

where C is a design matrix specific to the particular experiment design and v_k represents the uncertainty in the estimate.

As we saw in §6, the observation model does not have to be linear and can usually be linearized to a form approximately equal to above. As I have said, I am primarily interested in applying phase estimation techniques as system estimators. Phase estimation algorithms produce an observation of the system state x_k up to some uncertainty determined by the last trusted generation.

My use of phase estimation in this context allows a practitioner to ignore the complicated, non-linear relationship between errors and empirical distributions collected from a device, which is accounted for at the level of estimator design – rather than control law design. The separation principle (discussed below § 7.2) argues that an optimal control law can usually be derived independently of the estimator.

7.2 Principles of control law design

We have seen that a quantum system can be cast as a classical system model. In this section, I review general principles of classical control algorithm design. These will later be applied to quantum devices

To begin, recall the usual linear, time-variant model of classical control

$$x_{k+1} = x_k + B_k u_k + w_k \quad (7.15)$$

$$y_k = Cx_k + v_k. \quad (7.16)$$

For error state x , control model B , control update u , observation y , design matrix C and noises w, v .

Control laws define rules to select control updates u_k based on observations y_k . A usual approach is to assume time-invariant dynamics and combine a linear system observer with a linear control gain. The system observer produces an estimate of the next error state², $\hat{x}_{k+1}$ given the prior state estimate $\hat{x}_k$, the control action u_k , and the observation y_k , as

$$\hat{x}_{k+1} = \hat{x}_k + K(y_k - C\hat{x}_k) + Bu_k \quad (7.17)$$

for an observer gain K (see § 7.3). The control law is then defined via a linear feedback gain

$$u_{k+1} = -G\hat{x}_{k+1}. \quad (7.18)$$

²here, I have dropped the usual conditional notation for Kalman filters, e.g. $\hat{x}_{k+1|k}$ and $\hat{x}_{k|k}$, and I write $\hat{x}_k$ to denote $\hat{x}_{k|k-1}$, i.e. the estimate at k given $k-1$ observations. This follows from our discussion of the proper analogy between Kalman filters and Luenberger observers Eq. 7.26.

Observation of y_{k+1} is then fed to the observer to produce a new estimate $\hat{x}_{k+2}$ and the iterations continue.

The art of designing of well functioning control laws balances computational, performance, and robustness constraints. The remainder of this section addresses key issues in designing control laws, including observability, controllability, the separation principle, dual control, and certainty equivalence control. Synthesis of these ideas produces sound control algorithms to regulate quantum device performance.

Controllability

The input control updates effect the error state according to the model Eq 7.6. When the system is controllable [Kla08; Kla13], then one can always select sequence of control inputs $\mathcal{U}_k = (u_1, u_2, \dots, u_k)$ to drive the error state x_k to a desired target value. In our case, we will be exclusively interested in calibrating the error state to 0, which is possible if one can always pick a u^* for every x such that $x + Bu^* = 0$.

Controllability is usually quantified in terms of the rank of the tangent space of the effect of control inputs on parameters. In our context, with no system momentum or acceleration (no A matrix in the linear system model), a system is controllable when B has rank equal to the length of x . Note that this actually allows for more control inputs than error parameters. Nevertheless, it is usually best to design a system with the number of control inputs equal to the number of error parameters when possible. When $\dim(u) = \dim(x)$, then the system is controllable if B has full rank.

Observability

Observability describes how well the hidden state x can be estimated from observations y . Given a general linear or nonlinear system, there is no guarantee that all aspects of the state can be estimated from the observations. A simple example would be trying to estimate the position of a robot with access to only one coordinate of its GPS reading. Observability was first introduced by Kalman [Kal63], and in the context of linear, time invariant systems it is straightforward to derive a canonical observation form that decouples the observable parameters from the unobservable parameters. Given a canonical observation form, one simply estimates the observable parts and ignores the unobservable parts. Linear Kalman estimation can be proven to converge when performed on a set of parameters that are completely observable. However, for nonlinear systems with time variant observations, as in the case of GST, the problem of observability is much more difficult [KET73].

A system is observable if all aspects of the error state x can be inferred from observations y . In the context of the usual linear system model Eq. 7.15, the system is observable if C has rank equal to the dimension of x , which is like controllability but with observations y in place of controls u . y can be of larger dimension, but –as with controllability – it is usually best to have the number of observations equal the dimension of the error state. When $\dim(y) = \dim(x)$, then the system is observable if C is full rank.

Separation principle

The separation principle [GL13] states that the system estimator and the system feedback law may be designed independently in an optimal controller. As long as both the estimator and feedback law are optimal, then the composite control law will also be optimal. The separation principle is useful because it allows us to design the estimator and feedback law independently of each other.

In the case of a linear system like Eq. 7.15, a general estimator may be written

$$\dot{\hat{x}} = (\mathbb{1} - LC)\hat{x} + Bu + Ly \quad (7.19)$$

for a gain L (see discussion of observers §7.3). A general control feedback law is

$$u(t) = -G\hat{x}. \quad (7.20)$$

The separation principle states that the optimal choice of L and G are independent of each other.

The dual control problem

Dual control (§ 7.5) refers to the dual problem of control model estimation. The “primal” or fundamental problem of classical control is to set $x = 0$ using control updates u . In order to do so, a controller must understand how u affects x , which is described in the control model. Estimation of the control model can be done with a variety of techniques, including neural networks and deep learning. In this chapter, I concern myself with simple, linear control models. Sometimes, the control model is already known, but when it is not, we must estimate it to solve the fundamental control problem.

It is sometimes possible to write down a control model from physical principles. For example, a phase in an idle operation accrues like the product of the frequency of the qubit and the time of the idle. If the control is frequency updates, then the control model can be written from the known duration of the idle. However, other times there are unknown parameters that can only be measured. For example, the frequency of Rabi oscillations in Transmon qubits depends on an unknown qubit-drive coupling parameter that changes between qubits.

Certainty equivalence

Certainty equivalence [VW81] refers to using an estimated model in place of the actual driving model of system dynamics. In our context, Certainty equivalence appears when an estimate $\hat{B}$ of the control model B is used. In this context, it is also usual to discuss streaming dual estimators that update a control model estimate based on observations of the effects of controls. Under certain conditions, a control law that employs an on-line dual estimator converges to the optimal control law [KV16].

7.3 Classical observers of quantum devices

The notion of a system observer formalizes the system estimation problem in the context of classical control. An important linear system observer is the Luenberger observer [Afr+17], which operates robustly in a deterministic setting. The Kalman filter (see § 6) provides a straightforward way to define an optimal observer in a stochastic setting. Kalman filters operated over infinite time intervals converge to non-adaptive observers. In this section, I introduce linear observers of quantum device performance parameters and discuss how the Kalman filter operates as a system observer.

A linear, time-invariant, deterministic state observer assumes the generic model

$$x_{k+1} = x_k + Bu_k \quad (7.21)$$

$$y_k = Cx_k \quad (7.22)$$

where y_k is an *observation* collected from measurements of the device with specific settings, and C relates the measurement to the error state.

Luenberger observers

Given a linear-time invariant model as above, the a general Luenberger observer [Afr+17] takes the form

$$\hat{x}_{k+1} = \hat{x}_k + L[y_k - C\hat{x}_k] + Bu_k \quad (7.23)$$

$$(7.24)$$

where L a Luenberger gain matrix. An observer produces a series of estimates $\hat{x}_k$ given observations y_k . The initial estimate $\hat{x}_0$ may be defined $C^{-1}y_k$. The Luenberger gain L should be understood as a tuneable hyperparameter matrix that governs the performance of an observer.

Luenberger observers can be used to analyze expected performance. The observer state error $e_k = \hat{x}_k - x_k$ can be shown³ to evolve according to

$$e_{k+1} = (\mathbb{1} - LC)e_k \quad (7.25)$$

³Observe that

$$\begin{aligned} e_{k+1} &= \hat{x}_{k+1} - x_{k+1} \\ &= \hat{x}_k + L[y_k - C\hat{x}_k] + Bu_k - x_{k+1} \\ &= \hat{x}_k + LCx_k - LC\hat{x}_k + Bu_k - (x_k + Bu_k) \\ &= \hat{x}_k - x_k - LC(\hat{x}_k - x_k) \\ &= (\mathbb{1} - LC)e_k. \end{aligned}$$

Hence, if $\mathbb{1} - LC$ has eigenvalues in the unit circuit (ie. LC has eigenvalues of modulus no greater than 1), then the system is asymptotically stable. At this point, one usually follows one of several pole placement techniques [GS14] to derive a stable observer, which I will not discuss. Instead, I derive optimal observers from a Kalman filter viewpoint, discussed below.

The non-expert reader may wonder why one does not select $L = C^{-1}$. Indeed, this choice would imply that $\mathbb{1} - LC = 0$ and $\hat{x}_k = x_k$ for all k , seemingly obviating the observation problem. For deterministic systems with no observation noise, then this is the optimal choice, and sometimes it is convenient to make this choice when the observation noise is known to be very small. However, no real system has zero observation noise, and Luenberger observers provide a rigorous framework to analyze convergence rates and robustness to noise via pole analysis, which, again, I do not discuss and instead point the reader to the vast literature on the topic [GS14].

Kalman observers

Another approach to observer design is that of Kalman filters, which were already discussed in §6. In this section, I will demonstrate how the Kalman filter formalism defines a general observer in the language of classical control. Kalman filters are particularly attractive because they allow for precise bookkeeping of uncertainties, and they are not restricted to time-invariant systems. As I discussed in Chap. 6, with some generalizations, Kalman filters can also be applied to nonlinear system identification.

As with Luenberger observers, one can analyze the error evolution of a Kalman filter. Because the Luenberger estimate $\hat{x}_{k+1}$ is conditioned on only k observations, the proper Kalman analogy of Eq. 7.25 is

$$e_{k+1|k} = \hat{x}_{k+1|k} - x_{k+1} \quad (7.26)$$

which can be shown⁴ to be

$$e_{k+1|k} = (1 - K_k C) e_{k|k-1} + K_k v_k + w_k. \quad (7.28)$$

If we take the expectation of $e_{k+1|k}$ over v_k and w_k , we recover the Luenberger error evolution with gain K_k . In this way, a Kalman filter may be understood as a Luenberger observer

⁴Observe:

$$\begin{aligned} \hat{x}_{k+1|k} - x_{k+1} &= \hat{x}_{k|k} + B_k u_k - (x_k + B_k u_k + w_k) \\ &= \hat{x}_{k|k-1} + K_k (y_k - C \hat{x}_{k|k-1} - v_k) - x_k + w_k \\ &= \hat{x}_{k|k-1} + K_k (C x_k - C \hat{x}_{k|k-1} - v_k) - x_k + w_k \\ &= (1 - K_k C) (\hat{x}_{k|k-1} - x_k) - K_k v_k + w_k. \end{aligned}$$

Since v_k is zero-mean, the sign on it may be flipped without consequence. Hence

$$\hat{x}_{k+1|k} - x_{k+1} = (1 - K_k C) (\hat{x}_{k|k-1} - x_k) + K_k v_k + w_k. \quad (7.27)$$

with an adaptive gain. The Kalman filter form also highlights the interaction between the gain matrix K_k and the observation noise v_k ; the gain matrix amplifies observation noise, so if the observation noise is high, then the gain matrix should be low, which will imply slower convergence but more robustness to observation error.

When operated over infinite time periods, Kalman filters converge to an asymptotic form, which has further highlights their connection to Luenberger observers. Now, expanding the Kalman covariance update equations yields

$$P_{k+1|k} = P_{k|k-1} - P_{k|k-1}C^T(CP_{k|k-1}C^T + R)^{-1}CP_{k|k-1} + Q, \quad (7.29)$$

which one may recognize as discrete time Riccati equation [GS14]. There is a unique stabilizing solution that solves the fixed point equation

$$P_\infty = P_\infty - P_\infty C^T (C P_\infty C^T + R)^{-1} C P_\infty + Q, \quad (7.30)$$

which may be found through standard techniques [GS14]. The simplest method is to iterate the Kalman covariance update until convergence. After convergence to P_∞ , the Kalman update reduces to a non-adaptive observer

$$\hat{x}_{k+1|k} = \hat{x}_{k|k-1} + Bu_k + K_\infty[y_k - C\hat{x}_{k|k-1}] \quad (7.31)$$

where

$$K_\infty = P_\infty C^T (R + C P_\infty C^T)^{-1}. \quad (7.32)$$

7.4 Linear-quadratic regulators

A feedback law selects a sequence of control updates u to regulate the error state x to some fixed value, usually $x = 0$. A very simple control law would be to always pick the next control such that $Bu_{k+1} = -x_k$, i.e. set the error state to 0. In the absence of model uncertainty, drift, observation noise, or other stochastic effects, then this is the best one can do. However, we generally operate with uncertain models and stochastic fluctuations in parameters, which should be accounted for with regularized control laws. One way to accomplish this is to include a cost on the controls as well as the errors. In this way, one penalizes large controls, which results in smoother trajectories. This is the idea of the linear quadratic regulator (LQR). The LQR assume a linear system model and a quadratic cost on the errors and controls to derive an optimal control law with minimal cost.

The *linear-quadratic regulator* (LQR) is a fundamental technique in classical control theory. One may liken the LQR of classical control theory with the harmonic oscillator of quantum physics; both techniques provide a first approximation or toy model for later, more sophisticated analysis. LQR theory has been very well studied [A J08], with numerous generalizations. LQR may be understood as a first step to elegant model-based control techniques⁵, which might be augmented with modern reinforcement learning (RL) algorithms.

⁵Model-based control is a large field. In fact, the later combination of a dual Kalman filter with LQR (Tab. 7.5) may be understood as a model-based control technique as it goes beyond textbook theory.

Discrete-time, finite-horizon (DT-FH)	
System model	$x_{k+1} = x_k + Bu_k$
Cost function	$J = x_K^T Q_K x_K + \sum_{k=0}^{K-1} (x_k^T Q x_k + u_k^T R u_k)$
Control gain	$F_k = (R + B^T P_{k+1} B)^{-1} B^T P_{k+1}$
Riccati iterate	$P_{k-1} = P_k - P_k B (R + B^T P_k B)^{-1} B^T P_k + Q$
Discrete-time, infinite-horizon (DT-IH)	
System model	same as DT-FH
Cost function	$J = \sum_{k=0}^{\infty} (x_k^T Q x_k + u_k^T R u_k)$
Control gain	$F = (R + B^T P B)^{-1} B^T P$
Riccati equation	$P = P - P B (R + B^T P B)^{-1} B^T P + Q$
Continuous-time, finite-horizon (CT-FH)	
System model	$\dot{x} = x + Bu$
Cost function	$J = x^T(\tau) Q_\tau x(\tau) + \int_0^\tau (x^T Q x + u^T R u) dt$
Control gain	$F = R^{-1} B^T P(t)$
Riccati equation	$-\dot{P}(t) = 2P(t) - P(t) B R^{-1} B^T P(t) + Q$
Continuous-time, infinite-horizon (CT-IH)	
System model	same as CT-FT
Cost function	$J = \int_0^\infty (x^T Q x + u^T R u) dt$
Control gain	$F = R^{-1} B^T P$
Riccati equation	$2P - P B R^{-1} B^T P + Q = 0$

Table 7.1: Formulas for linear quadratic regulation. See also [A J08]. Here, Q is the state cost, and R is the control cost.

By formulating the quantum calibration problem as an LQR problem, I provide theoretical foundation to derive very efficient classical calibration algorithms of quantum devices that must one day scale to thousands and millions of qubits.

The linear-quadratic assumptions of LQR theory involve: 1) a linear system model and 2) a quadratic cost on the errors x and the controls u . When these assumptions are satisfied, then the LQR control law is the *optimal* controller⁶ that minimizes the cost. An LQR controller may be deployed in discrete- or continuous-time and with a finite or infinite horizon. Discrete-time LQR appropriately models control updates after batches to tomographic data, and continuous-time might be adapted to work with single-shot updates. Tab. 7.1 summarizes the various formulations of LQR control.

In any formulation the final feedback control law provided by an LQR approach has the form

$$u_k = -G_k \hat{x}_{k|k} \quad (7.33)$$

⁶It is usually very difficult – if not impossible – to identify an optimal control law, so the fact that LQR provides a simple formula to write an optimal control law that requires only linear algebra is extremely useful.

ie. a linear function of the estimated state. Here, the estimate is assume to be an optimal estimate given k observations and $k - 1$ previous control steps.

7.5 Dual control estimation

To derive an optimal control law, we assume knowledge of the control model B . However, as I have said, there are elements of B that are not known *a priori* and must be estimated by exploring the effects of control actions on error parameters. One approach would be to form an estimate of B by linear regression on an initial exploration dataset, described below, and then exploit that estimate for control. However, this approach would be undesirable in a context where device time is limited, and furthermore, naive linear regression does not account for the fact that the *the control model itself may drift* over time. As an alternative, one can automatically estimate the control model by observing the effects of the LQR control law calculated with a prior estimate of B , updating B based on the observed effects, and recalculating a new control law with the updated B . One approach, described below, is a dual Kalman filter. Including an on-line dual control estimator in a calibration loop provides an algorithm with the ability to self-regulate.

Estimating B with linear regression

An naive approach to estimate B is linear regression. This approach is not suitable for real-time regulation, but a batched estimation of B can be useful as an initial exploration step or verify to ensure the linear assumption. Furthermore, understanding the regression structure required to estimate B is a necessary first step to derive a dual Kalman filter for this problem.

The problem of estimating B can be formulated as an optimization problem with a cost function

$$J(B) = \sum_k |x_{k+1} - (x_k + Bu_k)|^2 \quad (7.34)$$

where x_k is estimated at each step. The optimal B that minimizes the cost is the control model that best fits the trajectory of x given a sequence of control actions u . The specific control actions may be selected according to various different strategies, including selecting u on a grid or by sampling Gaussian random variables.

We may reformulate the cost function Eq. 7.34 as a linear regression problem. Let Δ be a column vector of vertically-stacked innovations:

$$\Delta = \begin{bmatrix} x_{k+1} - x_k \\ x_k - x_{k-1} \\ \vdots \\ x_2 - x_1 \end{bmatrix}. \quad (7.35)$$

Furthermore, let $\mathcal{U}_k = \mathbb{1}_m \otimes u_k^T$ (see ⁷) where m is the length of u and $\mathbb{1}_m$ is an $m \times m$ identity matrix, so that $\mathcal{U}_k \vec{B} = Bu_k$, where $\vec{B}$ is a vectorized form of the matrix B . Finally, define $\mathcal{U} = \sum_k \mathcal{U}_k$. It follows that

$$\Delta = \mathcal{U} \vec{B}. \quad (7.36)$$

In this way, we may estimate $\vec{B}$ as $\mathcal{U}^{-1} \Delta$ for a sufficiently large number of control actions so that $\mathcal{U}$ is invertible.

Dual Kalman filter

A more attractive approach to estimating B is to do so based on streams of data collected by using prior estimates of B to minimize the system error x . We employ a dual Kalman filter [WN03] for this problem. The operation of the Kalman filter in this context can be thought of as streaming linear regression.

To derive the dual Kalman filter, we define our dual observations to be innovations $\Delta_k = x_{k+1} - x_k$, which are related to the dual state $\vec{B}$ by the relation

$$\Delta_k = \mathcal{U}_k \vec{B} + v_k \quad (7.37)$$

where v_k is a random variable that represents the uncertainty of an RPE estimate of the innovation Δ_k . Then, the Kalman update equations that provide an optimal estimator $\hat{B}_k$ of $\vec{B}$ and its uncertainty Σ_k are

$$\hat{B}_{k+1} = \hat{B}_k + K_{k+1}(\Delta_{k+1} - \mathcal{U}_{k+1} \hat{B}_k) \quad (7.38)$$

$$\Sigma_{k+1} = (\mathbb{1} - K_{k+1} \mathcal{U}_{k+1}) \Sigma_k \quad (7.39)$$

where K_k is the dual Kalman gain

$$K_k = \Sigma_{k-1} \Delta_k^T (\Delta_k \Sigma_{k-1} \Delta_k^T + R_k)^{-1} \quad (7.40)$$

and R_k is the covariance of RPE observations, that may be set to twice the square of the standard RPE uncertainty Eq. 5.20.

To initialize a dual Kalman filter to estimate B , we require an initial prior $\hat{B}_0$ and uncertainty P_0 . This may be defined in at least two ways: 1) using physical intuition about the effects of controls or 2) starting with a small linear regression experiment. The prior estimate may have substantial error, and the dual estimate will correct this error in real time.

⁷ $\mathbb{1} \otimes u_k^T$ is the tensor product of an identity matrix and a row vector, which is a rectangular matrix with rows that contain u_k padded by 0's, as $\mathbb{1}_m \otimes u_k^T = \begin{bmatrix} u_k & \vec{0} & \dots & \vec{0} \\ \vec{0} & u_k & \dots & \vec{0} \\ \vdots & \vdots & \ddots & \vdots \\ \vec{0} & \vec{0} & \dots & u_k \end{bmatrix}$

Our dual estimation approach may also be straightforwardly amended to allow for the possibility of drift in the control model itself. This is done, per the usual Kalman recipe, by adding a matrix Q to the dual Kalman covariance Σ_k before each update. In practice, one should be careful about how this is done. We do not expect B to drift on the time-scale of a calibration iterate, but we might expect B to change over long time periods between calibration iterates. To model this effect, we add uncertainty to Σ_k that grows like the time between calibration (and dual estimation) cycles.

7.6 Calibration algorithm

I now combine the principles of control law design discussed above to derive a an efficient algorithm for calibration of quantum device. The algorithm consists of three core parts:

1. a classical observer
2. a control law
3. a dual estimator.

These parts are general for any quantum calibration problem. The observer estimates the system error x , the control law minimizes the error state x with controls actions u , and the dual estimator infers the control model from the effects of u and x . The algorithm presented here solves these three parts of the problem independently, and the separation principle and the certainty-equivalence principle imply that the resulting controller will perform well.

To solve the three parts of a quantum calibration problem, I use:

1. an RPE-based system observer of coherent errors
2. a LQR-based control law
3. an estimated system model,

which are combined to yield a high-precision calibration algorithm.

RPE-based system observer

I use RPE techniques to infer coherent error parameters (see also §5). An RPE experiment measures a vector of physical phases y that are related to the error parameters. Let us assume that we are working with a model such that the output of the estimator are equal to the error parameters themselves. Then the relevant observation model is simply

$$y_k = x_k + v_k. \quad (7.41)$$

This form of observation model described the output of an RPE estimation algorithm. The error bounds are encoded in the covariance of the observation noise v_k .

Given this form of RPE observations and assuming time-invariant control, the relevant classical observer model is

$$\hat{x}_{k+1|k+1} = \hat{x}_{k|k} + Bu_k + K(y_{k+1} - (\hat{x}_{k|k} + Bu_k)) \quad (7.42)$$

for some gain K .

No smoothing

Choosing the maximal gain $K = \mathbb{I}$ reduces the observer model to

$$\hat{x}_{k+1|k+1} = y_{k+1}, \quad (7.43)$$

which corresponds to performing control directly on estimated RPE phases without smoothing. This would be appropriate in the absence of system drift or observation uncertainty. Including a smoothing observer model provides smoother trajectories.

Kalman smoothing

A Kalman filter can be employed on top of RPE estimates to smooth the observations with prior data. In the usual way, define a prior with mean $x_{0|0} = 0$ and some initial covariance P_0 . Then in our case, the associated Kalman gain is written⁸

$$K_k = (P_{k-1} + Q)(P_{k-1} + Q + R)^{-1} \quad (7.44)$$

and the adaptive observer model is

$$\hat{x}_{k+1|k+1} = \hat{x}_{k|k} + Bu_k + K_{k+1}(y_{k+1} - (\hat{x}_{k|k} + Bu_k)). \quad (7.45)$$

The covariance updates are given

$$P_{k+1} = (\mathbb{I} - K_{k+1})(P_k + Q). \quad (7.46)$$

Control law

The control law defines a map between observations and control updates. Each control update u_k is chosen to fix the estimated error state based on knowledge of the prior history of observations and controls $\mathcal{Y}_k, \mathcal{U}_{k-1}$. Operating in the usual LQG framework, the optimal control law has the form

$$u_k = -G\hat{x}_{k|k}. \quad (7.47)$$

Producing a new configuration

$$\theta_{k+1} = \theta_k + u_k. \quad (7.48)$$

The expected effect on the error state is

$$x_{k+1} = x_k - G\hat{x}_{k|k}. \quad (7.49)$$

⁸The usual definition... reduces to...

Estimating the control model

In order to derive an LQG controller, one must know the control model B that maps configuration updates to parameter effects. On-line dual control techniques, such as a dual Kalman filter, are possible. However, below I consider first performing an initial experiment that measures B . This initial experiment consist of selecting a grid of points in configuration space and collecting data from phase estimation with each of the configurations. This amounts to essentially measuring the slopes of different lines.

A simple technique to measure B is to consider a perturbation c_j (a positive 1-hot vector) to each configuration parameter $\theta \mapsto \theta \pm c_j$. If the perturbation is chosen sufficiently small that a linear approximation holds, then the row of B corresponding to parameter j can be estimated as

$$[\hat{B}]_j = \frac{\Xi(\theta + c_j) - \Xi(\theta - c_j)}{2|c_j|}. \quad (7.50)$$

A particularly elegant formalism of the initial dual control experiment can be formulated with sigma-point Kalman filtering techniques. In this way, define a prior “covariance” matrix Σ to encode the ranges over which the control model is expected to be linear. A good definition would be

$$\Sigma = \text{diag}(c_1^2, c_2^2, \dots, c_n^2) \quad (7.51)$$

where c_j is the maximum variation over parameter j that is expected to exhibit linear dynamics. Given a Σ , the a batched dual estimation experiment can be formulated as observation of

$$\theta^{\pm j} = \theta_0 \pm s_j \quad (7.52)$$

where s_j is a sigma-point, for example using a cubature approach (Tab. 6.3). In the case of a diagonal covariance matrix, the sigma-points are 1-hot vectors with magnitude c_j located at parameter j .

The final estimate of B can then be found via linear regression following §7.5

$$\hat{B} = \mathcal{U}^+ \Delta. \quad (7.53)$$

The natural questions would then be to discuss goodness-of-fit and bound linearization errors.

Explicit algorithms

The analysis above yields three explicit algorithms for regularization of performance parameters. I present three version of the algorithm with increasing complexity and capability:

1. Direct LQR (without observation smoothing)
2. Kalman filtered LQR (LQG control)
3. dual LQG control (with a streaming dual estimator)

Name	Reference	Description
Direct LQR	Tab .7.3	Direct application of LQR on RPE phases. Uses no observation smoothing. Assumes known (estimated), static control model. Does not account for state drift or observation noise, but tuning of LQR hyper-parameters permits robustness to these effects.
Kalman LQR (aka. LQG)	Tab .7.4	Application of LQR on Kalman filtered phases. Assumes known, static control model. Models drift and observation noise.
Dual LQG	Tab .7.5	LQG control with a dual Kalman filter for control model estimation. Learns an unknown, drifting control model. Models drift in both the control model and the state as well as observation noise.

Table 7.2: Summary of the three algorithms presented in this chapter.

The basic algorithm, Tab. 7.3, deploys LQR directly on RPE observations without any smoothing. Including a Kalman filter on top of the RPE observations may provide some increased performance guarantees. Finally, a dual Kalman filter can also be deployed to provide the algorithm with an ability to self-regulate. These algorithms illustrate how the principles and classical techniques discussed in this chapter can be combined to novel application in quantum processor performance regularization.

Simulations (see below) indicate that adding a Kalman filter provide little benefit in simple control cases. However, I expect that larger regularization algorithms will have to build sophisticated model-based control loops based with three components: 1) control law, 2) observation smoothing, 3) dual estimators, illustrated in these algorithms. Dual estimators will becomes essential when the control model drifts on timescales comparable to regularization intervals.

7.7 Numerical implementation

I implemented a simple python notebook to evaluate a test these techniques. I considered a simple 1-parameter experiment relevant to tracking qubit frequency as well as regulating gate over-rotation with amplitude (or timing) adjustments. I also derived a 3-parameter implementation relevant to the controlled-Z gate calibration problem §5.2.

In each protocol, I consider 1) an initial experiment that measures the control model using linear regression then 2) deploy an LQR control law based on observations of phase estimation circuits. I compare the performance of LQR control on raw RPE estimate trajectories as well as with a Kalman filtered version. I found little performance difference (less than an order of magnitude) between the raw and Kalman filtered versions, indicating the performing LQR

Require: Hyper-parameters N, M	▷ LQR matrices
Require: Control model B	
Require: Update interval τ	
1: $G \leftarrow \text{LQRGain}(B, N, M)$	
2: $\theta_1 \leftarrow$ initial configuration	
3: $k \leftarrow 1$	
4: while regulating do	
5: procedure OBSERVATION	
6: $y_k \leftarrow \Xi(\theta_k) + v_k$	▷ device observation
7: end procedure	
8: procedure LQR CONTROL	
9: $u_k = -Gy_k$	▷ control law
10: $\theta_{k+1} = \theta_k + u_k$	▷ configuration update
11: end procedure	
12: $k \leftarrow k + 1$	
13: wait (τ)	
14: end while	

Table 7.3: Phase-based linear-quadratic regularization without smoothing.

regularization directly on RPE phases is a promising protocol for real devices.

Single-parameter regulation

The 1-parameter protocol considers the usual linear model

$$x_{k+1} = x_k + Bu_k + w_k \quad (7.54)$$

$$y_k = x_k + v_k \quad (7.55)$$

where B is here a scalar parameter that relates configuration changes to effects on RPE phases. In the case of Ramsey RPE, B is equal to $2\pi\tau$ where τ is the delay time of the gate. In the case of Rabi RPE in superconducting device, B is related to qubit-drive coupling rates and should be measured experimentally. It is likely that B can be derived from first-principles in ion-traps.

Experimental measurement of B

To estimate the control model in the simple 1-parameter setting, I collected experimental data on a subset of qubits in a real superconducting device at the advanced quantum testbed

Require: Hyper-parameters P_0, Q, R, N, M	▷ Kalman and LQR matrices
Require: Control model B	
Require: Update interval τ	
1: $G \leftarrow \text{LQRGain}(B, N, M)$	
2: $\theta_0 \leftarrow$ initial configuration	
3: $\hat{x}_{0 0} \leftarrow 0$ initial estimate	
4: $k \leftarrow 0$	
5: while regulating do	
6: procedure LQR CONTROL	
7: $u_k = -G\hat{x}_{k k}$	▷ control law
8: $\theta_{k+1} = \theta_k + u_k$	▷ configuration update
9: end procedure	
10: procedure OBSERVATION	
11: $y_{k+1} \leftarrow \Xi(\theta_{k+1}) + v_{k+1}$	▷ device observation
12: end procedure	
13: procedure KALMAN SMOOTHING	
14: $K_{k+1} \leftarrow (P_k + Q)(P_k + Q + R)^{-1}$	
15: $\hat{x}_{k+1 k+1} \leftarrow \hat{x}_{k k} + Bu_k + K_{k+1}[y_{k+1} - \hat{x}_{k k} - Bu_k]$	
16: $P_{k+1} \leftarrow (\mathbb{I} - K_{k+1})P_k$	
17: end procedure	
18: $k \leftarrow k + 1$	
19: wait (τ)	
20: end while	

Table 7.4: Phase-based regularization with Kalman filtering. When $k = 0$, the observation update step $\hat{x}_k \leftarrow \dots$ should be replaced with $\hat{x}_0 \leftarrow y_0$ as there is no previous control or estimate.

Require: Hyper-parameters $P_0, Q, R, N, M, \Sigma_0, Q_D, R_D,$	$\triangleright$ Kalman, LQR, and dual
Kalman matrices	
Require: Control model estimate $\hat{B}_0$	
Require: Update interval τ	
1: $\theta_0 \leftarrow$ initial configuration	
2: $\hat{x}_{0 0} \leftarrow 0$ initial estimate	
3: $k \leftarrow 0$	
4: while regulating do	
5: procedure LQR CONTROL	
6: $G_k \leftarrow \text{LQRGain}(\hat{B}_k, N, M)$	
7: $u_k = -G_k \hat{x}_{k k}$	$\triangleright$ control law
8: $\theta_{k+1} = \theta_k + u_k$	$\triangleright$ configuration update
9: end procedure	
10: procedure OBSERVATION	
11: $y_{k+1} \leftarrow \Xi(\theta_{k+1}) + v_{k+1}$	$\triangleright$ device observation
12: end procedure	
13: procedure KALMAN SMOOTHING	
14: $K_{k+1} \leftarrow (P_k + Q)(P_k + Q + R)^{-1}$	
15: $\hat{x}_{k+1 k+1} \leftarrow \hat{x}_{k k} + \hat{B}_k u_k + K_{k+1}[y_{k+1} - \hat{x}_{k k} - \hat{B}_k u_k]$	
16: $P_{k+1} \leftarrow (\mathbb{I} - K_{k+1})P_k$	
17: end procedure	
18: procedure DUAL KALMAN UPDATE	
19: $\Delta_{k+1} \leftarrow \hat{x}_{k+1 k+1} - x_{k k}$	$\triangleright$ innovation
20: $K_{k+1}^D \leftarrow (\Sigma_k + Q)\Delta_{k+1}^T(\Delta_{k+1}(\Sigma_k + Q)\Delta_{k+1}^T + R_D)^{-1}$	$\triangleright$ dual gain
21: $\mathcal{U}_k = \mathbb{I} \otimes u_k^T$	
22: $\hat{B}_{k+1} = \hat{B}_k + K_{k+1}(\Delta_{k+1} - \mathcal{U}_k \hat{B}_k)$	$\triangleright$ dual update
23: $\Sigma_{k+1} = (\mathbb{I} - K_{k+1} \mathcal{U}_k)\Sigma_{k-1}$	$\triangleright$ dual covar. update
24: end procedure	
25: $k \leftarrow k + 1$	
26: wait (τ)	
27: end while	

Table 7.5: Phase-based regularization with on-line dual estimation. When $k = 0$, the observation update step $\hat{x}_k \leftarrow \dots$ should be replaced with $\hat{x}_0 \leftarrow y_0$ as there is no previous control or estimate.

(AQT) at LBNL. Fig. 7.4 depicts results obtained for Ramsey experiments and Fig. 7.5 depicts results obtained for X-gate experiments.

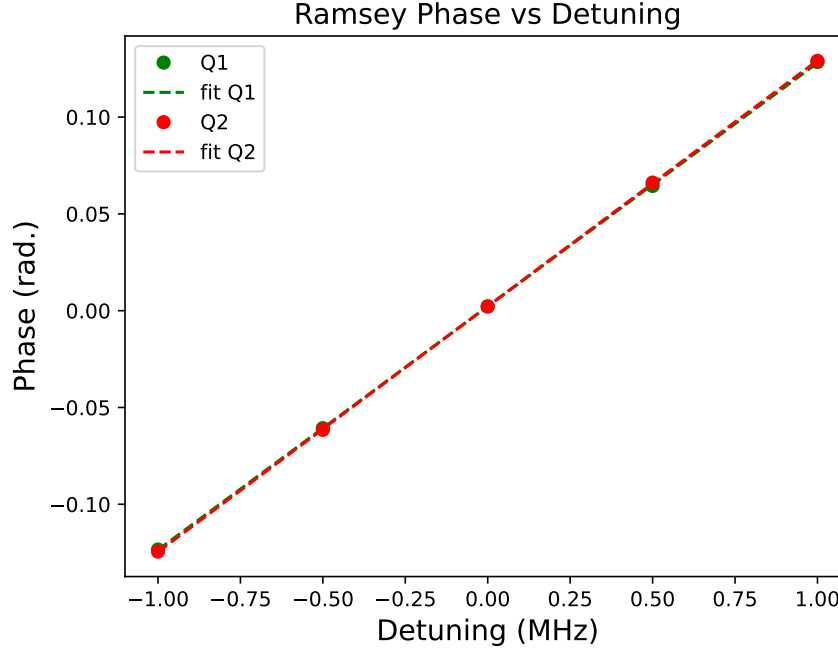


Figure 7.4: *Dual control estimation for Ramsey experiments with batched regression.* A grid of detunings was selected and each point was evaluated with Ramsey RPE experiments. The estimated phases clearly exhibit linear dynamics with detunings in this range. The slope was estimated on Q1 to be $0.1257(2)$ and on Q2 to be $0.1269(2)$. The idle gate was taken at 20 ns, so the expected slope is $40\pi \text{ ns} \approx 0.1257$, which exhibits good agreement with measured values.

Verification of Gaussian observation model

In order to deploy a Kalman filter as described above, it is necessary to verify that RPE observations are indeed distributed according to a Gaussian random distribution with well defined covariance. I did this numerically by simulating a simple 1-parameter RPE experiment in pyGSTi. In this case, I considered an idle gate with phase error and X and Y gates (for Ramsey state preparation) with no coherent error. I included an overall gate depolarization rate of 10^{-3} and SPAM depolarization rates of 10^{-2} . The results are shown in Fig. 7.6.

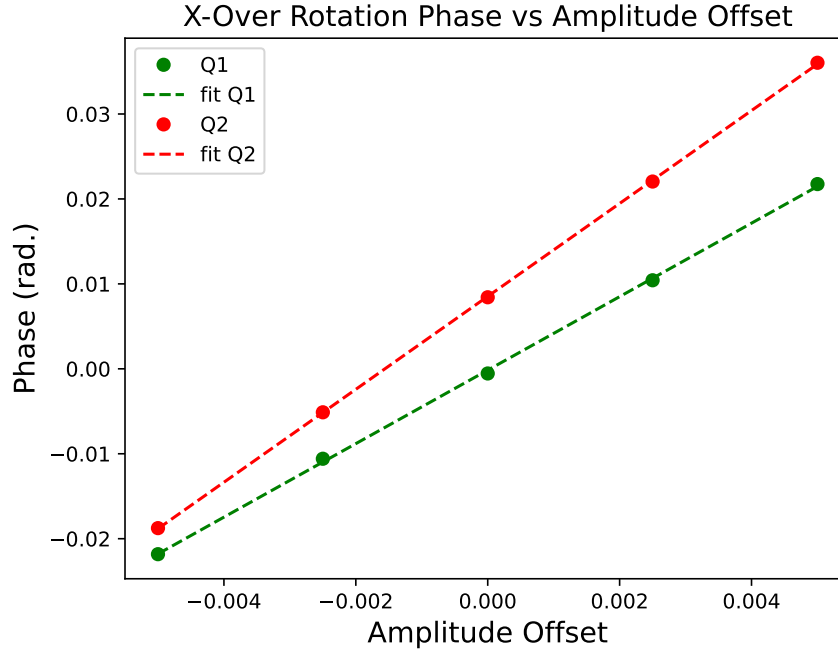


Figure 7.5: *Dual control estimation for X-gate over-rotation experiments with batched regression.* A grid of amplitude offsets (in device units) was selected and each point was evaluated with X-gate RPE experiments. The estimated phases clearly exhibit linear dynamics with amplitude changes in this range. The slope was estimated on Q1 to be $4.3262(2)$ and on Q2 to be $5.470(9)$. These results indicate that qubit-drive coupling rates differ for different qubits in a superconducting device.

Regulating under drift

Next, I consider regulating a single phase under coherent parameter drift. I consider deploying LQR on RPE observations without smoothing, using a gate set with a single drifting coherent parameter and uniform depolarization rates of 10^{-3} for gates and 10^{-2} for SPAM. The initial error is assumed relatively large (to simulate a large initial error, eg., after a cool down cycle) and then subject to stochastic noise sampled from a Gaussian distribution. Fig. 7.7 depicts the results. Bryson’s rule was used to set the error cost matrix in the LQR, and the control cost was chosen to depict performance in two regimes.

Based on these results, we see that the control cost matrix has significant impact on the long-term performance of a regulation algorithm. When controls are very expensive (under-damped), then the controller takes only small updates in configuration space that do not fully account for the effects of the noise. In the limit of infinitely expensive controls, no regulation is performed and the system evolves under free evolution. A well-tuned LQR can regulate drift in the system up to the covariance of the noise.

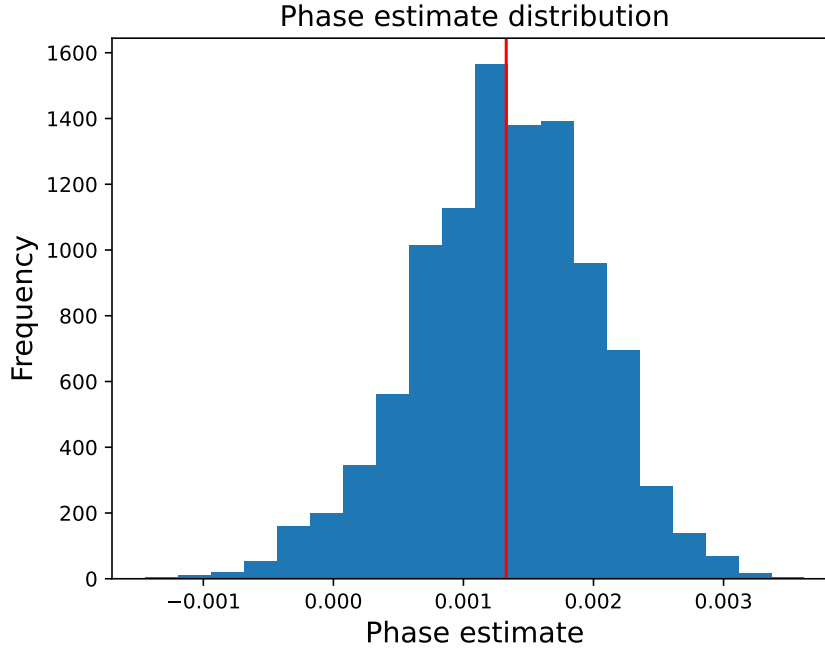


Figure 7.6: *Histogram of samples collected from 1-parameter RPE observations.* 10,000 samples were collected from a gate set with coherent idle error of 1.329×10^{-3} (chosen arbitrary) on one gate and uniform depolarization on the other gates (see main text). An RPE experiment was simulated up to a maximum depth of 2^9 with 50 shots per circuit. The distribution is well approximated by a Gaussian. The standard deviation was calculated to be 6.8×10^{-4} , which should be compared to $\pi/(2^9\sqrt{50}) \approx 8.7 \times 10^{-4}$.

Average performance comparisons

In my final 1-parameter experiment, I compare the average performance of LQR regulation with and without Kalman filtering on RPE observations. In this example, I use the same model as above, with the same initial coherent error, drift process, and depolarization. I simulate 100 random trajectories (with randomly sampled drift processes) with and without Kalman filtering and compare the RMSE in the underlying state. I set the hyper-parameters to reasonable values with Bryson’s rule and some tuning. The RMSE performance is depicted in Fig. 7.8. One can observe that the average RMSE in a Kalman filtered and direct LQR control approach are essentially identical in this example. Each regulate the coherent drift in the phase up to the standard deviation of the drift process (recall that $Q = 10^{-4}$ so 10^{-2} is the expected noise floor).

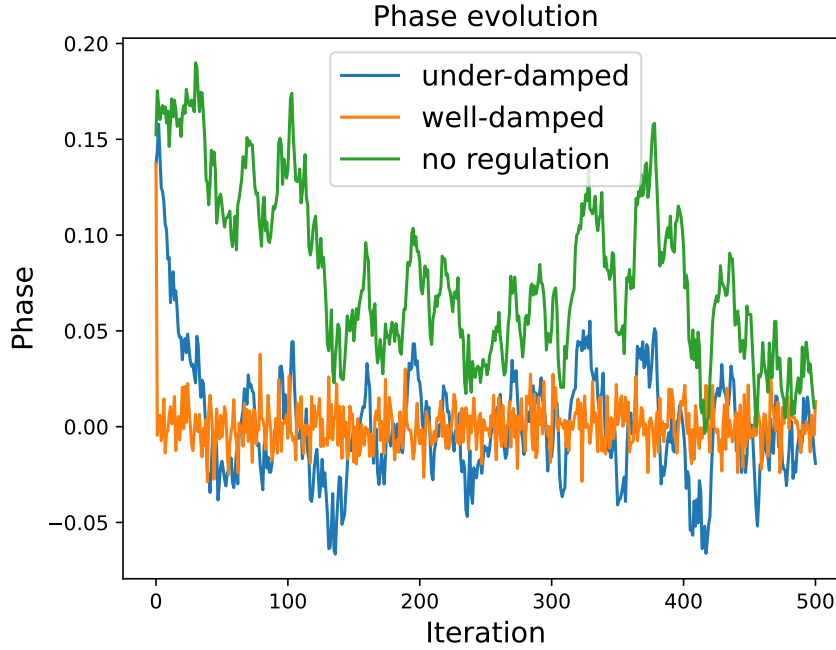


Figure 7.7: *Performance of LQR regulation in two regimes.* An LQR experiment is simulated in pyGSTi using a known control model (with no estimate errors). The coherent parameter being regulated is subject to a drift process of covariance 10^{-4} per step. An under-damped LQR control law assumes a large control cost, which is not responsive enough to regulate drift. Choosing a lower control cost yields better performance. In the limit of infinite cost on controls, one recovers free evolution under the drift process.

Multi-parameter regulation

The effects of 1-parameter configuration update u do not always decouple into 1-parameter effects in state space x . Sometimes (see next paragraph) it is possible to design multiple 1-parameter servos to solve a coupled parameter problem, but this will usually be sub-optimal. All the theory developed in this chapter made no assumptions on the size of the state space x or control space u . Thus, the control law design principles developed here provide a more direct approach to solve multi-parameter coupling by making a single, simultaneous control update to all coupled parameters. I now present some experiment results and numerical studies that support a multi-parameter servo for regulation of CZ gate performance.

We saw an example of multi-parameter coupling in the CZ RPE (§5.2), where changes in the drive amplitude affect the ZZ -coupling as well as a local IZ -phase and ZI -phase. In [Rud+25] this was dealt with by designing a set of three 1-parameter regulation protocols: The first tunes the ZZ -phase through amplitude changes, which results in random IZ - and ZI -phases, and the second fixes the local phases through phase shifts. However, because

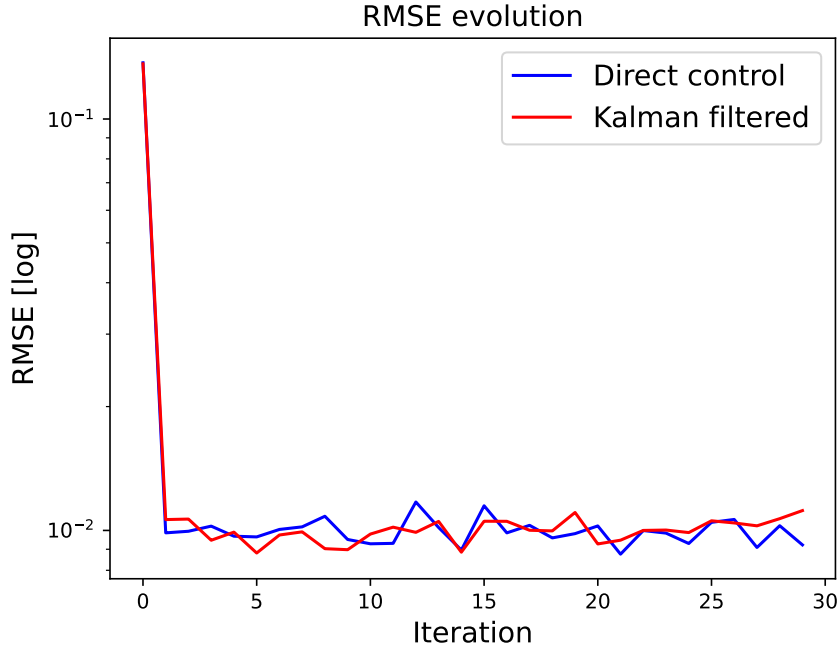


Figure 7.8: *Evolution of the RMSE over 1000 random trajectories with and without Kalman filtering*

the 3-parameters are coupled in state space, all three must be estimated in each of the 1-parameter experiments, which results in inefficient experimental overhead. Furthermore, it is generally not possible to decompose all multi-parameter couplings into 1-parameter servos. The only reason it was possible for the CZ gate is that phase shifts only affect local phases (their action is decoupled), but there are likely other cases where this is not true.

In the case of the CZ gate, amplitude affects all three phases $\theta_{IZ}, \theta_{ZI}, \theta_{ZZ}$, and local phases shifts only affect local phases. A linear model that describes this is

$$Bu = \begin{bmatrix} 1 & 0 & \eta_{IZ} \\ 0 & 1 & \eta_{IZ} \\ 0 & 0 & \eta_{ZZ} \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \alpha \end{bmatrix} \quad (7.56)$$

where φ_1, φ_2 are local phase shifts on qubits 1 and 2 and α is amplitude change of the drive. To understand the 1's on the diagonal, recall that local phase shifts affect result in additive effects on local phases.

To verify linearly, I measured the effects of amplitude shifts on the same Transmon chip used for the 1-parameter studies. The results are shown in Fig. 7.9. One can observe that the linear model is satisfied in this parameter regime and that the amplitude shifts affect local phases more strongly than the ZZ entangling phase.

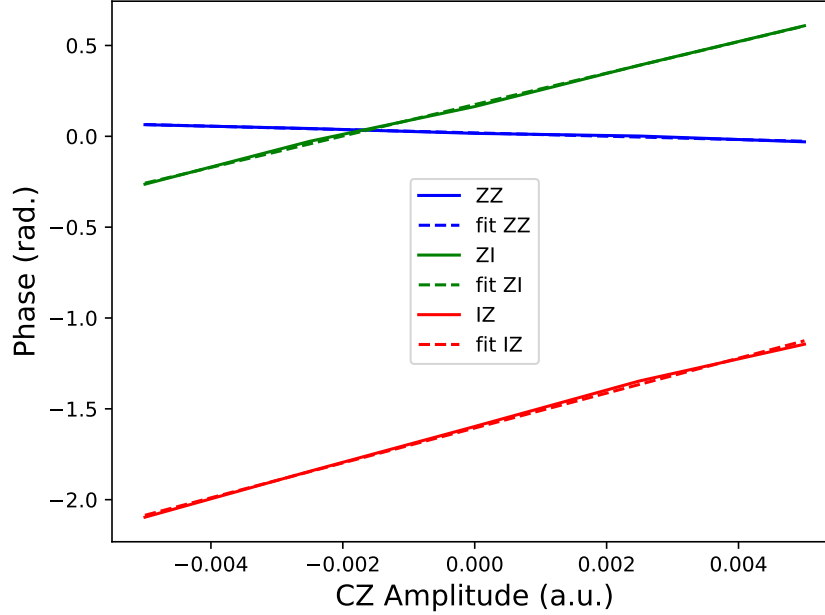


Figure 7.9: *Dual control estimation for CZ gate phases with batched regression.* This data was taken on a Transmon device and verifies that a linear model holds in the parameter regime considered.

Next, I simulated 100 random trajectories with drift in the error state starting from some fixed random initial state. I deployed direct LQR control on each trajectory. The average performance is shown in Figs. 7.10, 7.11. These results indicate that direct LQR can regulate the local and entangling phases of the CZ gate simultaneously.

7.8 Discussion

In this chapter, I described how a number of ideas in classical control theory can be applied to quantum processor regulation. I derived classical system models of quantum devices that can be regulated with existing techniques, and I derived a dual Kalman filter to estimate control models in real devices in a streaming fashion. These results were synthesized in explicit algorithms that were tested numerically.

I described an explicit solution to multi-parameter control coupling with an LQR. This approach solves a common issue in processor calibration and permits efficient deployment on large-scale devices.

Another common issue on real device is over-determined control, where there are more control inputs than state space parameters. For example, in the CZ gate, both drive ampli-

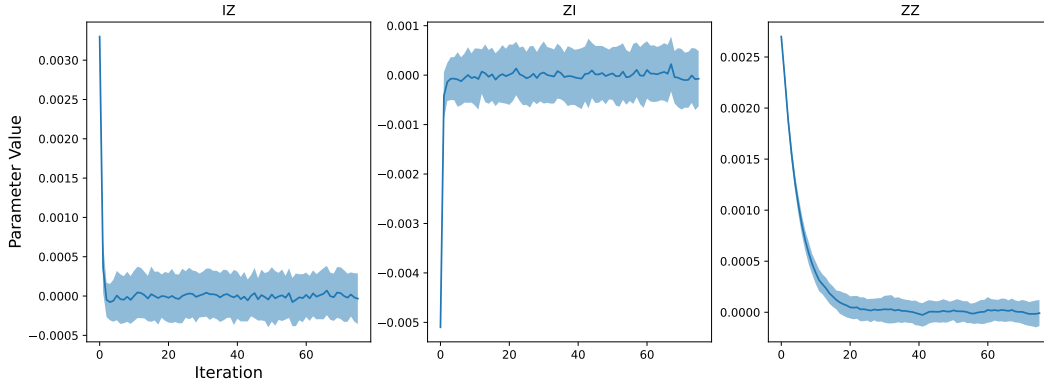


Figure 7.10: *Average performance of direct regulation.* 100 random trajectories were simulated with a drift covariance of 10^{-6} uniform on the three phases. The RPE experiments were taken up to max depth 2^9 with 50 shots per circuit. The gate set had uniform depolarizing noise at 10^{-3} on each gate and no other out-of-model errors.

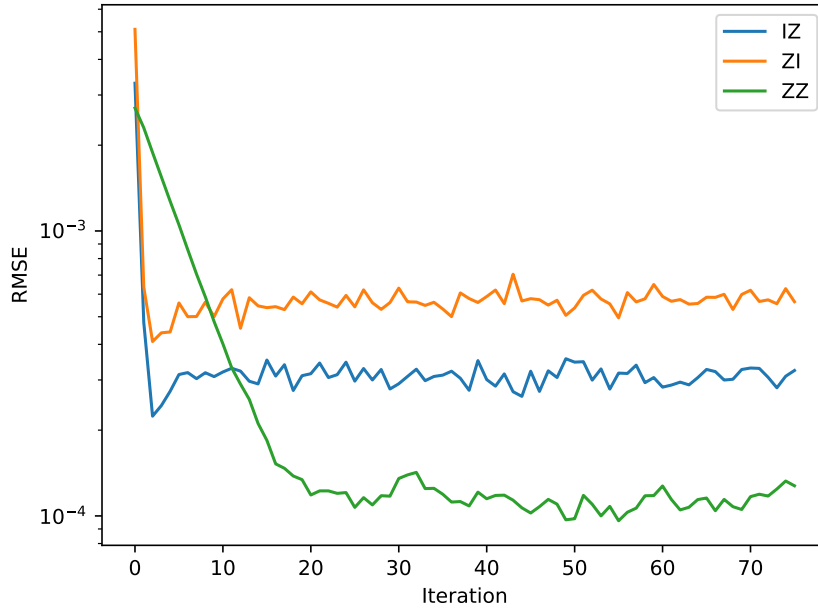


Figure 7.11: *RMSE evolution over 100 random trajectories.* The RMSE in each time step calculated on the data of Fig. 7.10. The specific control model used in simulation had a slightly larger ZI effect from drive amplitude variations, which explains the higher noise floor. This higher noise floor illustrates how stochastic noise can couple across multiple parameters.

tude and frequency shifts affect the ZZ phase. We solved this by fixing the frequency and just varying the drive amplitude. However, this is not the only possible approach. The LQR can still be applied in a case where the control update vector u is larger than the state space x .

The techniques introduced in this chapter should be understood as foundational algorithms for large system development. While some experimental evidence has shown that linear models work in certain regimes, non-linear models will likely be required in some circumstances. The techniques here provide a first approximation that can later be built upon with sophisticated model-based control.

Part IV

Correction

Chapter 8

Topics in quantum error correction

Quantum error correction (QEC) codes are thought to be necessary to ensure reliable, large-scale quantum computation on noisy physical devices. The theory of QEC is already vast [LB13] and considers codes of both applied and pure interest defined on as few as two-qubits to the infinite-limit quantum register. Codes may generally be non-linear [Rei+21]; however, it is thought that codes of practical utility will be linear quantum codes – specifically “stabilizer” codes, which are the focus on this work.

The generic operation of a QEC code consists of a sequence of basic cycles (Fig. 8.1): 0) an initial encoding, 1) syndrome extraction, 2) decoding, 4) logical operation. The initial encoding prepares the logical 0 state in the code space. The syndrome extraction cycles measure stabilizers on the quantum device. The decoding cycles infer error effects from syndrome information and are done on classical hardware. Logical operation cycles perform fault-tolerant transformations of logical states. After encoding, the other cycles are iterated continuously.

Syndrome measurement and logical operations are usually considered separately in QEC code design. Whereas syndrome measurement and decoding extracts entropy from a quantum state, logical operation performs a (large) transformation. Memory cycles *extract entropy* from encoded information; Logical cycles *apply transformations* to encoded information. This distinction is often made in the literature, where one discusses a *memory* paradigm (where it is desired to preserve a static state) and a *logical operation* paradigm (where it is desired to correctly transform the state). Co-design of good memory and logical protocols is a fundamental problem in practical QEC implementation, and there are many open questions around this intersection. This work focuses almost entirely on the memory paradigm.

Numeric simulations of a code can be performed with increasing realism and associated complexity. The three usual divisions in the literature are: 1) code capacity, 2) phenomenological, and 3) circuit level. I find it useful to further sub-divide the circuit level model into A) circuit level capacity and B) fault tolerant circuit level to distinguish circuit level error propagation on ancilla qubits. These models are defined in Fig. 8.2.

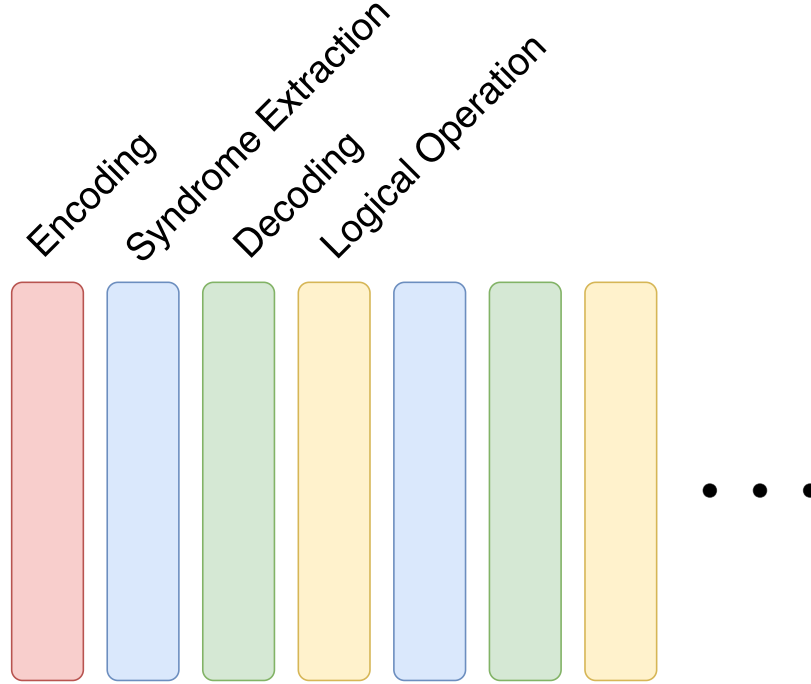
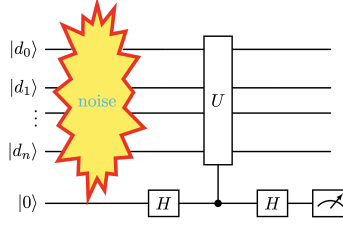


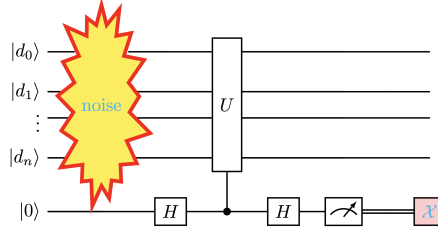
Figure 8.1: *Cycles of QEC code operation.* Operation of a QEC code can be modelled as a sequence of discrete cycles. An initial encoding cycle prepares the logical 0 state. Syndrome extract measures stabilizers and performs the subspace projection. Decoding infers the logical effect from syndrome information. The logical operation perform transformations on the encoded logical dof. The cycles are iterated over. Sometimes, it is useful to perform multiple rounds of syndrome extraction before logical operation. In the memory paradigm, no logical operation is performed, and syndrome extraction is only interleaved with decoding.

8.1 Stabilizer codes

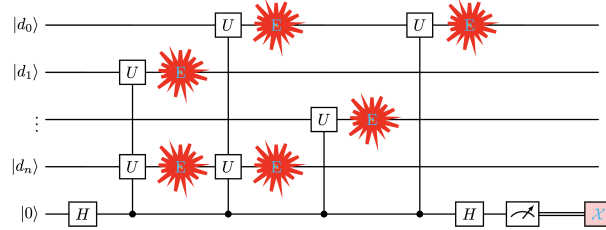
A stabilizer code is the most widely studied form of quantum error correction. The first examples of stabilizer codes were discovered by Shor and Steane [Sho95; Ste96] and theory was first formalized by Gottesmann [Got97]. Stabilizer codes have since seen wide development [LB13] and experimental demonstrations [Rei+24; Ngu+21; Goo25]. The original formulation of stabilizer codes deals with a partitioning of Hilbert space into a subspace structure. It was later realized that a further partitioning of the logical subspace into two subsystems could be useful, which led to the notion of “operator” or subsystem QEC [Pou05; Kri+05]. Subsystem codes combine the original notion of subspace QEC with that of decoherence free subspaces [Lid12] that are represented as arbitrary “gauge” degrees of freedom. It should be



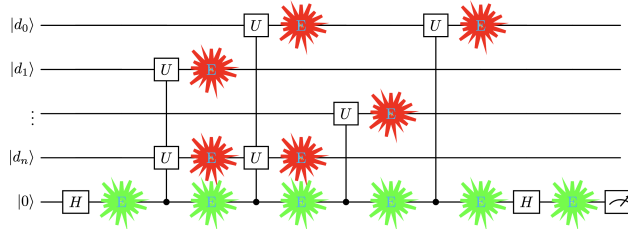
(a) *Code capacity model of QEC.* A Pauli stochastic noise channel across the entire register is sampled before perfect syndrome measurement with ancilla qubits.



(b) *Phenomenological model of QEC.* As in the code capacity model, a noise channel is sampled before syndrome measurement. After this, a classical bit flip channel is applied to the syndrome measurement outcomes to represent faults in measurements.



(c) *Circuit level phenomenological model of QEC.* Noise is simulated at the circuit level, with specific noise channels occurring after each gate in the circuit. Noise can also be included on idles (not shown). A classical bit flip channel is applied to syndrome measurement outcomes.



(d) *Circuit level fault tolerance model of QEC.* A full circuit-level simulation is performed with quantum noise on all components including on the ancilla (shown in green). Here, errors can propagate from the ancilla to the data register through the controlled operation, which is mitigated through fault tolerant circuit design.

Figure 8.2: *Four models for classical simulation of QEC codes.*

understood that all stabilizer codes are subspace codes, and gauged codes contain a further subsystem structure.

A stabilizer code provides an encoding that maps a state on some register to an encoded state in a larger register. An additional decoding mechanism maps erroneous states to the nearest logical state after the system is subjected to noise. A code accomplishes this by distributing logical information across a highly non-local subspace of the total Hilbert space. The logical subspace is spread out across the whole Hilbert space of the register with non-local symmetries. Hence, most local errors¹ map a state outside of the logical subspace and represent an error. After correctly decoding a syndrome, the local error can be corrected for – possibly by applying a non-local logical transformation.

In the embedded codespace, the logical states are separated by a characteristic code distance. The *distance* of a quantum stabilizer code is the minimum Hamming weight over Pauli-operators that map a logical state to a different logical state. The distance of a code quantifies the maximal weight of Pauli errors that are promised to be corrected. Given a code with distance d , the maximum promised correctable (MPC) weight is $\lfloor (d-1)/2 \rfloor$ and the maximum promised detectable (MPD) weight is $d-1$, which follow from a simple geometric argument, see Fig. 8.3 and caption. All errors less than the MPC weight are correctable and all errors less than the MPD weight are detectable. It is important to understand that there will generally be errors with weight greater than the MPC weight that are still correctable (however, the code will fail for other errors with the same weight).

A labeling scheme for stabilizer codes can be useful. A standard approach is to prescribe the label $[[n, k, d]]$ to a stabilizer code that consists of k logical qubits encoded into n physical qubits with distance d . For subsystem code with gauge qubits, one may use the notation $[[n, k, d, g]]$ for g the number of gauge qubits. If g is not included, it is assumed 0. The number of stabilizers is $n - k - g$. An obvious question is whether the $[[n, k, d]]$ notation uniquely labels a stabilizer code. However, this is not the case, as it is possible for two codes to have the same number of stabilizer but different stabilizer groups, e.g., the two versions of a 9-qubit Shor code² that are both $[[9, 1, 3]]$ codes.

Subspace projection

As I said, stabilizer measurement projects on a subspace structure. This is one of the core conceptual concepts in QEC, and I now develop this mathematically.

A stabilizer code partitions a Hilbert space $\mathcal{H}$ into its code part C and its orthogonal complement (“redundant” or syndrome part) $C^\perp$ as

$$\mathcal{H} = \mathcal{H}^C \oplus \mathcal{H}^{C^\perp}. \quad (8.1)$$

¹Here, the local vs. non-local divide is relative to physical locations associated to the actual arrangement of qubits in a register in a processor.

²One Shor code follows from concatenation of a bit-flip with a phase flip and the other follows from concatenation in the opposite direction.

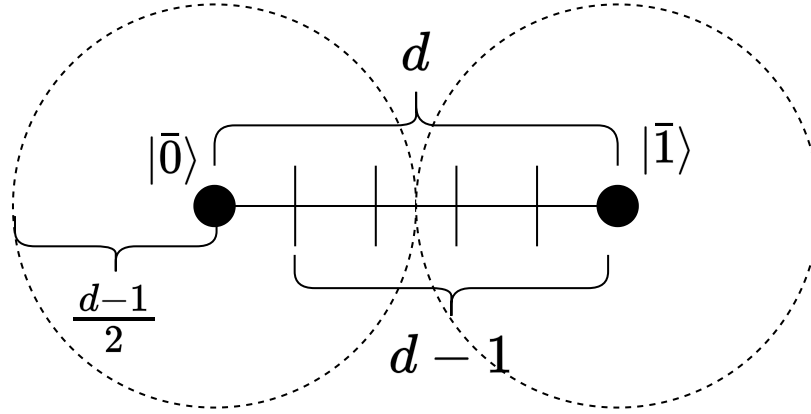


Figure 8.3: *Visualization of the code distance in QEC.* Logical states $|\bar{0}\rangle$ and $|\bar{1}\rangle$ are separated by Hamming distance 5 in this example. We see that an error of weight greater than 2 will map one logical state into the radius of the other logical state, which will cause the decoder to map the resulting state to the other logical state, implying a logical error. However, any error that is not a logical operation (i.e. maps one logical state to another) will be detectable.

C contains the logical information of a quantum state and $C^\perp$ may be understood as redundancy used to protect against error. Now, consider a set of commuting Pauli operators $\mathcal{G}$ that generate a stabilizer group. Let the number of generators be written $m = |\mathcal{G}|$. Measurement of all the stabilizer generators $\mathcal{G}$ acts like a projection in the subspace $\mathcal{H}^{C^\perp}$. To see this, model stabilizer measurement as a PVM (projection valued measure § 3) over the set of projectors

$$\{\Pi_y : y \in \mathcal{Y}\} \quad (8.2)$$

where $\Pi_y \in \mathcal{B}(\mathcal{H}^{C^\perp})$ is a projector on the subspace labeled by the syndrome y and $\mathcal{Y}$ is the set of all syndrome strings (of size 2^m). The syndrome may be represented as a length- m binary vector with elements $y(i)$. A syndrome subspace projector Π_y may be decomposed into a tensor product structure of an identity on the codespace and a sequence of rank-1 projectors on the complementary space

$$\Pi_y = \mathbb{1}^C \otimes |y(1)\rangle\langle y(1)| \otimes |y(2)\rangle\langle y(2)| \otimes \dots \otimes |y(m)\rangle\langle y(m)| \quad (8.3)$$

where $\mathbb{1}^C \in \mathcal{H}^C$. In this way, given a general state $\rho \in \mathcal{H}$, measurement of the stabilizer generators with outcome syndrome y corresponds to a map

$$\rho \mapsto \text{Tr}_{C^\perp} [\rho] \otimes |y\rangle\langle y|. \quad (8.4)$$

See also § 8.5.

The syndrome projector Π_y in Eq. 8.3 is written in the codespace basis, where the state is neatly separable into a logical and a syndrome part. One may also write the syndrome projector in the physical basis through a change of basis

$$\Pi'_y = B\Pi_y B^{-1}. \quad (8.5)$$

The physical basis is in the basis in which errors are written, and the codespace basis is the basis in which the syndrome projectors are separable. Stabilizer codes generally have a Clifford encoding, so B is usually an element of the Clifford group that normalizes the Pauli group.

An equivalent decomposition of a syndrome subspace projector may be obtained

$$\Pi_y = \prod_{g_i \in \mathcal{G}} \left(\frac{\mathbb{1} + (-1)^{y_i} G_i}{2} \right) \quad (8.6)$$

where G_i the Pauli operator that represents the stabilizer generator indexed by i and y_i is the associated syndrome bit.

Subsystem factorization

A subsystem code results from further factorizing the code subspace $\mathcal{H}^C$ into two subsystems: the logical and gauge subspace. The total Hilbert space decomposition is then

$$\mathcal{H} = \mathcal{H}^A \otimes \mathcal{H}^B \oplus \mathcal{H}^{C^\perp}. \quad (8.7)$$

Here, $\mathcal{H}^A$ represents the logical components, $\mathcal{H}^B$ represents the gauge components, and $\mathcal{H}^{C^\perp}$ represents the stabilizer component. The logical and gauge spaces are subsystems of the code space $\mathcal{H}^A \otimes \mathcal{H}^B = \mathcal{H}^C$.

The **gauge group** of a code is the group generated by the gauge operators. Unlike the stabilizer group, the gauge group is non-Abelian. The gauge group is generated by a canonical set of Z-type and X-type operators that obey usual Pauli commutation relations. In general, for a code with l gauge qubits, there are $2l$ gauge generators.

Like stabilizers, gauge operators can be measured. However, the gauge group is non-Abelian, and measurement of non-commuting sets of operators will be non-deterministic. If one measures a complete set of commuting gauge operators, then the resulting measurement will project general states onto the gauge space. Some proposals [LB13] use gauge measurements to infer stabilizer measurements, which can help reduce the weight of stabilizer measurements. High-weight stabilizer measurements can be challenging in physical hardware.

Gauged codes encode logical dof. into noiseless subsystems for passive protection of logical information; in a subsystem code, an error that is equivalent to a gauge operator does not change the logical state. In this way, gauged codes are related to noiseless subsystem and decoherence free subspaces [Kri+05].

There is an interesting relationship between code performance and gauge dof. [Sur24]. For codes with the same number of logical dof. and the same number of physical qubits, introducing more gauge degrees of freedom comes at the cost of less stabilizer dof. Having less stabilizer dof. means that less information is known about the errors, yet having more gauge dof. means that some errors do not affect the logical state.

The identification of a gauge group can be used to implement fault tolerance protocols. For example, in code deformation [Vui+19] one invokes a code's gauge group to perform fault-tolerant operations on the logical subspace. Code deformation is equivalent to lattice surgery [Vui+19] and is of a similar nature to magic state distillation. It would seem that gauge qubits are useful for practical fault tolerance and implementation of logical operation.

Furthermore, gauge groups are relevant to discussions of dynamical qubits [AR24], such as the Floquet codes of recent discovery [HH21].

Codespace operators and destabilizers

A useful mental model for stabilizer codes can be developed through the notion of codespace operators. This is the observation that the properties of a stabilizer code are most conveniently discussed in a natural operator basis analogous to the canonical Pauli basis associated with a register of qubits. Recall that the usual Pauli basis is generated by a set of X- and Z-type operators $\bigcup_i \{X_i, Z_i\}$ that anti-commute in pairs and otherwise commute for $i \neq j$. An analogous basis can be constructed for a QEC code. Here, one understands a codespace operator as a change of basis from the physical Pauli operators of X, Z to encoded operators. These may be long, non-local Pauli operators when written in the physical basis, but they obey the same canonical commutation relations.

This idea of a codespace operator basis is best understood by associating logical, syndrome, and gauge dof. to physical qubits in the register and then performing a change of basis. The physical operators associated with the logical dof. are mapped to the logical X- and Z-operators of the code, which clearly exhibit canonical commutation relations. Similarly, the physical operators associated with the gauge dof. are mapped to the gauge X- and Z-operators. Finally, the operators associated with syndrome dof. are mapped to the stabilizers *and a canonical anti-commuting pairing* called “destabilizers.” If stabilizers are understood as the encoded Z-type operators for the associated dof., then the destabilizers are the corresponding X-type operators. Destabilizers anti-commute with their associated stabilizer and commute with all others including logical and gauge operators.

The notion of destabilizers can be extremely useful when reasoning about code operation and decoding. Usually, a code is defined via a list of logical operators and stabilizers. The destabilizers are then defined implicitly as the unique operators³ that satisfy canonical

³The destabilizers are fixed uniquely after fixing X- and Z-type logical operators. Indeed, a logical operator may always be multiplied by a stabilizer to find an equivalent logical operator. In this process, the corresponding destabilizer must be changed to satisfy canonical commutation relations. Alternatively, one may fix a Clifford encoding that maps physical X- and Z-type operators to codespace X- and Z-type operators.

commutation relations. Explicit construction of a codespace operator basis can be useful in simulations and analysis of QEC codes. For example, the destabilizer part of an error determines the syndrome pattern that is measured and can be used to fix the syndrome part (see eg. [DP10a]).

Error discretization

An important and somewhat subtle concept in quantum error correction is that of *error discretization*. While quantum channels are continuous operations, the QEC codes we have defined are discrete objects defined on a finite syndrome alphabet. A QEC code effectively transforms the continuous operations of a quantum noise channel into discrete operations. One can show that the choice of basis for the noise channel is arbitrary (Theorem 10.2 of [NC02]); if a code can correct for a set of Pauli errors $\mathcal{Q}$ that support⁴ a general error E , then the code can correct for the effect of E . The proof of this fact essentially argues that any incoherent noise channel can be diagonalized with a unitary transformation and written as a diagonal Pauli transfer matrix. Hence without loss of generality, we analyze stabilizer codes in terms of distributions on Pauli errors.

Knill-Laflamme conditions

The *Knill-Laflamme conditions* [KL97] provide necessary and sufficient conditions under which a subspace error correcting code completely corrects a quantum noise channel. Consider a quantum noise channel $\mathcal{K}$ composed of Kraus operators K_s , i.e. the map

$$\mathcal{K}[\rho] = \sum_s K_s \rho K_s^\dagger. \quad (8.8)$$

The Knill-Laflamme conditions state that a code with stabilizers $\mathcal{S}$ completely corrects for the effects $\mathcal{K}$ iff

$$\Pi^C K_i^\dagger K_j \Pi^C = \alpha_{ij} \Pi^C \quad (8.9)$$

for some Hermitian matrix α of complex numbers (Theorem 10.1 of [NC02]).

The matrix α provides some insight into the nature of a code, and I will call the “degeneracy matrix.” If the degeneracy matrix is full-rank, then all errors in the set $\mathcal{K}$ are distinguishable. However, unlike in classical error correction, some errors in QEC may be degenerate. Two errors **degenerate** if they are non-distinguishable yet still correctable. In such a case, α will not have full rank but still be Hermitian.

Degeneracy

As already noted, QEC codes exhibit a unique, non-classical property of *degeneracy* with respect to correctable errors. This is the property that multiple *correctable* errors can have

⁴This means that $\mathcal{Q}$ should be such that when the error E is written in the Pauli basis, each non-zero element is contained in $\mathcal{Q}$.

the same syndrome. All codes, whether classical or quantum, will exhibit degeneracy with respect to the set of all possible errors; it is always possible to find at least two errors with the same syndrome for a given code. However, quantum codes can be degenerate with respect to the set of correctable errors. In other words, quantum codes can sometimes correct different errors with the same syndrome. This is a uniquely quantum phenomenon. No classical code can correct for two different errors with the same syndrome.

A simple example of a degenerate quantum code is Shor’s 9-qubit code (in either concatenation order or “bias”). Consider the bias where the Z-type stabilizers appear like the 6 cyclic permutations of $ZZIIIIII$ and the X-type stabilizers are $XXXXXXII$ and $IIIXXXXX$. Here, a phase flip (Z) error on any of the first three qubits will all trigger the same stabilizer and hence have the same syndrome. However, all three errors are correctable with the same correction (that can be chosen to be a phase flip on any of the first three qubits) because any weight-2 Z-type Pauli operator on the first 3 qubits is equivalent to a stabilizer. The same story goes for the middle three qubits and the final three qubits. Hence, this code is “degenerate” because the set of correctable errors contains multiple elements with the same syndrome.

Degeneracy is both a boon and a bane for QEC code design. Clearly, it is good that more errors can be corrected with the same amount of syndrome information. However, degeneracy can cause issues with decoding. In fact, non-degenerate QEC decoding is easier than degenerate decoding [IP15]. A longstanding question in the field has been whether degeneracy can be built into decoding schemes naturally. The challenge has been that many of our quantum decoding schemes still carry over intuition from classical QEC, where degeneracy does not exist.

8.2 Tableau representation of a stabilizer code

The Clifford tableau representation of stabilizer codes is very convenient for numerical simulations of code performance. In the tableau representation, the Hilbert space partitioning $\mathcal{H}^C \oplus \mathcal{H}^{C^\perp}$ is associated with two sub-registers of an n -qubit input register and a *Clifford encoding operator* maps states in the computational basis to states in the *encoded basis*, see Fig. 8.4. Stabilizer measurements are then represented as operations on the input register in the CHP formalism §2.4 and noise is simulated through stochastic sampling of a Pauli channel §3.4.

The encoding Clifford can be used to “back-out” the effect of noise and calculate effects of Pauli errors on the various parts of the code. Consider a Pauli noise channel acting on a codestate, described by the channel

$$\begin{aligned} \mathcal{N}(|\phi(\psi, 0)\rangle\langle\phi(\psi, 0)|) &= \sum_j p_j \mathbf{P}_j B |\psi\rangle\langle\psi| \otimes |0\rangle\langle 0| B^{-1} \mathbf{P}_j \\ &= \sum_j p_j B \text{Ad}_B^{-1}(\mathbf{P}_j) |\psi\rangle\langle\psi| \otimes |0\rangle\langle 0| (B \text{Ad}_B^{-1}(\mathbf{P}_j))^\dagger \end{aligned} \quad (8.10)$$

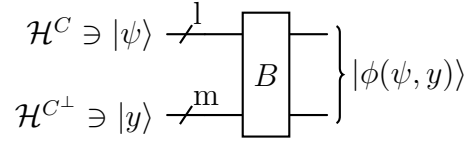


Figure 8.4: *Codespace partitioning of Hilbert space.* An initial separable state (in the codespace) is mapped through an encoding unitary B to a state in the physical basis. In the physical basis the state is highly entangled across the register. In the codespace basis, the state is separable. The encoder can be understood as a change of basis, and it is usually a Clifford operation.

where $\text{Ad}_B^1(\mathbf{P}_j) = B^{-1}\mathbf{P}_jB$. In this way, one can see that the effects of the noise on the code subspaces are described by a new Pauli $\mathbf{Q}$ in the codespace basis (which is also separable!) that is the adjoint of the original n -qubit Pauli in the physical basis

$$\mathbf{Q}_j = \text{Ad}_B^{-1}(\mathbf{P}_j). \quad (8.11)$$

Here each index the Pauli string can be associated with a logical, gauge, or stabilizer dof.

The encoding Clifford may also be used to directly prepare encoded states; one writes a circuit compilation of the encoding Clifford and operates on the physical register with the derived circuit. However encoding operations are themselves noisy, so an alternative fault-tolerant codespace preparation circuit routine is usually run, see §8.6.

8.3 Codespace entropy calculus

Errors that develop in QEC codes appear as changes in the logical and stabilizer degrees of freedom (I will ignore gauge dof. in the following). It is desirable to have a calculation formalism to concisely model these changes. Codespace entropy, as defined here, can be used to quantify average variations in the codespace of a QEC code subject to a particular noise distribution. These techniques are foundational to my arguments in §9.

Codespace error distributions

I begin by introducing notation and calculation techniques for codespace errors distribution arising from Pauli stochastic noise. In the following, let X be a random variable that labels a Pauli error sampled from the noise channel, Y be a random variable that labels the syndrome of a Pauli error $\mathbf{P}_X$, and Z be a random variable that indexes the logical effect of the error $\mathbf{P}_X$. Y and Z are found by calculating commutation relations between the error and the logical operators. Observe that X fixes $Y = y(X)$ and $Z = z(X)$.

Recall the commutation indicator used throughout this work

$$\bar{\delta}[X, Z] = \begin{cases} 0 & \text{if } [X, Z] = 0 \\ 1 & \text{if } [X, Z] \neq 0. \end{cases} \quad (8.12)$$

I will be somewhat loose with this notation in the following. In particular, I will use $\bar{\delta}[a, b]$ to represent the commutation indicator both when a, b are operators and labels of operators.

The syndrome is given by the commutation relations

$$y(X) = \bigotimes_{\mathbf{S} \in \mathcal{S}_C} \bar{\delta}[\mathbf{S}, \mathbf{P}_X] \quad (8.13)$$

and the logical effect by

$$z(X) = \bigotimes_{\mathbf{L} \in \mathcal{L}_C} \bar{\delta}[\mathbf{L}, \mathbf{P}_X], \quad (8.14)$$

resp. that produce binary strings that represent commutation relations between the error $\mathbf{P}_X$ and the stabilizers $\mathcal{S}$ and the logicals $\mathcal{L}$. $y(X)$ has length equal to the number of syndromes, $|y(X)| = n - k$ for k logical qubits, and $z(X)$ has length equal to twice the number of logical dof. (for X and Z type logicals), $|z(X)| = 2k$.

Assume a Pauli noise channel with error distribution $\Pr(X)$. The implied joint distribution between errors, logical effects, and syndromes is

$$\Pr(X, Y, Z) = \Pr(X) \mathbb{1}(Y = y(X)) \mathbb{1}(Z = z(X)) \quad (8.15)$$

where $\mathbb{1}$ is an indicator function that is 1 if the clause is true else 0.

All of the following entropy calculations derive from the marginals of the total joint distribution Eq. 8.15. The logical-stabilizer marginal is the probability that a stabilizer and logical part appear together under all possible physical errors:

$$\Pr(Z, Y) = \sum_{x \in \mathcal{X}} \Pr(X = x) \mathbb{1}(Z = z(x)) \mathbb{1}(Y = y(x)). \quad (8.16)$$

The logical marginal is the probability that a logical part appear under all possible physical errors:

$$\Pr(Z) = \sum_{x \in \mathcal{X}} \Pr(X = x) \mathbb{1}(Z = z(x)) \quad (8.17)$$

and similarly for the syndrome marginal

$$\Pr(Y) = \sum_{x \in \mathcal{X}} \Pr(X = x) \mathbb{1}(Y = y(x)). \quad (8.18)$$

The conditional probability of logical on the stabilizer will be of significant interest:

$$\begin{aligned} \Pr(Z|Y) &= \frac{\Pr(Z, Y)}{\Pr(Y)} \\ &= \frac{\sum_{x \in \mathcal{X}} \Pr(X = x) \mathbb{1}(Z = z(x)) \mathbb{1}(Y = y(x))}{\sum_{x \in \mathcal{X}} \Pr(X = x) \mathbb{1}(Y = y(x))} \end{aligned} \quad (8.19)$$

Codespace entropy

The logical effects Z represent the effects of errors X when restricted to the codespace. Operating over a large number of cycles, a particular error realization $x^{(n)}$ results in a sequence of logical effects $z^{(n)}(x^{(n)})$. In the large n limit,

$$\Pr(Z^n = z^{(n)}) \rightarrow 2^{-nH(Z)}. \quad (8.20)$$

In this way, we have defined a notion of entropy generation in the codespace.

Joint logical-syndrome entropy

The logical effects Z and syndromes Y are generated jointly by errors X . Operating over a large number of cycles, a particular error sequence $x^{(n)}$ results in a joint sequence $z^{(n)} \times y^{(n)}$ according to the joint logical-syndrome marginal. In the large n limit,

$$\Pr(Z^n = z^{(n)}, Y^n = y^{(n)}) = 2^{-nH(Z,Y)}. \quad (8.21)$$

This notion defines a rate of entropy generation in both the logical and syndrome spaces.

Conditional codespace entropy

After a sufficient number of samples from a Pauli channel, only typical syndrome sequences $y^{(n)}$ occur and each occurs with a typical subset of all possible logical sequences $z^{(n)}$. Hence, in the large n limit

$$\Pr(z^{(n)}|y^{(n)}) \rightarrow 2^{-nH(Z|Y)}. \quad (8.22)$$

$H(Z|Y)$ may be calculated from $P(Z|Y)$ or from $H(Z,Y) - H(Y)$. The conditional codespace entropy quantifies the residual entropy in Z after knowing Y .

Codespace information

The mutual information between Z and Y quantifies how much information a syndrome carries about the logical effect Z . The mutual information may be calculated as the difference between codespace and conditional codespace entropy

$$I(Z; Y) = H(Z) - H(Z|Y). \quad (8.23)$$

8.4 Decoding

A decoder attempts to infer a correction operation based on syndrome information. If the decoder is successful, the product of the error that occurred and the recovery operation will be equivalent to a stabilizer. Mathematically, decoding can be represented as a map $\hat{R}$ from

a syndrome vector to a Pauli recovery; ie. from the binary field on m syndrome outcomes to the Pauli group on n qubits:

$$\hat{R}: \mathbb{F}_2^m \rightarrow \mathcal{P}^n. \quad (8.24)$$

Given an error $\mathbf{P}_X \in \mathcal{P}^n$ and observed syndrome $y(X) \in \mathbb{F}_2^m$, a decoder is correct if the product of the error and the decoder's recovery operation has the proper commutation relations with the logical operators and the syndromes; ie.:

$$[\hat{R}E, L] = 0 \quad \forall L \in \mathcal{L}, \quad (8.25)$$

$$[\hat{R}E, S] = 0 \quad \forall S \in \mathcal{S}. \quad (8.26)$$

So we see that the decoder must fix both the logical and the syndrome part. The syndrome part is the easy part to fix; a list of the code's destabilizers can be used to do this directly. Inferring the logical effect to fix the logical part is the fundamental challenge in real decoding.

Decoders are often designed assuming a uniform depolarizing model over Pauli errors. In this model, the probability of an error is entirely determined by its Hamming weight. However, under highly biased noise models, decoders designed for uniform depolarization may perform sub-optimally. Biased decoder assumes a prior distribution on Pauli errors to infer the most probable action that will return a correct recovery.

Constructing a good decoder can be computationally challenging. In fact, the general problem of optimal decoding of random codes is very hard [IP15]. Optimal decoding requires optimizing the probability of successful decoding over every possible input string. There are 4^n possible Pauli errors on n qubits and 2^m syndrome strings for an m -stabilizer code, so the problem size grows very quickly. Below, I discuss several decoding strategies, both of theoretical and practical interest.

Look up table decoding

The simplest decoding approach is the “look up table error” approach, which is appropriate for small codes. Here, Pauli errors are listed in descending order according to their probability of occurrence, and the syndrome of each effect is calculated. Then, a new table is constructed from each unique syndrome of highest weight. In practice, one can stop calculating syndromes of Paulis after all unique syndromes have been resolved for likely errors. For example, in a 3-qubit bit flip code (assuming stabilizer generators $G_1 = IZZ, G_2 = ZZI$) under depolarizing error with rate $p < 0.5$ one lists the distribution on 4^3 Paulis

Error	Prob	Syndrome
<i>III</i>	$(1-p)^3$	(0, 0)
<i>IIX</i>	$(1-p)^2p$	(0, 1)
<i>IIZ</i>	$(1-p)^2p$	(0, 0)
<i>IXI</i>	$(1-p)^2p$	(1, 1)
<i>IZI</i>	$(1-p)^2p$	(0, 0)
<i>XII</i>	$(1-p)^2p$	(1, 0)
<i>ZII</i>	$(1-p)^2p$	(0, 0)
	$\vdots$	

and the Pauli-look-up table is then the map defined by:

$$\begin{aligned}
\hat{R}(0, 0) &= I \\
\hat{R}(0, 1) &= X_3 \\
\hat{R}(1, 0) &= X_1 \\
\hat{R}(1, 1) &= X_2.
\end{aligned}$$

Here, I ignored Y type errors even though they are as equally probable as X under depolarizing noise. A bit flip code cannot distinguish Y -type from X -type errors.

Optimal error decoding

The optimal error decoder, aka. the maximum likelihood⁵ of error decoder, selects the correction $\hat{R}(y)$ by solving

$$\hat{R}(y) = \operatorname{argmax}_x \Pr(x|y). \quad (8.27)$$

This decoder finds the error that is compatible with y and has the highest probability of occurring. One can observe that optimal error decoding is equivalent to look up table decoding. The construction of an optimal error decoder is NP-complete [IP15].

Optimal error class decoding

Because quantum codes can be degenerate, selecting the most probable Pauli operator is not always sufficient for optimal decoding. The most probable Pauli decoder does not take into account the fact that two physically distinct errors may have the same syndromes and effect on the logical state and hence be correctable by the same operation. In certain cases, ignoring this fact can lead to sub-optimal decoding strategies. One of the cases that leads to different

⁵Those familiar with Bayesian analysis may notice that a maximum likelihood technique should formally optimize the likelihood $L(x|y) = \Pr(y|x)$ rather than the conditional probability as done here. However, $\Pr(y|x)$ is deterministic given x . For this reason, it is common in the literature (eg. [IP15]) to refer to this sort of decoding scheme as “maximum likelihood” even though it is maximum a posteriori decoding.

performances is when two degenerate errors have lower probability of occurring than another error with a different logical effect, but the sum of the probabilities of the degenerate errors is higher than the probability of the other error. In the case of vanishingly small error rate, degenerate error class decoding and maximum Pauli error decoding are equivalent [IP15]. However, for finite error rates, degenerate error class decoding can outperform Pauli error decoding.

The optimal error class decoder first infers the logical part of the error from the syndrome then constructs a correction that corrects both the logical and the syndrome part. First, the decoder identifies the most likely logical effect given a syndrome

$$\hat{z}(y) = \operatorname{argmax}_{z \in \mathcal{Z}} \Pr(z|y) \quad (8.28)$$

where $\Pr(z|y)$ is the probability of a logical effect given a syndrome, which may be evaluated as

$$\Pr(z|y) = \frac{\sum_{x \in \mathcal{X}} \Pr(X = x) \bar{\delta}[z, x] \bar{\delta}[y, x]}{\sum_{x \in \mathcal{X}} \Pr(X = x) \bar{\delta}[y, x]}. \quad (8.29)$$

Finally, the recovery operation is $\hat{z}(y)T(y)$ where $T(y)$ is the associated destabilizer for syndrome y .

Jointly typical decoders

A final decoder of theoretical interest that will play a key role in our analysis of § 9 is the jointly typical decoder. This decoder is somewhat different in nature from the single-shot decoders discussed above in that it applies only after a sequence of syndrome vectors have been collected for a sufficiently large number of syndrome measurements cycles. For this reason, construction of a jointly typical decoder is impractical in experiment. However, jointly typical decoding will turn out to be a very useful tool in theoretical bounds of best achievable code performance.

Given n cycles of syndrome measurement, the jointly typical decoder infers a sequence of logical effects $z^{(n)}$ given a sequence of syndrome vectors $y^{(n)}$. The logical component is selected if the following two conditions are satisfied: 1) $(z^{(n)}, y^{(n)})$ is jointly typical, and 2) $y^{(n)}$ is *not* jointly typical with *any* other typical logical effect sequence $z'^{(n)}$. See § 9 for a detailed discussion of this decoder and see § 2.5 for a discussion of joint typicality.

Efficient decoders

The optimal and jointly typical decoders discussed here are of theoretical interest, and they would not be suitable for real-time quantum error correction. In practical use cases, efficient decoders must be constructed. These decoders give up promises of optimality and settle for approximately optimal correction. As long as certain approximations are satisfied, efficient decoders can achieve high entropy extraction rates with significantly less computational resources. There are many proposed efficient decoders in the literature, and the design of an efficient decoder is usually specific to a particular QEC code.

The most prevalent efficient decoder is the minimum-weight, perfect-matching (MWPM) decoder [Hig21]. MWPM decoding is thought to be the correct approach for the surface code and has been extensively studied. Its protocol is based on an old algorithm in computer science [Edm65] for finding the maximum matching of a graph.

Another popular decoder of recent study is the union find decoder [DH21]. This decoder uses the union find algorithm to isolate clusters of errors and convert these into erasure locations. Erasure corrections can be computed more efficiently than bit- and phase-flip error corrections, so the union find algorithm is very efficient in classical resources.

Other popular decoding technique follow from Bayesian inference techniques. The belief propagation of classical decoding can be generalized to the quantum setting [LP08; KL23]. Here, a message passing algorithm is performed between nodes of a graph that represent the decoding problem, and a decision is made after a confidence threshold is reach. Another related technique is renormalization techniques [DP10b].

8.5 Channel representations of quantum codes

In the following, I derive channel representations of stabilizer measurement, logical measurement, and decoding.

Stabilizer measurement channel

I begin with the stabilizer measurement channel. It suffices to consider a gauge-free subspace code because the stabilizer measurement channel projects onto the logical subspace, and the gauge space can be understood as a subsystem of the logical subspace. To derive the channel, start by labeling each outcome of a stabilizer measurement with a binary vector y with elements y_j corresponding to the stabilizer generator G_j . Write the corresponding measurement projection effects⁶ as

$$|y\rangle\langle y| = |y_1\rangle\langle y_1| \otimes |y_2\rangle\langle y_2| \otimes \dots \otimes |y_m\rangle\langle y_m|. \quad (8.30)$$

that act on the stabilizer subspace $\mathcal{H}^{C^\perp}$ in the codespace basis. Next, define an embedding of each stabilizer projection effect in the codespace basis of the total Hilbert space $\mathcal{H}$ per

$$\Pi_y = \mathbb{1} \otimes |y\rangle\langle y|. \quad (8.31)$$

Here, $\mathcal{E}$ denotes that the projector is written in the encoded codespace basis. $\mathbb{1}$ acts on the logical subspace $\mathcal{H}^C$ of the subspace code partition $\mathcal{H} = \mathcal{H}^C \oplus \mathcal{H}^{C^\perp}$, which is untouched by stabilizer measurement. Next, convert the projection effects to the physical basis by application of the Clifford encoding operator B

$$\Pi'_y = B(\mathbb{1} \otimes |y\rangle\langle y|)B^{-1} \quad (8.32)$$

⁶See projection value measures PVMs in § 3.1.

Now, a stabilizer measurement of a general $\rho \in \mathcal{H}$ in the physical basis with measurement outcome y in the syndrome alphabet $\mathcal{Y}$ may then be represented as the map

$$\rho \mapsto \mathcal{S}_y[\rho] \otimes |y\rangle\langle y| \quad (8.33)$$

where

$$\mathcal{S}_y[\rho] = \text{Tr}_{\mathcal{H}^{C^\perp}} [\Pi'_y \rho] . \quad (8.34)$$

A stabilizer measurement applies a specific projection Π_y conditioned on having obtained outcome $y \in \mathcal{Y}$; this is equivalent to invocation of a quantum instrument [Wat18]. The quantum instrument that models stabilizer measurement is the quantum-classical channel

$$\Phi^{\mathcal{S}}(\rho) = \sum_{y \in \mathcal{Y}} \mathcal{S}_y[\rho] \otimes |y\rangle\langle y| . \quad (8.35)$$

One can understand the stabilizer quantum instrument as describing the situation in which one measures stabilizers but does not yet know the outcome that was measured. If one knows the outcome of stabilizer measurement; i.e., if one has knowledge of the syndrome, then the corresponding channel is

$$\Phi_y^{\mathcal{S}}[\rho] = \mathcal{S}_y[\rho] \otimes |y\rangle\langle y| . \quad (8.36)$$

Joint noise-syndrome channel

Writing the joint channel that describes application of Pauli noise followed by stabilizer measurement is illuminating and is done in the following.

The Pauli noise channel is

$$\mathcal{E}[\rho] = \sum_j p_j \mathbf{P}_j \rho \mathbf{P}_j \quad (8.37)$$

and the definite-outcome stabilizer measurement (Eq. 8.35) channel is

$$\Phi_y^{\mathcal{S}}[\rho] = \mathcal{S}_y[\rho] \otimes |y\rangle\langle y| .$$

Composition yields

$$\Phi_y^{\mathcal{S}}[\rho] \circ \mathcal{E} = \sum_j p_j \mathcal{S}_y[\mathbf{P}_j \rho \mathbf{P}_j] \otimes |y\rangle\langle y| . \quad (8.38)$$

If we assume that ρ starts out separable in the codespace basis

$$\rho = B(|\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'|)B^{-1} \quad (8.39)$$

then

$$\Phi_y^{\mathcal{S}}[B(|\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'|)B^{-1}] \circ \mathcal{E} = \sum_j p_j \mathcal{S}_y[\mathbf{P}_j B(|\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'|)B^{-1} \mathbf{P}_j] \otimes |y\rangle\langle y| . \quad (8.40)$$

Now observe that

$$\begin{aligned}\mathcal{S}_y[\mathbf{P}_j B(|\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'|) B^{-1} \mathbf{P}_j] &= \text{Tr}_{\mathcal{H}^{C^\perp}} [\Pi_y^{\mathcal{S}} \mathbf{P}_j |\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'| \mathbf{P}_j] \\ &= \text{Tr}_{\mathcal{H}^{C^\perp}} [B(\mathbb{1} \otimes |y\rangle\langle y|) B^{-1} \mathbf{P}_j B(|\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'|) B^{-1} \mathbf{P}_j]\end{aligned}\quad (8.41)$$

where I used the definition of $\mathcal{S}_y$ and the definition of the syndrome projector in the physical basis in the first and second equalities. Define

$$\mathbf{Q}_j \equiv \text{Ad}_B^{-1}(\mathbf{P}_j) = B^{-1} \mathbf{P}_j B. \quad (8.42)$$

It follows that

$$\begin{aligned}\mathcal{S}_y[\mathbf{P}_j |\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'| \mathbf{P}_j] &= \text{Tr}_{\mathcal{H}^{C^\perp}} [(\mathbb{1} \otimes |y\rangle\langle y|) \mathbf{Q}_j |\psi\rangle\langle\psi| \otimes |y'\rangle\langle y'| \mathbf{Q}_j] \\ &= |\langle y | \mathbf{Q}_j | y' \rangle|^2 \cdot \text{Tr}_{\mathcal{H}^{C^\perp}}(\mathbf{Q}_j) |\psi\rangle\langle\psi| \text{Tr}_{\mathcal{H}^{C^\perp}}(\mathbf{Q}_j).\end{aligned}\quad (8.43)$$

In this way, I show that

$$\Phi_y^{\mathcal{S}}[|\psi\rangle\langle\psi| \otimes |0\rangle\langle 0|] \circ \mathcal{E} = \sum_j p_j |\langle y | \mathbf{Q}_j | 0 \rangle|^2 \cdot \text{Tr}_{\mathcal{H}^{C^\perp}}(\mathbf{Q}_j) |\psi\rangle\langle\psi| \text{Tr}_{\mathcal{H}^{C^\perp}}(\mathbf{Q}_j) \otimes |y\rangle\langle y|. \quad (8.44)$$

Next, define

$$\mathbf{W}_j \equiv \text{Tr}_{\mathcal{H}^{C^\perp}}(\text{Ad}_B^{-1}(\mathbf{P}_j)). \quad (8.45)$$

One then has

$$\Phi_y^{\mathcal{S}}[|\psi\rangle\langle\psi| \otimes |0\rangle\langle 0|] \circ \mathcal{E} = \sum_j p_j |\langle y | \text{Ad}_B^{-1}(\mathbf{P}_j) | 0 \rangle|^2 \cdot \mathbf{W}_j |\psi\rangle\langle\psi| \mathbf{W}_j \otimes |y\rangle\langle y|. \quad (8.46)$$

The next simplification comes from observation that

$$|\langle y | \text{Ad}_B^{-1}(\mathbf{P}_j) | 0 \rangle|^2$$

is deterministic (it is determined by the destabilizers). Hence one can re-write

$$\sum_j p_j |\langle y | \text{Ad}_B^{-1}(\mathbf{P}_j) | 0 \rangle|^2 \cdot \mathbf{W}_j |\psi\rangle\langle\psi| \mathbf{W}_j \otimes |y\rangle\langle y| = \sum_y \sum_{k \in \mathcal{I}^{-1}(y)} \eta(y, k) \mathbf{W}_k |\psi\rangle\langle\psi| \mathbf{W}_k \otimes |y\rangle\langle y|. \quad (8.47)$$

where $\mathcal{I}^{-1}(y)$ is an index set over all errors that map to the same syndrome

$$\mathcal{I}^{-1}(y) = \{j : |\langle y | \text{Ad}_B^{-1}(\mathbf{P}_j) | 0 \rangle|^2 = 1\} \quad (8.48)$$

and $\eta(y, k)$ is the composite distribution

$$\eta(y, k) = \sum_{\{j : \text{Tr}_{\mathcal{H}^{C^\perp}}(\text{Ad}_B^{-1}(\mathbf{P}_j)) = \mathbf{W}_k\}} p_j. \quad (8.49)$$

The point of all this is that the composition of noise and syndrome extraction on a codestate with a definite stabilizer part results in a separable state that is equal to a definite syndrome part and a Pauli distribution on the codestate that is determined by the different errors with the same syndrome parts and varying codespace effects. In this way, we may write

$$\Phi_y^S[|\psi\rangle\langle\psi| \otimes |0\rangle\langle 0|] \circ \mathcal{E} = \sum_{z \in \mathcal{Z}} \lambda(y, z) \mathbf{W}_z |\psi\rangle\langle\psi| \mathbf{W}_z \otimes |y\rangle\langle y|. \quad (8.50)$$

Where $\mathbf{W}_z$ is the unique logical effect labeled by z and λ is a distribution on y, z .

Logical measurement channel

In a subsystem code, a logical measurement channel may be defined for a set of observables on the logical subspace. Of particular interest is $\bar{Z}$ -type and $\bar{X}$ -type logical measurement. I derive the logical channel representation for $\bar{Z}$, from which the $\bar{X}$ channel follows upon interchange of symbols.

The general, unknown-outcome logical- $\bar{Z}$ measurement channel is represented

$$\Psi^{\bar{Z}}[\rho] = \sum_{\nu \in \mathcal{Z}} |\nu\rangle\langle\nu| \otimes T_\nu[\rho] \quad (8.51)$$

where ν is a binary vector representing the logical bits ν_i measured for each logical $\bar{Z}_i$ and T_ν is defined analogously to $\mathcal{S}_y$ for stabilizer measurement. This form represents logical measurement without knowledge of the outcome. If one has knowledge of the outcome, then

$$\Psi_\nu^{\bar{Z}}[\rho] = |\nu\rangle\langle\nu| \otimes T_\nu[\rho]. \quad (8.52)$$

It is remarkable to note that the composition of a stabilizer measurement and a logical measurement with known outcomes in a gauge-free code in the absence of any errors yields the channel

$$\Psi_\nu^{\bar{Z}} \circ \Phi_y^S[\rho] = |\nu\rangle\langle\nu| \otimes |y\rangle\langle y| \quad (8.53)$$

which is a fact commonly used to prepare a state in the codespace.

Decoding measurement channel

A conditional decoding channel applies unitary U_y conditioned on having measured syndrome y

$$\mathcal{D}_y[\rho] = U_y \rho U_y^\dagger \quad (8.54)$$

for U_y defined on the total space $\mathcal{H}$

The general decoding channel may be modeled as measurement composed with conditional recovery, i.e., the channel

$$\mathcal{D}[\rho] = \sum_{y \in \mathcal{Y}} U_y \mathcal{S}_y[\rho] \otimes |y\rangle\langle y| U_y^\dagger. \quad (8.55)$$

The decoding unitary U can be separated into a commuting stabilizer part $T(y)$ and a logical part $R(y)$

$$U_y = T(y)R(y). \quad (8.56)$$

The stabilizer part acts on the stabilizer subspace and is designed so that it rotates back into the ground-state, i.e., sets $y = \vec{0}$. The logical part of the recovery operation is designed to undo the effect of the error on the logical state.

8.6 Fault tolerance

A introduction to QEC would not be complete without a discussion of the notion of fault tolerance in code operation. Fault-tolerant code design is a vital and difficult component of practical QEC implementation. The theory has been developed in the literature [Got24]. My treatment here is very brief.

The essential challenge of fault tolerant circuit design is that naive operation of a QEC code is not enough to ensure reliable operation. This is basically due to the fact that *faults can propagate* between encoded blocks in a large scale QEC design. If the fault propagate and *multiply* then error cascades will quickly overwhelm the capabilities of an entropy extraction protocol. Fault-tolerant circuit design involves designing physical implementation (often call “gadgets”) of stabilizer measurement and logical operation such that one error in a block causes at most one error in another block after the operation.

The three essential cases one deals with when considering fault tolerant circuit design are:

1. Error propagation from ancilla to data qubits in syndrome measurement,
2. Error propagation between code blocks in logical operations.
3. Error propagation in the initial encoding

Each of these may be understood as ensuring that errors in one block do not propagate to another. In the case of syndrome extraction, the ancilla qubits are one block and the data qubits the other block. In the case of encoding, each physical qubit is a block.

To ensure fault tolerant syndrome extraction, there are currently two popular approaches: 1) designing stabilizer gadgets that bound the number of errors that can be introduced to the data qubits from ancilla errors, and 2) flag qubits that identify errors on ancilla qubits. In the first approach (see eg. [HB20]), stabilizer measurements are performed in a particular order, which ensures that if an error does occur on the ancilla register that would usually hook to the data register, the error effect compiles to at most a single error composed with a stabilizer of the code. In the second approach (see [CR20]), additional ancilla qubits are included and are coupled to the stabilizer measurement qubits and measured similarly to the syndrome qubits. However, it should be clear that flag qubits are not just “another layer of syndrome extraction” and one cannot infer a correction based on flag outcomes. Instead,

flag qubits flag that a hook error likely occurred, which means that an erasure error likely occurred on the data register.

To ensure fault tolerant logical operation one must ensure that the operations done between code blocks do not multiply errors from one block into the other block. The simplest way to do this is transversal code operation. In this case, multi-qubit operations are done between pairings of qubits across blocks, so errors cannot multiply. However, the Eastin-Knill theorem [EK09] states that no QEC code can have a transversal gate set that is universal. Thus, one has to include at least one non-transversal gate in the gate set. Usually, this is the “T-gate,” which is the point of T-state distillation protocols and “magic state factories” [OC16; Sil+24]. Other approaches to fault tolerant logical operations exist, including code deformation, lattice surgery, and gauge fixing [Vui+19].

To ensure fault tolerant encoding, one can follow the procedure illustrated in Fig. 8.5. Here, one uses the fact that syndrome measurement projects onto the codespace. After syndrome measurement, measurement of logical-Z operators project into a definite codestate. Logical-X operators can be applied to

In discussions of fault tolerance, the threshold theorem [KLZ96] is usually brought up. This is the statement that there exists some error rate below which concatenations (or lattice dilation) will improve the logical error rate. Above the threshold, concatenations or lattice dilation make the logical error rate larger since one is using more of a noisy resource. Calculations of code thresholds can be done by concatenating (or dilating) a code and evaluating the logical error rate under varying noise. The point at which the logical error rate crosses the performance of unencoded computation approaches a limit that is the threshold of the code family. A modern approach [Col+23] relies on coherent information calculations to determine thresholds.

The original approaches [Sho95; Ste96] to fault tolerance showed that circuit design could be done by hand for certain code families. Modern approaches often rely on classical computer aides to verify circuit design [Che+25].

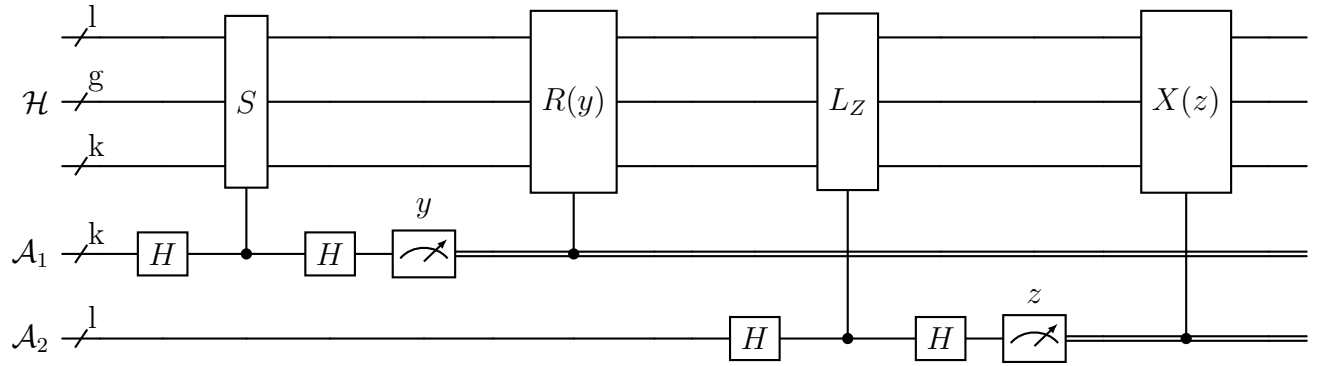


Figure 8.5: *Measurement based stabilizer encoding.* Measurement of the stabilizers projects onto a definite syndrome state y . Decoding the syndrome y returns a correction that maps to $\rho \otimes |0\rangle\langle 0|$. Measurement of logical-Z operators projects into a definite logical state $|z\rangle\langle z| \otimes |0\rangle\langle 0|$ and logical-X operators can be applied to map to $|0\rangle\langle 0| \otimes |0\rangle\langle 0|$.

Chapter 9

Entropic measures of code performance

Designing codes that perform “optimally” under a given noise distribution is a central problem in engineering efficient quantum information processors. In this problem, one must select a particular “best” code for a device. For a large number of qubits, there are a very large number of possible QEC codes, each with specific performance when run on a given error distribution and decoder. Hence, it is desirable to have a metric of “best” performance that can be used to compare different codes. Logical error rate under a given decoder is a common metric. However, logical error rate is a decoder dependent metric. If one operates the code under the optimal decoder (§ 8.4), then one finds an upper bound on the best achievable performance of *any* decoder for that code and error distribution. Yet, the definition of the optimal decoder is #P hard.

In this chapter, I propose an entropic metric of code performance (CCE theorem § 9.2). This metric quantifies the expected success rate of an optimal decoder after n rounds of stabilizer extraction. Formally, calculation of this metric is as difficult as construction of the optimal decoder. However, I hypothesize that careful importance sampling techniques based on the method of types – as discussed in § 9.6 – can render this metric an operationally useful cost function for code optimization. Entropy optimization also has a long history in the study of classical channel capacity.

The ideas introduced in this chapter are useful for reasoning about the operation of QEC codes operating under general noise channels in the large cycle limit. Code space entropy (§ 8.3) quantifies rates of disorder and information in subspaces of a QEC code. The logical entropy rate quantifies the disorder in the logical subspaces caused by the noise. The conditional codespace entropy (CCE) quantifies the residual disorder in the logical subspaces after the syndrome information is known. When the CCE is low, the decoder can infer the logical effect with high probability. The difference between the logical entropy and CCE results in the mutual information between logical effects and syndromes, which quantifies how much information syndromes convey about logical effects. These abstractions can be used to reason about the performance of large QEC systems that are operated over long

time scales.

Codespace entropy metrics quantify expected entropy generation and extraction rates in the logical subspace of a *particular* stabilizer code when operated over a large number of cycles. This concept should be compared with the well studied metric of quantum coherent information [XX07], which quantifies the maximal rate of quantum communication using *any* QEC code for a given noise distribution. Recent work applied quantum coherent information to QEC codes to determine threshold performance [Col+23].

Codespace entropy measures upper bound code performance, allowing us to make statements like “under n uses of this code in this noise environment, the maximum expected success probability is p ” and “the rate of entropy extraction of this code under this noise is r ” and “out of a given set of possible codes, this particular code will have the lowest achievable error rate.” These statements may be made without specifying a decoder. Hence, codespace entropy metrics provide a decoder-independent metric to identify optimal codes.

The outline of this chapter is as follows: I begin with an introduction to entropy as a metric of code performance. Next, I state and prove a conditional codespace entropy (CCE) theorem, which quantifies optimal decoder performance in terms of a conditional entropy metric. I then discuss applications of CCE, its relationship to coherent information, and demonstrate some numerical studies. I conclude with a discussion of future prospects. I recommend the reader refer to § 2.5 and § 8.3 for relevant background.

9.1 Entropy as a measure of code performance

Entropy measures the amount of disorder in a system. In the context of QEC, entropy can be applied to different aspects of the protocol. For example, one can discuss the entropy of the Pauli error channels that the device is subject to. Similarly, one can discuss the entropy of the syndromes that come from stabilizer measurement or the entropy of the logical degrees of freedom as they flip. Furthermore, conditional entropy measures residual entropy in one random variable after another, correlated random variable is observed. Using this idea, I define the conditional entropy between logical dof. in a code and stabilizers. When conditional entropy is small, the syndromes contain information that can be used to uniquely infer the logical effect. The CCE theorem § 9.2 formalizes conditional entropy between logical effects and syndromes as a measure of optimal code performance.

The performance of a QEC can be quantified as the probability that a QEC cycle successfully corrects for the effects of the noise. In order to apply information theoretic analysis, is necessary to discuss this probably in the limit of a large number of samples from the code. In a code capacity model (See Fig 8.2 in § 8), this situation can be modeled as a the probability that a QEC cycle corrects for the effects¹ $x^{(n)} = (x_1, x_2, \dots, x_n)$, ie.

$$\Pr(x^{(n)} \text{ is corrected}).$$

¹Formally, $x^{(n)}$ is a list of labels of errors. In this chapter, I will be somewhat loose with the distinction between an error represented as a Pauli operator and the labels of the error

The decoding success probability may be rewritten in terms of syndrome information

$$y^{(n)}(x^{(n)}) = (y_1, y_2, \dots, y_n) \quad (9.1)$$

and logical information

$$z^{(n)}(x^{(n)}) = (z_1, z_2, \dots, z_n). \quad (9.2)$$

Here, $y^{(n)}$ is a sequence of *differences* in binary vectors obtained from measurement of the stabilizers and $z^{(n)}$ is a sequence of binary vectors that describe *changes* in the logical state of the code. In real experiments, $y^{(n)}$ is the stream of data that comes from the code, and $z^{(n)}$ describes the (hidden) effect on the logical dof. In simulation, $z^{(n)}$ can be computed by sampling Pauli channels using efficient Clifford tableau techniques (§ 8). Under Pauli noise and Clifford encoding, the logical error stream $z^{(n)}$ can be represented by a binary vector of length $2k$, for k logical qubits. Here, the first k bits describe the X-logical part and the last k bits describe the Z-logical part.

The elements of $y^{(n)}$ are iid. samples from a random variable Y that is determined by the underlying error distribution $\Pr(X = x)$, see § 8.3. Similarly, the elements of $z^{(n)}$ are iid. distributed according to $\Pr(Z = z)$ that is fixed by the underlying error distribution. The correlations between $y^{(n)}$ and $z^{(n)}$ are set by the underlying error distribution *and* the particular code. Different stabilizer measurements have different sensitivities to different errors, which changes the correlations between Z and Y .

Following the discussion of decoders in (§ 8.4) the success probability of an error class decoder is determined by its ability to infer the logical effect sequence from stabilizer measurements. In this way,

$$\Pr(x^{(n)} \text{ is corrected}) = \Pr(\hat{z}^{(n)}(y^{(n)}) = z^{(n)}) \quad (9.3)$$

where $\hat{z}^{(n)}(y^{(n)})$ is the decoder's estimate of the error effect sequence given the syndrome sequence. Entropy as a metric of code performance makes the observation that the success probability of an error class decoder converges in the limit of a large number of code cycles to a simple function of the conditional entropy of the random variable that describes the logical effects Z conditioned on the random variable that describes the syndromes Y .

The analysis in this chapter mostly follows a code capacity model. This model permits the most straightforward route to analysis of QEC codes. However, as I discuss in § 9.6, I believe it would be straightforward to generalize these techniques to phenomenological and circuit level noise models, as was done in [Col+23].

9.2 The codespace entropy theorem

Conditional codespace entropy (CCE) theorem. Given a stabilizer code in the code-capacity model operating under a large number n of syndrome extraction cycles, the probability that an optimal error class decoder² successfully corrects a sequence of logical error

²Recall from § 8 that I defined the optimal error class decoder to be the maximum probability of error class, sometimes called “maximum likelihood of error class” in the literature.

effects $z^{(n)} = (z_1, \dots, z_n)$ given a sequence of syndrome measurements $y^{(n)} = y_1, \dots, y_n$ lies in the interval

$$(1 - \varepsilon)2^{-n(H(Z|Y)+\varepsilon)} \leq \Pr(\hat{z}^{(n)} = z^{(n)}|y^{(n)}) \leq 2^{-n(H(Z|Y)-\varepsilon)} \quad (9.4)$$

where $H(Z|Y)$ is the conditional entropy between logical effects and syndrome measurements and ε is a positive constant that may be made arbitrary small with increasing n .

Discussion. The CCE theorem establishes $H(Z|Y)$ as a relevant metric of code performance. For sufficiently large n , the performance of an optimal error class decoder follows a trend established by $H(Z|Y)$. The ε that appears in the theorem is the same as the ε in the construction of a jointly and conditionally typical set. Here ε may set to an arbitrarily small, positive constant as the number of syndrome extraction cycles n increases.

Lemma. For sufficiently large number of syndrome extraction cycles n , the failure rate of a code is approximately

$$r = H(Z|Y). \quad (9.5)$$

Invoking the lemma, one can make statements such as “the code can probably be used about $1/H(Z|Y)$ before failure.”

Proof of the theorem

This proof applies to the code capacity model (Fig. 8.2). It proceeds in two steps:

1. The jointly typical decoder succeeds with probability in the interval

$$(1 - \varepsilon)2^{-n(H(Z|Y)+\varepsilon)} \leq \Pr(\hat{z}^{(n)} = z^{(n)}|y^{(n)}) \leq (1 - \varepsilon)2^{-n(H(Z|Y)-\varepsilon)}$$

2. The success probability of the jointly typical decoder converges to the success probability of optimal error class decoding in the large sample limit.

The proof of the first statement essentially invokes properties of jointly typical and conditionally typical sets. The proof of the second statement follows from AEP (§ 2.5) for conditionally typical sets. The second statement is also essentially equivalent to a key part of the proof of Shannon’s theorem in [TT12].

Statement 1

It shall be shown that the success probability of the jointly typical decoder lies in the interval $[(1 - \varepsilon)2^{-n(H(Z|Y)+\varepsilon)}, (1 - \varepsilon)2^{-n(H(Z|Y)-\varepsilon)}]$ for sufficiently large n . Define logical effect sequence $z^{(n)} = (z_1, z_2, \dots, z_n)$, syndrome sequence $y^{(n)} = (y_1, y_2, \dots, y_n)$, and joint sequence $w^{(n)} = ((z_1, y_1), (z_2, y_2), \dots, (z_n, y_n))$. Recall the two failure conditions of the jointly typical decoder (§ 2.5) and define events

- $E_1 \equiv \{w^{(n)} \in B_\epsilon^{(n)}\}$ (the sequence is jointly typical)
- $E_2 \equiv \{((z'_1, y_1), \dots, (z'_n, y_n)) \notin B_\epsilon^{(n)} \mid \forall z'^{(n)} \neq z^{(n)}\}$ (the syndrome sequence is not jointly typical with any other logical effect)

Then, the success probability of the jointly typical decoder may be written

$$\Pr(\hat{z}^{(n)}(y^{(n)}) = z^{(n)}) = \Pr(E_1 \cap E_2) \quad (9.6)$$

$$= P(E_2|E_1)P(E_1). \quad (9.7)$$

Now, the probability that the sequence is jointly typical, $\Pr(E_1)$, is the probability that the joint sequence $w^{(n)}$ is in the jointly typical set (§ 2.5), which is $1 - \epsilon$. It follows that

$$\Pr(\hat{z}^{(n)}(y^{(n)}) = z^{(n)}) = (1 - \epsilon) \Pr(E_2|E_1) \quad (9.8)$$

$$= (1 - \epsilon) \Pr(((z'_1, y_1), \dots, (z'_n, y_n)) \notin B_\epsilon^{(n)} \mid \forall z'^{(n)} \neq z^{(n)} \mid w^{(n)} \in B_\epsilon^{(n)})$$

The next step is to argue that

$$2^{-n(H(Z|Y)+\epsilon)} \leq P(E_1|E_2) \leq 2^{-n(H(Z|Y)-\epsilon)} \quad (9.9)$$

which follows from properties of conditionally typical sequences and AEP (§ 2.5). Observe that $2^{n(H(Z|Y)-\epsilon)}$ lower bounds the number of elements in the conditionally typical set. By AEP, each element of the typical set occurs with approximately equal probability bounded above by $2^{-n(H(Z|Y)-\epsilon)}$. Similarly, $2^{n(H(Z|Y)+\epsilon)}$ upper bounds the number of elements, so $2^{-n(H(Z|Y)+\epsilon)}$ lower bounds the probability of a particular jointly typical sequence. Hence, in the limit of large n , the probability that a given typical syndrome sequence is jointly typical with more than one sequence of logical effects is given by Eq. 9.9.

In this way, I have shown that the jointly typical decoder's success probability lies in the interval

$$(1 - \epsilon)2^{-n(H(Z|Y)+\epsilon)} \leq \Pr(\hat{z}^{(n)}(y^{(n)}) = z^{(n)}) \leq (1 - \epsilon)2^{-n(H(Z|Y)-\epsilon)}. \quad (9.10)$$

Observe that one may pick $\epsilon = \epsilon$. Hence, it follows that

$$(1 - \epsilon)2^{-n(H(Z|Y)+\epsilon)} \leq \Pr(\hat{z}^{(n)}(y^{(n)}) = z^{(n)}) \leq 2^{-n(H(Z|Y)-\epsilon)}. \quad (9.11)$$

Statement 2

It shall be shown that the jointly typical decoder converges to optimal error class decoding in the large sample limit. It is sufficient to show that jointly typical decoding is equivalent to optimal error class decoding in the large sample limit. This equivalence follows from AEP for conditionally typical sets. Observe that given a syndrome string $y^{(n)}$, the average number of jointly typical logical strings $z^{(n)}$ is given by the size of the conditionally typical set

$$|C_\epsilon^{(n)}| \approx 2^{nH(Z|Y)}. \quad (9.12)$$

In the large sample limit, each jointly typical $z^{(n)}$ given $y^{(n)}$ occur with approximately uniform probability. So, the best an optimal error class decoder can do is pick one at random, yielding success probability

$$\Pr(\text{optimal error class decoder success}) \approx 2^{-nH(Z|Y)}$$

9.3 Applications

Code performance metrics

CCE was originally derived to be a cost function for optimal code design. To illustrate its utility, I begin by defining a few possible cost functions, discuss their short-comings, and present conditional entropy as a promising metric.

Decoder error cost

The simplest and most common cost function is decoder error cost. This cost function penalizes all errors that are not corrected for by the decoder. It is defined as 0 if the decoder's correction has the proper commutation relation with the logical operators and $\lambda(x)$ otherwise, where λ is a cost measure on possible physical errors. For a single logical qubit, the indicator of the decoder error cost is

$$I_{\text{decoder error}}(x) = \bar{\delta}[R(y(x))E(x), \bar{X}] \bar{\delta}[R(y(x))E(x), \bar{Z}] \quad (9.13)$$

where $\bar{\delta}$ returns 0 if the operations commute and 1 otherwise. The total cost is then

$$r_{\text{decoder error}} = \sum_x \lambda(x) I_{\text{decoder error}}(x) \quad (9.14)$$

If one picks the underlying distribution of Pauli errors for cost measure λ , then one recovers the usual logical error rate

$$r_{\text{decoder error}} = \sum_x \Pr(X = x) I_{\text{decoder error}}(x), \quad (9.15)$$

which is a very common metric of code performance. Obviously, this cost function is decoder dependent. If one chooses the optimal error class decoder, then the resulting cost provides an upper bound on the cost of using any other sub-optimal decoder.

Knill-Laflamme Cost

The Knill-Laflamme (KL) cost $r_{\mathcal{C}}$ [Oll+24] of a code $\mathcal{C}$ with stabilizer group $\mathcal{S}_{\mathcal{C}}$ can be defined

$$r_{\mathcal{C}} = - \sum_{x \in \mathcal{P}^n} \lambda(x) K_{\mathcal{C}}(x)$$

where $\lambda(x)$ is a cost measure and $K_{\mathcal{C}}(x)$ is an indicator function that is 1 if : 1) the error x anti-commutes with at least one stabilizer of $\mathcal{C}$ OR 2) the error x is a stabilizer in $\mathcal{S}_{\mathcal{C}}$. In other words, $K_{\mathcal{C}}(x)$ encodes the Knill-Laflamme conditions. Said formally: $K_{\mathcal{C}}$ is 1 if

$$|\{g : [E_x, g] \neq 0 | g \in \mathcal{S}_{\mathcal{C}}\}| > 0 \quad (9.16)$$

or

$$E_x \in \mathcal{S}_{\mathcal{C}} \quad (9.17)$$

else 0. The KL cost has the interpretation that it penalizes codes that cannot detect errors that affect the logical information. The KL cost is decoder independent. However, the KL cost cannot account for effects of degenerate quantum errors .

Jointly typical decoding cost

The jointly typical decoding cost is simply CCE

$$H(Z|Y) \quad (9.18)$$

CCE may be calculated exactly (which is as hard as optimal decoding) or estimated empirically by collecting a large number of samples from the code (see below). I hypothesize that careful importance sampling can render CCE efficient to approximate. The jointly typical decoding cost deals with degeneracy and is decoder independent.

Empirical estimation

Up to now, I have discussed exact calculation of CCE from knowledge of the underlying noise distribution $\Pr(X)$. It is important to observe that calculation of CCE is as difficult as construction of the optimal error class decoder. For noise distributions on a large number of qubits, it is impractical to exactly calculate $H(Z|Y)$. However, one can estimate $H(Z|Y)$ empirically.

Empirical estimation of $H(Z|Y)$ is done by simulating the dynamics of the code for a large number of samples n and using the estimator

$$H(z^{(n)}|y^{(n)}) = - \sum_{y \in \mathcal{Y}} \hat{p}_Y(y) \sum_{z \in \mathcal{Z}} \hat{p}_{Z|Y}(z|y) \log \hat{p}_{Z|Y}(z|y). \quad (9.19)$$

Here, the empirical syndrome distribution estimator is calculated

$$\hat{p}_Y(\cdot) = \frac{\text{count}(\cdot|y^{(n)})}{n} \quad (9.20)$$

for count function that returns the number of times the argument appears in the sequence. Similarly, the empirical conditional distribution estimator is calculated

$$\hat{p}_{Z|Y}(a|b) = \frac{\text{count}((a, b)|z^{(n)} \times y^{(n)})}{\text{count}(b|y^{(n)})} \quad (9.21)$$

for joint count function that returns the number of times the joint argument appears in the observed pairing.

It is nice to observe that the estimator may be re-written

$$H(z^{(n)}|y^{(n)}) = -\frac{1}{n} \sum_{z,y \in \mathcal{Z} \times \mathcal{Y}} \text{count}(z, y) \log \frac{\text{count}(z, y)}{\text{count}(y)} \quad (9.22)$$

where the conditional arguments of the count function are dropped and should be clear from context.

Bounding single-shot performance

My analysis has been of typical sequences that result from many samples of a Pauli error channel and their resulting syndrome patterns and logical effects. This sort of perspective deviates from many usual studies of quantum error correction, where one considers decoding the logical effect of a Pauli error from its syndrome in a single shot. In fact, numerical analysis (§ 9.5) of the conditional entropy reveals some discrepancy between single-shot thresholds and the performance crossing of jointly typical decoding and raw computation, which merits further investigation. Nevertheless, the conditional entropy between logical effects and syndromes that I identified in the previous section can also be used to bound the performance of single-shot decoding via the Fano bound.

The conditional entropy $H(Z|Y)$ provides a lower bound on the the probability of estimator error $\Pr(Z \neq \hat{Z})$ where $\hat{Z}$ is *any* estimator of Z given observation of Y . This follows from Fano's inequality

$$H(Z|Y) \leq H_b(e) + P(e) \log(|\mathcal{Z}| - 1) \quad (9.23)$$

where $P(e)$ is the probability of an error $\Pr(\hat{Z} \neq Z)$ and $|\mathcal{Z}|$ is the cardinality of the support of the logical effects.

Writing $H_b(e) \leq 1$, Eq. (9.23) can be weakened and written as

$$P(e) \geq \frac{H(Z|Y) - 1}{\log(|\mathcal{Z}| - 1)} \quad (9.24)$$

9.4 Comparison with coherent information

CCE, as I have defined it, is a classical entropy measure defined for a stabilizer representation of a quantum code. It quantifies the success rate under known Pauli noise and is well-defined only for Clifford encodings. A recent publication [Col+23] defined the coherent information in a QEC code that has many similarities to CCE, which is analyze here.

I begin with a general definition of coherent information. Then, I define the coherent information in a code following [Col+23]. Finally, I discuss the similarities and differences between coherent information of a logical state and CCE.

Coherent information

The *coherent information* I_{CI} of a quantum state ρ subject to a noise channel $\mathcal{E}$ is an upper limit of the amount of quantum information in ρ that can be recovered after the noise $\mathcal{E}$. Quantum coherent information is defined in terms of a purification of the input state with a reference state along with an additional ancilla register, which makes its presentation somewhat complicated.

A **purification** [NC02] of a density state $\rho \in \mathcal{H}^A$ is a pure state $|\Psi^{AB}\rangle \in \mathcal{H}^A \otimes \mathcal{H}^B$ with the property that

$$\text{Tr}_B[|\Psi\rangle\langle\Psi|] = \rho \quad (9.25)$$

It suffices to define $\mathcal{H}^B$ as a copy of $\mathcal{H}^A$ so that $\dim(\mathcal{H}^A) = \dim(\mathcal{H}^B)$. Purification connects to the notion that every incoherent quantum process can be understood as a coherent process on a larger Hilbert space [NC02]. Given a density matrix, one can always define a spectral decomposition

$$\rho = \sum_j p_j |\psi_j\rangle\langle\psi_j| \quad (9.26)$$

for some (generally non-unique) set of pure states $|\psi_j\rangle$; the purification of ρ is then

$$\rho \mapsto \sum_j p_j |\psi_j\rangle \otimes |a_j\rangle \quad (9.27)$$

for some basis $|a_j\rangle$ in $\mathcal{H}^B$. Notice that if ρ is a pure-state, then its purification is separable and if ρ is mixed then its purification is entangled. In this way, one obtains an operational meaning to the notion that “decoherence results from entanglement with an environment.”

The **entropy exchange** [Sch96; XX07] $\mathbb{S}^E(\rho; \mathcal{E})$ between a state ρ and a channel $\mathcal{E}$ is the von Neumann entropy of the state that results from applying the channel to part of a purified representation of ρ ;

$$\mathbb{S}^E(\rho; \mathcal{E}) \equiv \mathbb{S}(\text{Tr}_B[\mathcal{E} \otimes \mathbb{1}[|\Psi^{AB}\rangle\langle\Psi^{AB}|]]) \quad (9.28)$$

where $|\Psi^{AB}\rangle \in \mathcal{H}^A \otimes \mathcal{H}^B$ is a purification of ρ . Notice that for purification $|\Psi^{AB}\rangle = \sum_i p_i |\psi_i\rangle \otimes |a_i\rangle$ one has

$$\begin{aligned} \mathbb{S}^E(\rho; \mathcal{E}) &= \mathbb{S}\left(\sum_j p_j \text{Tr}_B[\mathcal{E}[|\psi_j\rangle\langle\psi_j|] \otimes |a_j\rangle\langle a_j|]\right) \\ &= \mathbb{S}\left(\sum_j p_j \mathcal{E}[|\psi_j\rangle\langle\psi_j|]\right) \end{aligned} \quad (9.29)$$

The entropy exchange can be understood as quantifying the amount information that is exchanged between a subsystem $\mathcal{H}^A$ and its environment during the noise process $\mathcal{E}$ [XX07]. Alternatively, the entropy exchange quantifies the rate at which a noise source entangles a system with its environment [Sch96].

The **coherent information** [XX07] $I_{\text{CI}}(\rho; \mathcal{E})$ of a quantum state ρ and channel $\mathcal{E}$ is defined as the difference between the entropy of output state of the channel $\mathcal{E}[\rho]$ and the exchange entropy between the state and the channel

$$I_{\text{CI}}(\rho; \mathcal{E}) = \mathbb{S}(\mathcal{E}[\rho]) - \mathbb{S}^E(\rho; \mathcal{E}). \quad (9.30)$$

Coherent code information

Coherent code information, as used in [Col+23], is defined (Eq. 1 in [Col+23]) as a difference of von Neumann entropies

$$I = S(\rho_Q) - S(\rho_{RQ}) \quad (9.31)$$

where ρ_Q is state of the register after application of a noise channel and ρ_{RQ} is the resultant joint state that includes a purifying reference register that is maximally entangled with the logical subspace, see Fig. 1 in [Col+23].

Comparison

Coherent codespace information is defined on the output of a noise channel *before stabilizer measurement*. This differs from the codespace entropies and information I discussed above, which apply after stabilizer measurement. Furthermore, coherent codespace information is defined for general noise channels, while the other entropy metrics are classical measures that only apply for Pauli noise and Clifford encoding. Coherent information and related ideas may provide the proper quantum generalization of the ideas I have introduced in this chapter.

9.5 Numerical studies

I implemented a number of numeric studies that are collected in the online repository [Mar25]. Here, I present three particular studies in that repository: 1) calculation of codespace entropy measures for the 5-qubit code $[[5, 1, 3]]$ and the 9-qubit surface code surface- $[[9, 1, 3]]$, 2) a study of conditional entropy of the 5-qubit code under biased noise, and 3) convergence rates of empirical estimators of CCE to the truth. The first comparison demonstrates the concepts of codespace entropy and highlights its use cases. The second study illustrates an expected use case of this theory where conditional entropy is used as a cost function. The final study demonstrates how these techniques can be deployed in numeric analysis of large codes.

Entropy measures of 5-qubit and surface codes

In my first numeric study, I examine the entropy rates of the perfect 5-qubit code and the 9-qubit surface code. I consider a depolarization channel $\mathcal{D}_p$ on 5-qubits and 9-qubits, and

I calculate the exact joint distribution of logical and syndrome effects by evaluating the probability of joint effects for all errors in the distribution

$$\Pr(Z = z, Y = y) = \sum_{x \in \mathcal{X}} \Pr(X = x) \bar{\delta}[z, x] \bar{\delta}[y, x]. \quad (9.32)$$

I then calculate the CCE

$$H(Z|Y) = - \sum_{z,y} \Pr(Z = z, Y = y) \log \frac{\Pr(Z = z, Y = y)}{\Pr(Y = y)}, \quad (9.33)$$

the codespace entropy

$$H(Z) = - \sum_z \Pr(Z = z) \log \Pr(Z = z), \quad (9.34)$$

and the codespace information

$$I(Z; Y) = H(Z) - H(Z|Y). \quad (9.35)$$

Additionally, I also calculate the entropy of a multi-qubit Pauli channel

$$\mathbb{S} \left(\sum_j p_j \mathbf{P}_j(\cdot) \mathbf{P}_j \right) = - \sum_j p_j \log p_j. \quad (9.36)$$

Each of these entropy measures are calculated for a varying depolarization rate p (recall depolarization is a Pauli channel). I consider an equally spaced line of depolarization rates and evaluate the measures numerically. It seems that a logarithmic sampling of depolarization rates would also be appropriate.

It is interesting to compare the 5-qubit code and the surface code entropy curves. In Fig. 9.3 I plot the difference of the 5-qubit entropy curves minus the surface entropy curves on 9-qubits. There are a few interesting observations one can make, listed below

1. Observe that the CCE is higher in the 5-qubit than the surface code for small depolarizing probability and crosses at some value around 0.17 to become higher in the surface code. This fact should imply that the 5-qubit code achieves lower asymptotic-optimal (in the sense of the CCE theorem) logical error rates at small depolarization rates, but the surface code achieves lower error rates at larger rates after some critical point.
2. Observe that the logical entropy is always lower in the surface code than the 5-qubit code. This fact should imply that the logical degrees of freedom flip less frequently in the surface code than in the 5-qubit code.
3. Observe the the mutual information is always lower in the surface code than the in the 5-qubit code. This fact should imply that the syndromes of the 5-qubit code provides more information about the noise than in the surface code, which is surprising. Perhaps this is a result of the symmetric nature of depolarization noise?

This comparison of differences between codespace entropies for varying codes is interesting and merits further investigation. It would be very interesting to compare codespace differences for a sequence of codes in a family, such as a sequence of dilated surface code with increasing lattice size.

Biased noise evaluation

For my second numeric experiment, I calculate the CCE $H(Z|Y)$ under biased noise distributions. I consider the biased Pauli noise channel defined on a single qubit as

$$\mathcal{E}_{p_X, p_Z} = (1 - \alpha)\rho + p_X X\rho X + p_Z Z\rho Z + p_X p_Z ZX\rho XZ \quad (9.37)$$

for normalization

$$\alpha = 1 - p_X - p_Z - p_X p_Z. \quad (9.38)$$

Given biases p_X, p_Z , the channel is valid for all biases with valid normalization $\alpha \in [0, 1]$. This noise channel provides a simple two-axis decomposition of biased noise suitable for plotting on a 2-dimensional surface. However, it is not necessarily realistic noise distribution. The m -qubit channel is defined via tensor product of the single-qubit channels

$$\mathcal{E}_{p_X, p_Z}^{(m)} = \bigotimes_{i=1}^m \mathcal{E}_{p_X, p_Z} \quad (9.39)$$

The results are displayed in Fig. 9.4.

This demonstration of biased CCE calculation over a grid of values illustrates the utility of CCE as a cost function for optimization. One may observe that the CCE value varies for different noise profiles. A similar calculation for code optimization and fixed noise could be done. Here, one would create a grid of different codes, calculate the joint distributions with exact evaluation, then calculate the CCE. Furthermore, one could instead sample from the noise channel for sufficiently large n and estimate the CCE empirically.

Distributions of empirical estimates

For my final numeric experiment, I calculate distributions of CCE *empirical estimators* for varying sample size n . In this case, I consider a value of 0.11 for the depolarizing rate. This is a somewhat arbitrary value, and was chosen to be somewhere near the crossing of CCE and 1-qubit entropy in the 5-qubit code. In general, smaller entropy rates will be more difficult to estimate, and 0.11 is a relatively large depolarization strength.

The results of the simulation are depicted in Fig. 9.5. One may observe the estimator is biased when the number of samples is not sufficiently large. As the number of samples increases, the estimator converges to start to look like a Gaussian distribution about the truth.

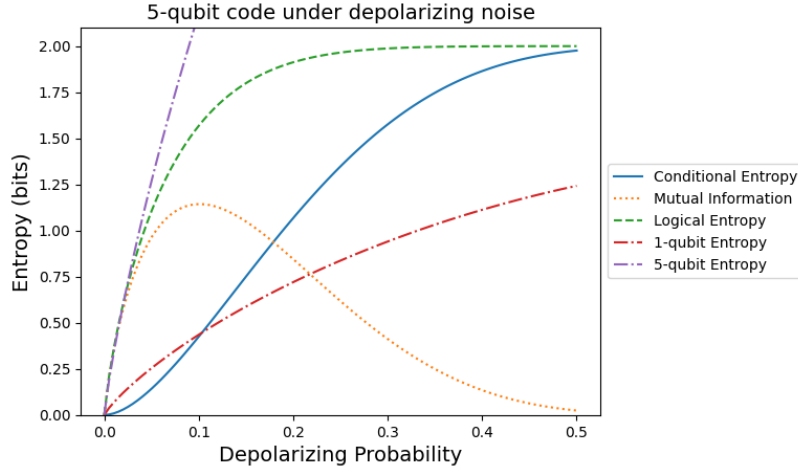


Figure 9.1: *Entropic measures in the 5-qubit code under depolarizing noise.* Exact codespace error distributions were calculated for varying rate of depolarization error on a grid of 100 equally spaced depolarization rates. CCE $H(Z|Y)$, mutual information $I(Z;Y)$, and logical codespace entropy $H(Z)$ were calculated at each point. Additionally, the single-qubit entropy of the separable noise channel and the 5-qubit entropy of the total noise channel were calculated. The logical entropy peaks at 2 bits, corresponding to the four types of Pauli errors that can occur in a single-qubit logical state. The mutual information grows rapidly then decays more slowly after peaking at some fixed depolarization rate. CCE grows like a sigmoid function before peaking at the maximum logical entropy.

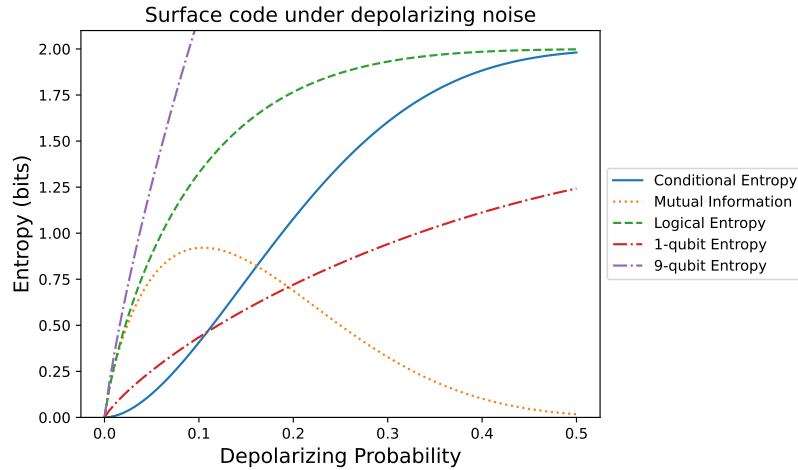


Figure 9.2: *Entropic measures in the surface code under depolarizing noise.* This plot should be compared with Fig. 9.1. Remark on the similarities in the structure of the entropy measures.

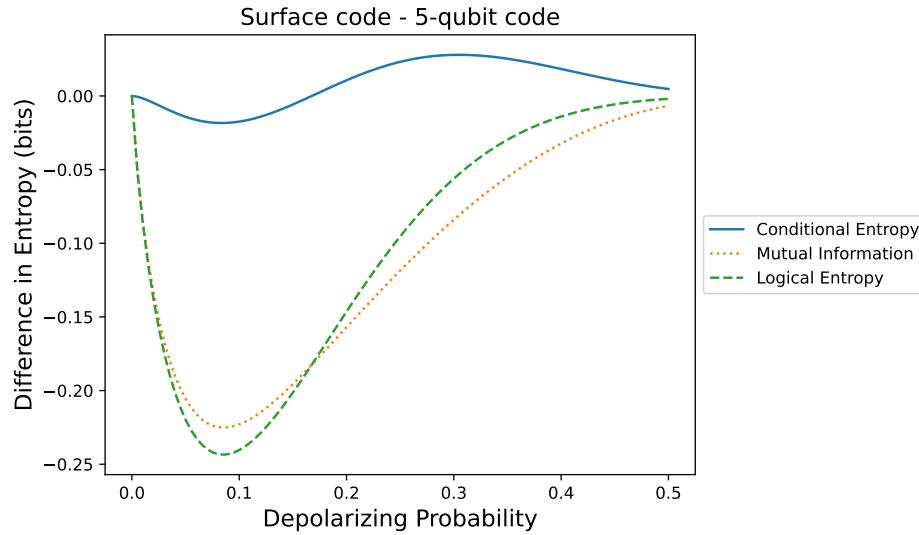


Figure 9.3: *Entropy difference between the surface codes and 5-qubit code.* Plot shows the surface code rates (Fig. 9.2) minus the 5-qubit rates (Fig. 9.1). See in-text for discussion.

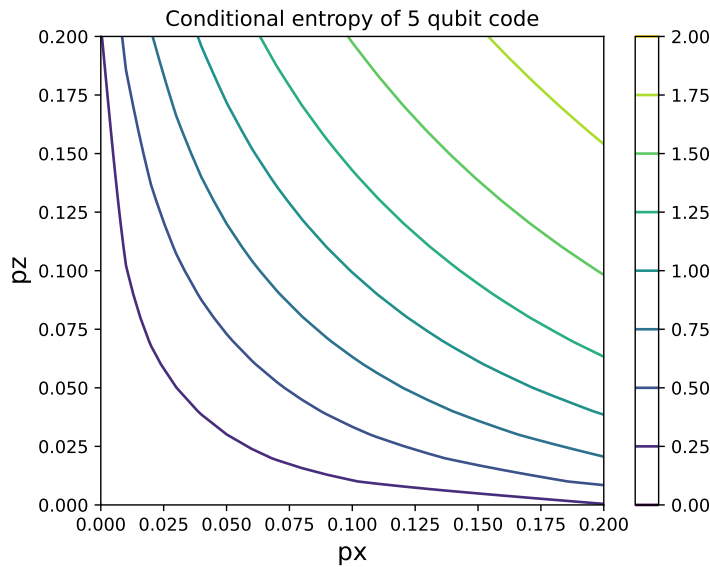


Figure 9.4: *Contour plot of the CCE as a function of X- or Z-bias.* A simple biased noise model was used to generate a 2-dimensional grid of conditional entropy estimates. The contour plot displays the characteristics of CCE in terms of the biases. Observe that for small biases in one axis, the CCE is small even for relatively large values in the other axis.

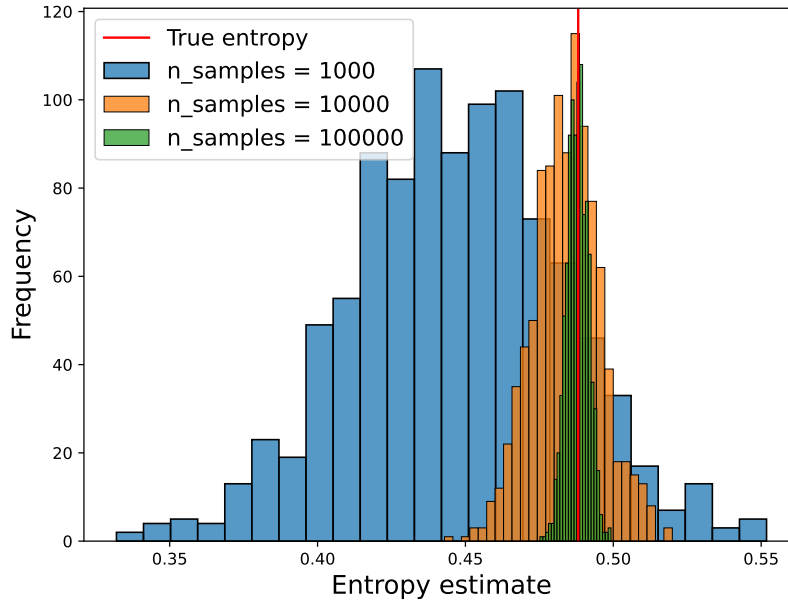


Figure 9.5: *Distribution of empirical estimates for the five-qubit code under fixed depolarization.* Using a fixed depolarizing noise strength of 0.11 (about the value that conditional entropy crosses single-qubit entropy), a number of different samples (in the list $[10^3, 10^4, 10^5]$) were collected from the composite noise-syndrome channel of the 5-qubit code and empirical entropy estimates were created using. This process was repeated for varying number of samples to create a histogram of the estimate distribution.

9.6 Future directions

Logical entropy extraction rates

Thus far, my focus has been on success probabilities. However, it is interesting and potentially useful to think in terms of entropy rates. A noise process generates entropy in the logical subspace, and the decoder removes some of this entropy. The rate of entropy removal goes like $H(Z|Y)$ and the rate of entropy generation like $H(Z)$. In fact, the difference between these two results in the *mutual information*

$$I(Z; Y) = H(Z) - H(Z|Y), \quad (9.40)$$

which may in some sense be interpreted as the rate at which entropy is extracted in the logical subspace.

Numerical studies reveal that the codespace information is zero for zero error rate, peaks at some definite value at a finite error rate, then decays back to zero. Studying this crossing point may prove fruitful.

Towards a phenomenological evaluation

My discussion of codespace entropy has been in terms of code capacity models without syndrome measurement errors and known noise distribution. It is desirable to generalize this approach to a phenomenological model with syndrome measurement errors. This should be possible and rather straightforward. One should write the quantum-classical channel representing phenomenological syndrome measurement noise, compose this with the noise-syndrome channel, then calculate the codespace entropy of the resulting map.

As I have said, another approach is to just simulate a phenomenological model classically and estimate the CCE empirically.

Analysis of fault tolerance

The analysis presented here provides exact calculations of codespace dynamics. It would interesting and probably fruitful to study ancilla hook error propagation in terms of codespace entropy rates. One could perform the usual numeric simulations to empirically estimate the CCE and study the growth of CCE as a function of ancilla qubit error rates. It would then be expected that non-fault tolerant circuit designs would “blow up” faster than fault tolerant circuit designs.

Tomographic analysis of the error channel

The codespace entropy, as I have developed in this chapter, is a classical metric. It is appropriate for stabilizer codes in the Heisenberg representation of quantum computers [Got98] because the tableau structure permits a binary representation. However, it has the deficiency that one must know the true error effect Z to calculate, which can be done in classical simulation but is impossible through empirical estimation on a quantum device. As is well known, one cannot measure two non-commuting observables (except in the case of continuous measurement of both [Hac+16]), so inferring if a given error commutes with a pair of non-commuting logical operators is not possible with projective measurement. Tomographic techniques should be applicable, and I believe that a full von Neumann entropy theory of codespace entropy would be fruitful.

Rates of convergence of the empirical estimator

I numerically investigated the rate of convergence of an empirical estimator of CCE $H(Z|Y)$ for a fixed depolarization rate and varying number of samples. A complete theory of these convergence rates would be extremely helpful for numerical evaluation of optimal codes with empirical estimation. Here, one samples a large number of errors and evaluates empirical estimators of the entropy rates. It would be useful to have a theory of the size of the bias and the spread of the empirical estimator about the truth. It seems that [Pan03] is a good starting place.

Method of types

Another promising route of further exploration along these directions is to adapt the method of types in this context to make arguments about typical error sequences, which connects to strong notions of typicality [TT12]. This is a promising route that *may permit more efficient training of model*; observe that the number of typical error types is exponentially smaller than the set of all possible errors, yet the typical error types occur with probability approaching 1 in the large sample limit. I think this direction is most promising and would render CCE a usable metric of performance optimization. I present some preliminary analysis on types below:

Let us say that the physical errors are iid. distributed according to a single-qubit error channel

$$\mathcal{E}[\rho] = (1 - p_x - p_y - p_z)I\rho I + p_x X\rho X + p_y Y\rho Y + p_z Z\rho Z \quad (9.41)$$

then the method of types allows us to efficiently optimize over the most likely sequences on a $d \times d$ lattice.

The quantum channel may be represented as a classical distribution $\mathcal{Q}(X)$ where X is a random variable on 4 elements of an alphabet $\mathcal{X} = (I, X, Y, Z)$. We have that $\mathcal{Q}(I) = (1 - p_x - p_y - p_z)$, $\mathcal{Q}(X) = p_x$, etc...

Type The type $P_{\vec{x}}$ of a sequence $\vec{x} = (x_1, x_2, \dots, x_n)$ on an alphabet $\mathcal{X}$ is an empirical distribution on the letters of the alphabet

$$P_{\vec{x}}(a) = \frac{\text{count}(a|\vec{x})}{n} \quad (9.42)$$

where $\text{count}(a|\vec{x})$ is the number of times the character $a \in \mathcal{X}$ appears in $\vec{x}$.

Single-qubit channel types The set of all types of length- n strings of single-qubit Paulis is a set of empirical distribution vectors

$$\left\{ \frac{1}{n} \begin{pmatrix} n - c_x - c_y - c_z \\ c_x \\ c_y \\ c_z \end{pmatrix} : c_x + c_y + c_z \leq n \quad | \quad (c_x, c_y, c_z) \in \mathbb{N} \right\}. \quad (9.43)$$

The set of all types can be represented by the set of all tuples (c_x, c_y, c_z) such that $c_x + c_y + c_z \leq n$.

Type class (composition class) Given a type (empirical distribution) P , the type class $T(P)$, is the set of all sequences that have type P :

$$T(P) = \{\vec{x} \in \mathcal{X}^n : P_{\vec{x}} = P\}. \quad (9.44)$$

Probability of an error string is determined by its type [TT12]

Given $X_1, X_2, \dots, X_n$ i.i.d. distributed according to $\mathcal{Q}$, the probability of realizing a sequence $\vec{x}$ in the type class $T(P)$ depends only on its type and is

$$\mathcal{Q}^n(\vec{x}) = 2^{-n(H(P_{\vec{x}}) + D(P_{\vec{x}} \parallel \mathcal{Q}))}. \quad (9.45)$$

Size of Pauli type classes The size of the type class of type $\frac{1}{n}(n - c, c_x, c_y, c_z)$ is a multinomial coefficient

$$\left| T\left(\frac{1}{n}(n - c, c_x, c_y, c_z)\right) \right| = \binom{n}{n - c, c_x, c_y, c_z} = \frac{n!}{(n - c)!c_x!c_y!c_z!} \quad (9.46)$$

Probability of a type class The probability of a type class $T(P)$ resulting from $\mathcal{Q}^n$ is

$$\mathcal{Q}^n(T(P)) = |T(P)|2^{-n(D(P \parallel \mathcal{Q}) + H(P))} \quad (9.47)$$

Number of Pauli type classes The number of tuples (a, b, c) such that $a + b + c \leq n$ is polynomial in size while the number of Pauli strings on n qubits is combinatorial.

Sampling strategy Our sampling strategy shall be to first evaluate the probability of each type class. Then, we discard all type classes that are below the minimum correctable error weight. Then, we select the remaining type classes that have the highest probability of occurring. Finally, we evaluate the cost function over this set of types.

Explicitly, we seek to calculate the cost of running a code $\mathcal{C}$ for distribution $\mathcal{Q}$

$$J_{\mathcal{Q}}(\mathcal{C}) = \sum_{\vec{x}} \mathcal{Q}^n(\vec{x}) J_{\mathcal{C}}(\vec{x}) \quad (9.48)$$

where $J_{\mathcal{C}}(\vec{x})$ is the cost of error $\vec{x}$ for the code $\mathcal{C}$. This can be, e.g., an indicator that $\vec{x}$ causes a decoder failure, Knill-Laflamme costs, etc...

We can re-write the above as

$$\sum_{\vec{x}} \mathcal{Q}^n(\vec{x}) J_{\mathcal{C}}(\vec{x}) = \sum_P \sum_{\vec{x} \in T(P)} \mathcal{Q}^n(x) J_{\mathcal{C}}(\vec{x}) \quad (9.49)$$

because all $\vec{x} \in T(P)$ have probability $2^{-n(D(P \parallel \mathcal{Q}) + H(P))}$, it follows that

$$\sum_{\vec{x}} \mathcal{Q}^n(\vec{x}) J_{\mathcal{C}}(\vec{x}) = \sum_P 2^{-n(D(P \parallel \mathcal{Q}) + H(P))} \sum_{\vec{x} \in T(P)} J_{\mathcal{C}}(\vec{x}) \quad (9.50)$$

Part V

Conclusion

Chapter 10

Conclusion

Noise robust engineering will produce quantum information processors that can reliably perform information processing tasks beyond the capabilities of any classical device. The primary challenge to this goal is the multifaceted and complex nature of quantum noise processes, which makes implementation of mitigation and correction schemes challenging. This work proposed a 3 step approach to combat the effects of noise in quantum processors: characterize, calibrate, and correct. These steps bridge the gap between low-level device verification and mid-level error correction schemes, see Fig. 1.1.

10.1 Summary

Characterization. In the characterization step, one defines detailed error models of device operation and associated estimation protocols. Chap. 3 discussed the many levels of quantum noise models, which are summarized in Tab. 3.1. Unitary models of coherent error are particularly suitable for coherent error characterization, and hence are most relevant in calibration problems. However, unitary models do not capture the complete dynamics of quantum noise. Quantum channels provide a complete Markovian description of noise that is useful in quantum error correction schemes. In particular, the Pauli noise channel – a specific type of quantum channel – seems to be the proper model for QEC code optimization. Non-Markovian models go beyond quantum channels to describe noise whose characteristics either change or are context dependent. Such sources of noise are particularly difficult to model in generality yet are expected to be present in current devices.

Given detailed models of quantum noise, the next step in characterization is the estimation problem. Here, one seeks to derive protocols that estimate the parameters of a quantum noise model. A complete theory of this problem is provided by GST, which provides a self-consistent approach to fit every free parameter in a Markovian model. Standard, well-established techniques may be deployed to design fiducials to probe the noise effects and germs to amplify the noise effects. Error generators provide a nice Lie theoretic approach model estimation with a convenient taxonomy. A longstanding gauge issue makes GST op-

timization and interpretation difficult. One potential solution to the gauge issue is to fix a particular invariant reference frame.

Despite its value as a complete theory, GST is resource intensive and impractical for calibration when one seeks to minimize processor downtime. As an alternative approach, phase estimation provides much of the benefit of GST without some of the drawback. RPE uses circuits that resemble GST in structure but are designed to estimate only the coherent errors in the device. RPE has the drawback that, unlike GST, the device's operation must be close enough to the target performance in order to operate. The exact allowable distance between the actual device's operation and target operation is quantified in robustness conditions that have been formulated as experimentally relevant consistency checks. When the consistency checks are satisfied, RPE produces precise and accurate estimates of coherent device errors 1) with many less quantum circuits and 2) with significantly less classical computation. Advanced phase techniques may be deployed by combining ideas in RPE with dynamical decoupling and twirling.

Calibration. Calibration concerns the removal of controllable sources of error in quantum devices. It is usually the case that the controllable sources of error are coherent errors that result from sub-optimal classical control configurations. Drift in quantum devices means that particular configuration that is optimal at one moment may no longer be optimal at a subsequent time. Protocols that are designed for real-time and fast-feedback calibration are hence extremely useful. On-line optimization loops that exploit information as it streams in from the device – as opposed to batched optimization after many circuits – are particularly suitable in this regime.

The first step in an on-line control loop is a streaming estimator that updates device model estimates in real-time based on partial observation. The Kalman filter is a celebrated and widespread approach to this problem and already forms the basis of many industrial control techniques. Here, individual quantum circuits are run on the device for some number of shots. While the empirical distribution collected does not look Gaussian in the small sample limit, after a sufficient number of samples (that depends on the Hilbert space of the register in question) the observed empirical distribution will start to look Gaussian distribution about an underlying mean. Furthermore, selecting circuits whose length grows as the uncertainty in model parameters shrinks permits reliable model linearization. In this way, Kalman filters can be deployed for estimation of quantum channels based on tomographic information obtained from probe circuits.

While a streaming estimator is the necessary first step, the actual calibration protocol follows from a feedback law that uses error estimates to update classical control configurations. A very well studied feedback law is the celebrated LQR, which together with the Kalman filter forms foundational theory for much of classical control. The LQR provides control updates are designed to compensate for drift in coherent errors. RPE is a natural protocol for the estimation problem. Using RPE as a basic primitive, an LQR controller may be deployed. I presented three protocols of interest (Tab. 7.2) 1) direct LQR, 2) Kalman

smoothed LQR (LQG), and 3) dual LQG with a changing control model. Numerical and experimental evidence indicate promise for large-scale deployment of this approach.

Correction. Not all sources of error in quantum devices will be controllable. Hence, to achieve high fidelity suitable for algorithm implementation, some active QEC will most likely be necessary. Stabilizer codes have been extensively studied and are thought to be the most promising path to large scale quantum fault tolerance. Chap. 8 provided an introduce to key ideas in QEC. In particular, I discussed various models of QEC code operation, mathematical models of subspace projection, the codespace basis, decoders, and channel representations of quantum codes.

Noise processes generate entropy in the logical and syndrome subspaces of a given code. These entropy rates can be calculated and be used to quantify the expected performance of a code when operated over a large number of syndrome extraction cycles. The CCE theorem demonstrates that conditional codespace entropy is a relevant metric of optimal code performance. While CCE is formally as difficult to evaluate as the construction of an optimal error class decoder, empirical estimation and importance sampling based on the method of types may render CCE an operationally useful metric of code performance.

10.2 Future directions

Characterization.

Markovian noise models of quantum devices are well understood, but non-Markovian noise is still notoriously difficult to model. Developing good non-Markovian models for real quantum devices is a key outstanding question in the field.

Gauge issues continue to cause issues both in interpretation and estimation of gate set models. A complete resolution of gate set gauge problems has proven elusive. Extending the method of invariant frames I introduce in § 4.5 may provide one route towards this goal.

Multi-parameter phase estimation of non-commuting models is difficult due to the complicated algebra of many-qubit operations. Decoupled and twirled RPE § 5.4 provide one route towards this goal. Experimental evaluation of these techniques is an important next step.

Calibration.

Kalman filters provide a means of streaming estimation of quantum tomographic models based on results from individual circuits. In this work, I identified the sigma-point Kalman filter as one of the most promising techniques for large-scale deployment. Full integration of sigma-point filters in GST protocols and experimental demonstrations are an important next step.

Systems with linear control models and Gaussian observation noise are solved optimally with the linear-quadratic regulator. I demonstrated that these approximations are satisfied in a reasonable regime of device operation in a superconducting transmon system. Experimental implementation of these methods for long-term regulation is an important next step.

When a system is not well described by a linear control model, LQR is no longer formally applicable. However, LQR theory can be generalized to polynomial control models and provides a starting point to sophisticated model-based control techniques. Extending the techniques introduced in this work to this regime would be an important and useful result.

Correction

Codespace entropy metrics quantify the expected performance of an optimal decoder in the limit of a large number of cycles. There are many new avenues of exploration offered by this approach, listed § 9 and reviewed below.

Codespace entropy, as defined here, is a classical metric suitable for evaluation in classical simulations of quantum devices. However, estimating codespace entropy on real devices will require tomographic techniques. One will similarly need to generalize the classical description of logical entropy to a fully quantum description. The ideas associated with coherent codespace entropy may provide one route towards this goal.

CCE is formally as difficult to evaluate as the construction of an optimal error class decoder. However, careful importance sampling based on the method of types may render CCE an operationally useful metric of code performance. Evaluation of CCE over a set of candidate codes will then reveal the code that is expected to perform optimally under a given noise distribution.

CCE grows like a sigmoid function under depolarizing noise. Comparing CCE between multiple different codes reveals crossing points in performance. This technique can likely be used to empirically estimate the threshold of optimal decoding without actually constructing an optimal decoder.

Bibliography

- [A J08] A. J. Shaiju and Ian R. Petersen. “Formulas for Discrete Time LQR, LQG, LEQG and Minimax LQG Optimal Control Problems”. In: *IFAC Proceedings Volumes* 41.2 (Jan. 2008), pp. 8773–8778.
- [Aar13] Scott Aaronson. *Quantum computing since Democritus*. en. Cambridge, England: Cambridge University Press, Mar. 2013.
- [Afr+17] Chouaib Afri et al. “State and parameter estimation: A nonlinear Luenberger observer approach”. In: *IEEE Trans. Automat. Contr.* 62.2 (Feb. 2017), pp. 973–980.
- [AH09] I Arasaratnam and S Haykin. “Cubature Kalman Filters”. en. In: *IEEE Trans. Automat. Contr.* 54.6 (June 2009), pp. 1254–1269.
- [AL16] Tameem Albash and Daniel A Lidar. “Adiabatic Quantum Computing”. In: *arXiv [quant-ph]* (Nov. 2016).
- [Ali02] R Alicki. “Invitation to Quantum Dynamical Semigroups”. In: *Dynamics of Dissipation*. Lecture notes in physics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 239–264.
- [AR24] M Sohaib Alam and Eleanor Rieffel. “Dynamical logical qubits in the Bacon-Shor code”. In: *arXiv [quant-ph]* (Mar. 2024).
- [Aru+19] Frank Arute et al. “Quantum supremacy using a programmable superconducting processor”. en. In: *Nature* 574.7779 (Oct. 2019), pp. 505–510.
- [AS09] Koenraad M R Audenaert and Stefan Scheel. “Quantum tomographic reconstruction with error bars: a Kalman filter approach”. en. In: *New J. Phys.* 11.2 (Feb. 2009), p. 023028.
- [ASS19] Nidhi Agarwal, Akanksha Sondhi, and Ghanapriya Singh. “State Estimation of a Quantum System Using Extended Kalman Filter”. In: *2019 International Conference on Cutting-edge Technologies in Engineering (ICoN-CuTE)*. Nov. 2019, pp. 97–100.
- [Bao+21] Feng Bao et al. “Fluxonium: an alternative qubit platform for high-fidelity operations”. In: *arXiv [quant-ph]* (Nov. 2021).

- [Bar+95] A Barenco et al. “Elementary gates for quantum computation”. In: *arXiv [quant-ph]* (Mar. 1995).
- [BB18] Axel Barrau and Silvère Bonnabel. “Invariant Kalman filtering”. en. In: *Annu. Rev. Control Robot. Auton. Syst.* 1.1 (May 2018), pp. 237–257.
- [Bin12] Adam M Bincer. *Lie Groups and Lie Algebras - A Physicist’s Perspective: A Physicist’s Perspective*. en. London, England: Oxford University Press, Oct. 2012.
- [Blu+13] Robin Blume-Kohout et al. “Robust, self-consistent, closed-form tomography of quantum logic gates on a trapped ion qubit”. In: *arXiv [quant-ph]* (Oct. 2013).
- [Blu+22] Robin Blume-Kohout et al. “A Taxonomy of Small Markovian Errors”. In: *PRX Quantum* 3.2 (May 2022), p. 020335.
- [Blu10] Robin Blume-Kohout. “Optimal, reliable estimation of quantum states”. en. In: *New J. Phys.* 12.4 (Apr. 2010), p. 043034.
- [BMR06] S Bonnabel, Ph Martin, and P Rouchon. “Symmetry-preserving observers”. In: *arXiv [math.OC]* (Dec. 2006).
- [BP02] Heinz-Peter Breuer and Francesco Petruccione. *The Theory of Open Quantum Systems*. en. Oxford University Press, 2002.
- [Bri+09] H J Briegel et al. “Measurement-based quantum computation”. en. In: *Nat. Phys.* 5.1 (Jan. 2009), pp. 19–26.
- [BRK23] Raphael Brieger, Ingo Roth, and Martin Kliesch. “Compressive gate set tomography”. en. In: *PRX quantum* 4.1 (Mar. 2023), p. 010325.
- [BT08] Dimitri Bertsekas and John N Tsitsiklis. *Introduction to Probability*. en. Athena Scientific, July 2008.
- [CAE19] Arnaud Carignan-Dugas, Matthew Alexander, and Joseph Emerson. “A polar decomposition for quantum channels (with applications to bounding error propagation in quantum circuits)”. In: *Quantum* 3 (Aug. 2019), p. 173.
- [CB19] Zhenyu Cai and Simon C Benjamin. “Constructing smaller Pauli twirling sets for arbitrary error channels”. en. In: *Sci. Rep.* 9.1 (Aug. 2019), p. 11281.
- [Cha+12] L Chang et al. “Marginalised iterated unscented Kalman filter”. en. In: *IET Control Theory Appl.* 6.6 (Apr. 2012), pp. 847–854.
- [Che+23] Senrui Chen et al. “The learnability of Pauli noise”. en. In: *Nat. Commun.* 14.1 (Jan. 2023), p. 52.
- [Che+24] Senrui Chen et al. “Efficient self-consistent learning of gate set Pauli noise”. In: *arXiv [quant-ph]* (Oct. 2024).
- [Che+25] Kean Chen et al. “Verifying fault-tolerance of quantum error correction codes”. In: *arXiv [quant-ph]* (Jan. 2025).

- [Cle17] Jeanne N Clelland. *From frenet to cartan: The method of moving frames*. Graduate studies in mathematics. Providence, RI: American Mathematical Society, Mar. 2017.
- [Col+23] Luis Colmenarez et al. “Accurate optimal quantum error correction thresholds from coherent information”. In: *arXiv [quant-ph]* (Dec. 2023).
- [CR20] Rui Chao and Ben W Reichardt. “Flag Fault-Tolerant Error Correction for any Stabilizer Code”. In: *PRX Quantum* 1.1 (Sept. 2020), p. 010302.
- [CRW20] Jahan Claes, Eleanor Rieffel, and Zhihui Wang. “Character randomized benchmarking for non-multiplicity-free groups with applications to subspace, leakage, and matchgate randomized benchmarking”. In: (Oct. 2020).
- [DC05] A Das and B K Chakrabarti. *Quantum annealing and related optimization methods*. en. Ed. by Arnab Das and B K Chakrabarti. 2005th ed. Lecture notes in physics. Berlin, Germany: Springer, Nov. 2005.
- [Deu89] D Deutsch. “Quantum computational networks”. en. In: *Proc. R. Soc. Lond.* 425.1868 (Sept. 1989), pp. 73–90.
- [DH21] Nicolas Delfosse and Matthew B Hastings. “Union-find decoders for homological product codes”. In: *Quantum* 5 (Mar. 2021), p. 406.
- [Di +20] Olivia Di Matteo et al. “Operational, gauge-free quantum tomography”. en. In: *Quantum* 4.364 (Nov. 2020), p. 364.
- [DP10a] Guillaume Duclos-Cianci and David Poulin. “A renormalization group decoding algorithm for topological quantum codes”. In: *2010 IEEE Information Theory Workshop*. Aug. 2010, pp. 1–5.
- [DP10b] Guillaume Duclos-Cianci and David Poulin. “A renormalization group decoding algorithm for topological quantum codes”. In: *2010 IEEE Information Theory Workshop*. IEEE, Aug. 2010, pp. 1–5.
- [Ebe+20] Donald Ebeigbe et al. “Poisson Kalman filter for disease surveillance”. In: *Phys. Rev. Res.* 2.4 (Oct. 2020), p. 043028.
- [Edm65] Jack Edmonds. “Paths, trees, and flowers”. en. In: *Canad. J. Math.* 17.0 (Jan. 1965), pp. 449–467.
- [EK09] Bryan Eastin and Emanuel Knill. “Restrictions on transversal encoded quantum gate sets”. en. In: *Phys. Rev. Lett.* 102.11 (Mar. 2009), p. 110502.
- [Eva+22] T J Evans et al. “Fast Bayesian Tomography of a Two-Qubit Gate Set in Silicon”. In: *Phys. Rev. Appl.* 17.2 (Feb. 2022), p. 024068.
- [Eve03] Geir Evensen. “The Ensemble Kalman Filter: theoretical formulation and practical implementation”. In: *Ocean Dyn.* 53.4 (Nov. 2003), pp. 343–367.
- [EWP17] Muhammad F Emzir, Matthew J Woolley, and Ian R Petersen. “A quantum extended Kalman filter”. en. In: *J. Phys. A: Math. Theor.* 50.22 (May 2017), p. 225301.

- [Fan+24] Yale Fan et al. “Randomized benchmarking with synthetic quantum circuits”. In: *arXiv [quant-ph]* (Dec. 2024).
- [Fer+18] Chris Ferrie et al. “Bayesian quantum noise spectroscopy”. In: *New J. Phys.* 20.12 (Dec. 2018), p. 123005.
- [Fey60] Richard P Feynman. “There’s plenty of room at the bottom”. en. In: *Eng. Sci.* 23.5 (Feb. 1960), pp. 22–36.
- [GA10] Mohinder S Grewal and Angus P Andrews. “Applications of Kalman Filtering in Aerospace 1960 to the Present [Historical Perspectives]”. In: *IEEE Control Syst. Mag.* 30.3 (June 2010), pp. 69–78.
- [GA14] Mohinder S Grewal and Angus P Andrews. *Kalman Filtering: Theory and Practice with MATLAB*. en. John Wiley & Sons, Dec. 2014.
- [GB21] Riddhi Swaroop Gupta and Michael J Biercuk. “Adaptive filtering of projective quantum measurements using discrete stochastic methods”. In: *Phys. Rev. A* 104.1 (July 2021), p. 012412.
- [GCC16] Christopher Granade, Joshua Combes, and D G Cory. “Practical Bayesian tomography”. en. In: *New J. Phys.* 18.3 (Mar. 2016), p. 033024.
- [Ger+03] J M Geremia et al. “Quantum Kalman filtering and the Heisenberg limit in atomic magnetometry”. en. In: *Phys. Rev. Lett.* 91.25 (Dec. 2003), p. 250801.
- [GL13] Tryphon T Georgiou and Anders Lindquist. “The separation principle in stochastic control, redux”. en. In: *IEEE Trans. Automat. Contr.* 58.10 (Oct. 2013), pp. 2481–2494.
- [GLM06] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. “Quantum metrology”. en. In: *Phys. Rev. Lett.* 96.1 (Jan. 2006), p. 010401.
- [Goo25] Google Quantum AI and Collaborators. “Quantum error correction below the surface code threshold”. en. In: *Nature* 638.8052 (Feb. 2025), pp. 920–926.
- [Gor+19] Wojciech Gorecki et al. “ π -Corrected Heisenberg Limit”. In: *arXiv [quant-ph]* (July 2019).
- [Got24] Daniel Gottesman. “Surviving as a Quantum Computer in a Classical World”. In: (2024).
- [Got97] Daniel Gottesman. “Stabilizer Codes and Quantum Error Correction”. In: *arXiv [quant-ph]* (May 1997).
- [Got98] Daniel Gottesman. “The Heisenberg representation of quantum computers”. In: *arXiv [quant-ph]* (July 1998).
- [Gre15] Daniel Greenbaum. “Introduction to Quantum Gate Set Tomography”. In: *arXiv [quant-ph]* (Sept. 2015).
- [Gri+19] Alexander Grimm et al. “The Kerr-cat qubit: Stabilization, readout, and gates”. In: *arXiv [quant-ph]* (July 2019).

- [GS14] Graham C Goodwin and Kwai Sang Sin. *Adaptive Filtering Prediction and Control*. en. Courier Corporation, May 2014.
- [Gu+21] Yanwu Gu et al. “Randomized Linear Gate-Set Tomography”. In: *PRX Quantum* 2.3 (Aug. 2021), p. 030328.
- [GZ13] Michael R Geller and Zhongyuan Zhou. “Efficient error models for fault-tolerant architectures and the Pauli twirling approximation”. en. In: *Phys. Rev. A* 88.1 (July 2013), p. 012314.
- [Hac+16] Shay Hacohe-Gourgy et al. “Quantum dynamics of simultaneously measured non-commuting observables”. en. In: *Nature* 538.7626 (Oct. 2016), pp. 491–494.
- [Hal16] Brian C Hall. *Lie groups, lie algebras, and representations*. en. 2nd ed. Graduate texts in mathematics. Cham, Switzerland: Springer International Publishing, Aug. 2016.
- [Has+24] Akel Hashim et al. “A practical introduction to benchmarking and characterization of quantum computers”. In: *arXiv [quant-ph]* (Aug. 2024).
- [Hay04] Simon Haykin. *Kalman Filtering and Neural Networks*. en. John Wiley & Sons, Mar. 2004.
- [Hay16] Masahito Hayashi. *Group representation for quantum theory*. en. 1st ed. Cham, Switzerland: Springer International Publishing, Nov. 2016.
- [HB20] Shilin Huang and Kenneth R Brown. “Fault-tolerant compass codes”. In: *Phys. Rev. A* 101.4 (Apr. 2020), p. 042312.
- [HBG24] Lukas Hantzko, Lennart Binkowski, and Sabhyata Gupta. “Pauli Transfer Matrices”. In: (Nov. 2024).
- [Hel+18] Jonas Helsen et al. “A new class of efficient randomized benchmarking protocols”. In: *arXiv [quant-ph]* (June 2018).
- [HH21] Matthew B Hastings and Jeongwan Haah. “Dynamically generated logical qubits”. In: *Quantum* 5 (Oct. 2021), p. 564.
- [Hig+07] B L Higgins et al. “Entanglement-free Heisenberg-limited phase estimation”. en. In: *Nature* 450.7168 (Nov. 2007), pp. 393–396.
- [Hig21] Oscar Higgott. “PyMatching: A Python package for decoding quantum codes with minimum-weight perfect matching”. In: *arXiv [quant-ph]* (May 2021).
- [HS15] Jindřich Havlík and Ondřej Straka. “Performance evaluation of iterated extended Kalman filter with variable step-length”. In: *J. Phys.: Conf. Ser.* 659 (Nov. 2015), p. 012022.
- [IOY10] Sanae Iida, Kentaro Ohki, and Naoki Yamamoto. “Robust quantum Kalman filtering under the phase uncertainty of the probe-laser”. In: *2010 IEEE International Symposium on Computer-Aided Control System Design*. Sept. 2010, pp. 749–754.

- [IP15] Pavithran Iyer and David Poulin. “Hardness of decoding quantum stabilizer codes”. en. In: *IEEE Trans. Inf. Theory* 61.9 (Sept. 2015), pp. 5209–5223.
- [Jou+16] Marine Jouin et al. “Particle filter-based prognostics: Review, discussion and perspectives”. en. In: *Mech. Syst. Signal Process.* 72-73 (May 2016), pp. 2–31.
- [JU04] S J Julier and J K Uhlmann. “Unscented filtering and nonlinear estimation”. en. In: *Proc. IEEE Inst. Electr. Electron. Eng.* 92.3 (Mar. 2004), pp. 401–422.
- [JU97] Simon J Julier and Jeffrey K Uhlmann. “New extension of the Kalman filter to nonlinear systems”. en. In: *Signal Processing, Sensor Fusion, and Target Recognition VI*. Vol. 3068. SPIE, July 1997, pp. 182–193.
- [Kal60] R E Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *J. Basic Eng* 82.1 (Mar. 1960), pp. 35–45.
- [Kal63] R E Kalman. “Mathematical Description of Linear Dynamical Systems”. In: *Journal of the Society for Industrial and Applied Mathematics Series A Control* 1.2 (Jan. 1963), pp. 152–192.
- [KET73] Shauying R Kou, David L Elliott, and Tzyh Jong Tarn. “Observability of nonlinear systems”. en. In: *Infect. Control* 22.1 (Feb. 1973), pp. 89–99.
- [KIF08] Rambabu Kandepu, Lars Imsland, and Bjarne A Foss. “Constrained state estimation using the Unscented Kalman Filter”. In: *2008 16th Mediterranean Conference on Control and Automation*. IEEE, June 2008, pp. 1453–1458.
- [KL23] Kao-Yueh Kuo and Ching-Yi Lai. “Generalized quantum data-syndrome codes and belief propagation decoding for phenomenological noise”. In: *arXiv [quant-ph]* (Oct. 2023).
- [KL97] Emanuel Knill and Raymond Laflamme. “Theory of quantum error-correcting codes”. In: *Phys. Rev. A* 55.2 (Feb. 1997), pp. 900–911.
- [Kla08] J Klamka. “Controllability of dynamical systems”. In: *Mat. Stosow.* (2008).
- [Kla13] J Klamka. “Controllability of dynamical systems. A survey”. In: *Bull. Pol. Acad. Sci. Tech. Sci.* 61.2 (June 2013), pp. 335–342.
- [KLY15] Shelby Kimmel, Guang Hao Low, and Theodore J Yoder. “Robust calibration of a universal single-qubit gate set via robust phase estimation”. In: *Phys. Rev. A* 92.6 (Dec. 2015), p. 062315.
- [KLZ96] E Knill, R Laflamme, and W Zurek. “Threshold Accuracy for Quantum Computation”. In: *arXiv [quant-ph]* (Oct. 1996).
- [Koc+07] Jens Koch et al. “Charge insensitive qubit design derived from the Cooper pair box”. In: *arXiv [cond-mat.mes-hall]* (Feb. 2007).
- [Kri+05] David W Kribs et al. “Operator quantum error correction”. In: *arXiv [quant-ph]* (Apr. 2005).
- [Kün13] H R Künsch. “Particle filters”. In: *Bernoulli* 19.4 (2013), pp. 1381–1403.

- [KV16] P R Kumar and Pravin Varaiya. *Stochastic Systems*. en. New York, NY: Society for Industrial & Applied Mathematics, 2016.
- [Lab20] Roger Labbe. *Kalman and Bayesian Filters in Python*. Github, 2020.
- [LB13] Daniel A Lidar and Todd A Brun. *Quantum Error Correction*. en. Cambridge University Press, Sept. 2013.
- [Leg05] A J Leggett. “The quantum measurement problem”. en. In: *Science* 307.5711 (Feb. 2005), pp. 871–872.
- [Lid12] Daniel A Lidar. “Review of decoherence free subspaces, noiseless subsystems, and dynamical decoupling”. In: *arXiv [quant-ph]* (Aug. 2012).
- [Lid19] Daniel A Lidar. “Lecture notes on the theory of open quantum systems”. In: *arXiv [quant-ph]* (Feb. 2019).
- [LP08] M S Leifer and D Poulin. “Quantum graphical models and belief propagation”. en. In: *Ann. Phys. (N. Y.)* 323.8 (Aug. 2008), pp. 1899–1946.
- [Man19] Daniel Manzano. “A short introduction to the Lindblad Master Equation”. In: *arXiv [quant-ph]* (June 2019).
- [Mar+21] John M Martyn et al. “Grand unification of quantum algorithms”. en. In: *PRX quantum* 2.4 (Dec. 2021).
- [Mar25] J P Marceaux. “Marceaux Thesis Repo”. May 2025.
- [Mer+13] Seth T Merkel et al. “Self-consistent quantum process tomography”. en. In: *Phys. Rev. A* 87.6 (June 2013), p. 062119.
- [MGE11] Easwar Magesan, J M Gambetta, and Joseph Emerson. “Scalable and robust randomized benchmarking of quantum processes”. en. In: *Phys. Rev. Lett.* 106.18 (May 2011), p. 180504.
- [Min22] Zlatko Minev. *A tutorial on tailoring quantum noise - Twirling 101 (Parts I–IV)*. en. <https://www.zlatko-minev.com/blog/twirling>. Accessed: 2024-11-20. Sept. 2022.
- [MPF18] Seth T Merkel, Emily J Pritchett, and Bryan H Fong. “Randomized Benchmarking as convolution: Fourier analysis of gate dependent errors”. In: *arXiv [quant-ph]* (Apr. 2018).
- [MS83] Richard J Meinhold and Nozer D Singpurwalla. “Understanding the Kalman Filter”. In: *Am. Stat.* 37.2 (May 1983), pp. 123–127.
- [MY23] J P Marceaux and Kevin Young. “Streaming quantum gate set tomography using the extended Kalman filter”. In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, Sept. 2023.
- [NC02] Michael A Nielsen and Isaac Chuang. “Quantum Computation and Quantum Information”. In: *Am. J. Phys.* 70.5 (May 2002), pp. 558–559.

- [Ngu+21] Nhung H Nguyen et al. “Demonstration of Shor encoding on a trapped-ion quantum computer”. In: *arXiv [quant-ph]* (Apr. 2021).
- [Nie+20] Erik Nielsen et al. “Probing quantum processor performance with pyGSTi”. en. In: *Quantum Sci. Technol.* 5.4 (July 2020), p. 044002.
- [Nie+21] Erik Nielsen et al. “Gate set tomography”. en. In: *Quantum* 5.557 (Oct. 2021), p. 557.
- [Niu+19] Murphy Yuezhen Niu et al. “Learning non-Markovian quantum noise from Moir{e}-enhanced swap spectroscopy with deep evolutionary algorithm”. In: *arXiv [quant-ph]* (Dec. 2019).
- [NYB22] Erik Nielsen, Kevin Young, and Robin Blume-Kohout. “First-order gauge-invariant error rates in quantum processors”. In: vol. 2022. Jan. 2022, p. M38.009.
- [OC16] Joe O’Gorman and Earl T Campbell. “Quantum computation with realistic magic state factories”. In: *arXiv [quant-ph]* (May 2016).
- [Oll+24] Jan Olle et al. “Simultaneous discovery of quantum error correction codes and encoders with a noise-aware reinforcement learning agent”. en. In: *Npj Quantum Inf.* 10.1 (Dec. 2024), pp. 1–17.
- [Olv99] Peter J Olver. *London mathematical society student texts: Classical invariant theory series number 44*. Cambridge, England: Cambridge University Press, Jan. 1999.
- [Ost+23] Corey Ostrove et al. “Near-minimal gate set tomography experiment designs”. In: *arXiv [quant-ph]* (Aug. 2023).
- [Pan03] L Paninski. “Estimation of entropy and mutual information”. In: *Neural Comput.* 15 (June 2003), pp. 1191–1253.
- [PDW10] Ksenia Ponomareva, Paresh Date, and Zidong Wang. “A new unscented Kalman filter with higher order moment-matching”. In: *Mathematical Theory of Networks and Systems*, 2010.
- [Pol+18] Felix A Pollock et al. “Non-Markovian quantum processes: Complete framework and efficient characterization”. en. In: *Phys. Rev. A* 97.1 (Jan. 2018), p. 012127.
- [Pou05] David Poulin. “Stabilizer formalism for operator quantum error correction”. In: *arXiv [quant-ph]* (Aug. 2005).
- [Pro+17] Timothy Proctor et al. “What randomized benchmarking actually measures”. In: *arXiv [quant-ph]* (Feb. 2017).
- [RB01] R Raussendorf and H J Briegel. “A one-way quantum computer”. en. In: *Phys. Rev. Lett.* 86.22 (May 2001), pp. 5188–5191.
- [Rei+21] Maximilian Reichert et al. “Nonlinear quantum error correction”. In: *arXiv [quant-ph]* (Dec. 2021).

- [Rei+24] Ben W Reichardt et al. “Demonstration of quantum computation and error correction with a tesseract code”. In: *arXiv [quant-ph]* (Sept. 2024).
- [RF23] Cambyse Rouzé and Daniel Stilck França. “Efficient learning of the structure and parameters of local Pauli noise channels”. In: *arXiv [quant-ph]* (July 2023).
- [RH11] Ángel Rivas and Susana F Huelga. “Open Quantum Systems. An Introduction”. In: *arXiv [quant-ph]* (Apr. 2011).
- [RMM22] Simone Roncallo, Lorenzo Maccone, and Chiara Macchiavello. “Pauli transfer matrix direct reconstruction: channel characterization without full process tomography”. In: *arXiv [quant-ph]* (Dec. 2022).
- [Rob96] Robert M. Corless and David J. Jeffrey. “The unwinding number”. In: *SIGSAM bull.* 116 (July 1996), pp. 28–35.
- [Rud+21] Kenneth Rudinger et al. “Experimental characterization of crosstalk errors with simultaneous gate set tomography”. en. In: *PRX quantum* 2.4 (Nov. 2021), p. 040338.
- [Rud+23] Kenneth M Rudinger et al. “Two-qubit gate set tomography with fewer circuits”. In: *arXiv [quant-ph]* (July 2023).
- [Rud+25] Kenneth Rudinger et al. “Heisenberg-limited calibration of entangling gates with robust phase estimation”. In: *arXiv [quant-ph]* (Feb. 2025).
- [Rud86] Walter Rudin. *Real and Complex Analysis*. en. 3rd ed. McGraw-Hill series in higher mathematics. New York, NY: McGraw-Hill Professional, Sept. 1986.
- [Rus+21a] A E Russo et al. “Evaluating Energy Differences on a Quantum Computer with Robust Phase Estimation”. en. In: *Phys. Rev. Lett.* 126.21 (May 2021), p. 210501.
- [Rus+21b] Antonio E Russo et al. “Consistency testing for robust phase estimation”. In: *Phys. Rev. A* 103.4 (Apr. 2021), p. 042609.
- [Sch05] Maximilian Schlosshauer. “Decoherence, the measurement problem, and interpretations of quantum mechanics”. en. In: *Rev. Mod. Phys.* 76.4 (Feb. 2005), p. 1267.
- [Sch96] Benjamin Schumacher. “Sending quantum entanglement through noisy channels”. In: *arXiv [quant-ph]* (Apr. 1996).
- [SDŠ14] Ondřej Straka, Jindřich Duník, and Miroslav Šimandl. “Measures of non-Gaussianity in unscented Kaiman filter framework”. In: *17th International Conference on Information Fusion (FUSION)*. July 2014, pp. 1–8.
- [Sha12] R Shankar. *Principles of quantum mechanics*. en. New York, NY: Springer, July 2012.
- [Shi+14] Seung Woo Shin et al. “How “quantum” is the D-Wave machine?” In: *arXiv [quant-ph]* (Jan. 2014).

- [Sho95] P W Shor. “Scheme for reducing decoherence in quantum computer memory”. en. In: *Phys. Rev. A* 52.4 (Oct. 1995), R2493–R2496.
- [Sil+24] Allyson Silva et al. “Optimizing multi-level magic state factories for fault-tolerant quantum architectures”. In: *arXiv [quant-ph]* (Nov. 2024).
- [Smi+24] M C Smith et al. “Single-qubit gates with errors at the 10^{-7} level”. In: *arXiv [quant-ph]* (Dec. 2024).
- [SN20] J J Sakurai and Jim Napolitano. *Modern quantum mechanics*. en. 3rd ed. Cambridge, England: Cambridge University Press, Sept. 2020.
- [SP19] Sergei Slussarenko and Geoff J Pryde. “Photonic quantum information processing: A concise review”. en. In: *Appl. Phys. Rev.* 6.4 (Dec. 2019), p. 041303.
- [Spe16] Maarten Speekenbrink. “A tutorial on particle filters”. en. In: *J. Math. Psychol.* 73 (Aug. 2016), pp. 140–152.
- [Ste96] Andrew Steane. “Multiple-particle interference and quantum error correction”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452.1954 (Nov. 1996), pp. 2551–2577.
- [Sur24] Samyak Surti. *Adaptive surface codes*. Unpublished. 2024.
- [The74] The Analytic Sciences Corporation. *Applied Optimal Estimation*. en. MIT Press, May 1974.
- [Tos13] Kato Tosio. “On the Adiabatic Theorem of Quantum Mechanics”. en. In: *Journal of the Physical Society of Japan* (Dec. 2013).
- [TT12] Thomas M. Cover and Joy A Thomas. *Elements of Information Theory*. en. 2nd ed. Nashville, TN: John Wiley & Sons, Nov. 2012.
- [VLK99] Lorenza Viola, Seth Lloyd, and Emanuel Knill. “Universal control of decoupled quantum systems”. In: *arXiv [quant-ph]* (June 1999).
- [Vui+19] Christophe Vuillot et al. “Code deformation and lattice surgery are gauge fixing”. en. In: *New J. Phys.* 21.3 (Mar. 2019), p. 033028.
- [VW81] H Van de Water and J Willems. “The certainty equivalence property in stochastic control theory”. en. In: *IEEE Trans. Automat. Contr.* 26.5 (Oct. 1981), pp. 1080–1087.
- [Wat18] John Watrous. *The Theory of Quantum Information*. en. Cambridge University Press, Apr. 2018.
- [WBC11] Christopher J Wood, Jacob D Biamonte, and David G Cory. “Tensor networks and graphical calculus for open quantum systems”. In: *arXiv [quant-ph]* (Nov. 2011).
- [WE16] Joel J Wallman and Joseph Emerson. “Noise tailoring for scalable quantum computation via randomized compiling”. en. In: *Phys. Rev. A* 94.5 (Nov. 2016), p. 052325.

- [Whi+21] Gregory A L White et al. “Non-Markovian Quantum Process Tomography”. In: *arXiv [quant-ph]* (June 2021).
- [Wil13] Mark M Wilde. *Quantum Information Theory*. en. Cambridge, England: Cambridge University Press, Apr. 2013.
- [WN03] Eric A Wan and Alex T Nelson. “Dual Extended Kalman Filter Methods”. In: *Kalman Filtering and Neural Networks*. New York, USA: John Wiley & Sons, Inc., 2003, pp. 123–173.
- [WV00] E A Wan and R Van Der Merwe. “The unscented Kalman filter for nonlinear estimation”. In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*. Oct. 2000, pp. 153–158.
- [XX07] Yang Xiang and Shi-Jie Xiong. “Entropy exchange, coherent information and concurrence”. In: *arXiv [quant-ph]* (May 2007).
- [ZM19] Junbo Zhao and Lamine Mili. “A Theoretical Framework of Robust H -Infinity Unscented Kalman Filter and Its Application to Power System Dynamic State Estimation”. In: *IEEE Trans. Signal Process.* 67.10 (May 2019), pp. 2734–2746.
- [ZTC22] Haiquan Zhao, Boyu Tian, and Badong Chen. “Robust stable iterated unscented Kalman filter based on maximum correntropy criterion”. en. In: *Automatica (Oxf.)* 142.110410 (Aug. 2022), p. 110410.