Low-depth phase oracle using a parallel piecewise circuit

Zhu Sun [1,2,*] Gregory Boyd [1] Zhenyu Cai [1] Hamza Jnane [1,2] Bálint Koczor [1,3] Richard Meister [1,4]
Romy Minko [1,5] Benjamin Pring [1,5] Simon C. Benjamin [1,2] and Nikitas Stamatopoulos [6]

[1] *Quantum Motion, 9 Sterling Way, London N7 9HJ, United Kingdom*
[2] *Department of Materials, University of Oxford, Parks Road, Oxford OX1 3PH, United Kingdom*
[3] *Mathematical Institute, University of Oxford, Woodstock Road, Oxford OX2 6GG, United Kingdom*
[4] *Department of Computing, Imperial College London, South Kensington Campus, London SW7 2AZ, United Kingdom*
[5] *School of Mathematics, University of Bristol, Fry Building, Woodland Road, Bristol BS8 1UG, United Kingdom*
[6] *Goldman Sachs, New York, New York 10282, USA*

(Received 11 October 2024; accepted 28 May 2025; published 16 June 2025)

We explore the important task of applying a phase $\exp(i\,f(x))$ to a computational basis state $|x\rangle$. The closely related task of rotating a target qubit by an angle depending on $f(x)$ is also studied. Such operations are key in many quantum subroutines, and frequently $f(x)$ can be well approximated by a piecewise function; examples range from the application of diagonal Hamiltonian terms (such as the Coulomb interaction) in grid-based many-body simulation to derivative pricing algorithms. Here, we exploit a parallelization of the piecewise approach so that all constituent elementary rotations are performed simultaneously, that is, we achieve a total rotation depth of one. Moreover, we explore the use of recursive catalyst "towers" to implement these elementary rotations efficiently. We find that strategies prioritizing execution speed can achieve circuit depth as low as $O(\log_2 n + \log_2 S)$ for a register of n qubits and a piecewise approximation of S sections (presuming prior preparation of enabling resource states), albeit total qubit count then scales with S. In the limit of multiple repetitions of the oracle, we find that catalyst tower approaches have an $O(S\,n)$ T-count.

DOI: 10.1103/m32k-7nq2

I. INTRODUCTION

The speedup promised by quantum computing has drawn intense attention over the last few decades. Quantum algorithms are explored for problems in various research fields including cryptography [1], linear systems of equations [2], optimization [3], and computational chemistry [4], among others. In these studies, algorithms usually start by assuming the ability, often granted by an "oracle," to encode some function value $f(x)$ related to $|x\rangle$ into a quantum state. Efficient implementation of such oracles is extensively studied in the context of, e.g., state preparation.

In this work, we study the common task of encoding a function value into phase, i.e.,

$$|x\rangle \mapsto \exp(if(x))\,|x\rangle\,, \tag{1}$$

and we propose an efficient circuit implementation for this type of oracle. With the addition of a few Clifford gates, one instead obtains a variant which implements

$$|x\rangle\,|0\rangle \mapsto |x\rangle\,(\cos f(x)\,|0\rangle + i \sin f(x)\,|1\rangle\,). \tag{2}$$

For either case, we use classical precomputation to make a piecewise approximation of the function $f(x)$, avoiding expensive arithmetic. By combining fan-out techniques with a repeat-until-success scheme, the circuit depth can be as low as $O(\log_2 n)$ for an n-qubit input register. We also employ "catalyst towers" to moderate the high demand on magic states, which is a common problem for conventional methods.

Our approach is beneficial for well-behaved functional forms and is particularly beneficial when the circuit is used as a repeatedly applied subroutine, such as time evolution, amplitude amplification, etc. A good example is the $1/r$ Coulomb potential. It has a simple functional form while being crucial to quantum chemistry simulations, which is one of the three application examples we present. Conversely, there are, of course, certain functions that are impractical for piecewise approximation (see Appendix B)—for example, certain oscillatory functions used to benchmark optimization algorithms, e.g., the Ackley function [5].

Our focus is on minimizing depth while maintaining an efficient use of resources such as magic states; to achieve this, we allow ourselves to increase the number of qubits that are concurrently active; essentially, a space-time trade-off. Consequently, the methods may be particularly relevant to quantum computing platforms where there is the hope of large numbers of qubits, such as silicon spin chip-based devices [6]. For optimization against other priorities, we point readers toward the complementary discussion in Ref. [7], which compares methods of applying a phase focusing on minimal-width implementations.

*Contact author: zhu.sun@exeter.ox.ac.uk

Historically, the word "oracle" has been widely used in quantum algorithms studies (as early as, e.g., Shor's algorithm [1] and Grover's algorithm [8]) to describe the black box which gives access to some function $f(x)$. Thus, in the context of algorithmic complexity, its construction details are irrelevant. Since this work discusses the exact implementations, we narrow down the definition of "oracle": a circuit that encodes some function $f(x)$ into a quantum state, and $f(x)$ must be known so that it can be approximated in a piecewise fashion. This, then, excludes the functions in, e.g., Shor's algorithm and Grover's algorithm, simply because knowing the relevant functions would directly solve the problem.

The rest of the paper is organized as follows: in Sec. II we discuss the prior work. The main implementation is explained in Sec. III, followed by the options for the key part of our implementation in Sec. IV, in which we introduce the catalyst towers. The cost analysis for different options can be found in Sec. V. In Sec. VI, we provide three example applications which can benefit from our work, namely, financial derivatives pricing, quantum chemistry, and quantum simulation.

II. PRIOR WORK

The need of encoding a function onto a quantum state goes back to the very early days of quantum computing and it is essential in many quantum computing applications. Depending on the problem, one might want to encode the information into.

(i) the register, $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$, e.g., Simon's problem [9];

(ii) the amplitude, $|x\rangle|0\rangle \mapsto |x\rangle(\sqrt{f(x)}|\psi\rangle + \sqrt{1-f(x)}|\psi^{\perp}\rangle)$ (for some normalized $|\psi\rangle$ and orthogonal $|\psi^{\perp}\rangle$), e.g., quantum optimisation [10]; or

(iii) the phase, $|x\rangle \mapsto \exp(if(x))|x\rangle$, e.g., quantum phase estimation [11].

A. Oracle models

For solving some specific problems in an application area, the three aforementioned models are usually cast into oracles and the related algorithms are evaluated by the query complexity. These three models of oracles are often closely related to each other. For example, in Ref. [10] the authors show that given an amplitude oracle, it can be used to simulate the corresponding phase oracle to ϵ precision with $O(\log_2(1/\epsilon))$ queries, and the conversion from phase oracle back to amplitude oracle also requires logarithmic overheads in the precision.

B. State preparation methods

The encoding of a function into amplitudes is generalized in the context of quantum state preparation, which seeks to prepare a state of the form $\sum_x f(x)/N_f|x\rangle$, where N_f is a normalizing constant. As a standard technique, the key step in the Grover-Rudolph method [12] first writes the bit string into an ancillary register using arithmetic. Applying rotations to the ancillae controlled on the bit strings will then transform the function value into the amplitude. Later methods like quantum rejection sampling [13] and quantum eigenvalue transformation [14] seek to address the same problem without the expensive arithmetic. Moreover, if the controlled-rotations in the Grover-Rudolph method are replaced by a phase $Ph(\theta) = \mathrm{diag}(1, \exp(i\theta))$ on each bit in the string, then it implements a phase oracle. Again, the use of arithmetic is significant in the total cost of the method.

C. The QROM approaches

The existence of the oracle $U_f|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$ has been assumed since the toy-problem era of quantum algorithms. One circuit implementation of it is the quantum read-only memory (QROM) [15] and it is widely used in quantum chemistry for, e.g., state preparation [16,17]. However, the use of this type of QROM is sometimes limited by its rapid scaling in T-count and T-depth [18]. Note that a more general model quantum random access memory (QRAM) [19] is frequently used to do the same task, except in its most rigorous definition QRAM must also allow data to be written to it. A detailed explanation of the subspecies of QRAM can be found in Ref. [20].

QROM approaches have also been used for the phase oracle [21], first by loading information about the linear interpolation of the function $\tilde{f}(x)$ and then transforming that information into a phase. This is done by calculating flags to obtain the region that contains the value of x, and for that region output the slope and interpolation for the linear approximation. The calculation of $|\tilde{f}(x)\rangle$ can then be performed with a single multiplication and addition. This register can then be used to apply a phase either by applying rotation blocks similar to those discussed in Sec. III or by adding this register into a phase-gradient state in order to apply $e^{i\tilde{f}(x)}$ via phase kickback [22]. The multiplication of x with the slope will likely have the largest cost in this approach, with methods requiring either logarithmic depth with an $O(n^{\log_2(3)})$ overhead [23] or $O(\log^2 n)$ depth with a logarithmic space overhead [24]. For the purposes of comparing the most optimal-depth methods in this work, we will primarily compare to the Karatsuba multiplication with $O(n^{\log_2(3)})$ T-cost [23], although we note that when the additional cost of this method becomes relevant, one could always switch to the alternative scheme with $O(n\log_2 n)$ cost and $O(\log^2 n)$ depth [24]. Once this multiplication has taken place, the subsequent addition of the intercept and application of the phase must be performed before uncomputing the arithmetic, resulting in a larger depth than some of the alternatives presented in this work. However, due to the efficiency in T count of reusing a phase gradient state to apply a phase (only a single addition that does not depend on ϵ), this method sees savings in this respect for algorithms requiring a large number of iterations of this oracle. We provide some further comments and comparison to the independent towers method in Sec. IV and Appendix E.

D. Quantum dynamics approaches

An eigenstate $|\phi\rangle$ of a Hamiltonian H under time evolution goes through the transformation $|\phi\rangle \mapsto \exp(-iEt)|\phi\rangle$, where E is the eigenvalue. The Hamiltonian simulation addresses the problem that given some H, its action on some state $|\psi\rangle = \sum_j c_j|\phi_j\rangle$ under time evolution is simulated by a unitary U such that $U|\psi\rangle \sim \sum_j c_j \exp(-iE_j t)|\phi_j\rangle$. Relevant methods

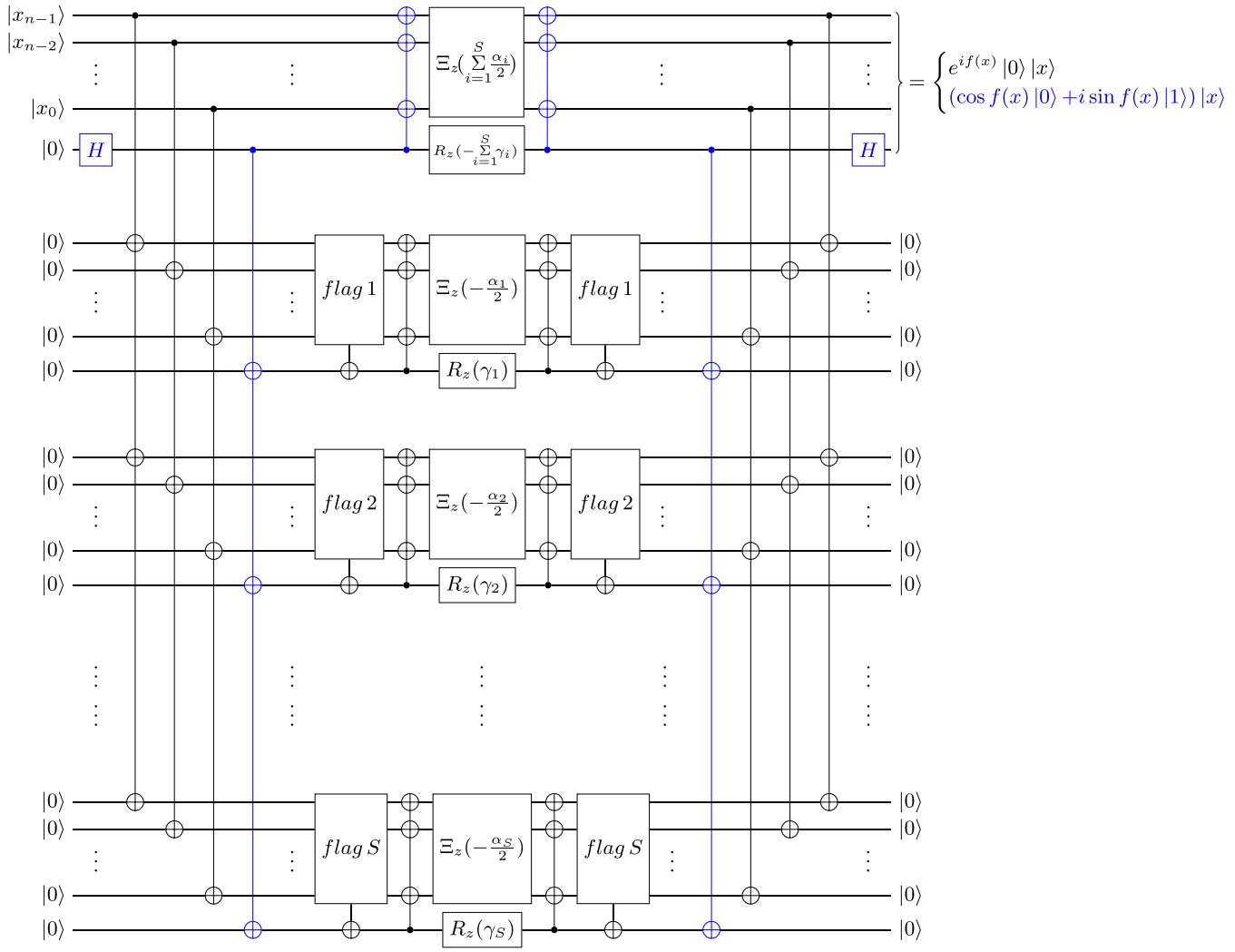


FIG. 1. A circuit parallelizing all elementary rotations. The central rotation blocks are defined by $\Xi_z(a) := \bigotimes_{i=0}^{n-1} R_z(2^i a)$, i.e., a set of single-qubit rotations performed in parallel. If the blue gates are implemented, then the top group of $n+1$ qubits is left in the state $(\cos f(x)|0\rangle + i\sin f(x)|1\rangle)|x\rangle$ (which is also in blue in the output); otherwise the group is left in state $e^{if(x)}|0\rangle|x\rangle$. Flag blocks flip their target qubit if and only if the controlling input is in the corresponding internal of the piecewise function, as explained in the text. Further optimizations to this circuit are possible (e.g., the fan-in/out), we provide details in Appendix A. Both black and blue variants were verified by exact simulation with QUESTLINK; this code is available from the authors on request.

such as Trotterization [25], quantum signal processing [26], and quantum walk [27] are extensively explored.

E. Continuous angle rotation

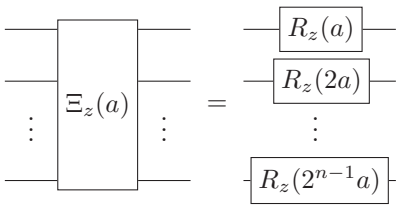
For state preparation, continuous angle rotations are ubiquitous and this is usually expensive. Given a universal basis set of unitary gates for implementation, such as the Clifford+T gate set, direct gate synthesis leverages the underlying number theoretic properties of the gate set to determine, classically and deterministically, a sequence of basis gates that will approximate the target rotation to within the desired accuracy ϵ. For single-qubit diagonal rotation gates, as in Fig. 1, this can be done optimally without using ancilla qubits for a T depth and T count of $O(3\log_2(1/\epsilon))$, using Clifford+T gates [28]. Additional techniques, such as fallback protocols [29], probabilistic mixing [30,31], and a combination of the two [32], achieve further reductions in T depth via using ancilla qubits and projective measurements to leading order of $\log_2(1/\epsilon)$, $1.5\log_2(1/\epsilon)$, and $0.5\log_2(1/\epsilon)$, respectively. Gate synthesis protocols using smaller angle rotations instead of T gate, e.g., employing higher orders in the Clifford hierarchy, were also explored in the literature [33,34]. However, the dependence of cost on rotation accuracy can be problematic for practical purposes. We detail two alternative approaches for implementing a group of rotations using an arrangement of generalized rotation catalyst circuits [35], which we call *catalyst towers*.

III. PARALLELIZING THE PIECEWISE FUNCTION

Piecewise polynomial approximation is commonly used in mathematics and it is also not new to quantum algorithms [21,36,37]. However, one of its main problems is the great circuit depth due to the serial rotations. Here, the solution we study is to parallelize the circuit using fan-out techniques, which is a width-depth trade-off. This is valuable for

algorithms that require time optimization, e.g., the derivative pricing algorithm example in Sec. VI.

To parallelize, we propose the circuit in Fig. 1. First, let us disregard the gates in blue to apply the phase $e^{if(x)}$ to the top register of n qubits (we will come back to these blue gates later). Suppose the target function $f(x)$ is sectioned into S pieces by some algorithm (we discuss one method in Appendix B), and each piece i is approximated by a linear function $\alpha_i x + \beta_i$, for x belonging to the ith interval. For simplicity, define $\gamma_i = \alpha_i \phi + \beta_i$, with $\phi = (2^n - 1)/2$. Note that although we use linear segments here for illustration, higher-order polynomial approximations can also be used. The corresponding circuit has $S + 1$ registers of $n + 1$ qubits. (Strictly speaking, only n qubits are used in the uppermost part of Fig. 1 when the blue gates are not employed; however, we include this redundant qubit to keep resource expressions compact and applicable to both black an blue variants.) The input state $|x\rangle = |x_0\rangle \ldots |x_{n-1}\rangle$ is first copied to the other S registers to parallelize the rotations. The circuit then computes the flags $1, 2, \ldots, S$ to see if the value x is in section $1, 2, \ldots, S$, respectively. The exact depth of a flag block depends on the choice of section boundaries; in the examples we consider here, the flag block is equivalent to a single $\lceil \log_2 S \rceil$-controlled Toffoli gate (the method in Appendix B uses a binary sectioning). The flags are followed by the large rotation blocks, denoted as $\Xi_z(a) := \bigotimes_{i=0}^{n-1} R_z(2^i a)$ or diagrammatically as



(3)

where $R_z(a) = \mathrm{diag}(e^{-ia/2}, e^{ia/2})$.

If the flag computation for the jth register indicates that the value of x is in section j, then the final qubit of that register, initially in state $|0\rangle$, is flipped to $|1\rangle$. The CNOT gates immediately preceding the rotation block are therefore activated, which effectively changes the sign on the phases applied by the rotation block. Observe, firstly, that the desired angles of α_j are halved and, secondly, the phases associated with the other $S - 1$ sections are still applied. This is dealt with by the rotation block acting on the first register. Here, the rotation angle is the sum of each of the angles in the S ancilla registers. Note that this block does not have a minus sign. Thus, the effect of this block is to cancel out the rotations applied on all registers for which the corresponding flag is not activated and applies the remaining half of the rotations for the activated one. The single qubit γ_i rotation offsets on the last qubit of each register are dealt with in a similar manner, via the single-qubit rotations of $-\sum_{i=1}^{S} \gamma_i$ on the final qubit of the top register. After further disentangling the registers, the first n qubits are in state $e^{if(x)}|x\rangle$. We also provide a variant of this circuit: if the blue gates are implemented, then the information encoded in phase is transformed into amplitude, such that the top group of $n + 1$ qubits ends up in the state $(\cos f(x)|0\rangle + i \sin f(x)|1\rangle)|x\rangle$.

The parallelization greatly reduces the circuit depth, but there remains another problem: the huge demand for magic states required to synthesize a large number of rotations in the "large rotation block" in Eq. (3). One contribution of this work is exploring various constructions for applying these rotations efficiently under different usage scenarios. In particular, we focus on the improvement in T gate cost when the circuit is repeatedly applied, e.g., as a subroutine in amplitude estimation. We will describe several options for this in the next section.

IV. OPTIONS FOR IMPLEMENTING PHASE APPLICATION

The options we explore for implementing the phase application are:

(1) Canonical gate synthesis (and a variant) using, e.g., Clifford+T;

(2) Deterministic in-circuit catalyst towers;

(3) Independent catalyst towers generating resource states, which are consumed via gate teleportation; and

(4) Loading a linear interpolation via QROM and using phase kickback to apply the rotation.

Two of these methods involve catalyst towers whose purpose is to generate multiple rotations, or rotation resources, from a single input rotation and a frugal set of T gates. The towers leverage the "generalized phase catalysis circuit" from Ref. [35], and we reproduce that circuit in Fig. 2 (left). In the present context we regard this as a one-layer tower. The $R_Z(\theta)|+\rangle$ state in the circuit is a so-called catalyst state. This acts analogously to a chemical catalyst in a reaction, in the sense that it makes the circuit less costly and remains intact after the process. For the first run of a tower circuit, the catalyst state must be synthesized in some standard way. Thus, the use of such circuits in our towers is useful in the scenario that the oracle must be applied many times, or over many "rounds," to take full advantage of the reusable nature of the catalyst.

There would be no advantage in the event that the oracle was used only once; since the logical AND costs 4 T states, the first round is always more expensive than canonical gate synthesis. However, from the second round onward one only needs to synthesize the seed rotation $R_Z(2\theta)$, plus four extra T states for the logical AND to apply two $R_Z(\theta)$ rotations. The higher cost of the first round is averaged over the repetitions, and we will see that the break-even point for T count is quickly reached. For simplicity, we denote this circuit as a "CT block" as shown in Fig. 2 (right).

The key to the in-circuit catalyst tower approach is the composition of the one-layer catalyst towers. We show an example of a two-layer in-circuit tower in Fig. 3, and higher towers can be generalized based on this. To implement the rotations in Eq. (3), replace the large rotation block with an n-layer tower, and then the desired rotation gates are applied directly on the target qubits (with one unused rotation). However, note that the scaling of the circuit depth of an n-layer tower is linear in n; this can be seen from the V shape of the gates of the corresponding circuit diagram of the two-layer tower (Fig. 3), as shown in Fig. 4. Thus, this would have a greater T depth than the direct-synthesis method, albeit with a reduced T cost.

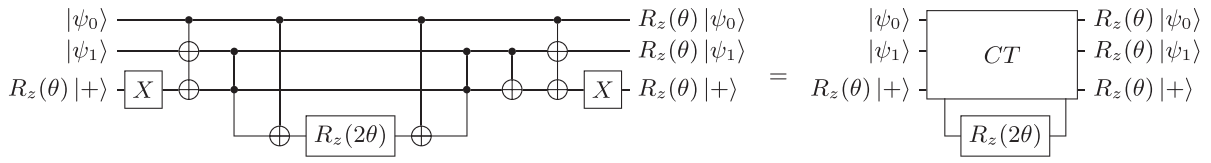


FIG. 2. Circuit for a one-layer tower. The "corners" notation for the AND gate follows Ref. [22], reproduced in Appendix A.

We note that a similar "in-circuit tower" is also discussed in Ref. [38]. In that paper, the authors are interested in an option pricing task and optimize the conventional circuit of block encoding of the sine function, where the parallel rotations $\bigotimes_{i=0}^{n-1} R_z(2^i a)$ are also required.

The third method attempts to achieve the "best of both worlds," with a depth that is typically below either of the earlier methods and a frugal T cost comparable to the second. This third, *independent catalyst tower* approach adopts a probabilistic scheme. Instead of preparing the required rotations in place, these towers can be seen as higher-level magic state factories. An example of a three-layer independent catalyst tower is depicted in Fig. 5, where we omit all the catalyst states for simplicity; to generalize it, concatenate one CT block at the top and one at the bottom. The resource states $R(2^i\theta)|+\rangle$ produced by a catalyst tower can then be used to implement a rotation on the target qubit via gate teleportation. However, since the rotation angles can be arbitrary, there is no efficient way to correct for the "wrong" measurement outcome of the gate teleportation directly, so we adopt the repeat-until-success approach with a success probability of 0.5. A wrong measurement outcome indicates that the implemented operation was a rotation of the desired angle but in the opposite direction. We now need to teleport a rotation with twice the target angle, which can correct the previous failure into a success, again with a probability of 0.5. This process is repeated on failure, and the angle doubles every trial until a measurement "success." Therefore, the expected measurement depth scales only logarithmically in the number of parallel rotations and thus logarithmically in n. This comes with the penalty that we somewhat increase the T count compared to the in-circuit towers, as we are implementing more rotations on average.

The repeat-until-success process thus involves a doubling of the desired angle, and this is in addition to the binary pattern already occurring within each Ξ_z block; thus, the towers' ability to generate multiple rotation resources, related to one another by powers of two, is well matched to our task. We design each independent catalyst tower such that

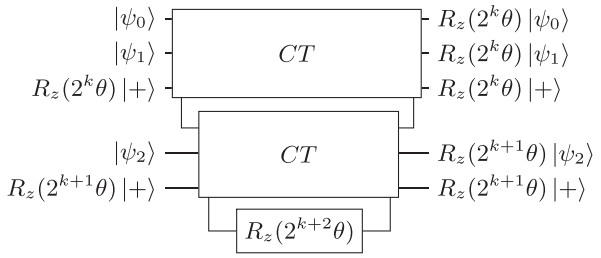


FIG. 3. The construction of a two-layer tower using the compact CT block.

a single run of the circuit produces the required species of rotations, which closely caps the expected number of rotations required by a group of n qubits (so we only need $S+1$ such towers in principle). However, due to the probabilistic nature of this approach, in practice, it is desirable to have a reservoir of the rotation resource states to account for any statistical fluctuation. As a side note, more generally, the independent catalyst towers are also beneficial in the circuit where multiple rotation gates of the same angle are required (and the circuit is repeatedly applied). One then needs to slightly modify the structure of the tower in Fig. 5 and implement multiple towers of different layers to match the tower outputs with the distribution of the repeat-until-success scheme.

We note that in Ref. [39], both ways of utilizing catalyst circuits, namely, the in-circuit and independent mode, are explored using the basic circuit to produce Clifford hierarchy rotations. However, in this work, we achieve our efficiency by composing multitier catalyst towers and matching their output to the rotations required by the circuit. As we will see in the next section, the advantage of introducing the composed towers is that, after the initial synthesis of single-qubit catalyst states, they achieve a better scaling of the T count when the circuit is repeated.

We compare the aforementioned piecewise function implementation to the method based on a parallelization of linear interpolation via QROM as described in Ref. [21]. QROM is used to load constants in the linear interpolation of a function $\tilde{f}(x) = a_i x + b_i$, where the slope and intercept a_i, b_i vary with x. The calculation of $|\tilde{f}(x)\rangle$ can then be performed with a single multiplication and addition, with the phase applied via addition into a phase gradient state. We discuss in more detail the choices we make for the arithmetic components in Appendix E and provide a circuit demonstrating the parallelized QROM approach in Fig. 6.

V. COST ANALYSIS

We use measurement depth as the metric for circuit depth and let a single-qubit measurement in Z or X basis have depth 1. We use this metric instead of T depth to account for teleporting the rotation resource states and also the uncomputation of the AND gate, which have the same delay as the conventional T-state teleportation circuit on the fault-tolerant level (assuming we are not limited by the magic state production rate). Therefore, T-state injection (the T depth), rotation resource state injection, and the uncomputation of AND all have measurement depth 1.

Let R_T be the T count of a rotation gate synthesized using Clifford+T. For a given accuracy ϵ, the optimal T count for deterministic and ancilla-free approximation of a rotation gate is $3\log_2(1/\epsilon)$ [28]. Given that we are in a regime that allows for an arbitrary number of ancilla, we can use the fallback,

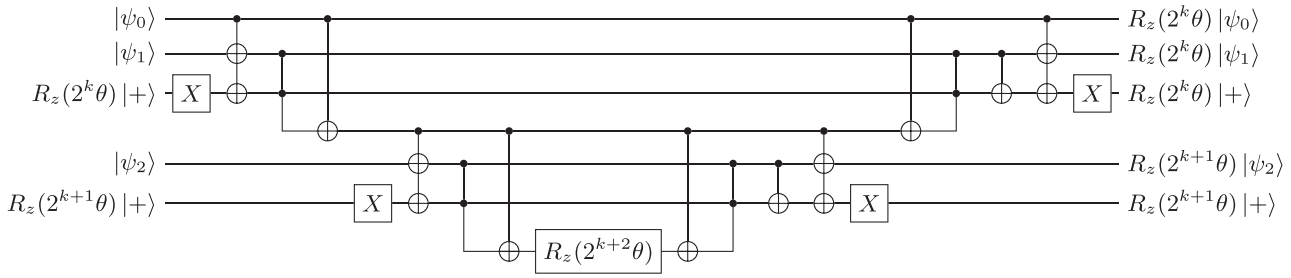


FIG. 4. The circuit for the two-layer tower in Fig. 3; note the V shape of the gates, which indicates the circuit is inherently sequential. When used as an in-circuit method, this is similar to the technique in Ref. [38].

or projective rotation, approximation method [29], which reduces the T count of a rotation [32] to

$$R_T \approx 1.03 \log_2 (1/\epsilon) + 5.75. \tag{4}$$

Using probabilistic mixing methods would further reduce this cost by a factor of 2 [30–32]. To implement a circuit with k rotations to precision ϵ_{circ}, take the error per rotation as $\epsilon = \epsilon_{\text{circ}}/k$ when calculating R_T. Since we are using Clifford+T, R_T is also the T depth and measurement depth.

We also note that the following expressions and calculations use one R_T parameter for all the rotations; this is enough for illustration here. However, for a more detailed analysis, one might want to use a different R_T for catalyst states than other rotations (the choice could be highly empirical) to further improve the performance.

For canonical gate synthesis, the T count of the circuit is

$$(S+1)(n+1)R_T + 8S(l-1), \tag{5}$$

where S is the number of sections and n is the number of qubits in a register. The first term is the contribution from the rotation towers, and the last term is due to the flag blocks. For the flag blocks we need $2S$ l-controlled Toffoli ($l = \lceil \log_2 S \rceil$), which can be constructed from $l-1$ regular Toffoli. Using the construction in Ref. [40], a Toffoli can be constructed with a T count of 4 and injected into the circuit with depth 1. Combined with the binary tree structure in Ref. [41], an l-controlled Toffoli has a T count of $4l-4$ and measurement depth $\lceil \log_2 l \rceil$.

The measurement depth of the circuit is

$$R_T + 2\lceil \log_2 l \rceil. \tag{6}$$

This method can alternatively be coupled with state injection to reduce circuit depth, at the cost of increased T count. The required rotations are synthesized as resource states, then injected in parallel with a success probability of 0.5. We expect to require two such resource states per rotation angle, hence the increase in T count to

$$2(S+1)(n+1)R_T + 8S(l-1). \tag{7}$$

The measurement depth is reduced to

$$\lceil \log_2(2.5(S+1)(n+1)+1.5) \rceil + 2\lceil \log_2 l \rceil, \tag{8}$$

which we explain further at the end of this section.

Meanwhile for the in-circuit catalyst towers, the T count of the circuit for r rounds of repetition (that is, r invocations of the phase oracle) is

$$(S+1)\{[R_T \cdot (n+1) + 4n] + (r-1)(R_T + 4n) + r \cdot R_T\} + 8Sr(l-1) \tag{9}$$

and the measurement depth of the circuit is

$$R_T + 2n + 2\lceil \log_2 l \rceil, \tag{10}$$

where the first two terms are the contribution from the towers. Each n-layer tower also needs $2n+1$ extra ancillary qubits, which is $(S+1)(2n+1)$ ancillary qubits in total (this does not include the ancillae used by the fallback synthesis method).

Finally, for the independent catalyst towers, the T count of the circuit for r rounds is

$$(S+1)\{[R_T \cdot (2n+1) + 8n] + (r-1)[R_T + 8n]\} + (S+1)\{(R_T \cdot 2 + 4) + (r-1)(R_T + 4)\} + 8Sr(l-1), \tag{11}$$

where the first term is the contribution of the independent towers associated with the large rotation blocks and the second term is from the one-layer independent towers that deal with the single-qubit γ_i rotations. This way of doing the γ_i rotations is more costly in terms of T count but it reduces the depth of the circuit.

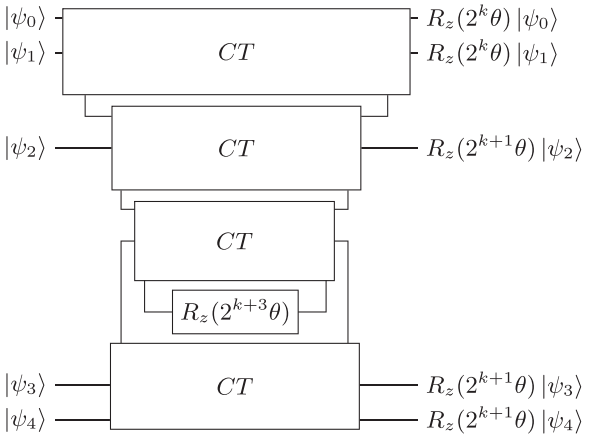


FIG. 5. An example of an independent catalyst tower with three layers [one way to see this is that the tower is seeded by $R_z(2^{k+3}\theta)$ and the lowest power output is $R_z(2^k\theta)$]. The catalyst states are omitted for simplicity. For the tower to produce resource states to be used in gate teleportation, set all $|\psi_i\rangle$ to $|+\rangle$ states.

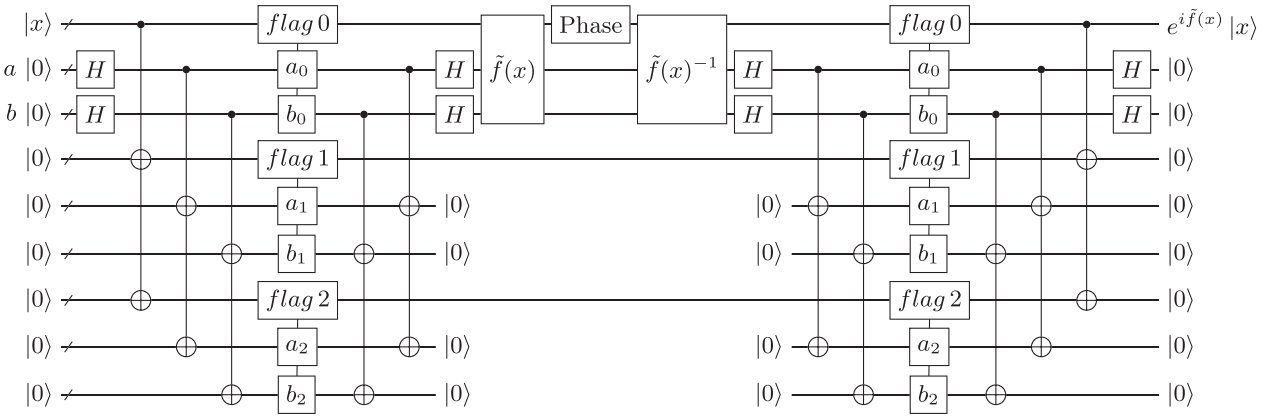


FIG. 6. Illustration of the parallelization of the application of a phase via QROM and linear interpolation as described in Ref. [21], using only one phase application to save resources. Here, a_i and b_i indicate applying circuits that apply a string of Pauli Zs, loading in the binary value of the slope and intercept for that section when condition on flag i is set. The four bundles of ancilla qubits used for loading the a_i and b_i are switched off after fan-in; these qubits can be repurposed during the $\tilde{f}(x)$ and phase computations. They are then reinitialized to be used in the second layer of flag computations. Note that as with Fig. 1, this circuit can be optimized using the methods outlined in Appendix A.

The expected measurement depth of the circuit is

$$\lceil \log_2(2.5(S+1)(n+1)+1.5)\rceil + 2\lceil \log_2 l\rceil. \tag{12}$$

Intuitively, when applying M parallel rotations using the repeat-until-success scheme, $M/2$ rotations are expected to fail, which requires a further round of rotations with $M/4$ expected failures, etc. The contribution to depth of this process is captured by the first term of Eq. (12), which is logarithmic in the number of parallel rotations. We provide a derivation of this expected depth in Appendix D, with the expression above being a numerical fitting that is within 0.04 of the true expectation for all $m \leqslant 10^6$, where m is the number of parallel rotations. For completeness, an n-layer ($n \geqslant 2$) independent tower uses $6n-5$ qubits and has a measurement depth of $R_T + 2n$. Note that this depth of the independent towers (which act like magic state factories) scales worse than the depth of the phase oracle circuit that consumes the resource states [Eq. (12)]. Therefore, in the cases where the phase oracle dominates the depth of the full circuit being repeated, one should use $R_T + 2n$ as the phase oracle depth. As the focus of this work is minimizing the depth, we note a relevant mitigation: one can break down an n-layer independent tower into several shorter towers, e.g., $n-1$ one-layer towers (in Fig. 2) with constant depth, at the cost of consuming more T states. The modular nature of the CT blocks makes this reconstruction simple.

However, in many applications the phase oracle is used as a subroutine of a much longer circuit, and such cases are suitable for the independent towers paradigm. For example, in the option pricing application [42] in the next section, each repetition for amplitude estimation involves preparing Gaussian states, followed by quantum arithmetic, and then applying the phase oracle. This allows enough time for resource state production to take place between the instances of the oracle.

The above expressions are summarized in Table I. From there it is clear that for repetition count $r = 1$, neither catalyst method gives any advantage over direct gate synthesis (without injection). However, only a modest number of repetitions is required to reach the break-even point for T count; for the in-circuit tower it needs

$$r > \left(1 - \frac{1}{n} - \frac{4}{R_T}\right)^{-1}, \tag{13}$$

while for the independent towers it requires more

$$r > \left(1 - \frac{n+2}{2n+1} - \frac{4}{R_T}\right)^{-1} \tag{14}$$

For the option-pricing application in the next section, where $S = 36$, $n = 15$, $\epsilon_{\text{circ}} = 10^{-3}$, and $r = 200$, $l = 6$, the in-circuit and independent towers need two and four rounds to bring advantage, respectively. Note that this analysis is at the fault-tolerant level. For a more complete calculation of these thresholds, one needs to consider the spacetime volume of the towers and the T factories at the quantum error-correcting code level.

As a rule of thumb for choosing between the towers when r is larger than the threshold value, if the goal is to minimize the T count then one should use the in-circuit towers. If the goal is to minimize the circuit depth (even when r is small), one might want to use the independent towers. As for the QROM method, it can be competitive in terms of T count when n is small, but one should note that the scaling is worse than linear (see discussions in Sec. VI B).

Finally, we used T count and T depth above as our metrics for the resource costs of our oracles. However, we note that recent development in magic state production [43] indicates that T gates can be generated at a cost comparable to a lattice surgery CNOT . Therefore, it might be worth noting that $O(S\,n)$ CNOT gates are also required for the circuit.

VI. EXAMPLES

In this section, we present explicit piecewise solutions and consequent resource counts for contexts involving payoff functions for option pricing, the Coulomb interaction, and a "double dot" confining potential.

TABLE I. The T count and measurement depth of the four options discussed in Sec. IV. For the independent-towers method, we assume the depth for resource state production does not dominate (see discussions in Sec. V). The expressions are evaluated for the three examples in Sec. VI, i.e., option pricing (accuracy, $\epsilon_{\text{circ}} = 10^{-3}$), Coulomb interaction ($\epsilon_{\text{circ}} = 10^{-3}$), and quantum dots confining potential ($\epsilon_{\text{circ}} = 10^{-2}$). The numerical results are in the form T count per round/measurement depth per round. We note that, in these examples which have relatively small n values, the $O(n^{\log_2(3)})$ T count of the linear QROM method outperforms some of the other methods that also depend on S (see Sec. VI B). We also note that the QROM method is the only method whose T count or depth does not increase with S, making it ultimately the preferred choice for sufficiently complex functions.

	T count (for r rounds)	Measurement depth (per round)	Pricing	Coulomb	Quantum dots
Gate synthesis	$r(S+1)(n+1)R_T + 8Sr(l-1)$	$R_T + 2\lceil\log_2 l\rceil$	16 536/32	3582/27	2214/23
Gate synthesis with injection	$2r(S+1)(n+1)R_T + 8Sr(l-1)$	$\lceil\log_2(2.5(S+1)(n+1)+1.5)\rceil + 2\lceil\log_2 l\rceil$	31 633/17	6852/13	4116/12
In-circuit towers	$(S+1)\{[R_T\cdot(n+1)+4n] + (r-1)(R_T+4n) + r\cdot R_T\} + 8Sr(l-1)$	$R_T + 2n + 2\lceil\log_2 l\rceil$	5618/62	1476/45	1191/35
Independent towers	$(S+1)\{[R_T\cdot(2n+3)+8n+4] + (r-1)(2R_T+8n+4)\} + 8Sr(l-1)$	$\lceil\log_2(2.5(S+1)(n+1)+1.5)\rceil + 2\lceil\log_2 l\rceil$	8061/17	2042/13	1583/12
Linear interpolation with QROM	$n\cdot R_T + r(14\cdot 3^{\lceil\log_2(n)\rceil} + 66n)$	$9\lceil\log_2(n)\rceil + 1$	2126/37	1728/37	774/28

A. Payoff function for option pricing

As our first example, we use part of the payoff function for option pricing that appears in Ref. [44], in which the authors use quantum signal processing (QSP) to replace the quantum arithmetic in Ref. [42]. For this example, we need the variant which implements $(f(x)|0\rangle + \sqrt{1-f(x)^2}|1\rangle)|x\rangle$ (where we change the target definition slightly to match the formalism in Ref. [44] more closely), as this is required by the subsequent iterative quantum amplitude estimation [45] step in their algorithm. For their use case, the circuit can be repeatedly applied up to $\sim 10^2$ iterations [46] and each iteration calls the oracle twice.

The target function to encode is

$$f(x) = \sqrt{1 - \left(K_T - e^{x^2 p} e^{-s}\right)}, \tag{15}$$

with the parameters $K_T = 1$, $p = 5$, and $s = 2^5$. This closely resembles Eq. (46) in Ref. [44] with the parameters of Fig. 4(a) of the same reference. The target here differs from the one used in Ref. [44] by the square of x in the exponent, which is required for QSP in Ref. [44] to work, but we can omit it here.

Figure 7 shows an intuitive visualization of the piecewise approximation. The depicted segmentation uses a quantum state of only seven qubits and a high tolerance threshold of

$$\max_x |f_{\text{target}}(x) - f_{\text{segmented}}(x)| \leqslant 10^{-2}, \tag{16}$$

as mentioned in the figure caption. This results in a total of nine segments. However, practically more relevant parameters would be, for example in this case, 15 qubits and a tolerance of only 10^{-3}, which then results in 36 segments. From Refs. [42,44], we use $S = 36$, $n = 15$, $\epsilon_{\text{circ}} = 10^{-3}$, $r = 2 \times 10^2$, and $l = 6$ for the expressions in Sec. V, and the cost of each method is shown in Table I. We can see that the in-circuit towers can reduce the T count per round by a

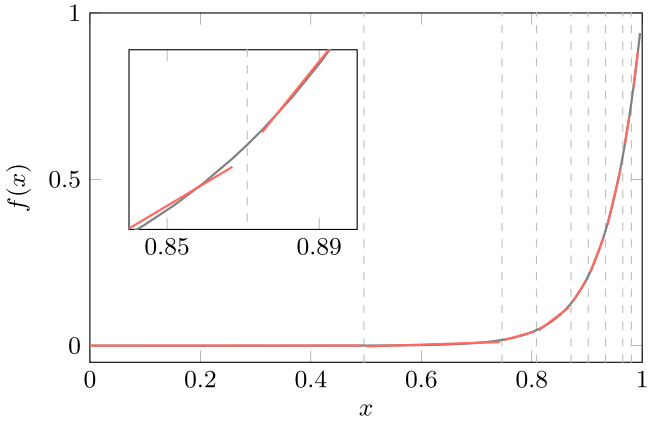


FIG. 7. Sectioning of the function in Eq. (15) with the parameters mentioned in the text thereafter. The target function is in gray, and the linear segments used for approximating the target are in red. Because of the similarity of the curves, the inset shows a close-up of segment edges. Dashed vertical lines show the edges of segments. For better visualization, the depicted segmentation uses a qubit count of 7 and a somewhat large tolerance between the target and segmented functions of $\max_x |f_{\text{target}}(x) - f_{\text{segmented}}(x)| \leqslant 10^{-2}$. More realistic values for these hyperparameters and their implications are discussed in the main text. Note that the possible x values are not continuous, but given by the discretization representable by the used number of qubits. Thus, the gaps in the linear approximation signify that there are no data points (quantum states) between the last value of the previous and the first value of the next linear section.

factor of 2.9 but have a factor of 1.9 increase in depth, in comparison to direct gate synthesis. The independent towers achieve a factor of 2.1 reduction in T count and a factor of 1.9 reduction in depth.

The width-depth trade-off is also favorable in this case. The quantum signal processing method in Ref. [44] results in a qubit count ∼30 and a T depth of ∼5700. In comparison, the deterministic in-circuit towers, for example, need $(S+1)(3n+2) \sim 1700$ qubits with measurement depth 62. So, our approach has a factor of ∼57 increase in width but achieves a factor of ∼92 reduction in depth. This improvement in depth can be vital for applications like derivative pricing where the time, usually on the scale of seconds, is critical for delivering quantum advantage. It is also worth noting that the prior quantum arithmetic step required by the option pricing algorithm [44] is much wider compared to the phase oracle, so the width-depth trade-off here does not actually incur new qubit overhead. In essence, the qubits are available anyway and thus *may as well* be put to use in the fashion we describe.

Note that on a larger scope, since the phase oracle is usually just a subroutine of an algorithm, reusing the catalyst (so they exist throughout the algorithm) states can cause the width of other parts of the circuit to increase.

B. Coulomb interaction

The next illustration we show is that of approximating a Coulomb potential, as is required by many algorithms in the context of materials science or chemistry. Prominent examples are ones leveraging grid-based methods to encode the state of (electronic) systems directly in the amplitudes of a register [47,48]. Such methods involve using quantum registers to represent particles on a discretized grid and often use the split-operator quantum fourier transform (SO-QFT) method to drive the dynamics. When combining with Trotterization, the method can apply time evolution by repeatedly transforming between real space and momentum space. The potential terms of the first-quantized Hamiltonian can be approximated as diagonal in real space (and the kinetic terms are diagonal in momentum space). Our methods for effecting a phase $\exp(iV(\mathbf{r}))$ onto the amplitudes is a natural fit for this simulation problem with $V(\mathbf{r})$ being the Coulomb potential. It has been shown in Ref. [49] that such simulation algorithms have superquadratic speedup (quartic in certain regimes) over classical methods, while the effort to establish exponential speedup remains an active area of research [50].

The one-dimensional (1D) example shown in Fig. 8 with $f(x) = V(x)$ uses nine qubits and tolerates a maximum deviation from the target function by 10^{-3}. The singularity at $x = 0$ is replaced by $f(0) = 2^9 \times 30/16$, as suggested by Ref. [48]. For comparability with the other examples, the function is also scaled down to have a maximum value of 1. This example requires 13 sections to approximate to the desired accuracy. Based on Ref. [47], we might need at least $r \sim 500$ to give a useful simulation (possibly far more). Assuming $S = 13$, $n = 9$, and $\epsilon_{\text{circ}} = 10^{-3}$ for convenience, the cost of each method is computed in Table I. It might appear that the register size $n = 9$ suggests a small problem size. Yet the grid-based method works by allocating relatively small registers of this kind to each particle (or particle dimension),

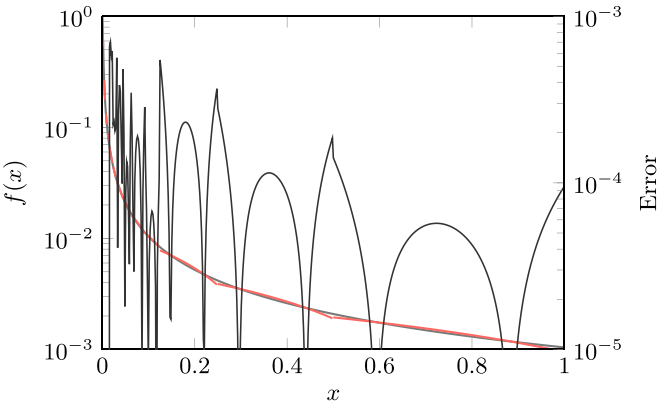


FIG. 8. Piecewise linear approximation of a (singularity-lifted) 1D Coulomb potential using nine qubits to represent x and a tolerance of 10^{-3}. The target is the gray x^{-1} curve with the (interrupted, as in Fig. 7) approximation in red on top. The black line shows the error across x. This approximation requires 13 sections in the piecewise representation to acquire the desired accuracy.

while the total number of modeled particles can be large. Importantly, we only need to apply two-body Coulomb interactions to these modest-sized registers, albeit this should be performed in parallel over multiple registers. Indeed, this also applies to the quantum dot example in the next section. It is worth noting that the tolerable level of imperfection ϵ_{circ} is highly dependent on the exact use case and will be determined in combination with other parameters like time step. We use $\epsilon_{\text{circ}} = 10^{-3}$ here as an example, since the error from other sources, e.g., the Trotterization formula, may be expected to be of this order [51].

We note that for these parameters, it seems that the independent towers outperform the QROM approach in depth while the QROM approach is more advantageous in terms of T count. When moving to larger n, the $O(n^{\log_2(3)})$ term in T count will dominate over the linear term when using multiplication methods that are depth competitive with the independent towers; refer to Sec. II for discussion. In fact, if the precision is slightly relaxed and S is reduced from 13 to 10, then the QROM approach's advantage in T count is eliminated.

C. Double quantum dot

As our final example, we examine a double quantum dot potential, which appears in the study of quantum dot structures. Typically, they are explored using classical techniques [48,52,53] for the development of semiconductor-based quantum computers. However, future quantum devices could help simulate more general nanostructures using grid-based methods [47,48]. Within this formalism, as in the previous example, one will have to apply functions of the type $\exp(iV(\mathbf{r}))$ to a given quantum state, with $V(\mathbf{r})$ meaning a double dot potential as in Fig. 9.

Here, we focus on a one-dimensional double quantum dot potential $V(x)$ and a quantum state of six qubits, but this approach can be extended to multiple dimensions supposing that the potential is separable (or nearly so). Such a potential is generated by a Poisson solver applied to a realistic device.

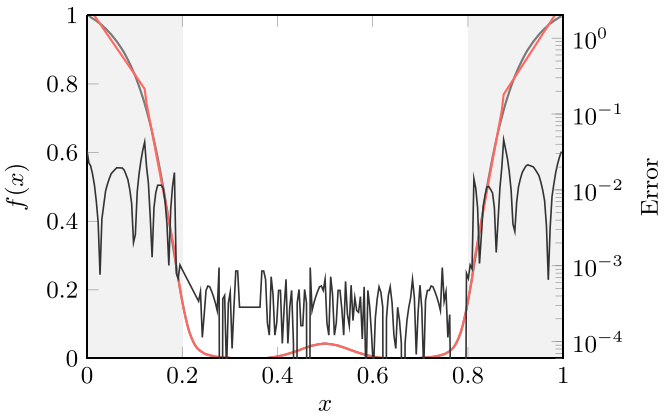


FIG. 9. Sectioning of a 1D double quantum dot potential. For this type of function, we can reduce the number of segments by choosing different tolerances in different zones. The gray areas can tolerate more errors as long as they provide enough confinement, as explained in the main text. For illustration purposes, we chose tolerances equal to 10^{-3} and 10^{-1} in the nonshaded and shaded areas, respectively. The error between the curves is plotted in black.

In Fig. 9, we plot a representation of the piecewise approximation. Unlike the other functions studied above, it is not necessary to accurately represent the potential over the entire range of x. As long as the potential in the shaded region provides adequate confinement, important properties of the double quantum dot structure. such as the wave function of the first two eigenstates (used to define the qubit states and extract the exchange coupling [48]), are approximately preserved. This allows us to significantly reduce the number of segments needed.

For instance, using a tolerance of 10^{-3} over both zones results in a 55-segment approximation, achieving an infidelity approximately equal to $\mathcal{I}_g \approx 10^{-8}$, defined as $\mathcal{I}_g = 1 - |\langle g|g_{pw}\rangle|$ with $|g\rangle$, the exact ground state (as computed in Ref. [48]) and the approximate one $|g_{pw}\rangle$ obtained using the piecewise approximation. By applying the same tolerance within the nonshaded region and a tolerance of 10^{-1} in the shaded region, we can reduce the number of segments to 31 while obtaining an infidelity $\mathcal{I}_g \approx 10^{-6}$, which is enough for our purposes. This is because the wavefunctions vanish in the shaded region, making them robust to modifications of the potential in that zone. It is worth noting that we obtain similar results for the infidelity $\mathcal{I}_e$ computed between the excited states.

Finally, one can further reduce the number of segments by using a tolerance of 10^{-2} and 10^{-1} in the nonshaded and shaded areas, respectively. This choice yields a 13-segment approximation while maintaining an infidelity $\mathcal{I}_g \approx 10^{-6}$.

We estimate a useful simulation would require at least 5×10^5 time steps [48]. Using $r = 5 \times 10^5$ (for the potential part of Hamiltonian only), $S = 13$, $n = 6$, and $\epsilon_{\text{circ}} = 10^{-2}$ for convenience. This is a suitable set of parameters for the 1D scenario, albeit in practice we may need to apply $V(x, y, z)$. The cost of each method is shown in Table I. For this large number of repetitions, we are in the asymptotic regime that, e.g., the in-circuit catalyst towers can apply n rotations ($n + 1$ rotations strictly speaking) by consuming only $R_T + 4n$ T states (one seed rotation plus n logical AND), which would cost $n \cdot R_T$ using the canonical gate synthesis.

VII. CONCLUSION

In this work, we study the task of applying a phase $\exp(i\, f(x))$ to a computational basis state $|x\rangle$, and the closely related task of applying a rotation dependent on $f(x)$. These tasks are well studied in the literature; our aim here is minimizing the depth (i.e., time cost) of the process. We consider a piecewise approximation to $f(x)$ and fully parallelize the process so that its rotation depth is unity.

Noting that in many applications, such "phase oracles" are applied numerous times within an algorithm, we explore a means to reduce the T-gate resource cost by use of catalyst towers, bespoke hierarchical structures based on the catalyst circuits from Ref. [35]. We find that the catalyst towers align very naturally with the requirements of the parallelized piecewise circuit, and can considerably reduce the T count for the rotation gates if the circuit is repeatedly applied. We also give explicit expressions for T count and measurement depth of the different methods and compare their behavior in three examples.

As for future research, it would be desirable to have a more detailed cost analysis of towers at the quantum error-correcting code level. As mentioned at the end of Sec. V, one may want to compare the spacetime volume of the T factories saved due to the reduction in T count with that of the towers added. Moreover, one must also account for the storage cost of the catalyst states, which could manifest as an increase in code distance.

ACKNOWLEDGMENTS

The numerical modeling involved in this study made use of the Quantum Exact Simulation Toolkit (QuEST) [54] via the QuESTlink [55] frontend. We are grateful to those who have contributed to all of these valuable tools. Aspects of this study were supported by the EPSRC projects Robust and Reliable Quantum Computing (RoaRQ, EP/W032635/1), Software Enabling Early Quantum Advantage (SEEQA, EP/Y004655/1), the EPSRC QCS Hub (EP/T001062/1), and the UKRI Future Leaders Fellowship (MR/Y015843/1). Richard Meister is partially supported by EPSRC Grant No. EP/W032643/1. Romy Minko and Benjamin Pring were partially supported by the Additional Funding Programme for Mathematical Sciences, delivered by EPSRC (EP/V521917/1) and the Heilbronn Institute for Mathematical Research.

APPENDIX A: CIRCUITRY

The computation and uncomputation of the logic AND gate (see Fig. 10), which cost 4 T states and a measurement.

We make heavy use of both fan-out and fan-in via CNOTs in this paper, hence it is worthwhile remarking upon implementation details for these primitives.

For fan-out to enable us to have parallel access to k copies of the n-qubit computational basis state $|x_1 \ldots x_n\rangle$, we want to implement n parallel copies of the map $|x_i\rangle\,|0^{k-1}\rangle \mapsto |x_i\rangle\,|x_i^{k-1}\rangle$, where $x_i \in \{0, 1\}$. Whilst certain

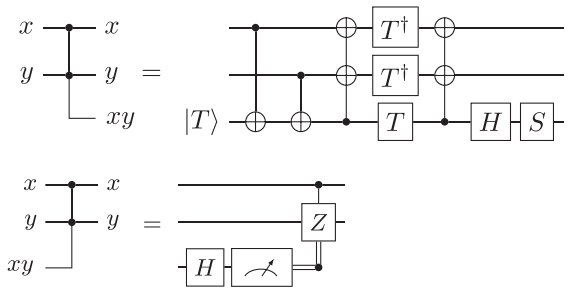


FIG. 10. The circuit diagram for the "corner" notation used in the main text, reproducing Fig. 3 of Ref. [22].

technologies allow for multitarget CNOT gates as a primitive [56], the real-time long-range routing might be a problem at the quantum error-correcting code level. Working with an abstract model of quantum computation for now, alternatively, by using one Hadamard gate and $k-1$ CNOT gates, we can prepare a k-qubit Greenberger–Horne–Zeilinger (GHZ) state $\frac{1}{\sqrt{2}}(|0\rangle|0^{k-1}\rangle + |1\rangle|1^{k-1}\rangle)$ and transport these qubits to where they will be required ahead of time. This state can then implement the required fan-out using a single CNOT gate from our control to the first qubit of the GHZ state to obtain $|x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle|0^{k-1}\rangle + |1\rangle|1^{k-1}\rangle) \mapsto |x\rangle \frac{1}{\sqrt{2}}(|x\rangle|0^{k-1}\rangle + |\bar{x}\rangle|1^{k-1}\rangle)$, followed by a Z-basis measurement of the second qubit of the resulting state, which gives us either 0, in which case we have $|x\rangle|x^{k-1}\rangle$, or 1, in which case we have $|x\rangle|\bar{x}^{k-1}\rangle$ and a Pauli correction of $X^{\otimes k-1}$ can be made to obtain the desired outcome.

For fan-in, where we wish to accomplish the map $|x\rangle|x^{k-1}\rangle \mapsto |x\rangle|0^{k-1}\rangle$, this can be accomplished via multi-CNOT gates, but it is easier to replace multiple-qubit operations which may require routing and additional circuitry with measurement-based uncomputation. Recall that if we have the two-qubit state $|x\rangle|x\rangle$ and perform a destructive X-basis measurement on the second qubit, we obtain the state $|x\rangle$ if we measure 0 and $(-1)^x|x\rangle$ if we measure 1. We can therefore perform $k-1$ such parallel X-basis measurements and perform the Pauli-Z correction conditioned upon the joint parity of these measurements.

One further trade-off is possible with regards to the T-count required for computing the flags in Figs. 1 and 6, which do not illustrate the ancilla required to implement the flags with minimal depth. We recall these flags will be an l-controlled Toffoli, for which the fastest method of implementing these functions is to structure them as a binary tree using $l-1$ quantum AND gates ($|a,b\rangle|0\rangle \mapsto |a,b\rangle|a\cdot b\rangle$), which can be implemented 4 T gates or a control-control-Z state. As soon as the final such AND gate is completed, the rotation stage can begin, and if we choose to keep these intermediate computations in memory, the second layer of flag computations can be completed in the same depth but without using any T gates via measurement-based uncomputation. The utility of whether this particular optimization is worthwhile is dependent upon whether the space could be used in a more efficient manner (whether as ancilla for other unitaries or for repurposing for magic state distillation, etc.) and if the rotation layer is implemented in a fast manner, this may be worthwhile.

APPENDIX B: HEURISTIC FOR SECTIONING FUNCTIONS

For the examples given in the main text we use a simple heuristic to divide the target function into (quasi-)linear segments. As we shall see, this method results in the particularly simple "flag" blocks of Fig. 1 mentioned in Sec. III, which is a simple l-controlled Toffoli.

The details of the used heuristic differ slightly depending on whether a phase should be applied to a state, i.e., $|x\rangle \mapsto \exp if(x)|x\rangle$, or whether an amplitude state should be prepared as in $|x\rangle|0\rangle \mapsto |x\rangle(f(x)|0\rangle + \sqrt{1-f(x)^2}|1\rangle)$. However, the basic principle remains the same. We will discuss the former case first, followed by a comment on the required changes for the latter case.

The task is to section the target function $f(x)$ into intervals Δx_j, where within the interval j, $A_j + B_j x \approx f(x)$, with some parameters A_j and B_j and $x \in \Delta x_j$. The following bisection method accomplishes this.

(i) Within some interval Δx_j, try to fit the parameters A and B such that $f(x) \approx A + Bx$.

(ii) If the maximum deviation is below some threshold δ, $\max_{x\in\Delta x_j}(|f(x) - (A+Bx)|) < \delta$, keep the interval Δx_j and return. Otherwise, split Δx_j in the middle, and recursively apply the same procedure for the left and right half separately.

Starting the above iteration with Δx_j being the whole interval of x results in the desired segmentation. Because of the specific construction of the intervals, in each recursive call, all x belonging to the same Δx_j have a common prefix of q_j (highest significance) qubits (corresponding to the recursion depth) in their bit-string representation, whose pattern we call $\vec{q}_j$. For example, $\vec{q}_j = (1,0,0,1,0)$ (most significant bit on the left) means that all states in the interval Δx_j start with the bit pattern 10010, followed by their varying lower-significance bits. Therefore, the *flag* gates in Fig. 1 are simple multi-(anti-)controlled Toffoli gates.

For the slightly different problem of creating the mapping $|x\rangle|0\rangle \mapsto |x\rangle(f(x)|0\rangle + \sqrt{1-f(x)^2}|1\rangle)$, the parametrized fit function changes from $A+Bx$ to $\cos(A+Bx)$; everything else can stay the same.

As a side note, one can generally quantify whether a function is suitable for piecewise approximation using the Lipschitz constant [57] of the function. A large Lipschitz constant may imply a poor piecewise approximation. Moreover, the number of segments S, the size of the quantum register n, and the tolerance in the approximation are implicitly related. One would intuitively expect that, e.g., to get a higher accuracy, one needs a larger register and a smaller tolerance; then the number of segments that come out of the bisection iteration would also be larger.

APPENDIX C: TILING CATALYST FACTORY TOWERS

Here, we provide discussion on tiling sequential applications of catalyst tower circuits to reduce idle qubits and reduce depth by a factor of 2 in the limit of a large number of layers n.

Figure 11 shows the sequential application of catalyst tower circuits, resulting in wasted space in the central region as the circuit must wait for the uncomputation on qubits 0,1

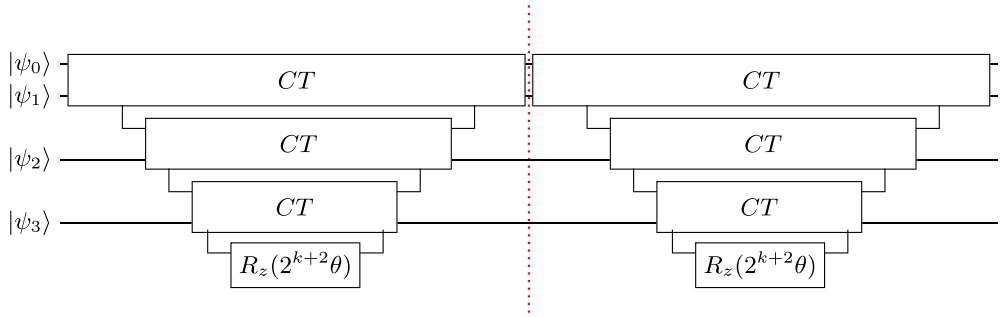


FIG. 11. Two applications of a three-layer tower, with wasted space in the center where the next tower is waiting for the previous tower to fully complete. The red dotted line indicates an undrawn layer of swapping out resource states and swapping in new states to have rotations applied. Catalyst states are not drawn for simplicity.

before beginning another application. We provide an alternative, shown in Fig. 12, that presents the application of towers, now with the orientations reversed. The qubits at the top of the tower can now immediately begin execution of the second circuit upon being freed. In this case, each layer of the tower now requires an additional catalyst qubit, as the rotation angles differ in the up and down orientations.

APPENDIX D: EXPECTED DEPTH OF PARALLEL REPEAT-UNTIL-SUCCESS ROTATIONS

As described in Sec. IV, rotations can be implemented using resource states in a repeat-until-success procedure [29], where gate teleportation succeeds with a probability of 0.5, and if failure occurs, another attempt is made with twice the angle: to fix the incorrect angle from the previous teleportation(s) and apply the desired rotation. For a single rotation, the expected depth of this procedure is 2, but when performing m of these rotations in parallel as in Fig. 1, we must wait for all the rotations to have finished before applying subsequent multiqubit gates. We therefore must account for the possibility that some of the rotations have a large number of failures, and the expected depth increases as we increase m.

Given the random variables D_1 and D_m which are the depth of the repeat-until-success procedure for 1 rotation and m rotations, respectively, we can write the probability that $D_m \leqslant d$ as

$$P(D_m \leqslant d) = P(D_1 \leqslant d)^m \quad = (1 - P(D_1 > d))^m$$
$$= (1 - 2^{-d})^m \tag{D1}$$

as we simply require that all m rotations have terminated by depth d.

We can then find the probability that the depth for m rotations is exactly d by using this cumulative probability to find the probability density:

$$P(D_m = d) = P(D_m \leqslant d) - P(D_m \leqslant d - 1), \tag{D2}$$

resulting in the expected depth:

$$\mathbb{E}[D_m] = \sum_{d=1}^{\infty} d((1 - 2^{-d})^m - (1 - 2^{-d+1})^m) \tag{D3}$$

APPENDIX E: COMMENTS ON LINEAR INTERPOLATION VIA QROM

Here, we provide further comments on the application of a phase of a linear interpolation of a function $\tilde{f}(x) = a_i x + b_i$, where the slope and intercept a_i, b_i vary with x via QROM and arithmetic functions [21]. The calculation of $|\tilde{f}(x)\rangle$ can then be performed with a single multiplication and addition, with the phase applied via addition into a phase gradient state.

We begin by making choices for the addition and multiplication methods used to calculate $\tilde{f}(x)$, choosing Brent-Kung carry look-ahead addition using logic AND gates where possible as described in Ref. [58], and the Karatsuba multiplication method from Jang *et al.* [23], which has the lowest known depth of quantum multiplication. Taking the addition, and converting from the Toffoli depth provided in Ref. [58], we need $3\lceil \log_2 n \rceil - 1$ measurement depth and $\sim 22n$ T count.

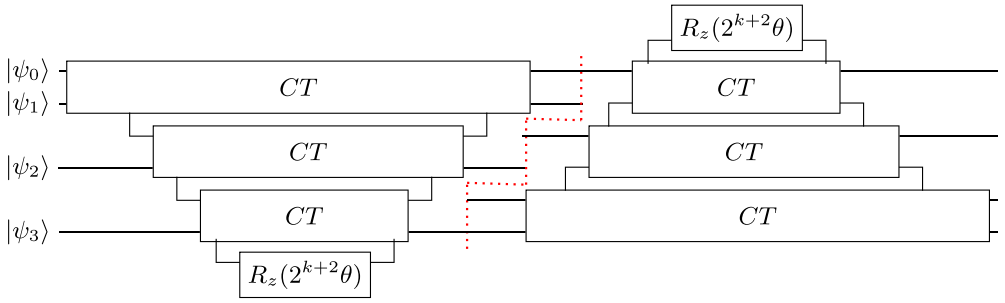


FIG. 12. Two applications of a three-layer tower tiled to reduce qubit idling. The red dotted line indicates an undrawn layer of swapping out resource states and swapping in new states to have rotations applied. In this case, two catalyst states are required for each layer of the tower, one for each of the up and down orientations.

The multiplication comes to a measurement depth of 4 (ignoring the log-depth Clifford circuits that are a component of this circuit) and the number of T gates required for level-p Karatsuba multiplication is $7 \times 3^p (n/2^p)^2$, which for n a power of 2 and fully parallelized to $p = \log_2(n)$ is $7 \times 3^{\log_2(n)}$ (this expression is only exact when n is a power of 2, otherwise $7 \times 3^{\lceil \log_2(n) \rceil}$ can be used as an upper bound) [23]. We note that this implementation of Karatsuba multiplication also requires a number of ancilla qubits scaling as $3 \times 3^{\log_2 n}$ [this method with its $O(n^{\log_2(3)})$ qubit count scaling is the only one presented in Table I with a superlinear scaling in n]. Overall, we find a measurement depth of $9\lceil \log_2 n \rceil + 1$ for the computation and uncomputation of $\tilde{f}(x)$, along with the central phase-gradient addition. We can therefore conclude that even when using the most depth-optimized multiplication circuits, the depth of this method is outperformed by the injection of resource states prepared by independent catalyst towers in the regime where a simple piecewise approximation of the function is applicable.

The overall T costs for the phase oracle using this circuit are given by two applications of multiplication and addition, addition with the phase gradient state, and the original creation of the phase gradient state which requires n single-qubit rotations.

We also note that the parallelization of this method will result in resource state savings when compared to the catalyst tower method described in Sec. IV, as the data from QROM output can be placed onto a single register, even when computed in parallel, meaning that the arithmetic and application of phase can be performed only once, resulting in resource state savings, and only requiring one phase gradient state. See Fig. 6 for an example circuit, with the SelectSwap method of Ref. [59] being an alternative for the parallel loading of the a_i, b_i, requiring fewer ancilla qubits but leaving many of them dirty for the central section of the circuit.

A further optimization is possible to improve the scaling of the central multiplication. The register x can take 2^n discrete values, but when the piecewise decomposition of the function has a largest segment of size Δx_L containing 2^q values, it is instead possible to perform the multiplication only against the *offset* from the start of this segment, with cost determined by the largest segment as $O(7 \times 3^{\log_2(q)})$, at the expense of performing extra additions and loading the start locations via QROM (although this can in some cases be avoided if the segments follow convenient patterns, such as following powers of two as described in Ref. [21]). However, functions that are desirable to implement in a piecewise fashion often have large segments that take up a large fraction of the total domain as in Figs. 7 and 8, so the possible improvement should be weighed against the slight additional cost from the extra addition in $a_i'(x - s_i) + b_i'$, where s_i is the value where the ith segment begins.

[1] P. W. Shor, SIAM Rev. **41**, 303 (1999).
[2] A. W. Harrow, A. Hassidim, and S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009).
[3] T. Hogg and D. Portnov, Info. Sci. **128**, 181 (2000).
[4] S. McArdle, S. Endo, A. Aspuru-Guzik, S. C. Benjamin, and X. Yuan, Rev. Mod. Phys. **92**, 015003 (2020).
[5] D. Ackley, *A Connectionist Machine for Genetic Hillclimbing* (Springer Science & Business Media, Berlin, 2012), Vol. 28.
[6] M. F. Gonzalez-Zalba, S. de Franceschi, E. Charbon, T. Meunier, M. Vinet, and A. S. Dzurak, Nat. Electron. **4**, 872 (2021).
[7] X. Huang, T. Kosugi, H. Nishi, and Y.-i. Matsushita, Quantum Inf. Process. **24**, 85 (2025).
[8] L. K. Grover, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (ACM Press, New York, 1996), pp. 212–219.
[9] D. R. Simon, SIAM J. Comput. **26**, 1474 (1997).
[10] A. Gilyén, S. Arunachalam, and N. Wiebe, in *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms* (SIAM, 2019), pp. 1425–1444.
[11] A. Y. Kitaev, Electron. Colloquium Comput. Complex. **TR96** (1995).
[12] L. Grover and T. Rudolph, arXiv:quant-ph/0208112.
[13] M. Ozols, M. Roetteler, and J. Roland, ACM Trans. Comput. Theory **5**, 1 (2013).
[14] S. McArdle, A. Gilyén, and M. Berta, arXiv:2210.14892.
[15] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, Phys. Rev. X **8**, 041015 (2018).
[16] D. W. Berry, C. Gidney, M. Motta, J. R. McClean, and R. Babbush, Quantum **3**, 208 (2019).
[17] J. Lee, D. W. Berry, C. Gidney, W. J. Huggins, J. R. McClean, N. Wiebe, and R. Babbush, PRX Quantum **2**, 030305 (2021).
[18] D. W. Berry, N. C. Rubin, A. O. Elnabawy, G. Ahlers, A. E. DePrince, III, J. Lee, C. Gogolin, and R. Babbush, npj Quantum Inf. **10**, 130 (2024).
[19] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett. **100**, 160501 (2008).
[20] S. Jaques and A. G. Rattew, arXiv:2305.10310.
[21] Y. R. Sanders, D. W. Berry, P. C. S. Costa, L. W. Tessler, N. Wiebe, C. Gidney, H. Neven, and R. Babbush, PRX Quantum **1**, 020312 (2020).
[22] C. Gidney, Quantum **2**, 74 (2018).
[23] K. Jang, W. Kim, S. Lim, Y. Kang, Y. Yang, and H. Seo, Sensors **23**, 3156 (2023).
[24] J. Nie, Q. Zhu, M. Li, and X. Sun, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **42**, 4791 (2023).
[25] M. Suzuki, Commun. Math. Phys. **51**, 183 (1976).
[26] G. H. Low and I. L. Chuang, Quantum **3**, 163 (2019).
[27] D. W. Berry and A. M. Childs, Quantum Inf. and Comput. **12**, 29 (2012).
[28] N. J. Ross and P. Selinger, Quantum Inf. Comput. **16**, 901 (2016).
[29] A. Bocharov, M. Roetteler, and K. M. Svore, Phys. Rev. A **91**, 052317 (2015).
[30] E. Campbell, Phys. Rev. A **95**, 042306 (2017).
[31] M. B. Hastings, Quantum Inf. Comput. **17**, 488 (2017).
[32] V. Kliuchnikov, K. Lauter, R. Minko, A. Paetznick, and C. Petit, Quantum **7**, 1208 (2023).

[33] G. Duclos-Cianci and D. Poulin, Phys. Rev. A **91**, 042315 (2015).
[34] E. T. Campbell and J. O'Gorman, Quantum Sci. Technol. **1**, 015007 (2016).
[35] C. Gidney and A. G. Fowler, Quantum **3**, 135 (2019).
[36] A. C. Vazquez, R. Hiptmair, and S. Woerner, ACM Trans. Quantum Comput. **3**, 1 (2022).
[37] T. Häner, M. Roetteler, and K. M. Svore, arXiv:1805.12445.
[38] G. Wang and A. Kan, Quantum **8**, 1504 (2024).
[39] G. J. Mooney, C. D. Hill, and L. C. Hollenberg, Quantum **5**, 396 (2021).
[40] C. Jones, Phys. Rev. A **87**, 022328 (2013).
[41] Y. He, M.-X. Luo, E. Zhang, H.-K. Wang, and X.-F. Wang, Int. J. Theor. Phys. **56**, 2350 (2017).
[42] S. Chakrabarti, R. Krishnakumar, G. Mazzola, N. Stamatopoulos, S. Woerner, and W. J. Zeng, Quantum **5**, 463 (2021).
[43] C. Gidney, N. Shutty, and C. Jones, arXiv:2409.17595.
[44] N. Stamatopoulos and W. J. Zeng, Quantum **8**, 1322 (2024).
[45] D. Grinko, J. Gacon, C. Zoufal, and S. Woerner, npj Quantum Inf. **7**, 52 (2021).
[46] F. Labib, B. D. Clader, N. Stamatopoulos, and W. J. Zeng, arXiv:2405.14697.
[47] H. H. S. Chan, R. Meister, T. Jones, D. P. Tew, and S. C. Benjamin, Sci. Adv. **9**, eabo7484 (2023).
[48] H. Jnane and S. C. Benjamin, arXiv:2403.00191.
[49] R. Babbush, W. J. Huggins, D. W. Berry, S. F. Ung, A. Zhao, D. R. Reichman, H. Neven, A. D. Baczewski, and J. Lee, Nat. Commun. **14**, 4058 (2023).
[50] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica *et al.*, arXiv:2310.03011.
[51] T. N. Ikeda, H. Kono, and K. Fujii, Phys. Rev. Res. **6**, 033285 (2024).
[52] J. D. Cifuentes, P. Y. Mai, F. Schlattner, H. E. Ercan, M. K. Feng, C. C. Escott, A. S. Dzurak, and A. Saraiva, Phys. Rev. B **108**, 155413 (2023).
[53] M. M. E. K. Shehata, G. Simion, R. Li, F. A. Mohiyaddin, D. Wan, M. Mongillo, B. Govoreanu, I. Radu, K. De Greve, and P. Van Dorpe, Phys. Rev. B **108**, 045305 (2023).
[54] T. Jones, A. Brown, I. Bush, and S. C. Benjamin, Sci. Rep. **9**, 10736 (2019).
[55] T. Jones and S. Benjamin, Quantum Sci. Technol. **5**, 034012 (2020).
[56] D. Litinski and F. von Oppen, Quantum **2**, 62 (2018).
[57] H. H. Sohrab, *Basic Real Analysis* (Springer, New York, 2003), Vol. 231.
[58] S. Wang, A. Mondal, and A. Chattopadhyay, ACM Trans. Quantum Comput. (2025), doi:10.1145/3743691.
[59] R. Krishnakumar, M. Soeken, M. Roetteler, and W. Zeng, in *2022 IEEE/ACM Third International Workshop on Quantum Computing Software (QCS)* (IEEE, New York, 2022), pp. 75–82.