A Bayesian Approach to Single-Qubit Gate Calibration in Trapped Ion Quantum Computers

by
Tai Xiang

Dr. Bryce Bjork, Prof. Gabriel Chandler, Prof. Theresa Lynn

A thesis submitted in partial fulfillment
of the requirements for the
Degree of Bachelor of Arts with Honors
in Physics

POMONA COLLEGE
Claremont, CA
May 2, 2023

Abstract

To realize an ion trap quantum computer that is capable of scaling to a qubit count useful for accomplishing user-defined tasks, additional refinements must be made to the calibration protocol for gating. In this thesis, we explore a multitude of improvements to the parameter estimation procedure involved in single qubit gate calibration with iterative Bayesian estimation, namely the estimation of the Rabi rate Ω and the detuning δ. We first build up the Bayesian formalism required for parameter estimation and explore a simplified case that only involves estimation of the Ω value. We then demonstrate a set of optimizations on speed and robustness for the Bayesian estimator, including an adaptive gating method that enables the estimator to beat the parametric rate of convergence, a set of parameter drift detection and correction mechanisms, and a batched estimation protocol. We explore the effects of SPAM error and parameter drift on our estimation procedure, finding that SPAM error has minimal effect while parameter drift at even low magnitudes can yield divergence. To combat this, we develop an efficient drift detection method and present two drift-resilient calibration methods. We then investigate the full two parameter estimation case, demonstrating that the optimizations realized in the simplified single parameter case translate effectively and determining the ideal experimental settings to yield the fastest rate of convergence for both parameters. We refine the two-parameter estimation procedure, realizing a robust and accurate two-parameter estimation protocol.

Executive Summary

Quantum computing substitutes quantum bits for classical bits and leverages quantum concepts such as superposition and entanglement to achieve speedups over their classical counterparts. Ion trap quantum computing has emerged as a strong hardware platform, demonstrating high fidelity single qubit[1] and two qubit gates[2], high coherence times[3] and schemes for scaling [4].

To ensure high fidelity single qubit gates, calibration of the Rabi rate Ω and the detuning δ, two parameters that arise from laser intensity and frequency respectively, to some desired Ω_{opt} and δ_{opt} is required. However, Ω and δ cannot be directly measured, and are rather estimated with knowledge of the ion populations transition probability. This is typically done by a trained operator through repeated measurements and a scan over a large set of bounded laser settings [5]. However, this procedure can be slow and data in-efficient, and thus we develop a Bayesian formulation for the estimation of Ω and δ.

We begin with a simplified single-parameter toy model that solely estimates Ω and assumes $\delta = 0$. To estimate Ω we perform a series of rotational gates on a qubit, with the number of gates performed being a function of the prior estimate of Ω, and then use the measurement of its final state to inform and update the estimator. In the first baseline case, to demonstrate the feasibility of the estimator, we only perform a single rotational gate prior to measurement. We show that the Bayesian estimator consistently converges at a rate of $n^{-1/2}$, a result consistent with the fastest possible rate of convergence for parametric problems. The results of this baseline estimator are shown in figure 1.

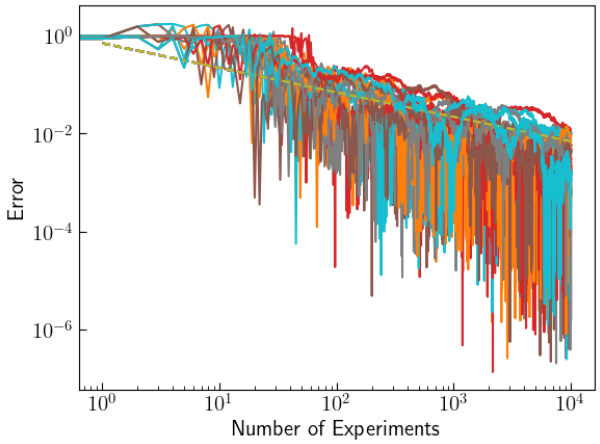


Figure 1: 20 estimation runs of the simplified single-parameter toy model. The yellow dashed line denotes the parametric rate $n^{-1/2}$.

We then improve upon this initial model with a set of optimizations, beginning with an implementation of an adaptive gating protocol that alters the quantity of rotational gates performed prior to readout as a function of the prior standard deviation. As we get better and better estimates of Ω, we perform more rotational gates, and thus gain more information on the system. Doing so yields an estimation protocol that overcomes the parametric rate of convergence, and the results of this protocol are displayed in figure 2.

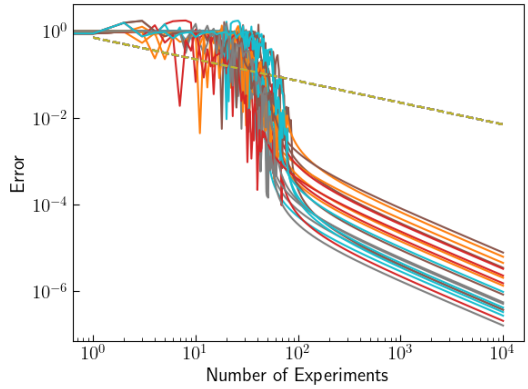
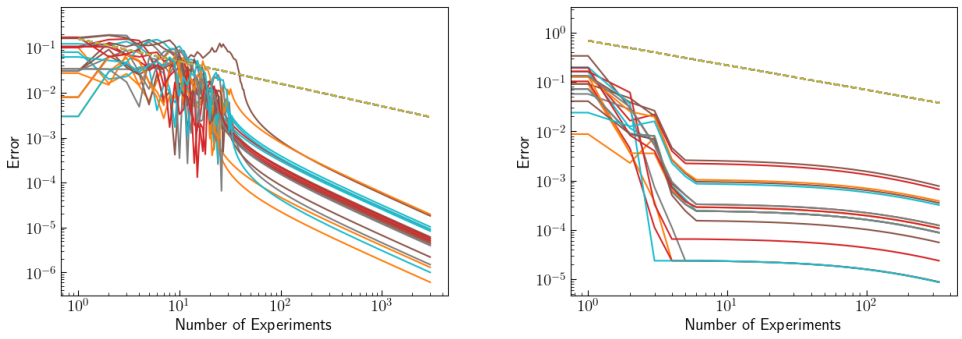


Figure 2: 20 estimation runs of the simplified single-parameter toy model with an adaptive rotational gating protocol The yellow dashed line denotes the parametric rate.

However, a key feature of figure 2 is the flat line of minimal information gain that occurs from experiments 0 - 80, which arises due to poor fitting of the posterior. To resolve this issue, we take advantage of the fact that our sampling function for the Bayes estimator yields Bernoulli distributions. Batching these Bernoulli distributions together yields a binomial distribution that is well-approximated by a normal distribution, and thus we develop a batched Bayesian estimator that combines normal priors with normal likelihoods. With

this batching paradigm, we develop another set of algorithms, including a batch-to-single (B2S) and efficient greedy batched (EGB) algorithm. These algorithms are capable of gaining information at the start of estimation, as the Bayesian estimation iterations take in a normal prior and a normal likelihood and thus result in a normal posterior. Trials for both algorithms are shown in figure 3.



(a) (b)

Figure 3: 20 runs of the B2S algorithm (3a) and 20 runs of the EGB algorithm (3b).

We benchmark this set of algorithms with metrics provided from the Honeywell QCCD paper [4], yielding the results shown in table 3.2. We find that the efficient greedy batched algorithm has the fastest physical run-time, though it requires additional computational machinery to protect against divergence.

We also investigate possible errors that may arise during the estimation process, such as state preparation and measurement (SPAM) error and parameter drift. We find that our Bayesian procedure is robust to SPAM error, though it is affected by parameter drift. We focus on linear parameter drift, developing a maximum likelihood estimation method that shows the strongly linear nature of the maximum likelihood estimates at linear drifts above 3.33×10^{-7}, as shown in figure 4.

Baseline

Metric Type	**Metric Value**	**Operation Type**	**Total Operation Duration (ms)**
Total data taken	1000	State preparation and measurement	130
Sum of all k values	1000	Single qubit gate	5

Total Time: 135ms

Single Cycle K-optimized

Metric Type	**Metric Value**	**Operation Type**	**Total Operation Duration (ms)**
Total data taken	208 ± 10	State preparation and measurement	27.04 ± 1.30
Sum of all k values	15787 ± 125	Single qubit gate	78.94 ± 0.63

Total Time: 105.98 ± 1.93ms

Efficient Greedy Batched

Metric Type	**Metric Value**	**Operation Type**	**Total Operation Duration (ms)**
Total data taken	266 ± 5	State preparation and measurement	34.58 ± 0.65
Sum of all k values	10892 ± 347	Single qubit gate	54.46 ± 0.17

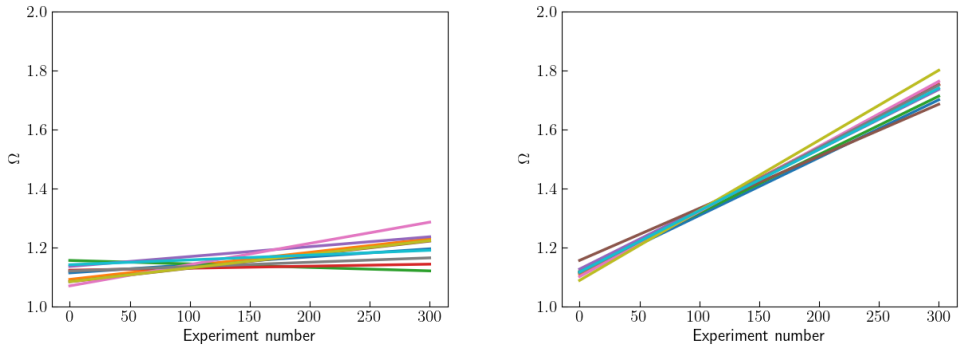
Total Time: 89.04 ± 0.82ms

B2S

Metric Type	**Metric Value**	**Operation Type**	**Total Operation Duration (ms)**
Total data taken	277 ± 2	State preparation and measurement	36.01 ± 0.26
Sum of all k values	15936 ± 84	Single qubit gate	79.68 ± 0.42

Total Time: 115.69 ± 0.68ms

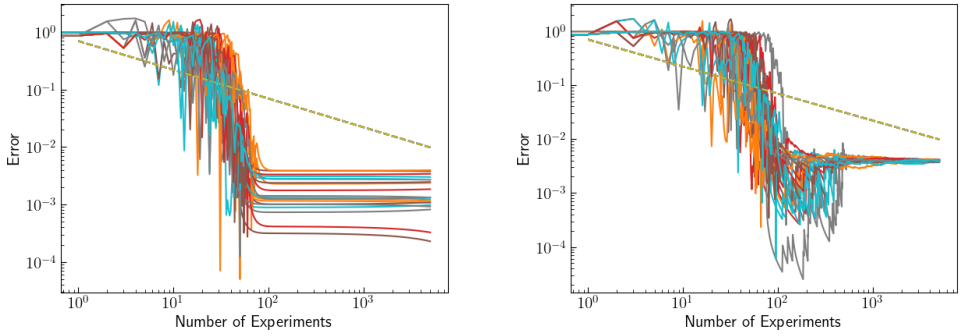
Table 1: A table of physical procedure timing metrics for baseline, single-cycle adaptive, greedy batched, efficient greedy batched, and B2S algorithms. The baseline case never converges to the desired Bayesian standard deviation and thus it always holds a high and fixed value.



(a) (b)

Figure 4: Linear regressions of MLE values obtain from runs at a linear magnitude of 3.33×10^{-7} per measurement (figure 4a), and at a linear magnitude of 3.33×10^{-6} per measurement (figure 4b). All MLE lines display strong linear relationships.

Alongside a detection protocol, we also develop two methods, one that involves sampling the number of rotational gates performed from a distribution and another that exponentiates the posterior standard deviation at a fixed quantity after each trial, to combat parameter drift. We successfully demonstrate convergence of the estimator with both methods, though at the cost of a higher standard deviation of convergence. These results are shown in figure 5.



(a) (b)

Figure 5: Figure 5a demonstrates the results of 20 estimation runs with a linear drift magnitude of 0.0001 over 3000 trials utilizing the exponentiation strategy. Figure 5b demonstrates the results of 20 estimation runs with a linear drift magnitude of 0.0001 over 3000 trials utilizing the adaptive rotational gate sampling strategy.

We then move on to the full two-parameter case, where we derive a new likelihood for the Ramsey experiment, an additional experiment that displays greater sensitivity for the estimation of δ. We validate that this new likelihood adequately converges by creating another simplified one-parameter calibration, this time fixing Ω and performing δ estimation

with the Ramsey likelihood. We observe that all of the optimizations demonstrated for the one parameter Ω case similarly carry over to the Ramsey experiment, as shown in figure 6, where we demonstrate the baseline estimator, an adaptive precession time-optimized algorithm similar to the adaptive gating algorithm, and a batching algorithm. We then

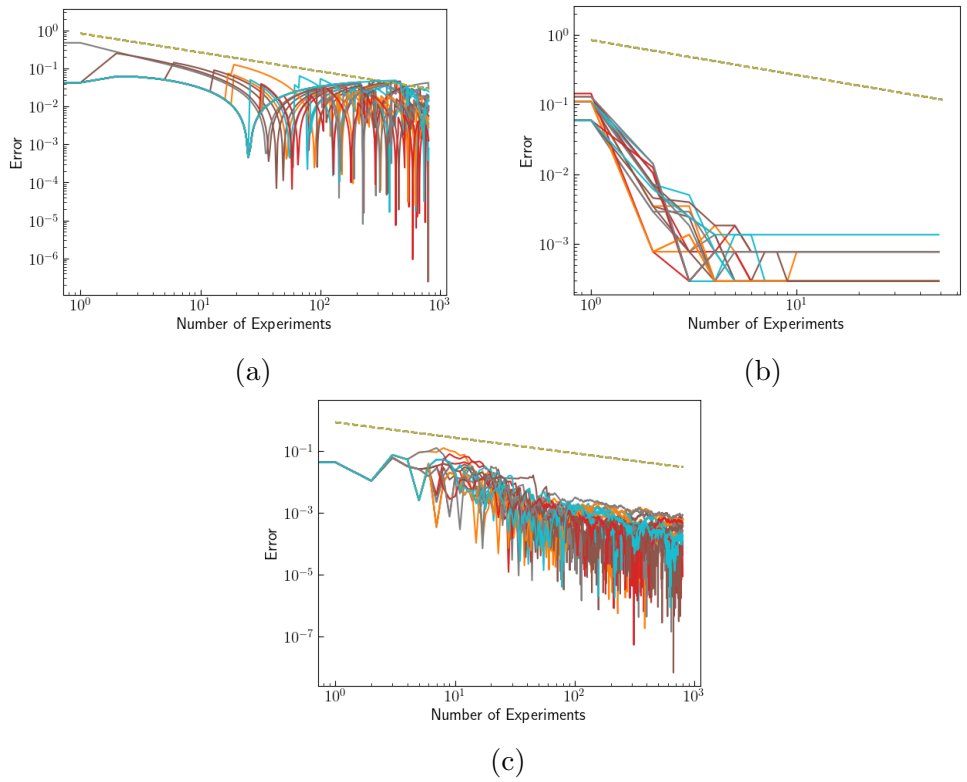
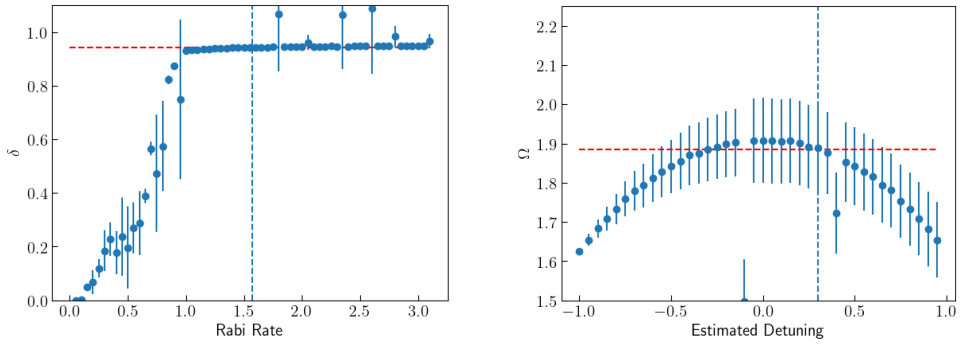


Figure 6: Demonstrations of the baseline case for estimation of δ with the Ramsey likelihood (figure 6a), an adaptively optimized case (figure 6c), and the batched case with adaptive optimization (figure 6b).

move into the full experimental regime where we begin with two unknown parameters. We investigate how error in one parameter affects the estimation of the second, finding that δ estimation with the Ramsey likelihood is sensitive to underestimates of Ω and less so to overestimates, as shown in figure 7a. We also observe that Ω estimation with the Rabi likelihood is more sensitive to δ error than the converse case, as shown in figure 7b.



(a) (b)

Figure 7: Figure 7a displays δ estimation performed at varying Ω values. The red dashed line denotes the true δ value and the blue dashed line denotes the true Ω value. Figure 7b displays Ω estimation performed at varying δ values. The blue dashed line denotes the true δ value and the red dashed line denotes the true Ω value.

Following this investigation, we develop a series of algorithms for the full two-parameter case, involving a back-and-forth estimation protocol that switches between Ω and δ estimation, estimating each parameter for a fixed number of trials, shown in figure 8a. We also demonstrate the same algorithm for a batching approach, shown in figure 8b, and develop an adaptive method that sets the iterations performed on each parameter based off the posterior standard deviation of the previous cycle, shown in figure 8c.

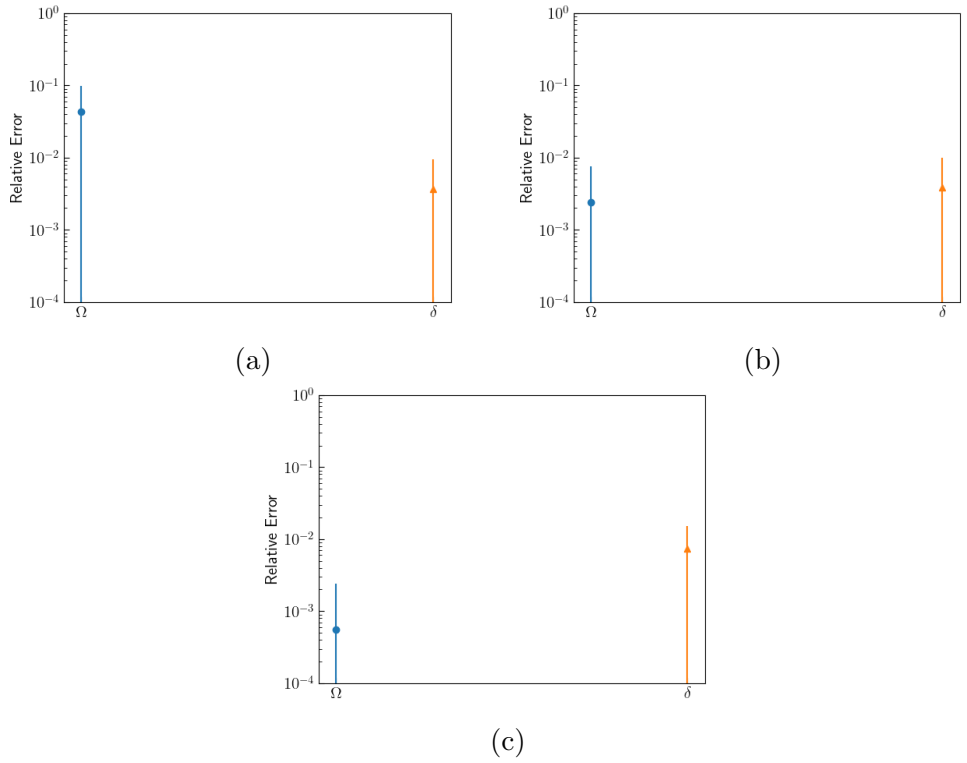


Figure 8: Figure 8a displays the mean and standard deviation in relative error for Ω and δ following 20 runs of the single cycle back-and-forth estimation. Figure 8b displays the mean and standard deviation in relative error for Ω and δ following 20 runs of the batched back-and-forth estimation. Figure 8c displays the mean and standard deviation in relative error for Ω and δ following 20 runs of the adaptive back-and-forth estimation.

However, for each of the plots shown in figure 8, we observe there is a high degree of standard deviation, with certain trials converging to values that yield high relative errors. We postulate that this effect is due to the estimator getting lost in local minima that arise during the adaptive gating and precession time algorithms. To resolve this issue, we opt to sample the number of gates performed and the precession time from a half normal distribution, enabling the algorithm to escape local minima and yielding the tightly bounded results shown in figure 9.

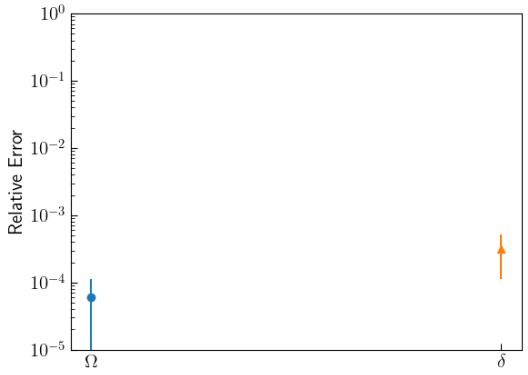


Figure 9: The mean and standard deviation of 20 estimation runs with the sampled back-and-forth algorithm.

Finally, we conclude by exploring possible extensions to this current work, including additional investigation of the effect of errors on the two unknown parameter case, improved parameter drift correction schemes, pre-computational and algorithmic optimizations, and experimental tests of the simulation results presented in this thesis.

Acknowledgments

I would first like to thank the folks at Quantinuum for giving me the opportunity to do such fun and interesting research and Dr. Bryce Bjork for setting the groundwork for the project and providing me with assistance throughout. I'd also like to thank my amazing faculty advisors Prof. Gabriel Chandler and Prof. Theresa Lynn, Prof. Chandler for offering a myriad of excellent statistical optimizations and bearing with me as we attempted to patch my admittedly lacking knowledge of statistical theory, and Prof. Lynn for helping me decipher all of the physics surrounding the project, lending me all the necessary experimental analogs that greatly helped inspire and validate new ideas, and of course, teaching the excellent quantum information course that first got me into the field.

I'd also like to thank Jeremy Parks, my manager during my summer at Quantinuum, for setting up all those meetings and helping me look for a potential senior thesis project in the field. This project likely would not have come to fruition without his commitment to helping me find ongoing or potential research I was interested in, as our search ran from experimental groups to software groups to theory groups before finally finding this project.

Finally, I'd like to thank all my faculty advisors, professors, and fellow students who have guided and supported my interest in physics and research. Prof. Joseph Osborn for giving me my first exposure to research in computer science and entertaining all of my ideas surrounding machine learning and neural networks, Prof. Janice Hudgings and Prof. Dwight Whitaker for all their advice and collaboration throughout my time in the department, and Dylan and Taylor for reading and commenting on the early versions of this thesis.

Contents

List of Figures

Chapter 1

Introduction

Quantum computing presents a regime of improvements upon tasks that are unaccomplishable or unrealistic when run on classical computers. Shor's algorithm [6], when run on a quantum computer, can theoretically decompose some integer N in polynomial in $\log N$ time, breaking RSA encryption and much of the cryptographical foundations that society is currently built upon. A myriad of other quantum algorithms promise speed-ups on classical problems [7, 8]. Quantum computers are also capable of performing accurate quantum chemistry simulations that simply cannot be achieved with classical computers [9] and further enable a wide array of cryptographical advancements [10–12]. There seem to be a multitude of possible advancements that can be achieved through the realization of a scalable quantum computer. This has ushered in an era of "quantum hype", as large corporations such as Google and IBM as well as smaller startups all crowd into the market, hoping to develop their own commercial quantum computer that can give customers the edge when it comes to the solving of complex problems. Several institutions, both companies and research laboratories, have been able to engineer quantum devices that are capable of completing tasks that no classical computer can solve within a realistic amount of time [13, 14], a concept known as quantum supremacy or quantum advantage.

However, challenging problems in the fidelity and scalability of these quantum systems hinder their current usefulness. To successfully run a process such as Shor's algorithm, one would need millions of quantum bits [15] (qubits) with application of quantum error correction. The current largest quantum computing system has a capacity of 433 qubits without a known reliable form of error correction. To realize the theoretical speedups that a quantum computer can achieve, much progress still must be made towards the achievement of higher fidelity systems and towards error correction methodologies.

In recent years, the development of quantum computing platforms and quantum information processors has seen exciting growth in a variety of directions [16–20]. Along with this advancement, improvements to error correction schemes, hardware platforms, and gating methodologies [21–27] have progressed quantum computing towards the necessary fidelities required for fault-tolerant quantum error correction, though the path to fully achieving such a goal is still long and difficult.

To achieve optimal performance in a quantum computer, one must perform calibration of the electronics and lasers that perform gates and readouts. This is typically completed by a trained operator and involves a time-consuming multi-step process of scans and measurements. Furthermore, systems can be subject to drift and measurement errors that slightly alter the calibration parameters and thus require periodic maintenance. If quantum computers seek to mature into a commercial technology, an automated and fast calibration routine that is capable of adjusting to drifts and noise that can keep systems at the optimal performance levels is highly desirable and necessary. A large portion of this calibration procedure can be distilled down to a parameter estimation problem, and to this end, a multitude of approaches for parameter estimation in a quantum system have emerged [28–32]. One such paradigm is Bayesian estimation [5, 33–38], a methodology that utilizes information from previous measurements and iterations to inform future iterations. We will present and investigate an array of methods to simplify and optimize pre-existing Bayesian estimation routines for single qubit gate calibration in a trapped-ion quantum computer.

1.1 Quantum Computing: Basic Theory

1.1.1 Qubits

Quantum computation hinges on the application of quantum mechanics to tasks of information processing and computation. Unlike classical computers, which are comprised of bits, quantum computers operate with quantum bits, or qubits [39].

A classical bit can be in the binary state 1 or 0, whereas qubits are described with the states $|0\rangle$ and $|1\rangle$. The primary advantage of qubits is that they can be in a linear combination of states (or a superposition) that can be described by:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.1}$$

where α and β are some complex coefficients that characterize the probability of the qubit $|\psi\rangle$ being in the state $|0\rangle$ or $|1\rangle$ and obey the relation $\alpha^2 + \beta^2 = 1$.

We can interpret the qubit $|\psi\rangle$'s state to be some vector within a two-dimensional complex vector space where the computational basis states, the set of states that form an orthonormal basis for the vector space, are described by $|0\rangle$ and $|1\rangle$. With this in mind, we can also express the state $|\psi\rangle$ to be

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.2}$$

$$= \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{1.3}$$

$$= \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{1.4}$$

A useful geometric representation of the two-level atom that we will sometimes use is the Bloch sphere. From the previous discussion of two-level qubit states, where we have written

$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, our coefficients α and β describe the probability amplitude of $|\psi\rangle$ being in the $|0\rangle$ state or $|1\rangle$ state. We can rewrite these probability amplitudes in terms of polar coordinates, transforming them to

$$\alpha = r_\alpha e^{i\phi_\alpha}, \tag{1.5}$$

$$\beta = r_\beta e^{i\phi_\beta}, \tag{1.6}$$

where r describes the amplitude and ϕ the angle. We can then wrap these values around the unit circle and rewrite ψ in this new representation to be

$$|\psi\rangle = r_\alpha e^{i\phi_\alpha}\,|0\rangle + r_\beta e^{i\phi_\beta}\,|1\rangle \tag{1.7}$$

$$= e^{-i\phi_\alpha}\left(r_\alpha\,|0\rangle + r_\beta e^{i(\phi_\beta-\phi_\alpha)}\,|1\rangle\right) \tag{1.8}$$

$$= r_\alpha\,|0\rangle + r_\beta e^{i(\phi_\beta-\phi_\alpha)}\,|1\rangle \tag{1.9}$$

where we have removed the global phase factor $e^{-i\phi\alpha}$. Since $\alpha^2 + \beta^2 = 1$, it follows that $r_\alpha^2 + r_\beta^2 = 1$, and we can subsequently rewrite them in terms of sines and cosines, yielding

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle \tag{1.10}$$

where θ and ϕ define the geometric components shown in figure 1.1 [40].

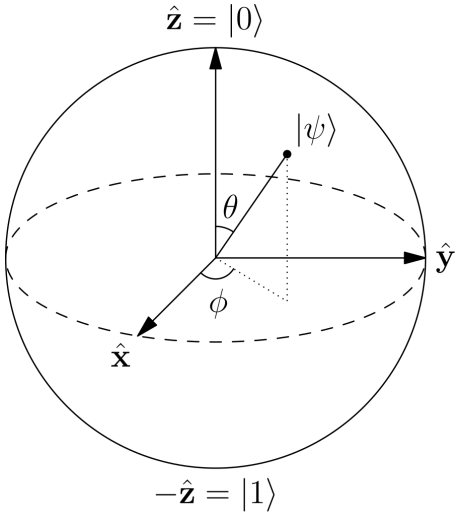


Figure 1.1: A common visualization of the Bloch sphere. The $+z$ and $-z$ axis correspond to the $|0\rangle$ and $|1\rangle$ states respectively.

1.1.2 Quantum Logic Gates

We can perform operations on these qubits through the application of logic gates, which can be defined to be any unitary transform on our initial qubit's state. It should be noted that this unitarity is the only constraint on quantum gates [39].

For example, if we seek to apply a quantum *not*, or bit flip, gate X to our state $|\psi\rangle$, we can write

$$X|\psi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{1.11}$$

$$= \begin{bmatrix} \beta \\ \alpha \end{bmatrix}. \tag{1.12}$$

Furthermore, quantum logic gates can be contained within a universal gate set. This paradigm is known as the circuit model, and states that any quantum logic gate can be decomposed into a product of smaller gates contained within the universal gate set. To construct this universal gate set, a set of local operations and two or multi-qubit entangling gates are required. This is especially useful experimentally, as complex gates can be reduced to a series of simplified, more experimentally feasible operations.

1.1.3 The Quantum Advantage

The framework of quantum computing has given rise to numerous quantum algorithms that improve the run-times of their classical counterparts, enabling solutions to tasks that were previously too computationally or time-expensive to classically compute. Shor's algorithm [6] presents a polynomial-time method for prime-factorization, which has far-reaching implications in cryptographic processes such as RSA security key generation [41]. Grover's algorithm presents a $O(\sqrt{N})$ method for searching an unordered list [42]. However, it should be noted that these algorithms cannot be executed on current quantum computers, primarily due to lack of qubit count and undesirable error rate. Irregardless, the landscape of algorithms that present a quantum speedup has continued to grow [7], drawing attention and investment into scalable and robust physical realizations of quantum computers.

1.1.4 Trapped Ion Quantum Computers

There are currently a multitude of quantum computing hardware plathforms, including superconducting qubits [43], neutral atoms [44], spin qubits [45], and others. One such scheme that has arisen is the trapped ion quantum computer (TIQC). TIQCs utilize electrodynamic confinements, such as the Paul trap displayed in figure 1.2, of which there are multiple variations [46], to hold individual ions in place, with these individual ions serving as qubits [47]. TIQCs are one of the leading hardware platforms for scalable quantum computing. The necessary components of a quantum computer, including single qubit gates [1], two qubit gates [2], state preparation and readout [48], and a scheme for scalability [4] have all been demonstrated for TIQCs at a fidelity that exceed most other hardware platforms.

Within this framework, one method for realizing distinct qubit states are through the use of ion hyperfine levels. Ions can be prepared in differing hyperfine levels of spin up or spin down, which then correspond to orthogonal basis states. Hyperfine levels arise from interactions between the spin of their nucleus and their angular momentum [49] and are an

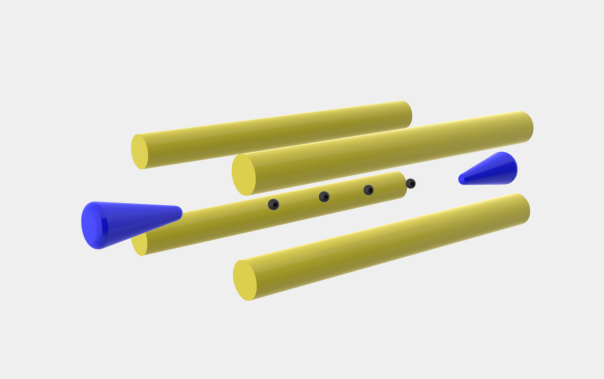


Figure 1.2: A commonly seen rod Paul ion trap (not to scale), with the four yellow cylindrical electrodes providing axial confinement of the ions and the two blue cones providing radial confinement of the ions.

attractive option for qubit states as they are fairly stable, with experimental demonstrations of coherence times exceeding 10 minutes [3]. To prepare states in a desired hyperfine level, optical pumping, a method in which photons are utilized to drive ions to a specific energy level, is used [50].

Logic gate application can then be completed through stimulated Raman transitions, transitions induced by Raman beams which are realized through the application of two laser beams with a frequency difference that is equal to the qubit's transition frequency. This effect is demonstrated in figure 1.3. This method of gate application is quite advantageous, as current technologies enable highly precise controls of beam frequencies [51]. Since beams are diffraction limited and can only be focused to a spot size on the order of the beams wavelength [52], optical-frequency beams, with wavelengths on the order of 10^2 nanometers, are the state-of-the-art for ion control, and are capable of addressing single ions.

Typically, acoustic-optic modulators (AOMs) are used to control experiments and operations. AOMs can pulse the laser and control the amplitude, phase, and frequency of light that interact with the ion through an electrical drive signal.

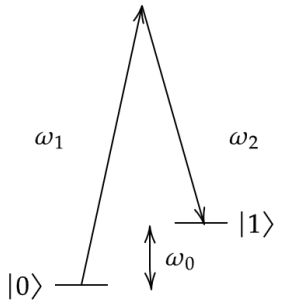


Figure 1.3: Simplified schematic of state transition through the application of two Raman pulses. The states $|0\rangle$ and $|1\rangle$ are separated by frequency ω_0, and transitions can be achieved through the coupling of two beams at frequency ω_1 and ω_2, respectively, with $\omega_1 - \omega_2 = \omega \approx \omega_0$.

Oftentimes, logic gates do not yield the exact expected output due to noise and error. Due to these effects, a common metric used to benchmark these gates is fidelity, which is defined as

$$F = |\langle \psi_g | \psi_e \rangle|^2 \tag{1.13}$$

where ψ_g denotes the goal state we expected to reach upon gate application and ψ_e the actual state we arrive at.

To measure the state of a qubit, one can project the trapped ion into a bright state or a dark state. When illuminated with a laser, a bright state will scatter a large quantity of photons (the $|1\rangle$ state in the two level system), whereas a dark state will scatter few (the $|0\rangle$ state in the two level system). From this fluorescence of the ion, the state can then be inferred.

1.1.5 Trapped Ion Quantum Computer Calibration: Single Qubit Gates

Calibration of AOMs used to drive single qubit gates is necessary to ensure high gate fidelity, where fidelity describes a qubit's ability to undergo an operation that is error free. To properly calibrate an AOM for a single qubit gate, we must estimate some parameters Ωt, where Ω defines the Rabi frequency and t defines the gate duration, and δ, the detuning of the laser. We will first focus on the parameters Ω and δ, before exploring the parameter t in further detail in section 3.2.

Ω is a function of laser power and describes the transition of ions between hyperfine levels for a given time in response to the driving laser field. To perform calibration of the Rabi rate, a typical procedure is to scan over a range of reasonable AOM input frequencies, measuring the differing outcomes that result from different inputs. A single shot of this experiment involves preparing a qubit in the ground state, sending in a pulse designed to transform the state of the qubit in a specific way, and then measuring the output state of the qubit. This

single shot is repeated multiple times and for differing frequencies, until enough data has been generated to fit a sinusoid and determine the frequency of best fit that provides the observed amount of transitions from $|0\rangle$ to $|1\rangle$ [5]. This process is demonstrated in figure 1.4.

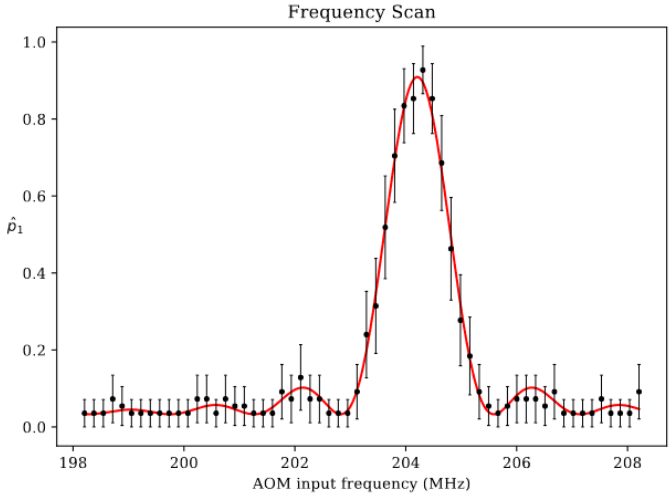


Figure 1.4: A sample calibration sinc function generated by de Neeve et al [5] with a set of simulated data. By scanning across a range of possibly frequencies, the frequency that yields the highest probability of transition from $|0\rangle$ to $|1\rangle$ ($\hat{p}_1$) can be determined. This frequency is then deduced to be the optimal frequency for gating operations.

The detuning δ describes the difference between the laser frequency and the resonant frequency of the qubit, and is defined by

$$\delta = \omega - \omega_0 \tag{1.14}$$

where $\omega = (\omega_1 - \omega_2)$ is the frequency of the Raman beam and ω_0 is the resonant frequency of the qubit. In later sections, we will simply define the Raman beam frequency to be $\omega = (\omega_1 - \omega_2)$. A similar scanning methodology known as the Ramsey scan can be applied to determine the optimal detuning.

During this calibration procedure, we continually shift the Ω and δ values until we eventually determine a set of values that yield the highest gate fidelities for our single qubit gates. However, we do not have a direct way of measuring Ω and δ, we can only take measurement of the qubit output states for the system. Parameter estimation is thus required for this problem, and an existing solution is the scanning methodology demonstrated in figure 1.4.

An additional point of difficulty is that both the Rabi rate and detuning are altered when laser frequency changes, resulting in coupling between the two parameters. The coupling of the estimation parameters adds additional complexity to the estimation procedure and can act as a source of bias.

1.2 Error

Another difficulty that may arise during calibration are the various forms of quantum error and noise that may cause parameter drift or lead to collapses in measurement. In this section, we will introduce error that arises from parameter drift and SPAM error.

1.2.1 Parameter Drift

It is well known that laser beams are imperfect, yielding variation in power and intensity due to electronic noise from the surrounding circuitry, changes in the surrounding thermal environment, and other sources of environmental noise [53][54]. This injects additional complexity into our problem, as any estimation mechanism we come up with must be capable of responding to non-stationary parameters. This also ensures that calibration is typically not a deterministic process. The optimal parameter cannot be based solely on a known physical quantity or property, rather it is altered by interactions with the surrounding environment.

1.2.2 SPAM Error

An error that can arise during the calibration process is state preparation and measurement (SPAM) error. SPAM error is the result of the non-ideality of our experimental procedures during the process of state initialization and measurement. Experimentally prepared qubits may not always start in the desired $|0\rangle$ and errors in equipment or environment during measurement may yield incorrect results (e.g. measuring a qubit in the $|1\rangle$ state when it is really in the $|0\rangle$ state). This effect is demonstrated in figure 1.5

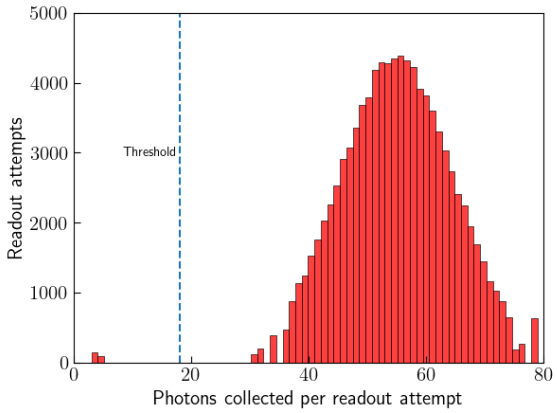


Figure 1.5: A toy graph depicting state measurement error during the measurement of a qubit that is in the $|1\rangle$ state. Though we expect to measure this qubit in the $|1\rangle$ state (readout yields photon collection numbers that are to the right of the threshold), a select handful of readout attempts yield the $|0\rangle$ state (readout yields photon collection number that are to the left of the threshold). Due to intrinsic error present in our measurement process, we may not measure them to be fluorescing at the proper level and incorrectly take them to be the $|0\rangle$ state.

SPAM error can afflict the measurement of our ions in our Rabi frequency quantum circuit, yielding a small possibility of an incorrect output when we measure the ion population following the application of the beam. This can be bothersome as it injects unpredictable randomness into our calibration routine. A robust calibration routine must be able to adapt to both of the aforementioned errors.

1.3 Overview of work

We will explore and develop methods for the automated, adaptive, and robust estimation of the Ω and δ values of an ion-trap quantum computer's single qubit gates with Bayesian methods. Chapter 2 builds the necessary foundational knowledge required to understand the dynamics of our physical system, introduces the Bayesian formalism, discusses the meaning of the Rabi frequency Ω and the detuning δ in greater detail, and explains the theory behind a simplified single parameter Rabi frequency estimation case. Chapter 3 discusses a software implementation of the baseline single estimation case and then goes on to discuss adaptive optimizations, approximations, methods to combat error such as SPAM error and laser drift, batched estimation methods, and presents a set of benchmarks for some of the improved estimation routines we have developed. Chapter 4 expands the estimation procedure to both the Rabi frequency and the detuning and discusses additional challenges, optimizations and improvements.

Chapter 2

Background

The two-level system with an energy separation of $\hbar\omega$, driven by a Raman transition at frequency omega, as shown in figure 1.3 can be modeled as an electron spin in a DC magnetic field along the z axis (time-independent) that is driven by some radio-frequency oscillating magnetic field along the x and y axis (time-dependent). With this approximation, we will derive a Hamiltonian for the system, extract the parameters we seek to estimate, and establish the Bayesian estimation framework for performing the parameter estimation procedure.

2.1 Solving the Hamiltonian

To understand the parameters we seek to estimate, we must first derive the Hamiltonian for our trapped ions pertaining to single qubit gates. We can model a trapped ion as an electron in some time-dependent magnetic field. We take this to be a two level system with some ground state $|0\rangle$ and some excited state $|1\rangle$ and define the magnetic field to be $B(t) = [B_x(t), B_y(t), B_z(t)]$. Within our model, we will approximate the energy shift of our ions as the $B_z(t)$ component of the field and the driving laser as the $B_x(t)$ and $B_y(t)$ component of the magnetic field. By the magnetic dipole approximation, the time-dependent perturbing Hamiltonian for an electron in a magnetic field can be reduced to

$$\hat{H} = -\mu \cdot B. \tag{2.1}$$

We can write the electron's dipole moment μ to be

$$\mu = \frac{\mu_B g}{2}\bar{\sigma} \tag{2.2}$$

$$= \frac{\mu_B g}{2}[\sigma_x, \sigma_y, \sigma_z], \tag{2.3}$$

where σ defines the set of Pauli spin matrices, μ_B is the Bohr magneton, and g describes the g-factor. The Pauli spin matrices are a set of Hermitian and unitary matrices that form an orthogonal basis and are defined as

$$\hat{\sigma}_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \hat{\sigma}_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \hat{\sigma}_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{2.4}$$

Rewriting the Hamiltonian in terms of these expressions, we observe

$$\hat{H} = \hbar \begin{pmatrix} \Omega_z & \Omega_{xy}^*(t) \\ \Omega_{xy}(t) & -\Omega_z \end{pmatrix}, \tag{2.5}$$

where $\Omega_z(t) = \frac{\mu_B g}{2} B_z(t)$ and $\Omega_{xy}(t) = \frac{\mu_B g}{2}(B_x(t) + iB_y(t)) = \frac{\mu_B g}{2} B_{xy}(t) e^{i\theta_B(t)}$. With this formulation, the $B_{xy}(t)$ is the time-dependent amplitude of the B-field in the x-y direction and $\theta_B(t)$ is the physical orientation of the field.

We set $\Omega_z(t)$ to be $\Omega_z = \omega_0/2$, fixing our model to reflect the time-independent energy shift of our two-level system. If we define $|B_{xy}|$ to be the magnitude of the magnetic field pointing in the orthogonal direction of z and again note θ_B to be the angle away from the x-axis in the xy plane, we can define the set of relations

$$B_x = B_{xy} \cos\theta_B, \tag{2.6}$$

$$B_y = B_{xy} \sin\theta_B. \tag{2.7}$$

We treat the driving field to be a wave of the form $\cos(\omega t + \phi)$ with a magnitude of Ω_{xy}. Applying Euler's rule and introducing driving field, we can rewrite our expression for $\Omega_{xy}(t)$ to be

$$\Omega_{xy}(t) \to \Omega_{xy} \cos(\omega t + \phi) e^{i\theta_B}. \tag{2.8}$$

We can then write our Hamiltonian in terms of 2.8, yielding

$$\hat{H} = \frac{\hbar}{2} \begin{pmatrix} \omega_0 & 2\Omega_{xy} \cos(\omega t + \phi) e^{i\theta_B} \\ 2\Omega_{xy} \cos(\omega t + \phi) e^{-i\theta_B} & -\omega_0 \end{pmatrix}. \tag{2.9}$$

This Hamiltonian cannot be solved analytically, but we can perform an approximation. Let us first re-write our Hamiltonian as an unperturbed component that contains time-independent pieces and a perturbed component that contains time-dependent pieces, or

$$\hat{H}_s = \hat{H}_0 + \hat{H}_1 \tag{2.10}$$

where we have defined

$$\hat{H}_0 = \frac{\hbar}{2} \begin{pmatrix} \omega & 0 \\ 0 & -\omega \end{pmatrix}, \tag{2.11}$$

$$\hat{H}_1 = \frac{\hbar}{2} \begin{pmatrix} -\omega + \omega_0 & 2\Omega_{xy} \cos(\omega t + \phi) e^{i\theta_B} \\ 2\Omega_{xy} \cos(\omega t + \phi) e^{-i\theta_B} & \omega - \omega_0 \end{pmatrix}. \tag{2.12}$$

We will move from the Schrodinger picture into the interaction picture and pass the time evolution due to H_0 into the operators instead of the state vectors. In the Schrodinger picture, states evolve in time in accordance to

$$|\psi_s(t)\rangle = \hat{U}_s(t) \, |\psi_s(0)\rangle \, , \tag{2.13}$$

where $\hat{U}_s(t)$ is the time-evolution operator and is equal to

$$\hat{U}_s(t) = e^{-i\hat{H}t/h} \tag{2.14}$$

if the Hamiltonian $\hat{H}$ is independent of time. The rotation generated by moving into the interaction picture transforms our Schrodinger picture state vectors $|\psi_s(t)\rangle$, yielding state vectors of the form

$$|\psi_I(t)\rangle = e^{i\hat{H}_0 t/\hbar} |\psi_s(t)\rangle . \tag{2.15}$$

Switching to the interaction picture introduces a "rotating frame" to the system, effectively unwinding a portion of the time-dependence in the state vector and introducing it to the operator. We can examine this through the transformation of our Schrodinger picture Hamiltonian $\hat{H}_s = \hat{H}_0 + \hat{H}_1$. The time dependence present in the interaction picture state vector shown in equation 2.15 is governed by the perturbing Hamiltonian $\hat{H}_1$. We can demonstrate this result by taking

$$i\hbar\frac{d}{dt}|\psi_I(t)\rangle = -\hat{H}_0 e^{i\hat{H}_0 t/\hbar}|\psi_S(t)\rangle + e^{i\hat{H}_0 t/\hbar} i\hbar\frac{d}{dt}|\psi_S(t)\rangle \tag{2.16}$$

$$= -\hat{H}_0 e^{i\hat{H}_0 t/\hbar}|\psi_S(t)\rangle + e^{i\hat{H}_0 t/\hbar}\left(\hat{H}_0 + \hat{H}_1\right)|\psi_S(t)\rangle \tag{2.17}$$

$$= e^{i\hat{H}_0 t/\hbar}\hat{H}_1|\psi_S(t)\rangle \tag{2.18}$$

$$= e^{i\hat{H}_0 t/\hbar}\hat{H}_1 e^{-i\hat{H}_0 t/\hbar}|\psi_I(t)\rangle . \tag{2.19}$$

We thus see that we can send H_1 from the Schrodinger picture to the interaction picture by taking

$$\hat{H}_{I1} = e^{i\hat{H}_0 t/\hbar}\hat{H}_1 e^{-i\hat{H}_0 t/\hbar}. \tag{2.20}$$

We can now perform the rotating wave approximation. The rotating wave approximation is a method to find an approximate analytical solution for a time dependent Schrodinger equation for a two-level system that experiences coupling with an oscillating electric field that is near-resonant to its transition frequency. We transform our Schrodinger Hamiltonian into the interaction Hamiltonian, finding

$$\hat{H}_I = -\hbar\omega_0\sigma_z/2 + e^{i\omega\sigma_z/2}\hat{H}_1 e^{-i\omega\sigma_z/2} \tag{2.21}$$

We will designate some $\hat{H_{I1}}$ where

$$\hat{H}_{I1} = e^{i\omega\sigma_z/2}\hat{H}_1 e^{-i\omega\sigma_z/2}. \tag{2.22}$$

and examine this term more explicitly. We also compute the exponential map for the $e^{i\omega\sigma_z}$

and $e^{-i\omega\sigma_z}$ terms, finding that

$$e^{i\omega\sigma_z} = \sum_{n=0}^{\infty} \frac{(i\omega)^2\sigma^n}{n!}\sigma_z^n \tag{2.23}$$

$$= I\cos(\omega) + i\sigma_z \sin(\omega) \tag{2.24}$$

$$= \begin{pmatrix} \cos(\omega) + i\sin(\omega) & 0 \\ 0 & \cos(\omega) - i\sin(\omega) \end{pmatrix} \tag{2.25}$$

$$= \begin{pmatrix} e^{i\omega} & 0 \\ 0 & e^{-i\omega} \end{pmatrix} \tag{2.26}$$

and

$$e^{-i\omega\sigma_z} = \begin{pmatrix} e^{-i\omega} & 0 \\ 0 & e^{i\omega} \end{pmatrix}. \tag{2.27}$$

We can then expand the matrix terms, finding that

$$\hat{H_{I1}} = \frac{\hbar}{2}\Omega_{xy} \begin{pmatrix} e^{i\omega} & 0 \\ 0 & e^{-i\omega} \end{pmatrix} \tag{2.28}$$

$$\begin{pmatrix} 0 & e^{i\theta_B} 2\cos(\omega t + \phi) \\ e^{-i\theta_B} 2\cos(\omega t + \phi) & 0 \end{pmatrix} \begin{pmatrix} e^{-i\omega} & 0 \\ 0 & e^{i\omega} \end{pmatrix}$$

$$= \frac{\hbar}{2}\Omega_{xy} \begin{pmatrix} 0 & \frac{e^{i\phi}e^{i\phi_B}e^{2i\omega t}}{2} + \frac{e^{-i\phi}e^{i\theta_B}}{2} \\ \frac{\Omega e^{i\phi}e^{-i\theta_B}}{2} + \frac{\Omega e^{-i\phi}e^{-i\phi_B}e^{-2i\omega t}}{2} & 0 \end{pmatrix} \tag{2.29}$$

$$= \frac{\hbar}{2} \begin{pmatrix} 0 & \Omega_{xy}e^{i(\phi-\theta_B)} \\ \Omega_{xy}e^{-i(\phi-\theta_B)} & 0 \end{pmatrix} \tag{2.30}$$

where in equation 2.29 we have converted the Cosine function to exponentials and in equation 2.30 we have neglected the fast oscillating exponential term $e^{-2i(\omega t+\phi)}$ as

$$\int e^{\pm in\theta} = \frac{e^{\pm in\theta}}{\pm in} \approx 0. \tag{2.31}$$

Upon returning this result to the interaction Hamiltonian shown in equation 2.21, we have

$$\hat{H_I} = -\hbar\omega\sigma_z/2 + e^{i\omega\sigma_z/2}\hat{H}_1 e^{-i\omega\sigma_z/2} \tag{2.32}$$

$$= \frac{\hbar}{2} \begin{pmatrix} -\delta & \Omega_{xy}e^{i\tilde{\phi}} \\ \Omega_{xy}e^{-i\tilde{\phi}} & \delta \end{pmatrix}, \tag{2.33}$$

where $\delta = \omega - \omega_0$ and $\tilde{\phi} \equiv \phi - \theta_B$. Upon converting the exponentials to trignometric functions, we find that the propagator for this Hamiltonian is

$$\hat{U(t)} = \begin{bmatrix} \cos\left(\frac{\alpha t}{2}\right) + \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & -\frac{i\Omega e^{i\tilde{\phi}}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \\ -\frac{i\Omega e^{-i\tilde{\phi}}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & \cos\left(\frac{\alpha t}{2}\right) - \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \end{bmatrix}, \tag{2.34}$$

where $\alpha^2 \equiv \Omega_{xy}^2 + \delta^2$. Going forwards, we will refer to Ω_{xy} as Ω.

We can then extract the probability of transitioning from ground to excited state as

$$P(|1\rangle) = |\langle 1| \hat{U} |0\rangle|^2 \tag{2.35}$$

$$= \frac{\Omega^2 \sin^2\left(\frac{\alpha t}{2}\right)}{\alpha^2} \tag{2.36}$$

and the probability of remaining in the ground state is conversely

$$P(|0\rangle) = 1 - \frac{\Omega^2 \sin^2\left(\frac{\alpha t}{2}\right)}{\alpha^2}. \tag{2.37}$$

We have obtained a sampling function that we can apply to our system to determine if measurement will return a qubit in the $|0\rangle$ or $|1\rangle$ state in terms of our estimation parameters Ω and δ that we can later apply in our Bayesian estimation procedure.

2.2 Rabi Frequency and Detuning Conversions

We will examine the meaning of the Rabi frequency Ω and the detuning δ in greater detail, beginning with a discussion of the Rabi frequency.

When the driving field is resonant with the trapped ion's frequency and the ion is in the ground state, then it is brought from the ground state to the first excited state. However, if the ion is in an excited state, stimulated emission can occur, and the ion can emit a photon at the same phase and frequency of the driving field and return to the ground state. Due to these excitations and de-excitations, the atom will oscillate back and forth between ground and excited states, cyclically absorbing photons and re-emitting them. This phenomena is known as Rabi flopping, and can be pictured in figure 2.1. The period of these oscillations is known as the Rabi frequency Ω. [55]

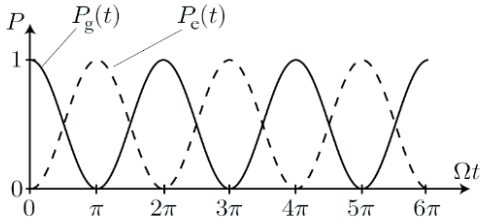


Figure 2.1: Figure taken from Quantum and Atom Optics, by Daniel A. Steck. Probability for finding the atom in the ground and excited state over time. The atom oscillates between ground and excited state at the Rabi frequency [55].

The detuning $\delta = \omega - \omega_0$ represents the difference in frequency between the driving laser and the resonant frequency of the atom, where ω is the driving frequency of the laser and ω_0

is the resonant frequency of the atom. If there exists some nonzero detuning in the system, then the driving laser will never be able to drive the atom from the ground state to the excited state with probability 1. In the case of the calibration of our trapped-ion quantum computer, we desire to minimize the detuning in our system.

For our parameter estimation, we will develop a set of unitless representations of Ω and δ. This will enable notational consistency and a convenient way to represent our initial priors in our Bayesian estimation procedure, which will be discussed in a later section.

In the laboratory, we can bound the Rabi frequency by the laser drive time, t_π, that is required for a qubit to complete one single π rotation about the Bloch sphere. We do not want to overshoot this time and exceed a π rotation, and thus we will pick some time $\tau < t_\pi$ that we will actually drive our laser for. With this in mind, we can designate a new parameter Ω^*, where

$$\Omega^* = \tau\Omega. \tag{2.38}$$

For example, if we know that a time of 10μs will result in a π pulse, we can instead pick some $\tau = 7\mu s$ to drive our laser at, thus avoiding a pulse that will exceed a π pulse and cause gating issues. Upon estimating Ω^*, if we then wish to compute the true Rabi frequency Ω, we can simply recover Ω by taking $\frac{\Omega^*}{\tau}$.

We will do the same for the detuning δ. We can bound δ within a known frequency and subsequently pick a maximal laser drive time such that we do not accumulate excessive phase during experiment. We can then perform a similar procedure and designate δ^* where

$$\delta^* = \tau'\delta. \tag{2.39}$$

We now have two parameters Ω^* and δ^* that will serve as unitless substitutes for our estimation parameters.

2.3 Bayesian Estimation for SQ Gate Calibration

This section will discuss the basic theory behind Bayesian estimation and its use in single qubit (SQ) gate calibration.

2.3.1 Bayesian Estimation Mathematics

Bayesian estimation is the process of applying Bayes' rule to determine some output probability distribution that describes the uncertainty of a parameter or a set of parameters that define a system [56]. In this instance, we seek to apply Bayesian estimation to determine the calibration parameters our laser system is set to. To obtain this result, we follow the following procedure:

1. Determine some prior distribution that will characterize the uncertainty in the parameters of the system that is consistent with pre-existing knowledge of the system. Initially, if there is no pre-existing knowledge on the system, a uniform distribution is used as the first prior.

2. Observe a random variable X from the distribution f from the system through measurement.

3. Identify the likelihood $L(\Theta|x_i) := \Pi_{i=1}^{n} f(x_i|\bar{\Theta})$, which describes the probability of observing specific data given a set of system parameters Θ.

4. Utilizing the likelihood and the prior, determine the conditional probability distribution that can best model the uncertainty in the parameters.

5. Iterate over this process.

In this instance, we want to estimate some set of experimental parameters Θ_{opt} given the measurement results we see and the past information we have obtained. At each iteration, we perform the Bayesian estimation uprocess and obtain a prediction of the calibration parameter, denoted by $\hat{\Theta}$. As iterations increase, the information from past trials (evidence) is accumulated and more observed data is obtained, yielding newly updated $\hat{\Theta}$ values that in theory converge to Θ_{opt}

Bayes' rule [57] describes a procedure to obtain some posterior $P(\Theta|m)$, a conditional probability distribution for obtaining some Θ given some set of measurements m. This procedure requires some prior distribution $P(\Theta)$, a probability of making some measurement $P(m)$, and a likelihood distribution that the measurement outcome will be yielded $P(m|\Theta)$. Combining these inputs, we can obtain the posterior output $P(\Theta|m)$.

The prior and posterior distributions are probability density functions (PDF). A PDF is defined to be a continuous function that describes the probability that a random variable X will be equal to a specific value within the sample space of Y, or

$$Pr(a \leq X \leq b) = \int_a^b f_X(x)dx. \tag{2.40}$$

A PDF must also satisfy two conditions:

1. $f(x) \geq 0 \quad \forall X \in Y$.

2. Integrating over $f(x) \ \forall X \in Y$ must yield 1.

The chief difference between a likelihood and a probability is the parameters each distribution is conditioned on. A typical probability $P(m|\Theta)$ will hold some parameter Θ fixed and yield varied sets of our measured data m. $P(m|\Theta)$ defines the probability we observe some set of outcomes m given the parameter Θ. However, in real-world cases, as well in this thesis, the parameter Θ is not known and must be estimated upon observation of data. To obtain an estimate of Θ, we maximize the likelihood, though in a discrete space this is equivalent to choosing Θ such that the probability of observing the data m is maximized. To perform this process, we write some likelihood function $L(\Theta|m) = P(m|\Theta)$ for a range of Θ values that is then maximized. The method of parameter estimation through likelihood maximization is termed Maximum Likelihood Estimation (MLE), and will be further discussed in later sections.

With this in mind, we can write Bayes' rule to be

$$P(\Theta|m) = \frac{L(\Theta|m)P(\Theta)}{P(m)} \tag{2.41}$$

which can be further simplified to

$$P(\Theta|m) \propto L(\Theta|m)P(\Theta) \tag{2.42}$$

where $P(m)$ is a normalization factor found in the denominator of 2.41 that keeps the posterior as a valid probability distribution.

After each iteration, our posterior $P(\Theta|m)$ is then fed back into another round of Bayesian estimation as the prior distribution. Throughout each successive cycle, the standard deviation σ ought to decrease until we eventually pick a stopping point upon reaching an acceptable σ value and converging to the correct value of Θ. We can describe an iterative process of n cycles of Bayesian estimation to be:

$$P(\Theta|m_1, m_2, \cdots, m_n) \propto P(\Theta) \prod_i^n L(\Theta|m_i) \tag{2.43}$$

where we take all measurement outcomes m_i to be independently and identically distributed random outcomes. After each iteration, we observe some decrease in uncertainty modeled by resultant posterior PDF. This can be visually observed as a narrowing of the posterior PDF. This effect can be observed in figure 2.2.

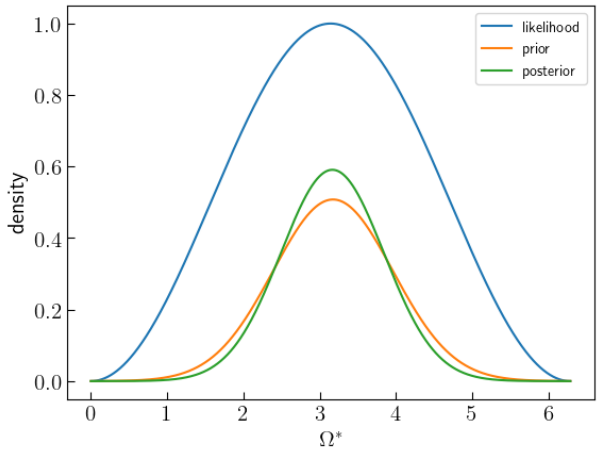


Figure 2.2: A cycle of Bayesian estimation for the determination some parameter Ω^*. Note that the posterior, the product of the likelihood and the prior, is more narrow than the prior, representing an increase in information on the true value of Ω^*.

2.3.2 Parameter Estimation in the Calibration Process

Calibration of the single qubit gates for a quantum computer, as described in section 1.1.5, is concerned with adjusting the amplitude of the driving laser until the desired Rabi frequency and detuning value are found, yielding an optimal gate fidelity. During this procedure, repeated parameter estimation is required for each change in amplitude, as the Rabi frequency and detuning value cannot be directly measured. The Bayesian estimation method will take as inputs the measurements we take of ion states and the past beliefs we have about the system and output predictions on the values of the unknown Ω and δ values.

2.4 Single Parameter Analysis

To better understand the process of Bayesian estimation for our specific calibration case, let us examine a simplified single parameter calibration process, where we seek to determine the optimal Rabi frequency Ω from our system by reading values for some quantum circuit.

Prior to performing the unitless conversion, Bayes' rule for this problem can be denoted by

$$P(\Omega|m) = \frac{L(\Omega|m)P(\Omega)}{P(m)} \tag{2.44}$$

where m denotes the state we measure the qubit to be in. To perform Bayesian estimation, we must find some general likelihood $L(\Omega|m)$, some prior $P(\Omega)$, and some normalization factor $P(m)$.

2.4.1 The Likelihood Function

We will start with the likelihood for this system. Recall that the coupled parameter probability of finding the qubit in the excited $|1\rangle$ or ground $|0\rangle$ state is defined as

$$P(|1\rangle) = \frac{\Omega^2 \sin^2\left(\frac{\alpha t}{2}\right)}{\alpha^2}, \tag{2.45}$$

$$P(|0\rangle) = 1 - \frac{\Omega^2 \sin^2\left(\frac{\alpha t}{2}\right)}{\alpha^2}. \tag{2.46}$$

We will examine a simplified Bayesian estimation workflow that focuses on simply estimating the Rabi frequency Ω and we will assume δ, our detuning, to be known and at the optimal value of $\delta = 0$. $\alpha = \sqrt{\Omega^2 + \delta^2}$, and thus when $\delta = 0$, our probabilities become

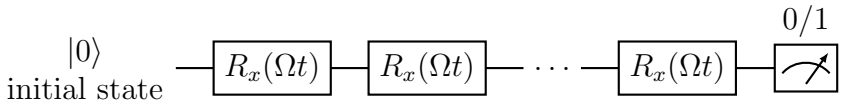
$$P(|1\rangle) = \sin^2\left(\frac{\Omega t}{2}\right), \tag{2.47}$$

$$P(|0\rangle) = \cos^2\left(\frac{\Omega t}{2}\right). \tag{2.48}$$

We can introduce the measurement m, where $m \in \{0, 1\}$, of the qubit state following the application of the driving laser, thus yielding the likelihood:

$$L(\Omega) = \cos^2\left(\frac{\Omega t}{2} - \frac{m\pi}{2}\right)^m \sin^2\left(\frac{\Omega t}{2} - \frac{m\pi}{2}\right)^{1-m}. \tag{2.49}$$

Measurement of the Rabi frequency for a given experimental set up can be completed through the following quantum readout circuit



where $R_x(\Omega t)$ defines a rotation by an angle Ωt about the x axis of the Bloch sphere.

We introduce an additional parameter $k \in \{0, 1, 2, ...\}$, a unitless value which denotes the number of R_x gates that are applied prior to measurement. Note that an increased k value serves to amplify errors in the Rabi frequency, as repeated applications of R_x gates with an erroneously estimated Rabi frequency that results in incorrect transition probabilities will stack and yield a greater degree of error upon final measurement. Taking note of the fact that the probability of measuring $|1\rangle$ or $|0\rangle$ is merely a $\frac{\pi}{2}$ phase shift that is dependent on the value of m and applying the unitless conversion of Ω to Ω^* that we have described in section 2.2, we can write the generalized likelihood of measuring an ion in the $|0\rangle$ or $|1\rangle$ state to be

$$L(\Omega^*|m) = \cos^2\left(\frac{k\Omega^*}{2} - \frac{m\pi}{2}\right). \tag{2.50}$$

k essentially encapsulates the parameter t, as additional applications of R_x gates can be thought of as driving the laser for an additional fixed time increment. This lets us absorb our time t into k in equation 2.50, and alongside the unitless Ω^* value, we have an entirely unitless likelihood.

2.4.2 Prior and Posterior Distributions

Moving onto the prior and posterior distribution, we can represent the prior probability-density function $P(\Omega^*)$ to be the univariate normal distribution $f_{\mu,\sigma}(x)$:

$$f_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{2.51}$$

where μ and σ denote the mean and standard deviation of our distribution, respectively.

In this case, we see that Ω^* values in the likelihood function $P(m|\Omega^*)$ that differ by integer multiples of 2π are equivalent, and thus replacing the univariate normal distribution with a variant that is simply wrapped around the interval $[0, 2\pi)$ is an advantageous simplification. Furthermore, the univariate normal distribution can cause some problems if it contains values

that leak out of the $[0, 2\pi]$ range, as these values tend to be nonsensical in terms of the Rabi frequency. We can define this wrapped normal distribution $f^{\circ}_{\mu,\sigma}(\Omega^*)$ to be [58]

$$f^{\circ}_{\mu,\sigma}(\Omega^*) = \sum_{n=-\infty}^{\infty} f^{\circ}_{\mu,\sigma}(\Omega^* + 2\pi n). \tag{2.52}$$

Thus we want to find some approximate posterior distribution $f^{\circ}(\Omega^*; \mu_1, \sigma_1)$ given a prior distribution $f^{\circ}(\Omega^*; \mu_0, \sigma_0)$ and a likelihood function $L(\Omega^*|m)$ generated from a measurement.

Our posterior distribution, $P(\Omega^*|m)$ is therefore:

$$P(\Omega|m) = \frac{L(\Omega^*|m) \cdot f^{\circ}(\Omega^*; \mu_0, \sigma_0)}{P(m)} \tag{2.53}$$

$$= \frac{1}{N} L(\Omega^*|m) \cdot f^{\circ}(\Omega^*; \mu_0, \sigma_0) \tag{2.54}$$

where $m \in \{0, 1\}$ is the result of a measurement and we take $N \equiv P(m)$ as a normalization constant to be determined later. However, our posterior distribution will not be in the form of a wrapped normal distribution, and thus we must find some method to fit it onto one. A simple way is to calculate the mean, μ, and variance, σ^2, of this distribution and generate a best-fit wrapped normal distribution with those parameters. For this, we use the relations [59]:

$$\mu = \operatorname{atan} 2(\operatorname{Im}(X), \operatorname{Re}(X)), \tag{2.55}$$

$$\sigma^2 = -\ln\big(|X|^2\big). \tag{2.56}$$

where X defines the first circular moment of the wrapped normal distribution in terms of the circular variable $e^{i\Omega^*}$. X can be denoted as:

$$X = \int_0^{2\Omega^*} e^{i\Omega^*} P(\Omega^*|m) d\Omega^* \tag{2.57}$$

$$= \frac{1}{N} \int_0^{2\pi} e^{i\Omega^*} L(m|\Omega^*) f^{\circ}(\Omega^*; \mu_0, \sigma_0) d\Omega^*. \tag{2.58}$$

Now, what remains is for us to calculate X. However, analytical calculation of this integral is cumbersome, and thus we choose to simplify it by representing part of it as a Fourier series. We write $P(m|\Omega^*)$ to be:

$$P(m|\Omega^*) = \sum_{n=-\infty}^{\infty} c_n e^{in\Omega^*} \tag{2.59}$$

where c_n are the Fourier coefficients.

To solve for the coefficients, we integrate the Fourier series expression of $P(m|\Omega^*)$ against $e^{-in\Omega^*}$ [60].

$$\int_0^{2\pi} e^{-in'\Omega^*} L(m|\Omega^*) d\Omega^* = \int_0^{2\pi} e^{-in'\Omega^*} \sum_{n=-\infty}^{\infty} c_n e^{in\Omega^*} d\Omega^* \tag{2.60}$$

$$= \sum_{n=-\infty}^{\infty} \int_0^{2\pi} c_n e^{i(n-n')\Omega^*} d\Omega^* \tag{2.61}$$

$$= \sum_{n=-\infty}^{\infty} c_n 2\pi \delta_{n,n'} \tag{2.62}$$

$$= 2\pi c_{n'}. \tag{2.63}$$

We see that we can solve for our coefficients c_n by performing the same integration against our likelihood function, $L(\Omega^*|m) = \cos^2\left(\frac{k\Omega^*}{2} - \frac{m\pi}{2}\right)$, which yields:

$$c_n = \frac{1}{2\pi} \int_0^{2\pi} e^{-in\Omega^*} \cos^2\left(\frac{k\Omega^*}{2} - \frac{m\pi}{2}\right) d\Omega^* \tag{2.64}$$

$$= \frac{1}{2\pi} \times \begin{cases} 0.5 & n = 0 \\ \frac{1}{4} e^{i\pi m} & \text{if } n = k \\ \frac{1}{4} e^{-i\pi m} & \text{if } n = -k \\ 0 & \text{otherwise} \end{cases} \tag{2.65}$$

$$= \frac{1}{2\pi} \times \begin{cases} 0.5 & \text{if } n = 0 \\ \frac{1}{4}(-1)^m & \text{if } n = \pm k \\ 0 & \text{otherwise}\,. \end{cases} \tag{2.66}$$

Returning this expression for $L(\Omega^*|m)$ into our equation for X, we find:

$$X = \frac{1}{N} \sum_{n=-\infty}^{\infty} c_n \int_0^{2\pi} e^{i(n+1)\Omega^*} f^{\circ}(\Omega^*; \mu_0, \sigma_0) d\Omega^*. \tag{2.67}$$

The expression $\int_0^{2\pi} e^{i(n+1)\Omega^*} f^{\circ}(\Omega^*; \mu_0, \sigma_0)$ is essentially the evaluation of the characteristic function for the variable X, and thus we can apply the following simplification for the characteristic function of a wrapped normal distribution [61]:

$$\int_0^{2\pi} e^{i(n+1)\Omega^*} f^{\circ}(\Omega^*; \mu_0, \sigma_0) = e^{i\mu_0(n+1)} e^{-(\sigma_0(n+1))^2/2}. \tag{2.68}$$

Returning this to our equation for X, we have:

$$X = \frac{1}{N} \sum_{n=-\infty}^{\infty} c_n e^{i\mu_0(n+1)} e^{-\sigma_0^2(n+1)^2/2}. \tag{2.69}$$

Since valid probability distributions must integrate to 1, we can then evaluate the normalization constant by enforcing the condition:

$$\frac{1}{N}\int_0^{2\pi} L(\Omega^*|m)f^\circ(\Omega^*;\mu_0,\sigma_0)d\Omega^* = \int_0^{2\pi} P(\Omega^*|m)d\Omega^* = 1. \tag{2.70}$$

This yields the following for N:

$$N = \int_0^{2\pi} L(\Omega^*|m)f^\circ(\Omega^*;\mu_0,\sigma_0)d\Omega^* \tag{2.71}$$

$$= \sum_{n=-\infty}^{\infty} c_n \int_0^{2\pi} e^{in\Omega^*} f^\circ(\Omega^*;\mu_0,\sigma_0)d\Omega^* \tag{2.72}$$

$$= \sum_{n=-\infty}^{\infty} c_n e^{in\mu_0} e^{-n^2\sigma_0^2/2}. \tag{2.73}$$

and we see that we now have a full set of equations to describe the output posterior probability distribution in terms of some input prior probability distribution, some likelihood, and a normalization factor.

2.5 Summary

Recall, from section 2.3.1 that to perform parameter estimation with Bayesian methods, we must apply Bayes rule. For the estimation of the unitless Rabi frequency Ω^*, this will be of the form

$$P(\Omega^*|m) = \frac{L(\Omega^*|m)P(\Omega^*)}{P(m)} \tag{2.74}$$

Bayes' rule requires some likelihood $L(m|\Omega^*)$, some prior probability distribution $P(\Omega^*)$ and some normalizing factor $P(m)$. In this section we have derived each of the above components analytically, finding that

$$L(m|\Omega^*) = \cos^2\left(\frac{k\Omega^*}{2} - \frac{m\pi}{2}\right), \tag{2.75}$$

$$P(\Omega^*) = f^\circ(\Omega^*;\mu_0,\sigma_0), \tag{2.76}$$

$$P(m) = \sum_{n=-\infty}^{\infty} c_n e^{in\mu_0} e^{-n^2\sigma_0^2/2}, \tag{2.77}$$

where μ_0 and σ_0 denote some prior mean and standard deviation, k describes the amount of rotation gates performed in the circuit prior to measurement, m denotes the measurement outcome, Ω^* describes the unitless Rabi frequency, and c_n denotes the set of Fourier

coefficients that are of the form

$$c_n = \begin{cases} 0.5 & \text{if } n = 0 \\ \frac{1}{4}(-1)^m & \text{if } n = \pm k \\ 0 & \text{otherwise}\,. \end{cases} \tag{2.78}$$

With these components, we can now perform Bayesian estimation to determine the unitless Rabi frequency Ω^* for our system given a set of measured data.

Chapter 3

Simplified Single Qubit Gate Calibration: Rabi Rate

We will first explore a simplified case of the single gate estimation process, which will only involve estimation of the unitless Rabi frequency. For convenience, we will refer to as the Rabi frequency Ω in the later sections.

3.1 Baseline Software Implementation

We first establish a software implementation of the baseline Rabi frequency estimation case, where we are only seeking to estimate the Rabi frequency for our gate, rather than the full two-parameter model. We accomplish this in the Python programming language by modeling the equations described in section 2.4.2. We employ an iterative approaching, reusing our posterior mean and posterior standard deviation as the prior mean and prior standard deviation for the next iteration. The mathematical modelings for a single iteration of the Bayesian algorithm are defined by the following:

```
def bayesian_update(prior_mu, prior_sigma, k, m):
    C0 =     0.5         / (2*np.pi)
    Cplusk = 0.25*(-1)**m / (2*np.pi)
    Cminusk = 0.25*(-1)**m / (2*np.pi)

    XtimesN = C0    * np.exp(1j*prior_mu)     * np.exp(-prior_sigma**2/2)      +
        \
              Cplusk * np.exp(1j*(k+1)*prior_mu) *
                 np.exp(-(k+1)**2*prior_sigma**2/2) + \
              Cminusk * np.exp(1j*(-k+1)*prior_mu) *
                 np.exp(-(-k+1)**2*prior_sigma**2/2)

    N =      C0                                                          + \
              Cplusk * np.exp(1j*k*prior_mu) * np.exp(-k**2*prior_sigma**2/2) + \
```

```
            Cminusk * np.exp(1j*(-k)*prior_mu) * np.exp(-k**2*prior_sigma**2/2)

    Xreal = np.real(XtimesN/N)
    Ximag = np.imag(XtimesN/N)

    posterior_mu = np.mod(np.arctan2(Ximag, Xreal), np.pi)

    absXsquare = (Xreal*Xreal + Ximag*Ximag)
    variance = -np.log(absXsquare)
    posterior_sigma = np.sqrt(variance)

    if (posterior_sigma > 20):
        posterior_sigma = 20
    if (posterior_sigma < 1e-6):
        posterior_sigma = 1e-6
    if (posterior_sigma/prior_sigma < 0.1):
        posterior_sigma = 0.1*prior_sigma

    return posterior_mu, posterior_sigma
```

where we have bound our posterior sigma values to ensure numerical stability. This method takes as input k, the number of rotational x gates we seek to perform prior to measurement, and m, the result from measuring the state of the ion. We will expand on the significance of k in section 3.2. For the sake of the baseline case, we will keep k fixed at 1.

During initialization of our experiment, we will pick an estimated initial prior μ and σ value and take a measurement of our ion. These values will then be fed into `bayesian_update`, which will yield a posterior μ and σ value for use in our next shot.

For the full multi-shot experiment, we apply the following procedure

Algorithm 1 Baseline Multi-shot Bayesian Inference

$\mu \leftarrow$ initial mean
$\sigma \leftarrow$ initial standard deviation
$k \leftarrow 1$
for i in number of experimental shots **do**
  $m \leftarrow$ Ion state measurement
  $\mu_i, \sigma_i \leftarrow$ `bayesian_update`$(\mu_{i-1}, \sigma_{i-1}, m, k)$
end for

where we store the resultant μ and σ of each trial for later analysis.

After each shot, we compute the relative error of each shot, defined by:

$$\text{Err} = \frac{|\mu - \Omega|}{\mu} \tag{3.1}$$

where μ defines the posterior mean following a shot and Ω defines the true Rabi frequency value. It should be noted that during an actual estimation, we are unable to compute the relative error as we do not have access to the true Ω value. However, for testing purposes this metric is highly illustrative and thus we utilize it throughout this thesis.

Applying our approach to a 10,000 shot experiment yields the changes in error and posterior standard deviation shown in figure 3.7 where we observe that the posterior standard

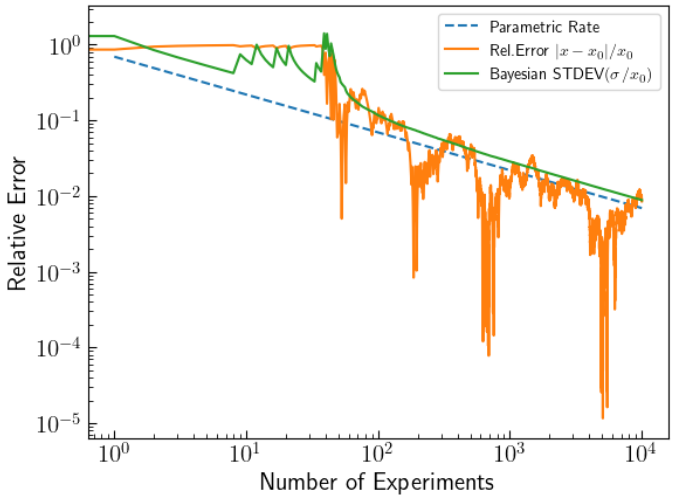


Figure 3.1: Baseline Bayesian estimation algorithm demonstrated over 10,000 experimental shots. The green line denotes the Bayesian standard deviation σ over each shot while the orange line denotes the relative error. The dashed blue line denotes the parametric rate of convergence.

deviation decreases at a similar rate to the parametric rate of convergence. The parametric rate of convergence is a limit on the rate of estimation error for independent and identically distributed (IID) samples in a parametric problems and is defined to be $n^{-1/2}$. From figure 3.2, where we run 20 trials of the baseline algorithm, we observe that the our baseline estimation consistently converges.

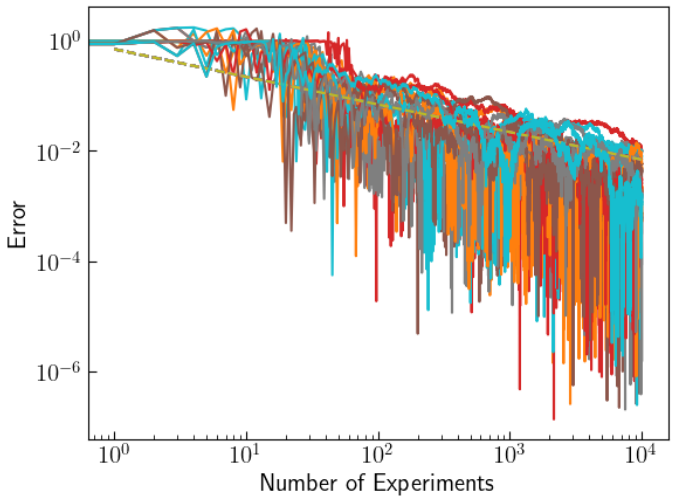


Figure 3.2: 20 runs of the baseline estimation algorithm.

We have thus developed a theoretically optimal model for the estimation of the Rabi frequency Ω.

3.1.1 Initial Flatness

In all the estimation runs, we see that in the initial set of shots spanning from $0 - 40$, we observe very little learning and the relative error stays flat. This effect results from poor distribution fitting in the initial shots of the experiment. In the initial trials, σ is high as we have little data and thus great uncertainty in our estimation. During this period, the wrapped normal distribution is a poor approximation for our posterior distribution, and thus our application of Bayes' rule in these early trials is not entirely effective. One way to remedy this is with the Fourier method proposed by van den Berg [62]. However, this effect is not catastrophic, as repeated measurements and applications of the likelihood function still enable our model to learn after enough data has been taken, and the wrapped normal eventually is able to properly model the data and yield convergent results. We will explore this defect further in section 3.5.

3.2 Adaptive Optimization

In typical Bayesian inference, the likelihood function stays fixed as the amount of shots increases. This is because samples are observed from the same system and thus characterized by the same likelihood throughout the experimental process. However, in our experimental case, we can adjust the amount of rotational gates we perform in-between experimental shots (described in section 2.4.1), as we have physical access to the system. This allows us to perform adaptive optimization of our likelihood over the course of the experiment and beat the parametric rate of convergence. We begin estimation with the likelihood displayed

in figure 3.3 where $k = 1$.

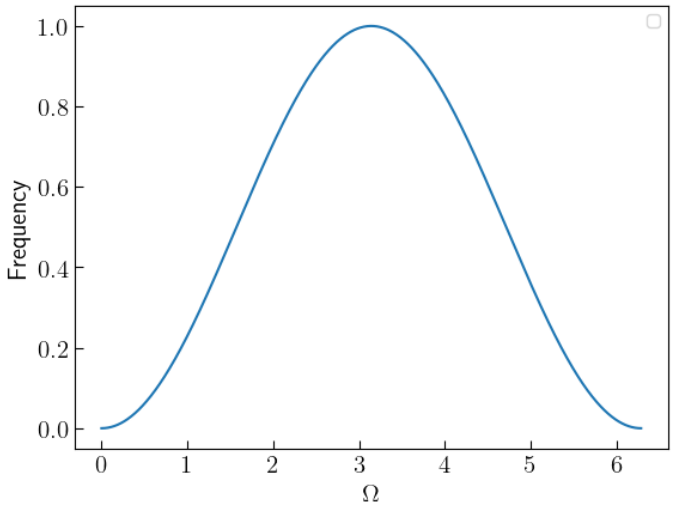


Figure 3.3: Likelihood function $L(\Omega|m) = \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)$ plotted over Ω values for $k = 1$

However, as we gain greater confidence in our Ω value, we can increase k, the number of rotational x gates we apply prior to measurement, yielding the effect shown in figure 3.4

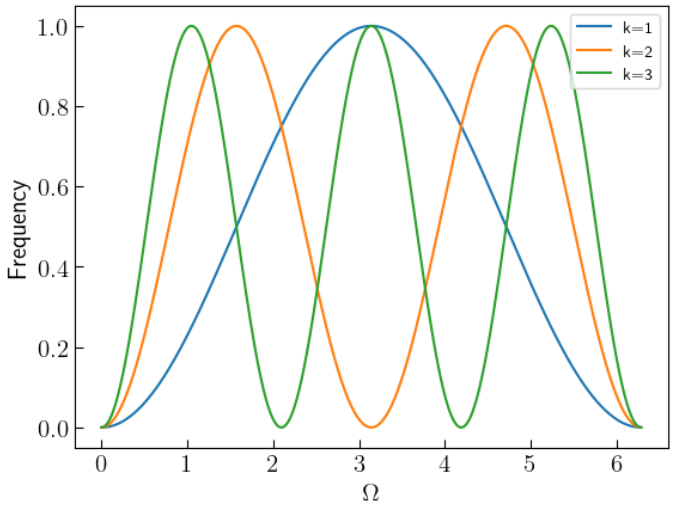


Figure 3.4: Likelihood function $L(m|\Omega) = \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)$ plotted over Ω for $k = [1, 3]$

We observe a narrower likelihood function as we increase k, though this comes at the cost of multi-modality. The multi-modality of the likelihood function can be dangerous, as our estimated Ω value can be captured in the incorrect mode and essentially become stuck in a local minima. This is a phenomena we will refer to as fringe or mode slipping. However, this can be avoided by setting k to be a function of σ. When we reach a desirable uncertainty,

we can safely narrow the likelihood while ensuring we do not trap our Ω estimation in the improper mode.

To determine k as a function of the standard deviation, we first examine the posterior uncertainty for a wrapped normal distribution, which is defined to be

$$\sigma = \sqrt{\ln\left(\frac{1}{X^2}\right)} \tag{3.2}$$

where we have defined X^2 to be

$$X^2 = |\langle e^{i\Omega}\rangle|^2 \tag{3.3}$$

$$= \int_0^{2\pi}\int_0^{2\pi} d\Omega_1 d\Omega_2 e^{i\Omega_1} e^{-i\Omega_2} P(\Omega_1)P(\Omega_2) \tag{3.4}$$

or the norm of the first moment of the wrapped normal distribution.

The goal of our estimation of Ω is to minimize the standard deviation σ. To accomplish this, we can manipulate k through additional rotational gates, and thus we can describe the minimization of σ with the relation

$$\frac{d\sigma}{dk} = 0. \tag{3.5}$$

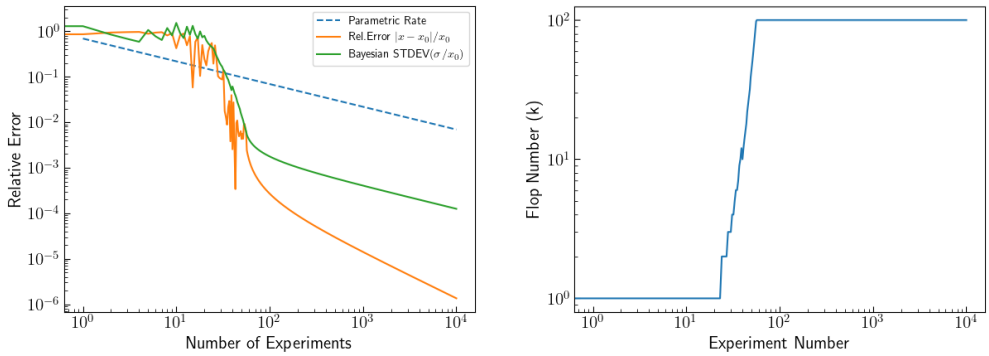
We observe that this relation is monotonic about X^2, and thus we can instead solve the relation

$$\frac{dX^2}{dk} = 0. \tag{3.6}$$

We can perform numerical differentiation of this expression and calculate the required standard deviation values that correspond to each k value. We can then write k as a function of σ, yielding

$$k(\sigma) = \begin{cases} \lfloor \frac{0.22507912}{\pi\sigma} \rceil & \sigma < 0.025258200269627846 \\ 4 & \sigma < 0.03269749744511768 \\ 3 & \sigma < 0.04684580115873045 \\ 2 & \sigma < 0.08688382635251184 \\ 1 & \text{otherwise.} \end{cases} \tag{3.7}$$

Prior to each experimental shot, we compute k and insert it into our likelihood. With this modification, we observe the outcome shown in figure 3.5 where we see we have overtaken the parametric rate of convergence, significantly increasing the rate at which our model learns.



(a) (b)

Figure 3.5: Figure 3.5a displays the Bayesian estimation of Rabi frequency with optimization of parameter k. The change in flop number over the course of the experiments is shown in 3.5b. As the Bayesian standard deviation decreases, we increase k and apply an increasingly multimodal likelihood to the estimation procedure until the k ceiling of $k = 100$ is reached.

Additionally, figure 3.6 demonstrates that this estimation algorithm also consistently converges.

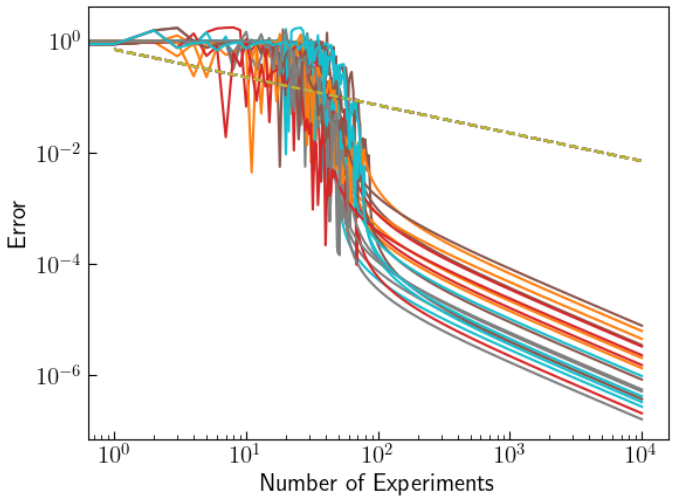


Figure 3.6: 20 runs of the k-optimize estimation algorithm. The dashed line denotes the parametric rate.

3.3 Univariate Normal Substitution of the Wrapped Normal Distribution

The use of the wrapped normal distribution during the Bayesian estimation process is to ensure that our Rabi frequency values are always wrapped within $[0, 2\pi]$. Values that leak out of the $[0, 2\pi]$ range tend to cause numerical issues and become nonsensical. Furthermore, we can find an analytical solution for the denominator of Bayes' rule for the wrapped normal distribution and avoid the numerical integration procedure that is often necessary during the calculation of Bayes' rule. This is the reason the univariate normal distribution is not used during the base estimation process.

However, the wrapped normal distribution can often be quite cumbersome when we seek to apply future optimizations and modifications, as changes to the likelihood and prior require additional calculations and re-definitions of the equations described in section 2.4. In contrast, the univariate normal distribution is better studied and enables simplified calculations. To this end, we seek to develop a method of approximation that enables the use of the univariate normal distribution without disturbing the model's ability to learn. This will greatly expedite and simplify the computational machinery required for some of the experiments we perform later.

To do so, we must develop a way to properly parametrize the normal distribution within a span of π in the bounds $[0, 2\pi]$. We accomplish this by utilizing the properties of the normal distribution's σ. We observe that for a normal distribution, data is distributed in accordance with figure 3.7 where each set of vertical lines moving away from the mean

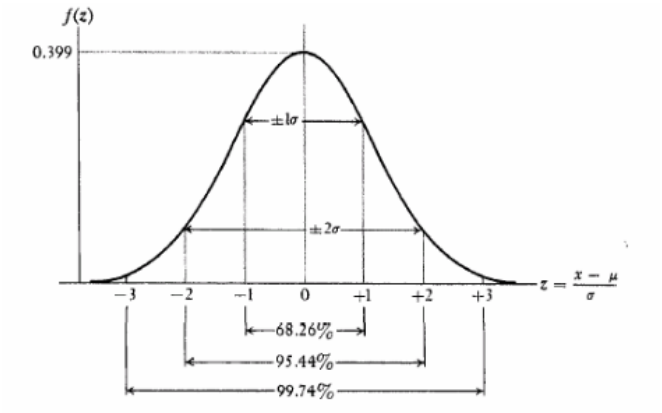


Figure 3.7: The univariate normal distributions plotted across standard deviations from its mean. This figure is taken from Richard Williams' statistics course notes [61].

centered about zero denotes an additional standard deviation. We observe that when we are four standard deviations away from the mean, the probability of measuring the data under the curve is close to zero. With this in mind, we can postulate that when our posterior σ reaches $\pi/4$ during the Bayesian estimation process, we can parametrize the subsequent

prior distribution within $[n\pi, n\pi + \pi]$ where $n \in \{-\infty, \infty\}$ and is an integer. We can start our estimation process with the wrapped normal distribution, and upon reaching the desired σ, swap to the univariate normal representation of the prior. However, it should be noted that since our likelihood is not normal, our output posterior will also not be normal and thus we will need to fit it to a normal curve to continue our estimation, similar to the wrapped normal.

In this instance, we utilize the least squares optimization method of fitting a normal curve to an existing curve. We seek to minimize the sum of squares error between our non-normal curve and the subsequently fitted normal curve. This is accomplished by taking the fitted normal μ and σ to be

$$\mu = \sum_{i=0}^{n} \frac{x_i y_i}{y_i}, \tag{3.8}$$

$$\sigma = \sqrt{|\frac{\sum_{i=0}^{n} y_i (x_i - \mu)^2}{\sum_{0}^{n} y_i}|} \tag{3.9}$$

where x and y define the x and y values of the curve we are fitting the normal to. Replacing the wrapped normal method with this form of curve fitting yields convergence at the rate depicted in figure 3.8.

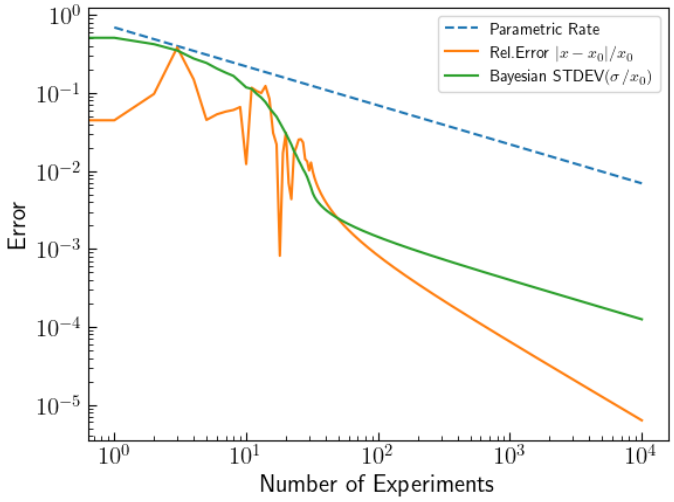


Figure 3.8: Multishot Bayesian filtering experiment with normal approximation when $\sigma = \pi/4$

As demonstrated in figure 3.9, over repeated trials, we see that the normal prior describes the wrapped normal distribution well. We will thus consider the univariate normal distribution to be an adequate replacement for the wrapped normal distribution when $\sigma < \frac{\pi}{4}$.

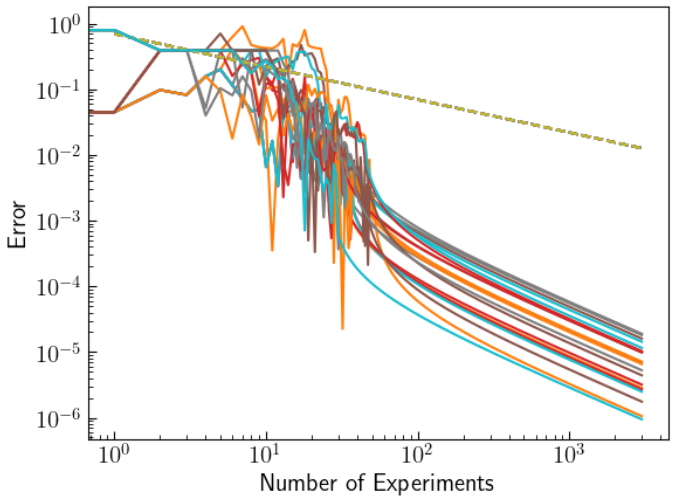


Figure 3.9: 20 repetitions of Bayesian estimation with the univariate normal distribution.

3.4 Error Sources and Robustness

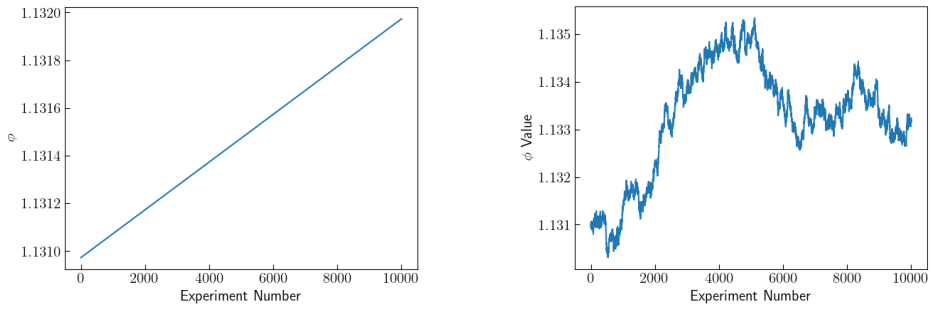
In this section, we will investigate the effect of error sources such as parameter drift and SPAM error and discuss methods to increase robustness of estimation and minimize the impacts of such errors.

3.4.1 Drift

Quantum gates are driven by lasers, which can drift in amplitude or frequency due to factors such as thermal noise. As a result, our optimal Ω value may not stay fixed over time and thus we must develop a model that can continually estimate this nonstationary parameter.

We model laser drift as both a linear and a Brownian drift. In the case of the linear drift, we simply take Ω to constantly increase linearly over time. This effect is demonstrated in figure 3.10a. In the case of Brownian drift, where we model Ω to walk stochastically over the course of experimental shots, we model drift dynamics with the Wiener method [63]. A sample of Ω linear drift is shown in figure 3.10a and a sample of Brownian drift is shown in figure 3.10b.

With this drift added to our Ω value, we observe the behavior shown in figure 3.11 and the baseline Bayesian estimation method no longer converges.



(a) (b)

Figure 3.10: Figure 3.10a displays Linear drift of Ω with total magnitude 0.001 over 10,000 experimental iterations. Figure 3.10b displays Brownian drift over 10,000 experiments with Brownian parameter $\Delta = 0.0001$ and $dt = 10$.

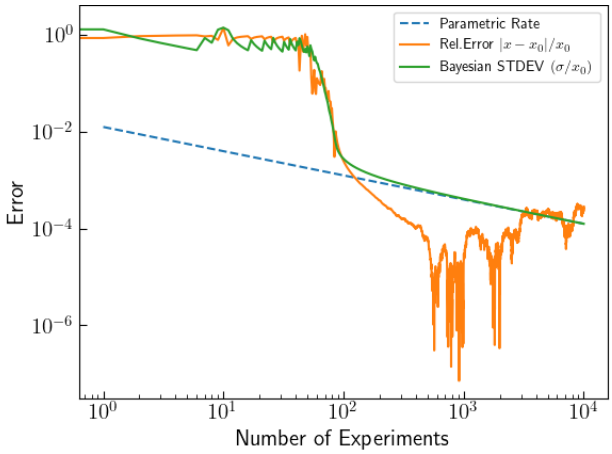
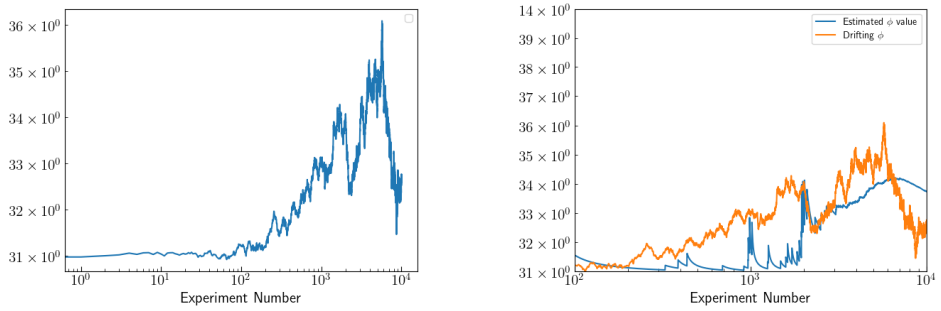


Figure 3.11: Bayesian estimation of Ω with Brownian drift of $\delta = 0.0001$ and $dt = 10$. Though the posterior standard deviation still decreases, the relative error no longer converges.

This result is due to a shortcoming in our Bayesian estimation process. During Bayesian estimation, we utilize our previous information in later cycles, and we weight each prior in the same manner. Though Ω is drifting away from its initial value, we still take into consideration the knowledge gained from the Bayesian estimation performed at the very start of our iterations, when Ω has not drifted much at all. In fact, all knowledge gained prior to the current iteration causes some amount of skew, as it all represents measurements taken with past values of Ω that now differ from the current drifting value.

Additionally, in the presence of parameter drift, the estimator eventually will become trapped in an incorrect mode. Once the posterior standard deviation becomes sufficiently low, the k value will become so high that the modes of the likelihood function will become highly peaked and narrow. As the Ω value drifts away, the estimator is unable to escape this mode and thus relative error will eventually diverge.

We visually observe this effect in figure 3.12 by extracting the change in estimated μ value over time and the drift in Ω.



(a) (b)

Figure 3.12: Figure 3.12a displays the general path of Ω with Brownian drift with the value of Ω on the y-axis and 3.12b displays a fine-grain image of both the estimated $\hat{\Omega}$ and the drifting true Ω plotted on iterations 100 – 10000 when the estimated $\hat{\Omega}$ begins to near the true Ω, where the y-axis again displays the Ω value. We see that $\hat{\Omega}$ initially undershoots Ω due to skew resulting from earlier values of Ω that are relatively smaller. Later on, as Ω begins to decrease, our estimated $\hat{\Omega}$ then overshoots, as it retains information from when Ω is larger in previous iterations. Furthermore, as the number of experiments increases, the estimated value remains bound at a particular value while the drifting Ω eventually goes to a series of different ones.

Though the magnitude of drift may be low or even negligible in most estimation cases, it is nevertheless useful to develop estimation routines that are resilient to drift and are capable of detecting when severe drift is occurring during the estimation cycle. In later sections, we will discuss drift detection and methods to track drift and retain convergence when there is drift in the system. We will primarily focus on linear drift, as that is the more realistic case that will arise in estimations on physical hardware. Linear drift can arise from the gradual increase in temperature during the day, heating of the environment due to atomic ovens, and expansions and contractions of the trap.

3.4.2 Drift Detection

This subsection implements an assortment of drift detection methods and discusses their effectiveness.

Predictive P-value

If there is some drift in the parameter we are attempting to estimate, we would expect the Bayesian estimation to attempt to track and follow the drifting parameter. In the more realistic case of a linear drift, we would thus expect the posterior mean to gradually increase

and track the increasing Ω as estimation continues. If there is some drift in the parameter, we would expect the output posterior distributions to similarly drift, especially during early cycles of the estimation while σ is still high. To detect this, we can utilize the Bayesian predictive p-value.

To generate the p-value, we perform the following steps:

1. Perform Bayesian estimation and obtain some posterior distribution.
2. Draw a series of samples $\hat{\Omega}$ from the posterior.
3. With each $\hat{\Omega}$, generate a set of data $\hat{x}$ with the sampling function. In this instance, our generated data will be a series of $|1\rangle$s and $|0\rangle$s. Note that all $\hat{x}$ are simulated data and do not require additional measurements.
4. Compare the set of $\hat{x}$ with the measurement data x that was drawn from the Bayesian estimation protocol.

When drift is present, we would observe subsequent posteriors to be shifted away from the true mean as the Bayesian estimation procedure responds to the drifting Ω value. When comparisons of $\hat{x}$ and x are made, we would expect to see a greater quantity of $|1\rangle$s in x than in $\hat{x}$ (when there exists a linear drift that is positive), a greater quantity of $|0\rangle$s in x than in $\hat{x}$ (when there exists a linear drift that is negative), or an oscillation between the two above cases when there exists some Brownian drift in the system.

Over the course of the estimation, some anomalous runs may yield false positives of drift, and thus to ensure the accuracy of this test, we will examine multiple p-values throughout consecutive cycles of estimation. If a constant pattern of skew or oscillatory behavior arises, then we will flag the estimation to be drifting. An additional source of false positives that we must be aware of is the natural variance in the amount of counts that will arise even when sampling with the observed data, we must also designate some threshold that count differences must exceed in order to generate a positive detection of drift.

This methodology appears to be unsuccessful at detecting drift, especially at lower magnitudes. Since the estimated Ω values in a drifting and non-drifting estimation cycle vary by so little (on the order of 10^{-2} during most runs), the share of $|1\rangle$s sampled does not differ much between non-drifting estimations and drifting estimations until extraordinarily high and unrealistic drift magnitudes exist in the system. The results of a set of predictive p-value check runs are displayed in figure 3.13 demonstrate this effect.

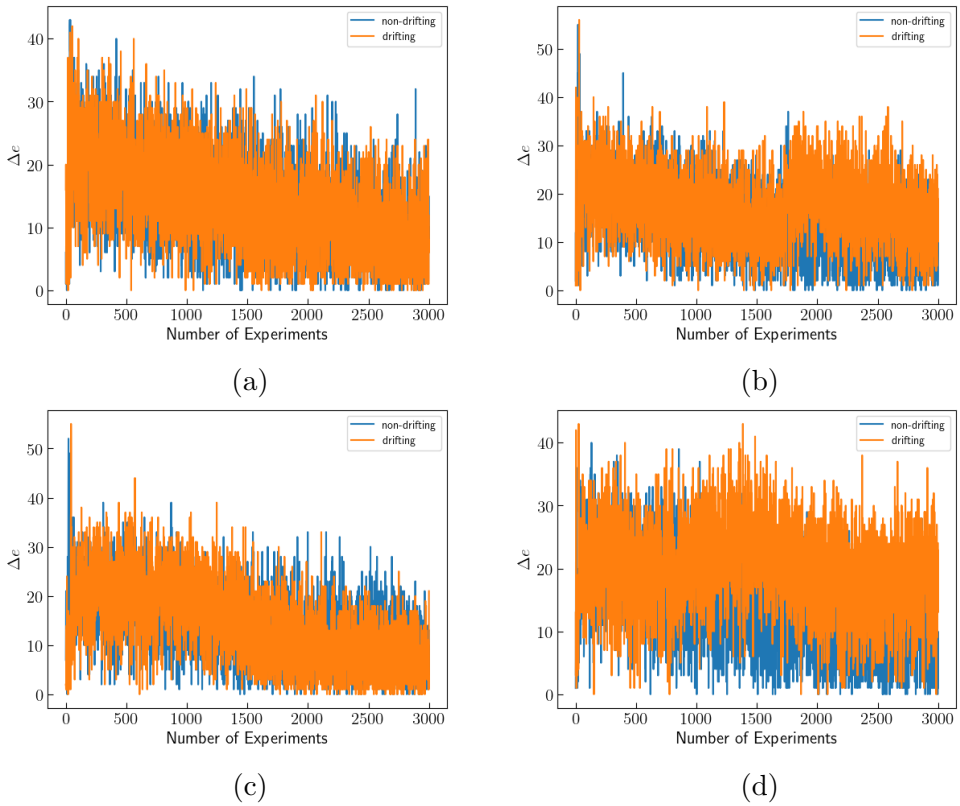


Figure 3.13: Posterior predictive check count differences for estimations run at various linear magnitudes (orange) against posterior predictive check count differences for baseline estimation without drift (blue) (Δe on the y-axis denotes the number of $|1\rangle$s measured). 3.13a drifts at a linear magnitude of 0.0001 over 3000 trials (3.33×10^{-8} per measurement), 3.13b at a linear magnitude of 0.001 over 3000 trials (3.33×10^{-7} per measurement), 3.13c at a linear magnitude of 0.01 over 3000 trials (3.33×10^{-6} per measurement), and 3.13d at a linear magnitude of 0.1 over 3000 trials (3.33×10^{-5} per measurement).

There is no immediately discernible trend that arises from the posterior predictive checks. The baseline and drifting cases tend to be overlap even at higher linear drift magnitudes where there is obvious divergence in the estimation process. From figure 3.13, we also observe a generally high variance from both baseline and drifting posterior predictive checks. Further investigation of this effect is displayed in figure 3.14.

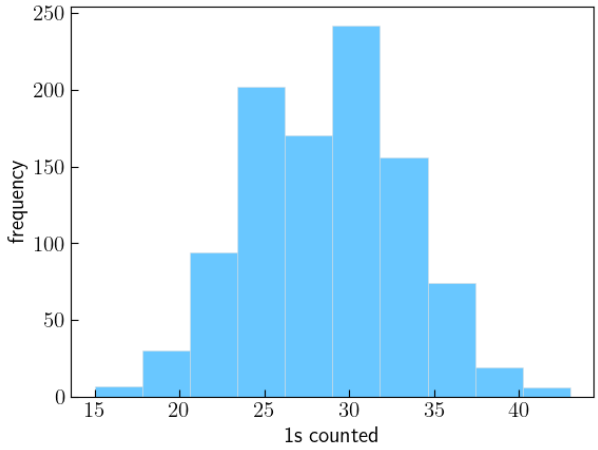


Figure 3.14: A histogram of 1s counted when 100 samples are drawn for a single Ω value over 1000 trials.

The standard deviation of the data generated from figure 3.14 is 4.55, and this high value is a major contributing factor to the lack of information provided by our posterior predictive checks. The difference in Ω values following a estimation without drift and a estimation with drift typically is only on the order of magnitude of 10^{-3} to 10^{-2}, and thus such a high variance in the 1 counts generates far too much noise to discern a drifting case from the baseline case. A drift detection methodology must maintain a high degree of precision to be able to generate a signal that can be aptly utilized, and the posterior predictive check is incapable of doing so.

Z-score Progression

An alternative to the aforementioned posterior predictive check is the z-score. The z-score is defined to be

$$\frac{\theta - \mu}{\sigma} \tag{3.10}$$

where θ denotes the mean of the observed values (measurements), μ denotes the mean of the sampled values (simulated), and σ denotes the standard deviation of the sampled values. We draw the observed values by making measurements of the system and draw the sampled values by generating measurements with our sampling function using the current posterior mean $\hat{\Omega}$. The z-score yields the number of standard deviations the sampled value $\hat{\Omega}$ value is from the observed Ω value, and due to the division by σ, may yield a tighter distribution of

values when compared to the posterior predictive check. To generate this metric, we perform the following procedure:

1. Generate n measurements of $|1\rangle$s and $|0\rangle$s and compute the mean of those measurements.
2. Generate n samples of of $|1\rangle$s and $|0\rangle$s from $\hat{\Omega}$ and compute the mean of those measurements.
3. With these components, compute the z-score with equation 3.10.

We perform z-score calculations for estimations with linear drift at magnitudes of 0.0001, 0.001, 0.01, and 0.1. We plot these z-scores alongside a baseline estimation without drift.

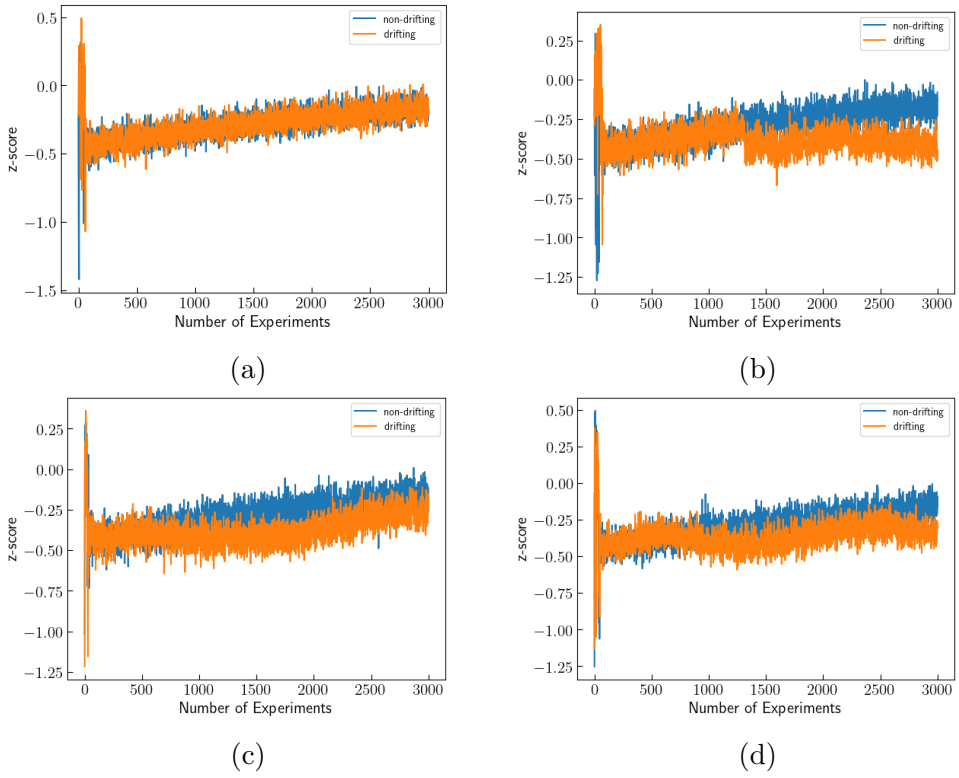


Figure 3.15: Z-scores of estimation procedures run at various linear magnitudes (orange) against z-scores of baseline estimation without drift (blue). 3.15a drifts at a linear magnitude of 0.0001 over 3000 trials (3.33×10^{-8} per measurement), 3.15b at a linear magnitude of 0.001 over 3000 trials (3.33×10^{-7} per measurement), 3.15c at a linear magnitude of 0.01 over 3000 trials (3.33×10^{-6} per measurement), and 3.15d at a linear magnitude of 0.1 over 3000 trials (3.33×10^{-5} per measurement).

Though we observe some clear distinctions between estimations with and without drift when the magnitude exceeds 0.0001 over the course of 3000 trials, there is not enough differentiability to concretely yield a set of tests that can consistently provide information

on whether drift occurs in the estimation or not. Though the z-score test performs better than the posterior predictive checks described in section 3.4.2, the variance in the sampled $\hat{\Omega}$ values and the variance in the $|1\rangle$s and $|0\rangle$s yielded from the sampling function coupled with the error magnitude that arises from drift (typically on the order of 10^{-3} to 10^{-2}) reduce its effectiveness.

Maximum Likelihood Estimation

When the Ω value is drifting linearly, an additional protocol that may be able to detect this change is repeated maximum likelihood estimation (MLE). MLE is a procedure that estimates some unknown parameter given observed data. We can define it to be

$$\hat{\Omega} = \arg\max L(\Omega|m) \tag{3.11}$$

Where $L(\Omega|m)$ defines a likelihood function, Ω defines the parameter and m defines the measured data. MLE seeks to maximize $L(\Omega|m)$ so that the observed data m is most probable.

Since maximum likelihood estimation solely draws its conclusions from data and, unlike our Bayesian estimation procedure, is not affected by the prior distribution, we can utilize it to determine if Ω is drifting. Maximum likelihood estimation performed with a linearly increasing Ω will yield larger and larger values of $\hat{\Omega}$, and if enough increasing $\hat{\Omega}$ values are accumulated, we can conclude that there is some increasing linear drift in the system. The same logic can be applied for the linearly decreasing case.

Our MLE protocol is simplified by the fact that our likelihood is a cosine likelihood. By taking multiple measurements, analogous to repeated multiplications of the cosine likelihood, we will end with a normal distribution describing the likelihood of each Ω value (this result will be discussed in greater detail in section 3.5. Since our output distribution is normal, we do not need any advanced sampling techniques to determine the value resultant from MLE. We can simply pick the mean of the data. Our MLE protocol will consist of the following:

1. Make n measurements.
2. Generate n likelihoods using each measurement outcome.
3. Multiply the likelihoods together.
4. Return the mean of the output distribution.

We perform iterative MLE for various degrees of linearly increasing Ω and compare it against a baseline case. The results are shown in figure 3.16.

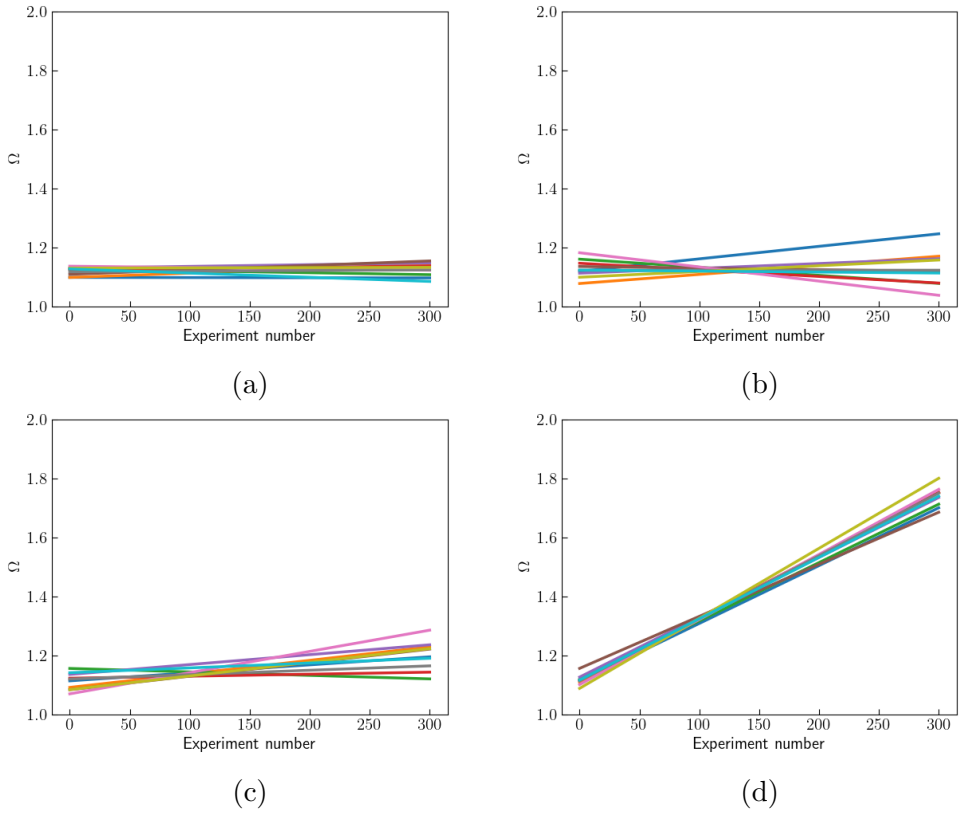


Figure 3.16: Linear regressions of MLE values obtain from runs at a drift-less base case and various linear magnitudes. 3.16a denotes the base case and has no drift. 3.16b drifts at a linear magnitude of 3.33×10^{-8} per measurement, 3.16c at a linear magnitude of 3.33×10^{-7} per measurement, and 3.16d at a linear magnitude of 3.33×10^{-6} per measurement.

Though there appears to be no discernible difference for the drift magnitude 3.33×10^{-8} per measurement and the base case, drift magnitudes of 3.33×10^{-7} per measurement and above demonstrate clear linear increases in output MLE values over the course of experiments. We can thus determine a threshold for slope of MLE increase over experiments and utilize it as a signal for a linearly drifting estimation cycle.

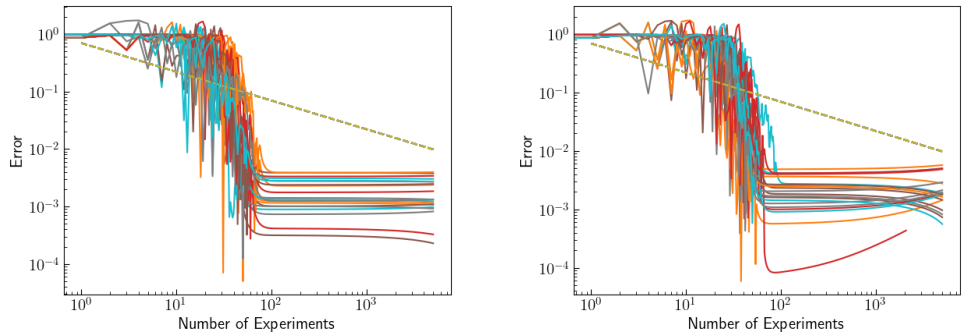
This MLE drift detection methodology pairs well with the batched estimation process that will later be described in section 3.5, as the batching process computes the majority of the components required to perform MLE in a single experimental shot. We can obtain the MLE by simply extracting the x-coordinate that corresponds to the maximum y value in the batched likelihood.

3.4.3 Drift Resilient Estimation

Forgetting

The problem of divergence in Bayesian estimation for a nonstationary parameter has been studied in the past. A myriad of schemes, including Bayesian unlearning and adaptive weighting methods [64–69], have been developed to resolve this issue, though we opt to first test a simpler solution of de-emphasizing previous trials through exponentiation.

After each iteration, we can exponentiate the output posterior - effectively expanding the σ of previous posterior trials to reduce their confidence in the estimated $\hat{\Omega}$ value. A reduction of confidence in the posterior theoretically enables the model to better learn and change during the estimation, thus allowing it to better track a drifting parameter. However, when we apply this method, we do not obtain suitable results, and our estimation process consistently diverges. Interestingly, when we instead exponentiate the posterior by a positive number (which we will term the forgetting factor), further reducing the width of the posterior, we observe eventual convergence during the estimation cycle. This appears counter-intuitive, as a lower uncertainty reduces the capacity for the model to learn. We investigate forgetting factors of 1.001 to 1.01 at increments of 0.001 and determine optimal forgetting factors at various drift magnitudes. Our results shown in figure 3.17 display general success in tracking the drifting Ω value at drift magnitudes of 0.0001 and 0.001 over 3000 trials.



(a) (b)

Figure 3.17: Figure 3.17a demonstrates the results of 20 estimation runs with a linear drift magnitude of 0.0001 over 3000 trials and a forget factor of 1.009. All 20 of the estimation runs converge with the application of exponential forgetting. Figure 3.17b demonstrates the results of 20 estimation runs with a linear drift magnitude of 0.001 over 3000 trials and a forget factor of 1.015. The majority of estimation runs tend to converge with exponential forgetting implemented, though some do diverge.

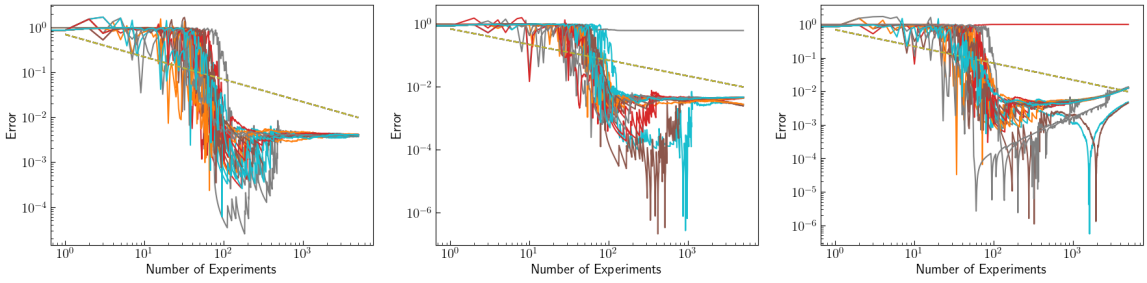
However, as the drift magnitudes begin to increase, our results become less desirable. At higher drift magnitudes, higher forgetting factors are required to maintain convergence, yet too high a forgetting factor may result in a estimation that quickly diverges. At drift magnitudes of 0.01 over 3000 trials our exponential forgetting method is no longer effective.

It should also be noted that this methodology of forgetting yields estimation runs that converge to a higher relative error, typically sitting in the range of 10^{-2} to 10^{-3}. This is likely due to the time lag that arises in the model's response to drift. Though the model is capable of tracking the changing Ω value, it is unable to instantaneously respond, and thus there always exists some distance between $\hat{\Omega}$ and Ω. If we desire to implement model that is capable of tracking drift, then it will most likely come at the cost of accuracy due to this latency.

Sampling k

Another parameter that contributes to divergence of the estimator when drift is present in the system is k, the number of rotational gates applied. k moderates the standard deviation of our posterior distribution as it narrows the standard deviation of each likelihood mode when increased. Furthermore, the k optimization method may also cause the estimator to be trapped in an incorrect mode when drift is present in the system. Thus one method to widen the posterior distribution and prevent the estimator from becoming trapped is to periodically decrease k from the calculated value described in section 3.2. However, excessively decreasing k may result in a model that is too slow to learn. To resolve this issue, we opt to sample k from a probability distribution through rejection sampling.

We test three different distributions, linear, exponential, and logarithmic bounded on $[0, k]$ that are primarily distinguished by their tails as they approach k. We run a estimation routine with linear drift magnitudes of 0.0001, 0.001, and 0.01 for each distribution. Out of the three, the linear and logarithmic tend to provide results that fail to converge, though the exponential distribution is able to continually track the drifting Ω parameter. It is capable of maintaining convergence on drift magnitudes of 0.0001 and 0.001, though is entirely unsuccessful at 0.01. The results of the estimation of a drifting Ω while sampling k from an exponential function are displayed in 3.18.



(a) (b) (c)

Figure 3.18: The results of estimations with drifting Ω and k-sampling. Figure 3.18a demonstrates the results of 20 estimation runs with a linear drift magnitude of 0.0001 over 3000 trials. All 20 of the estimation runs converge with the application of exponential forgetting. Figure 3.18b demonstrates the results of 20 estimation runs with a linear drift magnitude of 0.01 over 3000 trials. With this magnitude, the majority of estimation runs tend to diverge.

An advantage gained by sampling k rather then utilizing exponential forgetting is that our sampling distribution does not need to be adjusted in accordance to the drift magnitude, whereas the exponential forgetting technique requires differing forgetting factors at different magnitudes. This generalizability increases the robustness of the technique when applied to a real-world estimation.

It should also be noted that an increased error rate, analogous to the phenomena described in section 3.4.3 arises when applying this method of sampling k. Once again, we are sacrificing accuracy for a methodology that is capable of tracking the drifting parameter.

3.4.4 SPAM Error

Another source of error that may arise during estimation is SPAM error. This form of error is discussed in section 1.2.2 and results in a random probability of measuring the qubit to be in the ground or excited state regardless of the settings of the driving laser. With this effect in mind, we can rewrite our sampling function to be

$$P_{SPAM}(|1\rangle) = 0.999(\sin^2\left(\frac{k\Omega}{2}\right)) + 0.001(0.5). \tag{3.12}$$

If we do not adjust our likelihood to account for SPAM and proceed with the estimation procedure as described in section 3.2, we observe the results shown in figure 3.19. We see

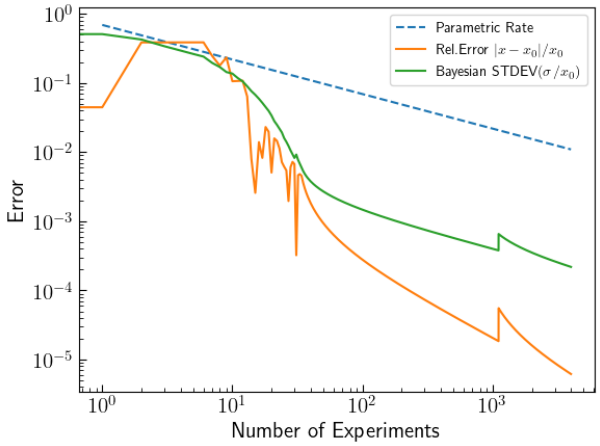


Figure 3.19: Estimation trial with 0.01% SPAM error in the system and no adjustment to the likelihood.

that SPAM error has the effect of introducing a sharp uptick of error in the estimator, which is seen at around iteration number 1000 in figure 3.19. This artifact in the plot is particularly interesting as it also manifests in the Bayesian posterior standard deviation, a variable that we have access to. It thus may be possible to analyze the posterior standard deviation in real-time during the estimation procedure and develop an algorithm to remove measurements and estimation cycles performed with SPAM error.

To verify the consistency in these occurrences, we run another 20 trials, observing the results shown in figure 3.20. We see that SPAM error has the constant effect of causing a

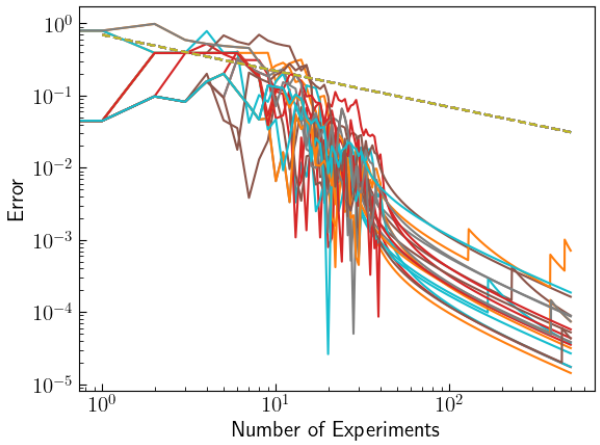


Figure 3.20: 20 cycles of estimation trials with 0.01% SPAM error in the system and no adjustment to the likelihood.

sharp uptick in error when it arises in a particular experimental shot, though subsequent experimental shots that are not afflicted eventually correct for it. To better understand the effect of SPAM error, we run estimation procedures where its occurrence is increased to 0.1%, observing the results shown in figure 3.21. When the error rate is increased, we observe a

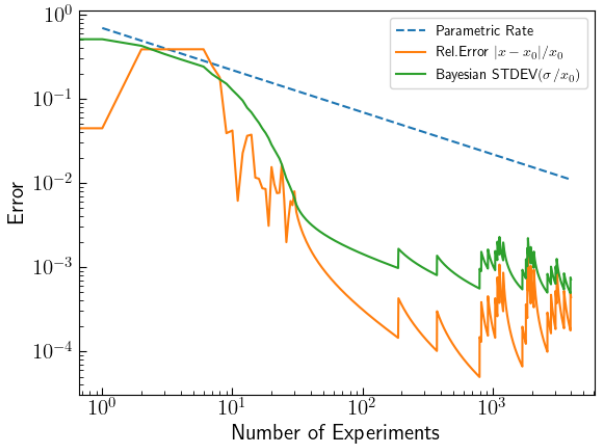


Figure 3.21: 20 cycles of estimation trials with 0.1%SPAM error in the system and no adjustment to the likelihood.

much more drastic frequency of error upticks, and the estimator appears to hit a floor at the posterior standard deviation of 10^{-3}. Though the typical SPAM error in our system will not be this high, it is still useful to know the relation between increasing SPAM error and the final output of the estimator.

To reduce the effect of SPAM error, we can similarly give the likelihood a 0.01% chance to collapse to a uniform distribution during each cycle of estimation. This is analogous to receiving a randomized measurement result, as the likelihood is uninformative for our Bayesian estimation procedure. Upon incorporating this effect, our single cycle likelihood becomes:

$$L_{SPAM} = 0.999(\cos^2\left(\frac{k\Omega}{2}\right) - \frac{m\pi}{2}) + 0.001(0.5). \tag{3.13}$$

Running the estimator with our new likelihood over 20 trials, we observe the results shown in figure 3.22. Though the updated likelihood is not entirely capable of removing

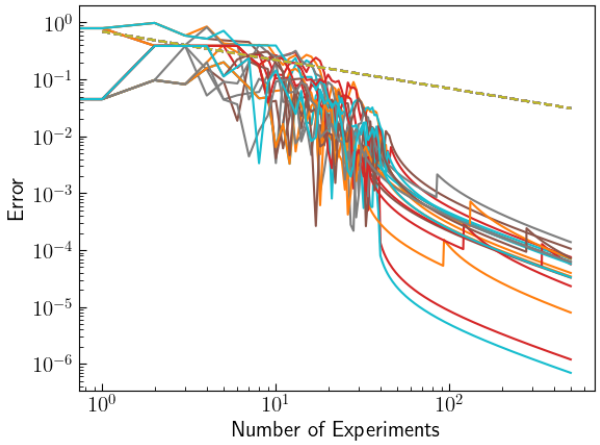


Figure 3.22: 20 cycles of Bayesian estimation trials with 0.01% SPAM error probability in the system and the updated SPAM error likelihood.

SPAM error, we observe that the artifacts present in the plot appear to introduce a lower magnitude of error uptick, suggesting that this updated likelihood slightly reduces the effect of SPAM error on the estimator.

We also want to better understand how use of the SPAM error likelihood affects the estimator when no SPAM errors afflict the system. We perform MLE with our regular likelihood and our SPAM error-afflicted likelihood for 100 samples and examine the difference in outputs for each, observing the results shown in figure 3.23.

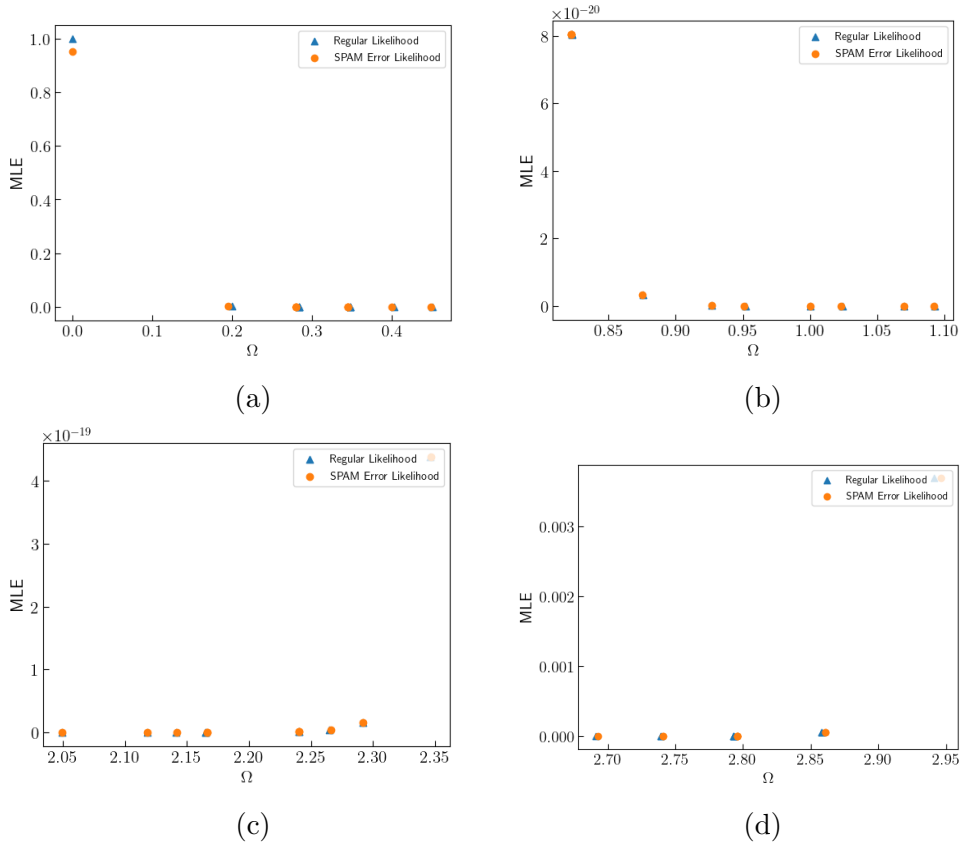


Figure 3.23: MLEs for SPAM likelihoods for various Ω values plotted with MLEs for regular likelihoods. Figure 3.23a is generated with $\Omega = 0.1\pi$, Figure 3.23a is generated with $\Omega = 0.4\pi$, figure 3.23a is generated with $\Omega = 0.7\pi$, and figure 3.23a is generated with $\Omega = 0.9\pi$. Each trial is run 10 times, with each point representing an individual trial. Note that some points overlap.

From figure 3.23, we see that MLEs drawn from the regular likelihood are almost identical to MLEs drawn from the SPAM likelihood. This demonstrates that the use of the SPAM error-afflicted likelihood is robust even in the case of no SPAM error in the system.

In summary, we see that though SPAM error does have an effect on an estimator in the form of sporadic error upticks, continual estimation trials can eventually remedy these errors, and a new likelihood that accounts for this occurrence can reduce disruptions to the estimator. Furthermore, when this new likelihood is applied in a series of trials where no SPAM error occurs, there are no adverse effects on the estimator.

3.5 Batching

We will now examine an alternative approach to the parameter estimation protocol. Instead of performing Bayesian estimation after taking a single measurement from the system, we will take a batch of measurements and then perform the estimation procedure. This batching approach to the Bayesian filtering process is advantageous as it enables a normal approximation to the cosine likelihood, a convenient and efficient replacement that leads to a variety of numerical simplifications and computational speed-ups.

Let us first examine our current likelihood, where we take a single measurement prior to performing estimation. We note that our likelihood function $L(\Omega|m) = \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)$ is a Bernoulli likelihood as it yields a value of either a $|1\rangle$ or a $|0\rangle$. When we take multiple measurements, which can be modeled as a sum of Bernoulli random variables, our output distribution can be characterized by the binomial distribution

$$P(n, s, p) = \begin{cases} \binom{n}{s} p^s \times (1-p)^{n-s} & \text{if } s = 0, 1, \cdots, n \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

where s defines the number of successes and p defines the probability the ion is measured in the $|1\rangle$ state. We can insert our cosine likelihood for p, yielding

$$P(n, s, \Omega, m, k) = \begin{cases} \binom{n}{s} \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)^s \left(1 - \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)\right)^{n-s} & \text{if } s = 0, 1, \cdots, n \\ 0 & \text{otherwise.} \end{cases} \tag{3.15}$$

The Central Limit Theorem states that as the number of samples n increases, the binomial distribution converges towards the normal distribution, and thus if we pick a sufficient batch size, we can replace our binomial distribution with a normal distribution where

$$\mu = np, \tag{3.16}$$

$$\sigma^2 = np(1-p). \tag{3.17}$$

However, to obtain a good approximation, we must abide by some conditions, namely:

$$\begin{cases} np > 5 \\ n(1-p) > 5. \end{cases} \tag{3.18}$$

3.5.1 Fisher Information

The batching method enables us to assume that our likelihood is approximately normal. However, we still need a method to fit a normal distribution to our data. Since we are now dealing with a likelihood that is approximately a normal distribution, we can choose to calculate the μ and σ of our batched likelihood with the Fisher Information. We first find the μ value of our approximated normal by taking it to be the x value corresponding to the maximum y value of our distribution (the mode of the distribution). Then, we can utilize

the Fisher information to calculate σ. For a given PDF $P(X|\theta)$, we can define some $\lambda(X|\theta)$ where

$$\lambda(X|\theta) = \log P(X|\theta). \tag{3.19}$$

We will define the score to be $\lambda'(X|\theta)$ where we take the first derivative with respect to θ. We can then define the Fisher information $I(\theta)$ to be the variance of the score, or

$$I(\theta) = E_\theta[\lambda'(X|\theta)^2] \tag{3.20}$$

where E_θ defines the expectation value of θ. We can also define the Fisher information in terms of the second derivative of λ:

$$I(\theta) = E_\theta[-\lambda''(X|\theta)]. \tag{3.21}$$

In the case of a normal distribution $\mathcal{N}(\mu, \sigma^2)$, we observe that

$$\lambda(x|\mu) = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(x-\mu)^2, \tag{3.22}$$

$$\lambda'(x|\mu) = \frac{1}{\sigma^2}(x-\mu), \tag{3.23}$$

$$\lambda''(x|\mu) = -\frac{1}{\sigma^2}. \tag{3.24}$$

We can then apply 3.21 to the Fisher Information for $\mathcal{N}(\mu, \sigma^2)$, finding that

$$I(\mu) = \frac{1}{\sigma^2} \tag{3.25}$$

and as a result

$$\sigma = \sqrt{\frac{1}{I(\mu)}}. \tag{3.26}$$

Thus we see that we can extract σ from our distribution directly from the Fisher information, provided it is a normal distribution. This provides a theoretical method of calculating a fitted μ and σ that does not require curve fitting. We note that this method is only applicable in the case of a batched likelihood function, as all of our distributions are now normal.

To calculate the second derivative of the log likelihood for the Fisher information, we simply apply an approximated differentiation through a Taylor expansion, writing the second derivative of our desired value λ to be

$$\lambda''(a) = \frac{\lambda(a+h) + \lambda(a-h) - 2\lambda(a)}{h^2}. \tag{3.27}$$

3.5.2 Normal-Normal Linear Combinations

Since both our prior and our likelihood are approximately normal within the batching framework, we can assume our posterior will also be normal, as a normal distribution multiplied

by a normal distribution yields another normal distribution. As a result, we can simply the calculation of the posterior μ and σ. Previously, when our prior and likelihood were not in this form, we perform the computations described in section 2.3.1 or direct multiplication of input distributions followed by an approximation for μ and σ as in subsection 3.3. Now, we are able to examine a linear combination of our two normal distributions and compute μ and σ to be

$$\sigma = \sqrt{\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 + \sigma_2^2}}, \tag{3.28}$$

$$\mu = \frac{\mu_1\sigma_2^2 + \mu_2\sigma_1^2}{\sigma_2^2 + \sigma_1^2} \tag{3.29}$$

where σ and μ denote the posterior mean and standard deviation, σ_1 and μ_1 denote the prior mean and standard deviation, and σ_2 and μ_2 denote the likelihood mean and standard deviation.

We observe that we can now compute the mean and standard deviation of our output posterior with a simplified computation of only four points.

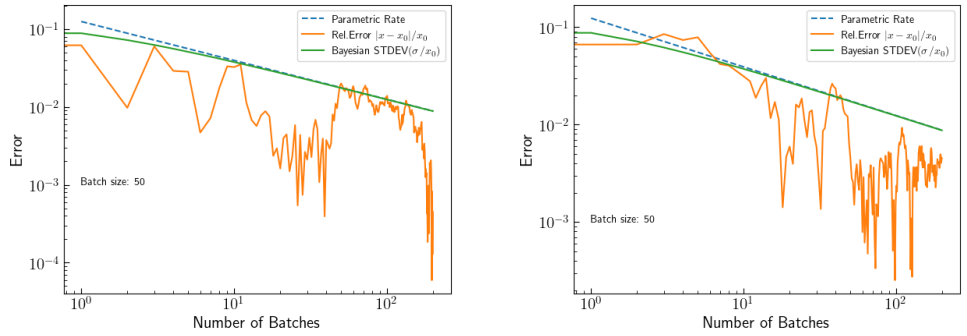
3.5.3 Batching Results

To first gather a set of baseline results, we apply the batching method first with no k optimization. We perform the following algorithm

Algorithm 2 Batched Bayesian Filtering

FirstPrior $\leftarrow$ Normal Distribution $\in [0, 2\pi]$
FirstLikelihood $\leftarrow$ BatchLikelihood(BatchSize)
$\mu, \sigma \leftarrow$ LinearCombination(FirstPrior, FirstLikelihood)
$k \leftarrow 1$
for i in NumberOfShots / BatchSize **do**
    NewLikelihood $\leftarrow$ BatchLikelihood(BatchSize)
    $\mu, \sigma \leftarrow$ `bayesian_update`(μ, σ, samples, k)
end for
return μ, σ

When we apply this algorithm, we observe the results shown in figure 3.24a. We also run the experiment with the curve-fitting method, shown in 3.24b, of fitting the posterior to a normal distribution to verify that both the Fisher information-based method and the curve-fitting method return consistent results. Our results align with the expected parametric rate of error convergence. The primary advantage of the Fisher information is the reduced amount



(a) (b)

Figure 3.24: Figure 3.24a displays the multishot Bayesian filtering experiment with batched likelihood, Fisher information, and linear combination method of computing posterior μ and σ. Figure 3.24b displays the multishot Bayesian filtering experiment with batched likelihood, curve-fitting method, and linear combination method of computing posterior μ and σ.

of computations required to fit a normal distribution when compared with the calculation of a normal distribution through the curve fitting method, which requires summation and multiplication over all points in the distribution.

We run another set of 20 estimations with the batching algorithm, and from figure 3.25, we see that our algorithm consistently converges. Here, we observe that, unlike the baseline

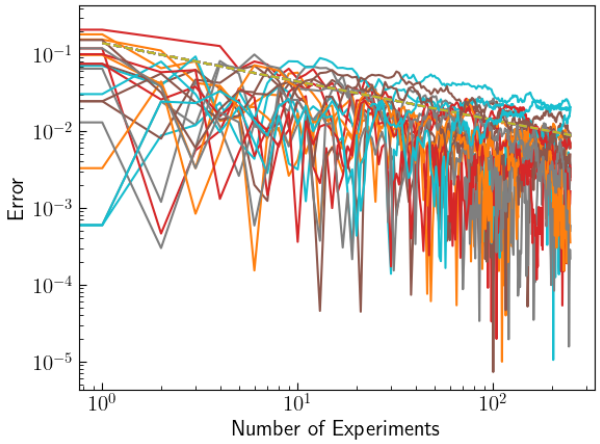


Figure 3.25: 20 runs of the baseline batched estimation algorithm.

and k-optimize estimation mechanisms, we are able to learn immediately when estimation begins, and do not suffer from the poor performance previously seen in the initial 0 – 40 experiments. We will examine this effect in greater detail in section 3.5.5.

3.5.4 k optimization

k optimization can also be applied within the batching paradigm. Like the initial cosine likelihood presented in section 3.2, altering k will have a similar effect on the batched binomial likelihood. We demonstrate a set of likelihoods with various k values and with various quantities of $|1\rangle$s measured in figure 3.26.

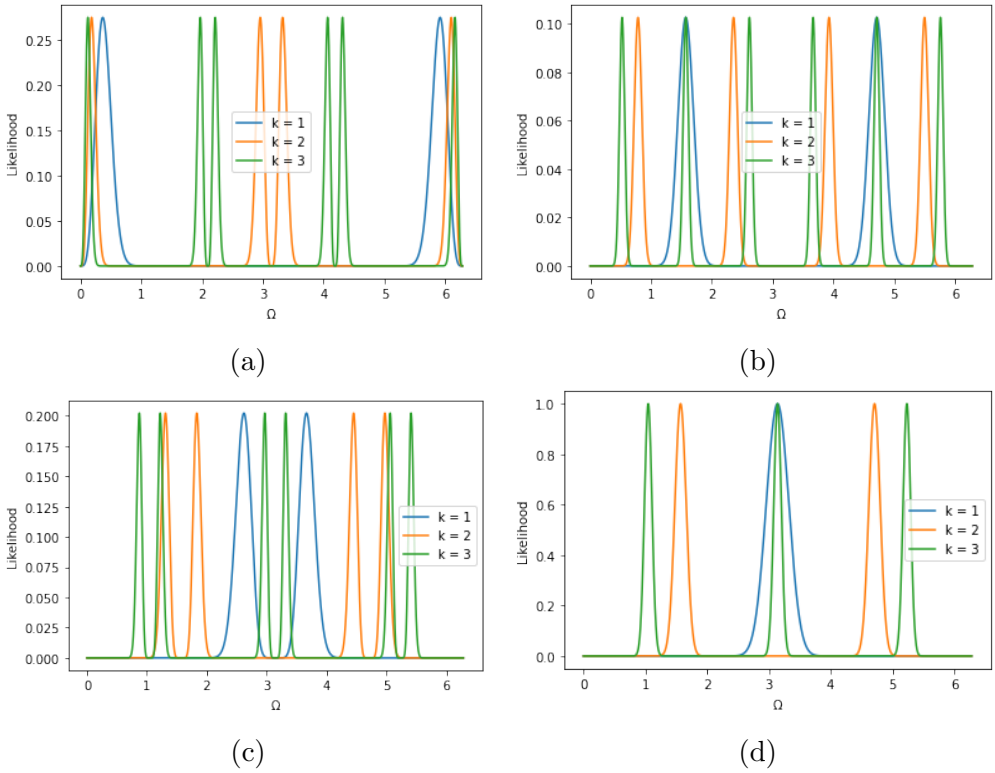


Figure 3.26: Likelihood functions for the batched Rabi frequency with a batch size of 60. Figure 3.26a displays the likelihood for a Rabi frequency where two $|1\rangle$s are measured, figure 3.26b displays the likelihood for a Rabi frequency where 30 $|1\rangle$s are measured, figure 3.26c displays the likelihood for a Rabi frequency where 56 $|1\rangle$s are measured, and figure 3.26d displays the likelihood for a Rabi frequency where 60 $|1\rangle$s are measured.

An interesting phenomena that occurs when we consider the batched likelihood is the tight multi-modality that arises when an extremely high or extremely low quantity of successes is measured, as demonstrated by figure 3.26a, with two successes measured, and figure 3.26c, with 56 successes measured. Consider the batched likelihood function

$$L(\Omega) = \binom{n}{s} \left(\sin^2\left(\frac{k\Omega}{2}\right)\right)^s \left(1 - \sin^2\left(\frac{k\Omega}{2}\right)\right)^{n-s}. \tag{3.30}$$

When s, the number of successes, is small, then $n - s$ is large, and we observe a subtraction of a sinusoid multiplied together a large quantity of times ($n - s \gg s$) from a sinusoid

multiplied a small quantity of times. To visualize this effect, we neglect the binomial term $\binom{n}{s}$ (as it just acts as a scaling factor), and plot the two functions $\sin^2\left(\frac{k\Omega}{2}\right)^s$ and $\sin^2\left(\frac{k\Omega}{2}\right)^{n-s}$ for $n = 60$, $s = 2$, and $k = 2$. Doing so yields the results display in figure 3.27.

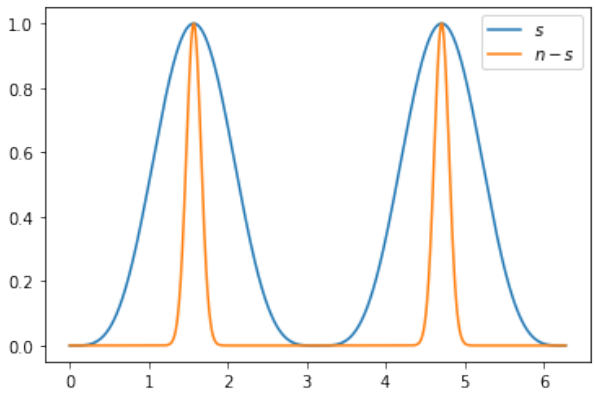


Figure 3.27: The two sine functions that comprise of the binomial likelihood plotted together for $s = 2$, $n = 60$, and $k = 2$. We would expect a similar image for a high success rate, say $n = 56$, though the curve with a lower standard deviation would be the s curve instead of the $n - s$ curve.

When subtraction between the s function and the $n - s$ function occurs, we observe a scooping out of the maxima of the s curve, thus resulting in a highly bimodal curve with minimal separation between the two maxima as seen in some of the likelihoods displayed in figure 3.26. This behavior makes it difficult to enforce the k optimization rules and increases the probability that the posterior may end up within the wrong likelihood mode and thus never converge to the correct value. We can demonstrate this by naively optimizing k without addressing this effect.

We test the batched method with the k optimization rules presented in section 3.2, where

$$k(\sigma) = \begin{cases} \left\lfloor \frac{0.22507912}{\pi\sigma} \right\rceil & \sigma < 0.025258200269627846 \\ 4 & \sigma < 0.03269749744511768 \\ 3 & \sigma < 0.04684580115873045 \\ 2 & \sigma < 0.08688382635251184 \\ 1 & \text{otherwise.} \end{cases} \tag{3.31}$$

Doing so yields the results demonstrated in figure 3.28. We see that with k optimization, the batched optimization method exceeds the parametric rate of convergence and quickly hits its floor error rate. However, in certain cases, the multi-modal likelihood causes the posterior slips into the incorrect mode, such as the example shown in figure 3.29. Due to this, we must insert an additional rule for k optimization during batched estimations. When there is a disproportionate quantity of ions in the ground or excited state following batched measurement, we see that the normal approximation no longer models the batched

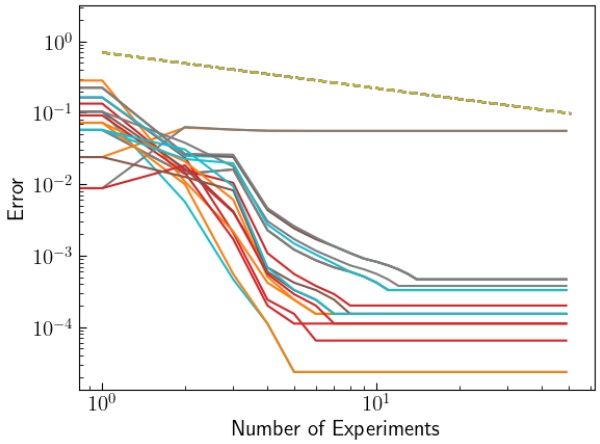


Figure 3.28: A set of 20 estimation runs utilizing batching with k optimization. Batch size is 60, total number of iterations is 500. The brown line that diverges is an example of an estimation that has entered the incorrect likelihood mode.

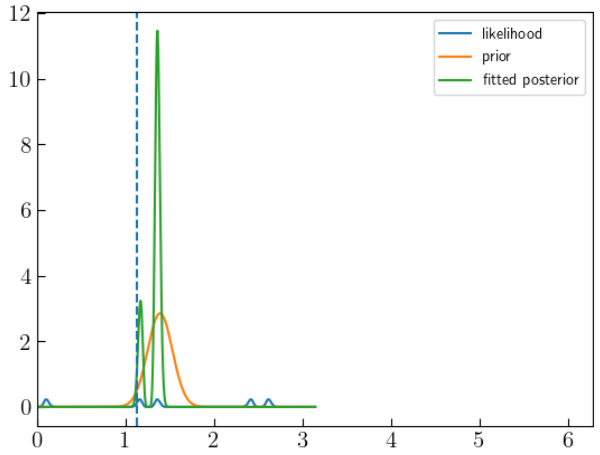


Figure 3.29: A cycle of batched k optimized Bayesian estimation where the prior is caught between two likelihood modes. The dashed blue line displays the position of the correct estimation value. The subsequent posterior that is resultant from the multiplication of prior and likelihood displays a mean value that has been influenced by the incorrect likelihood peak.

binomial likelihood very well anymore, thus introducing the tight multimodality. To detect this, we will simply introduce the condition described in equation 3.18 where the normal approximation is only valid if

$$\begin{cases} np > 5 \\ n(1-p) > 5. \end{cases} \tag{3.32}$$

If this condition holds, then we choose to optimize k accordingly. We observe that for a small p (meaning the ions mostly remain in the ground state), $np < 5$. For a large p (ions mostly are in excited state), $n(1-p) < 5$. Both of these instances violate our condition.

Having identified and developed a method to combat this problem, we will now present a series of estimation methods that involve batching with k optimization.

3.5.5 Greedy Batched Estimation

To quantify how often this multi-modality results in failure of the naive batched k optimization runs, we run 1000 trials to derive a proximal error rate. At a batch size of 60, we see that the failure rate is 2.4%. This number is relatively small, and thus we will propose a greedy batched estimation method that neglects the checks described in 3.5.4 and instead assumes that the batched estimation will converge. We will introduce an additional safeguard that corrects divergent trials that arise when we enter the incorrect likelihood mode. We will refer to this algorithm as the greedy batched algorithm.

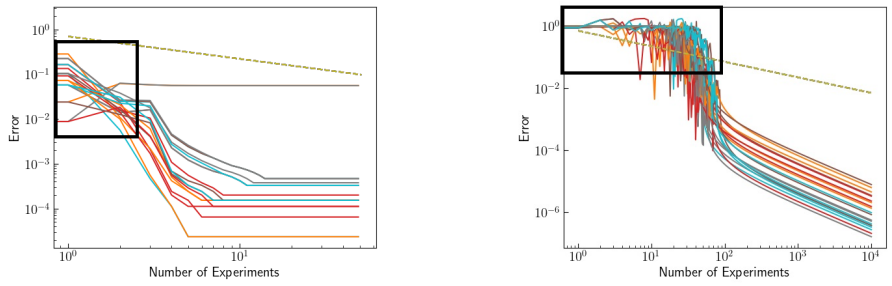
As the greedy batched algorithm runs, we will keep track of its progress by applying the z-score method described in section 3.4.2. After a series of trials, if we see that the greedy batched algorithm is entirely divergent, meaning its z-score does not improve over the course of iterations, then we will mark it to have failed. Upon failure, we will then re-use the measurements we took alongside the corresponding k values and re-run the Bayesian estimation procedure with the single cycle procedure described in 3.2. This enables us to still utilize the data we have taken, which is typically the most run-time expensive task (additional timing details will be discussed in section 3.6). This procedure is described in algorithm 3.

Algorithm 3 Greedy Batched

for i in Iterations / BatchSize **do**
  $k \leftarrow$ calculate_k(σ)
  Store k values in kValueArray
  measurements $\leftarrow$ take_measurements(BatchSize)
  Store measurements in MeasurementArray
  $\mu, \sigma \leftarrow$ Batched Bayesian posterior(μ, σ, k, measurements)
  Perform Z-score check(μ, σ)
  if Z-score check fails **then**
    for k in kValueArray, measurement in MeasurementArray **do**
      $\mu, \sigma \leftarrow$ Single cycle estimation (μ, σ, k, measurement)
    end for
    return μ, σ
  else
    continue
  end if
end for
return μ, σ

Doing so enables us to utilize the batched estimation procedure with k optimization without worry of divergence.

Examination of the greedy batched algorithm against the single cycle k optimized algorithm emphasizes the effect that batching (and subsequently the normality of the likelihood distribution) has on information gain in the initial trials of estimation. If we examine the two algorithms alongside one another, we see that batching enables us to consistently gain information in the earlier trials of estimation, as shown in figure 3.30. With batching, we have overcome the initial flatness described in section 3.1.1.



(a) (b)

Figure 3.30: Figure 3.30a highlights the information that is consistently gained in the initial cycles of the batched estimation procedure as all distributions are well approximated by normals. Figure 3.30b highlights the lack of information gain in the inital cycles of the single cycle estimation procedure when the wrapped normal is not a good approximation of the posterior distribution in the initial trials.

3.5.6 Data-efficient Greedy Batched

During actual runs of the estimation protocol, there will typically be a fixed posterior standard deviation that we desire to achieve. As a result, we will usually not run the estimation procedure for a fixed number of cycles, rather we will run it until we achieve the desired cutoff. When implemented in this framework, the use of the greedy batched protocol may result in a data wasting, as the fixed batch size causes the estimator to step down in relative error in large increments. This can be inefficient, as the final experimental shot may result in too many measurements that subsequently yield an excess of information gain and output a posterior standard deviation that is lower than the desired amount.

To combat this issue, we can predict the rate at which the Bayesian standard deviation decreases with each experimental shot, and with this knowledge, we can swap from the batched estimation method to the single shot method one batch before we converge to the desired Bayesian standard deviation. Doing so enables us to approach the desired Bayesian standard deviation with the exact number of measurements required, resulting in no wasted data.

To determine the swap point from batching to single-cycle, we can apply the Fisher information methodology described in section 3.5.1 to estimate our posterior standard deviation after each cycle.

For each cycle of Bayesian estimation, we will have some input standard deviation σ_1 from the prior. We will also have some likelihood standard deviation σ_2 that is a function of batch size, number of $|1\rangle$s measured, and k value. In the batched paradigm, we approximate our likelihood to be a normal distribution $\mathcal{N}(\Omega, \sigma)$, and we can write

$$\sigma = \sqrt{\frac{1}{I(\Omega)}} \tag{3.33}$$

$$= \sqrt{\frac{1}{E_\Omega[-\lambda''(\Omega|X)]}}. \tag{3.34}$$

$$\tag{3.35}$$

We will solve for $\lambda''(\Omega|X)$ for our likelihood function

$$L = \binom{n}{s} \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)^s \left(1 - \cos^2\left(\frac{k\Omega}{2} - \frac{m\pi}{2}\right)\right)^{n-s}, \tag{3.36}$$

finding that

$$\lambda''(\Omega|X) = \binom{n}{s} \frac{d^2}{d\Omega^2} \log\left(\cos\left(\frac{k\Omega}{2}\right)^{2(n-s)} \sin\left(\frac{k\Omega}{2}\right)^{2s}\right) \tag{3.37}$$

$$= \binom{n}{s}\left(-\frac{2s}{\Omega^2} + \frac{1}{6}(-3k^2n + 2k^2s) + \frac{\Omega^2}{120}(-15k^4n + 14k^4s) + \cdots\right). \tag{3.38}$$

Keeping second order terms, we have

$$\sigma_2 = \binom{n}{s} \sqrt{\frac{120\Omega^2}{240s - 20k^2\Omega^2(-3n+2s) - k^4\Omega^4(-15n+14s)}}. \tag{3.39}$$

We can then apply the method described in section 3.5.2 and write our posterior standard deviation σ_3 to be

$$\sigma_3 = \sqrt{\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2}}. \tag{3.40}$$

Note that the parameters n and s are present in equation 3.39, indicating that we need to take some samples to be able to solve for σ_2. Instead of physical measurements, we will instead draw simulated data by taking Ω to be the prior mean and applying the sampling function.

We demonstrate the results of this computation for a batch size of 60, observing the results shown in figure 3.31.

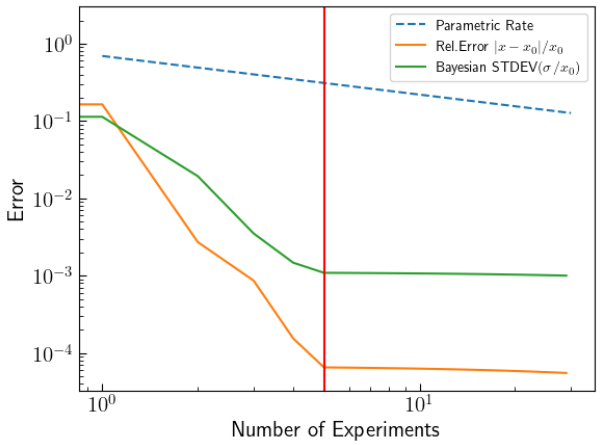


Figure 3.31: Demonstration of the Efficient Greedy Batched Algorithm, an estimation protocol that starts with the greedy batched algorithm and swaps to the single cycle algorithm one cycle prior to exceeding the desired standard deviation. The red vertical line denotes the experiment number at which the batched algorithm swaps to the single cycle algorithm.

We see that with this methodology, our estimation still converges to the right value and utilizes less measurements than an additional cycle of the batched protocol.

To test the robustness of the algorithm, we run 20 trials, displaying the results in figure 3.32

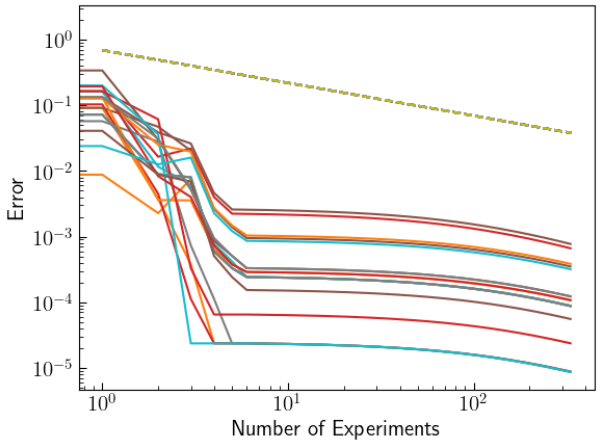


Figure 3.32: 20 runs of the EGB algorithm with a batch size of 60 and a posterior standard deviation cutoff of 10^{-3}. The dashed line denotes the parametric rate of convergence.

These trials demonstrate consistent convergence, confirming that the EGB algorithm (provided the behavior discussed in section 3.5.4 does not arise) consistently converges.

3.5.7 Batch to Single Cycle (B2S) Estimation

The B2S method begins by utilizing the batched Bayesian estimation procedure described in section 3.5. Beginning with this batching procedure enables us to overcome the initial lack of information gained by the algorithm in the initial 40 estimation iterations when the posterior is still poorly approximated with a wrapped normal distribution. However, purely batching can yield divergence due to the issues described in section 3.5.4, and thus upon completing the first 40 iterations, we then swap to the single-cycle procedure utilizing k optimization described in section 3.2. This enables the estimation to quickly converge to an accurate μ value while bypassing the early cycles of low information gain.

We run a similar set of experiments to ensure this methodology consistently converges and reaches the desired error rates. A set of 20 experimental runs is displayed in figure 3.33. All 20 runs converge, we successfully observe a decrease in relative error during initial stages of estimation, and we are able to quickly attain an error rate below 10^{-4}.

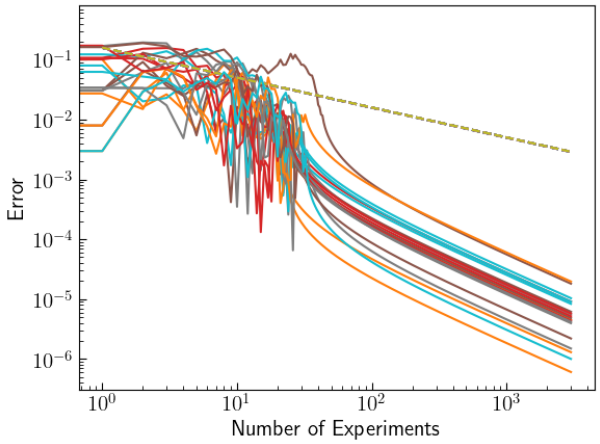


Figure 3.33: A batch of 20 estimation runs utilizing the batch to single cycle estimation methodology.

3.6 Benchmarks

This section compares the baseline, k-optimized, greedy batched, efficient greedy batched (EGB), and batch-to-single (B2S) estimation methodologies through a series of physical benchmarks. Each algorithm is allowed to run until a posterior standard deviation of 10^{-3} is reached. We will analyze the run-time of each algorithm through the physical procedures that take place throughout. Sampling the system and returning the state of the qubit corresponds to the physical process of state preparation and measurement and an increase in k correspond to an increase in circuit depth, as each additional k entails an additional rotational gate, as described in section 2.1.

We can map each of these actions to a set of timing metrics for an ion trap quantum computer realized by the then-Honeywell Quantum Solutions team (now Quantinuum) [4].

Operation	**Duration** (μs)
State Preparation	10
Measurement	120
Single Qubit Gate	5

Table 3.1: A table of time required to perform qubit operations that are applied during the estimation routine acquired from the Honeywell Quantum Solutions QCCD quantum computer [4].

We will set our desired Bayesian standard deviation at $\sigma = 10^{-3}$ and gather metrics for each methodology ten times, averaging over all trials. Note that the baseline algorithm converges very slowly towards the desired Bayesian standard deviation, thus we will cap it at 1000 iterations. We observe the results shown in figure 3.2.

Baseline

Metric Type	Metric Value	Operation Type	Total Operation Duration (ms)
Total data taken	1000	State preparation and measurement	130
Sum of all k values	1000	Single qubit gate	5

Total Time: 135ms

Single Cycle K-optimized

Metric Type	Metric Value	Operation Type	Total Operation Duration (ms)
Total data taken	208 ± 10	State preparation and measurement	27.04 ± 1.30
Sum of all k values	15787 ± 125	Single qubit gate	78.94 ± 0.63

Total Time: 105.98 ± 1.94ms

Greedy Batched

Metric Type	Metric Value	Operation Type	Total Operation Duration (ms)
Total data taken	300 ± 0	State preparation and measurement	39.00 ± 0
Sum of all k values	20268 ± 167	Single qubit gate	101.34 ± 0.84

Total Time: 140.34 ± 0.84ms

Efficient Greedy Batched

Metric Type	Metric Value	Operation Type	Total Operation Duration (ms)
Total data taken	266 ± 5	State preparation and measurement	34.58 ± 0.65
Sum of all k values	10892 ± 347	Single qubit gate	54.46 ± 0.17

Total Time: 89.04 ± 0.82ms

B2S

Metric Type	Metric Value	Operation Type	Total Operation Duration (ms)
Total data taken	277 ± 2	State preparation and measurement	36.01 ± 0.26
Sum of all k values	15936 ± 84	Single qubit gate	79.68 ± 0.42

Total Time: 115.69 ± 0.68ms

Table 3.2: A table of physical procedure timing metrics for baseline, k-optimized, greedy batched, efficient greedy batched, and B2S algorithms. The baseline case never converges to the desired Bayesian standard deviation and thus it always holds a high and fixed value.

We observe from the trials that the efficient greedy batched (EGB) is the most efficient algorithm with the overall lowest run-time, though the single cycle k-optimized and B2S algorithms follow closely. Outside of run-time, each of these algorithms have their own additional strengths and weaknesses. The EGB, though fast in run-time when analyzed through the physical procedures, requires the most computational machinery and complexity, with z-score checks being performed at each iteration and a possible swap to the single cycle algorithm if failure occurs, and thus may be inconsistent in certain cases. However, due to its primarily batched nature, the drift detection MLE test described in section 3.4.2 can be seamlessly integrated with no wasted data. This fact is true for all batched estimation methods.

The single cycle k-optimized, though lightweight and relatively fast, requires additional measurements to generate enough data to perform adequate MLE measurements. If one seeks to perform estimation in a lightweight fashion in an environment where drift is either not present or not significant, the single-cycle k-optimized methodology is recommended.

An alternative to both the EGB and single cycle k-optimized algorithm is the B2S estimation method. Though it is slightly slower than both, the B2S method enables some initial drift detection as the initial estimation iterations are batched, and also does not suffer from the error that arises from EGB, as the batching in this algorithm always occurs when the normal distribution adequately describes the binomial likelihood.

3.7 Summary

In this chapter, have investigated the case of the simplified single parameter Bayesian estimation of the Rabi frequency Ω. In section 3.1, we first began by demonstrating consistent convergence of a baseline single-cycle estimation algorithm that utilizes the wrapped normal distribution, then improved upon it by developing an adaptive likelihood-optimization algorithm in section 3.2 that yields a rate of convergence that far outclasses the parametric rate. In section 3.3, we also showed that a univariate normal substitution can be made when $\sigma \leq \frac{\pi}{4}$ to simplify the estimation procedure.

In section 3.4, we then explored various sources of error that may occur during the parameter estimation process, including parameter drift and SPAM error. To detect parameter drift, we developed a MLE regression method that is capable of detecting parameter drifts on the order of 10^{-7} per measurement. We have also demonstrated two estimation methods that are resilient to drift, one that involves exponentiation of the posterior standard deviation and one that involves sampling k. Both these methods enable the Bayesian estimation procedure to consistently converge when drift magnitudes on the order of 10^{-6} are present, albeit at the cost of higher relative error. We also showed that our estimation process is robust to SPAM error.

Additionally, in section 3.5, we presented a set of batched estimation methods that utilize multiple measurements per cycle rather than cycles of single measurements. This class of algorithms enable information gain at the very start of the estimation procedure and integrate well with the aforementioned MLE drift detection methodology.

Finally, in section 3.6 we performed a set of benchmarks on each of the algorithms we have developed, finding that the efficient greedy batched algorithm is the fastest to run and enables data-efficient drift detection, while the single-cycle k-optimized algorithm provides consistency at the cost of drift detection functionality. The B2S algorithm remedies the flaws of both these algorithms, albeit at the cost of a slight increase in run-time.

Chapter 4

Two-parameter Calibration

We will now examine the full two parameter estimation case where we desire to estimate both Ω and δ (note that these refer to their unitless variants Ω^* and δ^*). This case is quite challenging as there exists coupling between the two parameters, and thus errors in Ω can propagate to errors in δ and vice versa.

In this chapter, we will introduce an additional experiment that provides greater sensitivity to δ and discuss some challenges and solutions that come with two parameters.

4.1 Ramsey Interferometry and a New Likelihood

This section describes the theory behind Ramsey Interferometry and derives a new likelihood that is more sensitive to estimation of the detuning δ.

The estimation of a second parameter δ requires additional experimental machinery. In the first experiment, we made use of the approximation of laser pulses to an electron in a magnetic field to derive an expression for the Rabi frequency likelihood. The Rabi frequency likelihood is more sensitive to the Rabi frequency and not so much to the detuning. Here, we will discuss Ramsey interferometry, an experiment that will enable us to determine an expression for the a likelihood that is more sensitive to the detuning and enable a better estimate of our second unknown parameter δ.

Introduction to Ramsey Interferometry

To determine the likelihood for our detuning δ, we introduce a new experiment: Ramsey interferometry. Ramsey interferometry, when performed on a qubit in the ground state $|0\rangle$, involves the following procedure:

1. Apply a $\frac{\pi}{2}$ pulse to the qubit.
2. Allow the qubit to freely precess in the $x-y$ plane for some time t_p.
3. Apply another $\frac{\pi}{2}$ pulse.
4. Perform measurement of the qubit.

If we visualize this effect on the Bloch sphere, we have the pictoral representation in figure 4.1

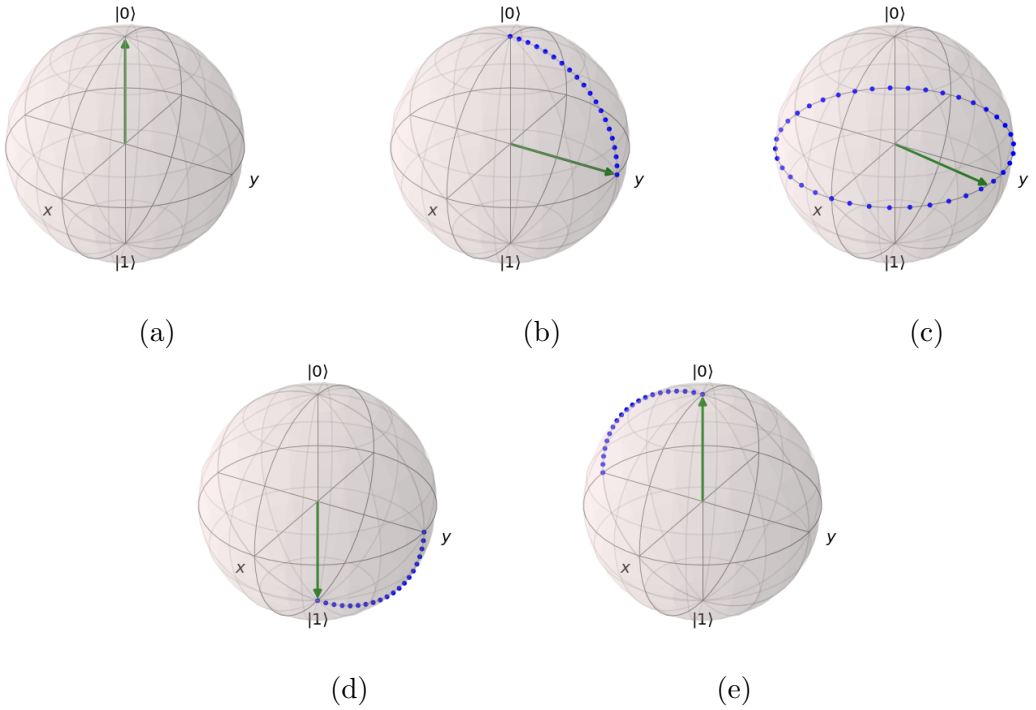


Figure 4.1: The change in the qubit at various stages in Ramsey interferometry are displayed. In figure 4.1e, we observe the prepared qubit in the $|0\rangle$ state. Figure 4.1b represents the qubit after a $\frac{\pi}{2}$ pulse has been applied. Figure 4.1c displays the free rotation of the qubit about the xy plane. Based upon the amount of free-rotation that occurs during this period, the final $\frac{\pi}{2}$ pulse will either send the qubit to the $|1\rangle$ state (or a range of transition probabilities that are biased towards the $|1\rangle$ state), as shown in figure 4.1d, or it can send the qubit back to the ground state $|0\rangle$ (or a range of transition probabilities that are biased towards the $|0\rangle$ state).

The qubit will precess in the xy plane as shown in 4.1c for an amount dependent on δ. The final $\frac{\pi}{2}$ pulse will thus yield a range of transition probabilities for the qubit as a function of δt_p, resulting in the well-known Ramsey fringes displayed in figure 4.2.

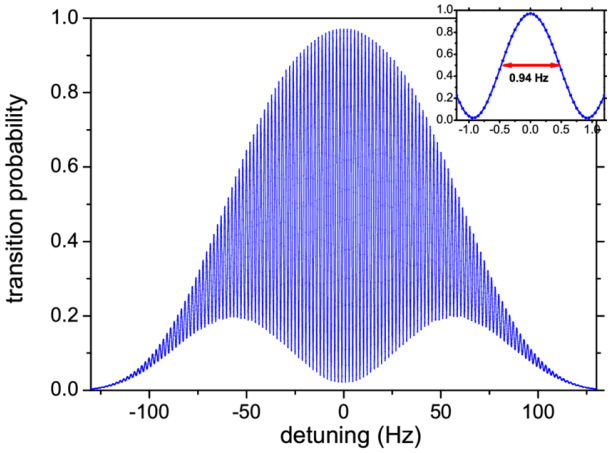


Figure 4.2: Figure from Bize et al. [70]. Experimental Ramsey fringes (transition probability as a function of the detuning) measured from a Cesium atomic clock. Note that the curved base and top of the fringes are due to decoherence present in an experimental system.

This leads to a probability function that defines the likelihood the qubit will go from $|0\rangle$ to $|1\rangle$ as a function of δ and t_p.

Deriving the Transition Probability

We will now derive the likelihood describing the Ramsey interferometry transition probability. Recall our Rabi frequency probability calculation from equation 2.36

$$P(|1\rangle) = \frac{\Omega^2 \sin^2\left(\frac{\alpha t}{2}\right)}{\alpha^2}. \tag{4.1}$$

A $\frac{\pi}{2}$ pulse places the qubit into a position that is equally likely to transition to the $|1\rangle$ state or remain in the $|0\rangle$ state. We can utilize this fact to find a representation for our time t, the length of time the laser pulse is applied for. If we set $P(|1\rangle) = \frac{1}{2}$, we find that

$$t = \frac{2}{\alpha} \arcsin\left(\frac{\alpha}{\Omega\sqrt{2}}\right). \tag{4.2}$$

To evolve some state with the $\frac{\pi}{2}$ pulse, we will write the rotational operator

$$\hat{R_{\frac{\pi}{2}}}(t) = e^{-i\hat{H}_I t} \tag{4.3}$$

as we are evolving the state of our qubit in the interaction picture. Recall that, from our earlier derivation of the likelihood with the Rabi frequency, we have

$$\hat{H}_I = \frac{\hbar}{2}\begin{pmatrix} -\delta & \Omega_{xy}e^{i\tilde{\phi}} \\ \Omega_{xy}e^{-i\tilde{\phi}} & \delta \end{pmatrix} \tag{4.4}$$

$$= \frac{\hbar}{2}(-\delta\sigma_z + e^{i\tilde{\phi}}\Omega\sigma_x). \tag{4.5}$$

Setting $\hbar = 1$ and omitting the phase factor $e^{i\tilde{\phi}}$, we can return our interaction Hamiltonian, yielding the $\frac{\pi}{2}$ rotational operator of

$$\hat{R_{\frac{\pi}{2}}}(t) = e^{i(\frac{\delta\sigma_z t}{2} - \frac{\Omega\sigma_x t2}{2})}. \tag{4.6}$$

From the results of our Rabi frequency derivation, we observe that

$$\hat{R}_{\frac{\pi}{2}}(t) = \begin{bmatrix} \cos\left(\frac{\alpha t}{2}\right) + \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & -\frac{i\Omega e^{i\phi}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \\ -\frac{i\Omega e^{-i\phi}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & \cos\left(\frac{\alpha t}{2}\right) - \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \end{bmatrix} \tag{4.7}$$

where $t = \frac{2}{\alpha}\arcsin\left(\frac{\alpha}{\Omega\sqrt{2}}\right)$.

We must now define a free rotation operator that represents rotation in the z axis of some time t_p. We will write this to be

$$\hat{R}_{free}(t_p) = e^{\frac{i\delta t_p}{2}\sigma_z} \tag{4.8}$$

and the full evolution operator for Ramsey interferometry can then be written to be

$$\hat{U} = \hat{R}_{\frac{\pi}{2}}(t)\hat{R}_{free}(t_p)\hat{R}_{\frac{\pi}{2}}(t). \tag{4.9}$$

Expanding this, we thus have:

$$\hat{U} = \begin{bmatrix} \cos\left(\frac{\alpha t}{2}\right) + \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & -\frac{i\Omega e^{i\phi}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \\ -\frac{i\Omega e^{-i\phi}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & \cos\left(\frac{\alpha t}{2}\right) - \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \end{bmatrix} \tag{4.10}$$

$$\begin{bmatrix} \cos\left(\frac{\alpha t}{2}\right) + \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha}e^{\frac{i\delta t}{2}} & -\frac{i\Omega e^{i\phi}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} \\ -\frac{i\Omega e^{-i\phi}\sin\left(\frac{\alpha t}{2}\right)}{\alpha} & \cos\left(\frac{\alpha t}{2}\right) - \frac{i\delta\sin\left(\frac{\alpha t}{2}\right)}{\alpha}e^{\frac{i\delta t}{2}} \end{bmatrix}. \tag{4.11}$$

We seek to solve for the probability of transitioning to the $|1\rangle$ state, which we can define to be

$$P_1 = |\langle 1|\,\hat{U}\,|0\rangle\,|^2. \tag{4.12}$$

This expression is equivalent to squaring the element in the first row, second column of $\hat{U}$, or $\hat{U}_{12}^2$. Performing this computation, we have

$$\hat{U}_{12} = \frac{-i\Omega}{\alpha}\left(\sin(\alpha t)\cos\left(\frac{\delta t_p}{2}\right) - 2\frac{\delta}{\alpha}\sin^2\left(\frac{\alpha t}{2}\right)\sin\left(\frac{\delta t_p}{2}\right)\right) \tag{4.13}$$

where we have converted exponentials to sinusoids. We can individually simplify some additional trignometric expressions in equation 4.13. We can write

$$\sin^2\left(\frac{\alpha t}{2}\right) = \sin^2\left(\frac{\sqrt{\Omega^2+\delta^2}}{2}\frac{2}{\Omega^2+\delta^2}\arcsin\left(\frac{\sqrt{\Omega^2+\delta^2}}{\Omega\sqrt{2}}\right)\right) \tag{4.14}$$

$$= \frac{\alpha^2}{2\Omega^2}. \tag{4.15}$$

We can also write

$$\sin(\alpha t) = \sin\left(\sqrt{\Omega^2+\delta^2}\frac{2}{\sqrt{\Omega^2+\delta^2}}\arcsin\left(\frac{\sqrt{\Omega^2+\delta^2}}{\Omega\sqrt{2}}\right)\right) \tag{4.16}$$

$$= \sin\left(2\arcsin\left(\frac{\sqrt{\Omega^2+\delta^2}}{\Omega\sqrt{2}}\right)\right). \tag{4.17}$$

Let us set some dummy variables $y = \arcsin\left(\frac{\sqrt{\Omega^2+\delta^2}}{\Omega\sqrt{2}}\right)$. We will rewrite our above expression for $\sin(\alpha t)$ to be

$$\sin(\alpha t) = \sin\left(2\arcsin\left(\frac{\sqrt{\Omega^2+\delta^2}}{\Omega\sqrt{2}}\right)\right) \tag{4.18}$$

$$= \sin(2y) \tag{4.19}$$

$$= 2\sin y\cos y \tag{4.20}$$

$$= 2\sin\left(y\sqrt{1-\sin^2 y}\right) \tag{4.21}$$

$$= 2\frac{\sqrt{\Omega^2+\delta^2}}{\Omega\sqrt{2}}\sqrt{1-\frac{\Omega^2+\delta^2}{2\Omega^2}} \tag{4.22}$$

$$= \frac{\sqrt{1-\frac{\delta^2}{\Omega^2}}\sqrt{\delta^2+\Omega^2}}{\Omega} \tag{4.23}$$

$$= \frac{\alpha}{\Omega}\sqrt{1-\frac{\delta^2}{\Omega^2}}. \tag{4.24}$$

We can now return these simplifications to our expression for $\hat{U}_{12}$, to write

$$\hat{U}_{12} = \frac{-i\Omega}{\alpha}\left(\sin(\alpha t)\cos\left(\frac{\delta t_p}{2}\right) - 2\frac{\delta}{\alpha}\sin^2\left(\frac{\alpha t}{2}\right)\sin\left(\frac{\delta t_p}{2}\right)\right) \tag{4.25}$$

$$= \frac{-i\Omega}{\alpha}\left(\frac{\alpha}{\Omega}\sqrt{1-\frac{\delta^2}{\Omega^2}}\cos\left(\frac{\delta t_p}{2}\right) - 2\frac{\delta}{\alpha}\frac{\alpha^2}{2\Omega^2}\sin\left(\frac{\delta t_p}{2}\right)\right) \tag{4.26}$$

$$= -i\left(\sqrt{1-\frac{\delta^2}{\Omega^2}}\cos\left(\frac{\delta t_p}{2}\right) - \frac{\delta}{\Omega}\sin\left(\frac{\delta t_p}{2}\right)\right). \tag{4.27}$$

To further simplify this expression, we designate some $\sin\xi = \frac{\delta}{\Omega}$. This enables us to rewrite

our expression for $\hat{U}_{12}$ to be

$$\hat{U}_{12} = -i\left(\sqrt{1-\sin^2\xi}\cos\left(\frac{\delta t_p}{2}\right) - \sin\xi\sin\left(\frac{\delta t_p}{2}\right)\right) \tag{4.28}$$

$$= -i\left(\cos\xi\cos\left(\frac{\delta t_p}{2}\right) - \sin\xi\sin\left(\frac{\delta t_p}{2}\right)\right) \tag{4.29}$$

$$= \frac{-i}{2}\left(\cos\left(\xi+\frac{\delta t_p}{2}\right) + \cos\left(\xi-\frac{\delta t_p}{2}\right) - \cos\left(\xi-\frac{\delta t_p}{2}\right) + \cos\left(\xi+\frac{\delta t_p}{2}\right)\right) \tag{4.30}$$

$$= -i\cos\left(\xi+\frac{\delta t_p}{2}\right). \tag{4.31}$$

Returning this to our expression for probability, we thus observe that

$$P(|1\rangle) = \hat{U}_{12}^2 \tag{4.32}$$

$$= \cos^2\left(\xi+\frac{\delta t_p}{2}\right) \tag{4.33}$$

$$= \cos^2\left(\arcsin\left(\frac{\delta}{\Omega}\right)+\frac{\delta t_p}{2}\right) \tag{4.34}$$

and

$$P(|0\rangle) = \sin^2\left(\arcsin\left(\frac{\delta}{\Omega}\right)+\frac{\delta t_p}{2}\right). \tag{4.35}$$

The goal of parameter estimation with the Ramsey likelihood for our experimental case is to minimize δ such that the laser frequency is as close to resonance with the ion frequency as possible. This is accomplished by tuning our driving frequency exactly to the transition frequency of the ion ω_0, a property of the ion that we do not precisely know. We can expand out the δ term in our likelihood, writing it to be $\delta = \omega - \omega_0$, where ω describes the applied frequency of our laser and ω_0 denotes the transition frequency of the ion. Since we do not have exact knowledge of ω_0, there will likely exist some detuning in the system, and an estimation of the detuning can inform us on the amount of adjustment to ω that must be made.

In later sections, for the sake of simplicity in the two parameter case, when we are performing parameter estimation on one parameter, we will refer to the other parameter as the opposing parameter (for example, if we are performing estimation on Ω, then the opposing parameter is δ and vice versa).

4.2 Likelihood Investigation

We have two probabilities, one for the estimation of the Rabi frequency Ω and one for the estimation of the detuning δ, which are defined to be

$$P_{\text{Rabi}}(|1\rangle) = \frac{\Omega^2 \sin^2\left(\frac{\alpha t}{2}\right)}{\alpha^2}, \tag{4.36}$$

$$P_{\text{Ramsey}}(|1\rangle) = \cos^2\left(\arcsin\left(\frac{\delta}{\Omega}\right) + \frac{\delta t_p}{2}\right). \tag{4.37}$$

We can transform these probabilities to likelihoods and perform an initial investigation of these likelihoods by visualizing the likelihood surfaces that arise for a given set of samples. This is done by incrementing by a phase of $\frac{m\pi}{2}$ where m defines the value measured from the circuit (a binary 1 or 0) and swapping our sines for cosines and vice-versa. Doing so yields the resultant likelihoods

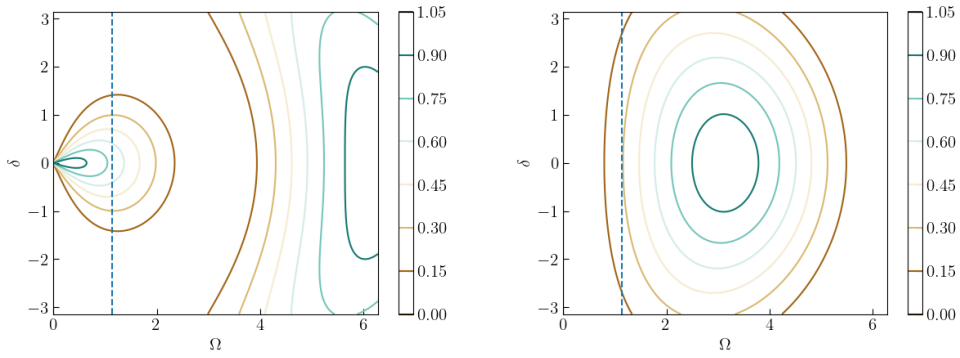
$$L_{\text{Rabi}}(|1\rangle) = \frac{\Omega^2}{\alpha^2}\cos^2\left(\frac{\alpha t}{2} - \frac{m\pi}{2}\right), \tag{4.38}$$

$$L_{\text{Ramsey}}(|1\rangle) = \sin^2\left(\arcsin\left(\frac{\delta}{\Omega}\right) + \frac{\delta t_p}{2} - \frac{m\pi}{2}\right). \tag{4.39}$$

where the Rabi likelihood has greater sensitivity to the Ω value and the Ramsey likelihood has greater sensitivity to the δ value.

To understand the general maxima and minima landscape of our likelihood functions, we will examine a set of contour plots with each. We will define some true Ω value and some true δ value, which will be used to generate samples. To generate the data points for the likelihood surface, we perform a grid search over a range of Ω and δ values from 0 to 2π and $-\frac{\pi}{2}$ to $\frac{\pi}{2}$, respectively.

For the Rabi likelihood, we first examine the case where $\delta = 0$, meaning we have found the precise ω value that yields zero detuning. We examine the instance in which a singular measurement is performed. These results are shown in figure 4.3.

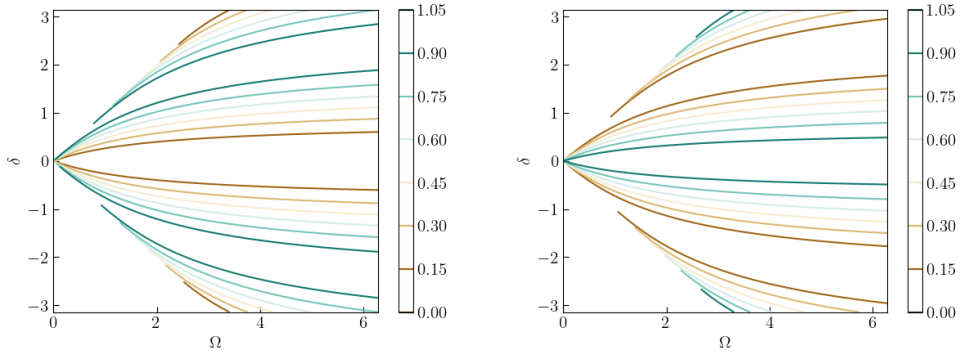


(a) (b)

Figure 4.3: The contour plots for the Rabi likelihood with a measurement of a $|0\rangle$ (figure 4.3a) or a measurement of a $|1\rangle$ (figure 4.3b) The blue line denotes the true Ω value.

From the contour plots, we observe that instead of an absolute point value that represents the true Ω values, we instead see a band of possible values for different Ω and δ values. This illustrates a central difficulty in the two parameter estimation problem: combinations of different Ω and δ values can yield the same sets of measurement data, and to estimate an Ω value with low error, we must perform the Ω estimation procedure with a δ value that is also close to the true δ value.

We perform the same tests on the Ramsey likelihood. In the Ramsey likelihood, $\delta = 0$ will always result in measurements of $|1\rangle$, thus to demonstrate the shape of the contour plot that arises for a measurement of $|0\rangle$, we pick $\Omega = 0.36\pi$ and $\delta = 0.3\pi$. We observe the plots displayed in figure 4.4 for measurement of a 0 or a 1.



(a) (b)

Figure 4.4: The contour plots for the Ramsey likelihood with a measurement of a 0 (figure 4.4a) or a measurement of a 1 (figure 4.4b).

We see that measuring a 1 or a 0 simply has the effect of shifting the Ramsey likelihood to different bands of δ values. As the detuning gets further and further from 0, the bands gradually move away from the origin in both positive and negative directions.

The contour plots shown in figure 4.4 also display that, unlike the single parameter estimation case, there are bands of possible Ω and δ values that yield the measured data. Similar to the Rabi contour plots shown in figure 4.3, an estimator that yields a δ value with low error must take in as input an Ω value with low error, or we risk outputting a δ value that still falls within the band but is far from the true value.

4.3 Baseline Tests

Having obtained likelihoods for each of the experiments, we can implement estimation methods with a set of simplifications to establish baseline results. In this case, will assume the following set of conditions

1. When estimating the Rabi frequency, we know what the true detuning value is.
2. When estimating the detuning, we know what the true Rabi frequency is.

Note that in an actual two-parameter estimation, this set of conditions is not true. When we first begin the estimation of Ω and take data from the Rabi experiment, our data will be biased by some unknown detuning, as we must first set our laser frequency to some bounded estimate that will likely not be close to the true ion transition frequency. As we perform more measurements and make more estimations, we will eventually get closer to our true δ value, though initially, there will be some error present. The same stipulation is true for estimating the detuning δ, in that our initial estimates of the Rabi frequency will likely not be precisely equal to the true Rabi frequency. These unknown parameters are what makes the two parameter estimation procedure difficult. The data taken is now biased by errors in the opposing parameter. However, before examining the full two parameter estimation case, we will explore some simplified cases by applying the above conditions and removing the aforementioned difficulties.

4.3.1 Maximum Likelihood Estimation

Prior to starting Bayesian estimation, we will first demonstrate that we can develop a maximum likelihood estimation (MLE) method that demonstrates that both Rabi and Ramsey likelihood functions converge to the correct values and validates our derived two-parameter likelihoods. MLE, a technique briefly discussed in section 3.4.2, is another method of parameter estimation. For some MLE model, the true parameter values are uncovered by searching over the parameter space and maximizing the likelihood that the model produces the observed data. We can thus extract each parameter through MLE by taking

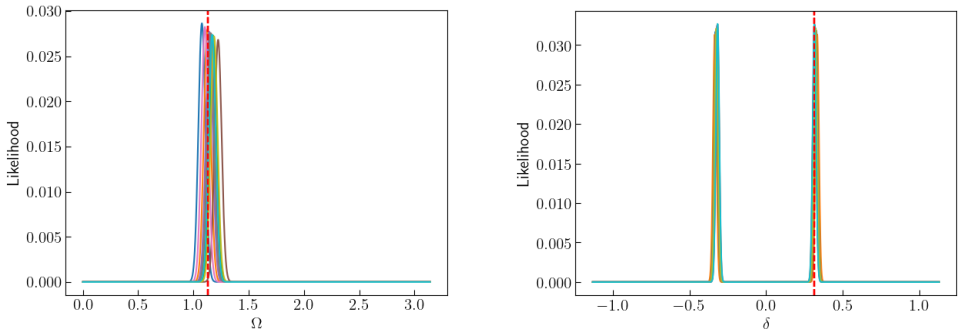
$$\hat{\Omega} = \arg\max L(\Omega|x_i), \tag{4.40}$$

$$\hat{\delta} = \arg\max L(\delta|x_i) \tag{4.41}$$

where x_i denotes the measurements we make.

A property of maximum likelihood estimators is consistency. As the number of samples approaches infinity, the estimator will always converge towards the true parameter value. MLE is thus an excellent baseline method to demonstrate that, for the given likelihoods, we will arrive at the true Ω and δ values.

To investigate the convergence of our MLEs, we perform 10 rounds of estimation with a sample size of 1000 and analyze the resultant distributions. Doing so yields the results shown in figure 4.5.



(a) (b)

Figure 4.5: Figure 4.5a displays the results of 10 MLE draws on the Rabi likelihood for $\Omega = 0.36\pi$ and $\delta = 0$. Figure 4.5b displays the results of 10 MLE draws on the Ramsey likelihood for $\delta = 0$ and $\Omega = 0.36\pi$. Both maximum likelihood estimators take 1000 samples. The dashed line denotes the true value of the parameter.

In both instances, MLE converges to approximately the correct value. One additional consideration is that the MLE of Ω is performed with the assumption that $\delta = 0$. However, most of the time, for a reasonable quantity of samples, we cannot calibrate a perfect δ value, and thus there will be a small quantity of detuning. We must ensure that Ω estimation will still converge to the correct value when this detuning is present in the system. From the contour plots displayed in section 4.2, we observe that there exists a range of δ values for which the correct Ω value can be found, and thus we expect that for a realistic δ value, Rabi frequency MLE will still converge to the correct value.

We perform a set of MLE on Ω while grid searching over a set of δ values from 0 to 0.2, observing the results shown in figure 4.6.

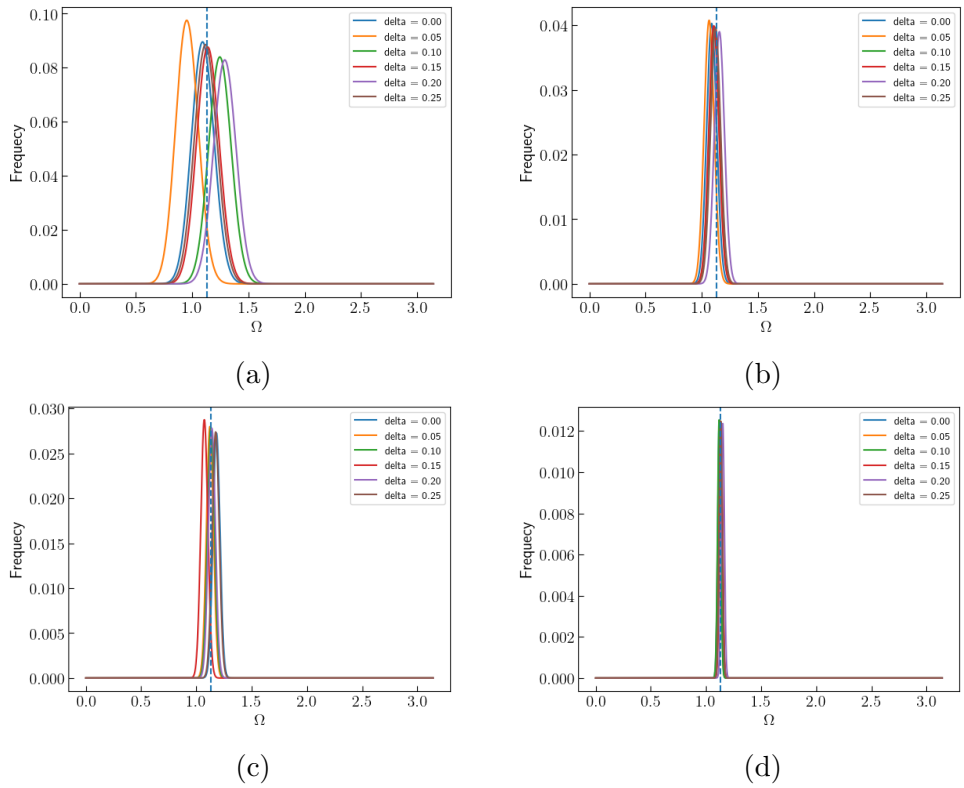


Figure 4.6: The results of Ω MLE with varied δ values at batch sizes of 100 (figure 4.6a), 500 (figure 4.6b), 1000 (figure 4.6c), and 5000 (figure 4.6d). The blue dashed line represents the true value of Ω

We observe that as we increase the number of samples, in accordance with MLE theory, we eventually converge to the true Ω value despite the variance in δ values.

Thus the Rabi likelihood still yields the correct results even when there is a degree of detuning in the system, as long as the value of the detuning is well known.

4.3.2 Baseline Bayesian Estimation: Detuning and Rabi Frequency

The Wrapped Normal Distribution

With the introduction of coupled parameters, we have the likelihood functions described in section 4.2

$$L_{\text{Rabi}} = \frac{\Omega^2}{\alpha^2}\cos^2\left(\frac{\alpha t}{2} - \frac{m\pi}{2}\right), \tag{4.42}$$

$$L_{\text{Ramsey}} = \sin^2\left(\arcsin\left(\frac{\delta}{\Omega}\right) + \frac{\delta t_p}{2} - \frac{m\pi}{2}\right). \tag{4.43}$$

We can perform the same procedure described in section 2.4.2, making use of the wrapped normal distribution. Our posterior will have mean and variance described by

$$\mu = \operatorname{atan2}(\mathrm{Im}(X), \mathrm{Re}(X)), \tag{4.44}$$

$$\sigma^2 = -\ln\big(|X|^2\big) \tag{4.45}$$

where X is defined by

$$X_{Rabi} = \frac{1}{N}\sum_{n=-\infty}^{\infty} c_n \int_0^{2\pi} e^{i(n+1)\Omega} f^{\circ}(\Omega; \mu_0, \sigma_0) d\Omega, \tag{4.46}$$

$$X_{Ramsey} = \frac{1}{M}\sum_{m=-\infty}^{\infty} c_m \int_0^{2\pi} e^{i(m+1)\delta} f^{\circ}(\delta; \mu_0, \sigma_0) d\delta \tag{4.47}$$

$$\tag{4.48}$$

where N and M are normalization constants we have integrated over the Fourier series and c_n and c_m denote some set of Fourier coefficients that correspond to each experiment.

To find the normalizing coefficients N and M, we must solve the following integrals

$$N = \int_0^{2\pi} L(\Omega|m) f^{\circ}(\Omega; \mu_0, \sigma_0) d\Omega, \tag{4.49}$$

$$M = \int_0^{2\pi} L(\delta|m) f^{\circ}(\delta; \mu_0, \sigma_0) d\delta \tag{4.50}$$

$$\tag{4.51}$$

However, unlike the calculations described in section 2.4.2, there is no analytical solution, and thus we must perform numerical integration at each Bayesian update step. This is an extremely costly procedure. If we desire to continue using the wrapped normal distribution for our single shot calculations, it is ideal to pre-compute a grid of numerical solutions for N and M. However, we will first investigate the results of utilizing the batched methodology described in section 3.5 and the substitution of the wrapped normal distribution with the univariate normal distribution described in section 3.3, as those frameworks are substantially less computationally intensive and easier to implement.

Univariate Normal-based Estimation

We will now examine the baseline case of Bayesian estimation of the detuning in the system. Similar to the procedure described in section 2.2, we can bound the maximal detuning and obtain some unitless representation of δ. With this bound, we can then perform estimation with the univariate normal distribution described in section 3.3.

Recalling the Bayesian estimation formula introduced in section 2.3.1, we see that a single

shot of the experiment will take the form

$$P(\delta|m) = \frac{L(\delta|m)P(\delta)}{P(m)} \tag{4.52}$$

$$= \frac{1}{N}\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{\delta-\mu}{\sigma}\right)^2}\sin^2\left(\arcsin\left(\frac{\delta}{\Omega}\right)+\frac{\delta t_p}{2}-\frac{m\pi}{2}\right). \tag{4.53}$$

where N is some normalizing factor. From this output, we will then fit the posterior $P(\delta|m)$ to a normal distribution and extract its posterior mean and posterior standard deviation. We run 500 iterations of this experiment to demonstrate the methodology, observing the results shown in figure 4.7.

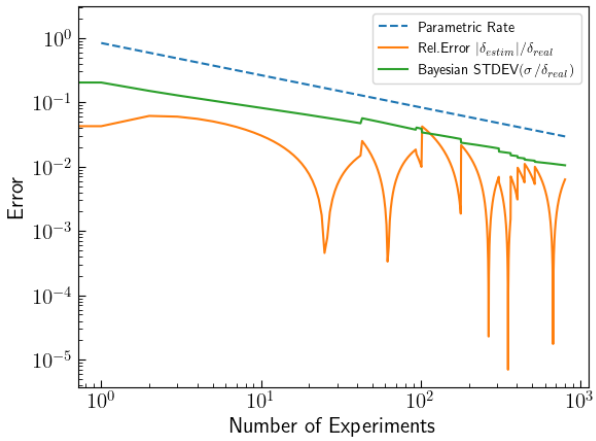


Figure 4.7: Parameter estimation of the detuning δ with a baseline Bayesian method.

We observe that the Bayesian method converges in accordance with the parametric rate, similar to the baseline Rabi frequency parameter estimation method discussed in section 3.1. To demonstrate that the parameter estimation method consistently converges, we run 20 trials of the parameter estimation. The results of this are displayed in 4.8.

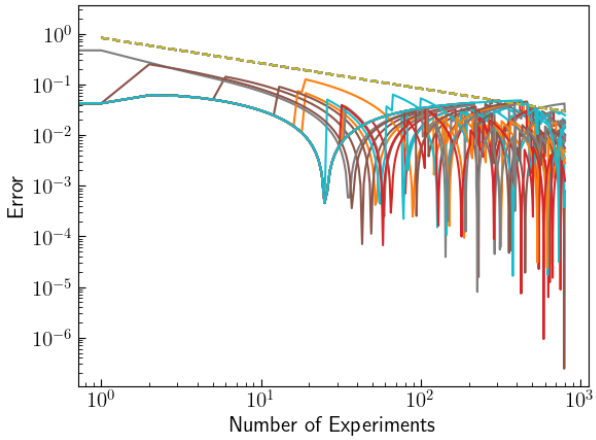


Figure 4.8: 20 runs of the δ estimation process with a lowered sampling rate (and thus an earlier error floor). The dashed line denotes the parametric rate.

We thus see that the Bayesian estimation procedure with the Ramsey likelihood consistently converges when the Rabi frequency value is well-known.

j optimization

Similar to the k optimization mentioned in section 3.2, the precession time t_p can be optimized over the experimental shots. Increased precession time leads to decreased uncertainty in δ and thus we can similarly adaptively optimize the precession time at various posterior standard deviations to improve our rate of convergence.

We will transform t_p to a unitless value by designating some fixed timestep τ_p and setting $j = \frac{t_p}{\tau_p}$. Increasing j will then lead to increases in precession time at fixed increments. When j is varied in the likelihood function, as demonstrated in figure 4.9, we once again observe a similar phenomena where the likelihood becomes increasingly multimodal. If we have sufficient confidence in our estimate of δ, meaning a low standard deviation, we can increase j and gain a tighter bound on the true value of our detuning.

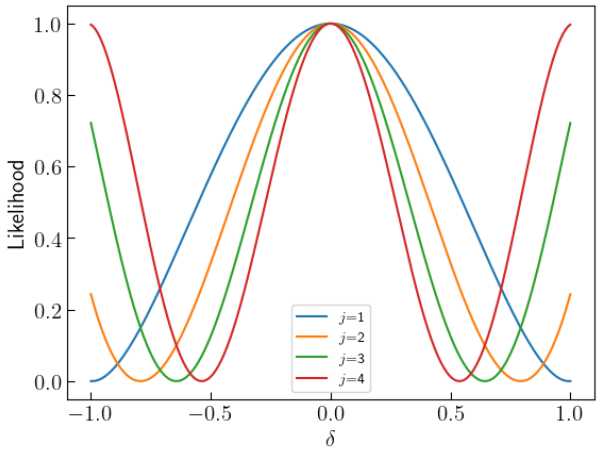


Figure 4.9: Ramsey likelihoods plotted for various precession times of the qubit following the initial application of the $\frac{\pi}{2}$ pulse.

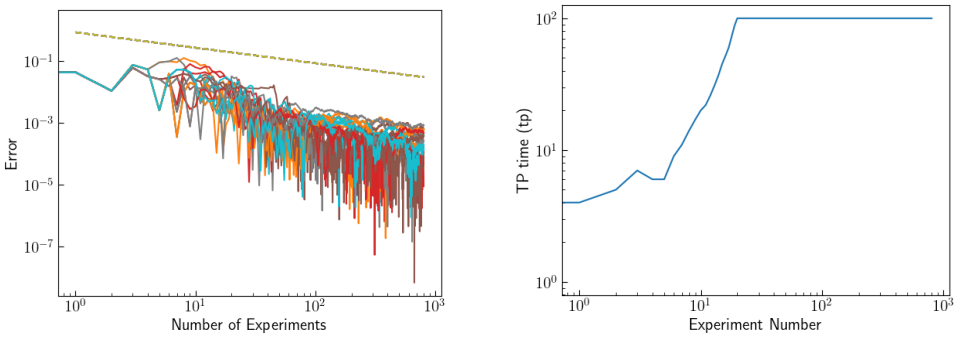
Like the Rabi likelihood, increasing the precession time leads to tightening standard deviations of the cosine likelihood. We will employ this to achieve a similar effect to the k-optimization demonstrated for the Rabi frequency.

In fact, since the Ramsey likelihood is just another $\cos^2$ likelihood, we can reuse the exact same ruleset we derived for k optimization in the Rabi case for our j optimization. We thus have

$$j(\sigma) = \begin{cases} \left\lfloor \frac{0.22507912}{\pi\sigma} \right\rceil & \sigma < 0.025258200269627846 \\ 4 & \sigma < 0.03269749744511768 \\ 3 & \sigma < 0.04684580115873045 \\ 2 & \sigma < 0.08688382635251184 \\ 1 & \text{otherwise.} \end{cases} \tag{4.54}$$

where σ is the standard deviation of the input prior distribution.

We implement this condition and re-run the estimation of the transition frequency with the simplification that the Ω value is known, yielding the results shown in figure 4.10.



(a) (b)

Figure 4.10: Figure 4.10a displays the Bayesian estimation of transition frequency with optimization of precession time. The change in precession time over the course of the experiments is shown in 4.10b. As the posterior standard deviation decreases, we increase the precession time and apply a thinner likelihood to the estimation procedure until the value ceiling for the precession time is reached.

This optimization enables us to estimate the detuning at a rate that once again exceeds the parametric rate.

Batching

We can also demonstrate that the batched k optimization methods demonstrated in section 3.5.4 similarly hold for the Ramsey likelihood and the δ estimation. We apply the same batching procedure as in section 3.5, taking multiple Bernoulli samples, generating a binomial distribution, and taking that to be a good approximation of a normal distribution. We can similarly apply the Fisher information to derive the output posterior. Upon integrating j optimization with the batching, we observe the results shown in figure 4.11.

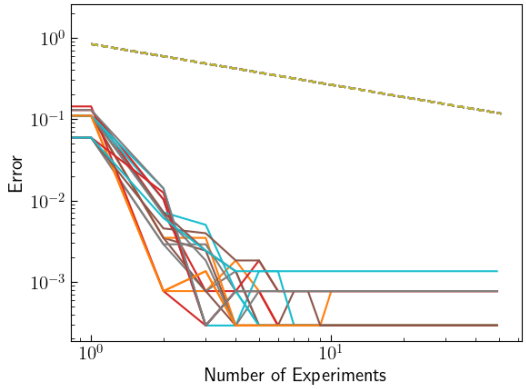


Figure 4.11: 20 runs of the batched j optimized δ estimation protocol. Batch size is set to 60 and the dashed line denotes the parametric rate.

We see that the batching with j optimization converge consistently and exceed the parametric rate.

4.4 Investigation of Coupling Errors

In the previous section, we demonstrated convergence of the estimators when one parameter is known and the other is unknown. In this section, we will now investigate the realistic experimental case where we initially have no knowledge of both parameters.

When we first begin the experiments, we can bound the transition frequency to a certain degree (and correspondingly, the detuning), but there will always exist some detuning in the system that we do not know. We can examine the probability of measuring a $|1\rangle$ or a $|0\rangle$ for the Rabi experiment as a function of the detuning. As detuning increases in magnitude, we observe the results shown in figure 4.12.

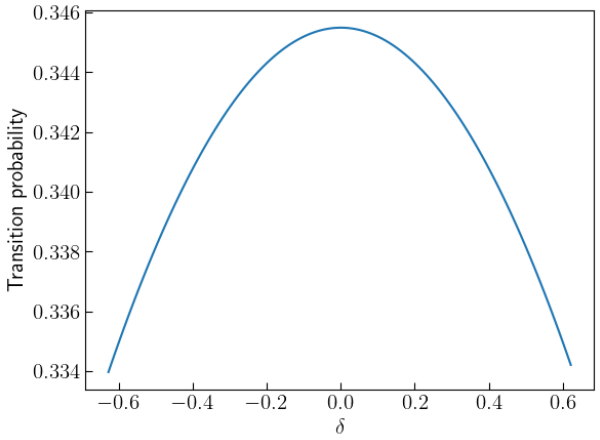


Figure 4.12: The transition probability as a function of δ for the Rabi experiment.

Equation 4.43 contains the parameter $\alpha = \sqrt{\Omega^2 + \delta^2}$ and thus additionally requires an estimate of δ. This increases the difficulty of the problem, as we initially start with zero knowledge of the detuning present in the system, and can only make an informed guess based off the pre-existing bound. From figure 4.12, we observe that differing values of δ can yield drastically different values for the transition probability. If we perform too many cycles of Ω estimation using a likelihood with an assumed δ that differs too much from the true δ of the system in which we pull samples from, then we may move towards the incorrect Ω value and waste data.

We can examine the effect various detuning error values have on the Rabi frequency estimation by performing a Rabi frequency estimation over a grid of detuning values. First, we examine the results of single cycle optimization performed with no k optimization, displayed in figure 4.13.

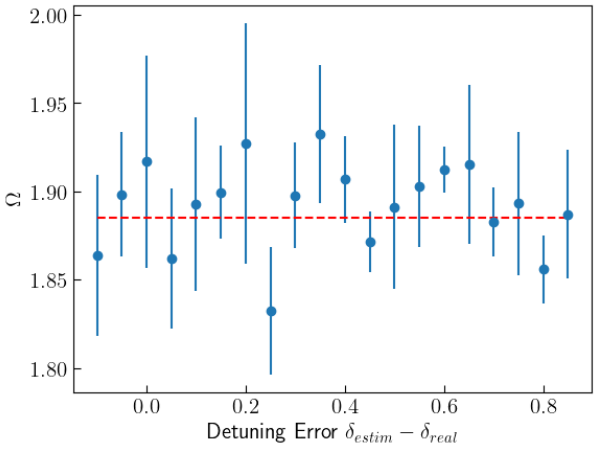


Figure 4.13: Variation in Rabi frequency estimation outcomes at different degrees of detuning estimation error. We perform the Bayesian estimation of the Rabi frequency for 500 iterations and average over four trials for each varied detuning error. The red dashed line denotes the true value of Ω

Though the results of this test appear to show an Ω value that is somewhat unbiased by the detuning errors when detuning errors are within a reasonable bound, we are not overly interested in performing the estimation procedure without k optimization. When we re-examine the effects of detuning error with k optimization, we observe the results shown in figure 4.14

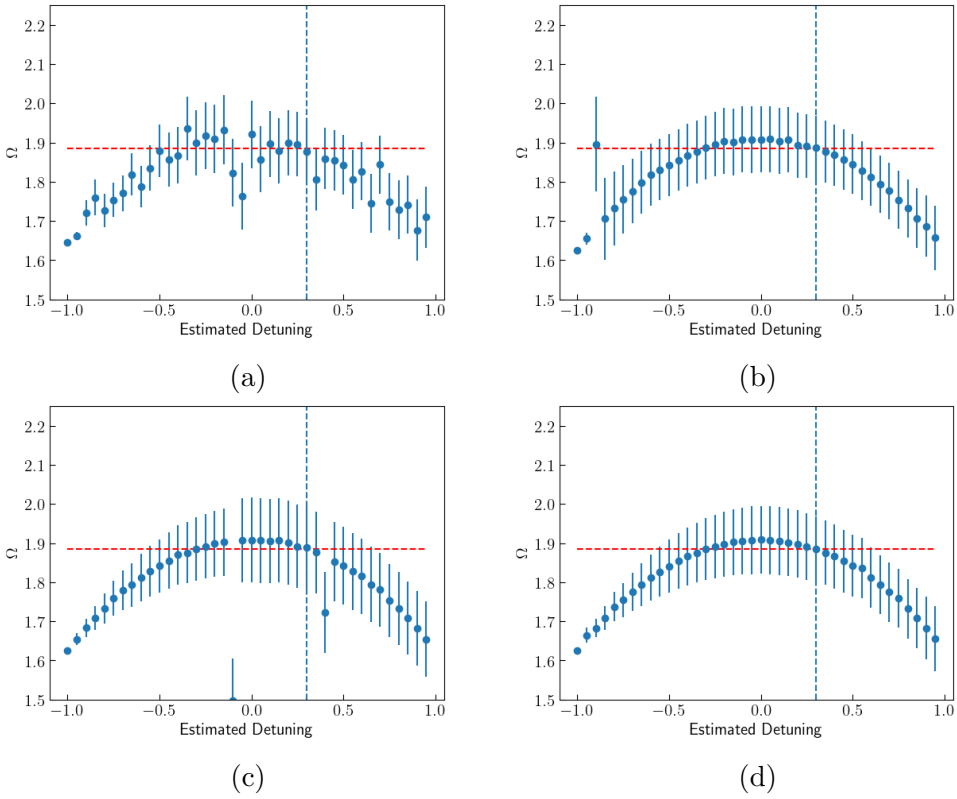


Figure 4.14: Variation in Rabi frequency with k optimization estimation outcomes at different degrees of detuning estimation error. Each data point is averaged over four trials. Figure 4.14a runs for 20 iterations, 4.14b runs for 60 iterations, figure 4.14c runs for 150 iterations, figure 4.14d runs for 500 iterations. The red dashed line denotes the true value of Ω and the blue dashed line denotes the true value of δ.

When k optimization is introduced, we begin to observe greater biasing effects, as the estimation tends to converge quicker and the increasing k values tend to amplify the error in the δ term (k multiplies $\alpha = \sqrt{\Omega^2 + \delta^2}$ in the likelihood function described in equation 4.43). We also observe that increasing the number of iterations of estimation that are performed does not yield any appreciable changes in the estimated Ω values, solely a tightened uncertainty.

This is also true for the Ramsey experiment and the estimation of the detuning. We can plot the transition probability from the Ramsey experiment as a function of Ω, observing the results shown in figure 4.15.

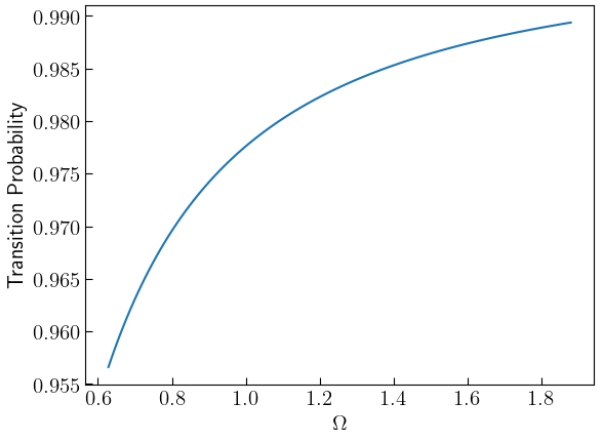
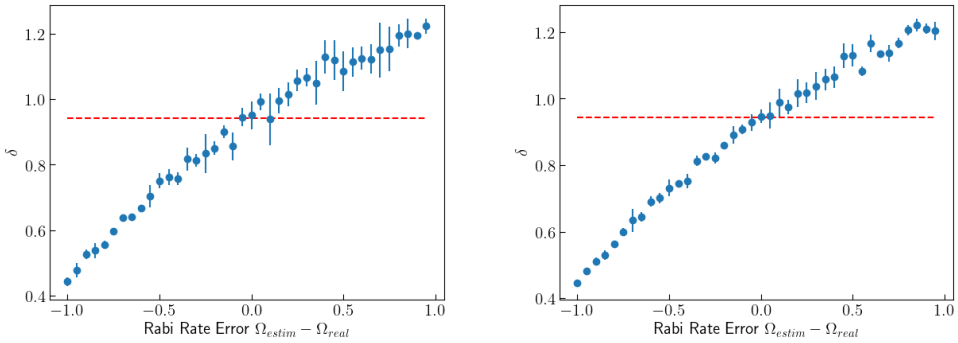


Figure 4.15: The transition probability as a function of Ω for the Ramsey experiment where $\delta = 0.1$.

When detuning is 0, the Ramsey transition probability will always yield an ion in the $|1\rangle$ state. Here, we observe that when there is $\delta = 0.1$ in the system (meaning the transition probability will not reach 1), higher Ω values push the transition probability towards 1, and thus inaccurately estimated Ω values will also introduce bias into the estimation of the detuning.

We again examine this effect in greater detail by performing estimations of the detuning over a grid of Ω error magnitudes. When we do not perform j optimization during the estimation procedure, we observe the results shown in figure 4.16.



(a) (b)

Figure 4.16: Variation in detuning estimation without k optimization estimation outcomes at different degrees of Rabi frequency estimation error. Each data point is averaged over four trials. Figure 4.16a runs for 60 iterations and 4.16b runs for 150 iterations. The red dashed line denotes the true value of δ.

We see that errors in the Rabi frequency have a significant effect on the estimate of the detuning. However, we observe that the number of iterations does not affect the magnitude

of bias in detuning.

We then perform the same series of tests, this time adding j-optimization to our δ estimation procedure. This yields the results shown in figure 4.17

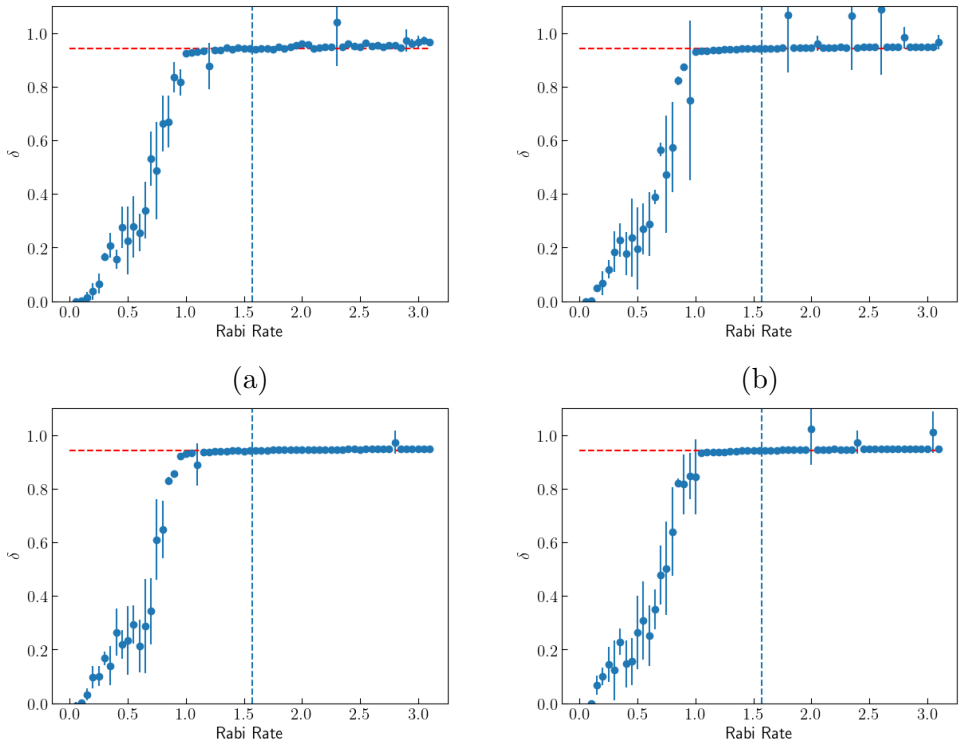


Figure 4.17: Variation in detuning estimation with j optimization estimation outcomes at different degrees of Rabi frequency estimation error. Each data point is averaged over four trials. Figure 4.17a runs for 30 iterations, 4.17b, 4.17c runs for 150 iterations, and 4.17d runs for 300 iterations. The red dashed line denotes the true value of δ and the blue dashed line denotes the true value of Ω.

At first glance, Rabi frequency errors of negative values ($\Omega_{estim} - \Omega_{true} < 0$) yield the greatest biasing of the output detuning, whereas higher valued errors ($\Omega_{estim} - \Omega_{true} > 0$) have a comparatively smaller effect. This can be explained through an examination of the Ramsey likelihood, equation 4.43. Ω error is present in the arcsine term within the $\sin^2$ function, and as we increase j, the effect of the erroneous term grows smaller in comparison to the $\frac{\delta j}{2}$ term, thus de-emphasizing the Ω error. The effect is not entirely negligible however. When we increase the resolution of our y-axis for the plots resultant from the trials shown in figure 4.17, we see the results shown in figure 4.18.

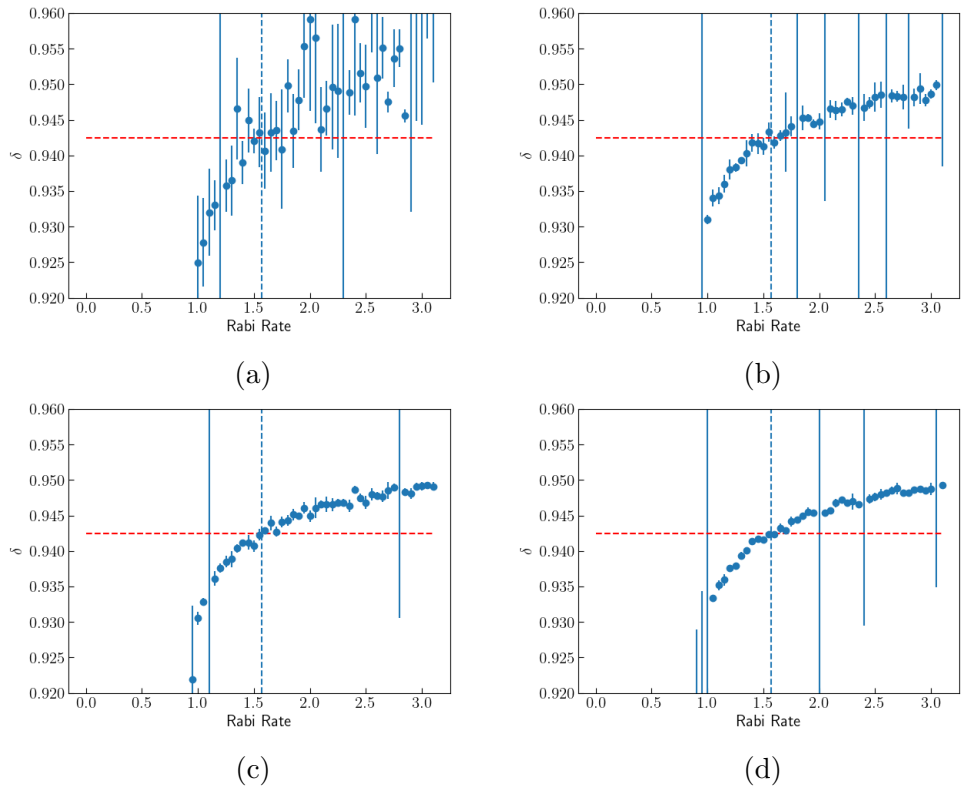


Figure 4.18: Error in detuning estimation with j optimization at different degrees of Rabi frequency estimation error bound from $\delta = [0.92, 0.96]$. Each data point is averaged over four trials. Figure 4.18a runs for 30 iterations, 4.18b runs for 60 iterations, 4.18c runs for 150 iterations, and 4.18d runs for 300 iterations. The red dashed line denotes the true value of δ and the blue dashed line denotes the true value of Ω..

At higher values of Ω error, the resultant biasing causes an overestimation of the detuning, though at a substantially lower magnitude than when Ω is lower. This suggests that overshooting the value of Ω during estimation is less detrimental than undershooting.

We also note that the Ω estimation procedure is far more sensitive to δ error than δ estimation to Ω error. This may be useful in determining the amount of iterations we want to run with each experiment before swapping to the other. We will explore this in greater detail in section 4.5.2.

4.5 Two Unknown Parameters: Simplified Tests

This section will investigate a set of two-parameter estimation algorithms that involve back-and-forth estimates of Ω and δ. These experiments will no longer contain the simplification described in section 4.3 and will incorporate error in parameter estimates, as described in section 4.4. However, all of the below algorithms will make the central simplification that

the opposing input parameter estimate is simply a value, and not a probability distribution with some mean and uncertainty.

Let us imagine a typical estimation protocol. We begin by estimating Ω for N trials, ending with some posterior probability distribution describing our predicted $\hat{\Omega}$. When we then perform the estimation of δ in the next step, instead of inserting the probability distribution describing $\hat{\Omega}$, we will simply insert the maximum a posteriori (the mode of the posterior $\hat{\Omega}$ distribution) into our δ likelihood, removing all information about the uncertainty of the $\hat{\Omega}$ posterior and assuming the $\hat{\Omega}$ value to be the "true" value. Though this simplification may not accurately quantify the true uncertainty in each posterior (and as a result cause issues with the k and j optimization algorithms), it greatly reduces the complexity of the mathematical and computational machinery required to investigate the problem and can still yield convergent results.

4.5.1 Symmetrical Single Cycle Investigation

Single Cycle Back and Forth (SCBF)

We will first investigate a single cycle procedure in which we perform a fixed number of cycles with one experiment, obtain an estimation of the desired parameter, and return the parameter estimate to the next cycle of estimation with a different experiment. This procedure is described in algorithm 4.

Algorithm 4 Single Cycle Back-and-Forth Protocol

$\Omega_1 \leftarrow$ RabiBayesianEstimation($\delta = 0$, iteration˙number $= n$)
$\delta_1 \leftarrow$ RamseyBayesianEstimation($\Omega = \Omega_1$, iteration˙number $= n$)
$\Omega_2 \leftarrow$ RabiBayesianEstimation($\delta = \delta_1$, iteration˙number $= n$)
$\delta_2 \leftarrow$ RamseyBayesianEstimation($\Omega = \Omega_2$, iteration˙number $= n$)
...

A single trial of this protocol is displayed in figure 4.19. We observe that both δ and Ω converge to a desirable relative error of 10^{-4}.

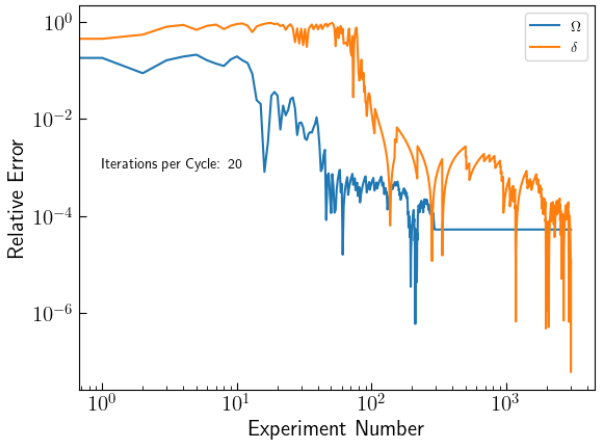
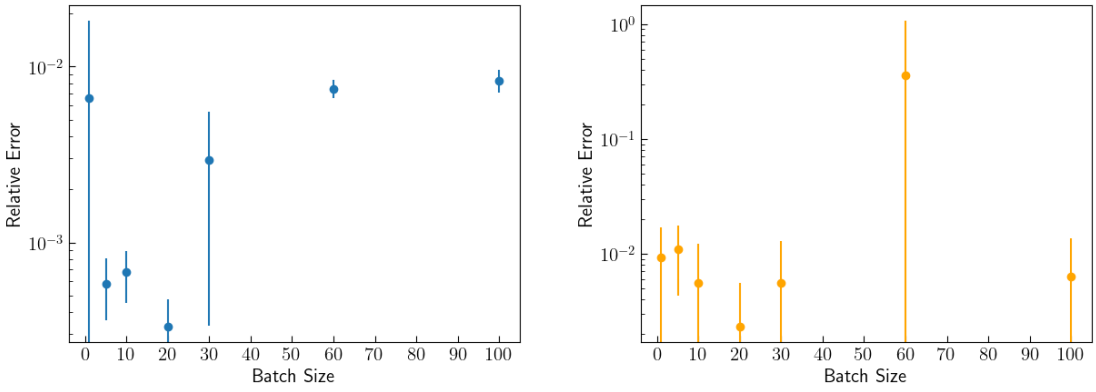


Figure 4.19: The relative error over experiment number for Ω and δ for the SCBF estimation protocol with 20 iterations per cycle.

We want to find the optimal iteration number that will yield the lowest relative error for both parameters, and thus we run the SCBF protocol for a multitude of single-cycle iteration sizes averaged over four trials, yielding the results shown in figure 4.20. Additional figures displaying individual trials for this protocol can be found in appendix B.2.



(a) (b)

Figure 4.20: Figure 4.20a displays the average relative error following SCBF for Ω at various numbers of single-cycle iterations. Figure 4.20b displays the average relative error following SCBF for δ at various numbers of single-cycle iterations. In both figures, each datapoint is averaged over four trials and in all experiments, a total of 3000 samples are taken. The batch size corresponds to the number of iterations performed per cycle.

We observe that an iteration number of 20 appears to be optimal for yielding an Ω and δ value with low relative error, though a wide array of values from 5 to 20 yield acceptable relative error. When the iteration number becomes too large, we observe that the relative

error does not decrease sufficiently. This is due to an excess of estimation trials performed with too high a degree of error in the opposing estimation parameter (too many iterations of Ω estimation performed with high δ error, too many iterations of δ estimation performed with high Ω error).

We examine this specific setting in greater detail, running and averaging 20 trials with an iteration number of 20, yielding the results shown in figure 4.21

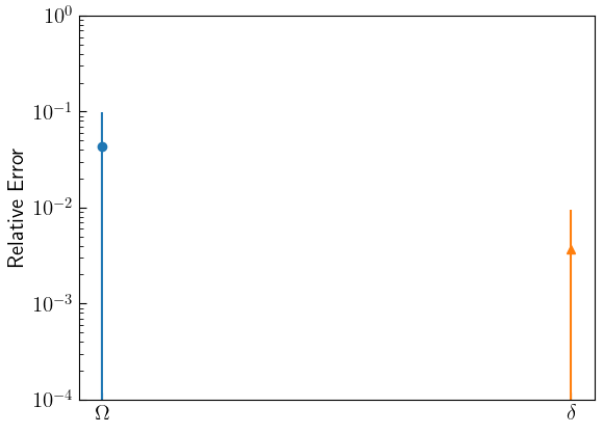
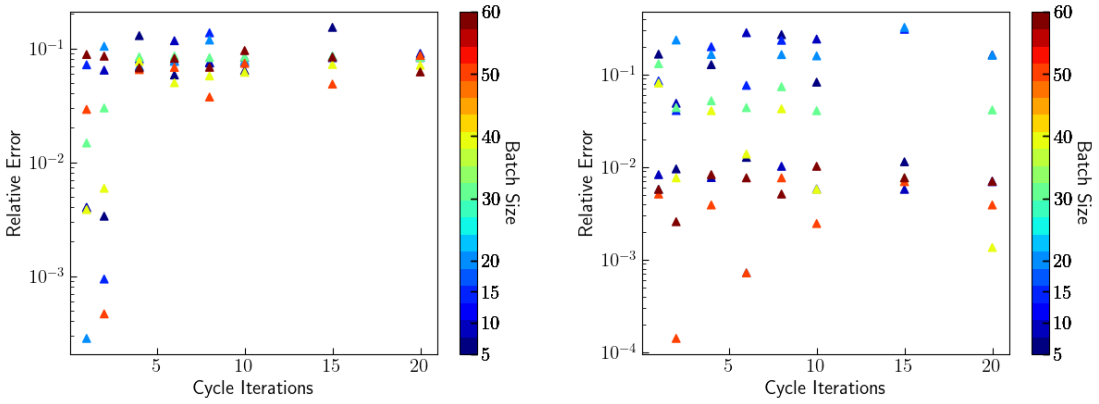


Figure 4.21: 20 runs of the SCBF algorithm with an iteration number of 20.

An increase in trial numbers yields an increasingly wide error bar, as trials begin to diverge. We categorize all final Ω and δ relative errors that exceed 10^{-2} to be failures, finding a failure rate of 20% for δ and 40% for Ω. These results are sub-optimal, and indicate that our estimator slips modes frequently, likely due to the simplification noted at the beginning of section 4.5 or due to biased estimates generated from estimation cycles with high error in the opposing parameter.

Batched Back and Forth (BBF)

We perform a similar test with the batched Bayesian estimation methods that we have developed, this time iterating over various batch sizes and number of batched iterations per cycle. Upon doing so, we observe the results shown in figure 4.22.



(a) (b)

Figure 4.22: Figure 4.22a displays the average relative error following BBF for Ω at various numbers of cycle iterations . Figure 4.22b displays the average relative error following BBF for δ at various numbers of cycle iterations. In both figures, each datapoint is averaged over five trials and in all experiments, a total of 3000 samples are taken.

We observe that the batch size of 50 with a single cycle iteration number of 2 (red triangle) yields the optimal results, and thus we examine it with finer granularity. We plot all trials run with a batch size of 50 and examine the error in these trials in figure 4.23.

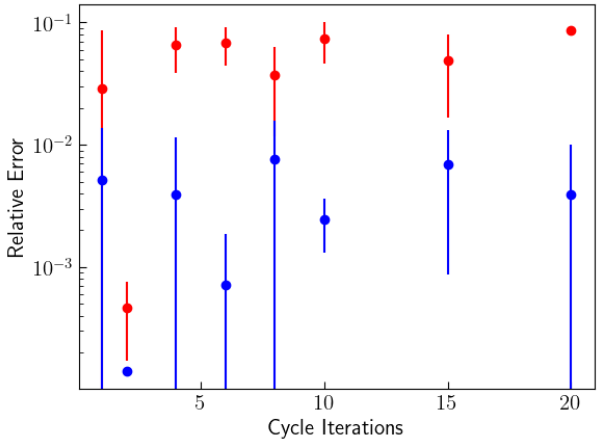


Figure 4.23: The relative error in Ω (blue) and δ (red) for experiments executed with a batch size of 50 from figure 4.22.

The standard deviations demonstrated for the trials completed with a batch size of 50 and a cycle iteration of 2 are fairly tight, indicating that it is the ideal iteration setting for the batched estimation.

We also investigate these settings in greater detail, running 20 trials with a batch size of 50 and a cycle iteration of 2, yielding the results show in 4.24.

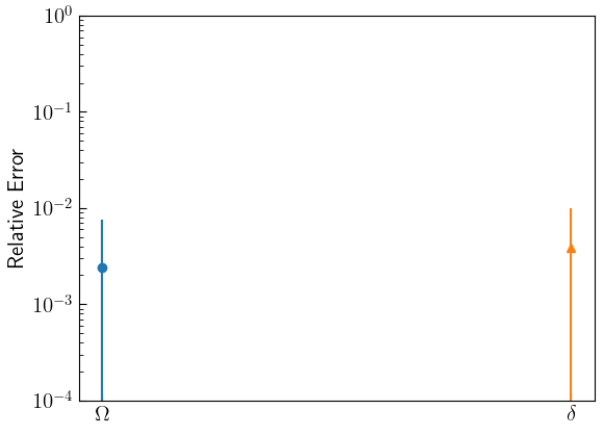


Figure 4.24: 20 trials of batched back and forth trials with a batch size of 50 and an iteration number of 2.

Though we observe generally better results than the SCBF algorithm shown in figure 4.21, there is still a wide variance in the relative error of our output trials. Defining failure to be any result that ends with a relative error greater than 10^{-2}, we have a 5% failure rate for Ω and a 15% failure rate of δ.

4.5.2 Asymmetric Fisher Information-based Estimation

To ensure a similar relative error for each parameter at the given estimation cycle, we desire for the posterior standard deviations of our parameters to be approximately equal, or

$$\sigma_\Omega \approx \sigma_\delta. \tag{4.55}$$

The two estimation procedures gain information at differing rates and thus the only method to ensure that their standard deviations are approximately equal is to estimate each quantity for an asymmetrical amount of iterations or batch sizes. We shall write the ideal ratio of batch sizes or iterations, α to be

$$\alpha = \frac{\sigma_\Omega}{\sigma_\delta}. \tag{4.56}$$

Since the standard deviation of a normal distribution can be described by the Fisher information through the equation

$$\sigma = \sqrt{\frac{1}{I(\mu)}} \tag{4.57}$$

we will focus on computing α through the Fisher information.

We observe that we can solve for α by computing the posterior standard deviation prior to the start of the iteration for each Rabi and Ramsey estimation, and then taking the ratio of both. The complexity for the two parameter case is increased due to the additional variables in the likelihood, though batching still yields a normal distribution. The prior distribution

for both Rabi and Ramsey experiments are normal distributions with a given μ and σ, and thus we will focus on the likelihoods for each experiment. Performing the same calculations as described in 3.5.6, we find that

$$\begin{aligned}\lambda''(\Omega|X) = &-\frac{4n^2}{\Omega^2} \\ &+ \frac{4n\left(\delta^2 kn - \delta^2 ks + 2\sqrt{\delta^2}n\cot\left[\frac{\sqrt{\delta^2}k}{2}\right] - \delta^2 ks\cot\left[\frac{\sqrt{\delta^2}k}{2}\right]^2\right)\tan\left[\frac{\sqrt{\delta^2}k}{2}\right]}{\delta^3} \\ &+ \cdots\end{aligned} \tag{4.58}$$

$$\begin{aligned}\lambda''(\delta|X) = &\frac{-2n+2s}{\delta^2} - \frac{24s + j\Omega(12 + J\Omega(6 + j\Omega))(n+2s)}{6\left(\Omega^2(2+j\Omega)\right)} \\ &- \big((jn\Omega(720 + j\Omega(720 + j\Omega(240 + j\Omega(60 + j\Omega(12 + j\Omega))))) \\ &+ 2(1440 + j\Omega(2520 + j\Omega(2160 + j\Omega(1200 + 7j\Omega(60 \\ &+ j\Omega(12 + j\Omega))))))s)\delta^2\big) / \big(120\left(\Omega^4(2+j\Omega)^2\right)\big) + \cdots\end{aligned} \tag{4.59}$$

where n denotes the number of trials, s denotes the number of 1s measured, k denotes the number of gates applied, and j denotes the unitless precession time. This expression's relative complexity at the second order is highly detrimental for our use case, as the second moment of the of the normal distribution represents the variance and thus we must keep second order terms in our Taylor series. Instead of this analytical computation, we shall re-use the numerical method of differentiation described in section 3.5.1 where

$$\lambda''(a) = \frac{\lambda(a+h) + \lambda(a-h) - 2\lambda(a)}{h^2}. \tag{4.60}$$

where in our case a represents the mean of normal distribution and h represents some point a small distance away from the mean. Furthermore, in the single cycle case, we can also omit any pre-calculation of the posterior standard deviation and instead run iterations indefinitely until $\sigma_{Ramsey} \approx \sigma_{Rabi}$.

4.5.3 Adaptive SCBF

In the single cycle case, we must first perform trials of estimation so that the posterior is well-approximated by a normal distribution (this problem is discussed in detail in section 3.5.3). Upon doing so, we can implement two adaptive asymmetrical protocol for the SCBF, shown in figure 4.25 and figure 4.29. We will explore these two techniques in greater detail in their respective sections.

Consecutive Adaptive Estimation (CAE)

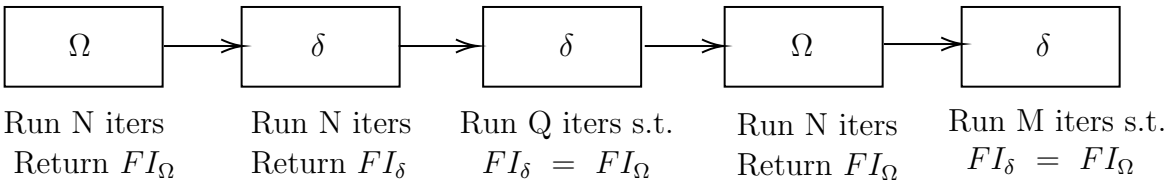
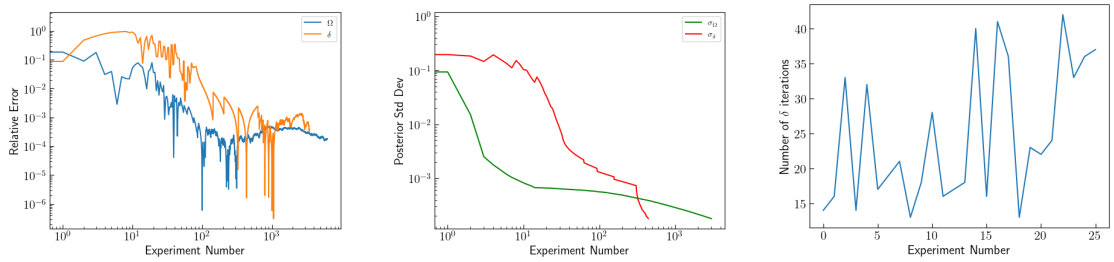


Figure 4.25: Diagrammatic visualization of adaptive SCBF protocol with double δ estimation. Each box represents the estimation cycle being run.

We begin with Ω estimation, first performing N iterations. We then return to δ estimation and similarly perform N iterations. We then run M additional iterations of δ estimation until the Fisher information for δ is approximately the same as Ω (note that in this example, we assume that Ω estimation gains information at a faster rate than δ estimation. In the case where δ estimation gains more information than Ω, we would instead run M additional iterations on Ω). This procedure is then repeated until the desired posterior standard deviations are reached. The primary advantage of this protocol is that the difference in Fisher information, and thus the standard deviation, of Ω and δ is minimized. By consecutively estimating δ after the first Ω estimation, the later cycles of Ω and δ estimation yield $\sigma_\Omega \approx \sigma_\delta$.

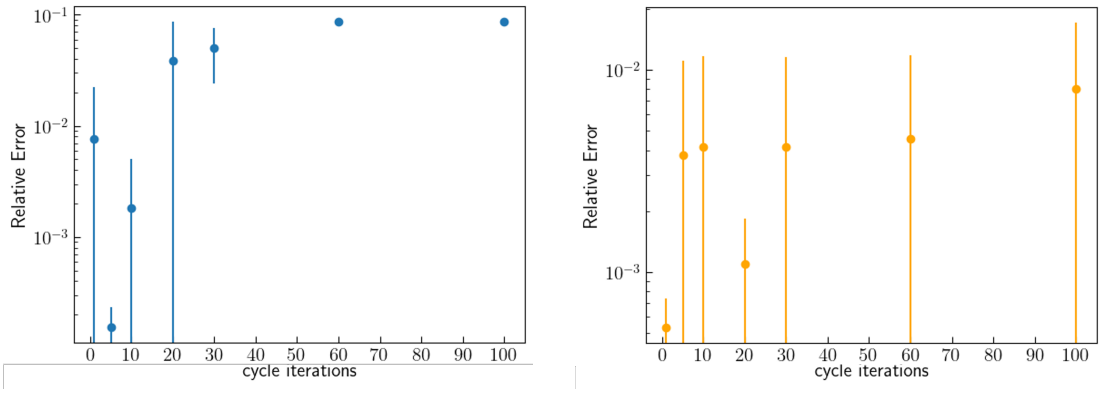
We run a sample estimation with $N = 20$, yielding the output shown in figure 4.26



(a) (b) (c)

Figure 4.26: Figure 4.26a demonstrates a sample result from an experimental run of the consecutive adaptive estimation. Figure 4.26b shows the posterior standard deviation for this trial (note that the final defect in the red curve is a result of numerical instability due to sampling limits). Figure 4.26c displays the adaptive changes in iteration number for δ.

To better test the various N values, we search over a grid for $N = [1, 5, 10, 15, 20, 30, 60, 100]$, averaging over five trials for each N value. We observe the results shown in figure 4.27.



(a) (b)

Figure 4.27: Relative errors for a grid of iteration numbers with the Fisher information adaptive optimization algorithm. Figure 4.27a displays the relative errors for Ω and figure 4.27b displays the relative errors for δ.

Though we observe generally favorable results, we see that we do not attain as low a relative error as the results shown during the symmetrical trials in section 4.5.1. We observe that for our grid of iteration numbers, $N = 5$ yields the best results, though the bound on δ is still relatively wide. To verify this result, we run an additional 20 trials with $N = 5$, yielding the results shown in figure 4.28.

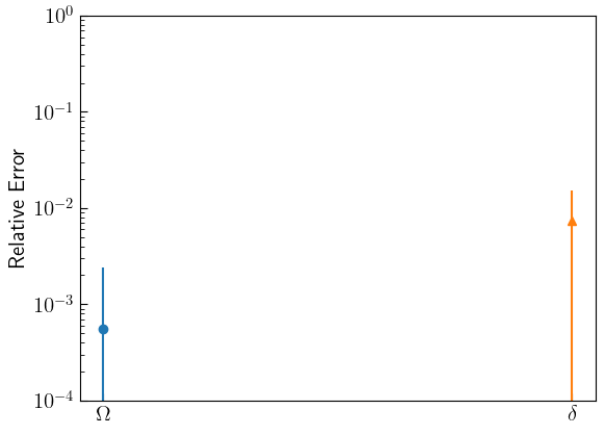


Figure 4.28: 20 trials of batched back and forth trials with an iteration number of 5.

For this set of trials, we observe a 0% failure rate for Ω and a 35% failure rate for δ. Though Ω is robust, the issue of slipping into incorrect likelihood modes is still prevalent for δ, hurting the overall consistency of the estimator. However, when compared to the symmetrical single iteration (figure 4.21) and symmetrical batched (figure 4.24) methods, we observe a tighter bound and a demonstrable improvement in the distribution of relative errors.

Non-Consecutive Adaptive Estimation (NCAE)

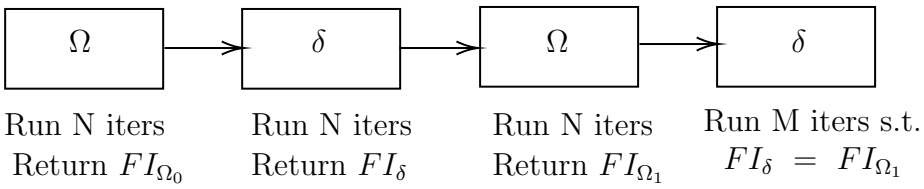
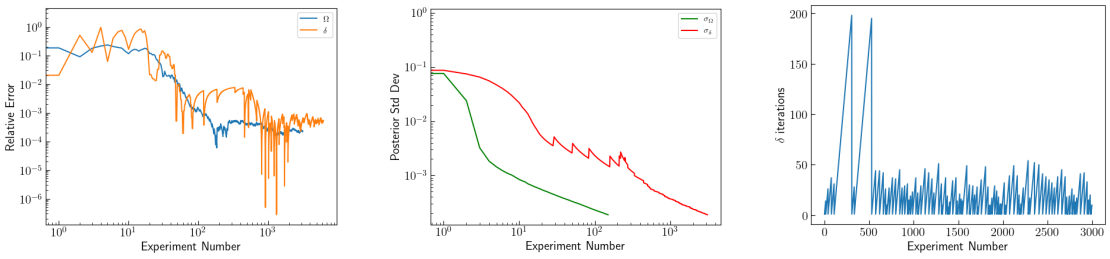


Figure 4.29: Diagrammatic visualization of adaptive SCBF protocol where each estimation protocol is never performed consecutively. Each box represents the estimation cycle being run.

In this case, we begin by performing both Ω and δ estimation for N iterations. However, instead of performing δ estimation again to match the standard deviation of the two parameters, we run Ω estimation again for N iterations. Upon doing so, we then run δ for M iterations until its Fisher information matches that of Ω. We then iterate over this procedure until we arrive at the desired posterior standard deviation for each parameter. Though we observe a larger gap between the Fisher information of Ω and δ, this protocol ensures that we do not run too many iterations of estimation on one parameter using an estimate of its opposing parameter that has a potentially high error, which may be advantageous in attaining a lower final relative error.

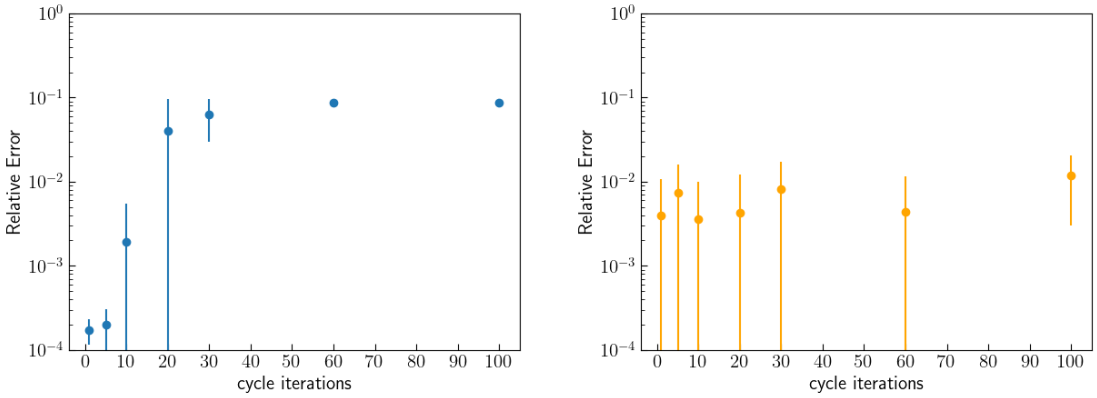
We run an initial experiment at $N = 20$ with this adaptive iteration scheme, observing the results shown in figure 4.30.



(a) (b) (c)

Figure 4.30: Figure 4.30a demonstrates a sample result from an experimental run of the nonconsecutive adaptive estimation with $N = 20$. Figure 4.30b shows the posterior standard deviation for this trial. Figure 4.30c shows the adaptive changes in δ iteration numbers over the course of the experiment.

This sample run demonstrates that this method converges for the arbitrary setting of $N = 20$. To determine the optimal N value, we again search over a grid of potential values, yielding the results shown in figure 4.31.



(a) (b)

Figure 4.31: Relative errors for a grid of iteration numbers with the Fisher information adaptive nonconsecutve optimization algorithm. Figure 4.31a displays the relative errors for Ω and figure 4.31b displays the relative errors for δ.

The non-consecutive methodology yields similar results to that of the consecutive version, though with slightly tighter bounds on the relative error of Ω for trials with smaller iterations numbers. There does not appear to be a discernible advantage to running the non-consecutive algorithm instead of the consecutive algorithm when it comes to final relative error, though in terms of computational and physical run-time, one method may be more efficient than another.

Like all of the algorithms within its class, the non-consecutive method is still plagued by mode slipping, though unlike the non-adaptive iteration methodologies, this is mainly prevalent in δ while Ω is able to consistently converge. This class of algorithms may serve as an adequate estimation method if a relative error of 10^{-2} is satisfactory, but to achieve more robust and lower error results, additional methods that do not eschew information of variance in the opposing estimation parameter or techniques that are able to move the estimator out of incorrect modes may be necessary.

4.6 Total Expectation and Variance

In section 4.5, we apply an approximation to the opposing estimation parameter and assume it to be "true", removing all information about its standard deviation and approximating it to be a point instead of a distribution. Though this approximation still yields convergence in most instances, there is a significant rate of failure that occurs due to the estimator slipping into improper likelihood modes. This phenomenon may arise when the posterior standard deviation we compute does not encapsulate the uncertainty in the opposing estimation parameter, and thus we always take an incorrectly narrower variance. One possible solution to ensure we do not slip into the wrong mode may be to instead compute the mean and standard deviation of the posterior with the effects of the standard deviation of the opposing

estimation parameter accounted for.

The method by which we analyze this full posterior mean and variance is the law of total expectation and variance. The law of total expectation, for two parameters Ω and δ, states that when we are estimating Ω

$$E(\Omega) = E(E(\Omega|\delta)). \tag{4.61}$$

To compute this value for Ω, we can simply draw $\hat{\delta}_i$ from the posterior distribution describing $\hat{\delta}$, make a set of measurements, and generate $\hat{\Omega}_i$ MLEs for all of the $\hat{\delta}_i$ draws. Taking the average of all $\hat{\Omega}_i$s then yields the value $E(\Omega)$

The law of total variance, for two parameters Ω and δ, states that

$$\mathrm{Var}_\Omega(\Omega) = E_\delta[\mathrm{Var}_\Omega(\Omega|\delta)] + \mathrm{Var}_\delta(E_\Omega[\Omega|\delta]) \tag{4.62}$$

where Var denotes the variance and E denotes the expectation value. We will examine the example of estimating Ω. To compute $E_\delta[\mathrm{Var}_\Omega(\Omega|\delta)]$, we will draw a set of $\hat{\delta}_i$s from the posterior PDF of the previous Ramsey experiment estimation cycle. For each $\hat{\delta}_i$, we will generate a set of $\hat{\Omega}$ likelihoods, compute the set of variances for each with the Fisher information, and take the mean of all the variance values.

To compute $\mathrm{Var}_\delta(E_\Omega[\Omega|\delta])$, we will perform a similar procedure, drawing a set of $\hat{\delta}_i$ from the posterior PDF of the previous Ramsey experiment estimation cycle. For each of these $\hat{\delta}_i$s, we will again generate $\hat{\Omega}_i$ MLEs, and compute the standard deviation of the distribution of $\hat{\Omega}_i$ values.

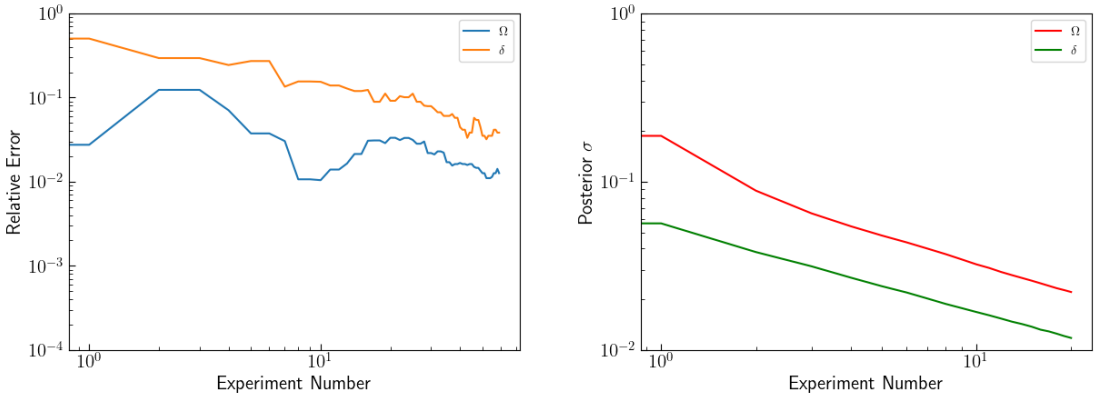
4.6.1 Implementation

A single estimation cycle for Ω with the law of total expectation and variance is displayed in algorithm 5.

Algorithm 5 Bayesian Estimation of Ω with total expectation and variance

```
measurements ← batch_measure(batch_size)
all_δ ← sample_δ_posterior(num_samples, δ_distribution)
all_likelihoods ← generate_likelihoods(all_δ, measurements)
all_posteriors ← prior_Ω_distribution × all_likelihoods
all_posterior_means ← get_mode(all_posteriors)
all_posterior_stddevs ← fisher_to_stddev(all_posteriors)
total_expectation ← mean(all_posterior_means)
total_variance ← mean(all_posterior_stddevs) + fisher_to_stddev(all_posterior_means)
return total_expectation, total_variance
```

The returned total expectation and total variance values represent the posterior mean and posterior variance for Ω with the uncertainty in δ accounted for. To verify the validity of this method, we begin by implementing a baseline version with $k = j = 1$, yielding the results shown in figure 4.32.

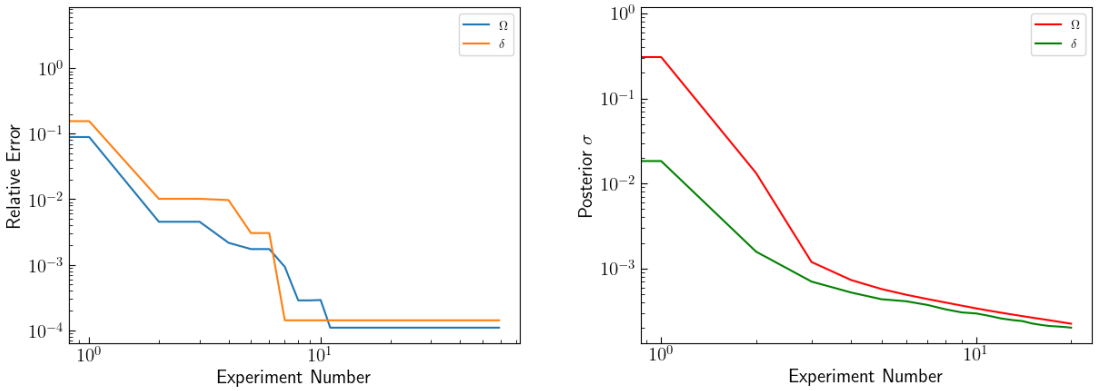


(a) (b)

Figure 4.32: Figure 4.33a displays the relative error for a trial of the BBF algorithm described in section 4.5.1 with the law of total expectation and variance without k or j optimization. Figure 4.33b displays the posterior standard deviation for the same trial.

We observe that the baseline case still converges at the parametric rate in accordance with our theoretical predictions.

We then implement this method with k and j optimization, yielding the results shown in figure 4.33.



(a) (b)

Figure 4.33: Figure 4.33a displays the relative error for a trial of the BBF algorithm described in section 4.5.1 with the law of total expectation and variance with k or j optimization. Figure 4.33b displays the posterior standard deviation for the same trial.

For this single trial, we observe that the estimator converges to a low relative error, quickly hitting the sampling floor. However, when we increase the number of trials and examine the final distribution of relative errors, we observe the results in figure 4.34.

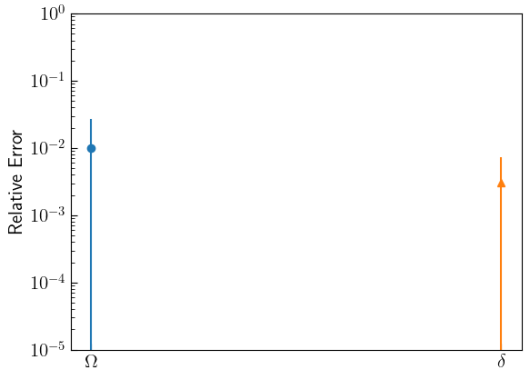


Figure 4.34: Distribution of relative error for 20 trials of the BBF algorithm with total expectation and variance.

Though the law of total expectation and total variance account for the uncertainty in the opposing estimation parameter, we still observe a distribution of relative error that is similar to the point-wise approximation results observed in section 4.4. This effect may be due to the defects in the approximated normal likelihood that are described in section 3.5.4 and sometimes yield divergence of the greedy batched estimation method shown in section 3.5.5. When k values increase in the batched paradigm, we sometimes observe tightly multi-modal likelihoods that cannot be well-described by approximated normal distributions. For the example of estimating Ω with the total expectation and variance, we see in figure 4.35 that the posteriors that arise when we draw from the posterior δ distributions are similarly bimodal.

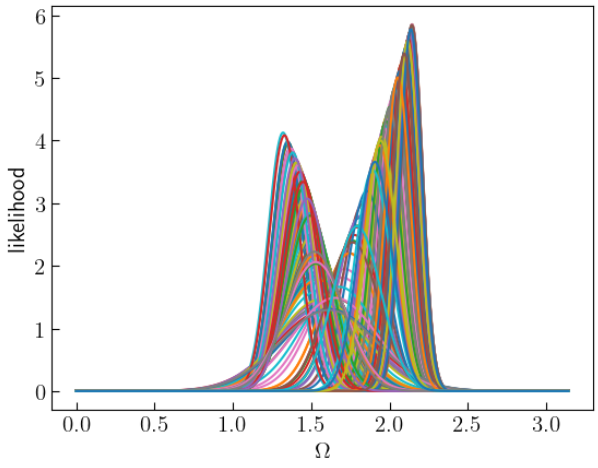


Figure 4.35: Output Ω posteriors generated from values of δ generated from the posterior δ distribution when $k > 1$.

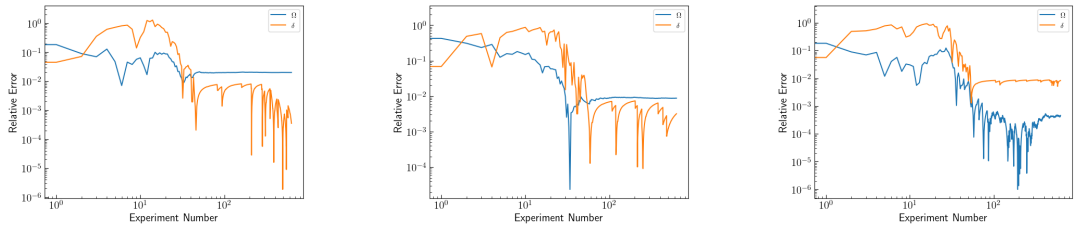
We observe a distribution of values that are strongly bi-modal. When the variance and expectation are subsequently calculated with the law of total variance and expectation for

the samples drawn from figure 4.35, our estimator breaks down. To resolve this issue, it may be necessary to add additional machinery to the batched estimation procedure, such as a peak finding algorithm that picks the likelihood mode with a MLE that is closet to the prior mean and sets all other modes to 0.

4.7 jk Sampling

In section 3.4.3, we applied a k sampling methodology to escape likelihood modes when parameter drift was present. In this section, we investigate applying a similar procedure to escape incorrect likelihood modes during the two-parameter estimation case.

When we begin the two-parameter estimation trials, we start with some initial δ and Ω prior distributions, though bounded, are relatively wide and uninformative. When we first begin estimating Ω, we use the mean of this δ distribution in the likelihood function, and vice versa for estimating δ. However, the relative error of these mean values is likely quite high initially, and though in section 4.5.1 we searched over iteration numbers per cycle to remedy this issue, error in the opposing estimation parameter may still propagate through the experiment. As a result, the estimator may move towards a value that is biased by this error, and when the posterior standard deviation is sufficiently low, become trapped in a likelihood mode that bounds the range of possible values the estimator can converge to. This can yield estimation procedures that become stuck within local minima, some of which are shown in figure 4.36.



(a) (b) (c)

Figure 4.36: Various two-parameter SCBF experiments where one of Ω and δ become trapped in the incorrect likelihood mode and are unable to escape, leading to a poor relative error.

To resolve this issue, we sample j and k from a distribution instead of using the optimization function derived in section 3.2. This ensures that if the estimator begins to move towards an incorrect value and becomes trapped in a likelihood mode, there is a probability that a lower k or j value is sampled, allowing it to escape and move to a different value.

We sample j and k from a half-normal distribution centered at the actual value of j or k. An example of this distribution is shown in figure 4.37.

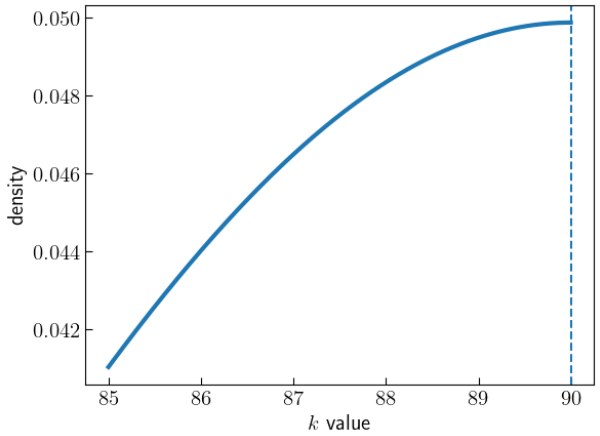


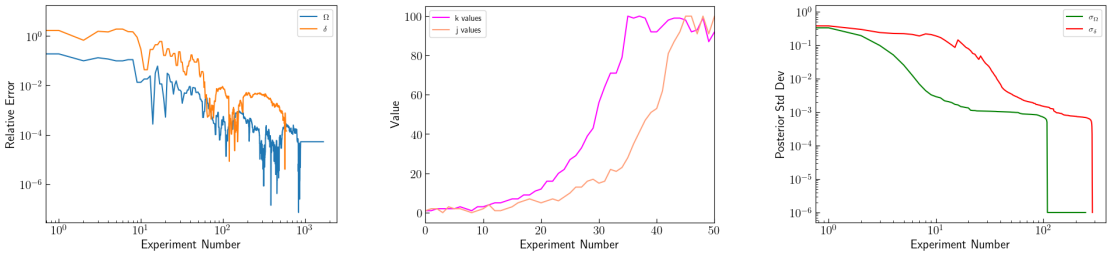
Figure 4.37: Half-normal distribution of k values. The dashed line denotes the true k value.

For computational simplicity, we instead subtract the absolute value of samples from a normal distribution centered at 0 with a designated standard deviation dependent upon the current k or j value from the current k or j value. Our updated optimization rule-set is computationally described as

```
if prior_std < 0.025258200269627846:
    k = np.round(0.11253956/x)
    return k - np.floor(np.abs(np.random.normal(0,k_std)))
elif prior_std < 0.03269749744511768:
    return 4 - np.floor(np.abs(np.random.normal(0,2)))
elif prior_std < 0.04684580115873045:
    return 3 - np.floor(np.abs(np.random.normal(0,1)))
elif prior_std < 0.08688382635251184:
    return 2 - np.floor(np.abs(np.random.normal(0,1)))
else:
    return 1
```

The parameter k_std can be thought of as a tunable hyperparameter, with larger k_std values yielding a higher frequency of lower k values.

We run a sample experiment with this technique integrated with the CAE scheme mentioned in scetion 4.5.3, setting k_std $= k/10$, yielding the results shown in figure 4.38.



(a) (b) (c)

Figure 4.38: A sample run of CAE with j, k sampled from the half-normal distribution at $N = 5$ over 50 rounds. Figure 4.38a displays the relative error of Ω and δ, figure 4.38b displays the sampled j and k values, and figure 4.38c displays the posterior standard deviation for Ω and δ (note that the defect in the tails of the curves are a result of sampling instability).

To analyze robustness of this methodology, we run 20 trials with the same experimental settings, observing the results shown in figure 4.39.

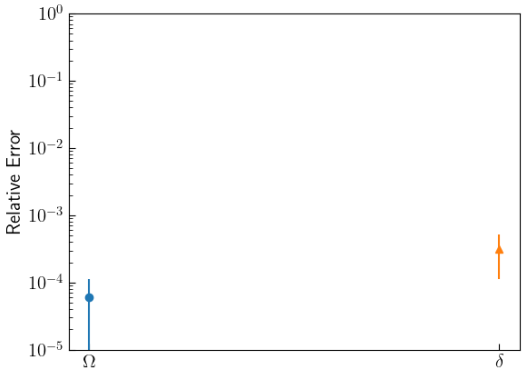


Figure 4.39: CAE with j, k sampling over 20 trials run at $N = 5$ over 600 rounds.

With the integration of jk sampling, we see that the CAE procedure now achieves an incredibly tight bound at a low relative error. Through this sampling method, we have resolved the issue of mode-trapping discussed at the beginning of this section and demonstrated a robust method for two-parameter estimation.

4.8 Summary

In this chapter, we have derived the likelihood for a Ramsey experiment and explored the two parameter likelihoods for both δ and Ω. We then demonstrated that the optimizations and approximations developed for the simplified single parameter case also carry to the two parameter case, and realize successful Bayesian estimation of the detuning δ and the Rabi frequency Ω in the simplified two parameter case where the second calibration parameter is assumed to be well-known.

Additionally, we have investigated the true experimental case where both parameters are unknown and error may be present in both δ and Ω. For δ, we have shown that there is greater sensitivity to an under-estimated Ω value than an over-estimated Ω value, and for Ω we have shown that the estimated Ω value is maximized when we choose $\delta = 0$ as the initial estimate.

We then developed a multitude of paradigms for two-parameter estimation. We began with the SCBF algorithm (4.5.1) where we estimate each parameter for a fixed number of single iterations before swapping to the opposing parameter and doing the same. We searched over a grid of iteration sizes, finding that a symmetrical iteration number of 20 yielded optimal results. We then explored the BBF algorithm in section 4.5.1, where we perform batched estimation on a single parameter for a fixed number of iterations before swapping to the opposing parameter and doing the same. We searched over a grid of batch sizes and iterations, finding that a batch size of 50 and an iteration number of 2 yielded optimal results.

We then developed a more generalizable method for determining iteration number and batch size in section 4.5.2, applying the Fisher information and estimating one parameter for a fixed iteration number and another until $\sigma_\Omega \approx \sigma_\delta$.

However, all of these methods suffered from a high rate of divergence and failure due to mode-slipping. In section 4.6 we first postulated that the use of the total expectation and variance during the computation of the posterior mean and standard deviation could resolve this issue, but this procedure still resulted in outputs that faced the same issues as those described in earlier sections. Instead, in section 4.7, we developed a method for sampling j and k so that the estimator could escape incorrect modes. This led to an extremely robust and consistent estimator that yielded relative errors of below 10^{-3} without divergence.

Chapter 5

Extensions

There is a large set of extensions that can be applied to the aforementioned developments. In the area of parameter drift, schemes of greater complexity, such as the use of a separate Bayesian memory [68] and adaptive forgetting filters [64] can be implemented. Furthermore, there are a handful of computational optimizations and pre-computation opportunities for each of the algorithms described in chapter 3, especially in the domain of the batched Fisher information-based estimation procedure. Furthermore, a more in-depth examination of the counter-intuitive forgetting algorithm described in section 3.4.3 may be helpful for understanding why decreasing the posterior standard deviation yields an algorithm that is resilient to drift. Additional investigations on alternative distributions to sample k, as well as further hyperparameter searches on each sampling distribution, may be helpful in yielding a sampling method that outperforms the current technique described in section 3.4.3. When there is SPAM error in the estimator, it also may be possible to develop an algorithm that analyzes the posterior standard deviation in real time and actively remove measurements and Bayesian estimation cycles that are done with SPAM error present in the system.

In chapter 3, the current code written for the computation of the Fisher information with the batching algorithm described in section 3.5.1 does not fully utilize the optimizations that arises when the Fisher information is used to fit the normal distribution, as it iterates over the entire array of points in the distribution. A simplified point-wise computation can be developed that changes the runtime of the algorithm from $O(n)$ to $O(1)$. Additionally, there are a myriad of possible pre-computation schemes that can be developed for the algorithms, which can drastically speed-up computational time.

The two-parameter case discussed in chapter 4 also has a wide array of possible points of interest. Additional MCMC algorithms, such as the particle filter described in Gerster et al.[33], may be useful to simplify and discretize calculations in the larger parameter space. A gridded pre-calculation scheme for the denominator of the Bayesian estimator may also be useful to develop, as current methods involve numerical integration of the normal distribution for single-cycle methods. A similar gridded pre-calculation scheme can be created to enable the use of the wrapped normal distribution. Furthermore, the investigation of additional distributions and the implementation of hyperparameter searches can potentially yield faster

convergence for the sampling technique described in section 4.7 and lead to an overall speed-up for the two-parameter calibration case. Benchmarks for each of these algorithms can also be performed in a similar fashion to the two parameter case described in section 3.6 to fully understand the cost of each method.

An exploration of how parameter drift affects the two parameter case is additionally a large space that has not been explored in this thesis. With the coupling between δ and Ω estimations, the introduction of parameter drift into the procedure will likely introduce additional complications and require methods for drift detection and correction that differ from the ones discussed in the single parameter cases.

It should also be noted that the set of methods developed in this thesis are purely simulated. To truly verify that these schemes can improve calibration speed and accuracy, validation on a physical quantum computer or through an experimental setup that is capable of performing Rabi and Ramsey experiments is necessary.

Appendix A

Code Repository

All code used in this thesis can be found at the Github repository. All figures used in this thesis can also be found in this repository. The repository is organized by folder for each parameter estimation case. The breakdown is listed below. All figures as well as additional graphs and diagrams can also be found in this repository.

A.1 Single Parameter

The single parameter folder covers the set of experiments and methods that cover the simplified single parameter case described in section 2.4. It contains methods related to the initial baseline algorithm, the k-optimized algorithm, as well as the various batched algorithms. It also contains the tests related to parameter drift and SPAM error. All of the notebooks used to generate graphs are contained here as well.

All of the currently in-use methods are contained in ExperimentClass.py. The lower level helper methods that comprise of the Bayesian estimation procedure are contained in BayesFuncs.py.

A.1.1 Subfolders

1. Archive: Contains a set of deprecated files and methods that were used for testing but were either made redundant, were updated, or were unsuccessful.

2. Lab: Contains the files used for testing and validation of different single parameter experimental methods

3. results: Contains sets of outputs from the various runs, including the baseline tests, the batching tests, the drift tests, the normal approximation tests, SPAM error tests, and von mises approximation tests

A.2 Two Parameter

The two parameter folder contains all of the experiments for the Rabi and detuning estimation. All of the currently in-use methods are contained in TwoParamBayesFuncs.py. The lower level helper methods for the Rabi rate estimation are contained in RabiFuncs.py, the lower level helper methods for the detuning estimation are contained in RamseyFuncs.py, and the lower level helper methods that are used by both are contained GeneralFuncs.py.

A.2.1 Subfolders

Archive: Contains a set of old files and exploration notebooks that are no longer in use

Lab: Contains the files used for testing and validation of different single parameter experimental methods

TwoParamFigs: Contains graphs and images generated for the two-parameter estimation

A.3 File Usage by Thesis Sections

Generally, all finalized and useful methods for the single parameter case are housed in SingleParameter\ExperimentsClass.py, with helper methods being drawn from SingleParameter\BayesFuncs.py. For the two parameter estimation, all useful methods are housed in TwoParameter\TwoParamBayesFuncs.py, with helpers methods from TwoParameter\GeneralFuncs.py, TwoParameter\RabiFuncs.py, and TwoParameter\RamseyFuncs.py.

- 2.1 Solving the Hamiltonian: Derivation can be found in SingleParameter\Lab\Electron in a Magnetic Field.ipynb
- 3.1 Baseline Software Implementation: Baseline method can be found in SingleParameter\ExperimentsClass.py
- 3.2 Adaptive Optimization: k-optimization derivation can be found in SingleParameter\Lab\k_opt_derive.ipynb
- 3.2 Adaptive Optimization: k-optimized estimation can be found in SingleParameter\ExperimentsClass.py
- 3.3 Normal Approximations: Normal approximation and fitting methods can be found in SingleParameter\ExperimentsClass.py
- 3.4 Error Sources and Robustness: Drift detection and fixing methods can be found in SingleParameter\ExperimentsClass.py, SingleParameter\lab.ipynb and SingleParameter\Lab\PPCTesting.ipynb
- 3.4.4 SPAM Error: Investigation of SPAM error can be found in SingleParameter\Lab\SPAMTests.ipynb

- 3.5 Batching: Batching methods can be found in SingleParameter\ExperimentClass.py
- 3.5.1 Fisher Information: Implementation of Fisher fitting can be found in SingleParameter\BayesFuncs.py
- 3.5.4 k-optimization: Testing of batched k-optimization can be found in SingleParameter\Lab\BinomLikKs.py and SingleParameter\Lab\BatchKTests.ipynb
- 3.5.5 Greedy Batched Estimation: The greedy batched and efficient greedy batched methods can be found in SingleParameter\ExperimentClass.py
- 3.5.7 B2S: The B2S methods can be found in SingleParameter\ExperimentClass.py
- 3.6 Benchmarks: The benchmarking tests are found in SingleParameter\Lab\QuantumBenchmarks.ipynb
- 4.2 Likelihood Investigation: The investigation methods can all be found in TwoParameter\archive\DataExplore.ipynb and TwoParameter\lab\LikelihoodExp.ipynb
- 4.3.1 MLE: MLE testing methods can be found in TwoParameter\lab\MLETests.ipynb
- 4.3.2 Baseline Bayesian Estimation of the Detuning and Rabi Frequency: Testing procedures and methods can be found in TwoParameter\lab\SingleCycleFapp.ipynb
- 4.5.1 Symmetrical Single Cycle Investigation: The exploration of the optimal parameters for this case can be found in TwoParameter\lab\SymInvestigate.ipynb
- 4.5.1 Symmetrical Single Cycle Investigation: The exploration of the optimal parameters for this case can be found in TwoParameter\lab\SymInvestigate.ipynb
- 4.5.1 Batched Back and Forth: The exploration of the optimal parameters for this case can be found in TwoParameter\lab\SymIinvestigate.ipynb
- 4.5.2 Asymmetric Fisher Information-based Estimation: The development and exploration of the optimal parameters for this case can be found in TwoParameter\lab\AsymInvestigate.ipynb
- 4.6 Total Expectation and Variance: The development and testing of this methodology can be found in TwoParameter\lab\TotalVariance.ipynb
- 4.7jk Sampling: The development and testing of this methodology can be found in TwoParameter\lab\SampleK.ipynb
- General plotting functions for building the stacked plots are in SingleParameter\Lab\Plotting.py and TwoParameter\lab\TwoPlot.py

Appendix B

Additional Plots

B.1 Brownian Drift-Influenced Estimation

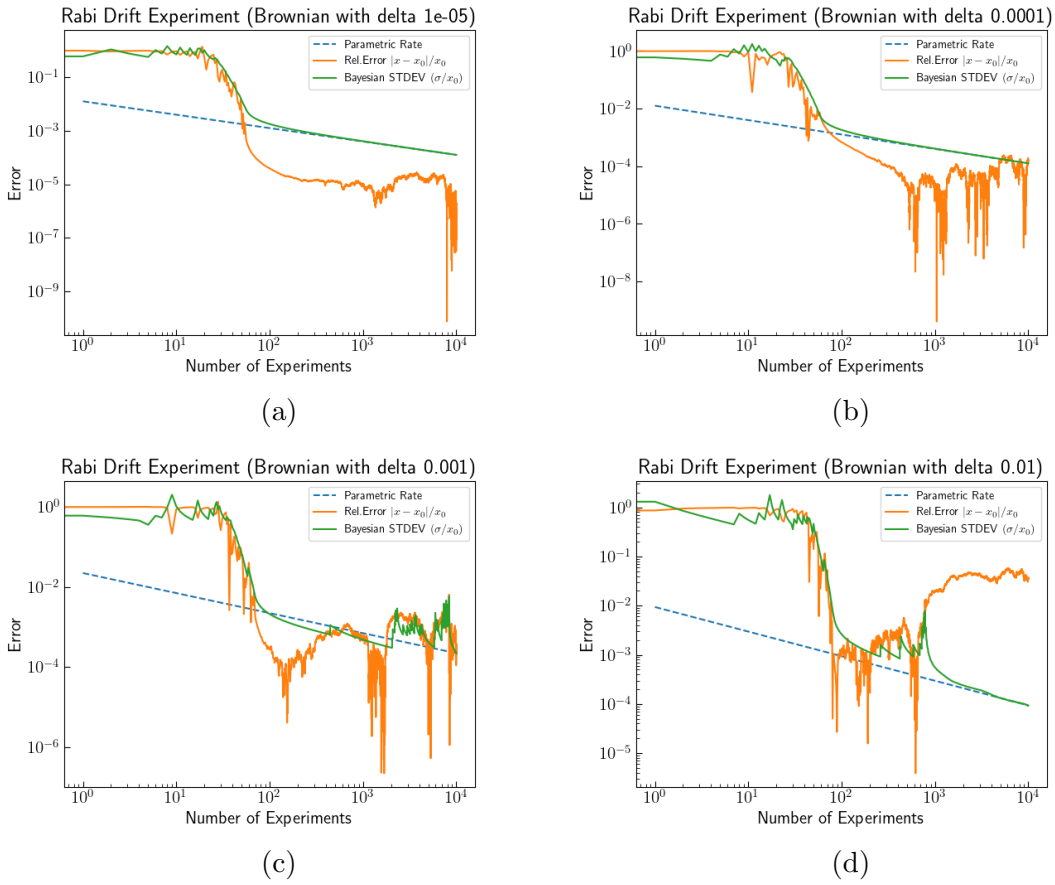
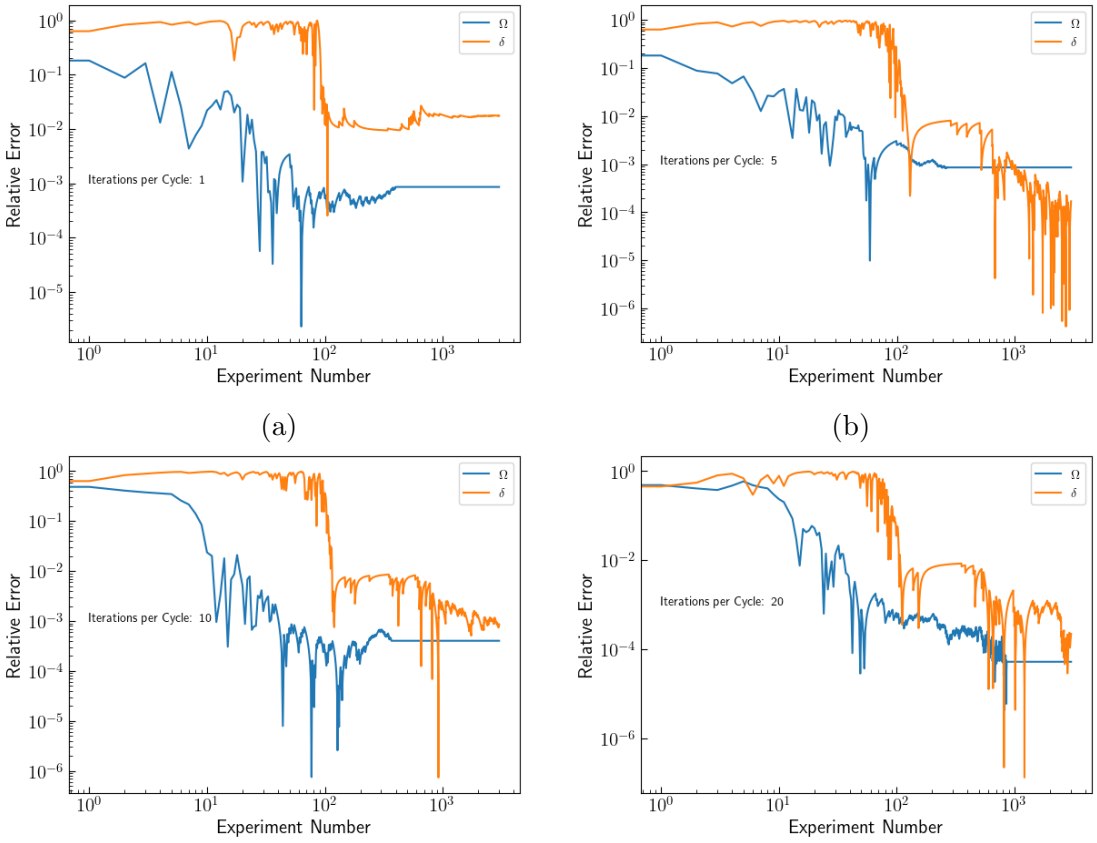


Figure B.1: Bayesian estimation trials with Brownian drift present.

B.2 SCBF



(a) (b)

(c) (d)

Figure B.2

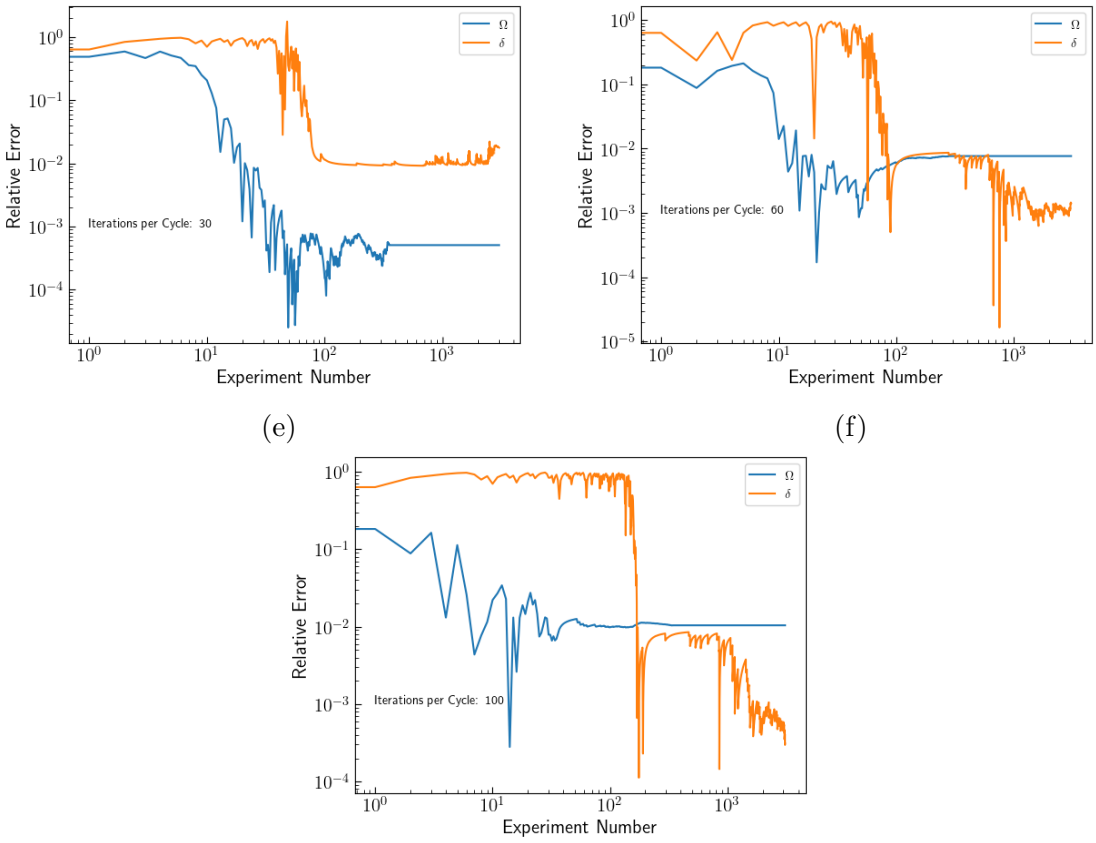


Figure B.2: Individual SCBF estimation trials for a variety of batch sizes.

B.3 Asymmetrical Consecutive Adaptive Fisher Information Optimization

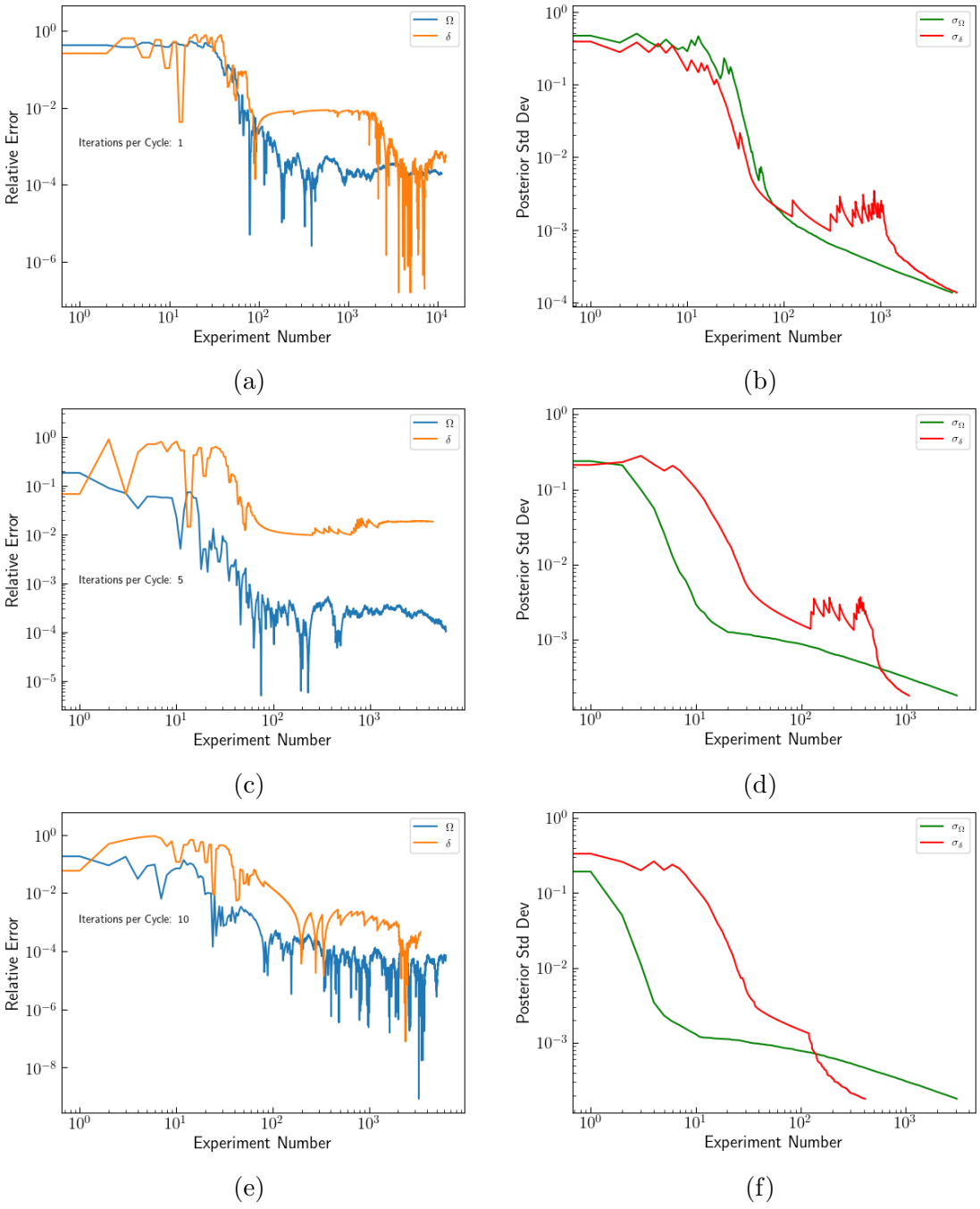


Figure B.3

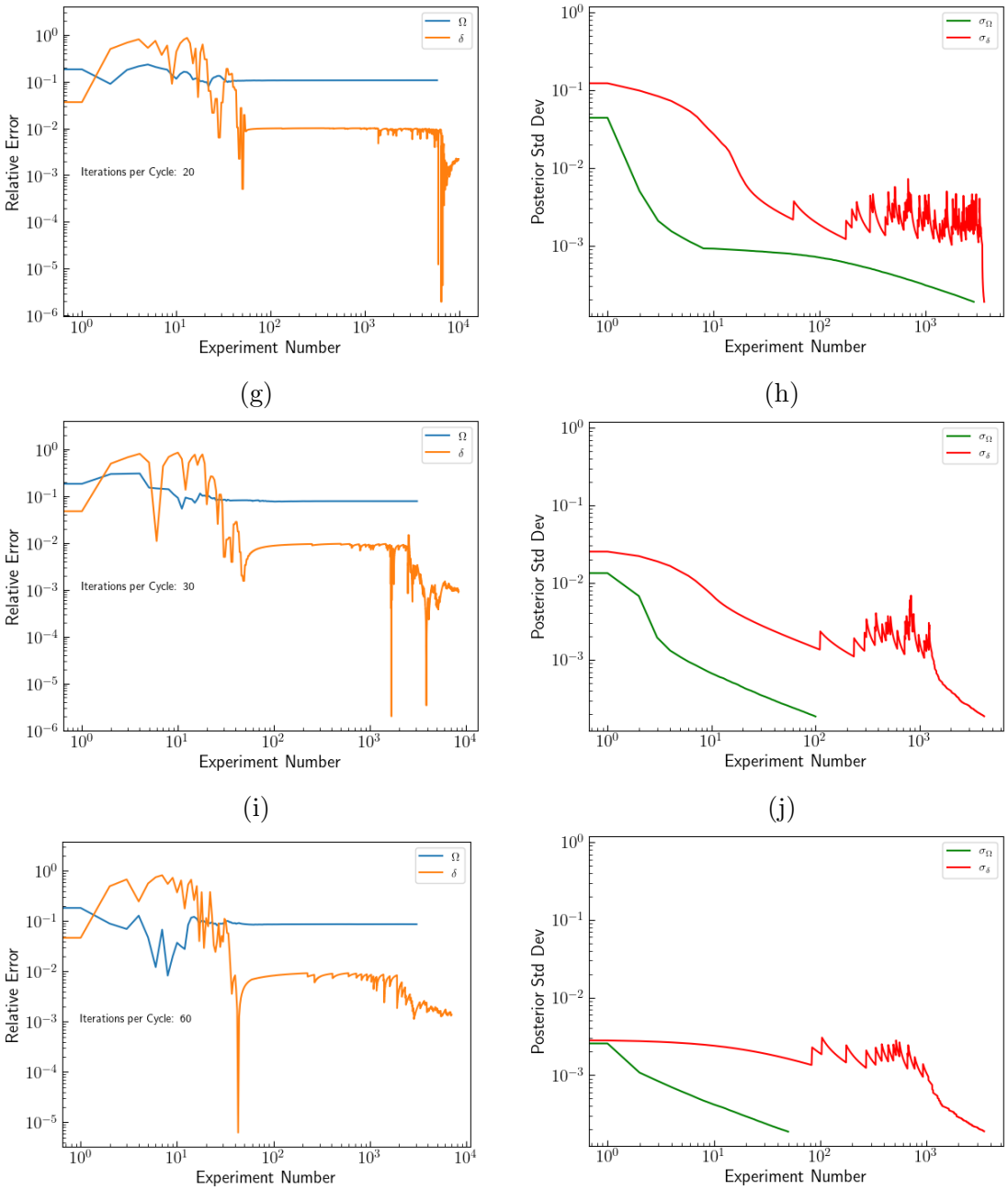
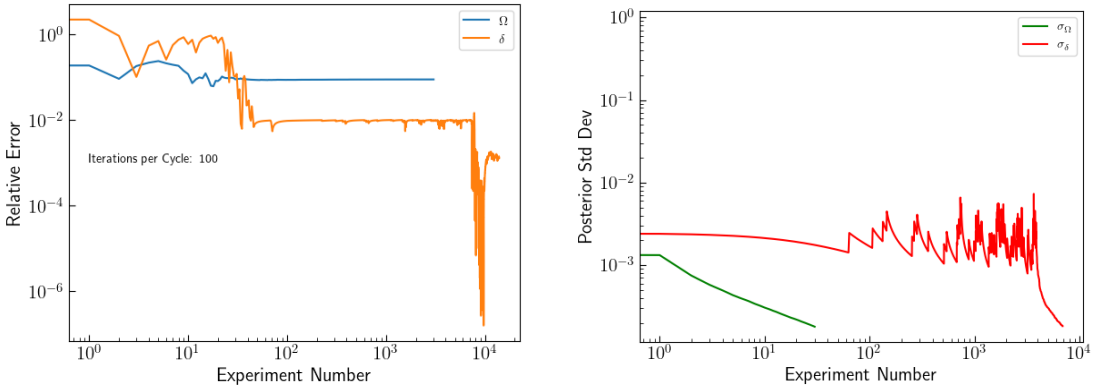

Relative Error
Iterations per Cycle: 20
Experiment Number
Posterior Std Dev
Experiment Number
(g)
(h)
Relative Error
Iterations per Cycle: 30
Experiment Number
Posterior Std Dev
Experiment Number
(i)
(j)
Relative Error
Iterations per Cycle: 60
Experiment Number
Posterior Std Dev
Experiment Number
(k)
(l)

Figure B.3



(m) (n)

Figure B.3: Additional asymmetric Fisher information-based consecutive optimization plots

Bibliography

[1] K. R. Brown, A. C. Wilson, Y. Colombe, C. Ospelkaus, A. M. Meier, E. Knill, D. Leibfried, and D. J. Wineland, Phys. Rev. A **84**, 030303 (2011).

[2] J. Benhelm, G. Kirchmair, C. F. Roos, and R. Blatt, Nature Physics **4**, 463 (2008).

[3] B. B. Blinov, D. Leibfried, C. Monroe, and D. J. Wineland, Quantum Information Processing **3**, 45 (2004).

[4] J. M. Pino, J. M. Dreiling, C. Figgatt, J. P. Gaebler, S. A. Moses, M. S. Allman, C. H. Baldwin, M. Foss-Feig, D. Hayes, K. Mayer, C. Ryan-Anderson, and B. Neyenhuis, Nature **592**, 209 (2021).

[5] B. de Neeve, "Calibrating an ion-trap quantum computer", unpublished thesis, 2017.

[6] P. Shor, in Proceedings 35th annual symposium on foundations of computer science (1994), pp. 124–134.

[7] A. Montanaro, npj Quantum Information **2**, 10.1038/npjqi.2015.23 (2015).

[8] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, Rev. Mod. Phys. **94**, 015004 (2022).

[9] S. McArdle, S. Endo, A. Aspuru-Guzik, S. C. Benjamin, and X. Yuan, Rev. Mod. Phys. **92**, 015003 (2020).

[10] T. Jennewein, C. Simon, G. Weihs, H. Weinfurter, and A. Zeilinger, Phys. Rev. Lett. **84**, 4729 (2000).

[11] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin, Journal of Cryptology **5**, 3 (1992).

[12] C. H. Bennett, Phys. Rev. Lett. **68**, 3121 (1992).

[13] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins, A. E. Lita, T. Gerrits, S. W. Nam, V. D. Vaidya, M. Menotti, I. Dhand, Z. Vernon, N. Quesada, and J. Lavoie, Nature **606**, 75 (2022).

[14] Q. Zhu, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan, M. Gong, C. Guo, C. Guo, S. Guo, L. Han, L. Hong, H.-L. Huang, Y.-H. Huo, L. Li, N. Li, S. Li, Y. Li, F. Liang, C. Lin, J. Lin, H. Qian, D. Qiao, H. Rong, H. Su, L. Sun, L. Wang, S. Wang, D. Wu, Y. Wu, Y. Xu, K. Yan, W. Yang, Y. Yang, Y. Ye, J. Yin, C. Ying, J. Yu, C. Zha, C. Zhang, H. Zhang, K. Zhang, Y. Zhang, H. Zhao, Y. Zhao, L. Zhou, C.-Y. Lu, C.-Z. Peng, X. Zhu, and J.-W. Pan, *Quantum computational advantage via 60-qubit 24-cycle random circuit sampling*, 2021.

[15] M. Ahsan, R. V. Meter, and J. Kim, ACM Journal on Emerging Technologies in Computing Systems **12**, 1 (2016).

[16] D. Kielpinski, C. Monroe, and D. Wineland, Nature **417**, 709 (2002).

[17] G. Wendin, Reports on Progress in Physics **80**, 106001 (2017).

[18] M. Saffman, Journal of Physics B: Atomic, Molecular and Optical Physics **49**, 202001 (2016).

[19] S. Slussarenko and G. J. Pryde, Applied Physics Reviews **6**, 041303 (2019).

[20] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, Applied Physics Reviews **6**, 021314 (2019).

[21] R. Srinivas, S. C. Burd, H. M. Knaack, R. T. Sutherland, A. Kwiatkowski, S. Glancy, E. Knill, D. J. Wineland, D. Leibfried, A. C. Wilson, D. T. C. Allcock, and D. H. Slichter, Nature **597**, 209 (2021).

[22] J. P. Gaebler, T. R. Tan, Y. Lin, Y. Wan, R. Bowler, A. C. Keith, S. Glancy, K. Coakley, E. Knill, D. Leibfried, and D. J. Wineland, Phys. Rev. Lett. **117**, 060505 (2016).

[23] Y. Zhao, Y. Ye, H.-L. Huang, Y. Zhang, D. Wu, H. Guan, Q. Zhu, Z. Wei, T. He, S. Cao, F. Chen, T.-H. Chung, H. Deng, D. Fan, M. Gong, C. Guo, S. Guo, L. Han, N. Li, S. Li, Y. Li, F. Liang, J. Lin, H. Qian, H. Rong, H. Su, L. Sun, S. Wang, Y. Wu, Y. Xu, C. Ying, J. Yu, C. Zha, K. Zhang, Y.-H. Huo, C.-Y. Lu, C.-Z. Peng, X. Zhu, and J.-W. Pan, Phys. Rev. Lett. **129**, 030501 (2022).

[24] W. Huang, C. H. Yang, K. W. Chan, T. Tanttu, B. Hensen, R. C. C. Leon, M. A. Fogarty, J. C. C. Hwang, F. E. Hudson, K. M. Itoh, A. Morello, A. Laucht, and A. S. Dzurak, Nature **569**, 532 (2019).

[25] S. S. Hong, A. T. Papageorge, P. Sivarajah, G. Crossman, N. Didier, A. M. Polloreno, E. A. Sete, S. W. Turkowski, M. P. da Silva, and B. R. Johnson, Phys. Rev. A **101**, 012302 (2020).

[26] P. H. Leung, K. A. Landsman, C. Figgatt, N. M. Linke, C. Monroe, and K. R. Brown, Phys. Rev. Lett. **120**, 020501 (2018).

[27] R. Acharya, I. Aleiner, R. Allen, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, J. Atalaya, R. Babbush, D. Bacon, J. C. Bardin, J. Basso, A. Bengtsson, S. Boixo, G. Bortoli, A. Bourassa, J. Bovaird, L. Brill, M. Broughton, B. B. Buckley, D. A. Buell, T. Burger, B. Burkett, N. Bushnell, Y. Chen, Z. Chen, B. Chiaro, J. Cogan, R. Collins, P. Conner, W. Courtney, A. L. Crook, B. Curtin, D. M. Debroy, A. D. T. Barba, S. Demura, A. Dunsworth, D. Eppens, C. Erickson, L. Faoro, E. Farhi, R. Fatemi, L. F. Burgos, E. Forati, A. G. Fowler, B. Foxen, W. Giang, C. Gidney, D. Gilboa, M. Giustina, A. G. Dau, J. A. Gross, S. Habegger, M. C. Hamilton, M. P. Harrigan, S. D. Harrington, O. Higgott, J. Hilton, M. Hoffmann, S. Hong, T. Huang, A. Huff, W. J. Huggins, L. B. Ioffe, S. V. Isakov, J. Iveland, E. Jeffrey, Z. Jiang, C. Jones, P. Juhas, D. Kafri, K. Kechedzhi, J. Kelly, T. Khattar, M. Khezri, M. Kieferová, S. Kim, A. Kitaev, P. V. Klimov, A. R. Klots, A. N. Korotkov, F. Kostritsa, J. M. Kreikebaum, D. Landhuis, P. Laptev, K.-M. Lau, L. Laws, J. Lee, K. Lee, B. J. Lester, A. Lill, W. Liu, A. Locharla, E. Lucero, F. D. Malone, J. Marshall, O. Martin, J. R. McClean, T. Mccourt, M. McEwen, A. Megrant, B. M. Costa, X. Mi, K. C. Miao, M. Mohseni, S. Montazeri, A. Morvan, E. Mount, W. Mruczkiewicz, O. Naaman, M. Neeley, C. Neill, A. Nersisyan, H. Neven, M. Newman, J. H. Ng, A. Nguyen, M. Nguyen, M. Y. Niu, T. E. O'Brien, A. Opremcak, J. Platt, A. Petukhov, R. Potter, L. P. Pryadko, C. Quintana, P. Roushan, N. C. Rubin, N. Saei, D. Sank, K. Sankaragomathi, K. J. Satzinger, H. F. Schurkus, C. Schuster, M. J. Shearn, A. Shorter, V. Shvarts, J. Skruzny, V. Smelyanskiy, W. C. Smith, G. Sterling, D. Strain, M. Szalay, A. Torres, G. Vidal, B. Villalonga, C. V. Heidweiller, T. White, C. Xing, Z. J. Yao, P. Yeh, J. Yoo, G. Young, A. Zalcman, Y. Zhang, and N. Zhu, *Suppressing quantum errors by scaling a surface code logical qubit*, 2022.

[28] P. Klimov, J. Kelly, J. M. Martinis, and H. Neven, ArXiv **abs/2006.04594** (2020).

[29] Y. Li, L. Pezzè, M. Gessner, Z. Ren, W. Li, and A. Smerzi, Entropy **20** (2018).

[30] J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, E. Lucero, M. Neeley, C. Neill, P. J. J. O'Malley, C. Quintana, P. Roushan, A. Vainsencher, J. Wenner, and J. M. Martinis, Phys. Rev. A **94**, 032321 (2016).

[31] S. Majumder, L. de Castro, and K. Brown, npj Quantum Information **6**, `10.1038/s41534-020-0251-y` (2020).

[32] A. Patterson, J. Rahamim, T. Tsunoda, P. Spring, S. Jebari, K. Ratter, M. Mergenthaler, G. Tancredi, B. Vlastakis, M. Esposito, and P. Leek, Phys. Rev. Applied **12**, 064013 (2019).

[33] L. Gerster, F. Martınez-Garcıa, P. Hrmo, M. W. van Mourik, B. Wilhelm, D. Vodola, M. Müller, R. Blatt, P. Schindler, and T. Monz, PRX Quantum **3**, 020350 (2022).

[34] B. Teklu, S. Olivares, and M. G. A. Paris, Journal of Physics B: Atomic, Molecular and Optical Physics **42**, 035502 (2009).

[35] J. M. Lukens, K. J. H. Law, A. Jasra, and P. Lougovski, New Journal of Physics **22**, 063038 (2020).

[36] V. Gebhart, A. Smerzi, and L. Pezzè, Phys. Rev. Applied **16**, 014035 (2021).

[37] N. Wiebe and C. Granade, Phys. Rev. Lett. **117**, 010503 (2016).

[38] F. Martınez-Garcıa, D. Vodola, and M. Müller, New Journal of Physics **21**, 123027 (2019).

[39] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information: 10th anniversary edition* (Cambridge University Press, 2010).

[40] N. Yanofsky and M. Mannucci, *Quantum computing for computer scientists* (Jan. 2008).

[41] R. L. Rivest, A. Shamir, and L. Adleman, Commun. ACM **21**, 120 (1978).

[42] L. K. Grover, *A fast quantum mechanical algorithm for database search*, 1996.

[43] H.-L. Huang, D. Wu, D. Fan, and X. Zhu, Science China Information Sciences **63**, 10.1007/s11432-020-2881-9 (2020).

[44] T. M. Graham, Y. Song, J. Scott, C. Poole, L. Phuttitarn, K. Jooya, P. Eichler, X. Jiang, A. Marra, B. Grinkemeyer, M. Kwon, M. Ebert, J. Cherek, M. T. Lichtman, M. Gillette, J. Gilbert, D. Bowman, T. Ballance, C. Campbell, E. D. Dahl, O. Crawford, N. S. Blunt, B. Rogers, T. Noel, and M. Saffman, Nature **604**, 457 (2022).

[45] M. Vinet, Nature Nanotechnology **16**, 1296 (2021).

[46] P. K. Ghoush, *Ion traps* (Clarendon Press, 1995).

[47] J. I. Cirac and P. Zoller, Phys. Rev. Lett. **74**, 4091 (1995).

[48] A. H. Myerson, D. J. Szwer, S. C. Webster, D. T. C. Allcock, M. J. Curtis, G. Imreh, J. A. Sherman, D. N. Stacey, A. M. Steane, and D. M. Lucas, Phys. Rev. Lett. **100**, 200502 (2008).

[49] C. J. Foot, *Atomic physics*, Oxford master series in atomic, optical, and laser physics (Oxford University Press, Oxford, 2007).

[50] B. Lauritzen, S. R. Hastings-Simon, H. de Riedmatten, M. Afzelius, and N. Gisin, Phys. Rev. A **78**, 043402 (2008).

[51] D. Leibfried, R. Blatt, C. Monroe, and D. Wineland, Rev. Mod. Phys. **75**, 281 (2003).

[52] E. Abbe, Archiv für Mikroskopische Anatomie **9**, 413 (1873).

[53] Z. Yusheng, Z. Zhifeng, L. Yang, L. Dongdong, L. Jintao, C. Yinhang, S. Yuling, L. Wenlong, and W. Xinjie, in 2015 international conference on optoelectronics and microelectronics (icom) (2015), pp. 107–110.

[54] S. Yang, J.-Y. Zhang, Y.-Y. Yang, J.-Y. Huang, Y.-R. Bai, Y. Zhang, and X.-C. Lin, Results in Physics **13**, 102201 (2019).

[55] D. A. Steck, *Quantum and atom optics* (2007).

[56] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*, 2nd ed. (Chapman and Hall/CRC, 2004).

[57] B. Lambert, *A student's guide to bayesian statistics* (SAGE Publications, 2018).

[58] G. Kurz, I. Gilitschenski, and U. D. Hanebeck, in 2014 sensor data fusion: trends, solutions, applications (sdf) (2014), pp. 1–5.

[59] K. V. Mardi and P. E. Jupp, *Directional statistics*, 2nd ed. (John Wiley & Sons, 1999).

[60] D. J. Griffiths and D. F. Schroeter, *Introduction to quantum mechanics*, 3rd ed. (Cambridge University Press, 2018).

[61] B. Wlodzimierz, *The normal distribution characterizations with applications*, 1st ed. (Springer New York, NY, 1995).

[62] E. van den Berg, Quantum **5**, 469 (2021).

[63] D. Revuz and M. Yor, *Continuous martingales and brownian motion* (Springer-Verlag, 2010).

[64] V. Moens, in Proceedings of the 35th international conference on machine learning, Vol. 80, edited by J. Dy and A. Krause, Proceedings of Machine Learning Research (Oct. 2018), pp. 3606–3615.

[65] J. Nassar, J. R. Brennan, B. Evans, and K. Lowrey, in International conference on learning representations (2022).

[66] Y. Wang, A. Kucukelbir, and D. M. Blei, in Proceedings of the 34th international conference on machine learning - volume 70, ICML'17 (2017), pp. 3646–3655.

[67] V. Moens and A. Zenon, `10.1101/294959` (2018).

[68] A. Marblestone, Y. Wu, and G. Wayne, *Product kanerva machines: factorized bayesian memory*, 2020.

[69] A. Honkela and H. Valpola, (2003).

[70] S. Bize, P. Laurent, M. Abgrall, H. Marion, I. Maksimovic, L. Cacciapuoti, J. Gruenert, C. Vian, F. Pereira dos Santos, P. Rosenbusch, P. Lemonde, G. Santarelli, P. Wolf, A. Clairon, A. Luiten, M. Tobar, and C. Salomon, Journal of Physics B: Atomic, Molecular and Optical Physics **38**, S449 (2005).