Bayesian Inference with Engineered Likelihood Functions for Robust Amplitude Estimation

Guoming Wang*[1], Dax Enshan Koh†[1], Peter D. Johnson‡[1], and Yudong Cao§[1]

[1]Zapata Computing, Inc.

June 17, 2020

Abstract

The number of measurements demanded by hybrid quantum-classical algorithms such as the variational quantum eigensolver (VQE) is prohibitively high for many problems of practical value. Quantum algorithms that reduce this cost (e.g. quantum amplitude and phase estimation) require error rates that are too low for near-term implementation. Here we propose methods that take advantage of the available quantum coherence to maximally enhance the power of sampling on noisy quantum devices, reducing measurement number and runtime compared to VQE. Our scheme derives inspiration from quantum metrology, phase estimation, and the more recent "alpha-VQE" proposal, arriving at a general formulation that is robust to error and does not require ancilla qubits. The central object of this method is what we call the "engineered likelihood function" (ELF), used for carrying out Bayesian inference. We show how the ELF formalism enhances the rate of information gain in sampling as the physical hardware transitions from the regime of noisy intermediate-scale quantum computers into that of quantum error corrected ones. This technique speeds up a central component of many quantum algorithms, with applications including chemistry, materials, finance, and beyond. Similar to VQE, we expect small-scale implementations to be realizable on today's quantum devices.

Contents

*guoming.wang@zapatacomputing.com
†dax.koh@zapatacomputing.com
‡peter@zapatacomputing.com
§yudong@zapatacomputing.com

1 Introduction

Which quantum algorithms will deliver practical value first? A recent flurry of methods that cater to the limitations of near-term quantum devices have drawn significant attention. These methods include the variational quantum eigensolver (VQE) [1–4], quantum approximate optimization algorithm (QAOA) [5] and variants [6], variational quantum linear systems solver [7–9], other quantum algorithms leveraging the variational principles [10], and quantum machine learning algorithms [11–13]. In spite of such algorithmic innovations, many of these approaches have appeared to be impractical for commercially-relevant problems owing to their high cost in terms of number of measurements [2,14] and hence runtime. Yet, methods offering a quadratic speedup in runtime, such as phase estimation, demand quantum resources that are far beyond the reach of near-term devices for moderately large problem instances [15].

Recently, the method of "α-VQE" [16] was proposed for interpolating between VQE and phase estimation in terms of the asymptotic tradeoff between sample count and quantum coherence. The basic idea is to start from the general framework of VQE, namely the iterative optimization of the energy expectation which is a sum of individual operator expectations, and proceed to estimate each individual operator with a Bayesian variant of the overlap estimation algorithm [17] that shares the same statistical inference backbone with known Bayesian parameter estimation schemes [18–21]. While phase estimation is commonly regarded as a quantum algorithm intended for fault tolerant quantum computers, previous works [16, 21] have demonstrated that in a noisy setting, Bayesian phase estimation can still yield quantum advantages in sampling. For instance, in [21, Sec. IIIA] it is shown that depolarizing noise reduces but does not eliminate the ability of the likelihood function to distinguish between different possible values of the parameter to be estimated.

This motivates the central question of our work: with realistic, *noisy* quantum computers, how do we maximize information gain from the coherence available to speed up operator expectation estimation, and in doing so, speed up algorithms such as VQE that rely on sampling? We note that this question is not only urgently relevant in the current era of noisy quantum devices without quantum error correction, but remains relevant for error-corrected quantum computation.

1.1 Prior work

Evaluating the expectation value of an operator O with respect to a quantum state $|A\rangle$ is a fundamental element of quantum information processing. In the simplest setting where samples are drawn repeatedly from measuring the same operator O on the same quantum state $|A\rangle$, the measurement process is equivalent to a sequence of independent Bernoulli experiments. For yielding an estimate of the expectation value within error ε (with high probability), the number of samples scales as $O(1/\varepsilon^2)$. We highlight the following key points regarding overlap estimation that are relevant to the context of this work:

1. **Cost scaling improvement using phase estimation**. Quantum effects introduce the opportunity to asymptotically accelerate the measurement process. In particular, there is a set of schemes [17] based on quantum phase estimation that is able to reduce the sample complexity to $O(\log \frac{1}{\varepsilon})$. This saturates the information theoretical lower bound for the number of samples since in order to determine a bounded quantity to resolution ε one must be able to distinguish $O(1/\varepsilon)$ different values, requiring at least $O(\log \frac{1}{\varepsilon})$ bits [20]. However, such optimal sample complexity is at a cost of $O(1/\varepsilon)$ amount of coherent quantum operations. This tradeoff between sample complexity and quantum coherence is also well understood in quantum metrology [22, 23].

2. **Overlap estimation as an amplitude estimation problem.** The connection between phase estimation and overlap estimation is solidified in [17], where the task of estimating an expectation $\Pi = \langle A|O|A\rangle$ is cast as a quantum counting [24] problem of estimating the angle θ between $|A\rangle$ and $O|A\rangle$ such that $\cos\theta = \langle A|O|A\rangle$. By alternating applications of reflection operators with respect to each of the states $|A\rangle$ and $O|A\rangle$ one can enact rotations in the two-dimensional subspace spanned by the two states. The same geometric picture also arises in Grover's search algorithm. It has also been demonstrated [25] that using generalized reflection operators (namely those with a phase parameter φ such that $R_\varphi = I - (1 - e^{i\varphi})|A\rangle\langle A|$ instead of the common reflection operator $R = I - 2|A\rangle\langle A|$), one can realize a much larger set of SU(2) rotations than with only common reflection operators. The set of SU(2) rotations realizable with such generalized construction has also been rigorously characterized [26] and later used for some of the most advanced Hamiltonian simulation algorithms such as qubitization [27] and signal processing [28].

3. **Bayesian inference perspective.** The problem of overlap estimation can be framed as a parameter estimation problem, common in statistical inference. In fact, previous work (for example [3, Sec. IVA]) has already pointed out a Bayesian perspective for considering the standard sampling process for VQE algorithms. The general setting is firstly to treat the operator expectation $\Pi = \langle A|O|A\rangle$ as parameter to be estimated. Then, a parametrized quantum circuit $V(\vec{\theta})$ that may be related to $|A\rangle$ and O is constructed. The ability to execute the circuit and collect measurement outcome d translates to the ability to sample from a likelihood function $p(d|\vec{\theta}, \Pi)$. For a given prior $p(\Pi)$ representing the current belief of the true value of Π, Bayesian inference incorporates a new measurement outcome d' and produces (or updates the prior to) a posterior distribution $p(\Pi|d) = \frac{p(d|\Pi,\vec{\theta})p(\Pi)}{\int p(d|\Pi,\vec{\theta})p(\Pi)d\Pi}$. For the settings considered in this paper, as well as in previous works [18–21], the prior and posterior distributions are maintained on the classical computer, while sampling from the likelihood function involves using a quantum device.

The combination of phase estimation and the Bayesian perspective gives rise to Bayesian phase estimation techniques [20, 21, 29] that are more suitable for noisy quantum devices capable of realizing limited depth quantum circuits than the early proposals [30]. Adopting the notation from point 3 above, the circuit parameters $\vec{\theta} = (m, \beta)$ and the goal is to estimate the phase Π in an eigenvalue $e^{i \arccos \Pi}$ of the operator U. An important note is that the likelihood function here,

$$p(d|m, \beta, \Pi) = \frac{1}{2}\left(1 + (-1)^d[\cos(\beta)\mathcal{T}_m(\Pi) + \sin(\beta)\mathcal{U}_m(\Pi)]\right), \tag{1}$$

where $\mathcal{T}_m$ and $\mathcal{U}_m$ are Chebyshev polynomials of first and second kind respectively, is shared in many settings beyond Bayesian phase estimation (c.f. [18, Eq. 2], [19, Eq. 1], [21, Eq. 2], and [16, Eq. 4]). This commonality makes the Bayesian inference machinery used for tasks such as Hamiltonian characterization [18, 19] relevant to phase estimation. In [19] the exponential advantage of Bayesian inference with a Gaussian prior over

other non-adaptive sampling methods is established by showing that the expected posterior variance σ decays *exponentially* in the number of inference steps. Such exponential convergence is at a cost of $O(1/\sigma)$ amount of quantum coherence required at each inference step [19]. Such scaling is also confirmed in [21] in the context of Bayesian phase estimation.

Equipped with the techniques of Bayesian phase estimation as well as the perspective of overlap estimation as an amplitude estimation problem (point 2 above), one may devise a Bayesian inference method for operator measurement that smoothly interpolates between the standard sampling regime and phase estimation regime. This is proposed in [31] as "α-VQE", where the asymptotic scaling for performing an operator measurement is $O(1/\varepsilon^{\alpha})$ with the extremal values of $\alpha = 2$ corresponding to the standard sampling regime (typically realized in VQE) and $\alpha = 1$ corresponding to the quantum-enhanced regime where the scaling reaches the Heisenberg limit (typically realized with phase estimation). By varying the parameters for the Bayesian inference one can also achieve α values between 1 and 2. The lower the α value, the deeper the quantum circuit needed for Bayesian phase estimation. This accomplishes the tradeoff between quantum coherence and asymptotic speedup for the measurement process (point 1 above).

It is also worth noting that phase estimation is not the only paradigm that can reach the Heisenberg limit for amplitude estimation [32–34]. In [32] the authors consider the task of estimating the parameter θ of a quantum state ρ_θ. A parallel strategy is proposed where m copies of the parametrized circuit for generating ρ_θ, together with an entangled initial state and measurements in an entangled basis, are used to create states with the parameter θ amplified to $m\theta$. Such amplification can also give rise to likelihood functions that are similar to that in Equation 1. In [33] it is shown that with randomized quantum operations and Bayesian inference one can extract information in fewer iteration than classical sampling even in the presence of noise. In [34] quantum amplitude estimation circuits with varying numbers m of iterations and numbers N of measurements are considered. A particularly chosen set of pairs (m, N) gives rise to a likelihood function that can be used for inferring the amplitude to be estimated. The Heisenberg limit is demonstrated for one particular likelihood function construction given by the authors. Both works [33, 34] highlight the power of parametrized likelihood functions, making it tempting to investigate their performance under imperfect hardware conditions. As will become clear, although the methods we propose can achieve Heisenberg-limit scaling, they do not take the perspective of many previous works that consider interfering many copies of the same physical probe.

1.2 Main results

The focus of this work is on estimating the expectation $\Pi = \langle A|O|A\rangle$ where the state $|A\rangle$ can be prepared by a circuit A such that $|A\rangle = A|0\rangle$. We consider a family of quantum circuits such that as the circuit deepens with more repetitions of A it allows for likelihood functions that are polynomial in Π of ever higher degree. As we will demonstrate in the next section with a concrete example, a direct consequence of this increase in polynomial degree is an increase in the power of inference, which can be quantified by Fisher information gain at each inference step. After establishing this "enhanced sampling" technique, we further introduce parameters into the quantum circuit and render the resulting likelihood function tunable. We then optimize the parameters for maximal information gain during each step of inference. The following lines of insight emerge from our efforts:

1. **The role of noise and error in amplitude estimation:** Previous works [16,21,29,33] have revealed the impact of noise on the likelihood function and the output estimation of the Hamiltonian spectrum. Here we investigate the same for our scheme of amplitude estimation. Our findings show that while noise and error does increase the runtime needed for producing an output that is within a specific statistical error tolerance, they do not necessarily introduce systematic bias in the output of the estimation algorithm. Systematic bias in the estimate can be suppressed by using active noise-tailoring techniques [35] and calibrating the effect of noise.

   We have performed simulation using realistic error parameters for near-term devices and discovered that the enhanced sampling scheme can outperform VQE in terms of sampling efficiency. Our results have also revealed a perspective on tolerating error in quantum algorithm implementation where higher fidelity does not necessarily lead to better algorithmic performance. For fixed gate fidelity, there appears to be an optimal circuit fidelity around the range of $0.5 - 0.7$ at which the enhanced scheme yields the maximum amount of quantum speedup.

Scheme	Bayesian inference	Noise consideration	Fully tunable LFs	Requires ancilla	Requires eigenstate
Knill et al. [17]	No	No	No	Yes	No
Svore et al. [20]	No	No	No	Yes	Yes
Wiebe and Grenade [21]	Yes	Yes	No	Yes	Yes
Wang et al. [16]	Yes	Yes	No	Yes	Yes
O'Brien et al. [29]	Yes	Yes	No	Yes	No
Zintchenko and Wiebe [33]	No	Yes	No	No	No
Suzuki et al. [34]	No	No	No	No	No
This work (Section 3)	Yes	Yes	Yes	No	No
This work (Appendix B)	Yes	Yes	Yes	Yes	No

Table 1: Comparison of our scheme with relevant proposals that appear in the literature. Here the list of features include whether the quantum circuit used in the scheme requires ancilla qubits in addition to qubits holding the state for overlap estimation, whether the scheme uses Bayesian inference, whether any noise resilience is considered, whether the initial state is required to be an eigenstate, and whether the likelihood function is fully tunable like ELF proposed here or restricted to Chebyshev likelihood functions.

2. **The role of likelihood function tunability:** Parametrized likelihood functions are centerpieces of phase estimation or amplitude estimation routines. To our knowledge, all of the current methods focus on likelihood functions of the Chebyshev form (Equation 1). For these Chebyshev likelihood functions (CLF) we observe that in the presence of noise there are specific values of the parameter Π (the "dead spots") for which the CLFs are significantly less effective for inference than other values of Π. We remove those dead spots by engineering the form of the likelihood function with generalized reflection operators (point 2 in Section 1) whose angle parameters are made tunable.

3. **Runtime model for estimation as error rates decrease:** Previous works [16, 34] have demonstrated smooth transitions in the asymptotic cost scaling from the $O(1/\varepsilon^2)$ of VQE to $O(1/\varepsilon)$ of phase estimation. We advance this line of thinking by developing a model for estimating the runtime t_ε to target accuracy ε using devices with degree of noise λ (c.f. Section 6):

$$t_\varepsilon \sim O\left(\frac{\lambda}{\varepsilon^2} + \frac{1}{\sqrt{2}\varepsilon} + \sqrt{\left(\frac{\lambda}{\varepsilon^2}\right)^2 + \left(\frac{2\sqrt{2}}{\varepsilon}\right)^2}\right). \tag{2}$$

   The model interpolates between the $O(1/\varepsilon)$ scaling and $O(1/\varepsilon^2)$ scaling as a function of λ. Such bounds also allow us to make concrete statements about the extent of quantum speedup as a function of hardware specifications such as the number of qubits and two-qubit fidelity, and therefore estimate runtimes using realistic parameters for current and future hardware.

The remainder of the paper is organized as the following. In Section 2 we present a concrete example of our scheme for readers who wish to glean only a first impression of the key ideas. Subsequent sections then expand on the general formulation of our scheme. Section 3 describes in detail the general quantum circuit construction for realizing ELFs, and analyzes the structure of ELF in both noisy and noiseless settings. In addition to the quantum circuit scheme, our scheme also involves 1) tuning the circuit parameter to maximize information gain, and 2) Bayesian inference for updating the current belief about the distribution of the true value of Π. Section 4 presents heuristic algorithms for both. We then show numerical results in Section 5 comparing our approach with existing methods based on CLFs. In Section 6 we construct a runtime model and derive the expression in (2). We conclude in Section 7 with implications of our results from a broad perspective of quantum computing.

2 A first example

There are two main strategies for estimating the expectation value $\langle A| P |A\rangle$ of some operator P with respect to a quantum state $|A\rangle$. The method of *quantum amplitude estimation* [24] provides a provable quantum

speedup with respect to certain computational models. However, to achieve precision ε in the estimate, the circuit depth needed in this method scales as $O(1/\varepsilon)$, making it impractical for near term quantum computers. The variational quantum eigensolver uses *standard sampling* to carry out amplitude estimation. Standard sampling allows for low-depth quantum circuits, making it more amenable to implementation on near term quantum computers. However, in practice, the inefficiency of this method makes VQE impractical for many problems of interest [2]. In this section we introduce the method of *enhanced sampling* for amplitude estimation. This technique draws inspiration from quantum-enhanced metrology [36] and seeks to maximize the statistical power of noisy quantum devices. We motivate this method by starting from a simple analysis of standard sampling as used in VQE. We note that, although the subroutine of estimation is a critical bottleneck, other aspects of the VQE algorithm also must be improved, including the optimization of the parameterized quantum circuit parameters [37, 38].

The energy estimation subroutine of VQE estimates amplitudes with respect to Pauli strings. For a Hamiltonian decomposed into a linear combination of Pauli strings $H = \sum_j \mu_j P_j$ and "ansatz state" $|A\rangle$, the energy expectation value is estimated as a linear combination of Pauli expecation value estimates

$$\hat{E} = \sum_j \mu_j \hat{\Pi}_j, \tag{3}$$

where $\hat{\Pi}_j$ is the (amplitude) estimate of $\langle A|P_j|A\rangle$. VQE uses the standard sampling method to build up Pauli expectation value estimates with respect to the ansatz state, which can be summarized as follows. Prepare $|A\rangle$ and measure operator P receiving outcome $d = 0, 1$. Repeat M times, receiving k outcomes labeled 0 and $M - k$ outcomes labeled 1. Estimate $\Pi = \langle A|P|A\rangle$ as $\hat{\Pi} = \frac{k-(M-k)}{M}$.

We can quantify the performance of this estimation strategy using the mean squared error of the estimator as a function of time $t = TM$, where T is the time cost of each measurement. Because the estimator is unbiased, the mean squared error is simply the variance in the estimator,

$$\mathrm{MSE}(\hat{\Pi}) = \frac{1 - \Pi^2}{M}. \tag{4}$$

For a specific mean squared error $\mathrm{MSE}(\hat{\Pi}) = \varepsilon^2$, the runtime of the algorithm needed to ensure mean squared error ε^2 is

$$t_\varepsilon = T\frac{1 - \Pi^2}{\varepsilon^2}. \tag{5}$$

The total runtime of energy estimation in VQE is the sum of the runtimes of the individual Pauli expectation value estimation runtimes. For problems of interest, this runtime can be far too costly, even when certain parallelization techniques are used [39]. The source of this cost is the insensitivity of the standard sampling estimation process to small deviations in Π: the expected information gain about Π contained in the standard-sampling measurement outcome data is low.

Generally, we can measure the information gain of an estimation process of M repetitions of standard sampling with the Fisher information

$$I_M(\Pi) = \mathbb{E}_D\left[\left(\frac{\partial}{\partial\Pi}\log\mathbb{P}(D|\Pi)\right)^2\right] = -\mathbb{E}_D\left[\frac{\partial^2}{\partial\Pi^2}\log\mathbb{P}(D|\Pi)\right] = \sum_D \frac{1}{\mathbb{P}(D|\Pi)}\left(\frac{\partial}{\partial\Pi}\mathbb{P}(D|\Pi)\right)^2, \tag{6}$$

where $D = \{d_1, d_2, \cdots, d_M\}$ is the set of outcomes from M repetitions of the standard sampling. The Fisher information identifies the likelihood function $\mathbb{P}(D|\Pi)$ as being responsible for information gain. We can lower bound the mean squared error of an (unbiased) estimator with the Cramer-Rao bound

$$\mathrm{MSE}(\hat{\Pi}) \geq \frac{1}{I_M(\Pi)}. \tag{7}$$

Using the fact that the Fisher information is additive in the number of samples, we have $I_M(\Pi) = MI_1(\Pi)$ where $I_1(\Pi) = 1/(1 - \Pi^2)$ is the Fisher information of a single sample drawn from likelihood function $\mathbb{P}(d|\Pi) = (1 + (-1)^d\Pi)/2$. Using the Cramer-Rao bound, we can find a lower bound for the runtime of the estimation process as

$$t_\varepsilon \geq \frac{T}{I_1(\Pi)\varepsilon^2}, \tag{8}$$

which shows that in order to reduce the runtime of an estimation algorithm we should aim to increase the Fisher information.

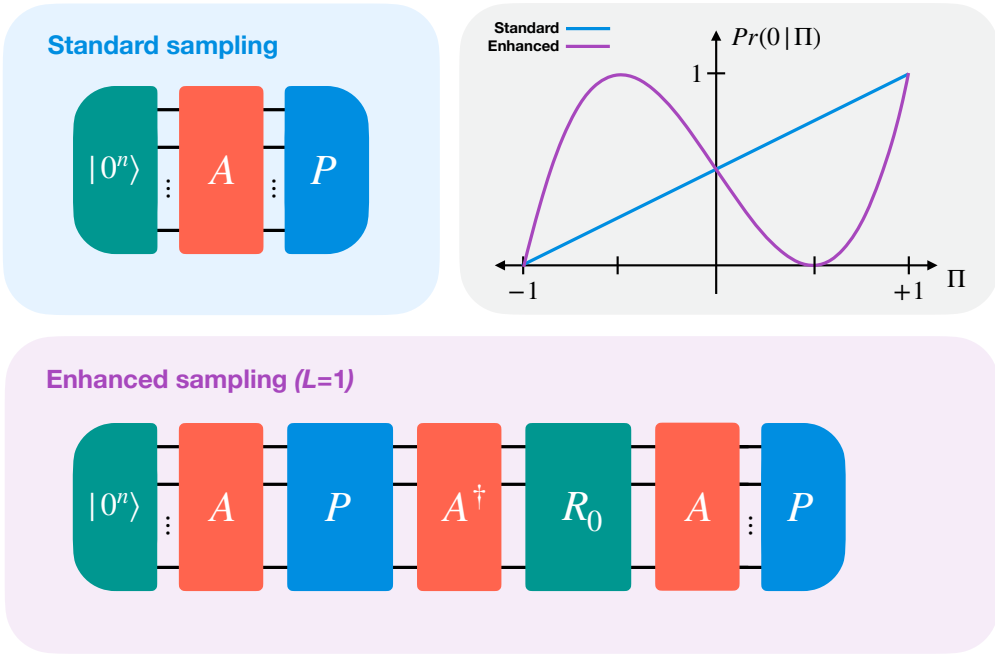


Figure 2.1: This figure exhibits the quantum circuits of standard sampling (used in VQE) and of the simplest non-trivial version of enhanced sampling, along with their corresponding likelihood functions. Enhanced sampling can yield a larger statistical power in this estimation of $\Pi = \langle A|\, P\,|A\rangle$. The likelihoods of the outcome data in enhanced sampling can depend more sensitively on the unknown value Π than they do in standard sampling. This increased sensitivity can reduce the runtime needed to achieve a target average error in the estimate of Π.

The purpose of enhanced sampling is to reduce the runtime of overlap estimation by engineering likelihood functions that increase the rate of information gain. We consider the simplest case of enhanced sampling, which is illustrated in Figure 2.1. To generate data we prepare the ansatz state $|A\rangle$, apply the operation P, apply a phase flip about the ansatz state, and then measure P. The phase flip about the ansatz state can be achieved by applying the inverse of the ansatz circuit A^{-1}, applying a phase flip about the initial state $R_0 = 2\left|0^N\right\rangle\left\langle 0^N\right| - I$, and then re-applying the ansatz circuit A. In this case, the likelihood function becomes

$$\mathbb{P}(d|\Pi) = \frac{1 + (-1)^d \cos(3\arccos(\Pi))}{2} = \frac{1 + (-1)^d(4\Pi^3 - 3\Pi)}{2}. \tag{9}$$

The bias is a degree-3 Chebyshev polynomial in Π. We will refer to such likelihood functions as *Chebyshev likelihood functions* (CLFs).

In order to compare the Chebyshev likelihood function of enhanced sampling to that of standard sampling, we consider the case of $\Pi = 0$. Here, $\mathbb{P}(0|\Pi = 0) = \mathbb{P}(1|\Pi = 0)$ and so the Fisher information is proportional to the square of the slope of the likelihood function

$$I_1(\Pi = 0) = \left(\frac{\partial \mathbb{P}(d = 0|\Pi)}{\partial \Pi}\right)^2. \tag{10}$$

As seen in Figure 2.1, the slope of the Chebyshev likelihood function at $\Pi = 0$ is steeper than that of the

standard sampling likelihood function. The single sample Fisher information in each case evaluates to

$$\begin{aligned} \text{Standard: } & I_1(\Pi = 0) = 1 \\ \text{Enhanced: } & I_1(\Pi = 0) = 9, \end{aligned} \tag{11}$$

demonstrating how a simple variant of the quantum circuit can enhance information gain. In this example, using the simplest case of enhanced sampling can reduce the number of measurements needed to achieve a target error by at least a factor of nine. As we will discuss later, we can further increase the Fisher information by applying L layers of $PA^\dagger R_0 A$ before measuring P. In fact, the Fisher information $I_1(\Pi) = \frac{(2L+1)^2}{1-\Pi^2} = O(L^2)$ grows quadratically in L.

We have yet to propose an estimation scheme that converts enhanced sampling measurement data into an estimation. One intricacy that enhanced sampling introduces is the option to vary L as we are collecting measurement data. In this case, given a set of measurement outcomes from circuits with varying L, the sample mean of the 0 and 1 counts loses its meaning. Instead of using the sample mean, to process the measurement outcomes into information about Π we use *Bayesian inference*. Section 3.2 describes the use of Bayesian inference for estimation.

At this point, one may be tempted to point out that the comparison between standard sampling and enhanced sampling is unfair because only one query to A is used in the standard sampling case while the enhanced sampling scheme uses three queries of A. It seems that if one considers a likelihood function that arises from *three* standard sampling steps, one could also yield a cubic polynomial form in the likelihood function. Indeed, suppose one performs three independent standard sampling steps yielding results $x_1, x_2, x_3 \in \{0, 1\}$, and produces a binary outcome $z \in \{0, 1\}$ classically by sampling from a distribution $\mathbb{P}(z|x_1, x_2, x_3)$. Then the likelihood function takes the form of

$$\mathbb{P}(z|\Pi) = \sum_{x_1,x_2,x_3} \mathbb{P}(z|x_1, x_2, x_3)\mathbb{P}(x_1, x_2, x_3|\Pi) = \sum_{i=0}^{3} \alpha_i \binom{3}{i} \left(\frac{1+\Pi}{2}\right)^i \left(\frac{1-\Pi}{2}\right)^{3-i}, \tag{12}$$

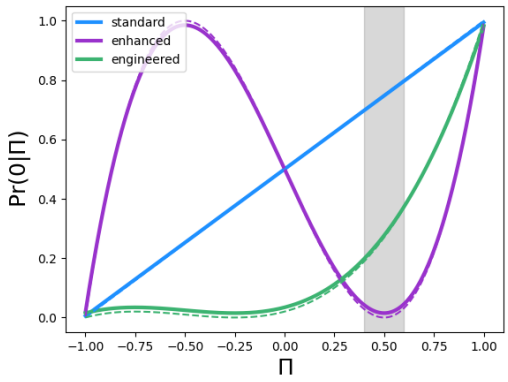
where each $\alpha_i \in [0, 1]$ is a parameter that can be tuned classically through changing the distribution $\mathbb{P}(z|x_1, x_2, x_3)$. More specifically, $\alpha_i = \sum_{x_1x_2x_3:h(x_1x_2x_3)=i} \mathbb{P}(z|x_1, x_2, x_3)$ where $h(x_1x_2x_3)$ is the Hamming weight of the bit string $x_1x_2x_3$. Suppose we want $\mathbb{P}(z = 0|\Pi)$ to be equal to $\mathbb{P}(d = 0|\Pi)$ in Equation 9. This implies that $\alpha_0 = 1$, $\alpha_1 = -2$, $\alpha_2 = 3$ and $\alpha_3 = 0$, which is clearly beyond the classical tunability of the likelihood function in Equation 12. This evidence suggests that the likelihood function arising from the quantum scheme in Equation 9 is beyond classical means.

As the number of circuit layers L is increased, the time per sample T grows linearly in L. This linear growth in circuit layer number, along with the quadratic growth in Fisher information leads to a lower bound on the expected runtime,

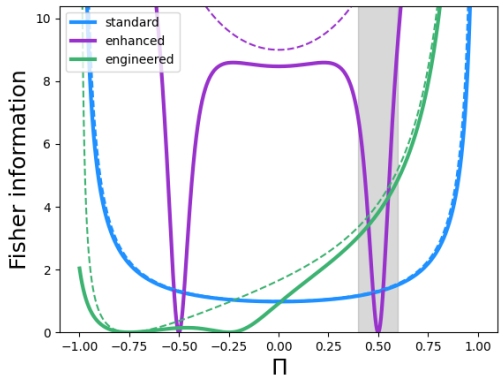
$$t_\varepsilon \in \Omega\left(\frac{1}{L\varepsilon^2}\right), \tag{13}$$

assuming a fixed-L estimation strategy with an unbiased estimator. In practice, the operations implemented on the quantum computer are subject to error. Fortunately, Bayesian inference can incorporate such errors into the estimation process. As long as the influence of errors on the form of the likelihood function is accurately modeled, the principal effect of such errors is only to slow the rate of information gain. Error in the quantum circuit accumulates as we increase the number of circuit layers L. Consequently, beyond a certain number of circuit layers, we will receive diminishing returns with respect to gains in Fisher information (or the reduction in runtime). The estimation algorithm should then seek to balance these competing factors in order to optimize the overall performance.

The introduction of error poses another issue for estimation. Without error, the Fisher information gain per sample in the enhanced sampling case with $L = 1$ is greater than or equal to 9 for all Π. As shown in Figure 2.2, with the introduction of even a small degree of error, the values of Π where the likelihood function is flat incur a dramatic drop in Fisher information. We refer to such regions as estimation *dead spots*. This observation motivates the concept of engineering likelihood functions (ELF) to increase their statistical power. By promoting the P and R_0 operations to generalized reflections $U(x_1) = \exp(-ix_1P)$ and $R_0(y_2) = \exp(-ix_2R_0)$ we can choose rotation angles such that the information gain is boosted around such dead spots. We will find that even for deeper enhanced sampling circuits, engineering likelihood functions allows us to mitigate the effect of estimation dead spots.



(a) Likelihood function comparison.



(b) Fisher information comparison

Figure 2.2: Dotted curves are the noiseless case, solid curves incorporate a 1% error per ansatz application. These figures demonstrate improvement in information gain if the likelihood function is engineered. The enhanced sampling likelihood function yields a large Fisher information for many values of Π relative to that of standard sampling. The introduction of even a small degree of error in the quantum circuits causes the Fisher information to become zero where the enhanced sampling likelihood function is flat (indicated by the grey bands). By tuning the generalized reflection angles ($(x_1, x_2) = (-0.6847, 0.6847)$), we can engineer a likelihood function to boost the information gain in the estimation "dead spot" (gray region).

3 Engineered Likelihood Functions

In this section, we propose the methodology of engineering likelihood functions for amplitude estimation. We first introduce the quantum circuits for drawing samples that correspond to engineered likelihood functions, and then describe how to tune the circuit parameters and carry out Bayesian inference with the resultant likelihood functions.

3.1 Quantum circuits for engineered likelihood functions

Our objective is to design a procedure for estimating the expectation value

$$\Pi = \cos(\theta) = \langle A| P |A\rangle, \tag{14}$$

where $|A\rangle = A|0^n\rangle$ in which A is an n-qubit unitary operator, P is an n-qubit Hermitian operator with eigenvalues ± 1, and $\theta = \arccos(\Pi)$ is introduced to facilitate Bayesian inference later on. In constructing our estimation algorithms, we assume that we are able to perform the following primitive operations. First, we can prepare the computational basis state $|0^n\rangle$ and apply a circuit A to it, obtaining $|A\rangle = A|0^n\rangle$. Second, we can implement the unitary operator $U(x) = \exp(-ixP)$ for any angle $x \in \mathbb{R}$. Finally, we can perform the measurement of P which is modeled as a projection-valued measure $\{\frac{I+P}{2}, \frac{I-P}{2}\}$ with respective outcome labels $\{0, 1\}$. We will also make use of the unitary operator $V(y) = AR_0(y)A^\dagger$, where $R_0(y) = \exp(-iy(2|0^n\rangle\langle 0^n| - I))$ and $y \in \mathbb{R}$. Following the convention (see e.g. [26]), we will call $U(x)$ and $V(y)$ the *generalized reflections* about the $+1$ eigenspace of P and the state $|A\rangle$, respectively, where x and y are the *angles* of these generalized reflections, respectively.

We use the ancilla-free [1] quantum circuit in Fig. 3.1 to generate the *engineered likelihood function* (ELF), that is the probability distribution of the outcome $d \in \{0, 1\}$ given the unknown quantity θ to be estimated. The circuit consists of a sequence of generalized reflections. Specifically, after preparing the ansatz state $|A\rangle = A|0\rangle^{\otimes n}$, we apply $2L$ generalized reflections $U(x_1)$, $V(x_2)$, ..., $U(x_{2L-1})$, $V(x_{2L})$ to it, varying the rotation angle x_j in each operation. For convenience, we will call $V(x_{2j})U(x_{2j-1})$ the j-th layer of the circuit,

[1] We call this scheme "ancilla-free" (AF) since it does not involve any ancilla qubits. In Appendix B, we consider a different scheme named the "ancilla-based" (AB) scheme that involves one ancilla qubit.

for $j = 1, 2, \ldots, L$. The output state of this circuit is

$$Q(\vec{x})\,|A\rangle = V(x_{2L})U(x_{2L-1})\ldots V(x_2)U(x_1)\,|A\rangle\,, \tag{15}$$

where $\vec{x} = (x_1, x_2, \ldots, x_{2L-1}, x_{2L}) \in \mathbb{R}^{2L}$ is the vector of tunable parameters. Finally, we perform the projective measurement $\{\frac{I+P}{2}, \frac{I-P}{2}\}$ on this state, receiving an outcome $d \in \{0, 1\}$.

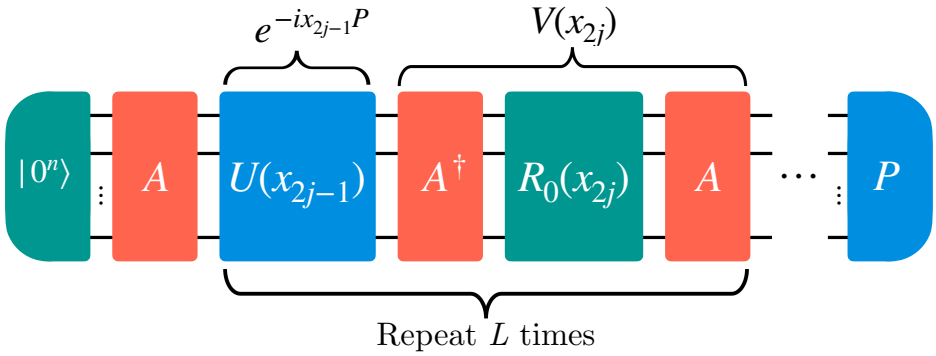


Figure 3.1: This figure depicts the operations used for generating samples that correspond to an engineered likelihood function. A is the state preparation circuit, P is the observable of interest, and $R_0(x_{i+1})$ is a generalized reflection about the state $|0^n\rangle$. The outcomes of measurement of P yield information about the expectation value $\Pi = \langle A|P|A\rangle$. The case of $L = 0$ simply prepares $|A\rangle$ and measures P. This corresponds to the standard sampling method used in VQE. Even with an error-prone implementation, we can enhance the information gain rate by applying a sequence of generalized reflections before the measurement. In such *enhanced sampling*, the likelihood of outcomes depends more sensitively on Π. These circuit elements are color-coded to highlight the commonalities in the way the features P (blue), A (red), and $|0^n\rangle$ (green) enter in the likelihood function.

As in Grover's search algorithm, the generalized reflections $U(x_{2j-1})$ and $V(x_{2j})$ ensure that the quantum state remains in two-dimensional subspace $S := \mathrm{span}\{|A\rangle\,, P\,|A\rangle\}$ [2] for any j. Let $|A^\perp\rangle$ be the state (unique, up to a phase) in S that is orthogonal to $|A\rangle$, i.e.

$$|A^\perp\rangle = \frac{P\,|A\rangle - \langle A|\,P\,|A\rangle\,|A\rangle}{\sqrt{1 - \langle A|\,P\,|A\rangle^2}}. \tag{16}$$

To help the analysis, we will view this two-dimensional subspace as a qubit, writing $|A\rangle$ and $|A^\perp\rangle$ as $|\bar{0}\rangle$ and $|\bar{1}\rangle$, respectively. Let $\bar{X}$, $\bar{Y}$, $\bar{Z}$ and $\bar{I}$ be the Pauli operators and identity operator on this virtual qubit, respectively. Then, focusing on the subspace $S = \mathrm{span}\{|\bar{0}\rangle\,, |\bar{1}\rangle\}$, we can rewrite P as

$$P(\theta) = \cos(\theta)\bar{Z} + \sin(\theta)\bar{X}, \tag{17}$$

and rewrite the generalized reflections $U(x_{2j-1})$ and $V(x_{2j})$ as

$$U(\theta; x_{2j-1}) = \cos\left(x_{2j-1}\right)\bar{I} - \mathrm{i}\sin\left(x_{2j-1}\right)\left(\cos\left(\theta\right)\bar{Z} + \sin\left(\theta\right)\bar{X}\right) \tag{18}$$

and

$$V(x_{2j}) = \cos\left(x_{2j}\right)\bar{I} - \mathrm{i}\sin\left(x_{2j}\right)\bar{Z}, \tag{19}$$

where $x_{2j-1}, x_{2j} \in \mathbb{R}$ are tunable parameters. Then the unitary operator $Q(\vec{x})$ implemented by the L-layer circuit becomes

$$Q(\theta; \vec{x}) = V(x_{2L})U(\theta; x_{2L-1})\ldots V(x_2)U(\theta; x_1). \tag{20}$$

[2] To ensure that S is two-dimensional, we assume that $\Pi \neq \pm 1$, i.e. $\theta \neq 0$ or π.

Note that in this picture, $|A\rangle = |\bar{0}\rangle$ is fixed, while $P = P(\theta)$, $U(x) = U(\theta; x)$ and $Q(\vec{x}) = Q(\theta; \vec{x})$ depend on the unknown quantity θ. It turns out to be more convenient to design and analyze the estimation algorithms in this "logical" picture than in the original "physical" picture. Therefore, we will stick to this picture for the remainder of this paper.

The engineered likelihood function (i.e. the probability distribution of measurement outcome $d \in \{0, 1\}$) depends on the output state $\rho(\theta; \vec{x})$ of the circuit and the observable $P(\theta)$. Precisely, it is

$$\mathbb{P}(d|\theta; \vec{x}) = \frac{1 + (-1)^d \Delta(\theta; \vec{x})}{2}, \tag{21}$$

where

$$\Delta(\theta; \vec{x}) = \langle\bar{0}| Q^\dagger(\theta; \vec{x}) P(\theta) Q(\theta; \vec{x}) |\bar{0}\rangle \tag{22}$$

is the *bias* of the likelihood function (from now on, we will use $\mathbb{P}'(d|\theta; \vec{x})$ and $\Delta'(\theta; \vec{x})$ to denote the derivatives of $\mathbb{P}(d|\theta; \vec{x})$ and $\Delta(\theta; \vec{x})$ with respect to θ, respectively). In particular, if $\vec{x} = (\frac{\pi}{2}, \frac{\pi}{2}, \ldots, \frac{\pi}{2}, \frac{\pi}{2})$, then we have $\Delta(\theta; \vec{x}) = \cos((2L + 1)\theta)$. Namely, the bias of the likelihood function for this $\vec{x}$ is the Chebyshev polynomial of degree $2L + 1$ (of the first kind) of Π. For this reason, we will call the likelihood function for this $\vec{x}$ the *Chebyshev likelihood function* (CLF). In Section 5 we will explore the performance gap between CLFs and general ELFs.

In reality, quantum devices are subject to noise. To make the estimation process robust against errors, we incorporate the following noise model into the likelihood function [21] [3]. We assume that the noisy version of each circuit layer $V(x_{2j})U(\theta; x_{2j-1})$ implements a mixture of the target operation and the completely depolarizing channel [4] acting on the same input state, i.e.

$$\mathcal{U}_j(\rho) = pV(x_{2j})U(\theta; x_{2j-1})\rho U^\dagger(\theta; x_{2j-1})V^\dagger(x_{2j}) + (1 - p)\frac{I}{2^n}, \tag{23}$$

where p is the *fidelity* of this layer. Under composition of such imperfect operations, the output state of the L-layer circuit becomes

$$\rho_L = p^L Q(\theta; \vec{x}) |A\rangle \langle A| Q^\dagger(\theta; \vec{x}) + (1 - p^L)\frac{I}{2^n}. \tag{24}$$

This imperfect circuit is preceded by an imperfect preparation of $|A\rangle$ and followed by an imperfect measurement of P. In the context of randomized benchmarking, such errors are referred to as *state preparation and measurement* (SPAM) errors [42]. We will also model SPAM error with a depolarizing model, taking the noisy preparation of $|A\rangle$ to be $p_{SP} |A\rangle \langle A| + (1 - p_{SP})\frac{I}{2^n}$ and taking the noisy measurement of P to be the POVM $\{p_M \frac{I+P}{2} + (1-p_M)\frac{I}{2}, p_M \frac{I-P}{2} + (1-p_M)\frac{I}{2}\}$. Combining the SPAM error parameters into $\bar{p} = p_{SP} p_M$, we arrive at a model for the noisy likelihood function

$$\mathbb{P}(d|\theta; f, \vec{x}) = \frac{1}{2}\left[1 + (-1)^d f \Delta(\theta; \vec{x})\right], \tag{25}$$

where $f = \bar{p}p^L$ is the *fidelity* of the whole process for generating the ELF, and $\Delta(\theta, \vec{x})$ is the bias of the ideal likelihood function as defined in Eq. (22) (from now on, we will use $\mathbb{P}'(d|\theta; f, \vec{x})$ to denote the derivative of $\mathbb{P}(d|\theta; f, \vec{x})$ with respect to θ). Note that the overall effect of noise on the ELF is that it rescales the bias by a factor of f. This implies that the less errored the generation process is, the steeper the resultant ELF is (which means it is more useful for Bayesian inference), as one would expect.

Before moving on to the discussion of Bayesian inference with ELFs, it is worth mentioning the following property of engineered likelihood functions, as it will play a pivotal role in Section 4. In [43], we introduced

[3] In practice, the establishment of the noise model requires a procedure for *calibrating the likelihood function* for the specific device being used. With respect to Bayesian inference, the parameters of this model are known as *nuisance parameters* [40, 41]; the target parameter does not depend directly on them, but they determine how the data relates to the target parameter and, hence, should be incorporated into the inference process. We will explore likelihood function calibration in future work. For the remainder of this article, we will assume that the noise model has been calibrated to sufficient precision so as to render the effect of model error negligible.

[4] The depolarizing model assumes that the gates comprising each layer are sufficiently random to prevent systematic build-up of coherent error. There exist techniques such as randomized compiling [35] that make this depolarizing model more accurate.

the concepts of *trigono-multilinear* and *trigono-multiquadratic* functions. Basically, a multivariable function $f : \mathbb{R}^k \to \mathbb{C}$ is trigono-multilinear if for any $j \in \{1, 2, \ldots, k\}$, we can write $f(x_1, x_2, \ldots, x_k)$ as

$$f(x_1, x_2, \ldots, x_k) = C_j(\vec{x}_{\neg j}) \cos(x_j) + S_j(\vec{x}_{\neg j}) \sin(x_j), \tag{26}$$

for some (complex-valued) functions C_j and S_j of $\vec{x}_{\neg j} := (x_1, \ldots, x_{j-1}, x_{j+1}, x_k)$, and we call C_j and S_j the *cosine-sine-decomposition (CSD)* coefficient functions of f with respect to x_j. Similarly, a multivariable function $f : \mathbb{R}^k \to \mathbb{C}$ is trigono-multiquadratic if for any $j \in \{1, 2, \ldots, k\}$, we can write $f(x_1, x_2, \ldots, x_k)$ as

$$f(x_1, x_2, \ldots, x_k) = C_j(\vec{x}_{\neg j}) \cos(2x_j) + S_j(\vec{x}_{\neg j}) \sin(2x_j) + B_j(\vec{x}_{\neg j}), \tag{27}$$

for some (complex-valued) functions C_j, S_j and B_j of $\vec{x}_{\neg j} := (x_1, \ldots, x_{j-1}, x_{j+1}, x_k)$, and we call C_j, S_j and B_j the *cosine-sine-bias-decomposition (CSBD)* coefficient functions of f with respect to x_j. The concepts of trigono-multilinearity and trigono-multiquadraticity can be naturally generalized to linear operators. Namely, a linear operator is trigono-multilinear (or trigono-multiquadratic) in a set of variables if each entry of this operator (written in an arbitrary basis) is trigono-multilinear (or trigono-multiquadratic) in the same variables. Now Eqs. (18), (19) and (20) imply that $Q(\theta; \vec{x})$ is a trigono-multilinear operator of $\vec{x}$. Then it follows from Eq. (22) that $\Delta(\theta; \vec{x})$ is a trigono-multiquadratic function of $\vec{x}$. Furthermore, we will show in Section 4.1 that the CSBD coefficient functions of $\Delta(\theta; \vec{x})$ with respect to any x_j can be evaluated in $O(L)$ time, and this greatly facilitates the construction of the algorithms in Section 4.1 for tuning the circuit angles $\vec{x} = (x_1, x_2, \ldots, x_{2L-1}, x_{2L})$.

3.2 Bayesian inference with engineered likelihood functions

With the model of (noisy) engineered likelihood functions in place, we are now ready to describe our methodology for tuning the circuit parameters $\vec{x}$ and performing Bayesian inference with the resultant likelihood functions for amplitude estimation.

Let us begin with a high-level overview of our algorithm for estimating $\Pi = \cos(\theta) = \langle A| P |A\rangle$. For convenience, our algorithm mainly works with $\theta = \arccos(\Pi)$ rather than with Π. We use a Gaussian distribution to represent our knowledge of θ and make this distribution gradually converge to the true value of θ as the inference process proceeds. We start with an initial distribution of Π (which can be generated by standard sampling or domain knowledge) and convert it to the initial distribution of θ. Then we iterate the following procedure until a convergence criterion is satisfied. At each round, we find the circuit parameters $\vec{x}$ that maximize the information gain from the measurement outcome d in certain sense (based on our current knowledge of θ). Then we run the quantum circuit in Fig. 3.1 with the optimized parameters $\vec{x}$ and receive a measurement outcome $d \in \{0, 1\}$. Finally, we update the distribution of θ by using Bayes rule, conditioned on d. Once this loop is finished, we convert the final distribution of θ to the final distribution of Π, and set the mean of this distribution as the final estimate of Π. See Fig. 3.2 for the conceptual diagram of this algorithm.

Next, we describe each component of the above algorithm in more detail. Throughout the inference process, we use a Gaussian distribution to keep track of our belief of the value of θ. Namely, at each round, θ has prior distribution

$$p(\theta) = p(\theta; \mu, \sigma) := \frac{1}{\sqrt{2\pi}\sigma} \mathrm{e}^{-\frac{(\theta-\mu)^2}{2\sigma^2}} \tag{28}$$

for some prior mean $\mu \in \mathbb{R}$ and prior variance $\sigma^2 \in \mathbb{R}^+$. After receiving the measurement outcome d, we compute the posterior distribution of θ by using Bayes rule

$$p(\theta|d; f, \vec{x}) = \frac{\mathbb{P}(d|\theta; f, \vec{x}) p(\theta)}{\mathbb{P}(d; f, \vec{x})}, \tag{29}$$

where the normalization factor, or model evidence, is defined as $\mathbb{P}(d; f, \vec{x}) = \int \mathrm{d}\theta \mathbb{P}(d|\theta; f, \vec{x}) p(\theta)$ (recall that f is the fidelity of the process for generating the ELF). Although the true posterior distribution will not be a Gaussian, we will approximate it as such. Following the methodology in [44], we replace the true posterior with a Gaussian distribution of the same mean and variance [5], and set it as the prior of θ for the next

[5] Although we can compute the mean and variance of the posterior distribution $p(\theta|d; f, \vec{x})$ directly by definition, this approach is time-consuming, as it involves numerical integration. Instead, we accelerate this process by taking advantage of certain property of engineered likelihood functions. See Section 4.2 for more details.

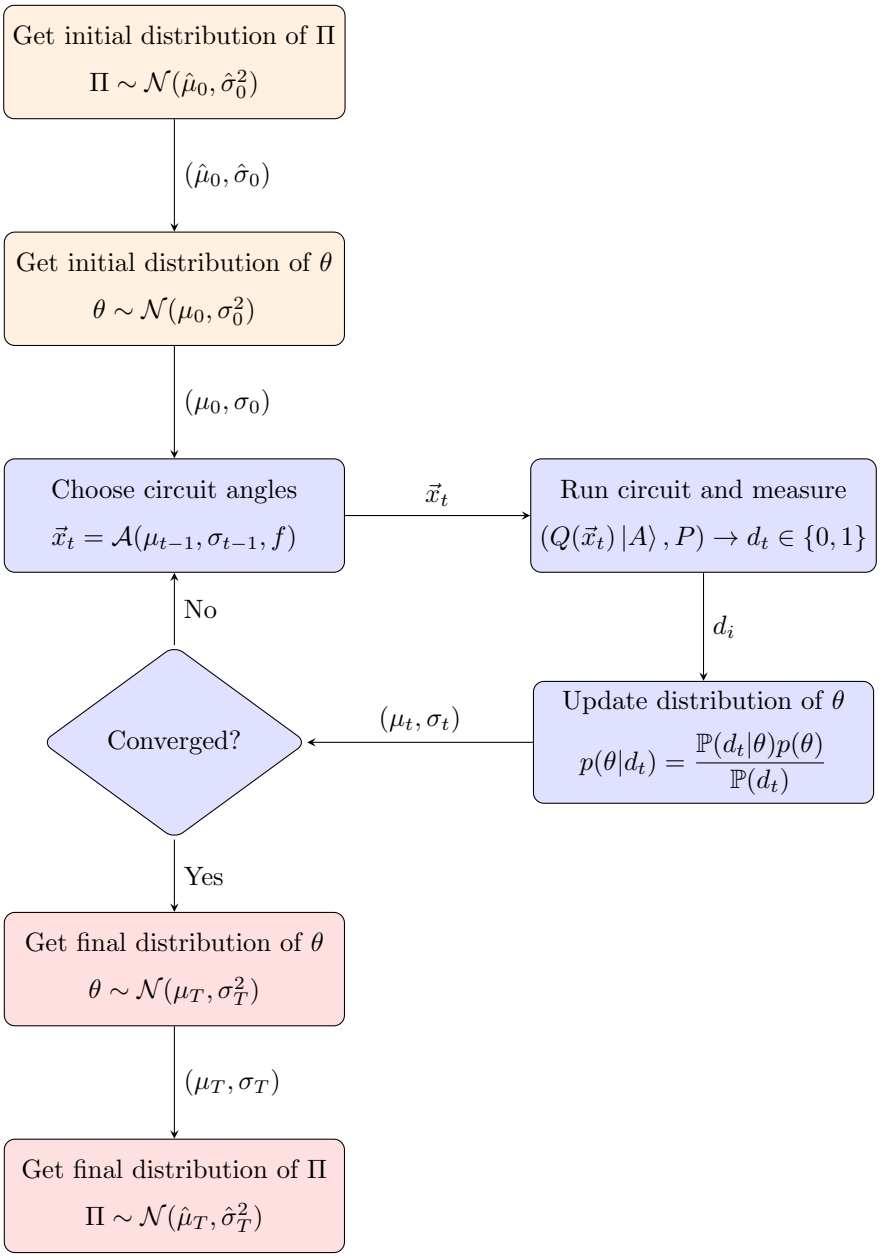


Figure 3.2: High-level description of the algorithm for estimating $\Pi = \cos(\theta) = \langle A| P |A\rangle$. Here f is the fidelity of the process for generating the ELF. This algorithm mainly works with θ instead of Π, and there are conversions between the distributions of θ and Π at the beginning and end of the algorithm. The final estimate of Π is $\hat{\mu}_T$. Note that only the "Run circuit and measure" step involves a quantum device.

round. We repeat this measurement-and-Bayesian-update procedure until the distribution of θ is sufficiently concentrated around a single value.

Since the algorithm mainly works with θ and we are eventually interested in Π, we need to make conversions between the estimators of θ and Π. This is done as follows. Suppose that at round t the prior distribution of θ is $\mathcal{N}(\mu_t, \sigma_t^2)$ and the prior distribution of Π is $\mathcal{N}(\hat{\mu}_t, \hat{\sigma}_t^2)$ (note that μ_t, σ_t, $\hat{\mu}_t$ and $\hat{\sigma}$ are random variables as they depend on the history of random measurement outcomes up to time t). The estimators of θ and Π at this round are μ_t and $\hat{\mu}_t$, respectively. Given the distribution $\mathcal{N}(\mu_t, \sigma_t^2)$ of θ, we compute the mean $\hat{\mu}_t$ and variance $\hat{\sigma}_t^2$ of $\cos(\theta)$, and set $\mathcal{N}(\hat{\mu}_t, \hat{\sigma}_t^2)$ as the distribution of Π. This step can

be done analytically, as if $X \sim \mathcal{N}(\mu, \sigma^2)$, then

$$\mathbb{E}[\cos(X)] = e^{-\frac{\sigma^2}{2}} \cos(\mu), \tag{30}$$

$$\mathrm{Var}[\cos(X)] = \frac{1}{2}\Big(1 - e^{-\sigma^2}\Big)\Big(1 - e^{-\sigma^2}\cos(2\mu)\Big). \tag{31}$$

Conversely, given the distribution $\mathcal{N}(\hat{\mu}_t, \hat{\sigma}_t^2)$ of Π, we compute the mean μ_t and variance σ_t^2 of $\arccos(\Pi)$ (clipping Π to $[-1, 1]$), and set $\mathcal{N}(\mu_t, \sigma_t^2)$ as the distribution of θ. This step is done numerically. Even though the cos or arccos function of a Gaussian variable is not truly Gaussian, we approximate it as such and find that this has negligible impact on the performance of the algorithm.

Our method for tuning the circuit angles $\vec{x}$ is as follows. Ideally, we want to choose them carefully so that the mean squared error (MSE) of the estimator μ_t of θ decreases as fast as possible as t grows. In practice, however, it is hard to compute this quantity directly, and we must resort to a proxy of its value. The MSE of an estimator is a sum of the variance of the estimator and the squared bias of the estimator. We find that the squared bias of μ_t is often smaller than its variance, i.e. $|\mathbb{E}[\mu_t] - \theta^*|^2 < \mathrm{Var}(\mu_t)$, where θ^* is the true value of θ. We also find that the variance σ_t^2 of θ is often close to the variance of μ_t, i.e. $\sigma_t^2 \approx \mathrm{Var}(\mu_t)$. Combining these facts, we know that $\mathrm{MSE}(\mu_t) \leq 2\sigma_t^2$ with high probability. So we will find the parameters $\vec{x}$ that minimize the variance σ_t^2 of θ instead.

Specifically, suppose θ has prior distribution $\mathcal{N}(\mu, \sigma^2)$. Upon receiving the measurement outcome $d \in \{0, 1\}$, the expected posterior variance [43] of θ is

$$\mathbb{E}_{\mathrm{d}}[\mathrm{Var}(\theta|d; f, \vec{x})] = \sigma^2 \left(1 - \sigma^2 \frac{f^2(\partial_\mu b(\mu, \sigma; \vec{x}))^2}{1 - f^2(b(\mu, \sigma; \vec{x}))^2}\right), \tag{32}$$

where

$$b(\mu, \sigma; \vec{x}) = \int_{-\infty}^{\infty} \mathrm{d}\theta \; p(\theta; \mu, \sigma) \Delta(\theta; \vec{x}) \tag{33}$$

in which $\Delta(\theta; \vec{x})$ is the bias of the ideal likelihood function as defined in Eq. (22), and f is the fidelity of the process for generating the likelihood function. We introduce an important quantity for engineering likelihood functions that we refer to as the *variance reduction factor*,

$$\mathcal{V}(\mu, \sigma; f, \vec{x}) := \frac{f^2(\partial_\mu b(\mu, \sigma; \vec{x}))^2}{1 - f^2(b(\mu, \sigma; \vec{x}))^2}. \tag{34}$$

Then we have

$$\mathbb{E}_{\mathrm{d}}[\mathrm{Var}(\theta|d; f, \vec{x})] = \sigma^2 \left[1 - \sigma^2 \mathcal{V}(\mu, \sigma; f, \vec{x})\right]. \tag{35}$$

The larger $\mathcal{V}$ is, the faster the variance of θ decreases on average. Furthermore, to quantify the growth rate (per time step) of the inverse variance of θ, we introduce the following quantity

$$R(\mu, \sigma; f, \vec{x}) := \frac{1}{T(L)} \left(\frac{1}{\mathbb{E}_{\mathrm{d}}[\mathrm{Var}(\theta|d; f, \vec{x})]} - \frac{1}{\sigma^2}\right) \tag{36}$$

$$= \frac{1}{T(L)} \frac{\mathcal{V}(\mu, \sigma; f, \vec{x})}{1 - \sigma^2 \mathcal{V}(\mu, \sigma; f, \vec{x})}, \tag{37}$$

where $T(L)$ is the time cost of an inference round. Note that R is a monotonic function of $\mathcal{V}$ for $\mathcal{V} \in (0, 1)$. Therefore, when the number L of circuit layers is fixed, we can maximize R (with respect to $\vec{x}$) by maximizing $\mathcal{V}$. In addition, when σ is small, R is approximately proportional to $\mathcal{V}$, i.e. $R \approx \mathcal{V}/T(L)$. For the remainder of this work, we will assume that the ansatz circuit contributes most significantly to the duration of the overall circuit. We take $T(L)$ to be proportional to the number of times the ansatz is invoked in the circuit, setting $T(L) = 2L + 1$, where time is in units of ansatz duration.

So now we need to find the parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in \mathbb{R}^{2L}$ that maximize the variance reduction factor $\mathcal{V}(\mu, \sigma; f, \vec{x})$ for given $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^+$ and $f \in [0, 1]$. This optimization problem turns out to be difficult to solve in general. Fortunately, in practice, we may assume that the prior variance σ^2 of θ is small (e.g. at

most 0.01), and in this case, $\mathcal{V}(\mu, \sigma; f, \vec{x})$ can be approximated by the Fisher information of the likelihood function $\mathbb{P}(d|\theta; f, \vec{x})$ at $\theta = \mu$, as shown in Appendix C, i.e.

$$\mathcal{V}(\mu, \sigma; f, \vec{x}) \approx \mathcal{I}(\mu; f, \vec{x}), \quad \text{when } \sigma \text{ is small,} \tag{38}$$

where

$$\begin{aligned} \mathcal{I}(\theta; f, \vec{x}) &= \mathbb{E}_d \left[\left(\partial_\theta \log \mathbb{P}(d|\theta; f, \vec{x}) \right)^2 \right] & (39) \\ &= \frac{f^2 (\Delta'(\theta; \vec{x}))^2}{1 - f^2 (\Delta(\theta; \vec{x}))^2} & (40) \end{aligned}$$

is the Fisher information of the two-outcome likelihood function $\mathbb{P}(d|\theta; f, \vec{x})$ as defined in Eq. (25). Therefore, rather than directly optimizing the variance reduction factor $\mathcal{V}(\mu, \sigma; f, \vec{x})$, we optimize the Fisher information $\mathcal{I}(\mu; f, \vec{x})$, which can be done efficiently by the algorithms in Section 4.1.1. Furthermore, when the fidelity f of the process for generating the ELF is low, we have $\mathcal{I}(\theta; f, \vec{x}) \approx f^2 (\Delta'(\theta; \vec{x}))^2$. It follows that

$$\mathcal{V}(\mu, \sigma; f, \vec{x}) \approx f^2 (\Delta'(\mu; \vec{x}))^2, \quad \text{when both } \sigma \text{ and } f \text{ are small.} \tag{41}$$

So in this case, we can simply optimize $|\Delta'(\mu; \vec{x})|$, which is proportional to the slope of the likelihood function $\mathbb{P}(d|\theta; f, \vec{x})$ at $\theta = \mu$, and this task can be accomplished efficiently by the algorithms in Section 4.1.2.

Finally, we make a prediction on how fast the MSE of the estimator $\hat{\mu}_t$ of Π decays as t grows, under the assumption that the number L of circuit layers is fixed during the inference process. Note that $\mathrm{MSE}(\hat{\mu}_t) = \Theta(\frac{1}{t})$ as $t \to \infty$ in this case. As $t \to \infty$, we have $\mu_t \to \theta^*$, $\sigma_t \to 0$, $\hat{\mu}_t \to \Pi^*$, and $\hat{\sigma}_t \to 0$ with high probability, where θ^* and Π^* are the true values of θ and Π, respectively. When this event happens, for large t, we get

$$\frac{1}{\sigma_{t+1}^2} - \frac{1}{\sigma_t^2} \approx \mathcal{I}(\mu_t; f, \vec{x}_t). \tag{42}$$

Consequently, by Eq. (31), we know that for large t,

$$\frac{1}{\hat{\sigma}_{t+1}^2} - \frac{1}{\hat{\sigma}_t^2} \approx \frac{\mathcal{I}(\mu_t; f, \vec{x}_t)}{\sin^2(\mu_t)}, \tag{43}$$

where $\mu_t \approx \arccos(\hat{\mu}_t)$. Since the bias of $\hat{\mu}_t$ is often much smaller than its standard deviation, and the latter can be approximated by $\hat{\sigma}_t$, we predict that for large t,

$$\mathrm{MSE}(\hat{\mu}_t) \approx \frac{1 - \hat{\mu}_t^2}{t \mathcal{I}(\arccos(\hat{\mu}_t); f, \vec{x}_t)}. \tag{44}$$

This means that the asymptotic growth rate (per time step) of the inverse MSE of $\hat{\mu}_t$ should be roughly

$$\hat{R}_0(\Pi^*; f, \vec{x}) := \frac{\mathcal{I}(\arccos(\Pi^*); f, \vec{x})}{(2L+1)(1 - (\Pi^*)^2)}, \tag{45}$$

where $\vec{x}$ is optimized with respect to $\mu^* = \arccos(\Pi^*)$. We will compare this rate with the empirical growth rate of the inverse MSE of $\hat{\mu}_t$ in Section 5.

4 Efficient heuristic algorithms for circuit parameter tuning and Bayesian inference

In this section, we present heuristic algorithms for tuning the parameters $\vec{x}$ of the circuit in Fig. 3.1 and describe how to efficiently carry out Bayesian inference with the resultant likelihood functions.

4.1 Efficient maximization of proxies of the variance reduction factor

Our algorithms for tuning the circuit angles $\vec{x}$ are based on maximizing two proxies of the variance reduction factor $\mathcal{V}$ – the Fisher information and slope of the likelihood function $\mathbb{P}(d|\theta; f, \vec{x})$. All of these algorithms require efficient procedures for evaluating the CSBD coefficient functions of $\Delta(\theta; \vec{x})$ and $\Delta'(\theta; \vec{x})$ with respect to x_j for $j = 1, 2, \ldots, 2L$. Recall that we have shown in Section 3.1 that $\Delta(\theta; \vec{x})$ is trigono-multiquadratic in $\vec{x}$. Namely, for any $j \in \{1, 2, \ldots, 2L\}$, there exist functions $C_j(\theta; \vec{x}_{\neg j})$, $S_j(\theta; \vec{x}_{\neg j})$ and $B_j(\theta; \vec{x}_{\neg j})$ of $\vec{x}_{\neg j} := (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_{2L})$ such that

$$\Delta(\theta; \vec{x}) = C_j(\theta; \vec{x}_{\neg j}) \cos(2x_j) + S_j(\theta; \vec{x}_{\neg j}) \sin(2x_j) + B_j(\theta; \vec{x}_{\neg j}). \tag{46}$$

It follows that

$$\Delta'(\theta; \vec{x}) = C'_j(\theta; \vec{x}_{\neg j}) \cos(2x_j) + S'_j(\theta; \vec{x}_{\neg j}) \sin(2x_j) + B'_j(\theta; \vec{x}_{\neg j}) \tag{47}$$

is also trigono-multiquadratic in $\vec{x}$, where $C'_j(\theta; \vec{x}_{\neg j}) = \partial_\theta C_j(\theta; \vec{x}_{\neg j})$, $S'_j(\theta; \vec{x}_{\neg j}) = \partial_\theta S_j(\theta; \vec{x}_{\neg j})$, $B'_j(\theta; \vec{x}_{\neg j}) = \partial_\theta B_j(\theta; \vec{x}_{\neg j})$ are the derivatives of $C_j(\theta; \vec{x}_{\neg j})$, $S_j(\theta; \vec{x}_{\neg j})$, $B_j(\theta; \vec{x}_{\neg j})$ with respect to θ, respectively. It turns out that given θ and $\vec{x}_{\neg j}$, $C_j(\theta; \vec{x}_{\neg j})$, each of $S_j(\theta; \vec{x}_{\neg j})$, $B_j(\theta; \vec{x}_{\neg j})$, $C'_j(\theta; \vec{x}_{\neg j})$, $S'_j(\theta; \vec{x}_{\neg j})$ and $B'_j(\theta; \vec{x}_{\neg j})$ can be computed in $O(L)$ time.

Lemma 1. *Given θ and $\vec{x}_{\neg j}$, each of $C_j(\theta; \vec{x}_{\neg j})$, $S_j(\theta; \vec{x}_{\neg j})$, $B_j(\theta; \vec{x}_{\neg j})$, $C'_j(\theta; \vec{x}_{\neg j})$, $S'_j(\theta; \vec{x}_{\neg j})$ and $B'_j(\theta; \vec{x}_{\neg j})$ can be computed in $O(L)$ time.*

Proof. See Appendix A. □

4.1.1 Maximizing the Fisher information of the likelihood function

We propose two algorithms for maximizing the Fisher information of the likelihood function $\mathbb{P}(d|\theta; f, \vec{x})$ at a given point $\theta = \mu$ (i.e. the prior mean of θ). Namely, our goal is to find $\vec{x} \in \mathbb{R}^{2L}$ that maximize

$$\mathcal{I}(\mu; f, \vec{x}) = \frac{f^2(\Delta'(\mu; \vec{x}))^2}{1 - f^2\Delta(\mu; \vec{x})^2}. \tag{48}$$

The first algorithm is based on *gradient ascent*. Namely, it starts with a random initial point, and keeps taking steps proportional to the gradient of $\mathcal{I}$ at the current point, until a convergence criterion is satisfied. Specifically, let $\vec{x}^{(t)}$ be the parameter vector at iteration t. We update it as follows:

$$\vec{x}^{(t+1)} = \vec{x}^{(t)} + \delta(t) \nabla \mathcal{I}(\mu; f, \vec{x})|_{\vec{x}=\vec{x}^{(t)}}. \tag{49}$$

where $\delta : \mathbb{Z}^{\geq 0} \to \mathbb{R}^+$ is the step size schedule[6]. This requires the calculation of the partial derivative of $\mathcal{I}(\mu; f, \vec{x})$ with respect to each x_j, which can be done as follows. We first use the procedures in Lemma 1 to compute $C_j := C_j(\mu; \vec{x}_{\neg j})$, $S_j := S_j(\mu; \vec{x}_{\neg j})$, $B_j := B_j(\mu; \vec{x}_{\neg j})$, $C'_j := C'_j(\mu; \vec{x}_{\neg j})$, $S'_j := S'_j(\mu; \vec{x}_{\neg j})$ and $B'_j := B'_j(\mu; \vec{x}_{\neg j})$ for each j. Then we get

$$\Delta := \Delta(\mu; \vec{x}) = C_j \cos(2x_j) + S_j \sin(2x_j) + B_j, \tag{50}$$

$$\Delta' := \Delta'(\mu; \vec{x}) = C'_j \cos(2x_j) + S'_j \sin(2x_j) + B'_j, \tag{51}$$

$$\chi_j := \frac{\partial \Delta(\mu; \vec{x})}{\partial x_j} = 2\left(-C_j \sin(2x_j) + S_j \cos(2x_j)\right), \tag{52}$$

$$\chi'_j := \frac{\partial \Delta'(\mu; \vec{x})}{\partial x_j} = 2\left(-C'_j \sin(2x_j) + S'_j \cos(2x_j)\right); \tag{53}$$

Knowing these quantities, we can compute the partial derivative of $\mathcal{I}(\mu; f, \vec{x})$ with respect to x_j as follows:

$$\gamma_j := \frac{\partial \mathcal{I}(\mu; f, \vec{x})}{\partial x_j} = \frac{2f^2\left[(1 - f^2\Delta^2)\Delta'\chi'_j + f^2\Delta\chi_j(\Delta')^2\right]}{\left(1 - f^2\Delta^2\right)^2}. \tag{54}$$

Repeat this procedure for $j = 1, 2, \ldots, 2L$. Then we obtain $\nabla \mathcal{I}(\mu; f, \vec{x}) = (\gamma_1, \gamma_2, \ldots, \gamma_{2L})$. Each iteration of the algorithm takes $O(L^2)$ time. The number of iterations in the algorithm depends on the initial point, the termination criterion and the step size schedule δ. See Algorithm 1 for more details.

[6] In the simplest case, $\delta(t) = \delta$ is constant. But in order to achieve better performance, we might want $\delta(t) \to 0$ as $t \to \infty$.

The second algorithm is based on *coordinate ascent*. Unlike gradient ascent, this algorithm does not require step sizes, and allows each variable to change dramatically in a single step. As a consequence, it may converge faster than the previous algorithm. Specifically, this algorithm starts with a random initial point, and successively maximizes the objective function $\mathcal{I}(\mu; f, \vec{x})$ along coordinate directions, until a convergence criterion is satisfied. At the j-th step of each round, it solves the following single-variable optimization problem for a coordinate x_j:

$$\arg\max_{z} \frac{f^2 \left(C_j' \cos(2z) + S_j' \sin(2z) + B_j'\right)^2}{1 - f^2 \left(C_j \cos(2z) + S_j \sin(2z) + B_j\right)^2}, \tag{55}$$

where $C_j = C_j(\mu; \vec{x}_{\neg j})$, $S_j = S_j(\mu; \vec{x}_{\neg j})$, $B_j = B_j(\mu; \vec{x}_{\neg j})$, $C_j' = C_j'(\mu; \vec{x}_{\neg j})$, $S_j' = S_j'(\mu; \vec{x}_{\neg j})$, $B_j' = B_j'(\mu; \vec{x}_{\neg j})$ can be computed in $O(L)$ time by the procedures in Lemma 1. This single-variable optimization problem can be tackled by standard gradient-based methods, and we set x_j to be its solution. Repeat this procedure for $j = 1, 2, \ldots, 2L$. This algorithm produces a sequence $\vec{x}^{(0)}$, $\vec{x}^{(1)}$, $\vec{x}^{(2)}$, $\ldots$, such that $\mathcal{I}(\mu; f, \vec{x}^0) \leq \mathcal{I}(\mu; f, \vec{x}^1) \leq \mathcal{I}(\mu; f, \vec{x}^2) \leq \ldots$. Namely, the value of $\mathcal{I}(\mu; f, \vec{x}^t)$ increases monotonically as t grows. Each round of the algorithm takes $O(L^2)$ time. The number of rounds in the algorithm depends on the initial point and the termination criterion. See Algorithm 2 for more details.

4.1.2 Maximizing the slope of the likelihood function

We also propose two algorithms for maximizing the slope of the likelihood function $\mathbb{P}(d|\theta; f, \vec{x})$ at a given point $\theta = \mu$ (i.e. the prior mean of θ). Namely, our goal is to find $\vec{x} \in \mathbb{R}^{2L}$ that maximize $|\mathbb{P}'(\mu; f, \vec{x})| = f|\Delta'(\mu; \vec{x})|/2$.

Similar to Algorithms 1 and 2 for Fisher information maximization, our algorithms for slope maximization are also based on gradient ascent and coordinate ascent, respectively. They both need to call the procedures in Lemma 1 to evaluate $C'(\mu; \vec{x}_{\neg j})$, $S'(\mu; \vec{x}_{\neg j})$ and $B'(\mu; \vec{x}_{\neg j})$ for given μ and $\vec{x}_{\neg j}$. However, the gradient-ascent-based algorithm uses the above quantities to compute the partial derivative of $(\Delta'(\mu; \vec{x}))^2$ with respect to x_j, while the coordinate-ascent-based algorithm uses them to directly update the value of x_j. These algorithms are formally described in Algorithms 3 and 4, respectively.

4.2 Approximate Bayesian inference with engineered likelihood functions

With the algorithms for tuning the circuit parameters $\vec{x}$ in place, we now describe how to efficiently carry out Bayesian inference with the resultant likelihood functions. In principle, we can compute the posterior mean and variance of θ directly after receiving a measurement outcome d. But this approach is time-consuming, as it involves numerical integration. By taking advantage of certain property of the engineered likelihood functions, we can greatly accelerate this process.

Suppose θ has prior distribution $\mathcal{N}(\mu, \sigma^2)$, where $\sigma \ll 1/L$, and the fidelity of the process for generating the ELF is f. We find that the parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L})$ that maximize $\mathcal{I}(\mu; f, \vec{x})$ (or $|\Delta'(\mu; \vec{x})|$) satisfy the following property: When θ is close to μ, i.e. $\theta \in [\mu - O(\sigma), \mu + O(\sigma)]$, we have

$$\mathbb{P}(d|\theta; f, \vec{x}) \approx \frac{1 + (-1)^d f \sin(r\theta + b)}{2} \tag{63}$$

for some $r, b \in \mathbb{R}$. Namely, $\Delta(\theta; \vec{x})$ can be approximated by a sinusoidal function in this region of θ. Fig. 4.1 illustrates one such example.

We can find the best-fitting r and b by solving the following least squares problem:

$$(r^*, b^*) = \arg\min_{r,b} \sum_{\theta \in \Theta} \left|\arcsin(\Delta(\theta; \vec{x})) - r\theta - b\right|^2, \tag{64}$$

where $\Theta = \{\theta_1, \theta_2, \ldots, \theta_k\} \subseteq [\mu - O(\sigma), \mu + O(\sigma)]$. This least-squares problem has the following analytical solution:

$$\begin{pmatrix} r^* \\ b^* \end{pmatrix} = A^+ z = (A^T A)^{-1} A^T z, \tag{65}$$

Algorithm 1: Gradient ascent for Fisher information maximization in the ancilla-free case

Input: The prior mean μ of θ, the number L of circuit layers, the fidelity f of the process for generating the ELF, the step size schedule $\delta : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{+}$, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in \mathbb{R}^{2L}$ that are a local maximum point of the function $\mathcal{I}(\mu; f, \vec{x})$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 0$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t)}, \ldots, x_{2L}^{(t)})$;
    Compute $C_j^{(t)} := C_j(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j^{(t)} := S_j(\mu; \vec{x}_{\neg j}^{(t)})$, $B_j^{(t)} := B_j(\mu; \vec{x}_{\neg j}^{(t)})$, $C_j^{\prime(t)} := C_j^{\prime}(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j^{\prime(t)} := S_j^{\prime}(\mu; \vec{x}_{\neg j}^{(t)})$, $B_j^{\prime(t)} := B_j^{\prime}(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 1;
    Compute $\Delta(\mu; \vec{x})$, $\Delta'(\mu; \vec{x})$ and their partial derivatives with respect to x_j at $\vec{x} = \vec{x}^{(t)}$ as follows:

$$\Delta^{(t)} := \Delta(\mu; \vec{x}^{(t)}) = C_j^{(t)} \cos(2x_j) + S_j^{(t)} \sin(2x_j) + B_j^{(t)}, \tag{56}$$

$$\Delta^{\prime(t)} := \Delta'(\mu; \vec{x}^{(t)}) = C_j^{\prime(t)} \cos(2x_j) + S_j^{\prime(t)} \sin(2x_j) + B_j^{\prime(t)}, \tag{57}$$

$$\chi_j^{(t)} := \frac{\partial \Delta(\mu; \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = 2\left(-C_j^{(t)} \sin(2x_j) + S_j^{(t)} \cos(2x_j)\right), \tag{58}$$

$$\chi_j^{\prime(t)} := \frac{\partial \Delta'(\mu; \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = 2\left(-C_j^{\prime(t)} \sin(2x_j) + S_j^{\prime(t)} \cos(2x_j)\right); \tag{59}$$

    Compute the partial derivative of $\mathcal{I}(\mu; f, \vec{x})$ with respect to x_j at $\vec{x} = \vec{x}^{(t)}$ as follows:

$$\gamma_j^{(t)} := \frac{\partial \mathcal{I}(\mu; f, \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = \frac{2f^2\left[(1 - f^2(\Delta^{(t)})^2)\Delta^{\prime(t)}\chi_j^{\prime(t)} + f^2\Delta^{(t)}\chi_j^{(t)}(\Delta^{\prime(t)})^2\right]}{\left[1 - f^2(\Delta^{(t)})^2\right]^2} \tag{60}$$

  end
  Set $\vec{x}^{(t+1)} = \vec{x}^{(t)} + \delta(t)\nabla\mathcal{I}(\mu; f, \vec{x})|_{\vec{x}=\vec{x}^{(t)}}$, where $\nabla\mathcal{I}(\mu; f, \vec{x})|_{\vec{x}=\vec{x}^{(t)}} = (\gamma_1^{(t)}, \gamma_2^{(t)}, \ldots, \gamma_{2L}^{(t)})$;
  if $|\mathcal{I}(\mu; f, \vec{x}^{(t+1)}) - \mathcal{I}(\mu; f, \vec{x}^{(t)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t+1)} = (x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{2L}^{(t+1)})$ as the optimal parameters.

Algorithm 2: Coordinate ascent for Fisher information maximization in the ancilla-free case

Input: The prior mean μ of θ, the number L of circuit layers, the fidelity f of the process for generating the ELF, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in (-\pi, \pi]^{2L}$ that are a local maximum point of the function $\mathcal{I}(\mu; f, \vec{x})$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 1$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t-1)}, \ldots, x_{2L}^{(t-1)})$;
    Compute $C_j^{(t)} := C_j(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j^{(t)} := S_j(\mu; \vec{x}_{\neg j}^{(t)})$, $B_j^{(t)} := B_j(\mu; \vec{x}_{\neg j}^{(t)})$, $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$, $B_j'^{(t)} := B_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 1;
    Solve the single-variable optimization problem

$$\arg\max_z \frac{f^2 \left(C_j'^{(t)} \cos(2z) + S_j'^{(t)} \sin(2z) + B_j'^{(t)} \right)^2}{1 - f^2 \left(C_j^{(t)} \cos(2z) + S_j^{(t)} \sin(2z) + B_j^{(t)} \right)^2}$$

    by standard gradient-based methods and set $x_j^{(t)}$ to be its solution;
  end
  if $|\mathcal{I}(\mu; f, \vec{x}^{(t)}) - \mathcal{I}(\mu; f, \vec{x}^{(t-1)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t)} = (x_1^{(t)}, x_2^{(t)}, \ldots, x_{2L}^{(t)})$ as the optimal parameters.

Algorithm 3: Gradient ascent for slope maximization in the ancilla-free case

Input: The prior mean μ of θ, the number L of circuit layers, the step size schedule $\delta : \mathbb{Z}^{\geq 0} \to \mathbb{R}^+$, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in \mathbb{R}^{2L}$ that are a local maximum point of the function $|\Delta'(\mu; \vec{x})|$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 0$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t)}, \ldots, x_{2L}^{(t)})$;
    Compute $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$ and $B_j'^{(t)} := B_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 1;
    Compute $\Delta'(\mu; \vec{x})$ at $\vec{x} = \vec{x}^{(t)}$ as follows:

$$\Delta'^{(t)} := \Delta'(\mu; \vec{x}^{(t)}) = C_j'^{(t)} \cos\left(2x_j^{(t)}\right) + S_j'^{(t)} \sin\left(2x_j^{(t)}\right) + B_j'^{(t)}; \tag{61}$$

    Compute the partial derivative of $\Delta'(\mu; \vec{x})$ with respect to x_j as follows:

$$\gamma_j^{(t)} := \frac{\partial \Delta'(\mu; \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = 2\left(-C_j'^{(t)} \sin\left(2x_j^{(t)}\right) + S_j'^{(t)} \cos\left(2x_j^{(t)}\right)\right); \tag{62}$$

  end
  Set $\vec{x}^{(t+1)} = \vec{x}^{(t)} + \delta(t)\nabla^{(t)}$, where $\nabla^{(t)} := (2\Delta'^{(t)}\gamma_1^{(t)}, 2\Delta'^{(t)}\gamma_2^{(t)}, \ldots, 2\Delta'^{(t)}\gamma_{2L}^{(t)})$ is the gradient of $(\Delta'(\mu; \vec{x}))^2$ at $\vec{x} = \vec{x}^{(t)}$;
  if $|\Delta'(\mu; \vec{x}^{(t+1)}) - \Delta'(\mu; \vec{x}^{(t)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t+1)} = (x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{2L}^{(t+1)})$ as the optimal parameters.

Algorithm 4: Coordinate ascent for slope maximization in the ancilla-free case

Input: The prior mean μ of θ, the number L of circuit layers, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in (-\pi, \pi]^{2L}$ that are a local maximum point of the function $|\Delta'(\mu; \vec{x})|$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 1$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t-1)}, \ldots, x_{2L}^{(t-1)})$;
    Compute $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$, $B_j'^{(t)} := B_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 1;
    Set $x_j^{(t)} = \mathrm{Arg}\left(\mathrm{sgn}\left(B_j'^{(t)}\right)(C_j'^{(t)} + iS_j'^{(t)})\right)/2$, where $\mathrm{sgn}(z) = 1$ if $z \geq 0$ and -1 otherwise;
  end
  if $|\Delta'(\mu; \vec{x}^{(t)}) - \Delta'(\mu; \vec{x}^{(t-1)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t)} = (x_1^{(t)}, x_2^{(t)}, \ldots, x_{2L}^{(t)})$ as the optimal parameters.

where

$$A = \begin{pmatrix} \theta_1 & 1 \\ \theta_2 & 1 \\ \vdots & \vdots \\ \theta_k & 1 \end{pmatrix}, \quad z = \begin{pmatrix} \arcsin\left(\Delta(\theta_1; \vec{x})\right) \\ \arcsin\left(\Delta(\theta_2; \vec{x})\right) \\ \vdots \\ \arcsin\left(\Delta(\theta_k; \vec{x})\right) \end{pmatrix}. \tag{66}$$

Figure 4.1 demonstrates an example of the true and fitted likelihood functions.

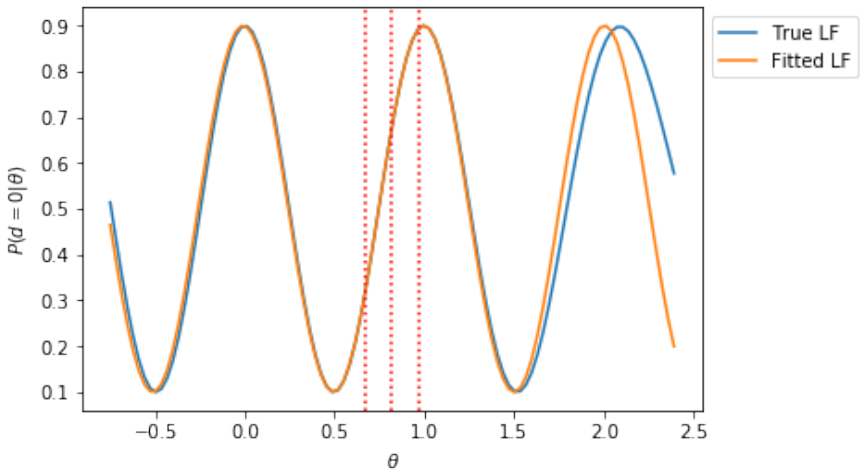


Figure 4.1: The true and fitted likelihood functions when $L = 3$, $f = 0.8$, and θ has prior distribution $\mathcal{N}(0.82, 0.0009)$. The true likelihood function is generated by Algorithm 2. During the sinusoidal fitting of this function, we set $\Theta = \{\mu-\sigma, \mu-0.8\sigma, \ldots, \mu+0.8\sigma, \mu+\sigma\}$ (i.e. Θ contains 11 uniformly distributed points in $[\mu-\sigma, \mu+\sigma]$) in Eq. (64). The fitted likelihood function is $\mathbb{P}(d|\theta) = (1+(-1)^d f \sin(r\theta+b))/2$, where $r = 6.24217$ and $b = -4.65086$. Note that the true and fitted likelihood functions are close for $\theta \in [0.67, 0.97]$.

Once we obtain the optimal r and b, we can approximate the posterior mean and variance of θ by the ones for

$$\mathbb{P}(d|\theta; f) = \frac{1+(-1)^d f \sin(r\theta+b)}{2}, \tag{67}$$

which have analytical formulas. Specifically, suppose θ has prior distribution $\mathcal{N}(\mu_k, \sigma_k^2)$ at round k. Let d_k be the measurement outcome and (r_k, b_k) be the best-fitting parameters at this round. Then we approximate the posterior mean and variance of θ by

$$\mu_{k+1} = \mu_k + \frac{(-1)^{d_k} f e^{-r_k^2\sigma_k^2/2} r_k \sigma_k^2 \cos(r_k\mu_k + b_k)}{1+(-1)^{d_k} f e^{-r_k^2\sigma_k^2/2} \sin(r_k\mu_k + b_k)}, \tag{68}$$

$$\sigma_{k+1}^2 = \sigma_k^2 \left(1 - \frac{f r_k^2 \sigma_k^2 e^{-r_k^2\sigma_k^2/2}[f e^{-r_k^2\sigma_k^2/2} + (-1)^{d_k} \sin(r_k\mu_k + b_k)]}{[1+(-1)^{d_k} f e^{-r_k^2\sigma_k^2/2} \sin(r_k\mu_k + b_k)]^2}\right). \tag{69}$$

After that, we proceed to the next round, setting $\mathcal{N}(\mu_{k+1}, \sigma_{k+1}^2)$ as the prior distribution of θ for that round.

Note that, as Fig. 4.1 illustrates, the difference between the true and fitted likelihood functions can be large when θ is far from μ, i.e. $|\theta - \mu| \gg \sigma$. But since the prior distribution $p(\theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\theta-\mu)^2}{2\sigma^2}}$ decays exponentially in $|\theta - \mu|$, such θ's have little contribution to the computation of posterior mean and variance of θ. So Eqs. (68) and (69) give highly accurate estimates of the posterior mean and variance of θ, and their errors have negligible impact on the performance of the whole algorithm.

5 Simulation results

In this section, we present the simulation results of Bayesian inference with engineered likelihood functions for amplitude estimation. These results demonstrate the advantages of engineered likelihood functions over unengineered ones, as well as the impacts of circuit depth and fidelity on their performance.

5.1 Experimental details

In our experiments, we assume that it takes much less time to implement $U(x) = \exp(-\mathrm{i}xP)$ and perform the projective measurement $\{\frac{I+P}{2}, \frac{I-P}{2}\}$ than to implement A. So when the number of circuit layer is L, the time cost of an inference round is roughly $2L+1$ (note that an L-layer circuit makes $2L+1$ uses of A and $A^\dagger$), where time is in units of A's duration. Moreover, we assume that there is no error in the preparation and measurements of quantum states, i.e. $\bar{p} = 1$, in the experiments.

Suppose we aim to estimate the expectation value $\Pi = \cos(\theta) = \langle A| P |A\rangle$. Let $\hat{\mu}_t$ be the estimator of Π at time t. Note that $\hat{\mu}_t$ itself is a random variable, since it depends on the history of random measurement outcomes up to time t. We measure the performance of a scheme by the root-mean-squared error (RMSE) of $\hat{\mu}_t$, that is given by

$$\mathrm{RMSE}_t := \sqrt{\mathrm{MSE}_t} = \sqrt{\mathbb{E}[(\hat{\mu}_t - \Pi)^2]}. \tag{70}$$

We will investigate how fast RMSE_t decays as t grows for various schemes, including the ancilla-based Chebyshev likelihood function (AB CLF), ancilla-based engineered likelihood function (AB ELF), ancilla-free Chebyshev likelihood function (AF CLF), and ancilla-free engineered likelihood function (AF ELF).

In general, the distribution of $\hat{\mu}_t$ is difficult to characterize, and there is no analytical formula for RMSE_t. To estimate this quantity, we execute the inference process M times, and collect M samples $\hat{\mu}_t^{(1)}$, $\hat{\mu}_t^{(2)}$, ..., $\hat{\mu}_t^{(M)}$ of $\hat{\mu}_t$, where $\hat{\mu}_t^{(i)}$ is the estimate of Π at time t in the i-th run, for $i = 1, 2, \ldots, M$. Then we use the quantity

$$\overline{\mathrm{RMSE}}_t := \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{\mu}_t^{(i)} - \Pi)^2}. \tag{71}$$

to approximate the true RMSE_t. In our experiments, we set $M = 300$ and find that this leads to a low enough empirical variance in the estimate to yield meaningful results.

We use coordinate-ascent-based Algorithms 2 and 6 to optimize the circuit parameters $\vec{x}$ in the ancilla-free and ancilla-based cases, respectively. We find that Algorithms 1 and 2 produce solutions of equal quality, and the same statement holds for Algorithms 5 and 6. So our experimental results would not change if we had used gradient-ascent-based Algorithms 1 and 5 to tune the circuit angles $\vec{x}$ instead.

For Bayesian update with ELFs, we use the methods in Section 4.2 and Appendix B.2 to compute the posterior mean and variance of θ in the ancilla-free and ancilla-based cases, respectively. In particular, during the sinusoidal fitting of ELFs, we set $\Theta = \{\mu - \sigma, \mu - 0.8\sigma, \ldots, \mu + 0.8\sigma, \mu + \sigma\}$ (i.e. Θ contains 11 uniformly distributed points in $[\mu - \sigma, \mu + \sigma]$) in Eqs. (64) and (206). We find that this is sufficient for obtaining high-quality sinusoidal fits of ELFs.

5.2 Comparing the performance of various schemes

To compare the performance of various quantum-generated likelihood functions, including AB CLF, AB ELF, AF CLF and AF ELF, we run Bayesian inference with each of them, fixing the number of circuit layers $L = 6$ and layer fidelity $p = 0.9$ (note that this layer fidelity corresponds to a 12-qubit experiment with two-qubit gate depth of 12 and two-qubit gate fidelity 99.92%, which is almost within reach for today's quantum devices). Figs. 5.1, 5.2, 5.3, 5.4 and 5.5 illustrate the performance of different schemes with respect to various true values of Π. These results suggest that:

- In both the ancilla-based and ancilla-free cases, ELF performs better than (or as well as) CLF. This means that by tuning the circuit angles $\vec{x}$, we do increase the information gain from the measurement outcome d at each round. So the estimation of Π becomes more efficient.

- AF ELF always performs better than AB ELF, whereas AF CLF may perform better or worse than AB CLF, depending on the true value of Π, but on average, AF CLF outperforms AB CLF. So overall the ancilla-free schemes are superior to the ancilla-based ones.
- While the RMSEs of AB-ELF-based and AF-ELF-based estimators of Π always converge to 0 as the inference process progresses, the same is not true for AB-CLF-based and AF-CLF-based estimators of Π. In fact, the performance of AB and AF CLFs depends heavily on the true value of Π. By contrast, AB and AF ELFs have more stable performance regardless of this value.

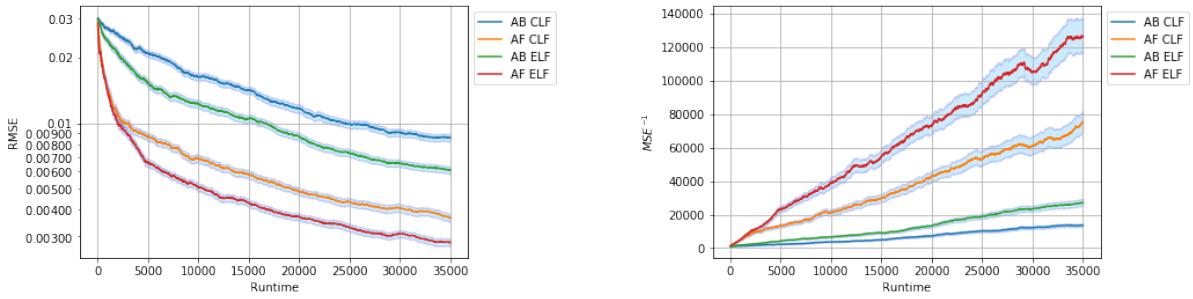


Figure 5.1: This figure compares the performance of AB CLF, AB ELF, AF CLF and AF ELF when the expectation value Π has true value -0.4 and prior distribution $\mathcal{N}(-0.43, 0.0009)$, the number of circuit layers is $L = 6$, and the layer fidelity is $p = 0.9$. Note that ELF outperforms CLF in both the ancilla-free and ancilla-based cases, and the ancilla-free schemes outperform the ancilla-based ones.

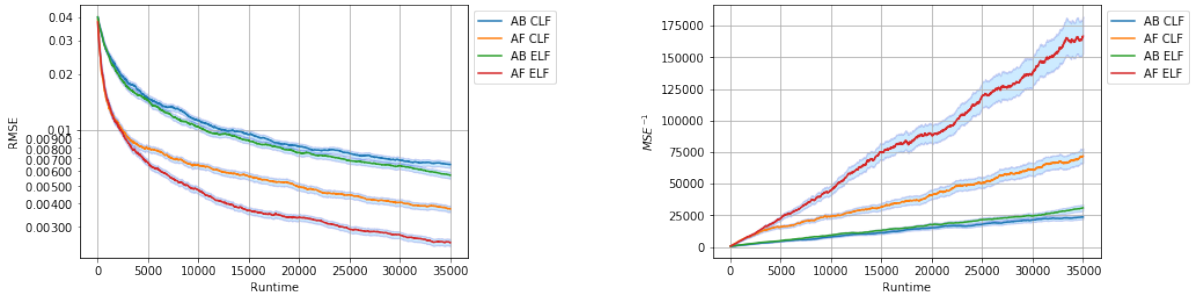


Figure 5.2: This figure compares the performance of AB CLF, AB ELF, AF CLF and AF ELF when the expectation value Π has true value 0.6 and prior distribution $\mathcal{N}(0.64, 0.0009)$, the number of circuit layers is $L = 6$, and the layer fidelity is $p = 0.9$. Note that AB ELF slightly outperforms AB CLF, while AF ELF outperforms AF CLF to a larger extent.

We may compare the above results with Fig. 5.6 that illustrates the $\hat{R}_0$ factors (as defined in Eq. (45)) of AB CLF, AB ELF, AF CLF and AF ELF in the same setting. One can observe from this figure that:

- The $\hat{R}_0$ factor of AB ELF is equal to or larger than that of AB CLF, and the same is true for AF ELF versus AB CLF. This explains why ELF outperforms CLF in both the ancilla-based and ancilla-free cases.
- The $\hat{R}_0$ factor of AF ELF is larger than that of AB ELF. Meanwhile, The $\hat{R}_0$ factor of AF CLF can be larger or smaller than that of AB CLF, depending on the value of Π, but on average AF CLF has larger $\hat{R}_0$ factor than AB CLF. This explains the superiority of the ancilla-free schemes over the ancilla-based ones.
- The $\hat{R}_0$ factors of AB and AF ELFs are bounded away from 0 for all $\Pi \in [-1, 1]$ [7]. This explains why their performance is not very sensitive to the true value of Π. On the other hand, the $\hat{R}_0$ factor

[7] Though not shown in Fig. 5.6, the $\hat{R}_0$ factors of AB and AF ELFs actually diverge to $+\infty$ as $\Pi \to \pm 1$, and this is true for any $L \in \mathbb{Z}^+$.

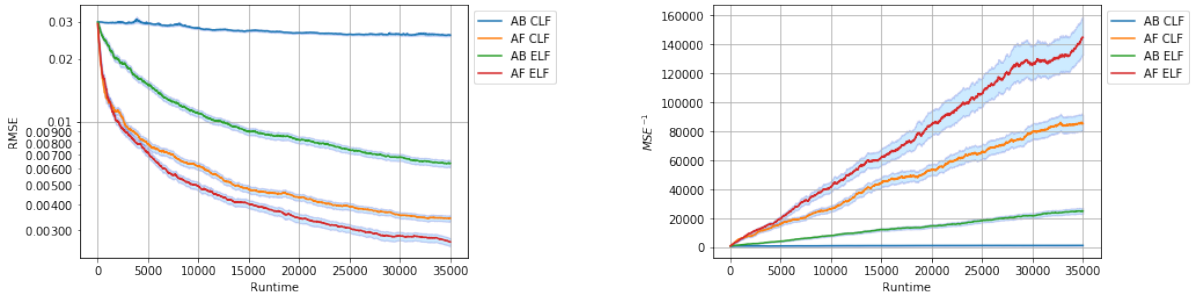


Figure 5.3: This figure compares the performance of AB CLF, AB ELF, AF CLF and AF ELF when the expectation value Π has true value 0.52 and prior distribution $\mathcal{N}(0.49, 0.0009)$, the number of circuit layers is $L = 6$, and the layer fidelity is $p = 0.9$. Note that the RMSE of the estimator of Π converges to 0 for all schemes except AB CLF. AF ELF achieves the best performance.

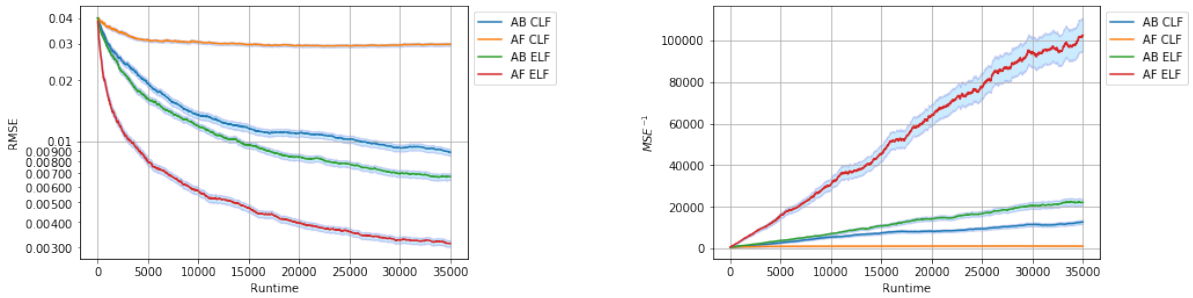


Figure 5.4: This figure compares the performance of AB CLF, AB ELF, AF CLF and AF ELF when the expectation value Π has true value -0.1 and prior distribution $\mathcal{N}(-0.14, 0.0009)$, the number of circuit layers is $L = 6$, and the layer fidelity is $p = 0.9$. Note that the RMSE of the estimator of Π converges to 0 for all schemes except AF CLF. AF ELF achieves the best performance.

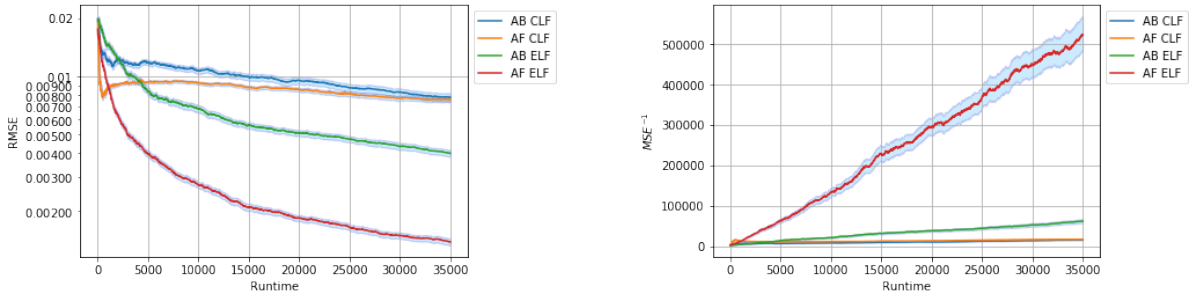


Figure 5.5: This figure compares the performance of AB CLF, AB ELF, AF CLF and AF ELF when the expectation value Π has true value 0.9 and prior distribution $\mathcal{N}(0.92, 0.0009)$, the number of circuit layers is $L = 6$, and the layer fidelity is $p = 0.9$. Note that the RMSE of the estimator of Π fails to converge to 0 for AF CLF and decreases slowly for AB CLF. Both AB and AF CLFs are outperformed by AB and AF ELFs, with AF ELF achieving the best performance.

of AB and AF CLFs changes dramatically for different Π's. In fact, one can verify that the $\hat{R}_0$ factor of AB CLF is 0 when $\Pi = \cos(j\pi/L)$ for $j = 0, 1, \ldots, L$, and the $\hat{R}_0$ factor of AF CLF is 0 when $\Pi = \cos(j\pi/(2L+1))$ for $j = 0, 1, \ldots, 2L+1$. This means that if the true value of Π is close to one of these "dead spots", then its estimator will struggle to improve and hence the performance of AB/AF CLF will suffer (see Figs. 5.3, 5.4 and 5.5 for examples.). AB and AF ELFs, on the other hand, do not have this weakness.

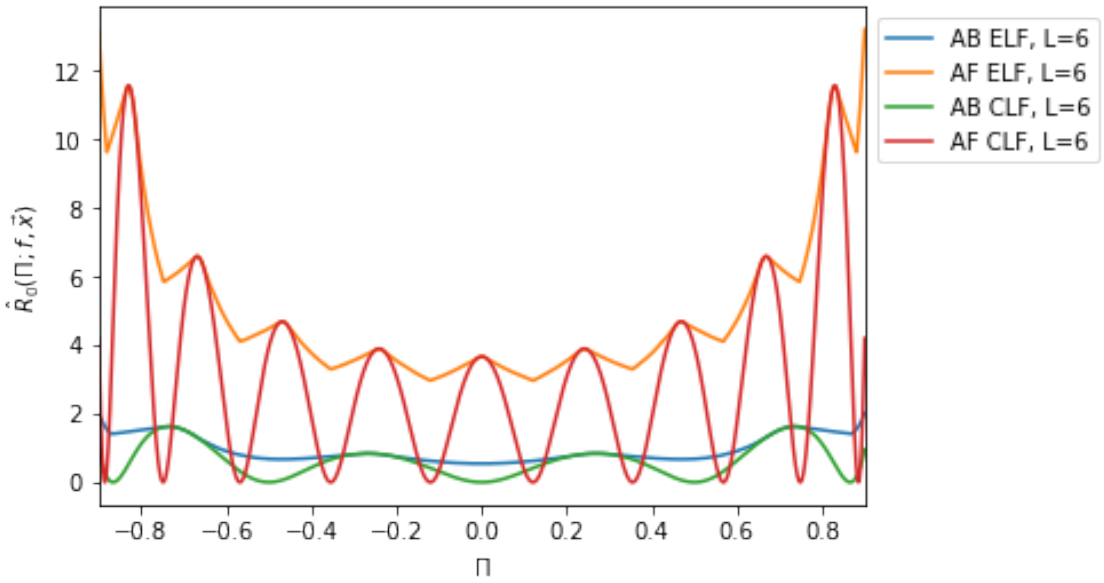


Figure 5.6: This figure shows the $\hat{R}_0$ factors of AB CLF, AB ELF, AF CLF and AF ELF for $\Pi \in [-0.9, 0.9]$, when the number of circuit layers is $L = 6$ and the layer fidelity is $p = 0.9$. Note that the $\hat{R}_0$ factors of AB and AF CLFs change dramatically for different Π's. In fact, they can be close to 0 for certain Π's. By contrast, the $\hat{R}_0$ factors of AB and AF ELFs are bounded away from 0 for all Π's.

5.3 Understanding the performance of Bayesian inference with ELFs

Having established the improved performance of ELFs over CLFs, we now analyze the performance of Bayesian inference with ELFs in more detail. Note that Figs. 5.1, 5.2, 5.3, 5.4 and 5.5 suggest that the inverse MSEs of both AB-ELF-based and AF-ELF-based estimators of Π grow linearly in runtime when the circuit depth is fixed. By fitting a linear model to the data, we obtain the empirical growth rates of these quantities, which are shown in Tables 2 and 3, respectively. We also compare these rates with the $\hat{R}_0$ factors of AB and AF ELFs in the same setting. It turns out that the $\hat{R}_0$ factor is a rough estimate of the true growth rate of the inverse MSE of an ELF-based estimator of Π, but it can be unreliable sometimes. We leave it as an open question to characterize more precisely the decay of the RMSEs of ELF-based estimators of Π during the inference process.

5.3.1 Analyzing the impact of layer fidelity on the performance of estimation

To investigate the influence of layer fidelity on the performance of estimation, we run Bayesian inference with AB and AF ELFs for fixed circuit depth but varied layer fidelity. Specifically, we set the number L of circuit layers to be 6, and vary the layer fidelity p from $0.75, 0.8, 0.85, 0.9$ to 0.95. Figs. 5.7 and 5.8 illustrate the simulation results in the ancilla-based and ancilla-free cases, respectively. As expected, higher layer fidelity leads to better performance of the algorithm. Namely, the less noisy the circuit is, the faster the RMSE of the estimator of Π decays. This is consistent with the fact that the $\hat{R}_0$ factors of AB and AF ELFs are monotonically increasing functions of $f = p^L$, as demonstrated by Fig. 5.9.

True value of Π	Prior distribution of Π	Predicated growth rate of the inverse MSE of estimator of Π	Empirical growth rate of the inverse MSE of estimator of Π
-0.4	$\mathcal{N}(-0.43, 0.0009)$	0.70654	0.75003
0.6	$\mathcal{N}(0.64, 0.0009)$	0.90765	0.85579
0.52	$\mathcal{N}(0.49, 0.0009)$	0.69926	0.74857
-0.1	$\mathcal{N}(-0.14, 0.0009)$	0.59899	0.68187
0.9	$\mathcal{N}(0.92, 0.0009)$	2.02599	1.83629

Table 2: The predicted and empirical growth rates of the inverse MSEs of AB-ELF-based estimators of Π in the five experiments in Section 5.2. In all of these experiments, the number of circuit layers is $L = 6$ and the layer fidelity is $p = 0.9$.

True value of Π	Prior distribution of Π	Predicated growth rate of the inverse MSE of estimator of Π	Empirical growth rate of the inverse MSE of estimator of Π
-0.4	$\mathcal{N}(-0.43, 0.0009)$	3.76842	3.69156
0.6	$\mathcal{N}(0.64, 0.0009)$	4.73018	4.68115
0.52	$\mathcal{N}(0.49, 0.0009)$	4.35941	4.19499
-0.1	$\mathcal{N}(-0.14, 0.0009)$	3.06583	3.07918
0.9	$\mathcal{N}(0.92, 0.0009)$	13.2193	14.74088

Table 3: The predicted and empirical growth rates of the inverse MSEs of AF-ELF-based estimators of Π in the five experiments in Section 5.2. In all of these experiments, the number of circuit layers is $L = 6$ and the layer fidelity is $p = 0.9$.

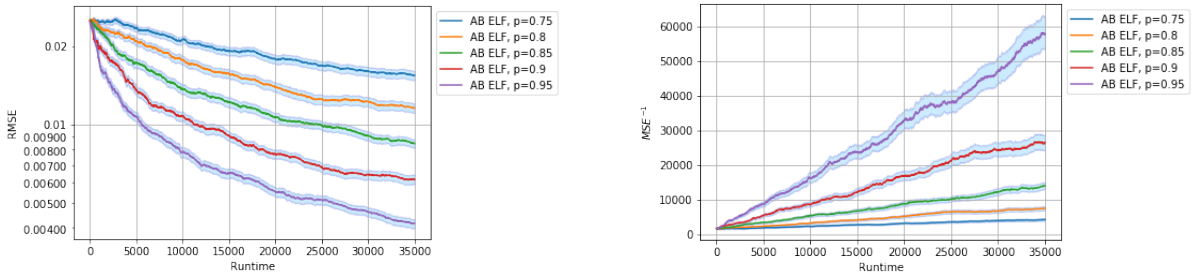


Figure 5.7: This figure demonstrates the impact of layer fidelity on the performance of AB ELF. Here Π has true value 0.18 and prior distribution $\mathcal{N}(0.205, 0.0009)$, the number L of circuit layers is 6, and the layer fidelity p is varied from $0.75, 0.8, 0.85, 0.9$ to 0.95. Note that higher layer fidelity leads to better performance of estimation.

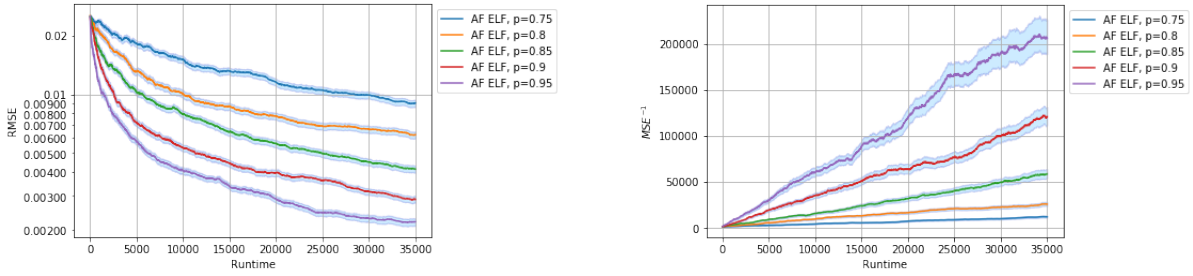


Figure 5.8: This figure demonstrates the impact of layer fidelity on the performance of AF ELF. Here Π has true value 0.18 and prior distribution $\mathcal{N}(0.205, 0.0009)$, the number L of circuit layers is 6, and the layer fidelity p is varied from $0.75, 0.8, 0.85, 0.9$ to 0.95. Note that higher layer fidelity leads to better performance of estimation.

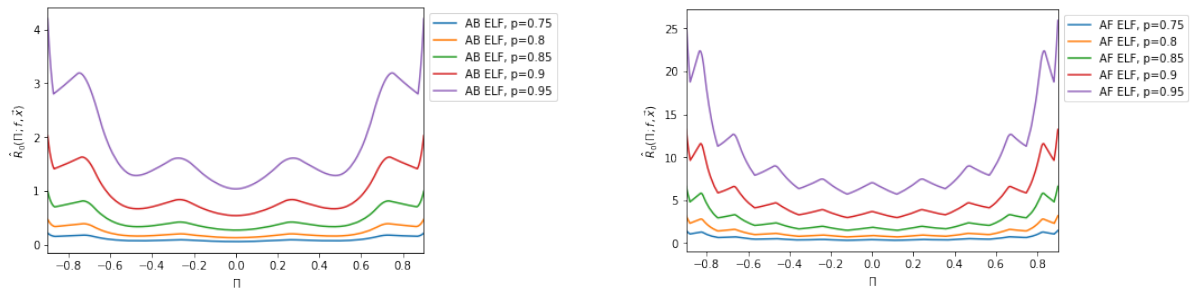


Figure 5.9: This figure shows the $\hat{R}_0$ factors of AB and AF ELFs for $\Pi \in [-0.9, 0.9]$, when the number L of circuit layers is 6, and the layer fidelity p is varied from $0.75, 0.8, 0.85, 0.9$ to 0.95. Note that higher layer fidelity leads to larger $\hat{R}_0$ factors of AB and AF ELFs.

5.3.2 Analyzing the impact of circuit depth on the performance of estimation

To investigate the influence of circuit depth on the performance of estimation, we run Bayesian inference with AB and AF ELFs for fixed layer fidelity but varied circuit depth. Specifically, we set the layer fidelity p to be 0.9, and vary the number L of circuit layers from 1 to 5. Figs. 5.10 and 5.11 illustrate the simulation results in the ancilla-based and ancilla-free cases, respectively. These results indicate that larger L (i.e. deeper circuit) does not necessarily lead to better performance. The optimal choice of L is indeed a subtle issue. This can be intuitively understood as follows. As L increases, the likelihood function becomes steeper [8] and hence the information gain from the measurement outcome becomes larger, if the circuit is noiseless. But on the other hand, the true fidelity of the circuit decreases exponentially in L and the implementation cost of this circuit grows linearly in L. So one must find a perfect balance among these factors in order to maximize the performance of the algorithm.

The above results are consistent with Figs. 5.12 that illustrate the $\hat{R}_0$ factors of AB and AF ELFs in the same setting. Note that larger L does not necessarily lead to larger $\hat{R}_0$ factor of AB or AF ELF. One can evaluate this factor for different L's and choose the one that maximizes this factor. This often enables us to find a satisfactory (albeit not necessarily optimal) L. It remains an open question to devise an efficient strategy for determining the best L that optimizes the performance of estimation given the layer fidelity p and a prior distribution of Π.

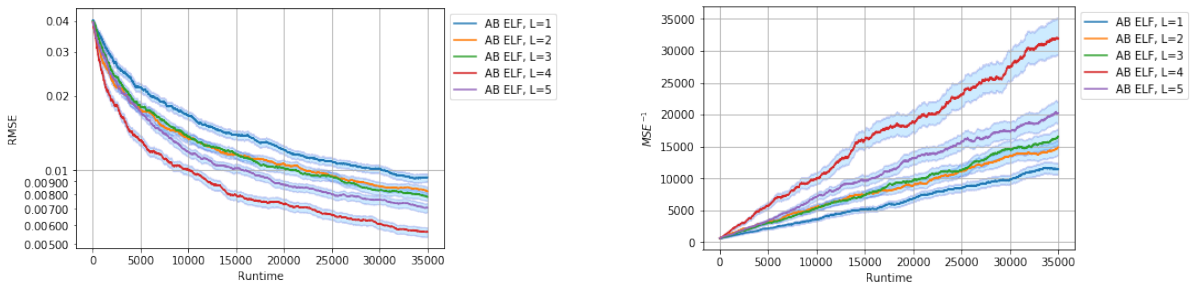


Figure 5.10: This figure demonstrates the impact of circuit depth on the performance of AB ELF. Here Π has true value 0.35 and prior distribution $\mathcal{N}(0.39, 0.0016)$, the layer fidelity p is 0.9, and the number L of layers is varied from 1 to 5. Note that the best performance is achieved by $L = 4$ instead of $L = 5$.

6 A Model for Noisy Algorithm Performance

Our aim is to build a model for the runtime needed to achieve a target mean-squared error in the estimate of Π as it is scaled to larger systems and run on devices with better gate fidelities. This model will be

[8] More precisely, the slopes of AB and AF ELFs scale linearly in the number L of circuit layers.

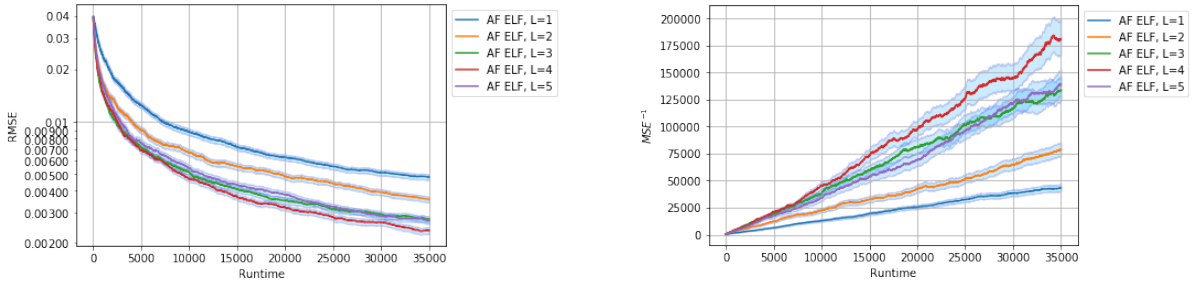


Figure 5.11: This figure demonstrates the impact of circuit depth on the performance of AF ELF. Here Π has true value 0.35 and prior distribution $\mathcal{N}(0.39, 0.0016)$, the layer fidelity p is 0.9, and the number L of layers is varied from 1 to 5. Note that the best performance is achieved by $L = 4$ instead of $L = 5$.

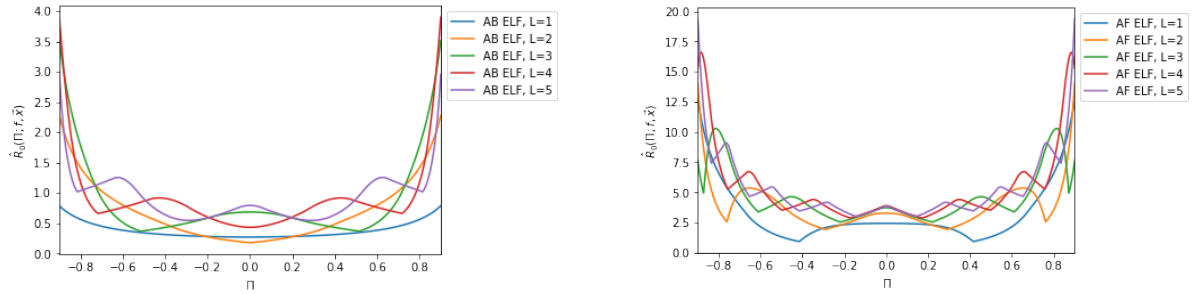


Figure 5.12: This figure shows the $\hat{R}_0$ factors of AB and AF ELFs for $\Pi \in [-0.9, 0.9]$, when the number L of circuit layers is varied from 1 to 5, and the layer fidelity p is 0.9. Note that the best L that maximizes the $\hat{R}_0$ factor depends heavily on the value of Π in both the ancilla-based and ancilla-free cases.

built on two main assumptions. The first is that the growth rate of the inverse mean squared error is well-described by half the inverse variance rate expression (c.f. Eq. (36)). The half is due to the conservative estimate that the variance and squared bias contribute equally to the mean squared error (simulations from the previous section show that the squared bias tends to be less than the variance). The second assumption is an empirical lower bound on the variance reduction factor, which is motivated by numerical investigations of the Chebyshev likelihood function.

We carry out analysis for the MSE with respect to the estimate of θ. We will then convert the MSE of this estimate to an estimate of MSE with respect to Π. Our strategy will be to integrate upper and lower bounds for the rate expression $R(\mu, \sigma; f, m)$ in Eq. (36) to arrive at bounds for inverse MSE as a function of time. To help our analysis we make the substitution $m = T(L) = 2L + 1$ and reparameterize the way noise is incorporated by introducing λ and α such that $f^2 = \bar{p}^2 p^{2L} = e^{-\lambda(2L+1)-\alpha} = e^{-\lambda m - \alpha}$.

The upper and lower bounds on this rate expression are based on findings for the Chebyshev likelihood functions, where $\vec{x} = (\pi/2)^{2L}$. Since the Chebyshev likelihood functions are a subset of the engineered likelihood functions, a lower bound on the Chebyshev performance gives a lower bound on the ELF performance. We leave as a conjecture that the upper bound for this rate in the case of ELF is a small multiple (e.g. 1.5) of the upper bound we have established for the Chebyshev rate.

The Chebyshev upper bound is established as follows. For fixed σ, λ, and m, one can show[9] that the variance reduction factor achieves a maximum value of $\mathcal{V} = m^2 \exp(-m^2\sigma^2 - \lambda m - \alpha)$, occurring at $\mu = \pi/2$. This expression is less than $m^2 \mathrm{e}^{-m^2\sigma^2}$, which achieves a maximum of $(\mathrm{e}\sigma^2)^{-1}$ at $m = \frac{1}{\sigma}$. Thus, the factor $1/(1-\sigma^2\mathcal{V})$ cannot exceed $1/(1-e^{-1}) \approx 1.582$. Putting this all together, for fixed σ, λ, and m, the maximum rate is upper bounded as $R(\mu, \sigma; \lambda, \alpha, m) \leq \frac{em}{e-1} \exp(-m^2\sigma^2 - \lambda m - \alpha)$. This follows from the fact that R

[9] For the Chebyshev likelihood functions, we can express the variance reduction factor as $\mathcal{V}(\mu, \sigma; f, (\frac{\pi}{2})^{2L}) = m_L^2 / \left(1 + \left(f^{-2}\mathrm{e}^{m_L^2\sigma^2} - 1\right)\csc^2(m_L\mu)\right)$ whenever $\sin(m_L\mu) \neq 0$. Then, $\csc^2(m_L\mu) \geq 1$ implies that $\mathcal{V}(\mu, \sigma; f, (\frac{\pi}{2})^{2L}) \leq f^2 m_L^2 \mathrm{e}^{-m_L^2\sigma^2}$.

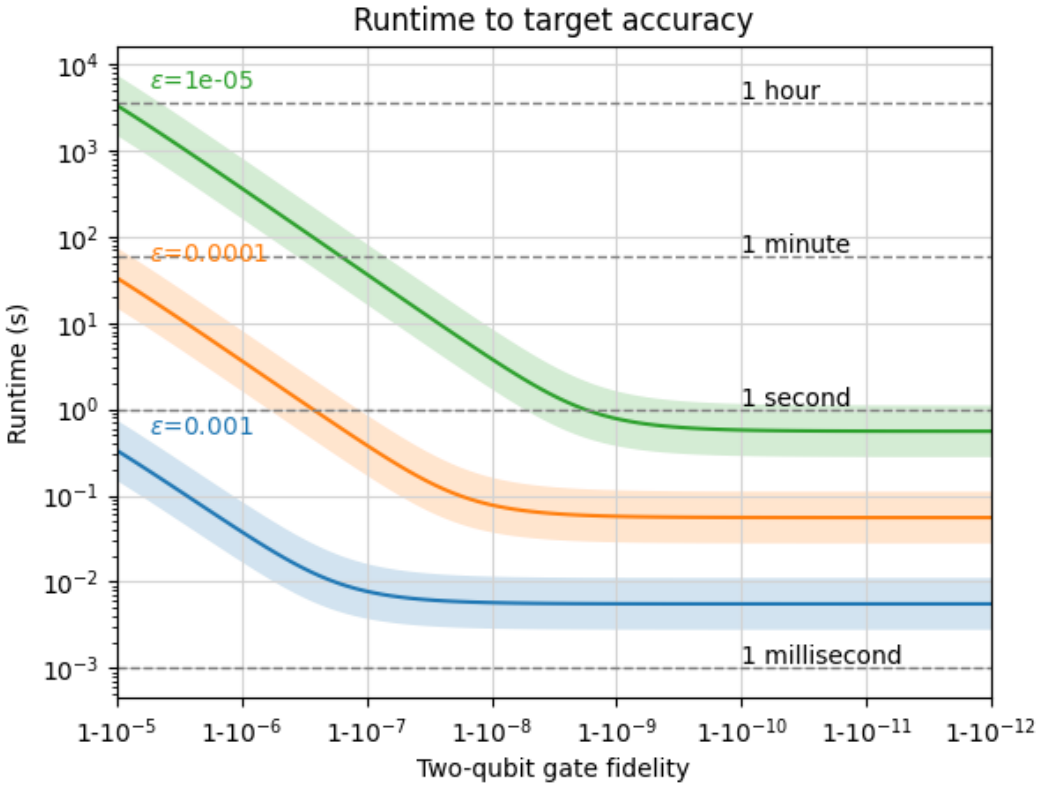


Figure 6.1: As two-qubit gate fidelities are improved, deeper enhanced sampling circuits warrant being implemented, yielding shorter estimation runtimes. Here, we consider the case of $n = 100$ qubits, $D = 200$ two-qubit gate depth per layer, and target accuracies of $\varepsilon = 10^{-3}$, $\varepsilon = 10^{-4}$, and $\varepsilon = 10^{-5}$. The bands indicate the upper and lower bounds of Eq. 81.

is monotonic in $\mathcal{V}$ and that $\mathcal{V}$ is maximized at $\mu = \pi/2$. In practice, we will aim to choose a value of L that maximizes the inverse variance rate. The rate achieved by discrete L cannot exceed the value we obtain when optimizing the above upper bound over continuous value of m. This optimal value is realized for $1/m = \frac{1}{2}\left(\sqrt{\lambda^2 + 8\sigma^2} + \lambda\right)$. We define $\bar{R}(\sigma; \lambda, \alpha)$ by evaluating $R(\pi/2, \sigma; \lambda, \alpha, m)$ at this optimum value,

$$\bar{R}(\sigma; \lambda, \alpha) = \frac{2e^{-\alpha-1}}{\sqrt{\lambda^2 + 8\sigma^2} + \lambda} \exp\left(\frac{2\sigma^2}{4\sigma^2 + \lambda^2 + \lambda^2\sqrt{8\sigma^2/\lambda^2 + 1}}\right), \tag{72}$$

which gives the upper bound on the Chebyshev rate

$$R_C^*(\mu, \sigma; \lambda, \alpha) = \max_L R(\mu, \sigma; \lambda, \alpha, m) \leq \frac{e}{e-1}\bar{R}(\sigma; \lambda, \alpha). \tag{73}$$

We do not have an analytic lower bound on the Chebyshev likelihood performance. We can establish an empirical lower bound based on numerical checks. For any fixed L, the inverse variance rate is zero at the $2L+2$ points $\mu \in \{0, \pi/(2L+1), 2\pi/(2L+1), \ldots, 2L\pi/(2L+1), \pi\}$. Since the rate is zero at these end points for all L, the global lower bound on R_C^* is zero. However, we are not concerned with the poor performance of the inverse variance rate near these end points. When we convert the estimator from $\hat{\theta}$ to $\hat{\Pi} = \cos\hat{\theta}$, the information gain near these end point actually tends to a large value. For the purpose of establishing useful bounds, we will restrict μ to be in the range $[0.1\pi, 0.9\pi]$. In the numerical tests [10] we find that for all $\mu \in [0.1\pi, 0.9\pi]$, there is always a choice of L for which the inverse variance rate is above $(e-1)^2/e^2 \approx 0.40$ times the upper bound. Putting these together, we have

$$\frac{e-1}{e}\bar{R}(\sigma; \lambda, \alpha) \leq R_C^*(\mu, \sigma; \lambda, \alpha) \leq \frac{e}{e-1}\bar{R}(\sigma; \lambda, \alpha). \tag{74}$$

It is important to note that by letting m be continuous, certain values of σ and λ can lead to an optimal m for which $L = (m-1)/2$ is negative. Therefore, these results apply only in the case that $\lambda \leq 1$, which ensures that $m \geq 1$. We expect this model to break down in the large-noise regime (i.e. $\lambda \geq 1$).

For now, we will assume that the rate tracks the geometric mean of these two bounds, i.e. $R_C^*(\sigma, \lambda, \mu) = \bar{R}(\sigma, \lambda)$, keeping in mind that the upper and lower bounds are small constant factors off of this. We assume that the inverse variance grows continuously in time at a rate given by the difference quotient expression captured by the inverse-variance rate, $R^* = \frac{\mathrm{d}}{\mathrm{d}t}\frac{1}{\sigma^2}$. Letting $F = 1/\sigma^2$ denote this inverse variance, the rate equation above can be recast as a differential equation for F,

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \frac{2e^{-\alpha-1}}{\lambda\sqrt{1 + 8/(F\lambda^2)} + \lambda} \exp\left(\frac{2}{4 + \lambda^2 F + \lambda^2 F\sqrt{1 + 8/(\lambda^2 F)}}\right). \tag{75}$$

Through this expression, we can identify both the Heisenberg limit behavior and shot-noise limit behavior. For $F \ll 1/\lambda^2$, the differential equation becomes

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \frac{e^{-\alpha-1/2}}{\sqrt{2}}\sqrt{F}, \tag{76}$$

which integrates to a quadratic growth of the inverse squared error $F(t) \sim t^2$. This is the signature of the Heisenberg limit regime. For $F \gg 1/\lambda^2$, the rate approaches a constant,

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \frac{e^{-\alpha-1}}{\lambda}. \tag{77}$$

This regime yields a linear growth in the inverse squared error $F(t) \sim t$, indicative of the shot-noise limit regime.

[10] We searched over a uniform grid of 50000 values of θ, L values from $L^*/3$ to $3L^*$, where L^* is to the optimized value used to arrive at Eq. 72, and σ and λ ranging over $[10^{-1}, 10^{-2}, \ldots, 10^{-5}]$. For each (σ, λ) pair we found the θ for which the maximum inverse variance rate (over L) is a minimum. For all (σ, λ) pairs checked, this worst-case rate was always between 0.4 and 0.5, with the smallest value found being $R = 0.41700368 \geq (e-1)^2/e^2$.

In order to make the integral tractable, we can replace the rate expression with integrable upper and lower bound expressions (to be used in tandem with our previous bounds). Letting $x = \lambda^2 F$, these bounds are re-expressed as,

$$\frac{2e^{-\alpha-1}\lambda}{1+1/\sqrt{12x}+(x+4)/\sqrt{x^2+8x}} \geq \frac{\mathrm{d}x}{\mathrm{d}t} \geq \frac{2e^{-\alpha-1}\lambda}{1+1/\sqrt{4x}+(x+4)/\sqrt{x^2+8x}}. \tag{78}$$

From the upper bound we can establish a lower bound on the runtime, by treating time as a function of x and integrating,

$$\int_0^t dt \geq \int_{x_0}^{x_f} \mathrm{d}x \frac{e^{\alpha+1}}{2\lambda}\left(1+1/\sqrt{12x}+(x+4)/\sqrt{x^2+8x}\right) \tag{79}$$

$$= \frac{e^{\alpha+1}}{2\lambda}\left(x_f+\sqrt{x_f/3}+\frac{1}{2}\sqrt{x_f^2+8x_f}-x_0-\sqrt{x_0/3}-\frac{1}{2}\sqrt{x_0^2+8x_0}\right). \tag{80}$$

Similarly, we can use the lower bound to establish an upper bound on the runtime. Here we introduce our assumption that, in the worst case, the MSE of the phase estimate ε_θ^2 is twice the variance (i.e. the variance equals the bias), so the variance must reach half the MSE: $\sigma^2 = \varepsilon_\theta^2/2 = \lambda^2/x$. In the best case, we assume the bias in the estimate is zero and set $\varepsilon_\theta^2 = \lambda^2/x$. We combine these bounds with the upper and lower bounds of Eq. (74) to arrive at the bounds on the estimation runtime as a function of target MSE,

$$(e-1)\frac{e^{-\lambda}}{2\bar{p}^2}\left(\frac{\lambda}{\varepsilon_\theta^2}+\frac{1}{\sqrt{3}\varepsilon_\theta}+\sqrt{\left(\frac{\lambda}{\varepsilon_\theta^2}\right)^2+\left(\frac{2\sqrt{2}}{\varepsilon_\theta}\right)^2}\right) \leq t_{\varepsilon_\theta} \leq \frac{e^2}{e-1}\frac{e^{-\lambda}}{\bar{p}^2}\left(\frac{\lambda}{\varepsilon_\theta^2}+\frac{1}{\sqrt{2}\varepsilon_\theta}+\sqrt{\left(\frac{\lambda}{\varepsilon_\theta^2}\right)^2+\left(\frac{2}{\varepsilon_\theta}\right)^2}\right), \tag{81}$$

where $\theta \in [0.1\pi, 0.9\pi]$.

At this point, we can convert our phase estimate $\hat{\theta}$ back into the amplitude estimate $\hat{\Pi}$. The MSE with respect to the amplitude estimate ε_Π^2 can be approximated in terms of the phase estimate MSE as

$$\begin{aligned}
\varepsilon_\Pi^2 &= \mathbb{E}(\hat{\Pi}-\Pi)^2 \\
&= \mathbb{E}(\cos\hat{\theta}-\cos\theta)^2 \\
&\approx \mathbb{E}((\hat{\theta}-\theta)\frac{\mathrm{d}\cos\theta}{\mathrm{d}\theta})^2 \\
&= \varepsilon_\theta^2 \sin^2\theta,
\end{aligned} \tag{82}$$

where we have assumed that the distribution of the estimator is sufficiently peaked about θ to ignore higher-order terms. This leads to $\varepsilon_\theta^2 = \varepsilon_\Pi^2/(1-\Pi^2)$, which can be substituted into the above expressions for the bounds, which hold for $\Pi \in [\cos 0.9\pi, \cos 0.1\pi] \approx [-0.95, 0.95]$. Dropping the estimator subscripts (as they only contribute constant factors), we can establish the runtime scaling in the low-noise and high-noise limits,

$$t_\varepsilon = \begin{cases} O(e^\alpha/\varepsilon) & \lambda \ll \varepsilon, \\ O(e^\alpha\lambda/\varepsilon^2) & \lambda \gg \varepsilon, \end{cases} \tag{83}$$

observing that the Heisenberg-limit scaling and shot-noise limit scaling are each recovered.

We arrived at these bounds using properties of Chebyshev likelihood functions. As we have shown in the previous section, by engineering likelihood functions, in many cases we can reduce estimation runtimes. Motivated by our numerical findings of the variance reduction factors of engineered likelihood functions (see, e.g. Fig. 5.6), we conjecture that using engineered likelihood functions increases the worst case inverse-variance rate in Eq. (74) to $\bar{R}(\sigma;\lambda,\alpha) \leq R_C^*(\mu,\sigma;\lambda,\alpha)$.

In order to give more meaning to this model, we will refine it to be in terms of number of qubits n and two-qubit gate fidelities f_{2Q}. We consider the task of estimating the expectation value of a Pauli string P with respect to state $|A\rangle$. Assume that $\Pi = \langle A|P|A\rangle$ is very near zero so that $\varepsilon^2 = \varepsilon_\Pi^2 \approx \varepsilon_\theta^2$. Let the two-qubit gate depth of each of the L layers be D. We model the total layer fidelity as $p = f_{2Q}^{nD/2}$, where we have ignored

errors due to single-qubit gates. From this, we have $\lambda = \frac{1}{2} nD \ln(1/f_{2Q})$ and $\alpha = 2\ln(1/\bar{p}) - \frac{1}{2} nD \ln(1/f_{2Q})$. Putting these together, we arrive at the runtime expression,

$$t_\varepsilon = e \frac{f_{2Q}^{nD/2}}{2\bar{p}^2} \left(\frac{nD \ln(1/f_{2Q})}{2\varepsilon^2} + \frac{1}{\sqrt{6}\varepsilon} + \sqrt{\left(\frac{nD \ln(1/f_{2Q})}{2\varepsilon^2} \right)^2 + \left(\frac{2\sqrt{2}}{\varepsilon} \right)^2} \right). \tag{84}$$

Finally, we will put some meaningful numbers in this expression and estimate the required runtime in seconds as a function of two-qubit gate fidelities. To achieve quantum advantage we expect that the problem instance will require on the order of $n = 100$ logical qubits and that the two-qubit gate depth is on the order of the number of qubits, $D = 200$. Furthermore, we expect that target accuracies ε will need to be on the order of $\varepsilon = 10^{-3}$ to 10^{-5}. The runtime model measures time in terms of ansatz circuit durations. To convert this into seconds we assume each layer of two-qubit gates will take time $G = 10^{-8}$s, which is an optimistic assumption for today's superconducting qubit hardware. Figure 6.1 shows this estimated runtime as a function of two-qubit gate fidelity.

The two-qubit gate fidelities required to reduce runtimes into a practical region will most likely require error correction. Performing quantum error correction requires an overhead that increases these runtimes. In designing quantum error correction protocols, it is essential that the improvement in gate fidelities is not outweighed by the increase in estimation runtime. The proposed model gives a means of quantifying this trade-off: the product of gate infidelity and (error-corrected) gate time should decrease as useful error correction is incorporated. In practice, there are many subtleties that should be accounted for to make a more rigorous statement. These include considering the variation in gate fidelities among gates in the circuit and the varying time costs of different types of gates. Nevertheless, the cost analyses afforded by this simple model may a useful tool in the design of quantum gates, quantum chips, error correcting schemes, and noise mitigation schemes.

7 Outlook

This work was motivated by the impractical runtimes required by many NISQ-amenable quantum algorithms. We aimed to improve the performance of estimation subroutines that have relied on standard sampling, as used in VQE. Drawing on the recent alpha-VQE [16] and quantum metrology [36], we investigated the technique of enhanced sampling to explore the continuum between standard sampling and quantum amplitude (or phase) estimation. In this continuum, we can make optimal use of the quantum coherence available on a given device to speed up estimation. Similar to standard sampling in VQE, enhanced sampling does not require ancilla qubits. Quantum advantage for tasks relying on estimation will likely occur within this continuum rather than at one of the extremes.

Our central object of study was the quantum generated likelihood function, relating measurement outcome data to a parameter of interest encoded in a quantum circuit. We explored engineering likelihood functions to optimize their statistical power. This led to several insights for improving estimation. First, we should incorporate a well-calibrated noise model directly in the likelihood function to make inference robust to certain error. Second, we should choose a circuit depth (reflected in the number of enhanced sampling circuit layers) that balances gain in statistical power with accrual of error. Finally, we should tune generalized reflection angles to mitigate the effect of "deadspots" during the inference process.

We used engineered likelihood functions to carry out adaptive approximate Bayesian inference for parameter estimation. Carrying out this process in simulation required us to build mathematical and algorithmic infrastructure. We developed mathematical tools for analyzing a class of quantum generated likelihood functions. From this analysis, we proposed several optimization algorithms for tuning circuit parameters to engineer likelihood functions. We investigated the performance of estimation using engineered likelihood functions and compared this to estimation using fixed likelihood functions. Finally, we proposed a model for predicting the performance of enhanced sampling estimation algorithms as the quality of quantum devices is improved.

These simulations and the model led to several insights. As highlighted in Section 5.2, for the degree of device error expected in the near-term (two-qubit gate fidelities of $\sim 99.92\%$), we have shown that enhanced sampling and engineered likelihood functions can be used to outperform standard sampling used in VQE.

Furthermore, these simulations suggest a non-standard perspective on the tolerance of error in quantum algorithm implementations. We found that, for fixed gate fidelities, the best performance is achieved when we push circuit depths to a point where circuit fidelities are around the range of $0.5 - 0.7$. This suggests that, compared to the logical circuit fidelities suggested in other works (e.g. 0.99 in [15]), we can afford a 50-fold increase in circuit depth. We arrive at this balance between fidelity and statistical power by taking estimation runtime to be the cost to minimize.

The runtime model developed in Section 6 sheds light on the trade-off between gate times and gate fidelity for estimation. For gate times that are one-thousand times slower, the gate fidelities must have three more nines to achieve the same estimation runtimes. The runtime model gives insight on the role of quantum error correction in estimation algorithms. Roughly, we found that for quantum error correction to be useful for estimation, the factor of increase in runtime from error correction overhead must be less than the factor of decrease in logical gate error rates. Additionally, the runtime model predicts that for a given estimation task, there is a level of logical gate fidelity beyond which further improvements do not reduce runtimes. For the 100-qubit example considered, seven nines in two-qubit gate fidelities sufficed.

We leave a number of questions for future investigation. In VQE a set of techniques referred to as "grouping" are used to reduce the measurement count [39, 45–48]. These grouping techniques allow sampling of multiple operators at once, providing a type of measurement parallelization. The grouping method introduced in [46, 48] decomposes a Pauli Hamiltonian into sets of mutually-anticommuting Pauli strings, which ensures that the sum within each set is a Hermitian reflection. This method of grouping is compatible with enhanced sampling, as the Hermitian reflections can be both measured and implemented as generalized reflections (i.e. an example of operator P). However, it remains to explore if the additional circuit depth incurred by implementing these reflections is worth the variance reduction in the resulting estimators. Beyond existing grouping techniques, we anticipate opportunities for parallelizing measurements that are dedicated to enhanced sampling.

Our work emphasizes the importance of developing accurate error models at the algorithmic level. Efficiently learning the "nuisance parameters" of these models, or *likelihood function calibration*, will be an essential ingredient to realizing the performance gain of enhanced sampling in the near term. The motivation is similar to that of randomized benchmarking [49], where measurement data is fit to models of gate-noise. An important problem that we leave for future work is to improve methods for likelihood function calibration. Miscalibration can introduce a systematic bias in parameter estimates. A back-of-the-envelope calculation predicts that the relative error in estimation due to miscalibration bias is roughly the relative error in miscalibration of the depolarizing. In future work we will explore to what precision we must calibrate the likelihood function so that the bias introduced by miscalibration (or model error) is negligible. Finally, we leave investigations into analytical upper and lower bounds on estimation performance to future work.

We have aimed to present a viable solution to the "measurement problem" that plagues VQE. It is likely that these methods will be needed to achieve quantum advantage for problems in quantum chemistry and materials. We hope that our model for estimation performance as a function of device metrics is useful in assessing the relative importance of a variety of quantum resources including qubit number, two-qubit gate fidelity, gate times, qubit stability, error correction cycle time, readout error rate, and others.

Acknowledgments

We would like to thank Pierre-Luc Dallaire-Demers, Amara Katabarwa, Jhonathan Romero, Max Radin, Peter Love, Yihui Quek, Jonny Olson, Hannah Sim, and Jens Eisert for insightful conversations and valuable feedback, and Christopher Savoie for grammatical tidying.

References

[1] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L Obrien. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5:4213, 2014.

[2] Dave Wecker, Matthew B. Hastings, and Matthias Troyer. Progress towards practical quantum variational algorithms. *Physical Review A*, 92(4), October 2015.

[3] Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, 2016.

[4] Jonathan Romero, Ryan Babbush, Jarrod R McClean, Cornelius Hempel, Peter J Love, and Alán Aspuru-Guzik. Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. *Quantum Science and Technology*, 4(1):014008, oct 2018.

[5] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[6] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, February 2019.

[7] Carlos Bravo-Prieto, Ryan LaRose, Marco Cerezo, Yigit Subasi, Lukasz Cincio, and Patrick J Coles. Variational quantum linear solver: A hybrid algorithm for linear systems. *arXiv preprint arXiv:1909.05820*, 2019.

[8] Dong An and Lin Lin. Quantum linear system solver based on time-optimal adiabatic quantum computing and quantum approximate optimization algorithm. *arXiv preprint arXiv:1909.05500*, 2019.

[9] Xiaosi Xu, Jinzhao Sun, Suguru Endo, Ying Li, Simon C Benjamin, and Xiao Yuan. Variational algorithms for linear algebra. *arXiv preprint arXiv:1909.03898*, 2019.

[10] Ying Li and Simon C. Benjamin. Efficient variational quantum simulator incorporating active error minimization. *Physical Review X*, 7(2), June 2017.

[11] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, aug 2017.

[12] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.*, 122:040504, Feb 2019.

[13] D. Zhu, N. M. Linke, M. Benedetti, K. A. Landsman, N. H. Nguyen, C. H. Alderete, A. Perdomo-Ortiz, N. Korda, A. Garfoot, C. Brecque, L. Egan, O. Perdomo, and C. Monroe. Training of quantum circuits on a hybrid quantum computer. *Science Advances*, 5(10), 2019.

[14] William J Huggins, Jarrod McClean, Nicholas Rubin, Zhang Jiang, Nathan Wiebe, K Birgitta Whaley, and Ryan Babbush. Efficient and noise resilient measurements for quantum chemistry on near-term quantum computers. *arXiv preprint arXiv:1907.13117*, 2019.

[15] Ryan Babbush, Craig Gidney, Dominic W. Berry, Nathan Wiebe, Jarrod McClean, Alexandru Paler, Austin Fowler, and Hartmut Neven. Encoding electronic spectra in quantum circuits with linear t complexity. *Physical Review X*, 8(4), October 2018.

[16] Daochen Wang, Oscar Higgott, and Stephen Brierley. Accelerated variational quantum eigensolver. *Physical review letters*, 122(14):140504, 2019.

[17] Emanuel Knill, Gerardo Ortiz, and Rolando D Somma. Optimal quantum measurements of expectation values of observables. *Physical Review A*, 75(1):012328, 2007.

[18] Alexandr Sergeevich, Anushya Chandran, Joshua Combes, Stephen D. Bartlett, and Howard M. Wiseman. Characterization of a qubit hamiltonian using adaptive measurements in a fixed basis. *Physical Review A*, 84(5), November 2011.

[19] Christopher Ferrie, Christopher E. Granade, and D. G. Cory. How to best sample a periodic probability distribution, or on the accuracy of hamiltonian finding strategies. *Quantum Information Processing*, 12(1):611–623, April 2012.

[20] Krysta M Svore, Matthew B Hastings, and Michael Freedman. Faster phase estimation. *arXiv preprint arXiv:1304.0741*, 2013.

[21] Nathan Wiebe and Chris Granade. Efficient bayesian phase estimation. *Physical review letters*, 117(1):010503, 2016.

[22] V. Giovannetti. Quantum-enhanced measurements: Beating the standard quantum limit. *Science*, 306(5700):1330–1336, November 2004.

[23] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Advances in quantum metrology. *Nature Photonics*, 5(4):222–229, March 2011.

[24] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum counting. In *International Colloquium on Automata, Languages, and Programming*, pages 820–831. Springer, 1998.

[25] Theodore J Yoder, Guang Hao Low, and Isaac L Chuang. Fixed-point quantum search with an optimal number of queries. *Physical review letters*, 113(21):210501, 2014.

[26] Guang Hao Low, Theodore J Yoder, and Isaac L Chuang. Methodology of resonant equiangular composite quantum gates. *Physical Review X*, 6(4):041067, 2016.

[27] Guang Hao Low and Isaac L. Chuang. Hamiltonian simulation by qubitization. *Quantum*, 3:163, July 2019.

[28] Guang Hao Low and Isaac L. Chuang. Optimal Hamiltonian simulation by quantum signal processing. *Physical Review Letters*, 118(1), January 2017.

[29] Thomas E O'Brien, Brian Tarasinski, and Barbara M Terhal. Quantum phase estimation of multiple eigenvalues for small-scale (noisy) experiments. *New Journal of Physics*, 21(2):023022, feb 2019.

[30] A. Yu. Kitaev. Quantum measurements and the abelian stabilizer problem, 1995.

[31] Zhihui Wang, Stuart Hadfield, Zhang Jiang, and Eleanor G Rieffel. Quantum approximate optimization algorithm for maxcut: A fermionic view. *Physical Review A*, 97(2):022304, 2018.

[32] Z. Ji, G. Wang, R. Duan, Y. Feng, and M. Ying. Parameter estimation of quantum channels. *IEEE Transactions on Information Theory*, 54(11):5172–5185, 2008.

[33] Ilia Zintchenko and Nathan Wiebe. Randomized gap and amplitude estimation. *Physical Review A*, 93(6), June 2016.

[34] Yohichi Suzuki, Shumpei Uno, Rudy Raymond, Tomoki Tanaka, Tamiya Onodera, and Naoki Yamamoto. Amplitude estimation without phase estimation. *Quantum Information Processing*, 19(2), January 2020.

[35] Joel J Wallman and Joseph Emerson. Noise tailoring for scalable quantum computation via randomized compiling. *Physical Review A*, 94(5):052325, 2016.

[36] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum metrology. *Physical review letters*, 96(1):010401, 2006.

[37] Ryan Sweke, Frederik Wilde, Johannes Meyer, Maria Schuld, Paul K Fährmann, Barthélémy Meynard-Piganeau, and Jens Eisert. Stochastic gradient descent for hybrid quantum-classical optimization. *arXiv preprint arXiv:1910.01155*, 2019.

[38] Kevin J Sung, Matthew P Harrigan, Nicholas C Rubin, Zhang Jiang, Ryan Babbush, and Jarrod R McClean. An exploration of practical optimizers for variational quantum algorithms on superconducting qubit processors. *arXiv preprint arXiv:2005.11011*, 2020.

[39] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M Chow, and Jay M Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549(7671):242–246, 2017.

[40] Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge University Press, 2003.

[41] Richard Royall. On the probability of observing misleading statistical evidence. *Journal of the American Statistical Association*, 95(451):760–768, 2000.

[42] Jay M Gambetta, AD Córcoles, Seth T Merkel, Blake R Johnson, John A Smolin, Jerry M Chow, Colm A Ryan, Chad Rigetti, S Poletto, Thomas A Ohki, et al. Characterization of addressability by simultaneous randomized benchmarking. *Physical review letters*, 109(24):240504, 2012.

[43] Dax Enshan Koh, Guoming Wang, Peter D Johnson, and Yudong Cao. A framework for engineering quantum likelihood functions for expectation estimation. *arXiv preprint*, 2020.

[44] Christopher E Granade, Christopher Ferrie, Nathan Wiebe, and David G Cory. Robust online hamiltonian learning. *New Journal of Physics*, 14(10):103013, 2012.

[45] Vladyslav Verteletskyi, Tzu-Ching Yen, and Artur F Izmaylov. Measurement optimization in the variational quantum eigensolver using a minimum clique cover. *The Journal of Chemical Physics*, 152(12):124114, 2020.

[46] Artur F Izmaylov, Tzu-Ching Yen, Robert A Lang, and Vladyslav Verteletskyi. Unitary partitioning approach to the measurement problem in the variational quantum eigensolver method. *Journal of chemical theory and computation*, 2019.

[47] Ophelia Crawford, Barnaby van Straaten, Daochen Wang, Thomas Parks, Earl Campbell, and Stephen Brierley. Efficient quantum measurement of pauli operators. *arXiv preprint arXiv:1908.06942*, 2019.

[48] Andrew Zhao, Andrew Tranter, William M Kirby, Shu Fay Ung, Akimasa Miyake, and Peter Love. Measurement reduction in variational quantum algorithms. *arXiv preprint arXiv:1908.08067*, 2019.

[49] Ian Hincks, Joel J Wallman, Chris Ferrie, Chris Granade, and David G Cory. Bayesian inference for randomized benchmarking protocols. *arXiv preprint arXiv:1802.00401*, 2018.

A Proof of Lemma 1

For convenience, we introduce the following notation. Let $W_{2i} = U^\dagger(\theta; x_{2i+1}) = U(\theta; -x_{2i+1})$, $W_{2i+1} = V^\dagger(x_{2i+2}) = V(-x_{2i+2})$, $W_{4L-2i} = U(\theta; x_{2i+1})$, and $W_{4L-2i-1} = V(x_{2i+2})$, for $i = 0, 1, \ldots, L-1$, and $W_{2L} = P(\theta)$. Furthermore, let $W'_j = \partial_\theta W_j$ for $j = 0, 1, \ldots, 4L$. Note that $W'_j = 0$ if j is odd. Then we define $P_{a,b} = W_a W_{a+1} \ldots W_b$ if $0 \le a \le b \le 4L$, and $P_{a,b} = I$ otherwise.

With this notation, Eq. (20) implies that

$$Q^\dagger(\theta; \vec{x}) = P_{0,a-1} W_a P_{a+1,2L-1}, \quad \forall 0 \le a \le 2L-1, \tag{85}$$

$$Q(\theta; \vec{x}) = P_{2L+1,b-1} W_b P_{b+1,4L}, \quad \forall 2L+1 \le b \le 4L, \tag{86}$$

$$Q^\dagger(\theta; \vec{x}) P(\theta) Q(\theta; \vec{x}) = P_{0,a-1} W_a P_{a+1,b-1} W_b P_{b+1,4L}, \quad \forall 0 \le a < b \le 4L. \tag{87}$$

Moreover, taking the derivative of Eq. (20) with respect to θ yields

$$\begin{aligned} Q'(\theta; \vec{x}) &= \partial_\theta Q(\theta; \vec{x}) && (88) \\ &= V(x_{2L}) U'(\theta; x_{2L-1}) V(x_{2L-2}) U(\theta; x_{2L-3}) \ldots V(x_4) U(\theta; x_3) V(x_2) U(\theta; x_1) \\ &\quad + V(x_{2L}) U(\theta; x_{2L-1}) V(x_{2L-2}) U'(\theta; x_{2L-3}) \ldots V(x_4) U(\theta; x_3) V(x_2) U(\theta; x_1) \\ &\quad + \ldots \\ &\quad + V(x_{2L}) U(\theta; x_{2L-1}) V(x_{2L-2}) U(\theta; x_{2L-3}) \ldots V(x_4) U'(\theta; x_3) V(x_2) U(\theta; x_1) \\ &\quad + V(x_{2L}) U(\theta; x_{2L-1}) V(x_{2L-2}) U(\theta; x_{2L-3}) \ldots V(x_4) U(\theta; x_3) V(x_2) U'(\theta; x_1), && (89) \end{aligned}$$

where

$$U'(\theta; \alpha) = \partial_\theta U(\theta; \alpha) = -\mathrm{i} \sin(\alpha) P'(\theta) = \mathrm{i} \sin(\alpha) (\sin(\theta) \bar{Z} - \cos(\theta) \bar{X}) \tag{90}$$

is the derivative of $U(\theta;\alpha)$ with respect to θ, in which

$$P'(\theta) = -\sin(\theta)\,\bar{Z} + \cos(\theta)\,\bar{X} \tag{91}$$

is the derivative of $P(\theta)$ with respect to θ. It follows that

$$\begin{aligned} Q'(\theta;\vec{x}) &= P_{2L+1,2L+1}W'_{2L+2}P_{2L+3,4L} \\ &\quad + P_{2L+1,2L+3}W'_{2L+4}P_{2L+5,4L} \\ &\quad + \ldots \\ &\quad + P_{2L+1,4L-3}W'_{4L-2}P_{4L-1,4L} \\ &\quad + P_{2L+1,4L-1}W'_{4L}. \end{aligned} \tag{92}$$

The following facts will be useful. Suppose A, B and C are arbitrary linear operators on the Hilbert space $\mathcal{H} = \mathrm{span}\{|\bar{0}\rangle, |\bar{1}\rangle\}$. Then by direct calculation, one can verify that

$$\langle\bar{0}|\, AV(-x)BV(x)C\,|\bar{0}\rangle = \langle\bar{0}|\, A\left[\cos(x)\,\bar{I} + \mathrm{i}\sin(x)\,\bar{Z}\right] B\left[\cos(x)\,\bar{I} - \mathrm{i}\sin(x)\,\bar{Z}\right] C\,|\bar{0}\rangle \tag{93}$$

$$\begin{aligned} &= \frac{1}{2}[\cos(2x)\,\langle\bar{0}|\, A\left(B - \bar{Z}B\bar{Z}\right)C\,|\bar{0}\rangle \\ &\quad -\mathrm{i}\sin(2x)\,\langle\bar{0}|\, A\left(B\bar{Z} - \bar{Z}B\right)C\,|\bar{0}\rangle \\ &\quad + \langle\bar{0}|\, A\left(B + \bar{Z}B\bar{Z}\right)C\,|\bar{0}\rangle], \end{aligned} \tag{94}$$

$$\langle\bar{0}|\, AU(\theta;-x)BU(\theta;x)C\,|\bar{0}\rangle = \langle\bar{0}|\, A\left[\cos(x)\,\bar{I} + \mathrm{i}\sin(x)\,P(\theta)\right] B\left[\cos(x)\,\bar{I} - \mathrm{i}\sin(x)\,P(\theta)\right] C\,|\bar{0}\rangle \tag{95}$$

$$\begin{aligned} &= \frac{1}{2}[\cos(2x)\,\langle\bar{0}|\, A\left(B - P(\theta)BP(\theta)\right)C\,|\bar{0}\rangle \\ &\quad -\mathrm{i}\sin(2x)\,\langle\bar{0}|\, A\left(BP(\theta) - P(\theta)B\right)C\,|\bar{0}\rangle \\ &\quad + \langle\bar{0}|\, A\left(B + P(\theta)BP(\theta)\right)C\,|\bar{0}\rangle], \end{aligned} \tag{96}$$

and

$$\langle\bar{0}|\, AU(\theta;-x)BU'(\theta;x)C\,|\bar{0}\rangle = \langle\bar{0}|\, A\left[\cos(x)\,\bar{I} + \mathrm{i}\sin(x)\,P(\theta)\right] B\left[-\mathrm{i}\sin(x)\,P'(\theta)\right] C\,|\bar{0}\rangle \tag{97}$$

$$\begin{aligned} &= \frac{1}{2}[-\cos(2x)\,\langle\bar{0}|\, AP(\theta)BP'(\theta)C\,|\bar{0}\rangle \\ &\quad -\mathrm{i}\sin(2x)\,\langle\bar{0}|\, ABP'(\theta)C\,|\bar{0}\rangle \\ &\quad + \langle\bar{0}|\, AP(\theta)BP'(\theta)C\,|\bar{0}\rangle]. \end{aligned} \tag{98}$$

The following fact will be also useful. Taking the derivative of Eq. (22) with respect to θ yields

$$\begin{aligned} \Delta'(\theta;\vec{x}) &= \langle\bar{0}|\, Q^\dagger(\theta;\vec{x})P(\theta)Q'(\theta;\vec{x})\,|\bar{0}\rangle \\ &\quad + \langle\bar{0}|\, Q^\dagger(\theta;\vec{x})P'(\theta)Q(\theta;\vec{x})\,|\bar{0}\rangle \\ &\quad + \langle\bar{0}|\, (Q'(\theta;\vec{x}))^\dagger P(\theta)Q(\theta;\vec{x})\,|\bar{0}\rangle \end{aligned} \tag{99}$$

$$\begin{aligned} &= 2\,\mathrm{Re}(\langle\bar{0}|\, Q^\dagger(\theta;\vec{x})P(\theta)Q'(\theta;\vec{x})\,|\bar{0}\rangle) \\ &\quad + \langle\bar{0}|\, Q^\dagger(\theta;\vec{x})P'(\theta)Q(\theta;\vec{x})\,|\bar{0}\rangle\,. \end{aligned} \tag{100}$$

In order to evaluate $C_j(\theta;\vec{x}_{\neg j})$, $S_j(\theta;\vec{x}_{\neg j})$, $B_j(\theta;\vec{x}_{\neg j})$, $C'_j(\theta;\vec{x}_{\neg j})$, $S'_j(\theta;\vec{x}_{\neg j})$ and $B'_j(\theta;\vec{x}_{\neg j})$ for given θ and $\vec{x}_{\neg j}$, we consider the case j is even and the case j is odd separately.

- Case 1: $j = 2(t+1)$ is even, where $0 \le t \le L-1$. In this case, $W_{2t+1} = V(-x_j)$, and $W_{4L-2t-1} = V(x_j)$. Then by Eqs. (22), (87) and (94), we obtain

$$\Delta(\theta;\vec{x}) = \langle\bar{0}|\, P_{0,2t}V(-x_j)P_{2t+2,4L-2t-2}V(x_j)P_{4L-2t,4L}\,|\bar{0}\rangle \tag{101}$$

$$= C_j(\theta;\vec{x}_{\neg j})\cos(2x_j) + S_j(\theta;\vec{x}_{\neg j})\sin(2x_j) + B_j(\theta;\vec{x}_{\neg j}), \tag{102}$$

where

$$C_j(\theta;\vec{x}_{\neg j}) = \frac{1}{2}\langle\bar{0}|\, P_{0,2t}\left(P_{2t+2,4L-2t-2} - \bar{Z}P_{2t+2,4L-2t-2}\bar{Z}\right) P_{4L-2t,4L}\,|\bar{0}\rangle\,, \tag{103}$$

$$S_j(\theta;\vec{x}_{\neg j}) = -\frac{i}{2}\langle\bar{0}|\, P_{0,2t}\left(P_{2t+2,4L-2t-2}\bar{Z} - \bar{Z}P_{2t+2,4L-2t-2}\right) P_{4L-2t,4L}\,|\bar{0}\rangle\,, \tag{104}$$

$$B_j(\theta;\vec{x}_{\neg j}) = \frac{1}{2}\langle\bar{0}|\, P_{0,2t}\left(P_{2t+2,4L-2t-2} + \bar{Z}P_{2t+2,4L-2t-2}\bar{Z}\right) P_{4L-2t,4L}\,|\bar{0}\rangle\,. \tag{105}$$

Given θ and $\vec{x}_{\neg j}$, we first compute $P_{0,2t}$, $P_{2t+2,4L-2t-2}$ and $P_{4L-2t,4L}$ in $O(L)$ time. Then we calculate $C_j(\theta;\vec{x}_{\neg j})$, $S_j(\theta;\vec{x}_{\neg j})$ and $B_j(\theta;\vec{x}_{\neg j})$ by Eqs. (103-105). This procedure takes only $O(L)$ time.

Next, we show how to compute $C'_j(\theta;\vec{x}_{\neg j})$, $S'_j(\theta;\vec{x}_{\neg j})$ and $B'_j(\theta;\vec{x}_{\neg j})$. Using Eq. (92) and the fact $P_{a,b} = P_{a,4L-2t-2}W_{4L-2t-1}P_{4L-2t,b}$ for any $a \le 4L-2t-1 \le b$, we obtain

$$\begin{aligned} Q'(\theta;\vec{x}) &= P_{2L+1,2L+1}W'_{2L+2}P_{2L+3,4L-2t-2}W_{4L-2t-1}P_{4L-2t,4L} \\ &\quad + P_{2L+1,2L+3}W'_{2L+4}P_{2L+5,4L-2t-2}W_{4L-2t-1}P_{4L-2t,4L} \\ &\quad + \ldots \\ &\quad + P_{2L+1,4L-2t-3}W'_{4L-2t-2}W_{4L-2t-1}P_{4L-2t,4L} \\ &\quad + P_{2L+1,4L-2t-2}W_{4L-2t-1}W'_{4L-2t}P_{4L-2t+1,4L} \\ &\quad + \ldots \\ &\quad + P_{2L+1,4L-2t-2}W_{4L-2t-1}P_{4L-2t,4L-3}W'_{4L-2}P_{4L-1,4L} \\ &\quad + P_{2L+1,4L-2t-2}W_{4L-2t-1}P_{4L-2t,4L-1}W'_{4L}. \end{aligned} \tag{106}$$

Then it follows from Eqs. (85) and (106) that

$$Q^\dagger(\theta;\vec{x})P(\theta)Q'(\theta;\vec{x}) = A_t^{(1)}W_{2t+1}B_t^{(1)}W_{4L-2t-1}C_t^{(1)} + A_t^{(2)}W_{2t+1}B_t^{(2)}W_{4L-2t-1}C_t^{(2)}, \tag{107}$$

$$= A_t^{(1)}V(-x_j)B_t^{(1)}V(x_j)C_t^{(1)} + A_t^{(2)}V(-x_j)B_t^{(2)}V(x_j)C_t^{(2)}, \tag{108}$$

where

$$A_t^{(1)} = P_{0,2t}, \tag{109}$$

$$B_t^{(1)} = P_{2t+2,4L-2t-2}, \tag{110}$$

$$C_t^{(1)} = \sum_{k=0}^{t} P_{4L-2t,4L-2k-1}W'_{4L-2k}P_{4L-2k+1,4L} \tag{111}$$

$$= \sum_{k=0}^{t} P_{4L-2t,4L-2k-1}U'(\theta;x_{2k+1})P_{4L-2k+1,4L}, \tag{112}$$

$$A_t^{(2)} = P_{0,2t}, \tag{113}$$

$$B_t^{(2)} = \sum_{k=t+1}^{L-1} P_{2t+2,4L-2k-1}W'_{4L-2k}P_{4L-2k+1,4L-2t-2} \tag{114}$$

$$= \sum_{k=t+1}^{L-1} P_{2t+2,4L-2k-1}U'(\theta;x_{2k+1})P_{4L-2k+1,4L-2t-2}, \tag{115}$$

$$C_t^{(2)} = P_{4L-2t,4L}. \tag{116}$$

Meanwhile, we have

$$Q^\dagger(\theta;\vec{x})P'(\theta)Q(\theta;\vec{x}) = A_t^{(3)}W_{2t+1}B_t^{(3)}W_{4L-2t-1}C_t^{(3)} \tag{117}$$

$$= A_t^{(3)}V(-x_j)B_t^{(3)}V(x_j)C_t^{(3)}, \tag{118}$$

where

$$A_t^{(3)} = P_{0,2t}, \tag{119}$$
$$B_t^{(3)} = P_{2t+2,2L-1}P'(\theta)P_{2L+1,4L-2t-2}, \tag{120}$$
$$C_t^{(3)} = P_{4L-2t,4L}. \tag{121}$$

Combining the above facts with Eqs. (94) and (100) yields

$$\Delta'(\theta;\vec{x}) = C'_j(\theta;\vec{x}_{\neg j})\cos(2x_j) + S'_j(\theta;\vec{x}_{\neg j})\sin(2x_j) + B'_j(\theta;\vec{x}_{\neg j}), \tag{122}$$

where

$$\begin{aligned} C'_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Re}\left(\langle\bar{0}|\, A_t^{(1)}\left(B_t^{(1)} - \bar{Z}B_t^{(1)}\bar{Z}\right)C_t^{(1)}\,|\bar{0}\rangle\right) \\ &\quad +\mathrm{Re}\left(\langle\bar{0}|\, A_t^{(2)}\left(B_t^{(2)} - \bar{Z}B_t^{(2)}\bar{Z}\right)C_t^{(2)}\,|\bar{0}\rangle\right) \\ &\quad +\frac{1}{2}\langle\bar{0}|\, A_t^{(3)}\left(B_t^{(3)} - \bar{Z}B_t^{(3)}\bar{Z}\right)C_t^{(3)}\,|\bar{0}\rangle, \end{aligned} \tag{123}$$

$$\begin{aligned} S'_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Im}\left(\langle\bar{0}|\, A_t^{(1)}\left(B_t^{(1)}\bar{Z} - \bar{Z}B_t^{(1)}\right)C_t^{(1)}\,|\bar{0}\rangle\right) \\ &\quad +\mathrm{Im}\left(\langle\bar{0}|\, A_t^{(2)}\left(B_t^{(2)}\bar{Z} - \bar{Z}B_t^{(2)}\right)C_t^{(2)}\,|\bar{0}\rangle\right) \\ &\quad -\frac{i}{2}\langle\bar{0}|\left[A_t^{(3)}\left(B_t^{(3)}\bar{Z} - \bar{Z}B_t^{(3)}\right)C_t^{(3)}\right]|\bar{0}\rangle \end{aligned} \tag{124}$$

$$\begin{aligned} B'_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Re}\left(\langle\bar{0}|\, A_t^{(1)}\left(B_t^{(1)} + \bar{Z}B_t^{(1)}\bar{Z}\right)C_t^{(1)}\,|\bar{0}\rangle\right) \\ &\quad +\mathrm{Re}\left(\langle\bar{0}|\, A_t^{(2)}\left(B_t^{(2)} + \bar{Z}B_t^{(2)}\bar{Z}\right)C_t^{(2)}\,|\bar{0}\rangle\right) \\ &\quad +\frac{1}{2}\langle\bar{0}|\, A_t^{(3)}\left(B_t^{(3)} + \bar{Z}B_t^{(3)}\bar{Z}\right)C_t^{(3)}\,|\bar{0}\rangle. \end{aligned} \tag{125}$$

Given θ and $\vec{x}_{\neg j}$, we first compute the following matrices in a total of $O(L)$ time by standard dynamic programming technique:

- $P_{0,2t}$, $P_{2t+2,4L-2t-2}$, $P_{4L-2t,4L}$, $P_{2t+2,2L-1}$, $P_{2L+1,4L-2t-2}$;
- $P_{4L-2t,4L-2k-1}$ and $P_{4L-2k+1,4L}$ for $k = 0, 1, \ldots, t$;
- $P_{2t+2,4L-2k-1}$ and $P_{4L-2k+1,4L-2t-2}$ for $k = t+1, t+2, \ldots, L-1$.

Then we compute $A_t^{(i)}$, $B_t^{(i)}$ and $C_t^{(i)}$ for $i = 1, 2, 3$ by Eqs. (109-112), (113-116) and (119-121). After that, we calculate $C'_j(\theta;\vec{x}_{\neg j})$, $S'_j(\theta;\vec{x}_{\neg j})$ and $B'_j(\theta;\vec{x}_{\neg j})$ by Eqs. (123-125). Overall, this procedure takes $O(L)$ time.

- Case 2: $j = 2t+1$ is odd, where $0 \le t \le L-1$. In this case, $W_{2t} = U(\theta;-x_j)$, and $W_{4L-2t} = U(\theta;x_j)$. They by Eqs. (22), (87) and (96), we get

$$\Delta(\theta;\vec{x}) = \langle\bar{0}|\, P_{0,2t-1}U(\theta;-x_j)P_{2t+1,4L-2t-1}U(\theta;x_j)P_{4L-2t+1,4L}\,|\bar{0}\rangle \tag{126}$$
$$= C_j(\theta;\vec{x}_{\neg j})\cos(2x_j) + S_j(\theta;\vec{x}_{\neg j})\sin(2x_j) + B_j(\theta;\vec{x}_{\neg j}), \tag{127}$$

where

$$C_j(\theta;\vec{x}_{\neg j}) = \frac{1}{2}\langle\bar{0}|\, P_{0,2t-1}\left(P_{2t+1,4L-2t-2} - P(\theta)P_{2t+1,4L-2t-1}P(\theta)\right)P_{4L-2t+1,4L}\,|\bar{0}\rangle, \tag{128}$$

$$S_j(\theta;\vec{x}_{\neg j}) = -\frac{i}{2}\langle\bar{0}|\, P_{0,2t-1}\left(P_{2t+1,4L-2t-1}P(\theta) - P(\theta)P_{2t+1,4L-2t-1}\right)P_{4L-2t+1,4L}\,|\bar{0}\rangle, \tag{129}$$

$$B_j(\theta;\vec{x}_{\neg j}) = \frac{1}{2}\langle\bar{0}|\, P_{0,2t-1}\left(P_{2t+1,4L-2t-1} + P(\theta)P_{2t+1,4L-2t-1}P(\theta)\right)P_{4L-2t+1,4L}\,|\bar{0}\rangle. \tag{130}$$

Given θ and $\vec{x}_{\neg j}$, we first compute $P_{0,2t-1}$, $P_{2t+1,4L-2t-1}$ and $P_{4L-2t+1,4L}$ in $O(L)$ time. Then we calculate $C_j(\theta;\vec{x}_{\neg j})$, $S_j(\theta;\vec{x}_{\neg j})$ and $B_j(\theta;\vec{x}_{\neg j})$ by Eqs. (128-130). This procedure takes only $O(L)$ time.

Next, we describe how to compute $C'_j(\theta;\vec{x}_{\neg j})$, $S'_j(\theta;\vec{x}_{\neg j})$ and $B'_j(\theta;\vec{x}_{\neg j})$. Using Eq. (92) and the fact $P_{a,b} = P_{a,4L-2t-1}W_{4L-2t}P_{4L-2t+1,b}$ for any $a \leq 4L-2t \leq b$, we obtain

$$\begin{aligned}
Q'(\theta;\vec{x}) &= P_{2L+1,2L+1}W'_{2L+2}P_{2L+3,4L-2t-1}W_{4L-2t}P_{4L-2t+1,4L} \\
&\quad +P_{2L+1,2L+3}W'_{2L+4}P_{2L+5,4L-2t-1}W_{4L-2t}P_{4L-2t+1,4L} \\
&\quad +\ldots \\
&\quad +P_{2L+1,4L-2t-3}W'_{4L-2t-2}P_{4L-2t-1,4L-2t-1}W_{4L-2t}P_{4L-2t+1,4L} \\
&\quad +P_{2L+1,4L-2t-1}W'_{4L-2t}P_{4L-2t+1,4L} \\
&\quad +P_{2L+1,4L-2t-1}W_{4L-2t}P_{4L-2t+1,4L-2t+1}W'_{4L-2t+2}P_{4L-2t+3,4L} \\
&\quad +\ldots \\
&\quad +P_{2L+1,4L-2t-1}W_{4L-2t}P_{4L-2t+1,4L-3}W'_{4L-2}P_{4L-1,4L} \\
&\quad +P_{2L+1,4L-2t-1}W_{4L-2t}P_{4L-2t+1,4L-1}W'_{4L}. \qquad (131)
\end{aligned}$$

Then it follows from Eqs. (85) and (106) that

$$\begin{aligned}
Q^\dagger(\theta;\vec{x})P(\theta)Q'(\theta;\vec{x}) &= A_t^{(1)}W_{2t}B_t^{(1)}W_{4L-2t}C_t^{(1)} \\
&\quad +A_t^{(2)}W_{2t}B_t^{(2)}W'_{4L-2t}C_t^{(2)} \\
&\quad +A_t^{(3)}W_{2t}B_t^{(3)}W_{4L-2t}C_t^{(3)}, \qquad (132) \\
&= A_t^{(1)}U(\theta;-x_j)B_t^{(1)}U(\theta;x_j)C_t^{(1)} \\
&\quad +A_t^{(2)}U(\theta;-x_j)B_t^{(2)}U'(\theta;x_j)C_t^{(2)} \\
&\quad +A_t^{(3)}U(\theta;-x_j)B_t^{(3)}U(\theta;x_j)C_t^{(3)}, \qquad (133)
\end{aligned}$$

where

$$\begin{aligned}
A_t^{(1)} &= P_{0,2t-1}, \qquad (134) \\
B_t^{(1)} &= P_{2t+1,4L-2t-1}, \qquad (135) \\
C_t^{(1)} &= \sum_{k=0}^{t-1} P_{4L-2t+1,4L-2k-1}W'_{4L-2k}P_{4L-2k+1,4L} \qquad (136) \\
&= \sum_{k=0}^{t-1} P_{4L-2t+1,4L-2k-1}U'(\theta;x_{2k+1})P_{4L-2k+1,4L}, \qquad (137) \\
A_t^{(2)} &= P_{0,2t-1}, \qquad (138) \\
B_t^{(2)} &= P_{2t+1,4L-2t-1} \qquad (139) \\
C_t^{(2)} &= P_{4L-2t+1,4L}, \qquad (140) \\
A_t^{(3)} &= P_{0,2t-1}, \qquad (141) \\
B_t^{(3)} &= \sum_{k=t+1}^{L-1} P_{2t+1,4L-2k-1}W'_{4L-2k}P_{4L-2k+1,4L-2t-1} \qquad (142) \\
&= \sum_{k=t+1}^{L-1} P_{2t+1,4L-2k-1}U'(\theta;x_{2k+1})P_{4L-2k+1,4L-2t-1}, \qquad (143) \\
C_t^{(3)} &= P_{4L-2t+1,4L}. \qquad (144)
\end{aligned}$$

Meanwhile, we have

$$\begin{aligned}
Q^\dagger(\theta;\vec{x})P'(\theta)Q(\theta;\vec{x}) &= A_t^{(4)}W_{2t}B_t^{(4)}W_{4L-2t}C_t^{(4)} \qquad (145) \\
&= A_t^{(4)}U(\theta;-x_j)B_t^{(4)}U(\theta;x_j)C_t^{(4)}, \qquad (146)
\end{aligned}$$

where

$$A_t^{(4)} = P_{0,2t-1}, \tag{147}$$
$$B_t^{(4)} = P_{2t+1,2L-1}P'(\theta)P_{2L+1,4L-2t-1}, \tag{148}$$
$$C_t^{(4)} = P_{4L-2t+1,4L}. \tag{149}$$

Combining the above facts with Eqs. (96), (98) and (100) yields

$$\Delta'(\theta;\vec{x}) = C'_j(\theta;\vec{x}_{\neg j})\cos(2x_j) + S'_j(\theta;\vec{x}_{\neg j})\sin(2x_j) + B'_j(\theta;\vec{x}_{\neg j}), \tag{150}$$

where

$$\begin{aligned} C'_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Re}\left(\langle\bar{0}| A_t^{(1)}\left(B_t^{(1)} - P(\theta)B_t^{(1)}P(\theta)\right)C_t^{(1)}|\bar{0}\rangle\right) \\ &\quad -\mathrm{Re}\left(\langle\bar{0}| A_t^{(2)}P(\theta)B_t^{(2)}P'(\theta)C_t^{(2)}|\bar{0}\rangle\right) \\ &\quad +\mathrm{Re}\left(\langle\bar{0}| A_t^{(3)}\left(B_t^{(3)} - P(\theta)B_t^{(3)}P(\theta)\right)C_t^{(3)}|\bar{0}\rangle\right) \\ &\quad +\frac{1}{2}\langle\bar{0}| A_t^{(4)}\left(B_t^{(4)} - P(\theta)B_t^{(4)}P(\theta)\right)C_t^{(4)}|\bar{0}\rangle, \end{aligned} \tag{151}$$

$$\begin{aligned} S'_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Im}\left(\langle\bar{0}| A_t^{(1)}\left(B_t^{(1)}P(\theta) - P(\theta)B_t^{(1)}\right)C_t^{(1)}|\bar{0}\rangle\right) \\ &\quad +\mathrm{Im}\left(\langle\bar{0}| A_t^{(2)}B_t^{(2)}P'(\theta)C_t^{(2)}|\bar{0}\rangle\right) \\ &\quad +\mathrm{Im}\left(\langle\bar{0}| A_t^{(3)}\left(B_t^{(3)}P(\theta) - P(\theta)B_t^{(3)}\right)C_t^{(3)}|\bar{0}\rangle\right) \\ &\quad -\frac{i}{2}\langle\bar{0}|\left[A_t^{(4)}\left(B_t^{(4)}P(\theta) - P(\theta)B_t^{(4)}\right)C_t^{(4)}\right]|\bar{0}\rangle \end{aligned} \tag{152}$$

$$\begin{aligned} B'_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Re}\left(\langle\bar{0}| A_t^{(1)}\left(B_t^{(1)} + P(\theta)B_t^{(1)}P(\theta)\right)C_t^{(1)}|\bar{0}\rangle\right) \\ &\quad +\mathrm{Re}\left(\langle\bar{0}| A_t^{(2)}P(\theta)B_t^{(2)}P'(\theta)C_t^{(2)}|\bar{0}\rangle\right) \\ &\quad +\mathrm{Re}\left(\langle\bar{0}| A_t^{(3)}\left(B_t^{(3)} + P(\theta)B_t^{(3)}P(\theta)\right)C_t^{(3)}|\bar{0}\rangle\right) \\ &\quad +\frac{1}{2}\langle\bar{0}| A_t^{(4)}\left(B_t^{(4)} + P(\theta)B_t^{(4)}P(\theta)\right)C_t^{(4)}|\bar{0}\rangle. \end{aligned} \tag{153}$$

Given θ and $\vec{x}_{\neg j}$, we first compute the following matrices in a total of $O(L)$ time by standard dynamic programming technique:

- $P_{0,2t-1}$, $P_{2t+1,4L-2t-1}$, $P_{4L-2t+1,4L}$, $P_{2t+1,2L-1}$, $P_{2L+1,4L-2t-1}$;
- $P_{4L-2t+1,4L-2k-1}$ and $P_{4L-2k+1,4L}$ for $k = 0, 1, \ldots, t-1$;
- $P_{2t+1,4L-2k-1}$ and $P_{4L-2k+1,4L-2t-1}$ for $k = t+1, t+2, \ldots, L-1$.

Then we compute $A_t^{(i)}$, $B_t^{(i)}$ and $C_t^{(i)}$ for $i = 1, 2, 3, 4$ by Eqs. (134-137), (138-140), (141-144) and (147-149). After that, we calculate $C'_j(\theta;\vec{x}_{\neg j})$, $S'_j(\theta;\vec{x}_{\neg j})$ and $B'_j(\theta;\vec{x}_{\neg j})$ by Eqs. (151-153). Overall, this procedure takes $O(L)$ time.

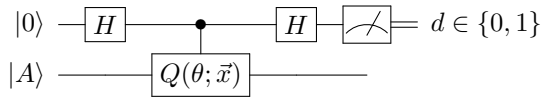
B Ancilla-based scheme

In this appendix, we present an alternative scheme, called the *ancilla-based* scheme. In this scheme, the engineered likelihood function (ELF) is generated by the quantum circuit in Fig. B.1, where $Q(\theta;\vec{x})$ is given by Eq. (20) [11], in which $\vec{x} = (x_1, x_2, \ldots, x_{2L-1}, x_{2L}) \in \mathbb{R}^{2L}$ are tunable parameters.

Assuming the circuit in Fig. B.1 is noiseless, the engineered likelihood function is given by

$$\mathbb{P}(d|\theta;\vec{x}) = \frac{1}{2}\left[1 + (-1)^d\Lambda(\theta;\vec{x})\right], \quad \forall d \in \{0,1\} \tag{154}$$

[11] The ancilla-free and ancilla-based schemes share the same $Q(\theta;\vec{x})$ operator.



where

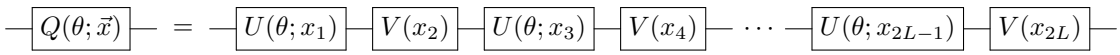


Figure B.1: Quantum circuit for the ancilla-based engineered likelihood functions

where

$$\Lambda(\theta;\vec{x}) = \mathrm{Re}(\langle A| Q(\theta;\vec{x}) |A\rangle) \tag{155}$$

is the bias of the likelihood function. It turns out that most of the argument in Section 3.1 still holds in the ancilla-based case, except that we need to replace $\Delta(\theta;\vec{x})$ with $\Lambda(\theta;\vec{x})$. So we will use the same notation (e.g. $|\bar{0}\rangle$, $|\bar{1}\rangle$, $\bar{X}$, $\bar{Y}$, $\bar{Z}$, $\bar{I}$) as before, unless otherwise stated. In particular, when we take the errors in the circuit in Fig. B.1 into account, the noisy likelihood function is given by

$$\mathbb{P}(d|\theta;f,\vec{x}) = \frac{1}{2}\left[1 + (-1)^d f \Lambda(\theta;\vec{x})\right], \quad \forall d \in \{0,1\} \tag{156}$$

where f is the fidelity of the process for generating the ELF. Note that, however, there does exist a difference between $\Delta(\theta;\vec{x})$ and $\Lambda(\theta;\vec{x})$, as the former is trigono-multiqudaratic in $\vec{x}$, while the latter is trigono-multilinear in $\vec{x}$.

We will tune the circuit angles $\vec{x}$ and perform Bayesian inference with the resultant ELFs in the same way as in Section 3.2. In fact, the argument in Section 3.2 still holds in the ancilla-based case, except that we need to replace $\Delta(\theta;\vec{x})$ with $\Lambda(\theta;\vec{x})$. So we will use the same notation as before, unless otherwise stated. In particular, we also define the variance reduction factor $\mathcal{V}(\mu,\sigma;f,\vec{x})$ as in Eqs. (33) and (34), replacing $\Delta(\theta;\vec{x})$ with $\Lambda(\theta;\vec{x})$. As shown in Appendix C,

$$\mathcal{V}(\mu,\sigma;f,\vec{x}) \approx \mathcal{I}(\mu;f,\vec{x}) = \frac{f^2\left(\Lambda'(\theta;\vec{x})\right)^2}{1 - f^2\left(\Lambda(\theta;\vec{x})\right)^2}, \quad \text{when } \sigma \text{ is small,} \tag{157}$$

and

$$\mathcal{V}(\mu,\sigma;f,\vec{x}) \approx f^2(\Lambda'(\mu;\vec{x}))^2, \quad \text{when both } \sigma \text{ and } f \text{ are small.} \tag{158}$$

Namely, the Fisher information and slope of the likelihood function $\mathbb{P}(d|\theta;f,\vec{x})$ at $\theta = \mu$ are two proxies of the variance reduction factor $\mathcal{V}(\mu,\sigma;f,\vec{x})$ under reasonable assumptions. Since the direct optimization of $\mathcal{V}$ is hard in general, we will tune the parameters $\vec{x}$ by optimizing these proxies instead.

B.1 Efficient maximization of proxies of the variance reduction factor

Now we present efficient heuristic algorithms for maximizing two proxies of the variance reduction factor $\mathcal{V}$ – the Fisher information and slope of the likelihood function $\mathbb{P}(d|\theta;f,\vec{x})$. All of these algorithms make use of the following procedures for evaluating the CSD coefficient functions of the bias $\Lambda(\theta;\vec{x})$ and its derivative $\Lambda'(\theta;\vec{x})$ with respect to x_j for $j = 1,2,\ldots,2L$.

B.1.1 Evaluating the CSD coefficient functions of the bias and its derivative

Since $\Lambda(\theta;\vec{x})$ is trigono-multilinear in $\vec{x}$, for any $j \in \{1,2,...,2L\}$, there exist functions $C_j(\theta;\vec{x}_{\neg j})$ and $S_j(\theta;\vec{x}_{\neg j})$, that are trigono-multilinear in $\vec{x}_{\neg j} = (x_1,\ldots,x_{j-1},x_{j+1},\ldots,x_{2L})$, such that

$$\Lambda(\theta;\vec{x}) = C_j(\theta;\vec{x}_{\neg j})\cos(x_j) + S_j(\theta;\vec{x}_{\neg j})\sin(x_j). \tag{159}$$

It follows that

$$\Lambda'(\theta;\vec{x}) = C'_j(\theta;\vec{x}_{\neg j})\cos(x_j) + S'_j(\theta;\vec{x}_{\neg j})\sin(x_j) \tag{160}$$

is also trigono-multilinear in $\vec{x}$, where $C'_j(\theta;\vec{x}_{\neg j}) = \partial_\theta C_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j}) = \partial_\theta S_j(\theta;\vec{x}_{\neg j})$ are the derivatives of $C_j(\theta;\vec{x}_{\neg j})$ and $S_j(\theta;\vec{x}_{\neg j})$ with respect to θ, respectively.

Our optimization algorithms require efficient procedures for evaluating $C_j(\theta;\vec{x}_{\neg j})$, $S_j(\theta;\vec{x}_{\neg j})$, $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$ for given θ and $\vec{x}_{\neg j}$. It turns out that these tasks can be accomplished in $O(L)$ time.

Lemma 2. *Given θ and $\vec{x}_{\neg j}$, each of $C_j(\theta;\vec{x}_{\neg j})$, $S_j(\theta;\vec{x}_{\neg j})$, $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$ can be computed in $O(L)$ time.*

Proof. For convenience, we introduce the following notation. Let $W_{2i} = V(x_{2L-2i})$, $W_{2i+1} = U(\theta;x_{2L-2i-1})$, for $i = 0,1,\ldots,L-1$. Furthermore, let $W'_j = \partial_\theta W_j$ for $j = 0,1,\ldots,2L-1$. Note that $W'_j = 0$ if j is even. Then we define $P_{a,b} = W_a W_{a+1}\ldots W_b$ if $0 \le a \le b \le 2L-1$, and $P_{a,b} = I$ otherwise.

With this notation, Eqs. (20) and (89) imply that

$$Q(\theta;\vec{x}) = P_{0,a-1} W_a P_{a+1,2L-1}, \quad \forall 0 \le a \le 2L-1, \tag{161}$$

and

$$Q'(\theta;\vec{x}) = P_{0,0}W'_1P_{2,2L-1} + P_{0,2}W'_3P_{4,2L-1} + \ldots P_{0,2L-4}W'_{2L-3}P_{2L-2,2L-1} + P_{0,2L-2}W'_{2L-1}. \tag{162}$$

In order to evaluate $C_j(\theta;\vec{x}_{\neg j})$, $S_j(\theta;\vec{x}_{\neg j})$, $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$ for given θ and $\vec{x}_{\neg j}$, we consider the case j is even and the case j is odd separately.

- Case 1: $j = 2(L-t)$ is even, where $0 \le t \le L-1$. In this case, $W_{2t} = V(x_j)$. Using the fact

$$\begin{aligned} Q(\theta;\vec{x}) &= P_{0,2t-1}W_{2t}P_{2t+1,2L-1} && (163)\\ &= P_{0,2t-1}(\cos(x_j)\,\bar{I} - \mathrm{i}\sin(x_j)\,\bar{Z})P_{2t+1,2L-1} && (164)\\ &= \cos(x_j)\,P_{0,2t-1}P_{2t+1,2L-1} - \mathrm{i}\sin(x_j)\,P_{0,2t-1}\bar{Z}P_{2t+1,2L-1}, && (165)\end{aligned}$$

  we obtain

$$\Lambda(\theta;\vec{x}) = C_j(\theta;\vec{x}_{\neg j})\cos(x_j) + S_j(\theta;\vec{x}_{\neg j})\sin(x_j), \tag{166}$$

  where

$$\begin{aligned} C_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Re}\left(\langle\bar{0}|\,P_{0,2t-1}P_{2t+1,2L-1}\,|\bar{0}\rangle\right), && (167)\\ S_j(\theta;\vec{x}_{\neg j}) &= \mathrm{Im}\left(\langle\bar{0}|\,P_{0,2t-1}\bar{Z}P_{2t+1,2L-1}\,|\bar{0}\rangle\right). && (168)\end{aligned}$$

  Given θ and $\vec{x}_{\neg j}$, we first compute $P_{0,2t-1}$ and $P_{2t+1,2L-1}$ in $O(L)$ time. Then we calculate $C_j(\theta;\vec{x}_{\neg j})$ and $S_j(\theta;\vec{x}_{\neg j})$ by Eqs. (167) and (168). This procedure takes only $O(L)$ time.

  Next, we describe how to compute $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$. Using Eq. (162) and the fact $P_{a,b} = P_{a,2t-1}W_{2t}P_{2t+1,b}$, for any $a \le 2t \le b$, we obtain

$$\begin{aligned} Q'(\theta;\vec{x}) = {} & P_{0,0}W'_1P_{2,2t-1}W_{2t}P_{2t+1,2L-1}\\ & +P_{0,2}W'_3P_{4,2t-1}W_{2t}P_{2t+1,2L-1}\\ & +\ldots\\ & +P_{0,2t-2}W'_{2t-1}W_{2t}P_{2t+1,2L-1}\\ & +P_{0,2t-1}W_{2t}W'_{2t+1}P_{2t+2,2L-1}\\ & +\ldots\\ & +P_{0,2t-1}W_{2t}P_{2t+1,2L-4}W'_{2L-3}P_{2L-2,2L-1}\\ & +P_{0,2t-1}W_{2t}P_{2t+1,2L-2}W'_{2L-1}. \end{aligned} \tag{169}$$

Let

$$A_t = \sum_{s=1}^{t} P_{0,2s-2} W'_{2s-1} P_{2s,2t-1} \tag{170}$$

$$= \sum_{s=1}^{t} P_{0,2s-2} U'(\theta; x_{2L-2s+1}) P_{2s,2t-1}, \tag{171}$$

$$B_t = \sum_{s=t+1}^{L} P_{2t+1,2s-2} W'_{2s-1} P_{2s,2L-1} \tag{172}$$

$$= \sum_{s=t+1}^{L} P_{2t+1,2s-2} U'(\theta; x_{2L-2s+1}) P_{2s,2L-1}. \tag{173}$$

Then Eq. (169) yields

$$Q'(\theta;\vec{x}) = A_t W_{2t} P_{2t+1,2L-1} + P_{0,2t-1} W_{2t} B_t \tag{174}$$

$$= A_t \left(\cos(x_j)\,\bar{I} - \mathrm{i}\sin(x_j)\,\bar{Z}\right) P_{2t+1,2L-1} + P_{0,2t-1} \left(\cos(x_j))\bar{I} - \mathrm{i}\sin(x_j)\,\bar{Z}\right) B_t \tag{175}$$

$$= \cos(x_j)\left(A_t P_{2t+1,2L-1} + P_{0,2t-1} B_t\right) - \mathrm{i}\sin(x_j)\left(A_t \bar{Z} P_{2t+1,2L-1} + P_{0,2t-1} \bar{Z} B_t\right), \tag{176}$$

which leads to

$$\Lambda'(\theta;\vec{x}) = C'_j(\theta;\vec{x}_{\neg j})\cos(x_j) + S'_j(\theta;\vec{x}_{\neg j})\sin(x_j), \tag{177}$$

where

$$C'_j(\theta;\vec{x}_{\neg j}) = \mathrm{Re}\left(\langle\bar{0}|\left(A_t P_{2t+1,2L-1} + P_{0,2t-1} B_t\right)|\bar{0}\rangle\right), \tag{178}$$

$$S'_j(\theta;\vec{x}_{\neg j}) = \mathrm{Im}\left(\langle\bar{0}|\left(A_t \bar{Z} P_{2t+1,2L-1} + P_{0,2t-1} \bar{Z} B_t\right)|\bar{0}\rangle\right). \tag{179}$$

Given θ and $\vec{x}_{\neg j}$, we first compute the following matrices in a total of $O(L)$ time by standard dynamic programming techniques:

- $P_{0,2s-2}$ and $P_{2s,2t-1}$ for $s = 1, 2, \ldots, t$;
- $P_{2t+1,2s-2}$ and $P_{2s,2L-1}$ for $s = t+1, t+2, \ldots, L$;
- $P_{0,2t-1}$ and $P_{2t+1,2L-1}$.

Then we compute A_t and B_t by Eqs. (171) and (173). After that, we calculate $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$ by Eqs. (178) and (179). Overall, this procedure takes $O(L)$ time.

- Case 2: $j = 2(L-t)-1$ is odd, where $0 \le t \le L-1$. In this case, $W_{2t+1} = U(\theta; x_j)$. Using the fact

$$Q(\theta;\vec{x}) = P_{0,2t} W_{2t+1} P_{2t+2,2L-1} \tag{180}$$

$$= P_{0,2t}\left(\cos(x_j)\,\bar{I} - \mathrm{i}\sin(x_j)\,P(\theta)\right) P_{2t+2,2L-1} \tag{181}$$

$$= \cos(x_j)\,P_{0,2t} P_{2t+2,2L-1} - \mathrm{i}\sin(x_j)\,P_{0,2t} P(\theta) P_{2t+2,2L-1}, \tag{182}$$

we obtain

$$\Lambda(\theta;\vec{x}) = C_j(\theta;\vec{x}_{\neg j})\cos(x_j) + S_j(\theta;\vec{x}_{\neg j})\sin(x_j), \tag{183}$$

where

$$C_j(\theta;\vec{x}_{\neg j}) = \mathrm{Re}\left(\langle\bar{0}|\,P_{0,2t} P_{2t+2,2L-1}\,|\bar{0}\rangle\right), \tag{184}$$

$$S_j(\theta;\vec{x}_{\neg j}) = \mathrm{Im}\left(\langle\bar{0}|\,P_{0,2t} P(\theta) P_{2t+2,2L-1}\,|\bar{0}\rangle\right). \tag{185}$$

Given θ and $\vec{x}_{\neg j}$, we first compute $P_{0,2t}$ and $P_{2t+2,2L-1}$ in $O(L)$ time. Then we calculate $C_j(\theta;\vec{x}_{\neg j})$ and $S_j(\theta;\vec{x}_{\neg j})$ by Eqs. (184) and (185). This procedure takes only $O(L)$ time.

Next, we describe how to compute $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$. Using Eq. (162) and the fact $P_{a,b} = P_{a,2t}W_{2t+1}P_{2t+2,b}$ for any $a \le 2t+1 \le b$, we get

$$\begin{aligned} Q'(\theta;\vec{x}) &= P_{0,0}W'_1P_{2,2t}W_{2t+1}P_{2t+2,2L-1} \\ &\quad +P_{0,2}W'_3P_{4,2t}W_{2t+1}P_{2t+2,2L-1} \\ &\quad +\dots \\ &\quad +P_{0,2t-2}W'_{2t-1}P_{2t,2t}W_{2t+1}P_{2t+2,2L-1} \\ &\quad +P_{0,2t}W'_{2t+1}P_{2t+2,2L-1} \\ &\quad +P_{0,2t}W_{2t+1}P_{2t+2,2t+2}W'_{2t+3}P_{2t+4,2L-1} \\ &\quad +\dots \\ &\quad +P_{0,2t}W_{2t+1}P_{2t+2,2L-4}W'_{2L-3}P_{2L-2,2L-1} \\ &\quad +P_{0,2t}W_{2t+1}P_{2t+2,2L-2}W'_{2L-1}. \end{aligned} \tag{186}$$

Let

$$A_t = \sum_{s=1}^{t} P_{0,2s-2}W'_{2s-1}P_{2s,2t} \tag{187}$$

$$= \sum_{s=1}^{t} P_{0,2s-2}U'(\theta;x_{2L-2s+1})P_{2s,2t}, \tag{188}$$

$$B_t = \sum_{s=t+2}^{L} P_{2t+2,2s-2}W'_{2s-1}P_{2s,2L-1} \tag{189}$$

$$= \sum_{s=t+2}^{L} P_{2t+2,2s-2}U'(\theta;x_{2L-2s+1})P_{2s,2L-1}. \tag{190}$$

Then Eq. (186) yields

$$Q'(\theta;\vec{x}) = A_tW_{2t+1}P_{2t+2,2L-1} + P_{0,2t}W'_{2t+1}P_{2t+2,2L-1} + P_{0,2t}W_{2t+1}B_t \tag{191}$$

$$\begin{aligned} &= A_t(\cos(x_j)\,\bar{I} - \mathrm{i}\sin(x_j)\,P(\theta))P_{2t+2,2L-1} \\ &\quad -\mathrm{i}\sin(x_j)\,P_{0,2t}P'(\theta)P_{2t+2,2L-1} \\ &\quad +P_{0,2t}(\cos(x_j)\,\bar{I} - \mathrm{i}\sin(x_j)\,P(\theta))B_t \end{aligned} \tag{192}$$

$$\begin{aligned} &= \cos(x_j)\,(A_tP_{2t+2,2L-1} + P_{0,2t}B_t) \\ &\quad -\mathrm{i}\sin(x_j)\,(A_tP(\theta)P_{2t+2,2L-1} + P_{0,2t}P'(\theta)P_{2t+2,2L-1} + P_{0,2t}P(\theta)B_t)\,, \end{aligned} \tag{193}$$

which leads to

$$\Lambda'(\theta;\vec{x}) = C'_j(\theta;\vec{x}_{\neg j})\cos(x_j) + S'_j(\theta;\vec{x}_{\neg j})\sin(x_j)\,, \tag{194}$$

where

$$C'_j(\theta;\vec{x}_{\neg j}) = \mathrm{Re}\left(\langle\bar{0}|\,(A_tP_{2t+2,2L-1} + P_{0,2t}B_t)\,|\bar{0}\rangle\right), \tag{195}$$

$$S'_j(\theta;\vec{x}_{\neg j}) = \mathrm{Im}\left(\langle\bar{0}|\,(A_tP(\theta)P_{2t+2,2L-1} + P_{0,2t}P'(\theta)P_{2t+2,2L-1} + P_{0,2t}P(\theta)B_t)\,|\bar{0}\rangle\right). \tag{196}$$

Given θ and $\vec{x}_{\neg j}$, we first compute the following matrices in a total of $O(L)$ time by standard dynamic programming techniques:

- $P_{0,2s-2}$ and $P_{2s,2t}$ for $s = 1, 2, \dots, t$;
- $P_{2t+2,2s-2}$ and $P_{2s,2L-1}$ for $s = t+2, t+3, \dots, L$;

– $P_{0,2t}$ and $P_{2t+2,2L-1}$.

Then we compute A_t and B_t by Eqs. (188) and (190). After that, we calculate $C'_j(\theta;\vec{x}_{\neg j})$ and $S'_j(\theta;\vec{x}_{\neg j})$ by Eqs. (195) and (196). Overall, this procedure takes $O(L)$ time.

□

B.1.2 Maximizing the Fisher information of the likelihood function

We propose two algorithms for maximizing the Fisher information of the likelihood function $\mathbb{P}(d|\theta;f,\vec{x})$ at a given point $\theta=\mu$ (i.e. the prior mean of θ). Namely, our goal is to find $\vec{x}\in\mathbb{R}^{2L}$ that maximize

$$\mathcal{I}(\theta;f,\vec{x})=\frac{f^2\left(\Lambda'(\theta;\vec{x})\right)^2}{1-f^2\left(\Lambda(\theta;\vec{x})\right)^2}. \tag{197}$$

These algorithms are similar to Algorithms 1 and 2 for Fisher information maximization in the ancilla-free case, in the sense that they are also based on gradient ascent and coordinate ascent, respectively. The main difference is that now we invoke the procedures in Lemma 2 to evaluate $C(\mu;\vec{x}_{\neg j})$, $S(\mu;\vec{x}_{\neg j})$, $C'(\mu;\vec{x}_{\neg j})$ and $S'(\mu;\vec{x}_{\neg j})$ for given μ and $\vec{x}_{\neg j}$, and then use them to either compute the partial derivative of $\mathcal{I}(\mu;f,\vec{x})$ with respect to x_j (in gradient ascent) or define a single-variable optimization problem for x_j (in coordinate ascent). These algorithms are formally described in Algorithms 5 and 6.

B.1.3 Maximizing the slope of the likelihood function

We also propose two algorithms for maximizing the slope of the likelihood function $\mathbb{P}(d|\theta;f,\vec{x})$ at a given point $\theta=\mu$ (i.e. the prior mean of θ). Namely, our goal is to find $\vec{x}\in\mathbb{R}^{2L}$ that maximize $|\mathbb{P}'(d|\theta;f,\vec{x})|=f|\Lambda'(\theta;\vec{x})|/2$.

These algorithms are similar to Algorithms 3 and 4 for slope maximization in the ancilla-free case, in the sense that they are also based on gradient ascent and coordinate ascent, respectively. The main difference is that now we invoke the procedures in Lemma 2 to evaluate $C'(\mu;\vec{x}_{\neg j})$ and $S'(\mu;\vec{x}_{\neg j})$ for given μ and $\vec{x}_{\neg j}$. Then we use these quantities to either compute the partial derivative of $\left(\Lambda(\mu;\vec{x})\right)^2$ with respect to x_j (in gradient ascent) or directly update the value of x_j (in coordinate ascent). These algorithms are formally described in Algorithms 7 and 8.

B.2 Approximate Bayesian inference with engineered likelihood functions

With the algorithms for tuning the circuit parameters $\vec{x}$ in place, we now briefly describe how to perform Bayesian inference efficiently with the resultant likelihood functions. The idea is similar to the one in Section 4.2 for the ancilla-free scheme.

Suppose θ has prior distribution $\mathcal{N}(\mu,\sigma^2)$, where $\sigma\ll 1/L$, and the fidelity of the process for generating the ELF is f. We find that the parameters $\vec{x}=(x_1,x_2,\ldots,x_{2L})$ that maximize $\mathcal{I}(\mu;f,\vec{x})$ (or $|\Lambda'(\mu;\vec{x})|$) satisfy the following property: When θ is close to μ, i.e. $\theta\in[\mu-O(\sigma),\mu+O(\sigma)]$, we have

$$\mathbb{P}(d|\theta;f,\vec{x})\approx\frac{1+(-1)^d f\sin\left(r\theta+b\right)}{2} \tag{205}$$

for some $r,b\in\mathbb{R}$. We find the best-fitting r and b by solving the following least squares problem:

$$(r^*,b^*)=\arg\min_{r,b}\sum_{\theta\in\Theta}\left|\arcsin\left(\Lambda(\theta;\vec{x})\right)-r\theta-b\right|^2, \tag{206}$$

where $\Theta=\{\theta_1,\theta_2,\ldots,\theta_k\}\subseteq[\mu-O(\sigma),\mu+O(\sigma)]$. This least-squares problem has the following analytical solution:

$$\begin{pmatrix}r^*\\b^*\end{pmatrix}=A^+z=(A^TA)^{-1}A^Tz, \tag{207}$$

Algorithm 5: Gradient ascent for Fisher information maximization in the ancilla-based case

Input: The prior mean μ of θ, the number L of circuit layers, the fidelity f of the process for generating the ELF, the step size schedule $\delta : \mathbb{Z}^{\geq 0} \to \mathbb{R}^+$, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in \mathbb{R}^{2L}$ that are a local maximum point of the function $\mathcal{I}(\mu; f, \vec{x})$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 0$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t)}, \ldots, x_{2L}^{(t)})$;
    Compute $C_j^{(t)} := C_j(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j^{(t)} := S_j(\mu; \vec{x}_{\neg j}^{(t)})$, $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$ and $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 2;
    Compute $\Lambda(\mu; \vec{x})$, $\Lambda'(\mu; \vec{x})$ and their partial derivatives with respect to x_j at $\vec{x} = \vec{x}^{(t)}$ as follows:

$$\Lambda^{(t)} := \Lambda(\mu; \vec{x}^{(t)}) = C_j^{(t)} \cos(x_j) + S_j^{(t)} \sin(x_j), \tag{198}$$

$$\Lambda'^{(t)} := \Lambda'(\mu; \vec{x}^{(t)}) = C_j'^{(t)} \cos(x_j) + S_j'^{(t)} \sin(x_j), \tag{199}$$

$$\chi_j^{(t)} := \frac{\partial \Lambda(\mu; \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = -C_j^{(t)} \sin(x_j) + S_j^{(t)} \cos(x_j), \tag{200}$$

$$\chi_j'^{(t)} := \frac{\partial \Lambda'(\mu; \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = -C_j'^{(t)} \sin(x_j) + S_j'^{(t)} \cos(x_j); \tag{201}$$

    Compute the partial derivative of $\mathcal{I}(\mu; f, \vec{x})$ with respect to x_j at $\vec{x} = \vec{x}^{(t)}$ as follows:

$$\gamma_j^{(t)} := \frac{\partial \mathcal{I}(\mu; f, \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = \frac{2f^2 \left[(1 - f^2(\Lambda^{(t)})^2)\Lambda'^{(t)}\chi_j'^{(t)} + f^2\Lambda^{(t)}\chi_j^{(t)}(\Lambda'^{(t)})^2\right]}{\left[1 - f^2(\Lambda^{(t)})^2\right]^2} \tag{202}$$

  end
  Set $\vec{x}^{(t+1)} = \vec{x}^{(t)} + \delta(t)\nabla\mathcal{I}(\mu; f, \vec{x})|_{\vec{x}=\vec{x}^{(t)}}$, where $\nabla\mathcal{I}(\mu; f, \vec{x})|_{\vec{x}=\vec{x}^{(t)}} = (\gamma_1^{(t)}, \gamma_2^{(t)}, \ldots, \gamma_{2L}^{(t)})$;
  if $|\mathcal{I}(\mu; f, \vec{x}^{(t+1)}) - \mathcal{I}(\mu; f, \vec{x}^{(t)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t+1)} = (x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{2L}^{(t+1)})$ as the optimal parameters.

Algorithm 6: Coordinate ascent for Fisher information maximization in the ancilla-based case

Input: The prior mean μ of θ, the number L of circuit layers, the fidelity f of the process for generating the ELF, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in (-\pi, \pi]^{2L}$ that are a local maximum point of the function $\mathcal{I}(\mu; f, \vec{x})$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 1$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t-1)}, \ldots, x_{2L}^{(t-1)})$;
    Compute $C_j^{(t)} := C_j(\mu; \vec{x}_{\neg j}^{(t)})$, $S_j^{(t)} := S_j(\mu; \vec{x}_{\neg j}^{(t)})$, $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$ and $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 2;
    Solve the single-variable optimization problem

$$\arg\max_z \frac{f^2 \left(C_j'^{(t)} \cos(z) + S_j'^{(t)} \sin(z) \right)^2}{1 - f^2 \left(C_j^{(t)} \cos(z) + S_j^{(t)} \sin(z) \right)^2}$$

    by standard gradient-based methods and set $x_j^{(t)}$ to be its solution;
  end
  if $|\mathcal{I}(\mu; f, \vec{x}^{(t)}) - \mathcal{I}(\mu; f, \vec{x}^{(t-1)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t)} = (x_1^{(t)}, x_2^{(t)}, \ldots, x_{2L}^{(t)})$ as the optimal parameters.

Algorithm 7: Gradient ascent for slope maximization in the ancilla-based case

Input: The prior mean μ of θ, the number L of circuit layers, the step size schedule $\delta : \mathbb{Z}^{\geq 0} \to \mathbb{R}^+$, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in \mathbb{R}^{2L}$ that are a local maximum point of the function $|\Lambda'(\mu; \vec{x})|$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 0$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t)}, \ldots, x_{2L}^{(t)})$;
    Compute $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$ and $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 2;
    Compute $\Lambda'(\mu; \vec{x})$ at $\vec{x} = \vec{x}^{(t)}$ as follows:

$$\Lambda'^{(t)} := \Lambda'(\mu; \vec{x}^{(t)}) = C_j'^{(t)} \cos\left(x_j^{(t)}\right) + S_j'^{(t)} \sin\left(x_j^{(t)}\right); \tag{203}$$

    Compute the partial derivative of $\Lambda'(\mu; \vec{x})$ with respect to x_j as follows:

$$\gamma_j^{(t)} := \frac{\partial \Lambda'(\mu; \vec{x})}{\partial x_j}|_{\vec{x}=\vec{x}^{(t)}} = -C_j'^{(t)} \sin\left(x_j^{(t)}\right) + S_j'^{(t)} \cos\left(x_j^{(t)}\right); \tag{204}$$

  end
  Set $\vec{x}^{(t+1)} = \vec{x}^{(t)} + \delta(t)\nabla^{(t)}$, where $\nabla^{(t)} := (2\Lambda'^{(t)}\gamma_1^{(t)}, 2\Lambda'^{(t)}\gamma_2^{(t)}, \ldots, 2\Lambda'^{(t)}\gamma_{2L}^{(t)})$ is the gradient of $(\Lambda'(\mu; \vec{x}))^2$ at $\vec{x} = \vec{x}^{(t)}$;
  if $|\Lambda'(\mu; \vec{x}^{(t+1)}) - \Lambda'(\mu; \vec{x}^{(t)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t+1)} = (x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{2L}^{(t+1)})$ as the optimal parameters.

Algorithm 8: Coordinate ascent for slope optimization in the ancilla-based case

Input: The prior mean μ of θ, the number L of circuit layers, the error tolerance ϵ for termination.

Output: A set of parameters $\vec{x} = (x_1, x_2, \ldots, x_{2L}) \in (-\pi, \pi]^{2L}$ that are a local maximum point of the function $|\Lambda'(\mu; \vec{x})|$.

Choose random initial point $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_{2L}^{(0)}) \in (-\pi, \pi]^{2L}$;
$t \leftarrow 1$;
while *True* **do**
  for $j \leftarrow 1$ **to** $2L$ **do**
    Let $\vec{x}_{\neg j}^{(t)} = (x_1^{(t)}, \ldots, x_{j-1}^{(t)}, x_{j+1}^{(t-1)}, \ldots, x_{2L}^{(t-1)})$;
    Compute $C_j'^{(t)} := C_j'(\mu; \vec{x}_{\neg j}^{(t)})$ and $S_j'^{(t)} := S_j'(\mu; \vec{x}_{\neg j}^{(t)})$ by using the procedures in Lemma 2;
    Set $x_j^{(t)} = \mathrm{Arg}\left(C_j'^{(t)} + iS_j'^{(t)}\right)$;
  end
  if $|\Lambda'(\mu; \vec{x}^{(t)}) - \Lambda'(\mu; \vec{x}^{(t-1)})| < \epsilon$ **then**
    break;
  end
  $t \leftarrow t + 1$;
end
Return $\vec{x}^{(t)} = (x_1^{(t)}, x_2^{(t)}, \ldots, x_{2L}^{(t)})$ as the optimal parameters.

where

$$A = \begin{pmatrix} \theta_1 & 1 \\ \theta_2 & 1 \\ \vdots & \vdots \\ \theta_k & 1 \end{pmatrix}, \quad z = \begin{pmatrix} \arcsin\left(\Lambda(\theta_1; \vec{x})\right) \\ \arcsin\left(\Lambda(\theta_2; \vec{x})\right) \\ \vdots \\ \arcsin\left(\Lambda(\theta_k; \vec{x})\right) \end{pmatrix}. \tag{208}$$

Figure B.2 illustrates an example of the true and fitted likelihood functions.

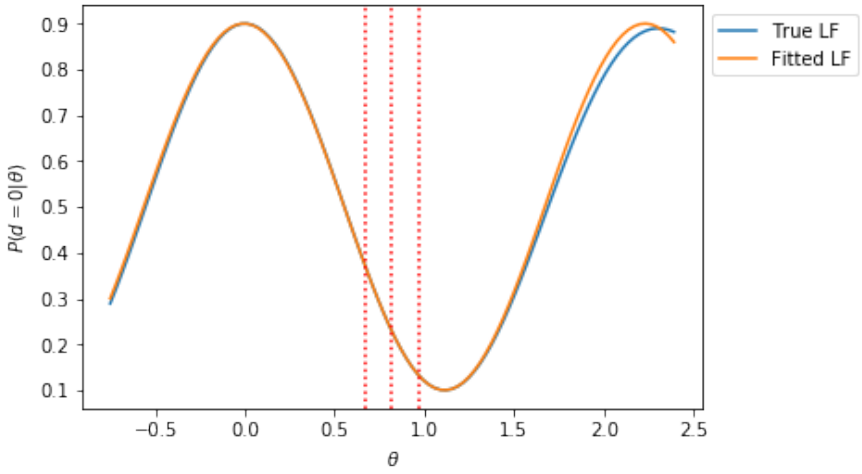


Figure B.2: The true and fitted likelihood functions when $L = 3$, $f = 0.8$, and θ has prior distribution $\mathcal{N}(0.82, 0.0009)$. The true likelihood function is generated by Algorithm 6. During the sinusoidal fitting of this function, we set $\Theta = \{\mu - \sigma, \mu - 0.8\sigma, \ldots, \mu + 0.8\sigma, \mu + \sigma\}$ (i.e. Θ contains 11 uniformly distributed points in $[\mu - \sigma, \mu + \sigma]$) in Eq. (206). The fitted likelihood function is $\mathbb{P}(d|\theta) = (1 + (-1)^d f \sin(r\theta + b))/2$, where $r = -2.81081$ and $b = 1.55477$. Note that the true and fitted likelihood functions are close for $\theta \in [0.67, 0.97]$.

Once we obtain the optimal r and b, we approximate the posterior mean and variance of θ with the ones for

$$\mathbb{P}(d|\theta; f) = \frac{1 + (-1)^d f \sin(r\theta + b)}{2}, \tag{209}$$

which have analytical formulas. Specifically, suppose θ has prior distribution $\mathcal{N}(\mu_k, \sigma_k^2)$ at round k. Let d_k be the measurement outcome and (r_k, b_k) be the best-fitting parameters at this round. Then we approximate the posterior mean and variance of θ by

$$\mu_{k+1} = \mu_k + \frac{(-1)^{d_k} f e^{-r_k^2\sigma_k^2/2} r_k \sigma_k^2 \cos(r_k\mu_k + b_k)}{1 + (-1)^{d_k} f e^{-r_k^2\sigma_k^2/2} \sin(r_k\mu_k + b_k)}, \tag{210}$$

$$\sigma_{k+1}^2 = \sigma_k^2 \left(1 - \frac{f r_k^2 \sigma_k^2 e^{-r_k^2\sigma_k^2/2} [f e^{-r_k^2\sigma_k^2/2} + (-1)^{d_k} \sin(r_k\mu_k + b_k)]}{[1 + (-1)^{d_k} f e^{-r_k^2\sigma_k^2/2} \sin(r_k\mu_k + b_k)]^2}\right). \tag{211}$$

After that, we proceed to the next round, setting $\mathcal{N}(\mu_{k+1}, \sigma_{k+1}^2)$ as the prior distribution of θ for that round. The approximation errors incurred by Eqs. (210) and (211) are small and have negligible impact on the performance of the whole algorithm for the same reason as in the ancilla-free case.

C Comparison of exact optimization with optimization of proxies

In this appendix, we compare the maximization of the variance reduction factor with the maximization of the proxies used in Section 4. We start with motivating the use of these proxies by studying the limiting behavior of $V(\mu, \sigma; \vec{x})$ as $\sigma \to 0$ or as $f \to 0$ or 1.

C.1 Limiting behavior of the variance reduction factor

Consider the following three limiting situations: (i) when the variance vanishes, i.e. $\sigma^2 \to 0$, (ii) when the fidelity is close to zero, i.e. $f \approx 0$, and (iii) when the fidelity is equal to one, i.e. $f = 1$. We will derive expressions for the variance reduction factor in these conditions (see Figure C.1 for a flowchart summarizing the results below). For simplicity, we will assume in this section that either (i) $0 \leq f < 1$ or (ii) $f = 1$ and $|\Delta(\mu; \vec{x})| \neq 1$. While the results here are stated in terms of the ancilla-free bias (22), they also hold for the ancilla-based bias (155) (just replace Δ with Λ).

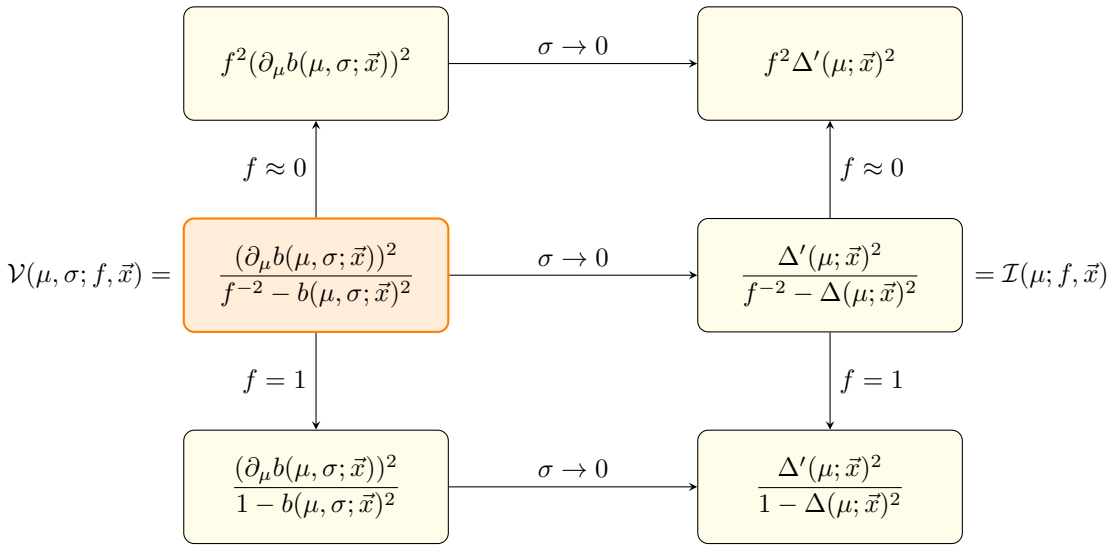


Figure C.1: Flowchart showing the behavior of the expected posterior variance $\mathcal{V}(\mu, \sigma; f, \vec{x})$ (in orange box) as (i) $\sigma \to 0$, (ii) $f \approx 0$, (iii) $f = 1$. For ease of notation we use $f = \bar{p}p^L$, where p is the depolarizing noise parameter in each layer and $\bar{p}$ captures the depolarizing parameter in the readout error.

- In Case (i), as $\sigma \to 0$, the expected bias (33) behaves as

$$\begin{aligned} b(\mu, \sigma; \vec{x}) &\to \Delta(\mu; \vec{x}), \\ \partial_\mu b(\mu, \sigma; \vec{x}) &\to \Delta'(\mu; \vec{x}), \end{aligned} \tag{212}$$

  which implies that the limit of the variance reduction factor as $\sigma \to 0$ is

$$\mathcal{V}_0(\mu; f, \vec{x}) := \lim_{\sigma \to 0} \mathcal{V}(\mu, \sigma; f, \vec{x}) = \mathcal{I}(\mu; f, \vec{x}) \tag{213}$$

  where

$$\mathcal{I}(\mu; f, \vec{x}) := \frac{f^2 \Delta'(\mu; \vec{x})^2}{1 - f^2 \Delta(\mu; \vec{x})^2} \tag{214}$$

  is the Fisher information corresponding to the noisy version of the engineered likelihood function given by Eq. (21).

- In Case (ii), we get

$$\mathcal{V}(\mu, \sigma; f, \vec{x}) \approx f^2 (\partial_\mu b(\mu, \sigma; \vec{x}))^2, \quad \text{for } f \approx 0. \tag{215}$$

- In Case (iii), we get

$$\mathcal{V}(\mu, \sigma; f, \vec{x}) = \frac{(\partial_\mu b(\mu, \sigma; \vec{x}))^2}{1 - b(\mu, \sigma; \vec{x})^2}, \quad \text{for } f = 1. \tag{216}$$

- Combining Cases (i) and (ii), we get

$$\mathcal{V}(\mu, \sigma; f, \vec{x}) \approx f^2 \Delta'(\mu; \vec{x})^2, \quad \text{for } f \approx 0,\ \sigma \approx 0. \tag{217}$$

- Combining Cases (i) and (iii), we get

$$\lim_{\sigma \to 0} \mathcal{V}(\mu, \sigma; f, \vec{x}) = \frac{\Delta'(\mu; \vec{x})^2}{1 - \Delta(\mu; \vec{x})^2} = \mathcal{I}(\mu; 1, \vec{x}), \quad \text{for } f = 1. \tag{218}$$

In the next part, we will show how these approximations give us good proxies for maximizing the variance reduction factor.

C.2 Implementing the exact variance reduction factor optimization and comparison with proxies

Consider the following optimization problems:

$$\begin{aligned} &\texttt{Input:} && (\mu, f), \text{ where } \mu \in \mathbb{R}, f \in [0, 1] \\ &\texttt{Output:} && \operatorname*{arg\,max}_{\vec{x} \in (-\pi, \pi]^{2L}} \mathcal{I}(\mu; f, \vec{x}) = \operatorname*{arg\,max}_{\vec{x} \in (-\pi, \pi]^{2L}} \frac{\Delta'(\mu; \vec{x})^2}{f^{-2} - \Delta(\mu; \vec{x})^2}. \end{aligned} \tag{219}$$

and

$$\begin{aligned} &\texttt{Input:} && \mu \in \mathbb{R} \\ &\texttt{Output:} && \operatorname*{arg\,max}_{\vec{x} \in (-\pi, \pi]^{2L}} |\Delta'(\mu; \vec{x})|. \end{aligned} \tag{220}$$

By Eq. (213), we expect that a solution to (219) would be a good proxy for maximizing the expected posterior variance when σ is small, i.e. if $\hat{x}_{\mu,f}$ is a solution to (219) on input (μ, f), then we expect that

$$\max_{\vec{x} \in (-\pi, \pi]^{2L}} \mathcal{V}(\mu, \sigma; f, \vec{x}) \approx \frac{\Delta'(\mu; \hat{x}_{\mu,f})^2}{f^{-2} - \Delta(\mu; \hat{x}_{\mu,f})^2} \qquad \text{when } \sigma \text{ is small.} \tag{221}$$

Similarly, by Eq. (217), we expect that a solution to (220) would be a good proxy for maximizing the expected posterior variance when both f and σ is small, i.e. if $\hat{x}_\mu$ is a solution to (220) on input μ, then we expect that

$$\max_{\vec{x} \in (-\pi, \pi]^{2L}} \mathcal{V}(\mu, \sigma; f, \vec{x}) \approx f^2 \Delta'(\mu; \hat{x}_\mu)^2 \qquad \text{when } \sigma \text{ and } f \text{ are small.} \tag{222}$$

To investigate the performance of the proxies (221) and (222), we numerically maximize the expected posterior variance and the proxies (219) and (220) for small problem sizes L. The results of this optimization are presented in Figures C.2–C.5. For $L = 1$, it turns out that the optimization problem (220) for the ancilla-free bias can be solved analytically. We present this analytical solution in Appendix C.3.

C.3 Analytical expressions for $L = 1$ slope proxy

In this appendix, we present an analytical solution to the optimization problem (220) for the ancilla-free bias when $L = 1$. In this case, the bias (22) can be written as the Fourier series

$$\Delta(\theta; x_1, x_2) = \sum_{l=0}^{3} \mu_l(x_1, x_2) \cos(l\theta), \tag{223}$$

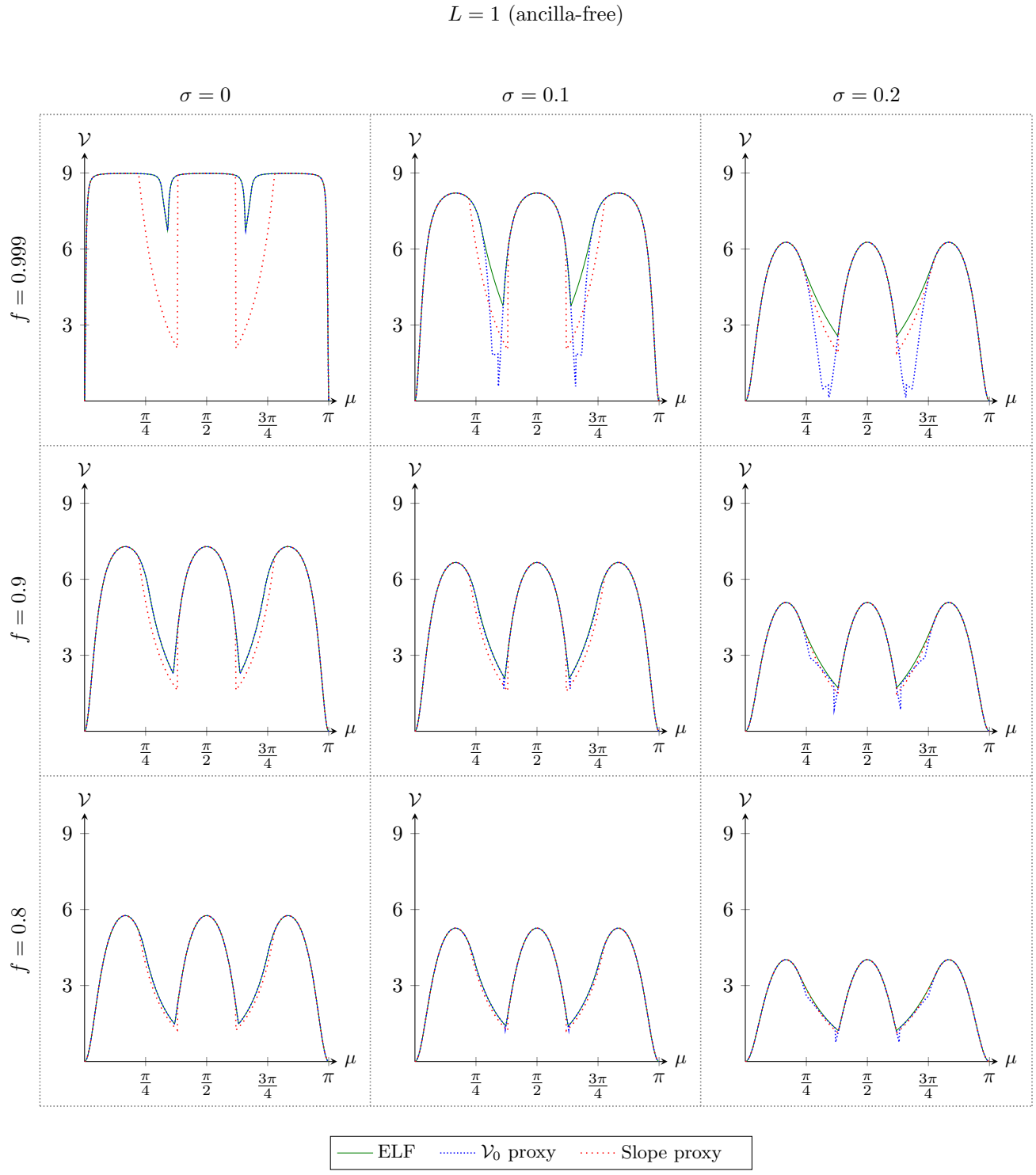


Figure C.2: Plots of the variance reduction factor $\mathcal{V}$ vs the prior mean μ for $L = 1$ for the ancilla-free scheme. The proxies work well when f and σ are not too large (say $f \leq 0.9$ and $\sigma \leq 0.1$). When f and σ are both large (for example, $f = 0.999$ and $\sigma = 0.2$), the proxies fail to be good approximations). These figures were obtained with Wolfram Mathematica's `NMaximize` RandomSearch method with 120 search points.

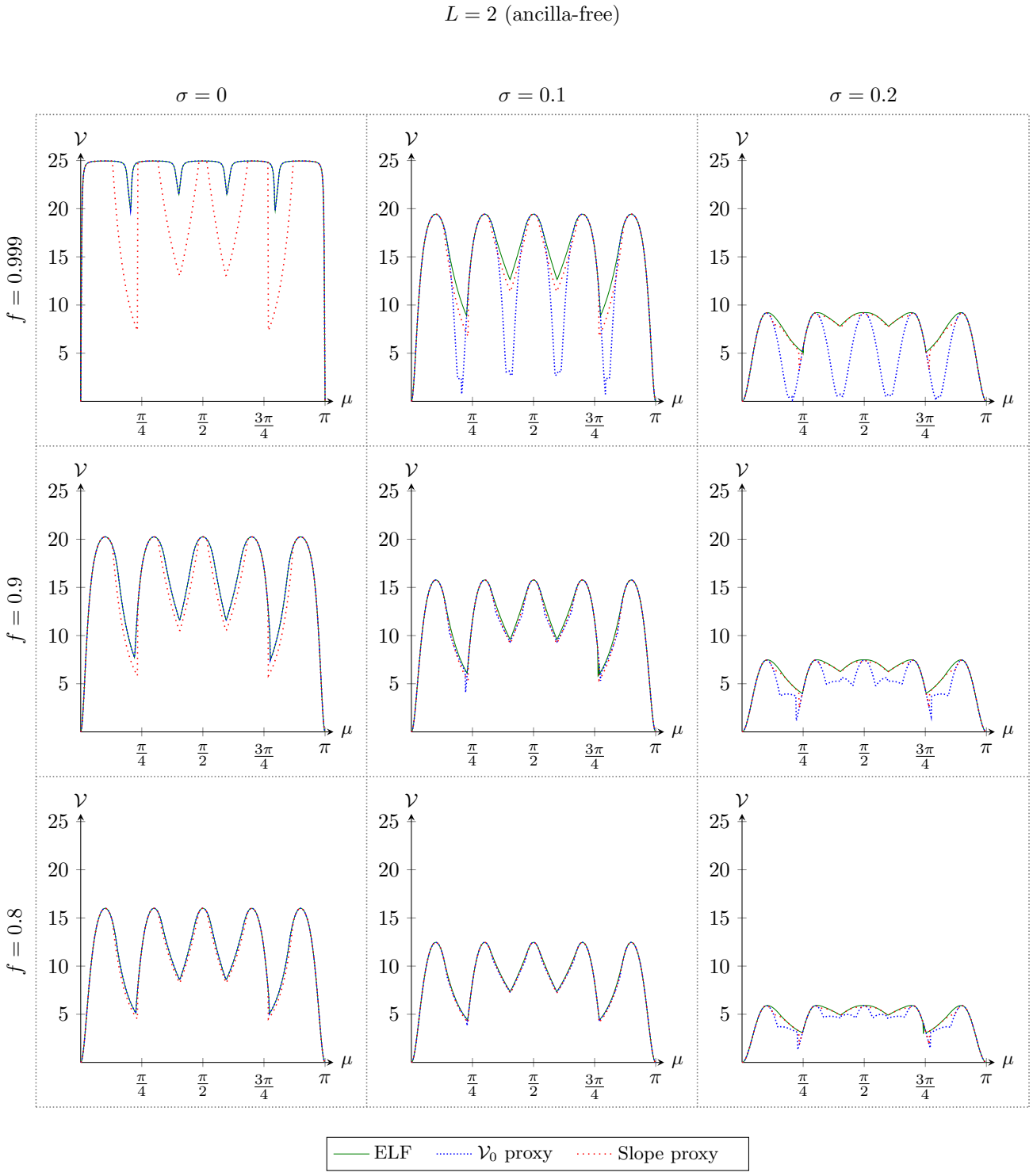


Figure C.3: Plots of the variance reduction factor $\mathcal{V}$ vs the prior mean μ for $L = 2$ for the ancilla-free scheme. The proxies work well when f and σ are not too large (say $f \leq 0.9$ and $\sigma \leq 0.1$). When f and σ are both large (for example, $f = 0.999$ and $\sigma = 0.2$), the proxies fail to be good approximations). These figures were obtained with Wolfram Mathematica's `NMaximize` RandomSearch method with 220 search points.

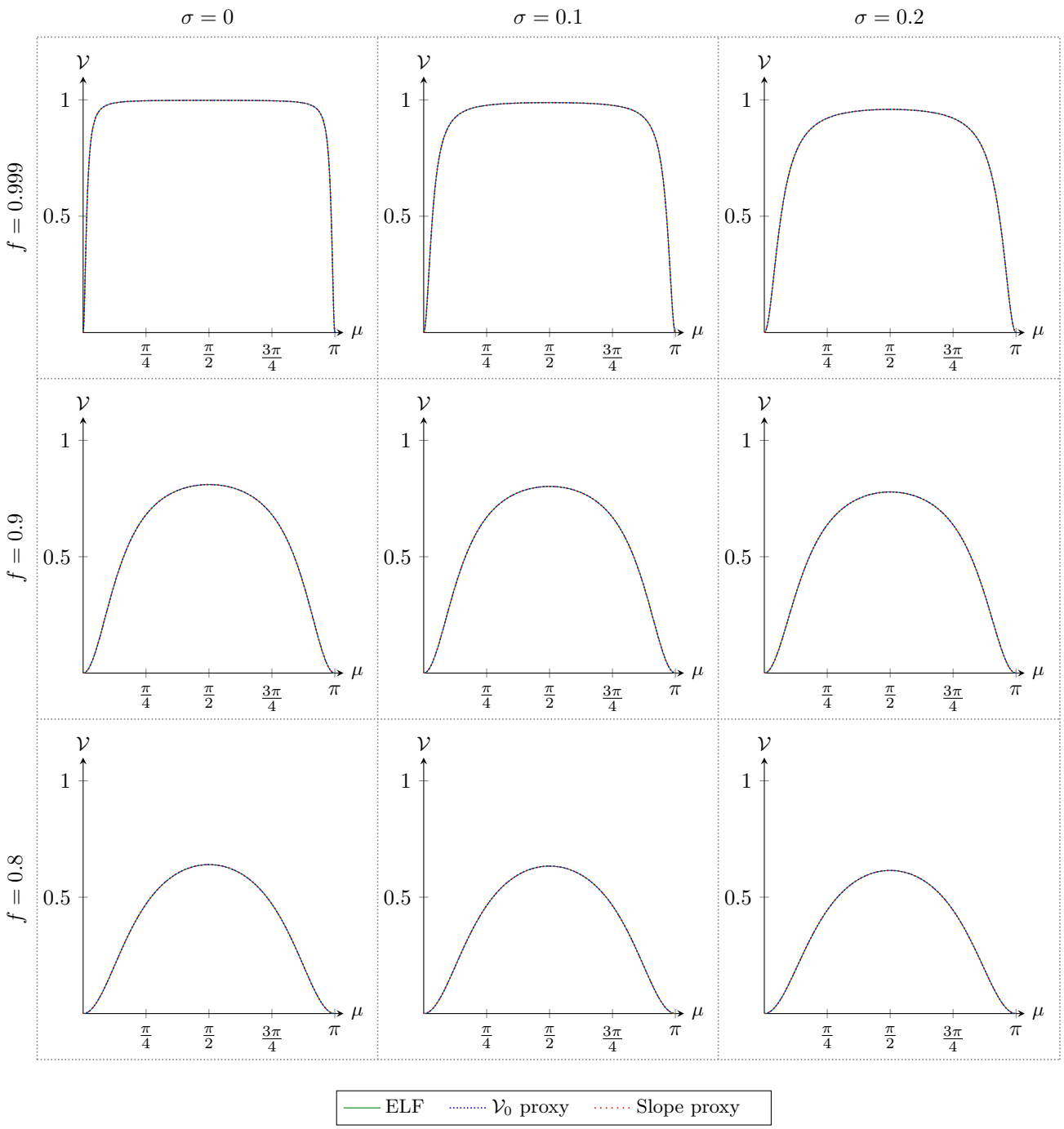


Figure C.4: Plots of the variance reduction factor $\mathcal{V}$ vs the prior mean μ for $L = 1$ for the ancilla-based scheme. The proxies give identical results to the exact optimization of the variance reduction factor. These figures were obtained with Wolfram Mathematica's `NMaximize` RandomSearch method with 120 search points.

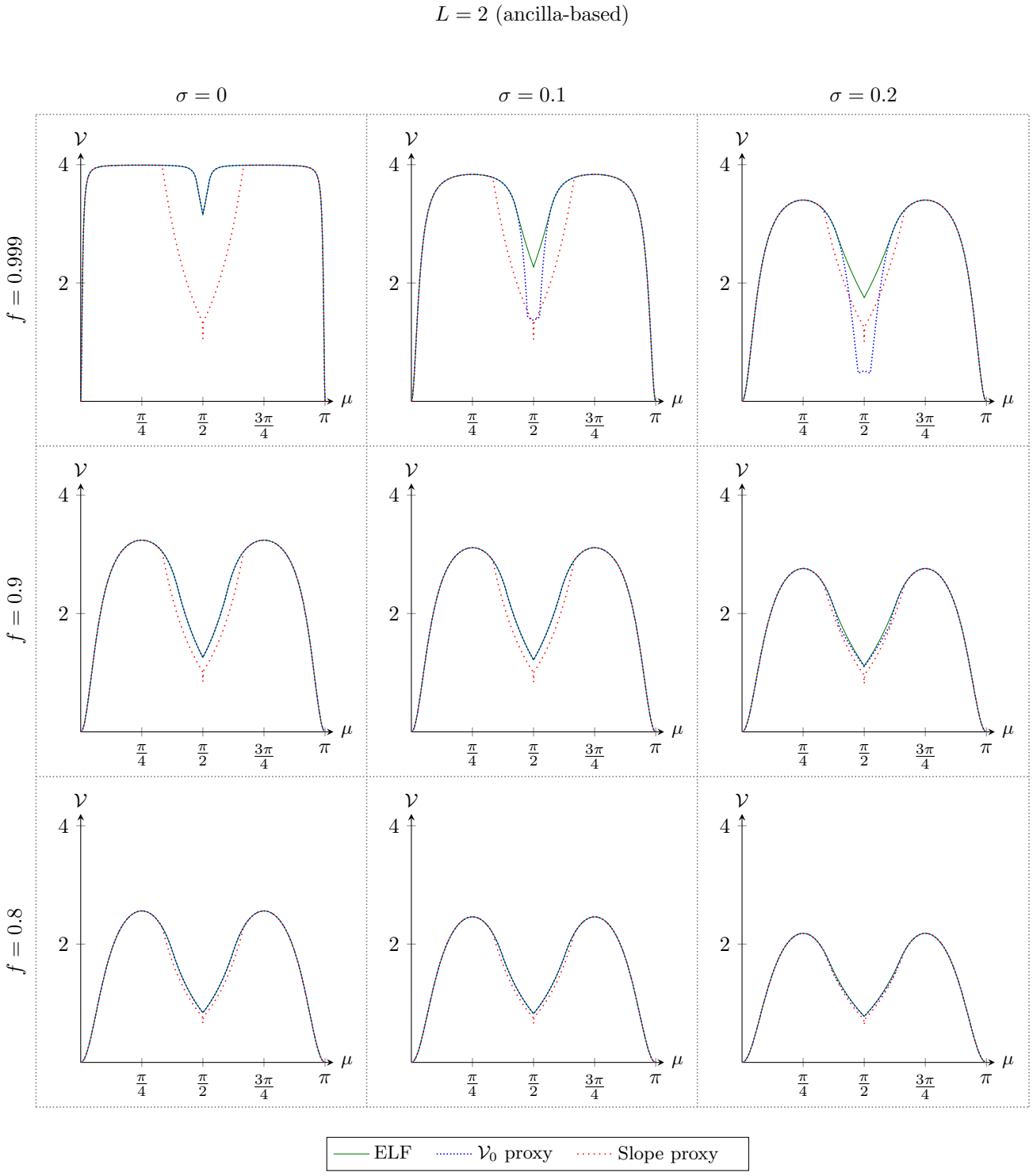


Figure C.5: Plots of the variance reduction factor $\mathcal{V}$ vs the prior mean μ for $L = 2$ for the ancilla-based scheme. The proxies work well when f and σ are not too large (say $f \leq 0.9$ and $\sigma \leq 0.1$). When f and σ are both large (for example, $f = 0.999$ and $\sigma = 0.2$), the proxies fail to be good approximations). These figures were obtained with Wolfram Mathematica's `NMaximize` RandomSearch method with 120 search points.

where

$$\begin{aligned}
\mu_0(x_1,x_2) &= 2\cos(x_1)\sin(x_2)\sin(x_1)\cos(x_2)\\
\mu_1(x_1,x_2) &= \cos^2(x_1)\cos^2(x_2)+\cos^2(x_1)\sin^2(x_2)+\cos^2(x_1)\sin^2(x_2)\\
\mu_2(x_1,x_2) &= -2\cos(x_1)\cos(x_2)\sin(x_1)\sin(x_2)\\
\mu_3(x_1,x_2) &= \sin^2(x_1)\sin^2(x_2).
\end{aligned} \tag{224}$$

The optimization problem (220) may be stated as

$$\begin{aligned}
\text{Maximize} \quad & |\Delta'(\mu;x_1,x_2)| = \big| -\left[\cos^2(x_1)\cos^2(x_2)+\cos^2(x_1)\sin^2(x_2)+\cos^2(x_1)\sin^2(x_2)\right]\sin(\mu)\\
& \qquad + 4\cos(x_1)\cos(x_2)\sin(x_1)\sin(x_2)\sin(2\mu)\\
& \qquad - 3\sin^2(x_1)\sin^2(x_2)\sin(3\mu)\big|\\
\text{subject to} \quad & x_1,x_2\in(-\pi,\pi].
\end{aligned} \tag{225}$$

Solving Eq. (225) gives the following solution.

Proposition 3. *The maximum magnitude of the slope of the $L=1$ likelihood function is*

$$\max_{(x_1,x_2)\in[0,\pi]^2}|\Delta'(\mu;x_1,x_2)| = \begin{cases} 3\sin(3\mu), & \mu\in[0,\mu_1]\cup[\mu_4,\pi]\\ \frac{4\cos^4(\mu/2)\cot(\mu/2)}{1+3\cos(\mu)}, & \mu\in(\mu_1,\mu_2)\\ -3\sin(3\mu), & \mu\in[\mu_2,\mu_3]\\ \frac{4\sin^4(\mu/2)\tan(\mu/2)}{1-3\cos(\mu)}, & \mu\in(\mu_3,\mu_4)\end{cases} \tag{226}$$

and an example of a pair of angles that achieves this maximum is

$$(\gamma_1,\gamma_2)\in\operatorname*{arg\,max}_{(x_1,x_2)\in[0,\pi]^2}|\Delta'(\mu;x_1,x_2)|, \tag{227}$$

where for $i=1,2$,

$$\gamma_i = \begin{cases} \frac{\pi}{2}, & \mu\in[0,\mu_1]\cup[\mu_2,\mu_3]\cup[\mu_4,\pi]\\ (-1)^i\operatorname{arccot}(\sqrt{1-3\cos(\mu)+\sec(\mu)}), & \mu\in(\mu_1,\mu_2)\\ \operatorname{arccot}(\sqrt{1+3\cos(\mu)-\sec(\mu)}), & \mu\in(\mu_3,\mu_4)\end{cases} \tag{228}$$

where μ_1,μ_2,μ_3,μ_4 are given by

$$\mu_1 = 2\arctan\left[\sqrt{\tfrac{1}{3}\left(4-\sqrt{13}\right)}\right]\approx 0.6957 \tag{229}$$

$$\mu_2 = 4\arctan\left[\sqrt{\operatorname{root}_3(p_2)}\right]\approx 1.1971 \tag{230}$$

$$\mu_3 = 4\arctan\left[\sqrt{\operatorname{root}_3(p_3)}\right]\approx 1.9445 \tag{231}$$

$$\mu_4 = 2\arctan\left[\sqrt{4+\sqrt{13}}\,\right]\approx 2.4459 \tag{232}$$

where p_2 and p_3 are octic polynomials given by

$$p_2(x) = 1+72x-1540x^2+8568x^3-16506x^4+8568x^5-1540x^6+72x^7+x^8, \tag{233}$$

$$p_3(x) = 9-264x+2492x^2-9016x^3+13302x^4-9016x^5+2492x^6-264x^7+9x^8. \tag{234}$$

The notation $\operatorname{root}_a(p)$ refers to the ath smallest (with starting index 1) real root of the polynomial p.

A plot of Eq. (226) is shown in Figure C.6.

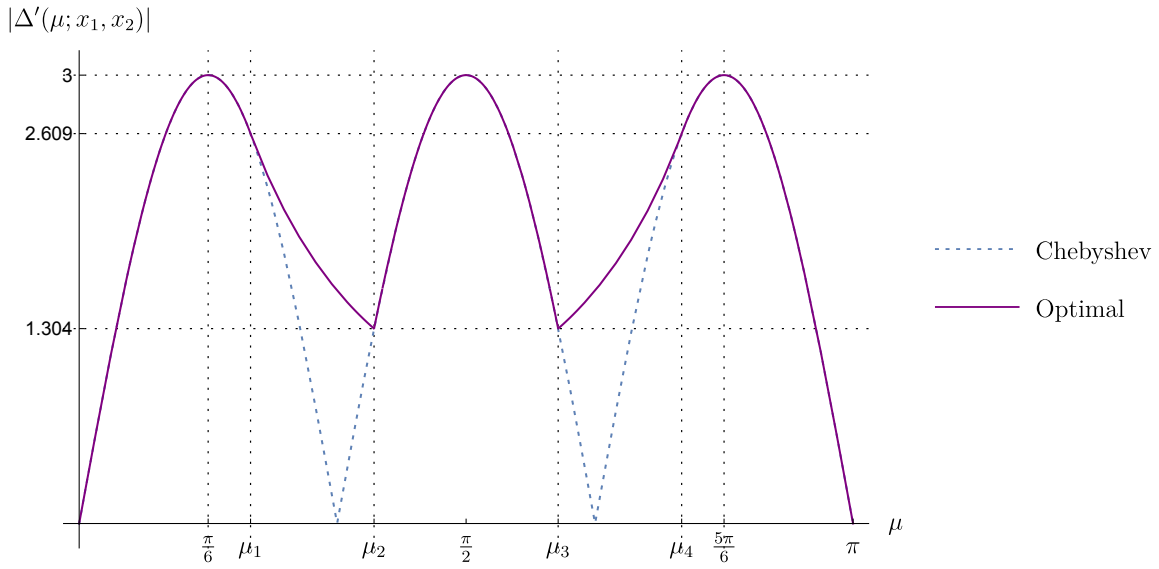


Figure C.6: Graph of the optimal slope, with angles μ_1, μ_2, μ_3, and μ_4 given by Eqs. (229), (230), (231), and (232), respectively.