Statistical Phase Estimation on PennyLane
Team: Kuantum

Anjishnu Bose
Department of Physics, University of Toronto.[*]

Praveen Jayakumar
Department of Chemistry, University of Toronto.[*]

Serene Shum
Department of Physics, University of Toronto. [*]

I. INTRODUCTION

Quantum phase estimation is a method of estimating the phase of an eigenvalue of a unitary operator, assuming controlled access to the unitary [1]. It forms a subroutine in many other algorithms, such as Shor's algorithm and the HHL algorithm. Additionally, it can be used for ground state energy estimation, an important problem in quantum chemistry. However, it requires a large number of ancilla qubits, long circuit depths, and the ability to perform many controlled operations. For these reasons, the standard quantum phase estimation algorithm is not well-suited for current hardware.

Several papers have proposed statistical versions of phase estimation, where measurement samples taken from short, simple quantum circuits are post-processed classically in order to estimate the eigenvalues of the operator [2]. This offloads a large part of the computation to a classical computer, and reduces the circuit depth for individual circuits at the cost of running a higher number of these short circuits. The circuit depth can be further reduced by randomly sampling the terms in the unitary decomposition of the time evolution operator instead of implementing the full Hamiltonian evolution.

This project implements the statistical phase estimation procedure of Lin and Tong [3] and the randomized statistical phase estimation procedure of Wan, Berta, and Campbell [4, 5] in PennyLane, along with the modified binary search procedure from Lin and Tong [3] in order to estimate the ground state energy of a given Hamiltonian.

II. THEORY

A. Estimating eigenvalues with cummulative distribution functions

Problem statement: Given a Hamiltonian that can be expressed in the form

$$H = \sum_{i=0}^{K-1} \lambda_i \Pi_i \tag{1}$$

[*] These authors contributed equally :).

where $\{\lambda_i\}$ are the eigenvalues of H arranged in ascending order ($\lambda_i < \lambda_{i+1}$) and $\{\Pi_i\}$ are projectors onto the corresponding eigenspaces, the problem of ground state energy estimation is to find λ_0 up to some additive precision Δ. This problem is QMA-hard in general, so we need additional information about the system in order to be able to solve it efficiently.

The phase estimation method we consider here to solve this problem consists of preparing an initial guess state ρ, over which we apply a controlled time evolution of the Hamiltonian and perform quantum measurements to perform a Hadamard test and estimate the ground state. We now elaborate this approach below.

Firstly, we will assume that we have some input state ρ that has some overlap with the ground state. Let

$$p_k = \mathrm{Tr}(\rho\Pi_{\mathrm{k}}), \tag{2}$$

be the overlap of the prepared state with the $(k+1)^{\mathrm{st}}$ eigenspace. Ideally, we want p_0 to be large for our method to be successful. We do not need to know p_0 prior to the measurements, but we assume a lower bound of η ($p_0 \geq \eta$). For small molecular systems with low orbital correlations, the Hartree-Fock state suffices for ρ.

Define a scaling parameter

$$\tau := \frac{\pi}{2\lambda + \Delta} \tag{3}$$

chosen to bound the spectral range and restrict the eigenspectrum of the Hamiltonian τH to $[-\pi/2, \pi/2]$. Here λ is the sum of the absolute values of the coefficients in the Pauli decomposition of H and Δ is a precision parameter chosen as required.

Let $p(y)$ be the probability distribution function (PDF) of the eigenspectrum of τH associated with ρ, defined as

$$p(y) = \sum_i p_i \delta(y - \tau\lambda_i), \tag{4}$$

where δ is the Dirac delta distribution. The PDF consists of peaks at the various eigenvalues of τH, with coefficients on the Dirac deltas determined by the overlap of the state with the corresponding eigenvalue. Then, the periodic CDF associated with the Hamiltonian and the state ρ is

$$C(x) = \int_{-\pi/2}^{\pi/2} p(y)\Theta(x-y)dy \tag{5}$$

$C(x)$ is a convolution of the PDF with the 2π-periodic heaviside function Θ defined as

$$\Theta(x) = \begin{cases} 1, & 2p\pi \leq x < 2p\pi + \pi \\ 0, & 2p\pi - \pi \leq x < 2p\pi \end{cases} \quad \text{where } p \in \mathbb{Z} \tag{6}$$

The CDF has jump discontinuities at values that correspond to the eigenvalues of τH. By approximating the CDF and finding where the first jump occurs, we can estimate the ground state energy of the Hamiltonian. We note that finding other jump positions can provide estimates of excited states, but is beyond the scope of our work. Figure 1 illustrates the CDF.

In this algorithm, we use a Hadamard test circuit to sample and reconstruct an approximated version $\tilde{C}(x)$ of the CDF $C(x)$, and then the ground state energy is estimated from the approximate CDF using a modified binary search procedure. Following [4], considering only finite points S_1 in the Fourier series of Θ, we approximate the CDF as

$$\tilde{C}(x) = \sum_{j\in S_1} F_j e^{ijx} tr(\rho e^{iHt_j/\lambda}) \tag{7}$$

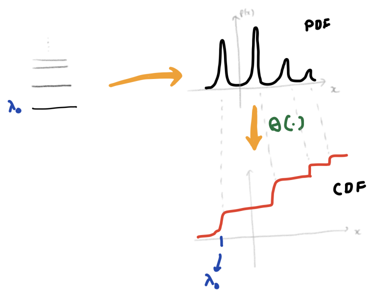


FIG. 1: Cumulative distribution function (CDF) is obtained by convolution of the PDF with the periodic Heaviside function Θ. The positions of the jumps are determined by the eigenspectrum of τH.

where $t_j = -j\tau\lambda$ and F_j and the Fourier coefficients. The approximate CDF that this algorithm aims to reconstruct, $\tilde{C}(x)$, satisfies

$$C(x-\delta) - \epsilon \leq \tilde{C}(x) \leq C(x+\delta) + \epsilon \tag{8}$$

for imprecision threshold $\epsilon > 0$. Since S_1 is finite, the Fourier series approximation is accurate up to ϵ for $x \notin (-\delta, \delta)$ and thus in Eq. (8) we allow for $\pm\delta$ error in x. We will follow the methods in [4] (proofs will not be provided here, see original paper for more details). First, we define the approximate Heaviside function,

$$F(x) = \sum_{k \leq N} F_k e^{ikx} \tag{9}$$

where

$$F_k = \begin{cases} \frac{1}{2} & k = 0 \\ -i\sqrt{\frac{\beta}{2\pi}} e^{-\beta} \frac{I_j(\beta)+I_{j+1}(\beta)}{2j+1} & k = 2j+1 \\ i\sqrt{\frac{\beta}{2\pi}} e^{-\beta} \frac{I_d(\beta)}{2j+1} & k = 2d+1 \end{cases} \tag{10}$$

where $0 \leq j \leq d-1$, $N = 2d+1$, $F_{-k} = -F_k$ for $k \neq 0$, $I_n(\beta)$ is the nth modified Bessel function of the first kind. The user determines the values of β and N; the Heaviside approximation gets better with increasing β, but the $|F_k|$ decay more slowly with increasing β, leading to a higher value of N needed. Thus, there is a tradeoff that needs to be made between the two.

Then, the approximate CDF is

$$\tilde{C}(x) = \int_{-\pi/2}^{\pi/2} p(y) F(x-y) dy \tag{11}$$

$$= \sum_j F_j e^{ijx} \mathrm{Tr}[\rho e^{iHt_j}] \tag{12}$$

$\mathrm{Tr}[\rho e^{iHt_j/\lambda}]$ can be computed using the expectation value of a Hadamard test circuit. Each iteration of the circuit requires one controlled implementation of the Hamiltonian evolution operator $e^{iHt_j/\lambda}$. This can be done using any standard Hamiltonian simulation methods such as Trotterization or qDRIFT. For larger, more correlated systems, the Hamiltonian contains a large number of terms. This would require a large number of quantum gate operations to simulate. We adopt a statistical method proposed in Ref. 4, where instead of deterministically implementing the entire evolution, the authors propose a new way of randomly sampling terms in the unitary decomposition of the time evolution operator. This cuts down on the gate cost and makes this algorithm better suited to near-term quantum computers. We now describe this method. Figure 5

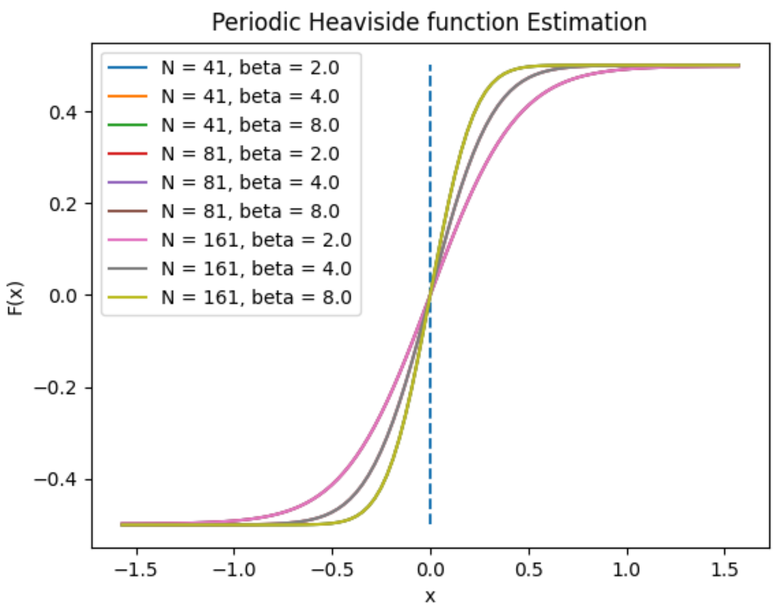


FIG. 2: Reconstructing the periodic heaviside function from its approximation in 10 for different values of $N = k_{max}$ and β.

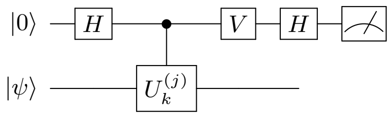


FIG. 3: Hadamard test circuit. The gate V is either $\mathbb{1}$ or $S^\dagger$ depending on whether the real or imaginary part of $\langle\psi|U_k^{(j)}|\psi\rangle$ is being estimated; the expectation value of measurement outcomes is $\frac{1}{2} + \frac{1}{2}\mathrm{Re}(\langle\psi|U_k^{(j)}|\psi\rangle)$ if the gate V is the identity and $\frac{1}{2} + \frac{1}{2}\mathrm{Im}(\langle\psi|U_k^{(j)}|\psi\rangle)$ if V is $S^\dagger$.

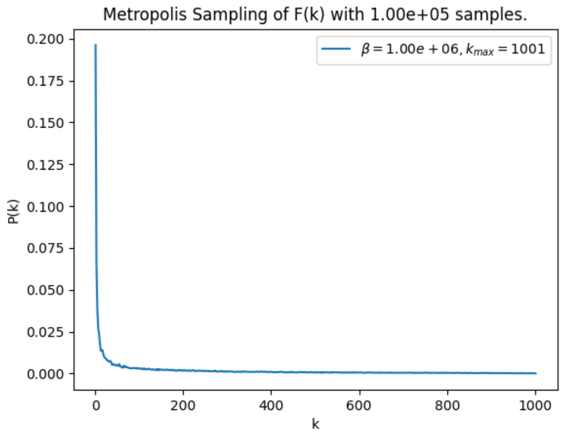


FIG. 4: Sampling odd integers k with probability proportional to $P(k) \propto |F(k; k_{max}, \beta)|$. The larger k values are sampled exponentially less.

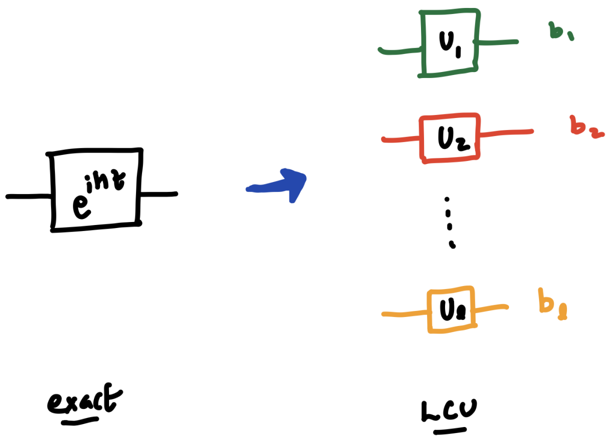


FIG. 5: LCU for Hamiltonian evolution e^{iHt}. b_k is the probability that each U_k will be randomly sampled.

B. Linear Combination of Unitaries for Hamiltonian Time Evolution

In our work, we consider the molecular electronic Hamiltonian

$$H = \sum_{pq} h_{pq} a_p^\dagger a_q + \sum_{pqrs} g_{pq,rs} a_p^\dagger a_q a_r^\dagger a_s \tag{13}$$

expressed in terms of fermionic creation $a_p^\dagger$ and annihilation a_p operators. Using a suitable fermion-qubit transform such as Jordan-Wigner or Bravyi-Kitaev, the fermionic Hamiltonian in Eq. (13) can be rewritten as a linear combination of Pauli words (on PennyLane, we can do this with the `pauli_decompose` function.)

$$H = \sum_{l=1}^{L} \alpha_l P_l \tag{14}$$

In Ref. 4, the authors show that by dividing the time of evolution to small time steps and Taylor-expanding the time evolution operator for each time step, we can rewrite the approximate CDF as

$$\tilde{C}(x) = \sum_{j,k} F_j e^{ijx} b_k^{(j)} \mathrm{Tr}[\rho U_k^{(j)}] \tag{15}$$

where each $U_k^{(j)}$ is a product of Pauli operators and Pauli rotations sampled from the terms in (14). Now, by sampling both j and k values, with probability proportional to $|F_j e^{ijx} b_k^{(j)}|$ and implementing the Hadamard test circuit shown in 3, we can approximately reconstruct $\tilde{C}(x)$. These circuits are controlled on the ancilla Hadamard test qubit and may consist a large number of Pauli words and Pauli rotations. Thus we optimize $U_k^{(j)}$ to reduce the circuit depth. Specifically, the operator is a product of r_j operators that each correspond to sampling one term in the Taylor expansion of the time evolution operator, each of which has the form:

$$W = P_{\ell_n} P_{\ell_{n-1}} \dots P_{\ell_1} e^{i\theta_n P_{\ell_0}} \tag{16}$$

$$\Rightarrow U_k^{(j)} = W_{r_j} W_{r_j - 1} \dots W_1 \tag{17}$$

where each P_ℓ is a Pauli operator. The n values (corresponding to the order of the Taylor expansion term taken) are randomly sampled according to their amplitude given by their Taylor expansion coefficients, q_n.

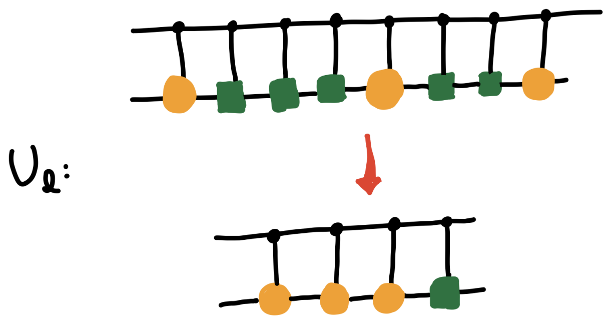


FIG. 6: Simplification of LCU. The squares are Pauli operations and the circles are Pauli rotations.

Thus, the sequence of operations consists of n_1 Pauli operations, one Pauli rotation, n_2 Pauli operations, one Pauli rotation, and so on. Pauli words and rotations have the following properties:

$$P_l P_m = sgn \cdot P_n \tag{18}$$

$$P_l e^{i\theta P_m} = \begin{cases} e^{i\theta P_m} P_l & [P_l, P_m] = 0 \\ e^{-i\theta P_m} P_l & \{P_l, P_m\} = 0 \end{cases} \tag{19}$$

where $sgn \in \{\pm 1, \pm i\}$ depending on P_l, P_m. Using these properties we can move all the Pauli words to the left of the rotations and simplify the circuit to the form

$$W^{(j)} = W_{r_j} W_{r_j - 1} \dots W_1 = P^{(j)} e^{i\theta P_{r_j}} e^{i\theta P_{r_j - 1}} \dots e^{i\theta P_1}. \tag{20}$$

Note that in our code we also multiply the Pauli rotations together because we were not running this on an actual quantum computer, so it was easier to write the circuit with only one Pauli rotation, and it would not make any difference since everything was calculated on a classical computer. However, if running this on quantum hardware, it would be beneficial to optimize the compilation of the Pauli rotations into gates before building the circuit. Figure 6 illustrates the circuit simplification.

C. Modified Binary Search

Finally, once we have the samples from the quantum circuits, we can apply the modified binary search procedure from [3] to find the x value where the first jump occurs, which we will call $x^\star$. The problem boils down to checking whether a guess is to the left or the right of the jump, and then moving to a new guess based on that result. However, we don't have access to the exact CDF, only a noisy sampled version of the approximate CDF. Instead of just reconstructing one ACDF $\tilde{C}(x)$ out of the circuit measurement samples, we will divide all our samples into N_b batches and implement a majority vote procedure on the batches to determine with high probability whether a chosen x point is to the left or the right of the jump, and then choose a new point to check accordingly.

First, we will begin by choosing our endpoints $x_0 = -\frac{\pi}{2}$, $x_1 = \frac{\pi}{2}$ and taking x to be the midpoint between them, $x = \frac{x_0 + x_1}{2}$. Then, in order to check whether x is to the left or to the right of $x^\star$, we will reconstruct a $\tilde{C}(x)$ out of each of the N_b batches of circuit measurement samples, and check whether most of these sampled $\tilde{C}(x)$ reconstructions give $\tilde{C}(x) > \frac{3\eta}{4}$. If more than half of the $\tilde{C}(x)$ approximations give $\tilde{C}(x) > \frac{3\eta}{4}$, we assume that x is to the right of $x^\star$, and we move x_1 to $x + \frac{2\delta}{3}$; in the other case, we move x_0 to $x - \frac{2\delta}{3}$.

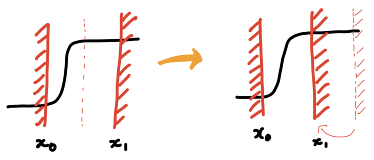


FIG. 7: A modified binary search based on majority vote is utilized to determine the position of the jump in the CDF.

Then, we take the new value of x as the midpoint of x_0 and x_1 and keep repeating the process until the size of the search window $x_0 - x_1$ is smaller than 2δ, and take our estimate of $x^\star$ to be $\frac{x_0+x_1}{2}$. See [3] for proof of how this gives the value of $x^\star$ to within δ error with high probability.

III. IMPLEMENTATION

The implementation can be found on the GitHub repo at https://github.com/Anjishnubose/Kuantum. In our implementation, all random sampling is done through a simple Metropolis Monte-Carlo algorithm with the appropriate probability distribution. This includes k: the Fourier series of the approximate Heaviside function, and n: the order of the LCU Taylor expansion. We let the Metropolis *thermalize* (converge to the desired probability distribution) for some iterations before starting to sample. All the circuit simulation and shot sampling is done on PennyLane using the appropriate parameters sampled during the Monte Carlo. From the circuit measurements, we construct an estimate of the CDF, and perform a modified binary search to get an estimate of the ground state energy.

A. Required packages

- PennyLane
- Numpy and Scipy
- Pyaml
- Pickle
- Matplotlib
- mpmath
- OpenFermion (optional)

B. Inputs

Inputs given in a `input.yaml` file:

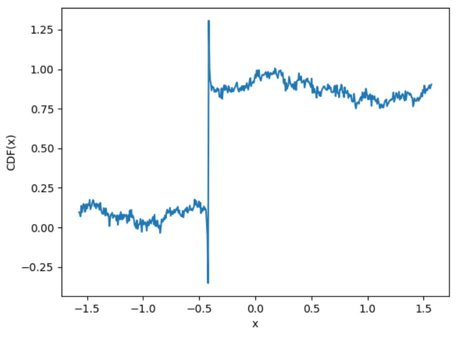
- Number of samples for the Fourier series, `sampling k: num_sample`
- Number of iterations for Monte Carlo simulation to thermalize, `sampling k: num_thermalization`.
- β value for the Heaviside Fourier series, `sampling k: beta`

- Maximum value of k for the Heaviside Fourier series, `sampling k: max_k`, which should be chosen to be in $O\left(\frac{1}{\delta}\log\frac{1}{\epsilon}\right)$
- Number of iterations for Monte Carlo simulation to thermalize during LCU sampling
- δ, tolerance of the x-value error in the binary search, `binary search: delta`.
- η, lower bound on overlap between input state and ground state, `binary search: eta`.
- Type of search to find ground state, 'mv' stands for majority vote, stored in `binary search: certification`.
- Full path to a Pickle file which stores a tuple of PennyLane Hamiltonian object, and a trial state in the form of a vector, `hamiltonian: file`.
- Norm bound for the scaled eigenvalues, `hamiltonian: norm_bound`, which is chosen to be $\pi/2$ by default.
- Error tolerance in eigenvalues, ϵ, stored in `hamiltonian: error_tolerance`.

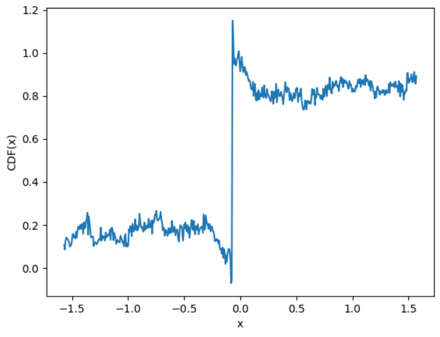
C. Running the code

A sample code is given in `samplecode.py`.

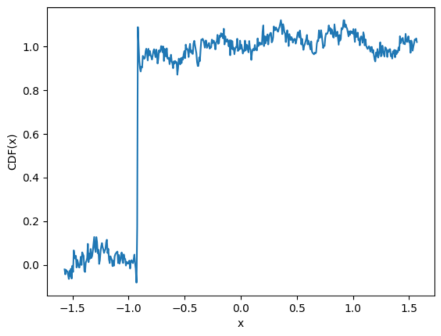
1. Make sure all required packages listed in III A are installed.
2. Copy the file path of the .pkl file of the Hamiltonian you want to use in the `hamiltonians` folder, and replace the file path in line 26 of the `inputs.yaml` file.
3. Change the parameters in `inputs.yaml` to the parameters you want to use. See [4] for more on how these parameters should be chosen.
4. If you would like a plot of the sampled approximate CDF and saved data: in line 13 of `samplecode.py`, set `to_plot` and `save_data` to `True`, and change the `plot_path` and `save_path` parameters in `StatisticalPhaseEstimation` to the paths where you want to save them on your computer. The filenames should include the proper suffixes (`.png` for the plot and `.pkl` for the saved data).
5. To run statistical phase estimation, two options are given. The default is to use the random sampling of the LCU decomposition of the time evolution from [4] through the `StatisticalPhaseEstimation_randomSampling` function. The other option is to exponentiate the Hamiltonian directly, and implement the time evolution operator exactly in the Hadamard test circuit. To run the latter option, change `StatisticalPhaseEstimation_randomSampling` in line 13 of `samplecode.py` to `StatisticalPhaseEstimation_wtLCU`.
6. Run `samplecode.py` to output the ground state energy estimate to the terminal and plots/data to the file paths indicated.



(a) H_2 Hamiltonian



(b) H_4 Hamiltonian



(c) LiH Hamiltonian

FIG. 8: Examples of the reconstructed approximate CDF function and the jump at the ground state energy using a Hartree-Fock state. The parameters chosen for these simulations were $k_{max} = 10^4$, $\beta = 10^6$, $\text{num}_{\text{samples}} = 10^4$, $\delta = 0.001$, $\eta = 0.5$. For H_2, we get estimate of $\tilde{E}_0 = -0.8972$ Ha using the modified binary search procedure; for H_4 we get an estimate of $\tilde{E}_0 = -0.9517$ Ha; for LiH we get get an estimate of $\tilde{E}_0 = -7.6600$ Ha. Here we used exact Hamiltonian exponentiation for the time evolution in the quantum circuit.

IV. RESULTS

As a first pass, we ran our code on H_2, H_4 and LiH Hamiltonians with their Hartree-Fock states as the trial state, and using exact exponentiation for the time evolution instead of randomized sampling. We ended up with results that are in good agreement with the exact ground state energies for these molecules, which were found by directly computing the eigenvalues of the Hamiltonian. We plot the resulting approximate CDF function of the corresponding Hamiltonians in Fig. 8. The jumps in these figures will correspond to τE_0. For H_2 with an exact ground state energy of $E_0 = -0.8964$ Ha, we get an estimate of $\tilde{E}_0 = -0.8972$ Ha. For H_4 with $E_0 = -0.9502$ Ha, we get an estimate of $\tilde{E}_0 = -0.9517$ Ha. For LiH with $E_0 = -7.6600$, we get an estimate of $\tilde{E}_0 = -7.6600$ Ha, which is the same to 5 significant figures.

A. Limitations

In [4], the relationships between the parameters are given in order to achieve a desired error bound ϵ. However, most of these are given as asymptotic scaling bounds, so we may not have had ideal values for these parameters. Also, since we did not have access to actual quantum hardware to run these circuits on, we had to do all the matrix multiplication operations on a classical computer, which would have taken too long to run. Unfortunately, we could not get the randomized version of the Hamiltonian simulation algorithm to converge in a reasonable amount of time, so we were only able to get results for the version with direct exponentiation.

V. FUTURE DIRECTIONS

There are multiple ways to approximate the time-evolution unitary, out of which we chose importance sampling its Taylor series. One can also try to use variational circuits with finite depth to implement an approximation of the time-evolution as done in [5]. This may be nicely implemented in PennyLane using its built-in optimizers.

A major cost of the current circuit implementations is the number of Pauli rotations required to approximate the LCU unitaries. A possible simplification would be to re-order and group the Pauli rotations into commuting Pauli rotations that can be diagonalized by Clifford unitaries. Additionally, approximation techniques have been utilized in other works to reduce and approximate Pauli rotations to Cliffords. This can be utilized to simply the unitaries further.

One can implement this algorithm for several applications. A straightforward use is to study phase diagrams of systems by looking at the second derivative of the estimated ground state energy *w.r.t* any parameters being tuned. Another extension might be to try estimating excited state energies. One can start with a simple Hartree-Fock state assuming it has a sizable overlap with the true ground state. Then, we choose a trial state randomly in the orthogonal sub-space of such a Hartree-Fock state, and find the lowest energy where a jump occurs in the CDF for such a trial state.

[1] Alexei Y. Kitaev. Quantum measurements and the abelian stabilizer problem. *Electron. Colloquium Comput. Complex.*, TR96, 1995.

[2] Alexandria J Moore, Yuchen Wang, Zixuan Hu, Sabre Kais, and Andrew M Weiner. Statistical approach to quantum phase estimation. *New Journal of Physics*, 23(11):113027, nov 2021.

[3] Lin Lin and Yu Tong. Heisenberg-limited ground-state energy estimation for early fault-tolerant quantum computers. *PRX Quantum*, 3:010318, Feb 2022.

[4] Kianna Wan, Mario Berta, and Earl T. Campbell. Randomized quantum algorithm for statistical phase estimation. *Phys. Rev. Lett.*, 129:030503, Jul 2022.

[5] Nick S. Blunt, Laura Caune, Róbert Izsák, Earl T. Campbell, and Nicole Holzmann. Statistical phase estimation and error mitigation on a superconducting quantum processor. *PRX Quantum*, 4:040341, Dec 2023.