Exqaliber Stage 1 Report

James Cruise[1], Joseph Tedds[1], Camille de Valk[2], and Walden Killick[1]

[1]Cambridge Consultants, Cambridge, UK
[2]Capgemini QLab

TBD

1 Introduction

A number of key quantum subroutines, including both quantum phase estimation and amplitude estimation, in their standard forms [**?** **?**] require fault tolerant quantum computers. This requirement is due to their long circuit depth coupled with their lack of resilience to any noise. This requirement means their viability in the coming decade is very low and could prevent the potential of quantum computing be unrealised for many years to come. A recent trend in quantum algorithms development has started to explore other alternative formulations [1, 2] with significantly lower requirements, due to both shorter circuits and potential for noise resilience. The fundamental idea is to use the quantum computer to cleverly generate samples from a range of distributions dependent on the information/parameter of interest and process these samples to provide an estimate for the parameter. For this reason we refer to this general approach for amplitude estimation as statistical amplitude estimation (SAE). This work builds these approaches to develop a new algorithm, Recursive Amplitude Estimation (RAE), through the utilisation a Bayesian framework to enable recursive approach sample collection. This enables us to produce an algorithm which both optimises the circuit execution in an online fashion reducing the total computational work to generate an estimate to a given accuracy but is also easily adaptable to the inclusion of noise.

Quantum amplitude estimation (QAE) is a fundamental subroutine in many application areas including quantum chemistry, machine learning, finance, and more. It is the problem of, given access to an $(n+1)$-qubit oracle $\mathcal{A}$ such that $\mathcal{A}|0\rangle = \sqrt{a}\,|\phi_1\rangle\,|1\rangle + \sqrt{1-a}\,|\phi_0\rangle\,|0\rangle$, estimating a. This is a problem of great current interest due to its promise of a quantum quadratic speedup, including potentially on noisy near-term devices.

The first such algorithm, proposed by Brassard et al. [3], works by combining the quantum phase estimation (QPE) algorithm [4] with the Grover iteration operator. Although this achieves a provable quantum speedup, this approach is unsuitable for use on near-term devices primarily due to its large circuit depth stemming from (1) the quantum Fourier transform (QFT) [5] and (2) the controlled unitaries required to prepare the input state of the QFT but also its lack of resilience to noise.

James Cruise: james.cruise@cambridgeconsultants.com
Joseph Tedds: joseph.tedds@cambridgeconsultants.com
Camille de Valk: camille.de.valk@capgemini.com
Walden Killick: walden.killick@cambridgeconsultants.com

[cdv 1] I don't think we should bring up the math in the first paragraph. Maybe we can keep it a bit more high-level.

[jc 1] New paragraph added to deal with this

Several recent approaches have studied algorithms achieving the same or similar asymptotic speedups without the use of the QFT, by repeated sampling of the state after further applications of the Grover operator [6, 1, 7]. To indicate the reliance on sampling we refer to these as Statistical Amplitude Estimation (SAE). The underlying motif of this paradigm is the learning of the amplitude by statistical sampling, in contrast to Brassard et al.'s method which allows for direct access to the amplitude with high probability. For further details on the statistical approach see Section 1.1.1.

This work introduces Recursive Amplitude Estimation (RAE) which builds upon the previous work in SAE through the introduction two stage algorithms, the first stage is an online optimisation of the sample generation scheme and the second is a post-processing step to generate the final amplitude estimate. Details of this algorithm can be found in Section **??**. The online optimisation utilises a Bayesian framework to enable a recursive estimate for both the amplitude and our current uncertainty which enables us to optimise the next quantum circuit to be run to maximise the information gain measured through the expected decrease in our uncertainty. Unfortunately, an approximation is required to reduce the computational burden of the online optimisation. A final post-processing step utilises all the obtained samples to obtain final amplitude estimation which removes the approximation errors.

1.1 Amplitude estimation

A key subroutine in many quantum algorithms is the estimation of an unknown amplitude for a state generated by a given oracle. The challenge of estimating this amplitude is formally given by this problem:

Problem 1 (Quantum Amplitude Estimation)**.** *Given some* $(n+1)$*-qubit oracle* $\mathcal{A}$ *which generates the state*

$$\mathcal{A}\left|0\right\rangle^{\otimes n+1} = \sqrt{a}\left|\phi_1\right\rangle\left|1\right\rangle + \sqrt{1-a}\left|\phi_0\right\rangle\left|0\right\rangle,$$

where $\left|\phi_1\right\rangle$ *and* $\left|\phi_0\right\rangle$ *are arbitrary normalised* n*-qubit states and* $0 \leq a \leq 1$*, quantum amplitude estimation (QAE) is the problem of estimating the unknown amplitude* $\sqrt{a}$*.*

The 'naive', classical way to solve this problem is to simply prepare and measure this state N times, counting the number of times K, that $\left|\phi_1\right\rangle\left|1\right\rangle$ was observed, then define the estimate for $\sqrt{a}$ as $\sqrt{K/N}$. Chebyshev's inequality tells us that taking $N \in O(\varepsilon^{-2})$ samples is sufficient to approximate $\sqrt{a}$ within additive error ε (with, say, 99% confidence). Furthermore, this is asymptotically optimal [8]. Given each sample requires the application of oracle once to generate the state to be measured the computational complexity is $O(\varepsilon^{-2})$.

In contrast, a quantum algorithm proposed in [3] provides a quadratic speed up. Here to obtain an estimated of accuracy ε we only need to carry out $O(\varepsilon^{-1})$ calls to the oracle. This is achieved by using the Grover operator [], see figure 1, with quantum phase estimation, see figure 2 for the circuit. The Heisenberg limit guarantees that there are no significant speed ups beyond this scaling. Note that to implement this on a quantum computer will require a very long circuit depth, as all the oracle operations are preformed in a single circuit, and it is not resilient to any noise.

An important and general application of QAE is Monte Carlo estimation [9, 10, 1] - a method for estimating an integral. A specific use case of this with applicability in finance and many other industries is an estimation of the mean value of a function through random sampling. More precisely, given some function $f : \{0,1\}^n \to [0,1]$, the task is to estimate

the mean value

$$\mathbb{E}[f(X)] = \frac{1}{2^n} \sum_{x=0}^{2^n-1} f(x).$$

Montanaro [10] presents a method for encoding this value in the amplitude of a quantum state, thereby reducing this problem to a case of amplitude estimation, which in turn offers a quadratic speedup for Monte Carlo estimation over classical methods.

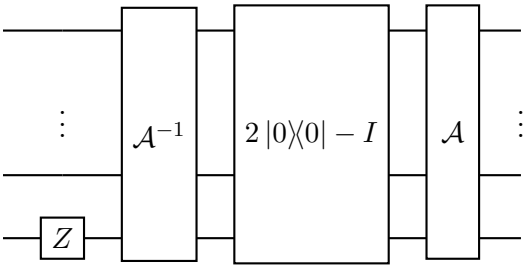


Figure 1: Grover operator, $\mathcal{U}$, where $\mathcal{A}$ is the oracle, $2\,|0\rangle\langle 0| - I$ is a reflection through the standard basis and Z is a phase flip.

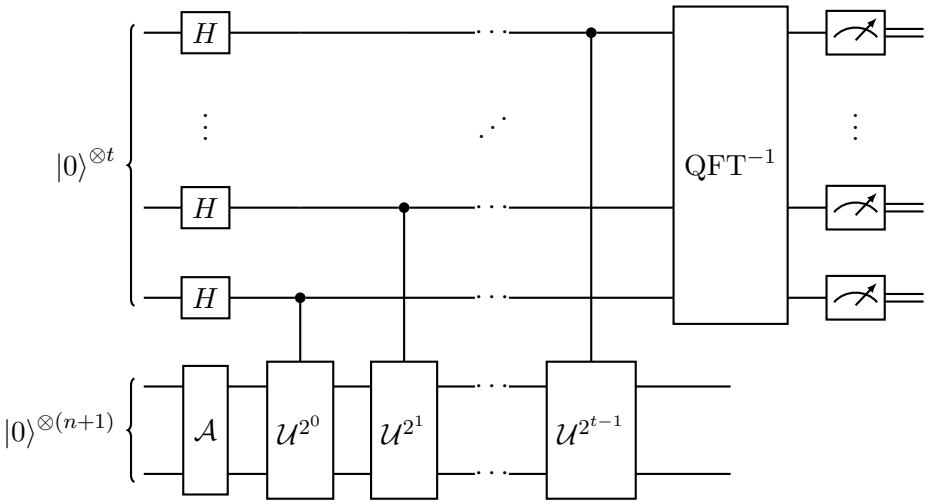


Figure 2: Amplitude estimation using quantum Fourier transform.

1.1.1 Statistical amplitude estimation

The original QAE algorithm described in the previous section requires a fault tolerant quantum computer due to its long circuit and lack of noise resilience. With current progress in quantum hardware, this means that we are unlikely to be able to utilise this algorithm for at least a decade. This has led to interest in alternative paradigms which takes a statistical approach to estimating the amplitude. Here an estimate of $\sqrt{a}$ is maintained. This is updated using the measurement from the execution of the circuit shown in figure 3 which is an amplitude amplification of the state $\mathcal{A}\,|0\rangle$. This measurement will follow a known distribution, see below, which depends on unknown amplitude and the amount of amplification applied. Through repeated updates, we obtain an estimate of the amplitude with increased confidence.

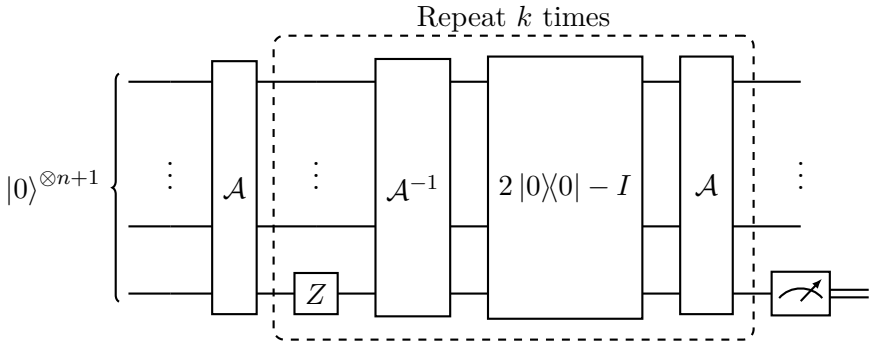


Figure 3: Circuit for amplitude amplification. The probabilities for the measurement outcomes are $\Pr(|1\rangle) = \sin^2(2k+1)\theta$ and $\Pr(|0\rangle) = \cos^2(2k+1)\theta$.

Define $\theta \in [0, \pi/2]$ such that $sin(\theta) = \sqrt{a}$, we can rewrite the state as

$$\mathcal{A}\,|0\rangle = \sin\theta\,|\phi_1\rangle\,|1\rangle + \cos\theta\,|\phi_0\rangle\,|0\rangle\,.$$

Now consider the Grover operator, see figure 1, given by

$$\mathcal{U} = \mathcal{A}\left(2\left|0^{n+1}\right\rangle\!\left\langle 0^{n+1}\right| - I_{2^{n+1}}\right)\mathcal{A}^{-1}\left(I_{2^n} \otimes Z\right),$$

see Figure 3. In the subspace spanned by $|\phi_1\rangle\,|1\rangle$ and $|\phi_0\rangle\,|0\rangle$, by a sequence of reflections, each application of $\mathcal{U}$ performs a rotation of angle 2θ towards $|\phi_1\rangle\,|1\rangle$. This gives,

$$\mathcal{U}^m\mathcal{A}\,|0\rangle = \sin(2m+1)\theta\,|\phi_1\rangle\,|1\rangle + \cos(2m+1)\theta\,|\phi_0\rangle\,|0\rangle$$

and so $\mathcal{U}^k$ takes the probability of measuring $|\phi_1\rangle\,|1\rangle$ from $\sin^2\theta$ to $\sin^2(2k+1)\theta$. Now by careful selection of k and processing the measurements we can estimate θ.

1.1.2 Decoherent noise model

An example of the importance of incorporating noise resilience, assume we are in a situation where the true angle $\theta = 0$. Under the assumption that the system is noiseless, no number of applications of the Grover operators will affect the state and so one would never measure $|\phi_1\rangle\,|1\rangle$. However, on actual noisy hardware the noise of the circuit implementation is likely to cause the probability of measuring $|\phi_1\rangle\,|1\rangle$ to rise above 0. Measuring this state without incorporating mitigation against noise will effectively eliminate the true value from the possible estimates. Incorporating a model for noise mitigates this by effectively decreasing confidence in the measurement outcome as the number of Grover iterations increases, i.e the noise level increases.

In near term quantum computation errors will play a significant role and algorithms will need to incorporate error resilience. To take the first steps into considering how to deliver this necessary resilience we consider an initial simple noise model specifically for depolarising noise.

For the simple model of depolarising noise used in this paper, we assume that each application of the oracle $\mathcal{A}$ also applies a depolarising channel. The practical effect of this is that after applying the oracle k times, with probability p^k we measure as if there had been no noise and with probability $1 - p^k$ we are equally likely to measure 0 or 1. This means that as the number of oracle applications the probability of interested converges to a $1/2$, so in the limit giving no information about θ and the original state.

In the previous section we showed for noiseless system the probability of measuring $|\phi_1\rangle|1\rangle$ after k iterations of U to $\mathcal{A}|0\rangle$ is $\sin^2(2k+1)\theta$, which can be rewritten as

$$\Pr(|\phi_1\rangle|1\rangle) = \sin^2((2k+1)\theta) = \frac{1}{2}(1-\cos(4k+2)\theta).$$

Now we incorporate the depolarising noise as discussed above. Rather than quantifying the level of noise by p, it is natural to let $p = e^{-\lambda}$ for $\lambda \geq 0$. In this formulation λ represents the expected number of depolarising errors per oracle application. Using this we get probability of measuring $|\phi_1\rangle|1\rangle$ after k iterations of $\mathcal{U}$ as

$$\Pr(|\phi_1\rangle|1\rangle) = \frac{1}{2}\left(1 - e^{-\lambda(2k+1)}\cos(4k+2)\theta\right).$$

Here, the exponent contains the term $2k+1$ since we need one application of $\mathcal{A}$ to prepare the state, then each iteration of $\mathcal{U}$ requires one call to $\mathcal{A}$ and another to its inverse.

[cdv 2] Here we have the 4 again, instead of the 2.

[jc 2] We need to fix this, for this piece it makes sense not have rescaled but can do the rescaling

1.2 Related work

Brassard et al. [3] authored the first QAE algorithm, now often referred to as canonical QAE. The idea is that the Grover operator $\mathcal{U}$, see Figure 1, as defined above contains the amplitude of the corresponding state in its eigenvalues. Thus the QPE algorithm can be used with $\mathcal{U}$ to extract the value of interest. This method allows for direct access to the amplitude to within additive error ε with high probability using $O(\varepsilon^{-1})$ queries to $\mathcal{A}$. However, its reliance on controlled Grover iterates and the QFT make it very difficult to implement on near-term devices, which has inspired much subsequent work on QAE without the use of the QFT. Other initial work on alternative QAE schemes can be found in [7, 6]. In both cases they still demonstrate the quadratic speed-up but require significant overheads.

Much of the interest in SAE we started by the work of Suzuki et al. [1]. They propose a QFT-free QAE algorithm based on maximum likelihood estimation , which we refer to as Maximum Likelihood Amplitude Estimation(MLAE). Numerical simulation show that their algorithm achieves the asymptotically optimal scaling of measurements. The sampling scheme which achieves this scaling incurs an exponentially increasing number of Grover iterates at each step and fails to account for the potential noise. Heuristic methods have been employed to take account of the noise [] but no clear framework is given for incorporating this into the sampling scheme. Furthermore, the sampling schedule is generated in an offline fashion with no ability to adjust based on both the current estimate of uncertainty and amplitude. As discussed in [11] this can cause problems in the estimation near amplitudes of the form $\frac{\pi}{2^n}$. In this work they a jitter to the scheme to mitigate this behaviour but in our work we actively mitigate this through online decisions about the number of applications of the Grover operator.

Other work in the class of SAE include, Grinko et al. [12] present an algorithm called Iterative AE (IAE) which combines ideas from previous works and greedily chooses the number of Grover iterates. They prove that their algorithm is optimal up to a double-logarithmic factor and has much smaller constant factors than comparable algorithms.

Giurgica-Tiron et al. [13] design algorithms which interpolate between classical and quantum amplitude estimation algorithms with the aim of utilising parallelism to minimise overall circuit depth. Their algorithms achieve a query complexity of $\tilde{O}\left(\varepsilon^{-(1+\beta)}\right)$ where

$\beta \in (0, 1]$ is some parameter corresponding to the balance between classical and quantum queries.

Smith et al. [2] present an algorithm for QFT-free QPE (which can equivalently be used for QAE by replacing the unitary operator with the Grover iterate) which is also based on maximising Fisher information. They prove that it achieves the Heisenberg limit in the noiseless case, and prove that it achieves the best possible query complexity in the presence of depolarising noise.

1.3 Contribution

In this work we develop a novel algorithm for SAE which we call Recursive Amplitude Estimation(RAE). This is a two stage algorithm, an online sampling scheme and then a post-processing estimation procedure. In comparison to previous algorithms, for our sampling schedule we aim to maximise the value of the quantum hardware by optimising the information gained per circuit execution by minimising the expected uncertainty. To do this we utilise a Bayesian framework which provides a natural tool set for this. The true posterior distribution is computationally expensive to fully specify, so we utilise an approximation based on the Normal distribution to reduce the computational overhead. This provides a good estimate while carrying out the online sampling scheme but can lead to errors in the final estimate. For this reason we use a post-processing step utilising all the obtained samples to produce the final estimate.

[jlt 1] Do we want to say anything about Power Law or Qo-Prime explicitly here? Feels like we should at least name check them.

[jc 3] Probably yes, to be done at next draft

The algorithmic framework put forward in this work allows us to integrate noise models both in the sampling schedule but also in the generation of the final estimate. To highlight this we have considered a simple model of depolarizing noise introduced in Section 1.1.2.

In the rest of this paper we provide more details of the algorithm and an experimental study of algorithmic performance. In Section **??** we describe the algorithm in more detail and the theoretical underpinnings. The first part of this section describes the online sampling scheme in both the noise free and depolarising noise scenarios. The later part of this section describes the post processing step. After this in Section 3 we carry out an experimental study of RAE and its performance in comparison to other SAE algorithms. Finally we include some concluding remarks and future looking thoughts in Section 4.

2 Algorithm

As discussed in **??** the algorithm is made up of two parts, the sampling algorithms which provides a dynamic sampling scheme to reach the desired accuracy and the post-processing algorithm which generate the final estimate of θ in an offline fashion. The reason for this two-stage approach is that maintaining a true representation of the current state and uncertainty during the sampling scheme is too computationally intensive so, we utilise a low dimensional approximation. In most cases, this provides a high quality estimate which enables us to carry out an online optimisation of sample number and depth. Once all the samples have been obtained, instead of returning the approximation, which does not have the full predictive power, we re-estimate θ with the full set of measurement data obtained during sampling. This means the errors introduced by the approximation procedure do not affect the final estimate.

In this paper, we demonstrate a novel method for generating a sample schedule dynamically by modelling an estimate of θ that is updated after each measurement. For the post-processing we use the maximum likelihood estimate (MLE) originally described in [**?**]. For this reason here we concentrate on the sampling algorithm and include a short introduction to the MLE at the end of this section.

2.1 Sampling algorithm

Here, we describe the sampling algorithm and associated theoretical underpinning initially in the noiseless case and then introduce the adjustment required to adapt to~~deal with the introduction of~~[jlt] depolarising noise. The fundamental idea is to~~that we~~[jlt] maintain a current state representing our belief about the value of the parameter of interest, μ, and the current level of uncertainty, σ^2. Then at each step we decide if enough samples have been obtained to terminate the process[jlt], i.e. σ^2 is small enough for the required accuracy, or to continue to generate further samples~~generate another sample~~[jlt].

If another sample is generated, we need to select the depth at which the sampling circuit is executed and using the result, update the state, i.e. our belief. In both cases we use a Bayesian framework to do this, see 2.1.1. Within this framework, the uncertainty σ^2 is an estimate of the posterior variance for θ and μ, the current belief of θ, is an estimate of the posterior mean. At each step, the next sampling depth is selected to minimise the expected posterior variance given the estimate of the current belief, i.e. the depth which, in expectation, moves us closest to terminating the sampling. After executing the sampling circuit, the state is updated by calculating the posterior mean and variance. Unfortunately, maintaining and manipulating an estimate of the true posterior distribution is computationally intensive so after every step we approximate the posterior distribution by a normal distribution. This is then used as the prior distribution for the next step, for more details see Section 2.1.1.

Having described the algorithm we now provide a pseudocode specification of the algorithm, see 1. There are a few points to note from this pseudocode. Firstly, the procedure 'Sample' executes the quantum circuit shown in Figure **??**, where d is the number of times the Grover operator is applied. The measurement of 0 or 1 is then stored as x. Secondly, as described in Section 2.2 we post-process the obtained samples to produce the final estimate of θ, so 'Store' stores the obtained sample with the associated depth for later use. Thirdly, there are three closed form mathematical functions, g_1, g_2, h, used within the pseudocode defined in Section 2.1.2. Note that the only change to this sampling algorithm when considering

[jlt 2] This needs filling in somewhere - we're not explicit enough about this yet.

[jlt 3] Not sure I understand what you mean by sample number.

[jlt 4] Think the above needs some more thought - I'm not clear that these two bits are distinct enough.

[cdv 3] I like this pseudocode. However, when I read it through the story, I don't fully understand it, because the equations 4, 2, and 3 come much later.

the depolarising noise is to g_1, g_2, h otherwise the algorithm stays the same.

Algorithm 1 Pseudocode for sampling algorithm

Require: $(\mu, \sigma^2), \epsilon, \alpha$
  while $\sigma > \Phi^{-1}(1-\alpha)\epsilon$ **do** ▷ Φ is the normal distribution function
    $d \leftarrow \arg\max_{d>0} h(\mu, \sigma^2, d)$ ▷ See equation 4
    $x \leftarrow$ SAMPLE(d) ▷ Execute circuit depth at d
    $\mu \leftarrow g_1(\mu, \sigma^2, x, d)$
    $\sigma^2 \leftarrow g_2(\mu, \sigma^2, x, d)$ ▷ Update state, see equations 2 and 3
    STORE(x, d) ▷ Store observation and sample depth for post-processing
  end while

[jlt 5] The update steps are simultaneous - does this pseudocode accurately reflect this?

2.1.1 Bayesian framework

Through the design of this algorithm we have embraced a Bayesian approach [], and used this to generate a sequential sampling algorithm. Note, an important difference from the standard sequential algorithms is that we are usually only interested in deciding when to stop sampling, but here we also have control of the sampling distribution.

The fundamental feature of this approach is that at all stages we maintain a belief about our current knowledge of the unknown parameter, θ, through a probability distribution, $\pi(\theta)$. This belief is updated after each sample using Bayes formula [] and the current belief is used to make the decision of whether to stop or generate a new sample and in that case what depth to sample at.

Unfortunately, keeping the full probability distribution is computationally infeasible so instead we approximate our belief by a normal distribution, i.e. $\pi(\theta) \sim N(\mu, \sigma^2)$, which is fully described by two parameters: the mean, μ, and the variance, σ^2. Therefore at each stage we approximate the posterior distribution, which becomes the prior when considering the next sample, by a normal distribution with the same variance and mean as the one step posterior distribution. Note that this mean and variance will not necessarily match the posterior mean and variance if all the obtained samples were considered, due to the repeated approximation.

The normal distribution is a sensible approximation since under suitable conditions, see [], in the limit of a large number of samples the posterior limits to a normal distribution. Further to this, using the normal distribution as our prior belief, we are able to use Bayes theorem to produce analytic expressions for a number of key posterior parameters including the posterior mean and variance and expected posterior variance, see Section 2.1.2. This is important as we could obtain estimates of these through sampling schemes [], but as we discuss in Section 2.1.5, this leads to a systematic bias.

Though using a normal approximation will generally provide a good ongoing estimate, there are approximation errors which can be substantial in nature. For some experimental examples of this see Section**??**. This is the reason that we carry out a two-stage algorithm and post-process the samples to provide the final estimate for the parameter of interest, Section 2.1.2. Two relevant examples of approximation errors are:

- The parameter of interest, θ, is best thought of as an angle, i.e. 0 and 2π should be identified with each other. This means that we have to be especially careful near boundary points, for example, 0.

- We are sampling from a Bernoulli distribution where the probability of a 1 is determined by the unknown parameter θ and the circuit depth. Unfortunately, as discussed in Section **??** the exact mapping to probability has a degeneracy when consider depths greater than 1, i.e. there are multiple values of θ which give the same value of a 1. If care is not taken this can lead to a multimodal posterior distribution.

2.1.2 Posterior properties

Two key steps in the algorithm are calculating the expected posterior variance, $Var(\theta)$ for a single sample, which is used to decide on the next sample depth and calculating the posterior mean and variance, $\mathbb{E}\left[\theta|X=x\right]$ and $Var(\theta|X=x)$. As we have already seen in the pseudocode these play a significant role in our algorithm. Under the assumption that the prior follows a normal distribution with known mean and variance, we can obtain closed-form expressions for all three quantities. Our derivation follows closely the work described in [] which considers a class of likelihoods of the same form as we see in this work. Details of the derivation can be found in Appendix A.1 and here we only include the final formulae for use with the algorithm.

$$\begin{aligned}\mathrm{Var}(\theta) &= \sigma^2\left(1-\sigma^2\frac{4\left(2d+1\right)^2 e^{-4(2d+1)^2\sigma^2}\sin^2\left(2\left(2d+1\right)\mu\right)}{1-e^{-4(2d+1)^2\sigma^2}\cos^2\left(2\left(2d+1\right)\mu\right)}\right) \\ &= \sigma^2\left(1-\sigma^2\mathcal{V}\right)\end{aligned} \tag{1}$$

$$\begin{aligned}\mathbb{E}\left[\theta|X=x\right] &= g_1(\mu,\sigma^2,x,d) \\ &= \mu-(-1)^x e^{-2(2d+1)^2\sigma^2}\frac{2\left(2d+1\right)\sigma^2\sin\left(2\left(2d+2\right)\mu\right)}{1+(-1)^x e^{-2(2d+1)^2\sigma^2}\cos\left(2\left(2d+1\right)\mu\right)}\end{aligned} \tag{2}$$

$$\begin{aligned}\mathrm{Var}(\theta|X=x) &= g_2(\mu,\sigma^2,x,d) \\ &= \sigma^2\left\{1-(-1)^x 4\left(2d+1\right)^2\sigma^2 e^{-2(2d+1)^2\sigma^2}\times\right. \\ &\qquad\left.\frac{\cos\left(2\left(2d+1\right)\mu\right)+(-1)^x e^{-2(2d+1)^2\sigma^2}}{\left(1+(-1)^x e^{-2(2d+1)^2\sigma^2}\cos\left(2\left(2d+1\right)\mu\right)\right)^2}\right\}\end{aligned} \tag{3}$$

2.1.3 Selecting next sample depth

The key advantage of the approach given in this work over the previous work, [], is the dynamic sampling scheme which utilises the current belief to minimise the work to obtain an estimate. To do this we look to maximise the information gain per sample or equivalently maximise the reduction in variance of our belief from each sample. The decision variable throughout is, the circuit depth d, for the next sample, i.e. the number of applications of the Grover operators.

To do this, we utilise the expected reduction in variance, see equation **??**. Given our current state, (μ,σ^2), we select the depth that minimises Equation 1. Note that the optimisation is over all non-negative integers. Rather than directly minimise this instead we look to maximise the variance reduction factor,

$$h(\mu,\sigma^2,d)=\mathcal{V}=\frac{4\left(2d+1\right)^2 e^{-4(2d+1)^2\sigma^2}\sin^2\left(2\left(2d+1\right)\mu\right)}{1-e^{-4(2d+1)^2\sigma^2}\cos^2\left(2\left(2d+1\right)\mu\right)} \tag{4}$$

which is equivalent to minimising the expected variance. See Figure **??** for an example of this function which is typical and integer points for which we are maximising over. There are a few features to note:

- The various reduction function is bounded by $4(2d+1)^2 e^{-4(2d+1)^2\sigma^2}$ which is maximised by $2(2d+1) = \frac{1}{\sigma}$ i.e., $d = O(\sigma^{-1})$. This reflects the scaling seen in the fixed strategy and in a number of works relating to quantum phase estimation [].
- The true maximum of integers might be a distance from the maximum of the bounding function due to the sinusoidal nature. Importantly, it is not necessarily either integer value which neighbours the bounding maximum.
- The issue with selecting the bounding maximum is partially observed in this work, []. The driving factor is the gradient of the probability function for the Bernoulli distribution, near probabilities of 0 or 1 this is small, but nearer $1/2$ this is maximised which improves the ability to discriminate different values of θ. This is something that can be clearly be observed in our experiments, see Section **??** and Figure **??**.

2.1.4 Adjustments for noise

When we incorporate decoherence noise, we incorporate an extra parameter λ which represents the noise level of the quantum computer. We assume in this work that the value of this parameter has been previously ascertained through other means. For reference, we repeat here the probability of measuring a value 1 as a function of θ and λ,

$$\mathbb{P}(X=1|\theta,\lambda) = \frac{1}{2}\left(1 - e^{-(2d+1)\lambda}\cos\left(2(2d+1)\theta\right)\right)$$

For more further details of this see Section **??**.

As mentioned previously, the basic sampling algorithm does not change, only the functions h, g_1, g_2 in order to accommodate the change in the measurement probability. These become

$$h(\mu,\sigma^2,d,\lambda) = \frac{4(2d+1)^2 e^{-2(2d+1)\lambda} e^{-4(2d+1)^2\sigma^2}\sin^2\left(2(2d+1)\mu\right)}{1 - e^{-2(2d+1)\lambda} e^{-4(2d+1)^2\sigma^2}\cos^2\left(2(2d+1)\mu\right)} \tag{5}$$

$$g_1(\mu,\sigma^2,x,d,\lambda) = \mu - (-1)^x e^{-(2d+1)\lambda} e^{-2(2d+1)^2\sigma^2}\frac{2(2d+1)\sigma^2\sin\left(2(2d+2)\mu\right)}{1 + (-1)^x e^{-(2d+1)\lambda} e^{-2(2d+1)^2\sigma^2}\cos\left(2(2d+1)\mu\right)} \tag{6}$$

$$g_2(\mu,\sigma^2,x,d,\lambda) = \sigma^2\left\{1 - (-1)^x 4(2d+1)^2\sigma^2 e^{-(2d+1)\lambda} e^{-2(2d+1)^2\sigma^2}\times \frac{\cos\left(2(2d+1)\mu\right) + (-1)^x e^{-(2d+1)\lambda} e^{-2(2d+1)^2\sigma^2}}{\left(1 + (-1)^x e^{-(2d+1)\lambda} e^{-2(2d+1)^2\sigma^2}\cos\left(2(2d+1)\mu\right)\right)^2}\right\} \tag{7}$$

Note that we recover the original functions by setting $\lambda = 0$. For the derivation see Appendix **??**.

2.1.5 Modelling the sampling scheme

It is useful to note that we can model the algorithm as a Markov process on $\mathbb{R}^2$ with the state as (μ, σ^2). We can see this from equations 2 and 3, since the state update only

depends on the current state, the depths which is again only a function of the current state and the random sample. This means that it obeys the Markov property, giving use a Markov process. If we look closely at equation 3 we note that the update is multiplicative in nature. By taking the log of the variance, $\nu = \log(\sigma^2)$, we change this to additive process which provides us with a number of insights.

Firstly, if we consider the expected decrease in ν by taking the log of Equation 1 at the optimal depth, we find this is $o(1)$ throughout, see Appendix C. This means that the number of circuit executions to reach an accuracy of order ϵ will be $-\log(\epsilon)$ and ν decreases linearly in the number of iterations. This is something we observe in the experimentations, see Section **??**.

Secondly, by consider ν we can see why we need to calculate analytically the change in variance rather than use a sampling scheme. It is well known that the sample variance is an unbiased estimator of the true variance, i.e. $\mathbb{E}\left[S^2\right] = \sigma^2$ where S^2 is the sample variance, which would indicate that we can use a sample from the posterior distribution to estimate the variance. Unfortunately, this is not true as the $\log(S^2)$ is not an unbiased estimator of $\log(\sigma^2)$, we have

$$\mathbb{E}\left[\log\left(S^2\right)\right] = \mathbb{E}\left[\log\left(\sigma^2\right)\right] + \kappa,$$

where $\kappa > 0$. For details, see Appendix B. This means that if the sample variance is used to estimate the variance of the posterior, we have a systematic drift hence the uncertainty is underestimated. This is not so problematic in the noiseless case as all steps of ν are order 1 but in the case with noise the step size decays with ν so the systematic drift dominates leading to under sampling.

Note: Can we prove something about optimality of greedy strategy given the maximum reduction factor and bounds on the value function? (we know bounded between inverse square and inverse)

2.2 Post processing

As mentioned previously, the use of the normal approximation for the posterior distribution can lead to a drift in the estimate of θ. For this reason, we carry out a post-processing step which utilises all the obtained samples by taking θ to be the maximum likelihood estimate (MLE) as described in []. There are two important points to note about this:

1. The use of the MLE in this case is not turning our back on the Bayesian approach in the rest of the paper. Here, assuming the prior distribution is uniformly distributed over the interval $[0, \pi]$, an uninformative prior, the mode of the posterior distribution, a standard point estimate in Bayesian statistics [], agrees with the MLE.
2. This approach is easily adapted to the case of decoherence noise. The likelihood is changed to incorporate the extra factor in a straightforward fashion.

ADD ALG AS A REMINDER

[jlt 6] Sure this is the right one?(I haven't thought about it)

[cdv 4] $1/\epsilon$, right?? Or we would have exponential speed-up over classical monte carlo.

[jlt 7] o(1) here too? Or whichever the correct version is

[jlt 8] Value function is phase 2 - unless you're being implicit

[cdv 5] I think we can make an argument that it's negligible, and that, if we wanted, we should be able to find the maximum analytically.

3 Experimentation

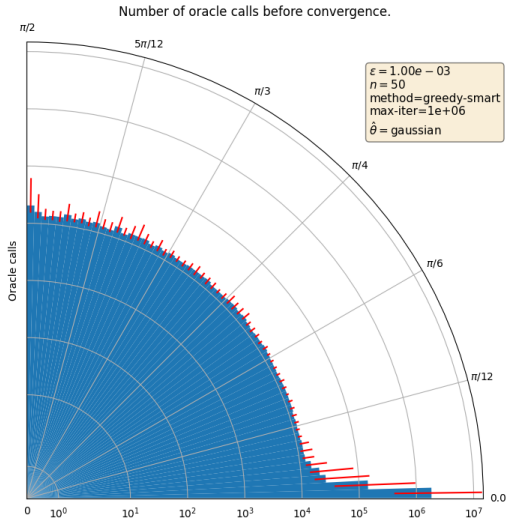
We conduct a series of experiments for the noiseless and noisy simulations of our algorithm in this section. We empirically compare against the IAE and MLAE algorithms. As the update steps for these algorithms are equivalent to sampling from a Bernoulli distribution for a given choice of θ_0 and noise level we sample directly from the Bernoulli distribution instead of simulating the dynamics of a quantum system for each algorithm. This algorithm has also been implemented and tested using Qiskit [14], with a view for future experiments with quantum simulators and quantum hardware.

[jlt 9] Need referencing either the actual papers, or just saying the acronyms earlier

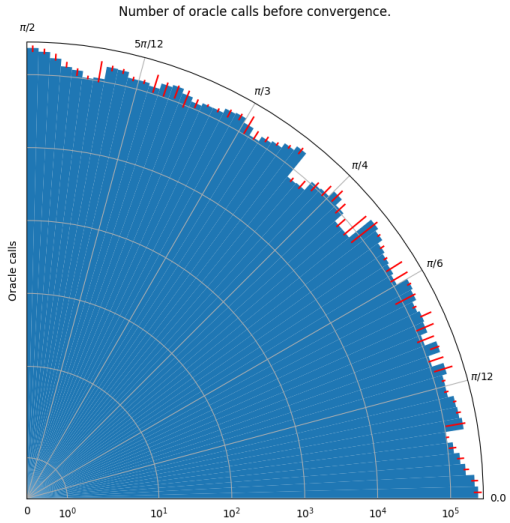
3.1 Simulation

3.1.1 Noiseless

First, we explore the convergence time for $\theta \in [0, \pi/2]$. As shown in [11], so-called exceptional points with poor convergence occur near to rational multiples of π, therefore we sample values of θ from continuous distributions instead of equally spaced angles. Figure 4a shows that the convergence for θ close to 0 is several orders of magnitude larger than convergence for other values. For this reason, we consider partitioning the space $\Theta = [0, \pi/2]$ into the central region $\Theta_0 = [\frac{\pi}{12}, \frac{5\pi}{12}]$ and the edge region $\Theta_1 = \Theta \setminus \Theta_0 = [0, \frac{\pi}{12})$. This is region of poor convergence is inherent to the use of likelihood-based sampling regimes, and is not shared by IAE (see 4b).



(a) Median number of oracle calls for RAE to converge to a precision of $\varepsilon = 10^{-3}$, with 25% and 75% quartiles shown. Each bar is a region of width $\pi/120$ with the median number of oracle calls calculated from 50 values of true value θ_0 selected uniformly at random. The prior for each iteration is taken to be $N(\tilde{\theta}_0, 1)$, where $\tilde{\theta}_0$ is sampled from a $N(\theta_0, 0.1)$ distribution and success probability $1 - \alpha$ with $\alpha = 0.01$.



(b) Median number of oracle calls for IAE to converge to a precision of $\varepsilon = 10^{-3}$, with 25% and 75% quartiles shown. Each bar is a region of width $\pi/120$ with the median number of oracle calls calculated from 50 values of true value θ_0 selected uniformly at random, and a success probability $1 - \alpha$ with $\alpha = 0.01$

(The following is all conjecture as we don't have a comparison graph.)
As seen in Figure **??**, the set Θ_1 performs poorly for all methods that take a statistical approach. This region contains points where the variance reduction factor approaches 0, with the extreme points where $\theta = 0, \pi$.

[cdv 6] Misses a caption of the 'full' figure.

[cdv 7] It's actually from 1000 values. I can change that either in the figure, or in the caption.

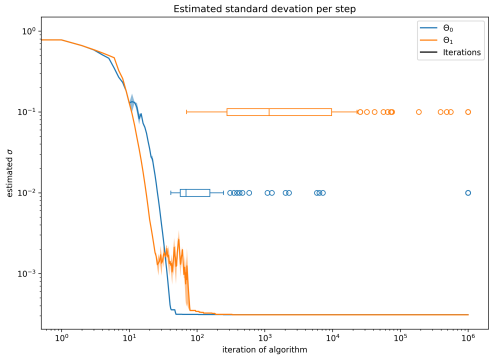


Figure 5: Median standard deviation for fixed error rate $\varepsilon = 10^{-3}$ against time step for $\theta_0 \in \Theta_0$ and $\theta_0 \in \Theta_1$. We sample θ_0 from a uniform distribution of x and 50 samples for Θ_0 and Θ_1 respectively. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1 - \alpha$ with $\alpha = 0.01$. The box plots show the Q1, Q2 and Q3 algorithm termination times for the recorded runs.

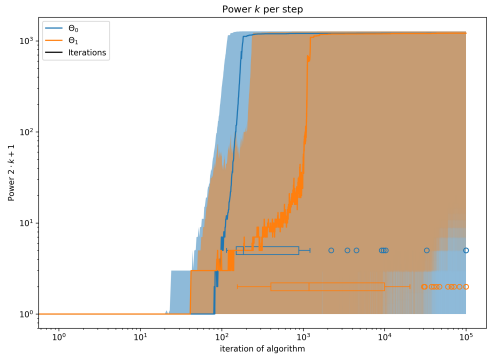


Figure 6: Median depth for fixed error rate $\varepsilon = 10^{-3}$ against time step for $\theta_0 \in \Theta_0$ and $\theta_0 \in \Theta_1$. We sample θ_0 from a uniform distribution of x and 50 samples for Θ_0 and Θ_1 respectively. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1 - \alpha$ with $\alpha = 0.01$. The box plots show the Q1, Q2 and Q3 algorithm termination times for the recorded runs.

From now on, we restrict to considering $\theta \in \Theta_0$.

A source of potential error in this algorithm is the normal approximation made at each step, which removes the theoretical guarantees of the Bernstein-von-Mises theorem. This can be seen in Figure 7, where should expect $(1-\alpha)\%$ of the points to lie above the diagonal. As a mitigation technique, we post-process the measurement data using the MLE estimator, and recover some of the desired accuracy.

[jlt 10] This may need references to earlier in the algorithm section

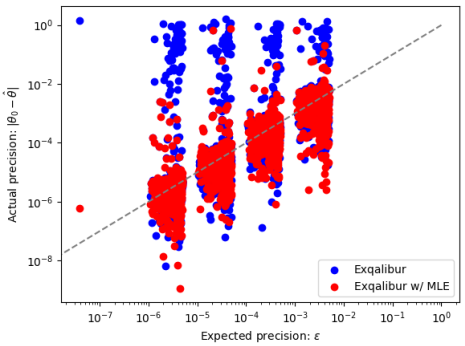


Figure 7: Actual precision of the final estimate for RAE with and without MLE post-processing. We sample $\theta_0 \in \Theta_0$ from a uniform distribution of x and 50 for target precisions of $\varepsilon = 10^{-3}, 10^{-4}, \ldots, 10^{-7}$. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1-\alpha$ with $\alpha = 0.01$.

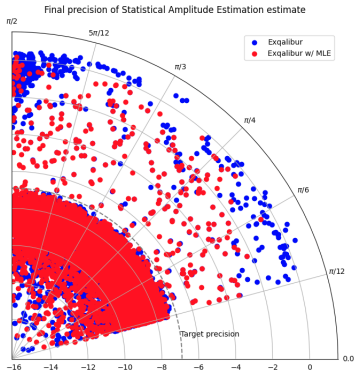


Figure 8: Actual precision of the final estimate for RAE with and without MLE post-processing. We sample $\theta_0 \in \Theta_0$ from a uniform distribution of x and 500 for a target precision of $\varepsilon = 10^{-3}$. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1-\alpha$ with $\alpha = 0.01$.

We compare the performance of RAE to IAE and MLAE, and observe that in the noiseless case we achieve similar numbers of oracle queries to IAE. However, there still remain some points for which RAE fails to satisfactorily converge.

For the RAE routine, we also calculate the expected and exact probability of the Bernoulli sample at each stage. 11 is a typical example of a RAE run, where the probability is initially expected to be 0 for a substantial amount of time, before approaching $\frac{1}{2}$ for the remainder of the run.

3.1.2 Decohering Noise

We explore the performance of our algorithm for a range of noise rates.

As shown in Figure 12, the algorithm will not go beyond a depth of $\sim \frac{1}{\lambda}$.

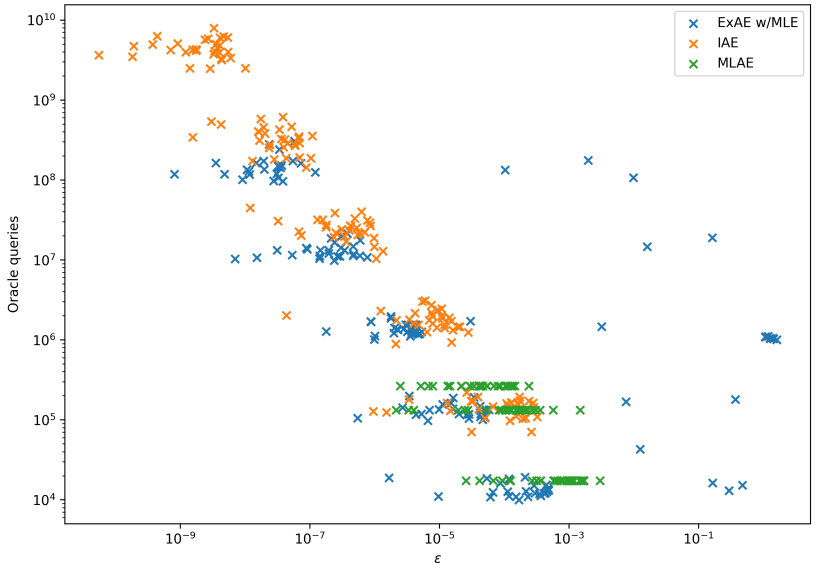


Figure 9: Actual precision versus total number of oracle queries used for IAE, MLAE and RAE. We target precisions of $\varepsilon = 10^{-3}, 10^{-4}, \ldots, 10^{-7}$, with each point a single value of $\theta_0 \in \Theta_0$ and take the final precision to be half the width of the confidence interval. We sample uniformly from Θ_0 for 30 values of θ_0. Each value of θ_0 is evaluated for all algorithms and each target ε. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1 - \alpha$ with $\alpha = 0.01$.

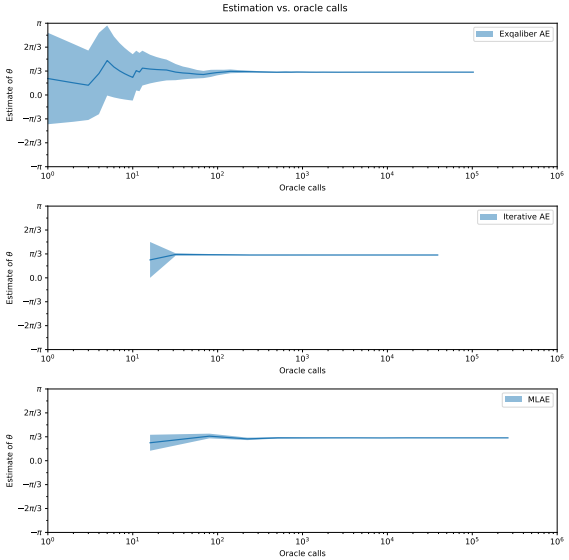


Figure 10: Confidence intervals for a single run of RAE, IAE and MLAE with $\theta_0 = 1$. The prior for RAE is taken to be $N(\pi/2, 1)$ and success probability $1 - \alpha$ with $\alpha = 0.01$. As IAE and MLAE use a fixed number of shots per step, the number of oracle calls does not start at 0.

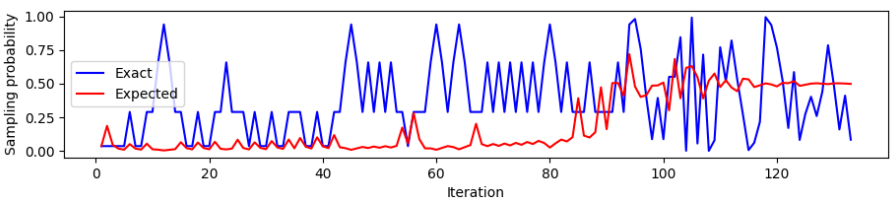


Figure 11: Exact and expected probability calculation for a single run of RAE.

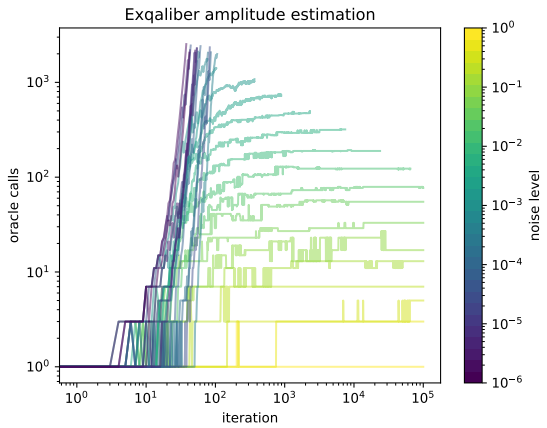


Figure 12: Depth of RAE for $\theta_0 = 1$ at varying levels of decohering noise characterised by $\lambda = 10^0, 10^{-1}, \ldots 10^{-6}$. The prior for each iteration is taken to be $N(\pi/2, 1)$, we target $\varepsilon = 10^{-3}$ and success probability $1 - \alpha$ with $\alpha = 0.01$.

Figure 13: Total number of oracle queries for RAE with decohering noise characterised by $\lambda = 10^{-1}, 10^{-2}, \ldots, 10^{-7}$. Each point is a single value of $\theta_0 \in \Theta_0$. We sample uniformly from Θ_0 for 30 values of θ_0. Each value of θ_0 is evaluated for all decohering noise values and each target ε. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1 - \alpha$ with $\alpha = 0.01$.

- Concerned that error comparisons make no sense here. These algorithms have no capacity to cope with noise so obviously they're bad - but others e.g. Hitachi, QoPrime, Power-law AE do

Figure 14: Final precision versus total number of oracle queries used for IAE, MLAE and RAE with decohering noise characterised by $\lambda = 10^{-3}$. We target precisions of $\varepsilon = 10^{-3}, 10^{-4}, \ldots, 10^{-7}$, with each point a single value of $\theta_0 \in \Theta_0$. We sample uniformly from Θ_0 for 30 values of θ_0. Each value of θ_0 is evaluated for all algorithms and each target ε. The prior for each iteration is taken to be $N(\theta_0, 1)$ and success probability $1 - \alpha$ with $\alpha = 0.01$.

4 Discussion

TO BE ADDED

- Summary what we have done
- The potential incorporating for error mitigation and error correction
- Wider noise models
- How can we further bring down the requirements to accelerate?
- How can it be used for QPE style

References

[1] Yohichi Suzuki, Shumpei Uno, Rudy Raymond, Tomoki Tanaka, Tamiya Onodera, and Naoki Yamamoto. "Amplitude estimation without phase estimation". Quantum Information Processing **19**, 1–17 (2020).

[2] Joseph G. Smith, Crispin H. W. Barnes, and David R. M. Arvidsson-Shukur. "An adaptive bayesian quantum algorithm for phase estimation" (2023). arXiv:2303.01517.

[3] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. "Quantum amplitude amplification and estimation". Contemporary Mathematics **305**, 53–74 (2002).

[4] A Yu Kitaev. "Quantum measurements and the abelian stabilizer problem" (1995). arXiv:quant-ph/9511026.

[5] Don Coppersmith. "An approximate fourier transform useful in quantum factoring" (2002). arXiv:quant-ph/0201067.

[6] Scott Aaronson and Patrick Rall. "Quantum approximate counting, simplified". Pages 24–32. Society for Industrial and Applied Mathematics. (2020). arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611976014.5.

[7] Chu-Ryang Wie. "Simpler quantum counting" (2019). arXiv:1907.08119.

[8] Paul Dagum, Richard Karp, Michael Luby, and Sheldon Ross. "An optimal algorithm for monte carlo estimation". SIAM Journal on computing **29**, 1484–1496 (2000). arXiv:https://doi.org/10.1137/S0097539797315306.

[9] Stefan Heinrich. "Quantum summation with an application to integration". Journal of Complexity **18**, 1–50 (2002).

[10] Ashley Montanaro. "Quantum speedup of monte carlo methods". Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **471**, 20150301 (2015). arXiv:https://royalsocietypublishing.org/doi/pdf/10.1098/rspa.2015.0301.

[11] Adam Callison and Dan E. Browne. "Improved maximum-likelihood quantum amplitude estimation" (2022). arXiv:2209.03321.

[12] Dmitry Grinko, Julien Gacon, Christa Zoufal, and Stefan Woerner. "Iterative quantum amplitude estimation". npj Quantum Information **7**, 52 (2021). arXiv:https://doi.org/10.1038/s41534-021-00379-1.

[13] Tudor Giurgica-Tiron, Iordanis Kerenidis, Farrokh Labib, Anupam Prakash, and William Zeng. "Low depth algorithms for quantum amplitude estimation". Quantum **6**, 745 (2022). arXiv:https://doi.org/10.22331/q-2022-06-27-745.

[14] Qiskit contributors. "Qiskit: An open-source framework for quantum computing" (2023).

A Posterior statistics

In this appendix, we calculate all statistics of interest for the posterior distribution.

A.1 Noiseless

In the noiseless case, we consider a prior distribution $\pi(\theta) \sim N(\mu, \sigma^2)$ and our sampling distribution $X \sim \text{Ber}(\frac{1}{2}(1 - \cos((4d+2)\theta))$. First, let us calculate all the moments of the posterior distribution, given the measurement:

$$\mathbb{E}\left[\theta^m | X = x\right] = \frac{\int_{-\infty}^{\infty} \theta^m \left(1 + (-1)^x \cos\left(2\left(2d+1\right)\theta\right)\right) e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} \mathrm{d}\theta}{\int_{-\infty}^{\infty} \left(1 + (-1)^x \cos\left(2\left(2d+1\right)\theta\right)\right) e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} \mathrm{d}\theta}.$$

Thus, the integrals of interest take the form.

$$I_m(\theta) = \int_{-\infty}^{\infty} \theta^m \left(1 + (-1)^x \cos\left(2\left(2d+1\right)\theta\right)\right) e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} \mathrm{d}\theta.$$

For the following, we're going to define the Fourier transform $\hat{f}(\omega)$ to be given by

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} \mathrm{d}x = \mathcal{F}\left\{f(x)\right\},$$

for some function $f(x)$. We also recall the following facts about the Fourier transform:

$$\mathcal{F}\left\{f(x-a)\right\} \mapsto e^{-ia\omega}\hat{f}(\omega) \tag{8}$$

$$\mathcal{F}\left\{f(ax)\right\} \mapsto \frac{1}{|a|}\hat{f}(\frac{\omega}{a}) \tag{9}$$

$$\mathcal{F}\left\{x^n f(x)\right\} \mapsto i^n \frac{\mathrm{d}^n \hat{f}(\omega)}{\mathrm{d}\omega^n} \tag{10}$$

$$\mathcal{F}\left\{e^{-x^2}\right\} = \sqrt{\pi} e^{-\frac{\omega^2}{4}}. \tag{11}$$

We can now rewrite $I_m(\theta)$ to be in a suitable Fourier form,

$$I_m(\theta) = \sqrt{2\pi\sigma^2}\mathbb{E}_{\pi(\theta)}\left[\theta^m\right] + \frac{(-1)^x}{2}\int_{-\infty}^{\infty} \theta^m \left(e^{2(2d+1)i\theta} + e^{-2(2d+1)i\theta}\right) e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} \mathrm{d}\theta.$$

Consider

$$\int_{-\infty}^{\infty} e^{-\frac{\theta^2}{2\sigma^2}} e^{-i\omega\theta} \mathrm{d}\theta = \sqrt{2\pi\sigma^2} e^{-\frac{\sigma^2\omega^2}{2}}, \qquad \text{by (11) and (10), taking } a = \frac{1}{\sqrt{2\sigma^2}},$$

$$\int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} e^{-i\omega\theta} \mathrm{d}\theta = \sqrt{2\pi\sigma^2} e^{-i\mu\omega} e^{-\frac{\sigma^2\omega^2}{2}}, \qquad \text{by (8), taking } a = \mu,$$

$$\int_{-\infty}^{\infty} \theta^m e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} e^{-i\omega\theta} \mathrm{d}\theta = i^m \frac{\mathrm{d}^m}{\mathrm{d}\omega^m}\left(\sqrt{2\pi\sigma^2} e^{-i\mu\omega} e^{-\frac{\sigma^2\omega^2}{2}}\right), \qquad \text{by (10)},$$

$$\int_{-\infty}^{\infty} \theta^m e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} e^{i\omega\theta} \mathrm{d}\theta = (-i)^m \frac{\mathrm{d}^m}{\mathrm{d}\omega^m}\left(\sqrt{2\pi\sigma^2} e^{i\mu\omega} e^{-\frac{\sigma^2\omega^2}{2}}\right), \qquad \text{by } \omega \mapsto -\omega.$$

From which we conclude

$$I_m(\theta) = \sqrt{2\pi\sigma^2}\left[\mathbb{E}_{\pi(\theta)}\left[\theta^m\right] + (-1)^x \left.\frac{\mathrm{d}^m}{\mathrm{d}\omega^m}\right|_{\omega=2(2d+1)} \left(\frac{e^{i\mu\omega} + (-1)^m e^{-i\mu\omega}}{2i^m} e^{-\frac{\sigma^2\omega^2}{2}}\right)\right].$$

In particular,

$$I_0(\theta) = \sqrt{2\pi\sigma^2}\left(1 + (-1)^x e^{-2(2d+1)^2\sigma^2} \cos\left(2\left(2d+1\right)\mu\right)\right]$$

$$\begin{aligned} I_1(\theta) &= \sqrt{2\pi\sigma^2}\left[\mu + (-1)^x \left.\frac{\mathrm{d}}{\mathrm{d}\omega}\right|_{\omega=2(2d+1)} \left(\sin\left(\mu\omega\right) e^{-\frac{\sigma^2\omega^2}{2}}\right)\right] \\ &= \sqrt{2\pi\sigma^2}\left[\mu + (-1)^x e^{-2(2d+1)^2\sigma^2}\left(\mu\cos\left(2\left(2d+1\right)\mu\right) - 2\left(2d+1\right)\sigma^2 \sin\left(2\left(2d+1\right)\mu\right)\right)\right] \end{aligned}$$

$$\begin{aligned} I_2(\theta) &= \sqrt{2\pi\sigma^2}\left[\sigma^2 + \mu^2 - (-1)^x \left.\frac{\mathrm{d}^2}{\mathrm{d}\omega^2}\right|_{\omega=2(2d+1)} \left(\cos\left(\mu\omega\right) e^{-\frac{\sigma^2\omega^2}{2}}\right)\right] \\ &= \sqrt{2\pi\sigma^2}\left[\sigma^2 + \mu^2 + (-1)^x e^{-2(2d+1)^2\sigma^2}\left(\left(\mu^2 + \sigma^2 - 4\left(2d+1\right)^2\sigma^4\right)\cos\left(2\left(2d+1\right)\mu\right)\right.\right. \\ &\qquad \left.\left. -4\left(2d+1\right)\mu\sigma^2 \sin\left(2\left(2d+1\right)\mu\right)\right)\right] \end{aligned}$$

Hence

$$\mathbb{E}\left[\theta|X=x\right]=\mu-\left(-1\right)^{x}e^{-2\left(2d+1\right)^{2}\sigma^{2}}\frac{2\left(2d+1\right)\sigma^{2}\sin\left(2\left(2d+1\right)\mu\right)}{1+\left(-1\right)^{x}e^{-2\left(2d+1\right)^{2}\sigma^{2}}\cos\left(2\left(2d+1\right)\mu\right)} \tag{12}$$

$$\mathrm{Var}\left(\theta|X=x\right)=\sigma^{2}\left(1-\left(-1\right)^{x}4\left(2d+1\right)^{2}\sigma^{2}e^{-2\left(2d+1\right)^{2}\sigma^{2}}\frac{\cos\left(2\left(2d+1\right)\mu\right)+\left(-1\right)^{x}e^{-2\left(2d+1\right)^{2}\sigma^{2}}}{\left(1+\left(-1\right)^{x}e^{-2\left(2d+1\right)^{2}\sigma^{2}}\cos\left(2\left(2d+1\right)\mu\right)\right)^{2}}\right) \tag{13}$$

$$=\sigma^{2}\left(1-\sigma^{2}\mathcal{V}_{x}\right), \tag{14}$$

where we define

$$\mathcal{V}_{x}=\left(-1\right)^{x}4\left(2d+1\right)^{2}e^{-2\left(2d+1\right)^{2}\sigma^{2}}\frac{\cos\left(2\left(2d+1\right)\mu\right)+\left(-1\right)^{x}e^{-2\left(2d+1\right)^{2}\sigma^{2}}}{\left(1+\left(-1\right)^{x}e^{-2\left(2d+1\right)^{2}\sigma^{2}}\cos\left(2\left(2d+1\right)\mu\right)\right)^{2}}.$$

If we note that $\mathbb{P}\left(X=x\right)=\frac{I_{0}(\theta,x)}{2\sqrt{2\pi\sigma^{2}}}$, then we recover $\mathbb{E}[\theta]=\mu$. Similarly, we obtain

$$\mathrm{Var}\left(\theta\right)=\sigma^{2}\left(1-\sigma^{2}\mathcal{V}\right),$$

where

$$\begin{aligned}\mathcal{V}&=\mathbb{P}\left(X=0\right)\mathcal{V}_{0}+\mathbb{P}\left(X=1\right)\mathcal{V}_{1}\\&=\frac{2\left(2d+1\right)^{2}e^{-2\left(2d+1\right)^{2}\sigma^{2}}}{1-e^{-4\left(2d+1\right)^{2}\sigma^{2}}\cos\left(2\left(2d+1\right)\mu\right)}\times\\&\quad\left\{\left(\cos\left(2\left(2d+1\right)\mu\right)+e^{-2\left(2d+1\right)^{2}\sigma^{2}}\right)\left(1-e^{-2\left(2d+1\right)^{2}\sigma^{2}}\cos\left(2\left(2d+1\right)\mu\right)\right)\right.\\&\quad\left.-\left(\cos\left(2\left(2d+1\right)\mu\right)-e^{-2\left(2d+1\right)^{2}\sigma^{2}}\right)\left(1+e^{-2\left(2d+1\right)^{2}\sigma^{2}}\cos\left(2\left(2d+1\right)\mu\right)\right)\right\}\\&=\frac{4\left(2d+1\right)^{2}e^{-4\left(2d+1\right)^{2}\sigma^{2}}\sin^{2}\left(2\left(2d+1\right)\mu\right)}{1-e^{-4\left(2d+1\right)^{2}\sigma^{2}}\cos^{2}\left(2\left(2d+1\right)\mu\right)}\end{aligned}$$

A.2 Noisy

In the noisy case, we consider the use of a depolarising quantum channel $\Phi\left(\cdot\right)$ with depolarising probability p. For any quantum state ρ, we have

$$\Phi\left(\rho\right)=\left(1-p\right)\rho+p\frac{\rho}{2^{n}}I,\quad 0\leq p\leq 1+\frac{1}{2^{2n}-1},$$

where I is the density matrix corresponding to the maximally mixed state on 2^{n} qubits. As the most error-prone part of the quantum circuit is the use of the state preparation routine $\mathcal{A}$, we consider only the probability that the state is properly prepared after each application of $\mathcal{A}$

$$\prod_{i}\left(1-p_{i}\right)\approx e^{-\lambda},\quad\text{for }\lambda=\sum_{i}p_{i},$$

for $\lambda\sim 1$ by a Poisson approximation, where p_{i} is the depolarising probability for each gate A_{i} applied. Hence, we approximate the probability that we generate $\mathcal{U}^{d}\mathcal{A}\left|0\right\rangle^{\otimes n}$ as $e^{-(2d+1)\lambda}$ and generate the maximally mixed state otherwise. Therefore,

$$\mathbb{P}\left(X=x|\,\theta,\lambda\right)=\frac{1}{2}\left(1-\left(-1\right)^{x}e^{-(2d+1)\lambda}\cos\left(2\left(2d+1\right)\theta\right)\right).$$

We note that this exponentially suppress the signal we get from our angle θ