Encoding Electronic Spectra in Quantum Circuits with Linear $T$ Complexity

Ryan Babbush,1,* Craig Gidney,2 Dominic W. Berry,3 Nathan Wiebe,4 Jarrod McClean,1 Alexandru Paler,5 Austin Fowler,2 and Hartmut Neven1

1Google Inc., Venice, California 90291, USA
2Google Inc., Santa Barbara, California 93117, USA
3Department of Physics and Astronomy, Macquarie University, Sydney, NSW 2109, Australia
4Microsoft Research, Redmond, Washington 98052, USA
5Institute for Integrated Circuits, Linz Institute of Technology, 4040 Linz, Austria

(Received 9 May 2018; revised manuscript received 1 August 2018; published 23 October 2018)

We construct quantum circuits that exactly encode the spectra of correlated electron models up to errors from rotation synthesis. By invoking these circuits as oracles within the recently introduced “qubitization” framework, one can use quantum phase estimation to sample states in the Hamiltonian eigenbasis with optimal query complexity $O(\lambda/\epsilon)$, where $\lambda$ is an absolute sum of Hamiltonian coefficients and $\epsilon$ is the target precision. For both the Hubbard model and electronic structure Hamiltonian in a second quantized basis diagonalizing the Coulomb operator, our circuits have $T$-gate complexity $O(N + \log(1/\epsilon))$, where $N$ is the number of orbitals in the basis. This scenario enables sampling in the eigenbasis of electronic structure Hamiltonians with $T$ complexity $O(N^3/\epsilon + N^2 \log(1/\epsilon)/\epsilon)$. Compared to prior approaches, our algorithms are asymptotically more efficient in gate complexity and require fewer $T$ gates near the classically intractable regime. Compiling to surface code fault-tolerant gates and assuming per-gate error rates of one part in a thousand reveals that one can correct phase estimation on interesting instances of these problems beyond the current capabilities of classical methods using only about a million superconducting qubits in a matter of hours.

DOI: 10.1103/PhysRevX.8.041015
Subject Areas: Chemical Physics, Quantum Information, Strongly Correlated Materials

I. INTRODUCTION

The ubiquitous problem of predicting material properties and chemical reactions from \textit{ab initio} quantum mechanics is among the most anticipated applications of quantum computing. The limitation of most classical algorithms for modeling the physics of superconductivity and molecular electronic structure arises from the seemingly exponential growth of entanglement required to accurately capture strong correlation in systems of interacting electrons. This apparent classical intractability was referenced by Feynman in his seminal work as a key motivation for why we need quantum computers [1,2]. Fourteen years later, Lloyd formalized the concept of a universal quantum simulator [3] and demonstrated an extension for treating systems of interacting electrons in second quantization [4].

Since then, most work developing fermionic quantum simulation methods has focused on time evolution as a means of estimating Hamiltonian spectra and preparing eigenstates [5] via the quantum phase estimation algorithm [6]. Beginning with the proposal of Ref. [7], the idea that one should use phase estimation and adiabatic state preparation [8–10] to extract quantum chemistry ground-state energies became especially popular. More recently, experimental demonstrations [11–15] have focused on the development of variational algorithms [16,17], which are often [18,19], but not always [20,21], inspired by time-evolution primitives.

Performing quantum phase estimation to sample Hamiltonian spectra requires a quantum circuit to implement an operation $W(H)$, which has eigenvalues that are a known (and efficient-to-compute) function of the eigenvalues of $H$. Most past work has analyzed phase estimation of circuits corresponding to dynamical Hamiltonian simulation, i.e., $W(H) \approx e^{-iH\tau}$ for some duration $\tau$ [6]. We denote by $f$ the cost of implementing a primitive circuit that is repeated to realize $W(H)$; e.g., a Trotter step [22] or Taylor series segment [23]. We further define $g(\epsilon)$ as the number of times that one must repeat that primitive to ensure that the error in the spectrum of $H$ encoded in the eigenphases of $W(H)$ is at most $O(\epsilon)$. Then, the cost of phase estimation is bounded by

---

*Corresponding author.
babbush@google.com

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article’s title, journal citation, and DOI.
\[\mathcal{O}\left(\frac{f \cdot g(e)}{\epsilon} \|W'(H)\|^{-1}\right),\]  

(1)

where \(\| \cdot \|\) denotes the spectral norm and we have taken the derivative of the function of the eigenvalues in the operation \(W'(H)\). In other words, \(W'(H)\) has eigenvalues that are a function of the eigenvalues of \(H\), and that function is the derivative of the function that gives the eigenvalues of \(W(H)\). For the case of dynamical time evolution,

\[W(H) \approx e^{-itH}, \quad \|W'(H)\|^{-1} = \| -i\epsilon e^{-itH} \|^{-1} = \frac{1}{\epsilon},\]  

(2)

implying that the cost of phase estimation is \(\mathcal{O}(f \cdot g(e)/\epsilon)\) in this context.

Modern Hamiltonian simulation methods such as the signal processing algorithm [24] and qubitization [25] have achieved the provably optimal scaling that is possible for signal processing algorithms and qubitization have achieved the provably optimal scaling that is possible for quantum computation, the physical resources required for error correcting a quantum circuit are mostly determined by (i) the number of logical qubits and (ii) the number of T gates.

The focus on T gates arises because applying a single T gate consumes many logical qubits and takes significantly longer than applying any other operation [36]. Preparing a |T\rangle state to enable a T gate requires hundreds of thousands of physical qubits. If the goal is to minimize the number of qubits required to execute an algorithm, it makes sense to prepare |T\rangle states serially. Typically, it also takes over 100 rounds of error detection to prepare a |T\rangle state, leaving plenty of time to perform Clifford gates in parallel with this preparation, meaning the execution time of the complete algorithm can be approximated as the total number of T gates multiplied by the time to prepare each |T\rangle state. Thus, throughout this work, we focus on T complexity as the primary cost model. We note, however, that for all algorithms presented or discussed in this work, the T complexity is within logarithmic factors of the gate complexity.

We focus on the two most-studied models of correlated electrons: the Fermi-Hubbard model and the molecular electronic structure Hamiltonian. The Hubbard Hamiltonian is an approximate model of electrons interacting on a planar lattice which some believe may qualitatively capture the behavior of high-temperature superconductivity in cuprates [37]. The molecular electronic structure Hamiltonian is a realistic model of electrons interacting via the Coulomb potential with real kinetic energy in the presence of an external potential (which usually arises from atomic nuclei), in a finite-sized basis [38]. We focus on simulating the electronic structure Hamiltonian in a basis diagonalizing the Coulomb potential [39–41]. For both the Hubbard model and molecular electronic structure Hamiltonian, we are able to provide circuits that simulate \(e^{i\text{arccos}(H/\lambda)}\) with T complexity \(\mathcal{O}(N + \log(1/\epsilon))\), where \(N\) is the number of orbitals in a second-quantized representation of the system. In Tables I and II, we compare the T complexity of past quantum simulation methods for these problems.

In Theorems 1 and 2, we concisely state the T complexity and ancilla requirements of our approach to phase estimation for both the electronic structure Hamiltonian and Hubbard model Hamiltonian, respectively. Both of these theorems are established throughout the paper, but especially in Eqs. (54), (55), (61), and (62). In addition to bounding the T complexity of our algorithms, we provide explicit circuits for their construction and compile all bottleneck primitives down to surface code fault-tolerant gates (topological braiding diagrams). Therefore, the
TABLE I. Progression of lowest T complexity algorithms for implementing a phase estimation unitary encoding eigenvalues of the electronic structure Hamiltonian in second quantization. Note that N is the number of spin orbitals, and ε is the target precision. Here and throughout the paper, $O(\cdot)$ indicates an upper bound, $\mathcal{O}(\cdot)$ indicates an upper bound ignoring polylogarithmic factors, and $O(\sim \cdot)$ indicates empirical scaling extrapolated from numerics. “Oracle T gates” refers to $f$ from Eq. (1), and “PEA queries” refers to the rest of the expression in Eq. (1). The scalings attributed to the work on general methods of Hamiltonian simulation assume that one uses the best oracles for electronic structure available at that point in time; e.g., the scaling attributed to Ref. [25] assumes the use of oracles from Ref. [42], and the scaling attributed to Ref. [26] assumes the use of oracles from Ref. [39]. While absent from this table since it did not asymptotically reduce T complexity, Ref. [43] was the first work to explicitly compile a quantum chemistry simulation to Clifford $+$ T gates.

<table>
<thead>
<tr>
<th>Year</th>
<th>Reference</th>
<th>Basis</th>
<th>Algorithm</th>
<th>Oracle T gates</th>
<th>PEA queries</th>
<th>Total T gates</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005</td>
<td>Aspuru-Guzik et al. [7]</td>
<td>Gaussians</td>
<td>Trotterization</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2010</td>
<td>Whitfield et al. [44]</td>
<td>Gaussians</td>
<td>Trotterization</td>
<td>$O(N^4 \log (1/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2013</td>
<td>Wecker et al. [45]</td>
<td>Gaussians</td>
<td>Trotterization</td>
<td>$O(N^4 \log (1/\varepsilon))$</td>
<td>$O(\sim N^6 \varepsilon^{3/2})$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2014</td>
<td>McClean et al. [46]</td>
<td>Gaussians</td>
<td>Trotterization</td>
<td>$O(\sim N^2 \log (1/\varepsilon))$</td>
<td>$O(\sim N^6 \varepsilon^{3/2})$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2014</td>
<td>Poulin et al. [47]</td>
<td>Gaussians</td>
<td>Trotterization</td>
<td>$O(N^4 \log (1/\varepsilon))$</td>
<td>$O(\sim N^6 \varepsilon^{3/2})$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2014</td>
<td>Babbush et al. [48]</td>
<td>Gaussians</td>
<td>Trotterization</td>
<td>$O(N^4 \log (1/\varepsilon))$</td>
<td>$O(\sim N^6 \varepsilon^{3/2})$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2015</td>
<td>Babbush et al. [42]</td>
<td>Gaussians</td>
<td>Taylorization</td>
<td>$O(N)$</td>
<td>$O(N^4 \log (N/\varepsilon))/[\varepsilon \log \log (N/\varepsilon)]$</td>
<td>$O(N^6 \varepsilon^{-3/2})$</td>
</tr>
<tr>
<td>2016</td>
<td>Low et al. [25]</td>
<td>Gaussians</td>
<td>Qubitization</td>
<td>$O(N)$</td>
<td>$O(N^4 \varepsilon^{-1/2} + \log (N/\varepsilon))/[\varepsilon \log \log (N/\varepsilon)]$</td>
<td>$O(N^6 \varepsilon^{-3/2})$</td>
</tr>
<tr>
<td>2017</td>
<td>Babbush et al. [39]</td>
<td>Plane waves</td>
<td>Taylorization</td>
<td>$O(N)$</td>
<td>$O(N^8/3 \log (N/\varepsilon))/[\varepsilon \log \log (N/\varepsilon)]$</td>
<td>$O(N^{11/3} \varepsilon^{-1/2})$</td>
</tr>
<tr>
<td>2017</td>
<td>Berry et al. [26]</td>
<td>Plane waves</td>
<td>Qubitization</td>
<td>$O(N)$</td>
<td>$O(N^{8/3} \varepsilon^{-1/2})$</td>
<td>$O(N^{11/3} \varepsilon^{-1/2})$</td>
</tr>
<tr>
<td>2018</td>
<td>Kivlichan et al. [40]</td>
<td>Plane waves</td>
<td>Qubitization</td>
<td>$O(N^2 + N \log N \log (1/\varepsilon))$</td>
<td>$O(\sim N^{3/2} \varepsilon^{-3/2})$</td>
<td>$O(\sim N^5 \varepsilon^{-3/2})$</td>
</tr>
<tr>
<td>2018</td>
<td>This paper</td>
<td>Plane waves</td>
<td>Qubitization</td>
<td>$O(N + \log (1/\varepsilon))$</td>
<td>$O(\sim N^{3/2} \varepsilon^{-3/2})$</td>
<td>$O(\sim N^5 \varepsilon^{-3/2})$</td>
</tr>
</tbody>
</table>

TABLE II. Progression of lowest T complexity algorithms for implementing a unitary encoding eigenvalues of the Hubbard model for phase estimation. Here, $N$ is the number of sites, and $\varepsilon$ is the target precision. As before, $O(\cdot)$ indicates an upper bound, $\mathcal{O}(\cdot)$ indicates an upper bound ignoring polylogarithmic factors, and $O(\sim \cdot)$ indicates empirical scaling extrapolated from numerics. “Oracle T gates” refers to $f$ from Eq. (1), and “PEA queries” refers to the rest of the expression in Eq. (1). The scalings attributed to the work on general methods of Hamiltonian simulation assume that one uses the best oracles for the Hubbard model available at that point in time. For instance, the scaling attributed to Refs. [25] and [26] assumes that one uses the SELECT oracles from Ref. [42], which also work for the Hubbard model. We assume that the work of Ref. [49] would use oracles from Ref. [27]. The scalings attributed to this work assume that our techniques are combined with those from Ref. [49], even though we do not focus on that strategy in our fault-tolerant analysis since those methods seem less effective for finite problem sizes near the classically intractable regime. Nonetheless, we discuss how our methods can be combined to provide the stated complexity in Sec. V D.

<table>
<thead>
<tr>
<th>Year</th>
<th>Reference</th>
<th>Algorithm</th>
<th>Ancillae</th>
<th>Oracle T gates</th>
<th>PEA queries</th>
<th>Total T gates</th>
</tr>
</thead>
<tbody>
<tr>
<td>1997</td>
<td>Abrams et al. [4]</td>
<td>Trotterization</td>
<td>$O(1)$</td>
<td>$O(N \log (1/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2015</td>
<td>Wecker et al. [50]</td>
<td>Trotterization</td>
<td>$O(1)$</td>
<td>$O(N \log (1/\varepsilon))$</td>
<td>$O(\mathcal{O}(N^2 \varepsilon^{3/2}))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2015</td>
<td>Babbush et al. [42]</td>
<td>Taylorization</td>
<td>$O(\log (N/\varepsilon))/[\varepsilon \log \log (N/\varepsilon)]$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N^2 \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2016</td>
<td>Low et al. [25]</td>
<td>Qubitization</td>
<td>$O(\log N)$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2017</td>
<td>Berry et al. [26]</td>
<td>Qubitization</td>
<td>$O(\log N)$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2017</td>
<td>Poulin et al. [27]</td>
<td>Qubitization</td>
<td>$O(N)$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2018</td>
<td>Haah et al. [49]</td>
<td>Qubitization</td>
<td>$O(\log N)$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2018</td>
<td>Kivlichan et al. [40]</td>
<td>Qubitization</td>
<td>$O(\log N)$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
<tr>
<td>2018</td>
<td>This paper</td>
<td>Qubitization</td>
<td>$O(\log N)$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(N \log (N/\varepsilon))$</td>
<td>$O(\text{poly}(N/\varepsilon))$</td>
</tr>
</tbody>
</table>
fault-tolerant aspect of our analysis goes further than prior estimates in the simulation literature [51], the most rigorous of which stopped at estimates of $T$ complexity for Trotter-based electronic structure phase estimation [43] and for a variety of techniques used to effect time-evolution of the one-dimensional Heisenberg model[52]. We show that one can perform fault-tolerant phase estimation on interesting instances of both Fermi-Hubbard and molecular electronic structure beyond the capabilities of known classical algorithms using roughly $1 \times 10^6$ physical qubits in the surface code, assuming an architecture with two-qubit error rates of about one part in a thousand.

**Theorem 1.** Consider the electronic structure Hamiltonian in a basis of $N$ spin orbitals, which diagonalizes the Coulomb operator, $H=\sum_{p,q} T(p-q)a_p^\dagger a_q + \sum_p U(p) n_p + \sum_{p\neq q} V(p-q)n_p n_q$, where $\{a_p, a_q\} = \delta_{pq}$ are fermionic raising and lowering operators. Furthermore, define $\lambda=\sum_{pq} |T(p-q)| + \sum_p U(p) + \sum_{p\neq q} |V(p-q)|$. Then, one can perform phase estimation to sample in the eigenbasis of $H$ with an additive error of at most $\epsilon$ in the eigenvalue using circuits with a number of $T$ gates scaling as $24\sqrt{2p\pi N\lambda/\epsilon} + O((\lambda/\epsilon)\log(N/\epsilon))$ and a number of ancilla qubits scaling as $\log(2N^3/\epsilon^3) + O(1)$.

**Theorem 2.** Consider the square planar Hubbard model with periodic boundary conditions in a basis of $N$ spin orbitals, $H = -t \sum_{(p,q)\sigma} a_{p,\sigma}^\dagger a_{q,\sigma} + (u/2) \sum_{p,\sigma\neq\bar{\sigma}} n_{p,\sigma} n_{p,\bar{\sigma}}$, where $\{a_{p,\sigma}, a_{q,\bar{\sigma}}\} = \delta_{pq}\delta_{\sigma\bar{\sigma}}$ are fermionic raising and lowering operators and the $\langle p, q \rangle$ notation implies a summation over nearest-neighbor orbitals on the periodic planar lattice. Furthermore, define $\lambda = 2Nt + Nu/2$. Then, one can perform phase estimation to sample in the eigenbasis of $H$ with an additive error of at most $\epsilon$ in the eigenvalue using circuits with a number of $T$ gates scaling as $10\sqrt{2\pi N\lambda/\epsilon} + O((\lambda/\epsilon)\log(N/\epsilon)/\epsilon)$ and a number of ancilla qubits scaling as $\log(2N^3/\epsilon^3) + O(1)$.

In Sec. II, we give an overview of the simulation strategy that we use to encode and sample eigenstates via phase estimation. Section II A discusses how one can synthesize $e^{i \arccos(H/\lambda)}$ within the linear combinations of unitaries query model requiring two oracle circuits: SELECT and PREPARE. Section II B introduces a particularly precise variant of phase estimation which queries SELECT and PREPARE oracles to estimate spectra with a precision exceeding the typical Holevo variance. Section II C analyzes the various sources of errors that we need to consider in this algorithm and then bounds the number of times we must query SELECT and PREPARE in order to perform phase estimation.

Sections III–V focus on explicit constructions of SELECT and PREPARE. Section III introduces important primitives for both SELECT and PREPARE. In Sec. III A, we describe circuits applying controlled unitaries such as the mapping $|\ell\rangle|\psi\rangle \mapsto |\ell\rangle X_\ell |\psi\rangle$ with $T$ complexity $O(L)$, where $L$ is the number of possible values of $\ell$. In Sec. III B, we show how to selectively apply a Majorana fermion operator, a primitive necessary for our implementation of SELECT in later sections. In Sec. III C, we use the result of Sec. III A to show a particularly efficient variety of quantum read-only memory (QROM), which we use for our PREPARE circuit. In Sec. III D, we describe a general technique for implementing PREPARE in a fashion that keeps $\lambda$ as small as possible.

Sections IV A and IV B discuss explicit constructions of SELECT and PREPARE circuits for the electronic structure Hamiltonian. Sections VA and VB discuss explicit constructions of SELECT and PREPARE circuits for the Hubbard model Hamiltonian. Sections IV C and VC focus on quantifying the number of $T$ gates and ancillae required by the algorithms described in Sec. IV and V. These sections include investigations of the finite-size magnitude of the $\lambda$ and target precisions required to implement our algorithms for interesting problems. In Section V D, we discuss how our Hubbard model simulation techniques can be combined with recent results to achieve even lower scaling based on the locality of the Hubbard Hamiltonian.

Finally, Sec. VI discusses the compilation of these routines to surface code fault-tolerant gates and estimates the physical resources required for error-correcting these algorithms under realistic assumptions about hardware. We conclude in Sec. VII with an outlook on future directions for quantum simulating correlated electron models.

**II. PHASE ESTIMATING SPECTRA OF HERMITIAN LINEAR COMBINATIONS OF UNITARIES**

The primary contribution of this paper is to demonstrate a particularly efficient method of using quantum computation to sample the spectra of correlated electron Hamiltonians. Though details of our implementation are specialized to electronic systems, our high-level simulation strategy represents a general framework for spectral estimation. While aspects of this approach were introduced recently in Refs. [26,27], the techniques involved emerged from a history of advances in Hamiltonian simulation prominently involving Szegedy quantum walks [53], the “linear combination of unitaries” (LCU) query model [54], and the method of Hamiltonian simulation known as “qubitization” [25].

Oracular methods of Hamiltonian simulation assume that information about a Hamiltonian is provided by querying “oracle” circuits [55]. These techniques aim to reduce the number of times one must query these oracles in order to effect the intended simulation to target accuracy. The techniques in this paper implement oracles from the LCU query model introduced in Ref. [54]. As the name suggests, this approach begins from the observation that any Hamiltonian can be decomposed as a linear combination of unitaries.
\[ H = \sum_{\ell=0}^{L-1} w_\ell H_\ell, \quad \text{s.t.} \quad (w_\ell \in \mathbb{R}) \land (w_\ell \geq 0), \]
\[ H_\ell^2 = 1, \tag{5} \]

where \( w_\ell \) are scalars and \( H_\ell \) are self-inverse operators that act on qubits; e.g., \( H_\ell \) could be strings of Pauli operators. The convention in this paper is that the \( w_\ell \) are real and non-negative, with any phases included in the \( H_\ell \).

LCU simulation techniques are formulated in terms of queries to two oracle circuits. The first oracle circuit, the “preparation oracle,” acts on an empty ancilla register of \( O(\log L) \) qubits and prepares a particular superposition state related to the notation of Eq. (5).

\[
\text{PREPARE} \equiv \sum_{\ell=0}^{L-1} \sqrt{\frac{w_\ell}{\lambda}} |\ell\rangle \langle 0|, \\
\text{PREPARE}[0]^\otimes \log L \mapsto \sum_{\ell=0}^{L-1} \sqrt{\frac{w_\ell}{\lambda}} |\ell\rangle \equiv |\mathcal{L}\rangle, \\
\lambda \equiv \sum_{\ell=0}^{L-1} w_\ell. \tag{6}
\]

The quantity \( \lambda \) is the same as that in Eq. (3) and turns out to have significant ramifications for the overall algorithm complexity. The second oracle circuit we require acts on the ancilla register \( |\ell\rangle \) as well as the system register \( |\psi\rangle \) and directly applies one of the \( H_\ell \) to the system, controlled on the ancilla register. For this reason, we refer to the ancilla register \( |\ell\rangle \) as the “selection register” and name the second oracle the “Hamiltonian selection oracle,”

\[
\text{SELECT} \equiv \sum_{\ell=0}^{L} |\ell\rangle \langle \ell| \otimes H_\ell, \\
\text{SELECT}|\ell\rangle |\psi\rangle \mapsto |\ell\rangle H_\ell |\psi\rangle. \tag{7}
\]

Note that the self-inverse nature of the \( H_\ell \) operators implies that they are both Hermitian and unitary, which means they can be applied directly to a quantum state.

### A. Encoding spectra in Szegedy quantum walks using qubitization oracles

The essential simulation primitive deployed here (a quantum walk operator based on SELECT and PREPARE) was first introduced as a subroutine to the qubitization approach for Hamiltonian time evolution [25]. However, the direct use of this primitive for phase estimation was first suggested more recently in Ref. [26]. In Sec. II B and the Appendix, we go beyond existing work and prove that, as long as the eigenphase of the walk operator is bounded away from zero (so that the Hamiltonian is not frustration-free), then this algorithm can, in principle, learn as quickly as traditional phase estimation using the Cramér-Rao bound.

We begin our discussion with the observation that the state \( |\mathcal{L}\rangle \) from Eq. (6) encodes \( H \) as a projection of SELECT onto \( |\mathcal{L}\rangle \),

\[
(\langle \mathcal{L} | \otimes 1) \text{SELECT} (|\mathcal{L}\rangle \otimes 1) = \frac{1}{\lambda} \sum_{\ell} w_\ell H_\ell = \frac{H}{\lambda}. \tag{8}
\]

This encoding is a general condition for qubitization [25], but the LCU oracles SELECT and PREPARE, as defined in Eqs. (7) and (6), are not necessarily the only constructions that meet this criterion; we refer to the broader family of circuits satisfying Eq. (8) as “qubitization oracles.” With this in mind, we discuss a walk operator \( \mathcal{W} \) that encodes the spectrum of \( H \) as a function of the eigenphases of \( \mathcal{W} \), although the spectrum of \( \mathcal{W} \) differs from that of the propagator \( e^{-iHt} \). One such walk operator \( \mathcal{W} \) may be constructed as

\[
\mathcal{W} \equiv \mathcal{R}_L \cdot \text{SELECT}, \quad \mathcal{R}_L \equiv (2|\mathcal{L}\rangle \langle \mathcal{L}| \otimes 1 - 1). \tag{9}
\]

This construction takes the form of a Szegedy walk [53] since it is composed of a product of two reflection operations. The operation \( \mathcal{R}_L \) is manifestly a reflection operation, and it can be seen that SELECT is a reflection operation because

\[
\text{SELECT}^2 = \left( \sum_{\ell} |\ell\rangle \langle \ell| \otimes H_\ell \right)^2 = \sum_{\ell} |\ell\rangle \langle \ell| \otimes H_\ell^2 = 1. \tag{10}
\]

Fig. 1 shows a circuit that implements \( \mathcal{W} \) controlled on an ancilla.

The action of \( \mathcal{W} \) partitions Hilbert space into a direct sum of two-dimensional irreducible vector spaces. Through reasoning about these eigenspaces, we can deduce the spectrum of \( \mathcal{W} \) as well as the eigenvectors. In particular, we claim that the state \( |\mathcal{L}\rangle |k\rangle \) and an orthogonal state \( |\phi_k\rangle \) span the irreducible two-dimensional space that \( |\mathcal{L}\rangle |k\rangle \) is in under the action of \( \mathcal{W} \) for arbitrary eigenstate \( |k\rangle \) of \( H \) with eigenvalue \( E_k \). This state \( |\phi_k\rangle \) is formally defined to be the component of \( \mathcal{W}|\mathcal{L}\rangle |k\rangle \) that is orthogonal to \( |\mathcal{L}\rangle |k\rangle \), which can be simplified, using Eq. (8), to

\[
|\phi_k\rangle = (1 - |\mathcal{L}\rangle \langle \mathcal{L}| \otimes |k\rangle \langle k|) \cdot \text{SELECT} |\mathcal{L}\rangle |k\rangle \\
\sqrt{(1 - (E_k^2)1)(|\mathcal{L}\rangle |k\rangle)} \\
= \left( \frac{\text{SELECT} - \frac{E_k}{\lambda} 1}{\sqrt{1 - (E_k^2)}} \right)(|\mathcal{L}\rangle |k\rangle). \tag{11}
\]

The matrix elements of \( \mathcal{W} \) can be computed for this state. The upper diagonal matrix element follows from Eq. (8),
The upper transition matrix element between $|k\rangle\langle L|L\rangle|k\rangle$ is given from Eqs. (10) and (11) as

$$\langle k|\langle L|W|L\rangle|k\rangle = \langle k|\langle L|SELECT|L\rangle|k\rangle \frac{E_k}{\lambda}. \quad (12)$$

Note that because phase estimation on $W$ projects the system to an eigenstate of $W$ and because $W$ and $H$ share an eigenbasis, we are only concerned with the action of this operator for eigenstates.

Equations (12) and (13) give the first row of the action of $W$. The remaining entries can be calculated in a similar way, giving the action of $W$ on this irreducible two-dimensional subspace as

$$W = \begin{pmatrix} \frac{E_k}{\lambda} & \sqrt{1 - \left(\frac{E_k}{\lambda}\right)^2} \\ -\sqrt{1 - \left(\frac{E_k}{\lambda}\right)^2} & \frac{E_k}{\lambda} \end{pmatrix} = e^{i \arccos(E_k/\lambda) Y}, \quad (14)$$

where $Y$ is the Pauli-$Y$ operator constrained to this two-dimensional space spanned by $|L\rangle|k\rangle$ and $|\phi_k\rangle$. Finally, we can see that the phases of the eigenvalues of $W$ in this subspace are $\pm \arccos(E_k/\lambda)$. Whereas the work of Ref. [25] focused on transforming the evolution under $\arccos(H)$ into evolution under $H$, the more recent work of Refs. [26,27] made the simple observation that by performing phase estimation directly on $W$, one can obtain the spectrum of $H$ as

$$\text{spectrum}(H) = \lambda \cos(\text{arg}(\text{spectrum}(W))]. \quad (15)$$

where arg is the argument function $\text{arg}(e^{i \phi}) = \phi$.

### B. Heisenberg-limited phase estimation of the qubitized quantum walk

Since the original work of Ref. [6], many approaches have been proposed for estimating eigenphases of a unitary operator. Whereas, in the past, iterative phase estimation approaches have been more popular in quantum simulation, here we propose using an entanglement-based approach. This approach has the virtue of requiring a number of applications of the unitary that saturates the Heisenberg limit. The ultimate precision that can be reached when one applies phase estimation by controlling a unitary when an ancilla is in $|1\rangle$ and applying the identity gate when the ancilla is in $|0\rangle$ is a Holevo variance of $\tan^2(\pi/(2^{m+1} + 1))$, where the total number of applications of the unitary is $2^m - 1$ and $m$ is the number of control qubits used. The Holevo variance is $\langle \cos(\hat{\phi} - \phi) \rangle^2 - 1$, where $\phi$ is the phase and $\hat{\phi}$ is the estimate of the phase given by the measurement. It is a convenient measure of variance for phase because it enables simple analytic results and is close to the mean-square error for narrowly peaked distributions. The states for these optimal phase measurements were given in Ref. [56]. To apply them to phase estimation of a unitary, one can take the control qubits to be in this superposition state, rather than in a uniform superposition of computational basis states.

We perform a slight optimization of that approach by applying the inverse unitary instead of the identity for the ancilla in the $|0\rangle$ state. Taking $|\phi\rangle$ to be an eigenstate of the unitary with eigenvalue $e^{i \phi}$, this means that instead of applying $|0\rangle|\phi\rangle \rightarrow |0\rangle|\phi\rangle$ and $|1\rangle|\phi\rangle \rightarrow e^{i \phi}|1\rangle|\phi\rangle$, we apply $|0\rangle|\phi\rangle \rightarrow e^{-i \phi}|0\rangle|\phi\rangle$ and $|1\rangle|\phi\rangle \rightarrow e^{i \phi}|1\rangle|\phi\rangle$. This doubles the effective phase difference and turns out to have the same complexity. As shown in Fig. 2, we accomplish the controlled inverse by removing controls from $W^n$ and inserting controlled reflection operators $R_L$ into the circuit, which will cause us to apply either $(W^n)^2$ or $W^n$ depending on the state of the ancilla. We can see why this works by examining the relation

$$R_L \cdot W^n \cdot R_L = R_L^2 \cdot (\text{SELECT} \cdot R_L)^n = (\text{SELECT} \cdot R_L)^n = (W^n)^n, \quad (16)$$

![Diagram](https://example.com/diagram.png)

**FIG. 1.** A circuit realizing the Szegedy quantum walk operator $W$ controlled on an ancilla qubit. The last three gates in the circuit on the right constitute the reflection $R_L$ controlled on an ancilla. Note that the $Z$ gate with the 0-control is actually controlled on the zero state of the entire $|\ell\rangle$ register and not just a single qubit. Accordingly, implementation of that controlled $Z$ has $O(\log L)$, where $\log L$ is the size of the $|\ell\rangle$ register. However, that overhead is always negligible compared to the cost of the PREPARE and SELECT operators in the constructions of this paper.
which holds for any integer $n$ as a consequence of the self-inverse nature of $R_L$ and $W$. Moreover, because $\mathcal{W}^m$ always ends with an $R_L$ operation, each controlled $R_L$ can be combined in the circuit with the $\mathcal{W}^m$ to yield a complexity no greater than the complexity of just performing the $\mathcal{W}^m$ operations.

This trick will result in measuring the phase modulo $\pi$. To eliminate the $\pi$ ambiguity, an additional controlled $\mathcal{W}$ can be performed without this trick. This case is shown as the first controlled operation in Fig. 2. For $m$ control qubits, the Holevo variance is still $\tan^2(2^m/2^n + 1)$, but the complexity is reduced by approximately half to $2^n$ applications of the unitary $\mathcal{W}$.

As seen in Fig. 2, our modified phase estimation algorithm begins with a unitary $\chi_m$, which prepares the state

$$\chi_m |0\rangle^\otimes m \rightarrow \sqrt{\frac{2}{2^m + 1}} \sum_{n=0}^{2^m-1} \sin \left( \frac{\pi (n + 1)}{2^m + 1} \right) |n\rangle.$$

(17)

To prepare this state with cost $\tilde{O}(m)$, we first perform Hadamards on $m + 1$ qubits (initially in the $|0\rangle$ state) to give

$$\sqrt{\frac{1}{2^{m+1}}} \sum_{n=0}^{2^m-1} |n\rangle \otimes (|0\rangle + |1\rangle).$$

(18)

Next, we perform a series of $m$ controlled rotations, with each of the first $m$ qubits as control and qubit $m + 1$ as target. For control qubit $k$, the rotation on the target qubit $m + 1$ is $e^{i\pi Z}/(2^m + 1)$. If we perform a further rotation of $e^{i\pi Z}/(2^m + 1)$ on qubit $m + 1$, the resulting state is

$$\frac{1}{2^{(m+1)}} \sum_{n=0}^{2^m-1} (e^{i\pi (n+1)/(2^m+1)} |n\rangle \otimes |0\rangle + e^{-i\pi (n+1)/(2^m+1)} |n\rangle \otimes |1\rangle).$$

(19)

We perform a Hadamard on qubit $m + 1$ and measure in the computational basis. Measuring $|1\rangle$ gives the state

$$i \sqrt{\frac{1}{2^m}} \sum_{n=0}^{2^m-1} \sin \left( \frac{\pi (n + 1)}{2^m + 1} \right) |n\rangle.$$

(20)

The probability of success is given by the normalization $(1 + 2^{-m})/2$. The scheme can be made unitary and deterministic via a single step of amplitude amplification. Clearly, this preparation scheme scaling as $\tilde{O}(m)$ will not dominate the cost of our overall phase estimation, which scales as $O(2^m)$, as we discuss in the next section.

### C. Error scaling and query complexity

Three sources of error enter our simulation: error due to performing PEA to finite precision, $e_{PEA}$; error due to approximate preparation of the Hamiltonian terms within the implementation of the PREPARE oracle, $e_{PREP}$; and the error in synthesizing the inverse QFT, $e_{QFT}$. We choose to measure error through the root-mean-square error of the estimator used within phase estimation, i.e.,

$$\Delta \phi = \sqrt{\mathbb{E}[\text{dist}(\phi_{est}, \phi_{true})^2]},$$

(21)

where the distance considered above is the angular distance between the estimated phase and the actual phase.

Provided phase estimation is performed on a unitary operation, the error in the estimate of the energy is at most
the error in implementing the unitary [43]. We break up
the estimated phase as the sum of two contributions,
\( \phi_{\text{est}} = \phi + \epsilon_{\text{PREP}} + \phi_{\text{true}} \). Here, \( \phi \) is a random variable with
zero mean \( \mathbb{E}(\phi) \) and Holevo variance \( \mathcal{V}_H(\phi) \) describing
the output of phase estimation, and \( \epsilon_{\text{PREP}} \) represents the
systematic errors in the phase that arise because of gate
synthesis. In the limit of small variance, we can express this
with high probability over the true phase as

\[
\Delta \phi \approx \sqrt{\mathbb{E}[(\phi_{\text{est}} - \phi_{\text{true}})^2]}
= \sqrt{\mathcal{V}_H(\phi) + (\epsilon_{\text{PREP}} + \pi \epsilon_{\text{QFT}})^2}
\approx \sqrt{\left(\frac{\pi}{2^{m+1}}\right)^2 + (\epsilon_{\text{PREP}} + \pi \epsilon_{\text{QFT}})^2},
\]

where \( m \) ancillary qubits are used within the phase
estimation algorithm. Note that such a division of the error
is suboptimal since the cost involved in reducing the error
for phase estimation is exponentially larger than that
involved in increasing the accuracy of the circuit synthesis
[43]; however, we take the two errors to be equal for simplicity.

Eq. (15) implies that error in the energy is at most

\[
|\delta E| = \lambda \Delta \cos(\phi) \leq \lambda \Delta \phi \approx \lambda \sqrt{\left(\frac{\pi}{2^{m+1}}\right)^2 + (\epsilon_{\text{PREP}} + \pi \epsilon_{\text{QFT}})^2}.
\]

This result suggests that we can choose to estimate the
phase to a number of bits given by

\[
m \leq \left\lfloor \log \left( \frac{\sqrt{2} \pi \lambda}{2 \Delta E} \right) \right\rfloor \leq \log \left( \frac{\sqrt{2} \pi \lambda}{\Delta E} \right).
\]

and the target errors can be chosen as

\[
\epsilon_{\text{PREP}} \leq \frac{\sqrt{2} \Delta E}{4 \lambda}, \quad \epsilon_{\text{QFT}} \leq \frac{\sqrt{2} \Delta E}{4 \pi \lambda}.
\]

Thus, using the phase estimation procedure from Sec. II B,
we need at most

\[
2^m < \frac{\sqrt{2} \pi \lambda}{\Delta E}
\]

queries to the SELECT oracle and at most twice as many
queries to the PREPARE oracle in order to estimate spectra to
within error \( \Delta E \). Supposing that the circuit PREPARE can be
applied at gate complexity \( P \) and the circuit SELECT can be
applied at gate complexity \( S \), the gate complexity of our
simulation [ignoring, for now, the cost of \( \chi_m \) and the cost of
the \( \text{QFT} \) since they scale as \( O(m) \)] is then approximately
bounded from above by

\[
\frac{\sqrt{2} \pi \lambda(S + 2P)}{\Delta E}.
\]

This paper discusses implementations of SELECT and
PREPARE that minimize \( S \) and \( P \) without increasing \( \lambda \).

To implement the inverse QFT that appears in Fig. 2, we
use the semiclassical algorithm described in Ref. [57]. This
version of the QFT requires just \( m - 1 \) rotation gates and \( m \)
Hadamards when implemented on \( m \) qubits. Thus, the error
in each rotation must be at most \( \epsilon_{\text{QFT}}/(\pi m) \), which implies
that the inverse QFT will have \( T \) complexity scaling as
\( O(m \log(m/e_{\text{QFT}})) \). As this is an additive cost to other parts
of our phase estimation algorithm with \( T \) complexity
scaling as \( O(2^m) \), the cost of performing the QFT within
the required error budget can be safely neglected.

How errors in the coefficients of the implemented
Hamiltonian propagate into \( \epsilon_{\text{PREP}} \) is slightly harder to bound
owing to the fact that the error in the eigenphase is a
nonlinear function of the error in the Hamiltonian
implementation. In particular, the error can diverge for
frustration-free Hamiltonians owing to the singularity of
arccos. The main result, shown in the Appendix, is that
PREPARE should be implemented so that if \( \tilde{\omega}_\ell \) is the
effective coefficient of \( H_\ell \) in the approximately
implemented Hamiltonian, then

\[
|\tilde{\omega}_\ell - \omega_\ell| \leq \delta = \frac{\sqrt{2} \Delta E}{4L(1 + \frac{\Delta E}{8C})} \left( 1 - \frac{\|H\|^2}{2} \right).
\]

### III. LOW T COMPLEXITY PRIMITIVES

**FOR LCU ORACLES**

In this section, we introduce three circuit primitives that
are helpful for implementing SELECT and PREPARE oracles
with low T-gate complexity. We use these primitives for
electronic structure simulation but expect them to be useful
more generally. These primitives enable black-box imple-
mentations of SELECT and PREPARE for any problem with
lower asymptotic complexity than prior constructions in
the literature. They also have low T counts at finite size. We
use these constructions extensively in Secs. IV and V of
this paper.

In Sec. III A, we introduce a technique for “streaming”
bits of an iterator running over a unary register. One
application is that this technique can be used to coherently
apply operations controlled on a register with \( \log L \) qubits
in superposition [e.g., the selection register in Eq. (6) and
Eq. (7)] using a number of \( T \) gates scaling as \( O(L) \), as
opposed to \( O(L \log L) \) as one might normally expect.
However, what is even more important is the versatile
way that these constructions can be applied.

In Sec. III B, we show how one can use the results of
Sec. III A to implement a primitive corresponding to
controlled application of a Majorana fermion operator.
This primitive is used directly in our implementation of 

\textbf{SELECT in Secs. IV A and VA.}

In Sec. III C, we show a straightforward application of the techniques in Sec. III A that allow us to develop a particularly efficient quantum data lookup, which we refer to as “quantum read-only memory” (QROM). In particular, for coherently querying a database with \( L \) words, our implementation of QROM has T complexity of \( 4L - 4 \) with no dependence on the word length, which is an asymptotic and constant-factor improvement over all prior literature. We will discuss QROM in more detail in a forthcoming work [58].

In Sec. III D, we discuss a technique for initializing a state with \( L \) unique coefficients (provided by a classical database), with the number of T gates scaling as \( 4L + O(\log(1/\epsilon)) \), where \( \epsilon \) is the largest absolute error that one can tolerate in the prepared amplitudes. This routine improves asymptotically over the gate complexity of prior constructions for a black box \textsc{Prepare}. It also has the advantage of implementing \textsc{Prepare} without increasing the value of \( L \) from Eq. (6), which has been a frequent problem with other implementations of \textsc{Prepare} [39,42,59] that attempted to obtain scaling sublinear in the number of terms in the linear combinations of unitaries decomposition.

\textbf{A. Unary iteration and indexed operations}

Many of the circuits in this paper rely heavily on a technique we refer to as “unary iteration.” The unary iteration process gradually produces, and then uncomputes, qubits that indicate whether an index register is storing specific values (with respect to the computational basis). We call the process unary iteration because the indicator qubits are made available one by one (iteration), and they correspond to the one-hot (unary) encoding of the index register value. While these techniques were developed independently, we note that a scheme similar to unary iteration is also used for implementing \textsc{Select} operations in Ref [52]. Compared to Ref. [52], we lower the T count from \( 6L - 4 \) to \( 4L - 4 \) and explain how to apply the scheme to sizes \( L \) that are not powers of 2.

For an index register storing an index in the interval \([0, L]\), the space overhead of converting the index register into a unary register (as in Ref. [27]) would normally be \( L \) qubits. By comparison, our unary iteration technique is exponentially more efficient in space without any increased T complexity, requiring only \( \log L \) ancillae. Our unary iteration has a T-count of \( 4L - 4 \) and can be parallelized if needed without increasing the T count. Despite its efficiency, unary iteration is still the dominant source of T complexity in our algorithms. We use it for indexed operations, Majorana operators, reversible preparation of states, and database lookup, all of which have T counts that scale like \( O(L) \).

To explain unary iteration, we first focus on how it is used to implement a controlled indexed not operation:

\[
|c\rangle|\psi\rangle \mapsto |c\rangle|\psi\rangle(X_{c'})^{|\gamma}\rangle|\psi\rangle,
\]

where \(|c\rangle\) is the control register, \(|\gamma\rangle\) is the selection register, \(|\psi\rangle\) is the system register, and the subscript \( c' \) on \( X_{c'} \) indicates that the \textsc{not} operation acts on qubit \( c' \) of the system register. From Eq. (7), it should not be surprising that this primitive is helpful for our constructions of \textsc{Select}.

A simple (but suboptimal) way to implement Eq. (29) would be to totally control the application of \( X_{c'} \) on all possible values that could occur in the register \(|\gamma\rangle\), as shown in Fig. 3. For instance, in order to apply \( X_{158} \) when \(|\gamma\rangle = |158\rangle \), the total-control approach would place a \textsc{not} gate targeting the qubit 158 in the system register \(|\psi\rangle\) but with a control on each index bit. The control’s type (ON or OFF) would be determined by the binary representation of 158 (158_{10} = 10011110_2), so there would be a must-be-off control on the low bit of the index register (because the low bit of 158 in binary is 0), a must-be-on control on the next bit (because the next bit of 158 in binary is 1), and so forth. In order to cover every case, a separate \textsc{not} gate with corresponding controls would be generated for every integer from 0 up to \( L - 1 \). This would produce \( L \) different \textsc{not} operations, each targeting a different qubit in the target register and each having a number of controls equal to the size of the index register (i.e., \( \log L \)). Thus, it takes \( O(L \log L) \) T gates to apply Eq. (29) using this approach. Unary iteration will improve this T count to \( 4L - 4 \).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig3.png}
\caption{Example total control circuit for performing a controlled indexed \( X_{c'} \) operation, with \( 0 \leq c' < L = 11 \). This is the (naive) starting point for producing a unary iteration circuit, before optimizations that asymptotically improve the T complexity. When indices outside the specified range do not occur, the highlighted runs of \textsc{off}-type controls reaching the right-hand side of the circuit can be removed without affecting the circuit. (There are also other controls that could be removed, but for our purposes, this would be counterproductive because it would interfere with later optimizations).}
\end{figure}
Consider that the controls for the operation targeting the qubit at offset $l = 158$ are almost identical to the controls for the operation targeting the qubit at offset $l = 159$. They differ only on the low bit of the index register, where 158 requires the bit to be off, whereas 159 requires the bit to be on. If we combine the log $L - 1$ other qubits of the index register into a single representative qubit that is set if and only if those controls are met, we could use this representative qubit once for the $l = 158$ case and again for the $l = 159$ case. Using the representative qubit twice, instead of computing it twice, decreases the total amount of work done. Unary iteration is the result of taking this kind of representative-reuse idea to its natural limit.

We define our unary iteration construction by starting with a total-control circuit and then applying a fixed set of simple transformations and optimizations. Fig. 3 shows an example starting point, a total-control circuit for $L = 11$. For unary iteration, we require that the index register never store an out-of-range value $l \geq L$. For example, consider what occurs when the $X_{10}$ operation from Fig. 3 is not conditioned on $l_0$ (the least significant bit of the index register). This would cause an $X_{10}$ to be applied to the target when $l = 11$, but this is fine since we know $l \neq 11$. We use the $l < L$ condition to omit several controls from the circuit. For each possible $l$, we look at the $X_{\ell}$ operation and remove the control on the $b$th index bit when the following two conditions are true: (i) the $b$th bit of $L - 1$ is not set, and (ii) setting the $b$th bit of $\ell$ would change $\ell$ into a value larger than $L - 1$. Visually, this removes “runs” of must-be-off controls that manage to reach the right side of the circuit as highlighted in Fig. 3.

After removing the specified controls, we expand the remaining controls into nested AND operations (the AND operation is defined in Fig. 4), always nesting so that lower controls are inside higher controls. For clarity, we consistently place the ancillae associated with an AND operation just below its lowest input qubit. The result is the “sawtooth” circuit shown in Fig. 5. By iteratively optimizing adjacent AND operations as shown in Fig. 6, the sawtooth circuit from Fig. 5 is optimized into the circuit shown in Fig. 7. This is our unary iteration circuit for $L = 11$. The optimized circuit always ends up with $L - 1$ AND computations (even when $L$ is not a power of 2), each AND takes 4 T gates to compute, and we have no other T-consuming operations in the circuit. Thus, the T count of this construction is $4L - 4$.

**B. Selective application of Majorana fermion operators**

Now that we have described unary iteration, we can begin using it to construct primitives relevant for the SELECT oracle. As discussed in detail in Secs. IV and V

---

**FIG. 4.** Computing and uncomputing AND operations, defined in terms of Toffoli gates and in terms of Clifford + T gates [60]. Computing an AND consumes 4 $|T\rangle$ states and is equivalent to applying a Toffoli gate to a target qubit known to be $|0\rangle$. Uncomputing an AND consumes no $|T\rangle$ states and is equivalent to applying a Toffoli gate to a target qubit guaranteed to end up in the $|0\rangle$ state. Drawing AND operations as “corners” instead of as $\oplus$ symbols is a visual cue that the target qubit will be (or was) off after (before) the operation. This is worth highlighting because it affects the T count of synthesizing the operation and means that the target is available for reuse as an ancilla in later operations.

**FIG. 5.** The “sawtooth” circuit resulting from removing unnecessary bits from Fig. 3 and then adding AND operations from Fig. 4 to combine the controls for performing a controlled indexed $X_{\ell}$ operation with $L = 11$ possible targets.
below, our approach for implementing SELECT will require that we have a circuit capable of selectively applying the Majorana fermion operator

\[ j_l \psi_i \mapsto j_l \psi_i \cdot C_1/a_l - a_l \cdot C_1 \psi_i \]

where the last equality holds under the Jordan-Wigner transformation [61]. In this section, we describe explicit circuits that accomplish the mapping of Eq. (30).

In Sec. III A, we discuss selectively applying \( X_l \) operations as a representative example of how one might use unary iteration. However, nothing intrinsic to the unary iteration construction requires that the indexed operation be so simple. For example, we could switch from applying \( X_l \) to applying \( Z_l \) halfway through the circuit. Or each \( X_l \) could be replaced by multiple Pauli operations targeting multiple qubits. In general, each index could be associated with its own unique set of Pauli operators to be applied to various target qubits.

We can also apply transformations to our quantum unary iterators (analogous to transformations of classical iterators). Iterators can be mapped, filtered, zipped, aggregated, batched, flattened, grouped, etc. For instance, given a classical stream of bits, one can aggregate over it with the \( \oplus \) operation. This produces a new iterator, which iterates over bits equal to the parity of the values so far from the original iterator. It is possible to apply this XOR-aggregation idea to the quantum unary iteration process. We can introduce an “accumulator” qubit and, as each iterated unary qubit is produced, CNOT it into the accumulator. In effect, if the index register is storing \( \ell \), then the accumulator will stay OFF until the \( \ell \)th qubit toggles it ON. The accumulator will then stay ON until the end of the iteration process, where it is uncomputed by a CNOT from the control qubit. By conditioning \( X_\ell \) on the accumulator qubit, instead of on the original unary qubits, efficient ranged operations such as \( |\ell\rangle |\psi\rangle \mapsto |\ell\rangle G_\ell \cdot G_{\ell+1} \cdots G_{\ell-k} |\psi\rangle \) are produced. We show an example of an accumulator-based ranged operation in Fig. 8.

By using both the accumulator qubit and the original unary qubits, we can apply a ranged indexed operation and an indexed operation in a single unary iteration, which gradually sweeps over the possible target qubits. The resulting combined operation, shown in Fig. 9, is a crucial part of our SELECT circuit, effecting the transformation of Eq. (30).

C. QROM for low T complexity data lookup

In this section, we explain how one can use the techniques of Sec. III A in order to implement a particular efficient form of what we call QROM [62], which is useful in the context of the SUBPREPARE routine, a subroutine of the PREPARE circuit described in Sec. IV B (in Fig. 16). Many quantum algorithms assume the existence of a hypothetical peripheral called “quantum random-access memory” (QRAM) [63], which allows classical or quantum data to be accessed via an index under superposition. The purpose of QROM is to read classical data indexed by a quantum register, i.e., to perform the following transformation:

\[ \text{QROM}_d \cdot \sum_{\ell=0}^{L-1} a_\ell |\ell\rangle |0\rangle = \sum_{\ell=0}^{L-1} a_\ell |\ell\rangle |d_\ell\rangle, \]

FIG. 6. When two AND operations are adjacent, the uncomputation-and-recomputation can be replaced by CNOT and NOT operations. Each such merger saves 4 T gates.

FIG. 7. An \( L = 11 \) unary iteration circuit that applies \( X_l \) to the qubit \( \ell \) in the system register \(|\psi\rangle\), where \( \ell \) is the value stored in the selection register. The circuit is obtained by merging AND operations from Fig. 5 using the method shown in Fig. 6. It computes 10 AND operations and so has a T count of \( 10 \times 4 = 40 = 4L - 4 \).
FIG. 8. Ranged operation construction implementing $|\ell\rangle|\psi\rangle \rightarrow |\ell\rangle \prod_{k=p}^{q-1} G_k|\psi\rangle$. It applies the $G$ operation to a range of values, instead of to a single value, by using an accumulator. The accumulator is guaranteed to be cleared after the final CNOT targeting it (drawn as a line merging into an ancilla qubit). This occurs because (unless CONTROL is not set and the accumulator simply stays unset) exactly one of the unary bits must have been set, and we targeted the accumulator with CNOTs controlled by each of those bits in turn. Note that $G_{(p,q)}$ refers to $G$ being applied to every qubit index $k$ satisfying $p \leq k < q$.

FIG. 9. Application of a selected Majorana fermion operator, $|\ell\rangle|\psi\rangle \mapsto |\ell\rangle Y_\ell Z_{\ell-1} \cdots Z_0|\psi\rangle$ as described in Eq. (30). This application is accomplished by performing a ranged operation (as shown in Fig. 8) and an indexed operation (similar to what is shown in Fig. 7) with a single pass through the selection register $|\ell\rangle$. It has a T count of $4L - 4$, where $L$ is the number of integer values that can be held by the selection register $|\ell\rangle$. 

RYAN BABBUSH et al.

PHYS. REV. X 8, 041015 (2018)
where \( l \) is the index to read, \( \alpha_l \) is the amplitude of \( |j_{li}\rangle \), and \( d_l \) is the word associated with index \( l \) in a classical list \( d \) containing \( L \) words. Our implementation of QROM is shown in Fig. 10. Note that our notion of QROM is unrelated to the discussion of ROM on a quantum computer in Ref. [64].

The read-only aspect of QROM makes it distinctly different from QRAM in that one can read from QROM but cannot write to it during the course of a computation. A few algorithms, such as the procedure introduced in Ref. [65], actually do require that one write to QRAM; thus, for such cases, QROM would not be appropriate. A notable difference between this paper and most previous work on QRAM [63,66–68] is that we describe the cost of QROM in terms of a fault-tolerant cost model: the number of T gates performed and the number of ancilla qubits required. Under such cost models, the “bucket brigade” QRAM design of Giovannetti et al. [63,66] has T complexity (and, thus, also time complexity under reasonable error-correction models) of \( O(L) \) regardless of the fact that it has depth \( O(\log L) \) because implementing it as an error-corrected circuit consumes \( O(L) \) T gates and \( O(L) \) ancilla qubits. Our implementation of QROM consumes only \( 4L \) T gates and \( \log L \) ancillas, which is a constant-factor improvement in T count and an exponential improvement in space usage over the construction of Giovannetti et al.

D. Subsampling the coefficient oracle

In this section, we introduce a technique for initializing a state with \( L \) unique coefficients (provided by a classical database) with a number of T gates scaling as \( 4L + O(\log(1/\epsilon)) \), where \( \epsilon \) is the largest absolute error that one can tolerate in the prepared amplitudes. This result constitutes a general procedure for implementing PREPARE such that the cost of circuit synthesis is additive, rather than multiplicative (as in most prior schemes). In particular, it improves on the database scheme from Ref. [42] (based on the procedure of Ref. [69]), which requires a number of T gates scaling as \( O(L \log(L/\epsilon)) \). Importantly, our scheme does not increase the value of \( L \) or \( \lambda \), which would usually be the case for most “on-the-fly” strategies for implementing PREPARE [23,42,59].

Generalizing the requirements of Eq. (6), we begin with the observation that it would be acceptable to have a PREPARE circuit that initializes the state

\[
|\mathcal{L}\rangle \equiv \sum_{\ell=0}^{L-1} \sqrt{\frac{w_{\ell}}{\lambda}} |\ell\rangle |\text{temp}_{\ell}\rangle,
\]

where \( |\text{temp}_{\ell}\rangle \) is an unspecified junk register entangled with \( |\ell\rangle \). Equivalently, any pure state \( |\mathcal{L}\rangle \) would suffice if
Because \( \text{SELECT} \) only uses \( |\ell\rangle \) to control operations, phase error (including entanglement with the junk register) in the state produced by \( \text{PREPARE} \) will commute across \( \text{SELECT} \) and be corrected by \( \text{PREPARE} \). However, \( \text{SELECT} \) itself necessarily introduces entanglement between its target register \( |w\rangle \) and the index register \( |\ell\rangle \) (plus associated junk register). So \( \text{PREPARE} \) will not exactly restore \( |\ell\rangle \) or the junk register. Although we only specify the action of \( \text{PREPARE} \) on the \( |0\rangle \) state, \( \text{PREPARE} \) will be applied to other states due to this imperfect uncomputation effect. This is accounted for by requiring that (i) qubits coming out of \( \text{PREPARE} \) are kept and fed back into the next \( \text{PREPARE} \) operation and (ii) the reflection step between \( \text{PREPARE} \) and \( \text{PREPARE} \) only affects the \( |0\rangle \) state.

Given the observation that the existence of an entangled junk register is acceptable, we seek to implement a circuit that effects the transformation,

\[
|0\rangle^{\otimes(1+2\mu+2 \log L)} \mapsto \sum_{\ell=0}^{L-1} \sqrt{\tilde{\rho}_\ell} |\ell\rangle |\text{temp}_\ell\rangle,
\]

(34)

where \( \tilde{\rho}_\ell \equiv \tilde{w}_\ell/\lambda \) are probabilities characterizing the approximate Hamiltonian we are encoding. Whereas the exact Hamiltonian would be associated with probabilities \( \rho_\ell = w_\ell/\lambda \), the value \( \tilde{\rho}_\ell \) is a \( \mu \)-bit binary approximation to \( \rho_\ell \) such that

\[
|\rho_\ell - \tilde{\rho}_\ell| = \left| \frac{w_\ell - \tilde{w}_\ell}{\lambda} \right| \leq \frac{1}{2^\mu L} \leq \frac{\delta}{\lambda},
\]

(35)

\[
\mu = \left[ \log \left( \frac{2\sqrt{2} \Delta E}{\Delta E} \right) + \log \left( 1 + \frac{\Delta E^2}{8\lambda^2} \right) - \log \left( 1 - \frac{\|H\|^2}{\lambda^2} \right) \right],
\]

(36)

where the expression for \( \delta \) comes from Eq. (A10) of the Appendix, and it bounds the largest acceptable deviation in the coefficients of the terms in a Hamiltonian approximating the one we mean to implement. The second log in Eq. (36) is \( O(1) \) because we do not take \( \Delta E \) larger than \( \lambda \). The Hamiltonians we consider are frustrated; thus, \( \|H\|/\lambda \) is no larger than a constant (less than 1), and the third log in Eq. (36) is \( O(1) \) as well.

The idea behind our scheme is to create the superposition in an indirect fashion, which involves starting in a uniform superposition over an initial index \( \ell \) and then using a precomputed binary representation of a probability (loaded from QROM), \( \text{keep}_\ell \), to decide whether we should keep \( \ell \) or swap it with a classically precomputed alternate index \( \text{alt}_\ell \), which is also loaded from QROM (see Fig. 11).

Specifically, our procedure creates a uniform superposition in \( |\ell\rangle \) over \( L \) values and then uses QROM (see Fig. 10 and Sec. III C) to load \( |\text{alt}_\ell\rangle \) and \( |\text{keep}_\ell\rangle \). Note that if \( L \) is not a binary power, one can prepare the initial superposition using the amplitude amplification circuit discussed later in Fig. 12. The procedure described thus far prepares the state

\[
\sum_{\ell=0}^{L-1} \sqrt{\nu_{\ell}} |\ell\rangle |\text{alt}_\ell\rangle |\text{keep}_\ell\rangle.
\]

(37)

We then construct a circuit that coherently swaps the registers \( |\ell\rangle \) and \( |\text{alt}_\ell\rangle \) with probability \( \text{keep}_\ell \) to create the state in Eq. (34). In order to create the state in Eq. (34) from Eq. (37), we need to introduce one additional register of size \( \mu \), which we refer to as \( |\sigma\rangle \). We put this entire register into a uniform superposition and then compare it to the probability represented by \( \text{keep}_\ell \). If \( \text{keep}_\ell \leq \sigma \), we swap registers \( |\ell\rangle \) and \( |\text{alt}_\ell\rangle \). Thus, after the procedure is finished, i.e., in Eq. (34), the garbage register will be in the state

\[
|\text{temp}_\ell\rangle = \frac{1}{\sqrt{2^\mu \lambda \rho_\ell}} \left( |\text{alt}_\ell\rangle |\text{keep}_\ell\rangle \sum_{\sigma=0}^{\text{keep}_\ell-1} |\sigma\rangle |0\rangle \right.
\]

\[
\left. + \sum_{k=\text{alt}_\ell=\ell}^{2^\mu-1} |k\rangle |\text{keep}_k\rangle \sum_{\sigma=\text{keep}_k}^{2^\mu-1} |\sigma\rangle |1\rangle \right),
\]

(38)

where the rightmost qubit is the result of a comparison between \( \text{keep}_\ell \) and \( \sigma \). For Eq. (34) to give the correct state, we need \( |\text{temp}_\ell\rangle \) to be normalized, which means that we require

\[
\text{keep}_\ell + \sum_{k=\text{alt}_\ell=\ell}^{2^\mu-1} \left( 2^\mu - \text{keep}_k \right) = \tilde{\rho}_\ell = \frac{\tilde{w}_\ell}{\lambda}, \quad \forall \ell \in \{0, L\}.
\]

(39)
We can find the values of keep_ε and alt_ε from Eq. (39) in a sequential way. At any step, let \( \mathcal{L} \) denote the set of \( \epsilon \) for which we have already found these values. Then, we can rewrite Eq. (39) as

\[
\text{keep}_\epsilon + \sum_{k \in \mathcal{L} | \text{alt}_\epsilon = \epsilon}(2^n - \text{keep}_k) = \tilde{\rho}_\epsilon - \sum_{k \in \mathcal{L} | \text{alt}_\epsilon = \epsilon}(2^n - \text{keep}_k) = \hat{\rho}_\epsilon. \tag{40}
\]

This expression involves only known quantities on the right-hand side, which we call \( \hat{\rho}_\epsilon \) for short. We show by induction that the average of \( \hat{\rho}_\epsilon \) for \( \epsilon \notin \mathcal{L} \) is in \( 1/2 \), and the \( \hat{\rho}_\epsilon \) are non-negative. These are clearly true initially because then \( \hat{\rho}_\epsilon = \tilde{\rho}_\epsilon \). Now, assume that these conditions are true at some step. If the values \( \tilde{\rho}_\epsilon \) are all equal for \( \epsilon \notin \mathcal{L} \), then we can just take \( \text{alt}_\epsilon = \epsilon \) and any value of \( \text{keep}_\epsilon \), and satisfy Eq. (40) for all remaining \( \epsilon \). Otherwise, there will be one value, \( \epsilon_0 \), where \( \tilde{\rho}_{\epsilon_0} \) is below the average \( 1/2 \) and another, \( \epsilon_1 \), where \( \tilde{\rho}_{\epsilon_1} \) is above \( 1/2 \). For \( \epsilon_0 \), we choose \( \text{keep}_{\epsilon_0} = 2^n L \tilde{\rho}_{\epsilon_0} \) and \( \text{alt}_{\epsilon_0} = \epsilon_1 \). Then, \( \text{alt}_0 \) is added to the set \( \mathcal{L} \), and the values of \( \hat{\rho}_\epsilon \) are updated. According to Eq. (40), the only value of \( \tilde{\rho}_\epsilon \) that is updated is that for \( \epsilon = \epsilon_1 \), where we replace it with \( \tilde{\rho}_{\epsilon_1} + \tilde{\rho}_{\epsilon_0} - 1/L \). This ensures that the average value of \( \tilde{\rho}_\epsilon \) for \( \epsilon \notin \mathcal{L} \) is still \( 1/2 \), and since we had \( \tilde{\rho}_{\epsilon_1} > 1/L \), the new value is non-negative.

A more intuitive way to understand our approach to preparation is that it is equivalent to classical alias sampling [72], which samples \( \epsilon \) with probability \( \tilde{\rho}_\epsilon \) by the following procedure:

1. Select \( \epsilon \) uniformly at random from \( [0, L) \).
2. Look up \( \text{alt}_\epsilon \) and \( \text{keep}_\epsilon \).
3. Return \( \epsilon \) with probability \( \text{keep}_\epsilon / 2^n \); otherwise return \( \text{alt}_\epsilon \).

The procedure for determining the \( \text{alt}_\epsilon \) and \( \text{keep}_\epsilon \) is then to work backwards starting from the distribution \( \tilde{\rho}_\epsilon \) and update this distribution by shifting probabilities from \( \epsilon_1 \) to \( \epsilon_0 \) until we obtain a uniform distribution [73].

This procedure is illustrated in Fig. 13. One starts with a histogram of the desired distribution and looks for a bar that is too small, fixes this by transferring probability from a bar that is too high, and so on until all bars have the correct height. Each probability transfer permanently solves the bar that was too low, and the remaining bars form a smaller instance of the same problem. Thus, it is not possible to get stuck in a loop or a dead end. See also the module util/ _lcu_util.py in version 0.6 of OPENFERMION [74,75] for open-source python code that performs this iterative matching process (and also handles discretizing the distribution) in \( O(L) \) time.

**IV. CONSTRUCTIONS FOR THE ELECTRONIC STRUCTURE HAMILTONIAN**

Using the appropriate discretization into a basis of \( N \) spin orbitals, the electronic structure Hamiltonian can be written as
\[
H = \sum_{p,q,\sigma} T(p-q)a_{p,\sigma}^\dagger a_{q,\sigma} + \sum_{p,\sigma} U(p)n_{p,\sigma} \\
+ \sum_{(p,a)\neq(q,\beta)} V(p-q)n_{p,a}n_{q,\beta}, \quad (41)
\]

where \(a_{p,\sigma}^\dagger\) and \(a_{p,\sigma}\) are fermionic creation and annihilation operators on spatial orbital \(p \in \{0, \ldots, N/2 - 1\}\) with spin \(\sigma \in \{\uparrow, \downarrow\}\), and \(n_{p,\sigma} = a_{p,\sigma}^\dagger a_{p,\sigma}\) is the number operator. These operators satisfy the canonical fermionic anti-commutation relations \(\{a_{p,\alpha}^\dagger, a_{q,\beta}\} = \{a_{p,\alpha}, a_{q,\beta}\} = 0\) and \(\{a_{p,\alpha}^\dagger, a_{q,\beta}\} = \delta_{p,0}\delta_{\alpha,\beta}\).

Mapping to qubits under the Jordan-Wigner transformation \([61,76]\), Eq. (41) becomes

\[
H = \sum_{p \neq q,\sigma} \frac{T(p-q)}{2}(X_{p,\sigma}Z_{q,\sigma} + Y_{p,\sigma}Z_{q,\sigma}) \\
+ \sum_{(p,a)\neq(q,\beta)} \frac{V(p-q)}{4} Z_{p,a}Z_{q,\beta} \\
- \sum_{p,\sigma} \left(\frac{T(0) + U(p)}{2} \right) Z_{p,\sigma} \\
+ \sum_{p} \left(\frac{T(0) + U(p)}{2} \right) 1, \quad (42)
\]

where we have introduced the notation \(Z\) that will be used throughout the paper, which we now explain. The tensor factors on which Pauli operators act can always be interpreted as some integer. For instance, \((p, \sigma)\) is mappable to an integer under a particular choice of canonical ordering in the Jordan-Wigner transformation. When placed between two Pauli operators, the notation \(A_j Z_{j+1} \ldots Z_{j-k} A_k\) denotes the operator \(A_j Z_{j+1} \ldots Z_{j-k} A_k\). The exact mapping between a spin orbital indexed by \((p, \sigma)\) and a qubit indexed by an integer is discussed later.

The forms of Eq. (41) and Eq. (42) encompass a wide range of fermionic Hamiltonians, including the molecular electronic structure (aka “quantum chemistry”) Hamiltonian in any basis that diagonalizes the Coulomb potential \([39]\). The particular coefficients will depend on the discretization scheme and basis functions chosen to represent the system. One such representation, derived for use in quantum simulations in Ref. \([39]\), prescribes the coefficients

\[
T(p) = \sum_{k} \frac{k^2 \cos(k \cdot r_p)}{2N},
\]

\[
U(p) = -\sum_{j \neq \beta} \frac{4\pi \xi_j \cos(k \cdot R_j - k_v \cdot r_p)}{\Omega k_v^2},
\]

\[
V(p) = \sum_{i \neq 0} \frac{2\pi \cos(k \cdot r_p)}{\Omega k_v^2}, \quad (43)
\]

where each spatial orbital \(p\) is associated with an orbital centroid \(r_p = p(2\Omega/N)^{1/3}\), and \(\Omega\) is the computational cell volume. The momentum modes are defined as \(k_v = 2\pi \nu/\Omega^{1/3}\), with \(\nu \in [-(N/2)^{1/3}, (N/2)^{1/3}]\). When dealing with molecular potentials, \(R_j\) and \(\xi_j\) are the position and charge of the \(j\)th nucleus, respectively.

As discussed in Ref. \([39]\), the Hamiltonian of Eq. (43) corresponds to discretization in a basis composed of rotated plane waves known as the “plane wave dual” basis. The basis set discretization error associated with the dual basis is asymptotically equivalent to a Galerkin discretization using any other single-particle basis functions, including Gaussian orbitals \([39]\). Thus, Eq. (43) is a general expression of the electronic structure problem that is asymptotically equivalent to any other representation. While well suited for simulating periodic materials, despite asymptotic equivalence, this basis set is not particularly compact for the simulation of molecules. Another basis set compatible with Eq. (41) and Eq. (42), while being much more appropriate for molecules, is the so-called “Gausslet” basis \([41]\). Gausslets are derived from a ternary wavelet transformation \([77]\) of Gaussian orbitals and have similar intrinsic basis set discretization errors to standard Gaussian orbitals \([41]\).

The simulation procedures here will make use of the structure in Eq. (42). Specifically, our algorithm will make use of the fact that the Hamiltonian consists of only four types of terms—\(Z_p, Z_p Z_q, X_p Z X_q\), and \(Y_p Z Y_q\)—and that there are only \(3N/2\) unique values of the coefficients. Our algorithms do not utilize any particular structure in the dual basis Hamiltonian in Eq. (43) beyond the fact that it satisfies the form of Eq. (41). This is important since it implies that the techniques of this paper are compatible with other representations of the electronic structure Hamiltonian, such as the finite difference discretization \([39]\), finite element methods, and Gausslet basis sets \([41]\), which produce Hamiltonians consistent with Eq. (41) but not Eq. (43).

## A. Electronic structure Hamiltonian selection oracle

In order to implement the \texttt{SELECT} and \texttt{PREPARE} oracles for the electronic structure Hamiltonian of Eq. (41), one must first define a scheme for indexing all of the terms. For the case of the general electronic structure Hamiltonian in Eq. (41), we index terms with the registers \(|\theta\rangle, |U\rangle, |V\rangle, |p\rangle, |a\rangle, |q\rangle, \text{and} |\beta\rangle\). The \(|p\rangle\) and \(|q\rangle\) registers are little-endian binary encodings of integers going from 0 to \(N/2 - 1\), thus using \(\log N - 1\) qubits each; the other registers are each a single bit, which we use to specify the unitary that \texttt{SELECT} should apply to the system register \(|\psi\rangle\).

The \(|a\rangle\) and \(|\beta\rangle\) bits are used to specify the spins \(\{\uparrow, \downarrow\}\), which, together with the spatial orbital specifications \(p\) and \(q\), index a spin orbital. Thus, a register set as \(|p\rangle|a\rangle|q\rangle|\beta\rangle\)
will index a Hamiltonian term that involves action on the spin orbitals indexed by \((p, \sigma)\) and \((q, \beta)\). Next, whenever \(|U| = 1\), it will be the case (by construction of our circuits) that \((p, \alpha) = (q, \beta)\), and we will apply the \(Z_{p,\alpha}\) terms. If \(|V| = 1\), we will apply the \(Z_{p,\alpha}Z_{q,\beta}\) terms. If \(|U| = 0, |V| = 0\) and \(p < q\), it will also be the case that \(\alpha = \beta\), and we will apply the \(X_{p,\alpha}\) terms; if \(|U| = 0, |V| = 0\) and \(p > q\), it will again be the case that \(\alpha = \beta\), and we will apply the \(X_{q,\alpha}\) terms. Finally, the \(|\theta\rangle\) register encodes whether the unitary should have a negative phase (if \(|\theta\rangle = |1\rangle\)). Thus, our SELECT circuit meets the following specification (where UNDEFINED means this case should not occur):

\[
\text{SELECT}_\text{CHEM}|\theta, U, V, p, \alpha, q, \beta\rangle|\psi\rangle = (-1)^q|\theta, U, V, p, \alpha, q, \beta\rangle
\]

\[
\begin{cases}
Z_{p,\alpha}|\psi\rangle & \text{if } U \wedge -V \wedge ((p, \alpha) = (q, \beta)) \\
Z_{p,\alpha}Z_{q,\beta}|\psi\rangle & \text{if } -U \wedge V \wedge ((p, \alpha) \neq (q, \beta)) \\
X_{p,\alpha}\overline{Z}X_{q,\alpha}|\psi\rangle & \text{if } -U \wedge -V \wedge (p < q) \wedge (\alpha = \beta) \\
Y_{q,\alpha}\overline{Z}Y_{p,\alpha}|\psi\rangle & \text{if } -U \wedge -V \wedge (p > q) \wedge (\alpha = \beta) \\
\text{UNDEFINED} & \text{otherwise.}
\end{cases}
\]

(44)

We present our implementation of SELECTCHEM in Fig. 14. Our circuit relies on the subroutines that we describe in Sec. III A, which provide a method for selectively applying strings of Pauli operators to a system register of size \(N\), with controls on log \(N\) qubits. Important notation for these subroutines is also defined in Sec. III A, and thus, that section is necessary for understanding the details of Fig. 14.

Since \(p\) and \(q\) are actually three-dimensional vectors with elements taking integer values \(p \in [0, (N/2)^{1/3} - 1]\) and \(\sigma \in \{\uparrow, \downarrow\}\), we should clarify how the spin orbitals \((p, \sigma)\) are mapped to an integer representing qubits. For ease of exposition, we define the following mapping function for a \(D\)-dimensional system,

\[
M \equiv (N/2)^{1/D}, \quad f(p, \sigma) = \delta_{\sigma,1}M^D + \sum_{j=0}^{D-1} p_jM^j,
\]

where \(D = 3\) for chemistry and \(D = 2\) for the Hubbard model. The \(\delta\) function behaves as one might expect: \(\delta_{\uparrow, \downarrow} = 0\) and \(\delta_{\downarrow, \downarrow} = 1\). Thus, it should be understood that \(X_{p,\sigma}\) implies the \(X\) operator acting on qubit \(f(p, \sigma)\).

**B. Electronic structure coefficient preparation oracle**

We see from Eq. (41) that there are only \(O(N)\) unique coefficients in the Hamiltonian, despite the Hamiltonian having \(O(N^2)\) different terms. Based on the indexing in Eq. (44) and definition in Eq. (6), our PREPARE initializes

![FIG. 14. SELECTCHEM circuit, with a T count of 12N + 8 log N + \(O(1)\), which implements the functionality specified by Eq. (44), conditioned on a “control” qubit. The unitaries performing Majorana and indexed operations (each requiring 4N T gates) are explicitly constructed in Sec. III A. As described in Fig. 9, the unitaries labeled as \(ZA_j\) apply the operation \(Z_0…Z_{j-1}A_j\) to the target register, depending on the value from the input \(|p\rangle\) register. These operations require an extra log \(N\) ancillae, so the overall circuit spans \(N + 3 \log N + O(1)\) qubits. The operation that targets the system register with \(Z_q\) is a variant of Fig. 7, with the \(X_\epsilon\) gate replaced by \(Z_\epsilon\). All indexed operations reuse the same ancillae.](041015-17)
\[
\text{PREPARE}_{\text{CHEM}} |0\rangle \otimes (3+2\log N) \\
\mapsto \sum_{p, \sigma} \hat{U}(p)|\theta_p\rangle|1\rangle_U|0\rangle_V|p, \sigma, p, \sigma\rangle \\
+ \sum_{p, q, \sigma} \hat{\tilde{T}}(p-q)|\theta^{(0)}_{p-q}\rangle|0\rangle_U|0\rangle_V|p, \sigma, q, \sigma\rangle \\
+ \sum_{(p, \alpha) \neq (q, \beta)} \hat{V}(p-q)|\theta^{(1)}_{p-q}\rangle|0\rangle_U|1\rangle_V|p, \alpha, q, \beta\rangle, \tag{46}
\]

where the values of the coefficients and the state of $|\theta\rangle$, related to the coefficients in Eq. (42), are defined as

\[
\hat{U}(p) = \sqrt{\frac{|T(0) + U(p) + \sum_q V(p-q)|}{2\lambda}}, \\
\hat{\tilde{T}}(p) = \sqrt{\frac{|T(p)|}{\lambda}}, \\
\hat{V}(p) = \sqrt{\frac{|V(p)|}{4\lambda}}, \\
\theta_p = \frac{1 - \text{sign}(-T(0) - U(p) - \sum_q V(p-q))}{2}, \\
\theta^{(0)}_p = \frac{1 - \text{sign}(T(p))}{2}, \\
\theta^{(1)}_p = \frac{1 - \text{sign}(V(p))}{2}. \tag{47}
\]

The $T(p)$ coefficient inside the square root in Eq. (47) differs from the coefficient in Eq. (42) by a factor of 2 since it occurs for each type of term only once depending on whether $p < q$ or $p > q$.

To implement PREPARE, we first synthesize a unitary referred to as SUBPREPARE, which acts as follows:

\[
\text{SUBPREPARE}|0\rangle \otimes (2+\log N) \\
\mapsto \sum_{d=0}^{N-1} (\hat{U}(d)|\theta_d\rangle|1\rangle_U|0\rangle_V + \hat{\tilde{T}}(d)|\theta^{(0)}_d\rangle|0\rangle_U|0\rangle_V \\
+ \hat{V}(d)|\theta^{(1)}_d\rangle|0\rangle_U|1\rangle_V)|d\rangle. \tag{48}
\]

Since in this step we initialize a state on $O(\log N)$ qubits, the techniques of Ref. [69] would allow one to implement SUBPREPARE with a $T$ count of $O(N \log (1/e))$. However, in Fig. 15, we show an even more efficient method for synthesizing SUBPREPARE with $T$-gate complexity $O(N + \log(1/e))$, based on the techniques introduced in Sec. III D. Using SUBPREPARE, we can implement the entire PREPARE circuit with the same asymptotic $T$ complexity. In our SUBPREPARE circuit, $\varepsilon$ is really a vector of integers; thus, we use “modular vector indices” such that if $v$ is a three-dimensional vector within a rectangular space with each dimension having $M$ values, then the function application $F(v)$ should be expanded to $F(v) = F(v \mod M) = F(v_x \mod M, v_y \mod M, v_z \mod M)$, consistent with the mapping introduced in Eq. (45).

While applying SUBPREPARE to create the state in Eq. (48), we also initialize the $|\alpha\rangle$ qubit in the $|+\rangle$ state with a Hadamard. We then use the UNIFORM$_{\mu}$ circuit from Fig. 12 to initialize the $|\mu\rangle$ register in an equal superposition in a way that is controlled on the $|\mu\rangle$ ancilla qubit being in the state $|0\rangle_U$. Subsequent to this step, the state becomes

---

**FIG. 15.** SUBPREPARE circuit for the electronic structure Hamiltonian, as in Fig. 16, with a $T$ count of $6N + O(\mu + \log N)$ and a qubit count of $2\mu + 3\log N + O(1)$, where $\mu$ is defined in Eq. (36). The data-loading subroutine is implemented as in Fig. 10 and has a $T$ count of $3 \times 4M^3 - 4 = 6N - 4$. The UNIFORM subroutine is implemented as in Fig. 12 and has a $T$ count of $O(\mu)$. The compare-and-swap operations have a negligible $O(\log N)$ $T$ count. As in Eq. (45), $D$ denotes the system dimension (usually $D = 3$), and $M$ refers to the number of values among each dimension such that $N = 2M^D$. Although we only specify the behavior of the circuit when the $U$, $V$, and $p$ qubits start in the $|0\rangle$ state, the circuit is also invoked in contexts where this is not the case.
The register labeled as $|d\rangle$ in Eq. (48) will ultimately become our $|p\rangle$ register, but immediately after SUBPREPARE, it is more appropriate to think of it as encoding a value $|p - q\rangle$. As we can see in Eq. (46), when $|V\rangle = |1\rangle$ and $p = q$, it is necessarily the case that $\alpha \neq \beta$. The middle part of our PREPARE circuit is dedicated to correctly initializing this tricky part of the superposition. To do this, we use an ancilla to apply a Hadamard gate to $|0\rangle_\beta$ only when $|V\rangle = |1\rangle$ and $|d\rangle \neq |0\rangle \otimes \log N$. In the event that $|V\rangle = |1\rangle$ and $|d\rangle = |0\rangle \otimes \log N^{-1}$, we apply a CNOT gate with an open control on $|\alpha\rangle$ which targets $|0\rangle_\beta$, thus ensuring that $|\beta\rangle \neq |\alpha\rangle$ when $p - q = 0$. Then, we set $|\beta\rangle = |\alpha\rangle$ for the $U$ and $T$ part of the superposition by applying a Toffoli gate with regular control on $|\alpha\rangle$ and open control on $|V\rangle$, targeting $|\beta\rangle$. After these operations, the state can be expressed as

$$
\text{Eq. (49)} \mapsto \sum_{d = 0}^{N/2 - 1} \sum_{\sigma} \left( \tilde{U}(d)|\theta_d\rangle|1\rangle_U|0\rangle_V |d, \sigma, 0, \sigma\rangle + \sum_{q = 0}^{N/2 - 1} \tilde{T}(d)|\theta_d^{(0)}\rangle|0\rangle_U|0\rangle_V |d, \sigma, q, \sigma\rangle \right) + \sum_{\rho = 1}^{N/2 - 1} \sum_{d = 0}^{N/2 - 1} \left( \tilde{V}(d)|\theta_d^{(1)}\rangle|0\rangle_U|1\rangle_V |d, \alpha, q, \beta\rangle \right). 
$$

The final step consists of converting the $|d\rangle$ register to values representing $|p\rangle$. To do this, we must add the $|q\rangle$ register into the $|d\rangle$ register when $|U\rangle = |0\rangle$ so that $|d + q\rangle = |p - q + q\rangle = |p\rangle$. However, we also want to copy the $|d\rangle$ register into the $|q\rangle$ register when $|U\rangle = |1\rangle$; thus, prior to this operation, we also implement a Fredkin gate, which swaps $|d\rangle$ and $|q\rangle$, conditioned on $|U\rangle = |1\rangle$. After the Fredkin gate and the addition of $|d\rangle$ into $|q\rangle$,

$$
\text{Eq. (50)} \mapsto \sum_{d = 0}^{N/2 - 1} \sum_{\sigma} \left( \tilde{U}(d)|\theta_d\rangle|1\rangle_U|0\rangle_V |d, \sigma, 0, \sigma\rangle + \sum_{q = 0}^{N/2 - 1} \tilde{T}(d)|\theta_d^{(0)}\rangle|0\rangle_U|0\rangle_V |d, q, \sigma, \sigma\rangle \right) + \sum_{\rho = 1}^{N/2 - 1} \sum_{d = 0}^{N/2 - 1} \left( \tilde{V}(d)|\theta_d^{(1)}\rangle|0\rangle_U|1\rangle_V |d + q, \alpha, q, \beta\rangle \right). 
$$

Then, simply by relabeling $d = p - q$ whenever $|U\rangle = |0\rangle$ and $d = p$ whenever $|U\rangle = |1\rangle$, we see that our state is identical to the desired one [from Eq. (46)]. We show how to use SUBPREPARE to implement PREPARE_{TEM} in Fig. 16. The gate complexity of SUBPREPARE is $O(N + \log(1/\epsilon))$, and the gate complexity of all other components of this circuit is $O(\log N)$. Thus, the overall gate complexity of PREPARE is $O(N + \log 1/\epsilon)$.

### C. Resources required for electronic structure simulation

The parameter $\lambda$ from Eq. (6) has significant implications for the complexity of our algorithm; as seen in Eq. (27), our circuit size will scale linearly in $\lambda$. For the case of general electronic structure, we can see from Eq. (41) that $\lambda$ is

$$
\lambda = \sum_{pq} T(p - q) + \sum_p |U(p)| + \sum_{p \neq q} |V(p - q)|. 
$$

This expression and the extremely naive assumption that all coefficients are $O(1)$ would imply that $\lambda \in \mathcal{O}(N^2)$. For the case of quantum chemistry in the dual basis, i.e., Eq. (43), the work of Ref. [39] obtains the same bound:

$$
\lambda \in \mathcal{O}\left(\frac{N^{7/3}}{\Omega^{1/3}} + \frac{N^{5/3}}{\Omega^{2/3}}\right) \in \mathcal{O}(N^2), 
$$

where the last relation holds when studying electronic structure systems that grow with fixed density $N \propto \Omega$, which is the usual situation. For encoding the electronic structure Hamiltonian, we also determine that $P = 6N + O(\log(N/\epsilon))$ and $S = 12N + O(\log N)$ in terms of $T$ complexity. Thus, from Eq. (27), we can conclude that the overall $T$ complexity of our procedure is roughly

$$
\frac{\sqrt{2\pi \lambda(S + 2P)}}{\Delta E} \approx 24\sqrt{2\pi \lambda}^{-N}. 
$$
Ancilla required are electron gas, also known as jellium. Jellium is a system intractable is a molecule without nuclei: the uniform material, and whether one scales toward continuum or the bases used, the geometry and atomic composition of the simulations of jellium typically introduce a bias to control the sign problem, such as the fixed-node approximation, full-configuration quantum Monte Carlo with initiators, or auxiliary-field quantum Monte Carlo with a constrained phase bias. The systematic error from these biases is thought to be as large as half a percent in the energy of \( \eta \) electrons with real kinetic energy and Coulomb interactions confined to a box of finite volume \( \Omega \) with periodic boundary conditions. Plane waves are a near-ideal basis for the simulation of jellium; the system is naturally expressed using the discretization of Eq. (43) with a constant external potential, i.e., \( \zeta_j = 0 \). Jellium is an interesting system to simulate on early quantum computers due to its simplicity, classical intractability [39], historical significance tied to breakthroughs in density functional theory [78] as well as the fractional quantum Hall effect [79], and tradition as a benchmark for classical electronic structure calculations.

The phase diagram of jellium is typically parametrized in terms of the Wigner-Seitz radius, which characterizes the electron density in three dimensions as \( r_s = (3\Omega/(4\pi))^{1/3} \), where \( q \) is the number of electrons. Although the ground state of jellium at high densities (metallic, \( r_s \sim 1 \) Bohr radii per particle) and at very low densities (insulating, \( r_s \sim 100 \) Bohr radii per particle) is well known, the phase diagram in the intermediate density regime is less certain [80–85]. Whereas perturbation theory performs well in the high-density regime [86,87], quantum Monte Carlo has been the most competitive simulation tool in the low- to intermediate-density regimes [88–91]. For systems with more than 50 electrons, quantum Monte Carlo simulations of jellium typically introduce a bias to control the sign problem, such as the fixed-node approximation, full-configuration quantum Monte Carlo with initiators, or auxiliary-field quantum Monte Carlo with a constrained phase bias.

Perhaps the simplest chemistry system that is classically intractable is a molecule without nuclei: the uniform electron gas, also known as jellium. Jellium is a system
[81,88], which is on a scale similar to the energy difference between competing phases in the intermediate-density regime. Even for modest system sizes such as 50 electrons and twice as many spin orbitals, quantum simulations can offer bias-free results that cannot be obtained by quantum Monte Carlo.

We include numerics in Fig. 17 that empirically estimate a tighter bound on $\lambda$ for jellium in the classically challenging regime corresponding to $r_s = 10$ Bohr radii at half-filling $N = [2\eta]$. Those numerics, shown in Fig. 17, indicate an empirical scaling of $\lambda = \mathcal{O}(\sim N^{5/3})$. If we target a chemical accuracy of $\Delta E = 0.0016$ Hartree, then from Eq. (54), we see that roughly $2 \times 10^7$ T gates would be required for jellium with 54 orbitals, $2 \times 10^8$ T gates would be required for jellium with 128 orbitals, and about a billion T gates would be required for jellium with 250 orbitals. While these numbers are promising, for small sizes, these simulations require a number of ancilla comparable to $N$. T counts and ancilla resources are tabulated for several jellium problem instances in Table III.

The dual basis of Eq. (43) is also a natural choice for periodic condensed phase systems (e.g., solids) besides jellium. Considering only this basis, there are two parameters that determine the accuracy of the simulation with respect to the true material. The first one is the number of plane waves used to discretize the cell, which determines the spacing of the quasipoints in the dual basis [39]. More plane waves equate to a finer grid and more accurate discretization. The second parameter is the size of the supercell, which determines the error one incurs by representing an infinite system with a finite, periodic one, also known as the finite-size error. There are different ways of reducing the finite-size error for a physical system. One common method used in density functional theory utilizes Bloch’s theorem to divide the sampling problem into so-called “k-points” within the first Brillouin zone [92]. The smoothness of the energy with respect to the k-points and additional symmetry provided can offer advantages in certain approaches at the cost of increased complexity, often resulting in a complex Hamiltonian representation at nonzero k-points. The origin k-point, also called the gamma point, maintains a real Hamiltonian for the appropriate basis functions. An alternative to k-point sampling is increasing the size of the supercell, which increases the relevance of the gamma point. For simplicity, here we only consider the gamma point, so the natural parameter to change is the number of repetitions of the unit cell that fixes the size of the supercell being simulated. A larger supercell tends to incur less finite-size error as the system is scaled to the thermodynamic limit.

It is clear that these two parameters are not entirely independent with respect to the accuracy of representation of the true system. For example, a much larger supercell with the same number of grid points clearly offers a coarser and less accurate representation of the true system. Moreover, in classical methods, it is common practice to extrapolate along both parameters to increase the accuracy for a given computational cost [92]. We do not introduce such complexities here but rather show empirically how the choice of these parameters influences the parameter $\lambda$, which determines the cost of our algorithms, leaving optimizations such as extrapolation schemes to future work.

Figure 18 shows the value of $\lambda$ as a function of the number of qubits being used to discretize the material cell at a fixed supercell size for several real materials.
The number of qubits here is equal to twice the number of plane waves since spin is being treated explicitly. Equal numbers of plane waves along each of the reciprocal lattice vectors of the supercell are used, as opposed to the more common spherical energy cutoff schemes [93], as this enables the use of the plane-wave dual basis [39]. The exponent (slope on the log-log plot) of a least-squares linear regression to the data is listed alongside the material, and we see empirically that the value of $\lambda$ scales just under $\lambda = \mathcal{O}(N^2)$ in the number of basis functions while keeping the size of the supercell fixed, which matches the analytical bound rather closely. From the form of the Hamiltonian in Eq. (43), one can see that at a fixed number of plane waves, increasing the volume of the supercell $\Omega$ tends to decrease $\lambda$ such that $\lambda = \mathcal{O}(\sim \Omega^{-1/2})$, due to representing lower-frequency modes with respect to the plane-wave representation of the kinetic energy, despite increasing the total nuclei present. We show this effect empirically in the center of Fig. 18.

The first two panels of Fig. 18 leave open the question of the impact on $\lambda$ of increasing the supercell size while maintaining a constant density of dual quasipoints or a constant density of plane waves, as we expect the impact of the last two aspects to compete in some way. Empirically, this is shown in the right portion of Fig. 18, which plots the values of $\lambda$ for a fixed density of points in increasing supercell sizes. Note that this scaling is most comparable to past studies on single molecules since molecular volume tends to grow as one adds electrons. We observe that the scaling in this case is more favorable as a function of the number of qubits than simply refining the grid alone, and in all cases, it is better than $\lambda = \mathcal{O}(\sim N^{3/2})$. From Eq. (54), this numerical data would suggest that the $T$ complexity of our overall algorithm is empirically bounded by $\mathcal{O}(\sim N^{5/2}/\Delta E)$ when the goal is simulation of real materials.

To treat molecules properly, one should further consider pseudopotentials [94], methods of extrapolation to continuum and thermodynamic limits, and embedding methods [95,96]. We leave such a thorough comparison of fault-tolerant resources required for specific instances of real materials other than jellium to future work. However, by comparing Fig. 17 and the right panel of Fig. 18, it is apparent that jellium is a reasonable proxy for other materials in that $\lambda$ values are comparable. As the rest of the simulation circuit is identical up to the particular angles of certain single-qubit rotations, one can estimate the cost of simulating these materials from our analysis of the fault-tolerant overheads required to simulate jellium in Sec. VI.

V. CONSTRUCTIONS FOR THE HUBBARD MODEL

In this section, we describe specialized implementations of the SELECT and PREPARE oracles for simulation of the planar repulsive-interaction Fermi-Hubbard model; we then estimate the overall $T$ complexity of simulating such models. The Hubbard model is a canonical model of a many-electron system often used to model superconductivity in cuprate superconductors. Despite its simplicity, the Hubbard model exhibits a wide range of correlated electron behavior including superconductivity, magnetism, and interaction-driven metal-insulator transitions [97].

The Hubbard model is essentially a special case of Eq. (41) when the model is restricted to a planar grid. The Hamiltonian can be expressed as

$$H = -t \sum_{\langle p,q \rangle, \sigma} a_{p,\sigma}^\dagger a_{q,\sigma} + \frac{U}{2} \sum_{p,\sigma \neq \beta} n_{p,\sigma} n_{p,\beta},$$

where $t$ is the hopping parameter, $U$ is the on-site repulsion energy, and $\mathcal{H}$ is the Hamiltonian. The $T$ complexity of simulating this Hamiltonian can be estimated using the techniques described in Sec. V.
where the notation \((p, q)\) implies that terms exist only between sites that are adjacent on a planar lattice with periodic boundary conditions. This Hamiltonian can be expressed under the Jordan-Wigner transformation as

\[
H = -\frac{t}{2} \sum_{\langle p, q \rangle, \sigma} (X_{p,\sigma} \tilde{Z} X_{p,\sigma} + Y_{p,\sigma} \tilde{Z} Y_{p,\sigma}) + \frac{u}{8} \sum_{p, \sigma \neq \beta} Z_{p,\sigma} Z_{p,\beta} - \frac{u}{4} \sum_{p, \sigma} Z_{p,\sigma} + \frac{uN}{4}. \tag{57}
\]

We focus on the Hubbard model with periodic boundary conditions (which is a more typical system to study than the Hubbard model with open boundary conditions).

A. Hubbard model Hamiltonian selection oracle

We see from Eq. (56) that there are only three unique coefficients in the Hubbard Hamiltonian: The coefficient of \(-XZX\) terms is \(t/2\), the coefficient of \(-YZY\) terms is \(u/8\), and the coefficient of local \(-Z\) terms is \(u/4\). This makes the implementation of the PREPARE circuit exceptionally simple. Ultimately, we show that the PREPARE circuit for the Hubbard model can be implemented at a cost of \(O(\log(1/\epsilon))\). This scaling virtually guarantees that for all problem sizes of interest, the scaling of the overall algorithms will be dominated by the cost of the SELECT circuit.

We index terms in the Hubbard Hamiltonian using the registers \(|U\rangle|V\rangle|p_x\rangle|p_y\rangle|\alpha\rangle|\beta\rangle\). Note that it is important for us to explicitly separate \(p_x\) and \(p_y\) in our construction of the Hubbard model circuits since this structure is fundamental to the efficiency of our scheme. Our indexing scheme is nearly identical to the scheme used for the arbitrary chemistry Hamiltonian in Eq. (7), but here we do not need the \(\theta\) parameter since we know the sign of the parameters in advance. Thus, our SELECT circuit for the Hubbard model will meet the following specifications:

\[
\text{SELECT}_\text{HUB} |U, V, p, \alpha, q, \beta\rangle |\psi\rangle = |U, V, p, \alpha, q, \beta\rangle \otimes \begin{cases} -Z_{p,\alpha}|\psi\rangle & U \land \neg V \land ((p, \alpha) = (q, \beta)) \\ Z_{p,\alpha}Z_{q,\beta}|\psi\rangle & \neg U \land V \land (p = q) \land (\alpha = 0) \land (\beta = 1) \\ -X_{p,\alpha} \tilde{Z} X_{q,\alpha}|\psi\rangle & \neg U \land \neg V \land (p < q) \land (\alpha = \beta) \\ -Y_{q,\alpha} \tilde{Z} Y_{p,\alpha}|\psi\rangle & \neg U \land \neg V \land (p > q) \land (\alpha = \beta) \\ \text{UNDEFINED} & \text{otherwise}, \end{cases}
\]

where, for ease of exposition, \(p = p_x + p_y M\) and \(q = q_x + q_y M\), consistent with the convention of Eq. (45) for \(D = 2\). By exploiting translational invariance in the Hubbard model, we are able to implement \(\text{SELECT}_\text{HUB}\) in a slightly more efficient fashion, achieving a \(T\) count of only \(10N + O(\log N)\). We show this more efficient implementation in Fig. 19.

B. Hubbard model coefficient preparation oracle

Our PREPARE circuit for the Hubbard model has the following specification:

\[
\begin{align*}
\text{PREPARE}_\text{HUB} |0\rangle^{\otimes (2 + 2\log N)} & \mapsto \sum_{p_x=0}^{M-1} \sum_{p_y=0}^{M-1} \left( \sqrt{\frac{u}{8\lambda}} |0\rangle_U |V\rangle |p_x\rangle |p_y\rangle |0\rangle_\alpha |p_x\rangle |p_y\rangle |1\rangle_\beta + \sqrt{\frac{u}{4\lambda}} \sum_{\sigma \in \{\pm\}} |1\rangle_U |0\rangle_V |p_x\rangle |p_y\rangle |\sigma\rangle |p_x\rangle |p_y\rangle |\sigma\rangle \\
& \quad + \sqrt{\frac{t}{2\lambda}} \sum_{\sigma \in \{\pm\}} \left( |0\rangle_U |0\rangle_V |p_x\rangle |p_y\rangle |\sigma\rangle |p_x + 1\rangle |p_y\rangle |\sigma\rangle + |0\rangle_U |0\rangle_V |p_x\rangle |p_y\rangle |\sigma\rangle |p_x + 1\rangle |\sigma\rangle \\
& \quad + \sqrt{\frac{t}{2\lambda}} \sum_{\sigma \in \{\pm\}} \left( |0\rangle_U |0\rangle_V |p_x\rangle |p_y\rangle |\sigma\rangle |p_x - 1\rangle |p_y\rangle |\sigma\rangle + |0\rangle_U |0\rangle_V |p_x\rangle |p_y\rangle |\sigma\rangle |p_x - 1\rangle |\sigma\rangle \right) \right), \tag{59}
\end{align*}
\]

where the first line above corresponds to \(-Z\) and \(ZZ\) terms, the second line corresponds to \(-XZ\) terms, and the final line corresponds to the \(-Y\) terms. Note that we are looking at a Hubbard model with periodic boundary conditions, so wherever something like \(|p_x + 1\rangle\) appears, we really mean \(|(p_x + 1) \mod M\rangle\), which we omitted from the above equation for clarity. Our implementation of \(\text{PREPARE}\) begins for the Hubbard model by
initializing a two-qubit state containing the three distinct coefficients for the $U$, $V$, and $T$ terms. This is done with standard circuit synthesis techniques with a T count of $O(\log(1/\epsilon))$ [69]. We then spread these coefficients over the various cases. We depict our implementation in Fig. 20.

C. Hubbard model resources

For the case of the planar Hubbard model in Eq. (56), it is readily apparent that

$$\lambda = 2Nt + \frac{Nu}{2} \in \mathcal{O}(N),$$

(60)
assuming that we are dealing with the spinful model with periodic boundary conditions. We also determine that \( P \in \mathcal{O}(\log(N/e)) \) and that \( S = 10N + \mathcal{O}(\log N) \). Thus, the total T cost of the Hubbard algorithm is

\[
\frac{\sqrt{2\pi}(S + 2P)}{\Delta E} = \frac{\sqrt{2\pi}(2t + u/2)N(S + 2P)}{\Delta E} \approx \frac{20\sqrt{2\pi} + 5\sqrt{2\pi}u}{\Delta E} N^3.
\]  

(61)

Ancillae required for our Hubbard model simulation come from two sources: qubits required for our entanglement-based phase estimation [see Eq. (24)] and ancillae actually required for our implementation of prepare and select, which for the Hubbard model is \( 3 \log N + \mathcal{O}(1) \). Putting these sources together, the total ancillae required are

\[
\log \left( \frac{\sqrt{2\pi}\lambda}{2\Delta E} \right) + 3 \log N + \mathcal{O}(1) = \log \left( \frac{\sqrt{2\pi}\lambda N^3}{2\Delta E} \right) + \mathcal{O}(1),
\]

(62)

where the additive constant is small and can usually be neglected for problem sizes of interest. This expression gives the ancillae count in Theorem 2.

While numerically exact solutions to the Hubbard model are available for one-dimensional [98] and infinite-dimensional systems [99], no known polynomial-time scaling classical methods can provide reliable solutions to the planar model in all parts of its phase diagram [97]. For state-of-the-art approximate methods, the most challenging low-temperature phase of the model appears to be the intermediate-interaction regime due to the presence of many competing phases, around \( u/t = 4 \) [97]. Accordingly, we focus our analysis on this regime. If \( u = 4t \), then \( \lambda = 4Nt \).

An interesting and classically challenging-to-obtain accuracy (beyond the agreement of state-of-the-art numerical methods [97]) for this regime would be in the vicinity of \( \Delta E \approx t/100 \) [100]; these choices would suggest a T complexity of approximately

\[
\frac{40\sqrt{2\pi}t}{t/100} N^2 < (1.8 \times 10^4) N^2
\]

(63)

and an ancilla count of approximately

\[
\log \left( \frac{8\sqrt{2\pi}tN^3}{t/100} \right) < 12 + 3 \log N.
\]

(64)

We summarize these resources for various interesting sizes of Hubbard model simulation in Table IV.

D. Exploiting locality in simulations of lattice Hamiltonians

Looking forward, another way that our circuits can be applied is to accelerate the recent Lieb-Robinson simulation method of Ref. [49]. Lieb-Robinson bounds reveal an intriguing fact about local Hamiltonians: Interactions spread out in a light cone similar in form to the causal diamonds used in relativity to indicate the regions of space-time that can have an impact on an event at a point in spacetime [101]. More specifically, Lieb-Robinson bounds show that information propagates at finite speeds (up to exponentially small errors) in systems with nearest-neighbor interactions. The idea behind Ref. [49] is to exploit this structure to break up the evolution into subpieces that can be independently simulated, thus reducing the cost of simulation.

We formalize this by envisioning that we have a lattice of \( N \) sites, \( \Lambda \), and a Hamiltonian that consists of terms that act upon these sites, \( H = \sum_{X \subseteq \Lambda} h_X \). Here, each \( h_X \) is local in that if \( h_X \) and \( h_Y \) act on different sites in the lattice, then \( \{h_X, h_Y\} = 0 \), and \( h_X \) only has support on sites that are a constant Euclidean distance away from each other. Note that this definition of locality also incorporates the terms within the Hubbard model. The final concept that we need in order to explain the method is that of distance between sites. We assume that for all \( X, Y \subseteq \Lambda \), dist\((X, Y)\) yields the minimum Euclidian distance between any two points within the lattice vectors contained within sets \( X \) and \( Y \). For example, given a lattice 1D on 10 sites, \( \text{dist}\{\{3, 4, 5\}, \{8, 9, 10\}\} = 3 \). The following lemma (a restatement of Lemma 6 in Ref. [49]) explains the impact that the locality imposed by the Lieb-Robinson bound has on simulation.

**Lemma 3** (patching lemma). Let \( \Lambda \) be a lattice on \( N \) sites with a Hamiltonian \( H = \sum_{X \subseteq \Lambda} h_X \), where each \( h_X \) is a local bounded Hamiltonian for every \( X \subseteq \Lambda \). Let \( A, B, C \) be subsets of \( \Lambda \), and let \( H_{P_1,\ldots,P_q} \) for any sequence

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Spin orbitals</th>
<th>Logical ancilla</th>
<th>Total logical</th>
<th>T count</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 × 6</td>
<td>72</td>
<td>33</td>
<td>105</td>
<td>9.3 × 10^7</td>
</tr>
<tr>
<td>8 × 8</td>
<td>128</td>
<td>33</td>
<td>161</td>
<td>2.9 × 10^8</td>
</tr>
<tr>
<td>10 × 10</td>
<td>200</td>
<td>36</td>
<td>236</td>
<td>7.1 × 10^8</td>
</tr>
<tr>
<td>20 × 20</td>
<td>800</td>
<td>42</td>
<td>842</td>
<td>1.2 × 10^10</td>
</tr>
</tbody>
</table>
$P: \{1, \ldots, q\} \mapsto \{A, B, C\}^q$ be for integer $q \geq 1$ $H_{P_1 \ldots P_q} = \sum_{X \in P_1 \cup \ldots \cup P_q} h_X$ (for example, $H_{AB} = \sum_{X \subseteq A \cup B} h_X$). There are constants $\nu \geq 0$, called the Lieb-Robinson velocity, and $\mu > 0$ such that

$$\| e^{-i H_{ABC t}} - e^{-i H_{AB} t} e^{i H_{BC} t} e^{-i H_{CD} t} \| \in O \left( \sum_{X \subseteq (A \cup B \cup C) \setminus (A \cup B) \cup C} \| h_X \| e^{\nu t - \mu \text{dist}(A,C)} \right).$$

(65)

Note that, in the above terminology, $(A \cup B \cup C) \setminus (A \cup B) \cup C$ is the boundary of the sets $AB$ and $C$, meaning the set of all terms within the Hamiltonian that act on sites contained in both $A$ or $B$ and $C$. Lemma 3 is the core of the simulation method. The central idea behind the proof is to use the patching lemma recursively to break up the evolution into a product of evolution operators, each of which contains terms that act on one or two of the constituent subsets of sites in the problem. This is conceptually similar to a Trotter decomposition; however, as the error in this approximation can be made exponentially small by choosing the patches in Lemma 3 to be linearly far apart, the error can be controlled in a tighter fashion without requiring short time steps (unlike Trotter decompositions [22,102]). For example, consider regions $A, B, C, D$. Then, we can write

$$e^{-i H_{ABC t}} \approx e^{-i H_{AB} t} e^{i H_{BC} t} e^{-i H_{CD} t} \approx e^{-i H_{AB} t} e^{i H_{BC} t} e^{-i H_{CD} t} e^{i H_{AC} t}.$$

(66)

In order to achieve scaling that is polylogarithmic in $1/\epsilon$, the evolution of each patch needs to be simulated using a method with polylogarithmic scaling in $1/\epsilon$, such as the truncated Taylor series simulation result [23] or qubitization [49]. Our circuits can be used to optimize this result since qubitization remains the best way to simulate the evolution, and our SELECT and PREPARE circuits meet the requirements of qubitization oracles. This result is formally given as Theorem 1 of Ref. [49], a special case of which is restated below for convenience.

**Theorem 4.** Assume the preconditions of Lemma 3 and that, for every unit ball in $\Lambda$ within the Euclidean metric space $\mathbb{R}^D$, at most $O(1)$ sites are contained within the ball and $h_X = 0$ if the diameter of the set $X$ is greater than 1. Additionally, let each $h_X$ be efficiently computable and have norm at most 1. Then, there exists a quantum algorithm that simulates the evolution of $H$ for time $\tau$ with accuracy $\epsilon$ that uses $O(\tau \text{polylog}(\tau N/\epsilon))$ 2-qubit gates and further has gate depth $O(\tau \text{polylog}(\tau N/\epsilon))$.

We claim that our approach can be used to achieve $\tilde{O}(N/\epsilon)$ scaling for simulating the Hubbard model with nearest-neighbor interactions. The Hubbard model satisfies the preconditions because each term in the Hamiltonian is local on the fermion lattice [49]. By using our constructions for the PREPARE and SELECT circuits, we can reduce constant factors (and some log factors in $T$ complexity) involved in the qubitized simulation while saturating the $\tilde{O}(\tau N)$ scaling of Theorem 4. We then choose $\tau \in O(1/\epsilon)$ and apply phase estimation on the result. In order to estimate the eigenvalue to within error $\epsilon$ with high probability, we need $O(1/\epsilon)$ repetitions of the circuit. Thus, by multiplying the two results, we find that the overall scaling for simulating such a Hubbard model is $\tilde{O}(N/\epsilon)$, as claimed.

This approach requires some follow-up work in order to determine exact $T$ counts. Specifically, we need to implement a full qubitized simulation (rather than $e^{-i \lambda \text{arccos}(H/\lambda)}$), This transformation is known to be achievable with a polylogarithmic-sized circuit [25]. While our work provides a highly optimized method for implementing the oracles needed in this process, more work remains to estimate constant factors associated with this simulation.

**VI. Resource Analysis for Fault-Tolerant Implementation**

Throughout this work, we have focused on the number of $T$ gates as the primary cost model of interest. The reasons for this are our focus on hardware consisting of a 2D nearest-neighbor coupled array of qubits, the intention to use the surface code [31–35], and the high relative overhead of $T$ gates compared to all others in that context. In this section, we discuss the overhead of the complete algorithm in detail.

When using the surface code, each $T$ gate is implemented by first preparing a magic state

$$|T\rangle \equiv |+\rangle = \frac{|0\rangle + e^{i\pi/4}|1\rangle}{\sqrt{2}},$$

(67)

that is consumed during the gate. The gate is probabilistic, and 50% of the time, $T^\dagger$ is actually applied instead of $T$. When the gate implemented is not as desired, an $S$ gate must be inserted to correct it. Preparing $T$ states requires a substantial amount of time and hardware, which we describe below. In an effort to minimize the number of physical qubits required, we therefore only prepare a

| TABLE V. Breakdown of the various elements that make up the Majorana operator circuit from Fig. 9 and the data lookup circuit from Fig. 10. Here, $N$ is the number of spin orbitals in the system that the circuits are being applied to. |
|----------------------------------------|-------|---------|---------|---------|
| Compute ANDs | Uncompute ANDs | Naked CNOTS | Subcircuits |
| Fig. 9 | $N - 1$ | $N - 1$ | $0.5N$ | $0.5N$ |
| Fig. 10 | $1.5N - 1$ | $1.5N - 1$ | $0.75N$ | $0.75N$ |
single T state at a time. We assume the availability of a correlated-error minimum-weight perfect matching decoder \[103\] capable of keeping pace with 1 $\mu$s rounds of surface code error detection, and capable of delivering feedforward in $10^{-20}$ $\mu$s. We calculate the qubit and time overhead for physical gate error rates $p = 10^{-3}$ and $p = 10^{-4}$.

The overhead is approximated by considering only the overhead of the Majorana operator circuit from Fig. 9 and the data lookup circuit from Fig. 10. It is expected that these circuits will account for over 90% of the total algorithm overhead. These circuits break down into a number of common pieces: compute ANDs, uncompute ANDs, naked CNOTs, and active subcircuits. The number of these pieces, in terms of the number of algorithm target qubits, $N$, is shown in Table V.

A surface code implementation of compute AND in Fig. 4 is shown in Fig. 21. The regular geometric structure can be decomposed into plumbing pieces, namely, cubic volumes each containing a single, small, light-colored cube. The compressed \[36\] version has a depth 15 plumbing pieces. The circumference of each stringlike structure (defect) is the surface code distance $d$, and the minimum separation of defects of the same color is also $d$. In the temporal direction (left-right), each unit of $d$ is a round of error detection. In the spatial directions (plane perpendicular to temporal), each unit of $d$ corresponds to two qubits. Note that a single CNOT, after compression, takes depth 1 plumbing piece as drawn. The overhead of any algorithm ultimately needs to be expressed as some number of qubits (space) and seconds (time). A plumbing piece is a convenient device-physics and code-distance independent measure of space-time volume. As described above, the $(5d/4)^3$ cubic volume of a plumbing piece can easily be converted to qubits and seconds, given a code distance $d$ and single-round error-detection time.

A plumbing piece depth 8 surface code implementation of uncompute AND in Fig. 4 is shown in Fig. 22. This could be compressed by performing the measurement differently so that no initial Hadamard would be necessary; however, the current surface code form is more easily identified with the original abstract form, and further compression is not necessary, as the execution time of the algorithm, as we shall see, is limited by our serial preparation of T states.

An effective plumbing piece depth 5 surface code implementation of the Majorana operator active subcircuit is shown in Fig. 23. The unusual pair-of-horns structure is
to permit an uncompute AND circuit to fit in before the final CNOT. The inner loop of the data circuit (Fig. 10) is just a pair of single-control multiple-target CNOTs, and a single additional CNOT. This case can be implemented in plumbing piece depth 4 and is not shown.

Preparing a T state is an involved process [106,107], which is shown for discussion purposes in Fig. 24. The important features for our purposes are the fact that this can be tiled vertically (meaning in time) every six plumbing pieces, and the whole structure occupies an area of 160 plumbing pieces. A significant amount of fast classical feedforward is required, as many T gates are potentially followed by S gates, and the paths of the connections from the first (small) level of distillation to the second (large) level must be determined based on which succeed. Our assumption of a 10–20-μs latency decoder is sufficient to make this work. We are interested in the overhead of solving instances of the electronic structure and Hubbard Hamiltonians discussed in prior sections. We must choose a target inaccuracy $\epsilon$ to fix the number of data logical qubits and gates required. To first order, the dependence of the gate count on $\epsilon$ can be ignored. We choose $\epsilon = 10^{-3}$. Table VI summarizes the circuit input parameters we will study.

Given Tables V and VI, and the plumbing piece depths of the various circuit elements, we can calculate the total number of data plumbing pieces $N_{\text{PL}}^{\text{data}}$ and hence the code distance required to ensure no more than a 1% chance of logical error in any data plumbing piece using $p_{L}(d, p) \approx 2d(50p)^{(d+1)/2} < 1/(100N_{\text{PL}}^{\text{data}})$. Similarly, knowing that the compute AND circuit contains 4 T gates and that no other part of the data or Majorana operator circuits contains T gates, we can calculate the total number of T gates, $N_T$, and hence the target T-state error rate from distillation of $1/(100N_T)$. We also calculate the total number of T gates.

![Figure 23](image1.png)

**FIG. 23.** Surface code implementation of the inner loop of the Majorana operator circuit. Contrast with Fig. 9, noting that the circuit has been somewhat modified to reduce its surface code spacetime volume. In particular, the controlled-$Y$ operations from Fig. 9 have been propagated through the controlled-$Z$ operations, producing CNOT operations that are cheaper to perform. This creates phase error, which must be corrected by an S gate on the control of the entire Majorana operator. Here, we do not show initial Hadamard gates on every target qubit.

![Figure 24](image2.png)

**FIG. 24.** Preparing a T state in the surface code. Physical $|T\rangle_{10}$ states are injected into the 16 lower factories in the lower rear right, which distill them into less noisy $|T\rangle_1$ states. Fifteen of the successful distillations are forwarded to the larger factory towards the front and left, which distills them into a $|T\rangle_2$ state with low enough error. The output is shown as the top-left dark U-shape.
distillation plumbing pieces, \( N_{\text{pp}} \), and a code distance to ensure that the chance of T plumbing piece error is below \(1/(100N_{\text{pp}}^T)\). We have elected to keep algorithm error rates low to ensure that, on average, only a few repetitions are required. For both \( p = 10^{-3} \) and \( p = 10^{-4} \) and all algorithm instances considered, T-state distillation of the form in Fig. 24 is sufficient to achieve the target logical error rate. This information is collectively sufficient to calculate the qubit and time overheads, shown for all cases in Table VII.

The previous paragraphs described a manual overhead estimation method. There are a number of approximations that go into such an estimate, in particular, assuming that it will always be possible to route gates in 3D spacetime without overhead beyond that of where the data qubits are stored. In order to strengthen the relevance of the presented results, we have also used a software-automated approximation method. The software is an improved version of the tool from Ref. [108]. Automated overhead approximation starts from a Clifford + T representation of the circuit to be analyzed (e.g., Fig. 8) and ends with a full surface code layout, having each gate translated into a corresponding configuration of plumbing pieces. Thus, the automated estimation work flow is similar to the manual one. However, certain circuit particularities are analyzed differently, so similarities and differences between the two methods are discussed.

The Clifford + T circuit is prepared according to a worst-case scenario, based on the available hardware restrictions, plumbing-piece layout problems, and T-gate correction mechanisms. The preparation of a single distilled T state at a time is a restriction that influences the resulting surface code layout: The Clifford + T gates have to be scheduled (laid out) in such a way that the T gates will

### Table VI. Cases for which we will calculate the fault-tolerant overhead, along with numbers relevant to estimating the non-negligible components of this overhead. Column 3 contains the number of times that the \( \mathcal{W} \) oracle is queried. For the Hubbard model, we consider the system at intermediate coupling \((u/t=4)\), implying that \( \lambda = 4N_t \), and we consider an accuracy of \( \Delta E = t/100 \). For jellium, values of \( \Delta \) are provided in Table III, and we target chemical accuracy, which is defined as \( \Delta E = 0.0016 \) Hartree.

<table>
<thead>
<tr>
<th>System</th>
<th>Spin orbitals (( N ))</th>
<th>( \mathcal{W} ) queries</th>
<th>Majorana(_N)</th>
<th>Majorana(_N/2)</th>
<th>QROM(_{3N/2})</th>
<th>Max qubits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hubbard model</td>
<td>72</td>
<td>( 1.3 \times 10^5 )</td>
<td>2.5 ( \times 10^5 )</td>
<td>1.3 ( \times 10^5 )</td>
<td>0</td>
<td>105</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>128</td>
<td>2.3 ( \times 10^5 )</td>
<td>4.6 ( \times 10^5 )</td>
<td>2.3 ( \times 10^5 )</td>
<td>0</td>
<td>161</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>200</td>
<td>3.6 ( \times 10^5 )</td>
<td>7.2 ( \times 10^5 )</td>
<td>3.6 ( \times 10^5 )</td>
<td>0</td>
<td>236</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>800</td>
<td>1.4 ( \times 10^6 )</td>
<td>2.8 ( \times 10^6 )</td>
<td>1.4 ( \times 10^6 )</td>
<td>0</td>
<td>842</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>54</td>
<td>1.8 ( \times 10^6 )</td>
<td>5.3 ( \times 10^6 )</td>
<td>0</td>
<td>3.5 ( \times 10^5 )</td>
<td>341</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>128</td>
<td>6.3 ( \times 10^6 )</td>
<td>1.9 ( \times 10^6 )</td>
<td>0</td>
<td>1.3 ( \times 10^5 )</td>
<td>210</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>250</td>
<td>1.7 ( \times 10^5 )</td>
<td>5.3 ( \times 10^5 )</td>
<td>0</td>
<td>3.5 ( \times 10^5 )</td>
<td>341</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>1024</td>
<td>1.8 ( \times 10^6 )</td>
<td>5.3 ( \times 10^6 )</td>
<td>0</td>
<td>3.5 ( \times 10^6 )</td>
<td>1136</td>
</tr>
</tbody>
</table>

### Table VII. Manual calculation of qubit and time overheads of general chemistry and Hubbard circuits, assuming gate error rates of \( p = 10^{-3} \) and \( p = 10^{-4} \), a 2D array of nearest-neighbor coupled qubits, and a surface code error-detection cycle time of 1 \( \mu \)s. The execution time being estimated is the duration of one complete run of the phase estimation process.

<table>
<thead>
<tr>
<th>System</th>
<th>Spin orbitals (( N ))</th>
<th>( p = 10^{-3} )</th>
<th>( p = 10^{-4} )</th>
<th>( p = 10^{-3} )</th>
<th>( p = 10^{-4} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hubbard model</td>
<td>72</td>
<td>1.4 ( \times 10^6 )</td>
<td>4.4 ( \times 10^5 )</td>
<td>4.6</td>
<td>2.6</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>128</td>
<td>2.1 ( \times 10^6 )</td>
<td>6.6 ( \times 10^5 )</td>
<td>15</td>
<td>8.4</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>200</td>
<td>3.2 ( \times 10^6 )</td>
<td>8.9 ( \times 10^5 )</td>
<td>40</td>
<td>21</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>800</td>
<td>1.4 ( \times 10^7 )</td>
<td>3.6 ( \times 10^6 )</td>
<td>6.7 ( \times 10^2 )</td>
<td>3.7 ( \times 10^2 )</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>54</td>
<td>1.4 ( \times 10^6 )</td>
<td>3.9 ( \times 10^5 )</td>
<td>0.82</td>
<td>0.43</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>128</td>
<td>2.4 ( \times 10^6 )</td>
<td>8.1 ( \times 10^5 )</td>
<td>9.9</td>
<td>5.6</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>250</td>
<td>4.4 ( \times 10^6 )</td>
<td>1.2 ( \times 10^6 )</td>
<td>58</td>
<td>30</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>1024</td>
<td>2.0 ( \times 10^7 )</td>
<td>4.8 ( \times 10^6 )</td>
<td>2.7 ( \times 10^3 )</td>
<td>1.4 ( \times 10^3 )</td>
</tr>
</tbody>
</table>
be executed as soon as possible, but not earlier than the availability of distilled T states. The state distillation form in Fig. 24 implies that a T gate can be executed, on average, every six plumbing pieces along the time axis. Additionally, T-gate implementations are probabilistic, and S-gate corrections may be necessary. Thus, our scenario considers that all T gates are followed by the corrective S gate, resulting in a synthetic increase of the Clifford + T circuit depth. Circuit preparation is followed by an optimization procedure, where as many Clifford gates as possible are scheduled between two subsequent T gates. The software simulates the availability of the distilled T states and places T gates whenever their execution is possible. If no T states are available, the T gates are delayed, which will later increase the approximated time overhead.

Finally, the Clifford + T circuit is translated into the surface code layout. From a resource estimation perspective, the complexity of this task is increased because the software currently only partially includes the optimization strategy presented in Fig. 21(b): Final boxes can be compressed to a single one, but distinct dark structures are not allowed to touch (for verification or debugging purposes). Due to this fact, the automated approximation uses a slightly different Clifford + T realization of the computing AND gate (cf. Fig. 4), which has the advantage of being more suitable for automatic placement in stairway-structured circuits (i.e., the arrangement of AND gates in Fig. 9). Automatic placement of those Clifford + T subcircuits results in a shorter depth of the generated surface code layouts. Overhead of the two basic circuits in units of plumbing pieces can be found in Table VIII. These data converted into qubits and time can be found in Table IX. The automatically generated estimations are comparable to the manual ones, though generally slightly higher, exceeding the manual estimates by 10%–20%. While the automated method is penalized by missing optimization strategies that are possible when analyzing circuits manually, and the need to provide explicit communication paths for long-range gates, some of this penalty is canceled by using algorithmic methods too complex to perform manually. The fact that both approximation methods lead to

### TABLE VIII. Automatically generated resource estimates of the Majorana and QROM circuits (Figs. 9 and 10). The area width, height, and time columns give the dimensions of the bounding box (e.g., Fig. 25) in units of plumbing pieces. The last column is the number of plumbing pieces estimated to be actively used within the bounding box. The volume numbers do not include the volume of the T factory, but they do include idle qubits that are present in the algorithm as a whole but not the individual circuits. "Braided volume" refers to the amount of actively used volume, i.e., nonempty space with defects used to encode qubits and operations. QROM circuits are indexed like QROM$_{2N}$ because, in context, the QROM index size $L$ is always 50% larger than the number of orbitals $N$.

<table>
<thead>
<tr>
<th>System</th>
<th>Circuit $(N)$</th>
<th>T count</th>
<th>Area (PP$^2$)</th>
<th>Time (PP)</th>
<th>Volume (PP$^3$)</th>
<th>Braided volume (PP$^3$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hubbard model</td>
<td>Majorana$_{72}$</td>
<td>284</td>
<td>17 × 16</td>
<td>1840</td>
<td>500 480</td>
<td>429 624</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>Majorana$_{128}$</td>
<td>508</td>
<td>25 × 16</td>
<td>3252</td>
<td>1 300 800</td>
<td>1 155 817</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>Majorana$_{256}$</td>
<td>796</td>
<td>36 × 16</td>
<td>5080</td>
<td>2 926 080</td>
<td>2 637 504</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>Majorana$_{512}$</td>
<td>3196</td>
<td>123 × 16</td>
<td>20 262</td>
<td>39 875 616</td>
<td>37 451 032</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>Majorana$_{32}$</td>
<td>212</td>
<td>20 × 16</td>
<td>1382</td>
<td>442 240</td>
<td>365 717</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>Majorana$_{64}$</td>
<td>508</td>
<td>32 × 16</td>
<td>3252</td>
<td>1 665 024</td>
<td>1 473 563</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>Majorana$_{128}$</td>
<td>996</td>
<td>51 × 16</td>
<td>6342</td>
<td>5 175 072</td>
<td>468 5164</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>Majorana$_{256}$</td>
<td>4092</td>
<td>165 × 16</td>
<td>25 932</td>
<td>68 460 480</td>
<td>64 114 531</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>QROM$_{254}$</td>
<td>320</td>
<td>20 × 16</td>
<td>2068</td>
<td>661 760</td>
<td>558 098</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>QROM$_{512}$</td>
<td>764</td>
<td>32 × 16</td>
<td>4872</td>
<td>2 494 464</td>
<td>2 273 711</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>QROM$_{1024}$</td>
<td>1496</td>
<td>51 × 16</td>
<td>9508</td>
<td>7 758 528</td>
<td>7 272 549</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>QROM$_{2048}$</td>
<td>6140</td>
<td>165 × 16</td>
<td>38 892</td>
<td>102 674 880</td>
<td>100 399 903</td>
</tr>
</tbody>
</table>
TABLE IX. Automatically generated qubit and time overheads of general chemistry and Hubbard circuits assuming gate error rates of $p = 10^{-3}$ and $p = 10^{-4}$, a 2D array of nearest-neighbor coupled qubits, and a surface code error-detection cycle time of 1 µs. The execution time being estimated is the duration of one complete run of the phase estimation process.

<table>
<thead>
<tr>
<th>Problem</th>
<th>$N$ spin orbitals</th>
<th>$p = 10^{-3}$</th>
<th>$p = 10^{-4}$</th>
<th>$p = 10^{-3}$</th>
<th>$p = 10^{-4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hubbard model</td>
<td>72</td>
<td>$1.7 \times 10^6$</td>
<td>$5.3 \times 10^5$</td>
<td>4.6</td>
<td>2.6</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>128</td>
<td>$2.4 \times 10^6$</td>
<td>$7.8 \times 10^5$</td>
<td>15</td>
<td>8.4</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>200</td>
<td>$3.8 \times 10^6$</td>
<td>$1.0 \times 10^6$</td>
<td>40</td>
<td>21</td>
</tr>
<tr>
<td>Hubbard model</td>
<td>800</td>
<td>$1.5 \times 10^7$</td>
<td>$4.2 \times 10^6$</td>
<td>$6.7 \times 10^2$</td>
<td>$3.7 \times 10^2$</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>54</td>
<td>$1.7 \times 10^6$</td>
<td>$4.7 \times 10^5$</td>
<td>0.85</td>
<td>0.44</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>128</td>
<td>$2.9 \times 10^6$</td>
<td>$9.5 \times 10^5$</td>
<td>10</td>
<td>5.7</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>250</td>
<td>$5.1 \times 10^6$</td>
<td>$1.4 \times 10^6$</td>
<td>58</td>
<td>30</td>
</tr>
<tr>
<td>Electronic structure</td>
<td>1024</td>
<td>$2.3 \times 10^7$</td>
<td>$5.6 \times 10^6$</td>
<td>$2.8 \times 10^3$</td>
<td>$1.4 \times 10^3$</td>
</tr>
</tbody>
</table>

such comparable qubit and time overheads strengthens our confidence in these estimates. The time estimates, in particular, are practically identical.

The data are highly encouraging, with physical qubit counts of order a million and times in hours for all but the largest cases considered. Significant further reduction is expected to be possible. For example, the $N$ qubits in the $|\Psi\rangle$ register are only operated on by the Majorana operator circuit, and this circuit targets just one of these qubits at a time. This implies that the remainder can be stored more compactly in square surface code patches while not being interacted with. This method could easily reduce the overhead of these $N$ qubits, which account for 70%–80% of the physical qubits, by a factor of 6. This result would conservatively lower overall physical qubit requirements by a factor of two.

VII. CONCLUSION

In this work, we introduced especially efficient fault-tolerant quantum circuits for using phase estimation to estimate the spectra of electronic Hamiltonians. Unlike past work, which has focused on realizing phase estimation unitaries encoding $e^{-iHt}$, corresponding to time evolution under $H$ for duration $t$, we focused on a recent idea that one might more cleverly realize phase estimation unitaries encoding the quantum walk $e^{i \arg \cos(H/\lambda)}$, where $\lambda$ is a parameter closely related to the induced 1-norm of the system Hamiltonian [26,27]. We construct explicit quantum circuits for realizing this quantum walk with T complexity linear in basis size for both the planar Hubbard model and electronic structure Hamiltonians in second quantization. We showed that phase estimation projects these systems to an eigenstate and estimates the associated eigenvalue to within additive error $\epsilon$ by querying the quantum walk operator an optimal number of times, scaling as $O(\lambda/\epsilon)$. To accomplish this result, we introduced general techniques that we conjecture are near optimal for streaming bits of a unary register and for implementing a quantum read-only memory. We introduced a new form of Heisenberg-limited phase estimation specialized to linear combinations of unitaries based simulations and provided bounds on T complexity and ancilla count, which remain tight even at small finite sizes.

In addition to providing explicit Clifford + T circuits, we compiled the bottleneck components of these simulations to fault-tolerant surface code gates in order to rigorously determine the resources that would be required for error correcting interesting problems. We performed this compilation both by hand and by using automatic tools and found similar overheads in both cases. We found that classically intractable instances of jellium and the Fermi-Hubbard model could be simulated with under $1 \times 10^6$ T gates and would require about $1 \times 10^6$ physical qubits in the surface code, with two-qubit error rates on the order of $10^{-3}$. At error rates of $10^{-4}$, about an order of magnitude fewer physical qubits would be required. We also priced out simulations of realistic solid-state materials such as diamond, graphite, silicon, metallic lithium, and crystalline lithium hydride and found that only slightly more than $1 \times 10^6$ T gates and a few million physical qubits would be required.

Despite focusing on different systems, our results are most readily comparable to the previous state-of-the-art results from Ref. [43]. Even though Ref. [43] sought empirical estimates of the T complexity rather than rigorous upper bounds as we did, they estimated that approximately $10^{15}$–$10^{16}$ T gates would be required for a 108-qubit simulation of the FeMoco molecule active space. By comparison, our upper bounds on the T complexity required to solve the classically intractable electronic structure problems studied here were roughly a million times less. The low T complexity is the result of designing a lean algorithm from the ground up, with insights matched to the Hamiltonian and with innovative algorithmic subroutines. The improvements are distributed across several parts of our approach, each of which provides 1 or 2 orders of magnitude improvement. Because our simulations require only a few times more physical qubits than is
required by a single T factory, it is reasonable to expect that the simulations we outline here will become practical on the first universal fault-tolerant quantum devices, many years before the simulations discussed in Ref. [43] would be viable.

Several important directions for future research pertain to the extension of these simulation techniques to representations that would be more effective for single molecules. While the dual basis described in Ref. [39] is well suited to treating solid-state materials such as the ones explored here (e.g., jellium, solid-state silicon, graphite, diamond, lithium and lithium hydride), by combining our techniques with the “Gausslet” basis sets of Ref. [41], we should also be able to simulate single molecules with similar resolution to Gaussian orbitals—thus extending our results to systems such as FeMoco with similar overheads to those observed in this work. However, deploying the Gausslet basis functions to systems with large atomic nuclei such as iron (as in FeMoco) will require further research. If basis errors are a concern, then future work should combine results from this paper with Refs. [109] and [26] in order to validate and implement in software; these constructions are explored surface code constructions that we were able to estimate in the upper bound on the number of physical qubits required by a single T factory, it is reasonable to expect that the simulations we outline here will become practical on the first universal fault-tolerant quantum devices, many years before the simulations discussed in Ref. [43] would be viable.

Several important directions for future research pertain to the extension of these simulation techniques to representations that would be more effective for single molecules. While the dual basis described in Ref. [39] is well suited to treating solid-state materials such as the ones explored here (e.g., jellium, solid-state silicon, graphite, diamond, lithium and lithium hydride), by combining our techniques with the “Gausslet” basis sets of Ref. [41], we should also be able to simulate single molecules with similar resolution to Gaussian orbitals—thus extending our results to systems such as FeMoco with similar overheads to those observed in this work. However, deploying the Gausslet basis functions to systems with large atomic nuclei such as iron (as in FeMoco) will require further research. If basis errors are a concern, then future work should combine results from this paper with Refs. [109] and [26] in order to determine the cost of encoding first-quantized electronic spectra in quantum circuits; in first quantization, basis errors are suppressed exponentially in the number of qubits used to represent the system.

Another remaining challenge is to compute a tighter upper bound on the number of physical qubits required by the algorithm. At the first moment that it becomes technology possible to distill magic states, the number of physical qubits available on one machine will still be extremely limited. Getting a meaningful computation to fit at all will be difficult. Fortunately, the qubit count estimates of this paper were fairly conservative: We only explored surface code constructions that we were able to validate and implement in software; these constructions are not necessarily optimal. For example, the logical qubit representation used in lattice surgery [110] requires fewer physical qubits than the double-defect representation used in the estimates of this paper. Furthermore, there are several places in our circuits where we preferred small multiplicative improvements in T count over small additive improvements in logical qubit count. For example, we delay uncomputing QROM lookups in order to avoid recomputation, and when performing phase estimation, we minimize the number of oracle queries by using a full-size phase register instead of a single phase qubit. Since we have managed to show that with error rates of $10^{-3}$ one can solve interesting problems in chemistry using on the order of a million physical qubits within the surface code, a next natural goal would be to try to further reduce the resources required to be on the order of a 100,000 physical qubits.

ACKNOWLEDGMENTS

The authors thank Yuval Sanders, Artur Scherer, Mária Kieferová, and Guang Hao Low for helpful discussions about linear combinations of unitaries based simulation methods. We thank Garnet Kin-Lic Chan and Kostyantyn Kechedzhi for discussions pertaining to the regimes in which the Hubbard model would be interesting to simulate. We thank Ian Kivlichan, Zhang Jiang, and Dave Bacon for helpful comments on an early version of this manuscript. D. W. B. is funded by an Australian Research Council Discovery Project (Grant No. DP160102426).

APPENDIX: PROPAGATING ERRORS FROM HAMILTONIAN COEFFICIENTS INTO PHASE ESTIMATE

In this appendix, we address the question of how accurately coefficients of the Hamiltonian must be prepared in the PREP oracle in order to estimate the Hamiltonian eigenvalues to precision $\epsilon$. As discussed in Sec. II A, our phase estimation scheme involves estimating the phases induced by the operator $e^{i \arccos(H/\lambda)}$. If one is near the singularity of arccos, then a small error in the Hamiltonian can have a significant impact on the phase. Let us define

$$\tilde{H} = \sum_{\ell=0}^{L-1} \tilde{w}_\ell H_\ell$$

for our approximate encoding of $H$. Using the state preparation technique in Sec. III D, we obtain

$$\lambda = \sum_{\ell=0}^{L-1} \tilde{w}_\ell.$$  (A2)

We denote by $\delta$ an upper bound on the approximation in any of the $\tilde{w}_\ell$, so

$$\delta \geq |\tilde{w}_\ell - w_\ell|.$$  (A3)

Next, note that the error in the eigenphase obeys

$$e_{\text{PREP}} \leq \left| e^{i \arccos(H/\lambda)} - e^{i \arccos(\tilde{H}/\lambda)} \right| \\
\leq \left| \arccos(H/\lambda) - \arccos(\tilde{H}/\lambda) \right| \\
\leq \sum_{p=1}^{\infty} \frac{(2p-1)!}{2^{2p+1}(2p+1)(2p)!} \| H_{2p+1} - \tilde{H}_{2p+1} \|.$$  (A4)

where $!!$ is the double factorial $z!! = z \cdot (z-2) \cdot (z-4) \ldots 1$ assuming $z$ is a natural number. It is straightforward to show inductively that for any $p > 0$,

$$\| H_{2p+1} - \tilde{H}_{2p+1} \| \leq (2p+1)(\max\{\|H\|,\|\tilde{H}\|\})^{2p} \| H - \tilde{H} \|.$$  (A5)

We then have from Eq. (A1) that
\[ \|H - \tilde{H}\| \leq \sum_{\ell=0}^{L-1} |w_{\ell} - \tilde{w}_{\ell}| \leq L\delta. \quad (A6) \]

We further have that
\[ \max\{\|H\|, \|\tilde{H}\|\} \leq \|H\| + L\delta. \quad (A7) \]

Substituting these equations into Eq. (A4) then gives
\[
\epsilon_{\text{PREP}} \leq \frac{\sum_{p=0}^{\infty} (2p - 1)!!}{2^p (2p)!!} \left( \frac{\|H\| + L\delta}{\lambda} \right)^{2p}
\leq \frac{L\delta}{\lambda} \sum_{p=0}^{\infty} (2p - 1)!! \left( \frac{\|H\| + L\delta}{\lambda} \right)^{2p}
\leq \frac{L\delta}{\lambda} \left[ 1 - \left( \frac{\|H\| + L\delta}{\lambda} \right)^{2\lambda} \right]^{-1/2}. \quad (A8) \]

This inequality can be solved for \( \delta \) to give
\[
\delta \geq \frac{\epsilon_{\text{PREP}}}{(1 + \epsilon_{\text{PREP}}^2)\lambda} \left( \sqrt{\lambda^2(1 + \epsilon_{\text{PREP}}^2) - \|H\|^2} - \epsilon_{\text{PREP}}\|H\| \right)
\geq \frac{\epsilon_{\text{PREP}}^2}{(1 + \epsilon_{\text{PREP}}^2)\lambda} \left( 1 - \|H\|^2/\lambda^2 \right). \quad (A9) \]

If we require that \( \epsilon_{\text{PREP}} \leq \sqrt{2\Delta E}/(4\lambda) \) as in Eq. (25), then this can be obtained by choosing
\[
\delta = \frac{\sqrt{2\Delta E}}{4L(1 + \Delta E/k_0^2)} \left( 1 - \frac{\|H\|^2}{\lambda^2} \right). \quad (A10) \]

[58] C. Gidney and R. Babbush, Quantum Read-Only Memory for Implement Efﬁcient Fault-Tolerant Quantum Oracles (in press).