Quantum algorithms for powering stable Hermitian matrices

Guillermo González,[*] Rahul Trivedi,[*,†] and J. Ignacio Cirac
Max-Planck-Institut für Quantenoptik, Hans-Kopfermann-Strasse 1, 85748 Garching, Germany and Munich Center for Quantum Science and Technology (MCQST), Schellingstrasse 4, D-80799 Munich, Germany

(Received 8 April 2021; revised 4 June 2021; accepted 10 June 2021; published 25 June 2021)

Matrix powering is a fundamental computational primitive in linear algebra. It has widespread applications in scientific computing and engineering and underlies the solution of time-homogeneous linear ordinary differential equations, simulation of discrete-time Markov chains, or discovering the spectral properties of matrices with iterative methods. In this paper, we investigate the possibility of speeding up matrix powering of sparse stable Hermitian matrices on a quantum computer. We present two quantum algorithms that can achieve speedup over the classical matrix powering algorithms: (i) a fast-forwarding algorithm that builds on construction of Apers and Sarlette [Quantum Inf. Comput. **19**, 181 (2019)] and (ii) an algorithm based on Hamiltonian simulation. Furthermore, by mapping the N-bit parity determination problem to a matrix powering problem, we provide no-go theorems that limit the quantum speedups achievable in powering non-Hermitian matrices.

DOI: 10.1103/PhysRevA.103.062420

I. INTRODUCTION

Recent years have seen rapid progress in the development of quantum computing hardware, and there have already been experimental demonstrations of quantum computations that are believed to be hard to simulate on classical computers [1,2]. While this progress in hardware has brought us closer to the monumental goal of building a fault-tolerant quantum computer, it has also provided us with access to noisy quantum hardware which might already solve problems that are hard for classical computers [3]. From a theoretical standpoint, it has become important to discover algorithms that can provide speedup over their classical counterparts on both fault tolerant quantum computers and near-term noisy quantum hardware.

Quantum computers are known to offer exponential speedup in simulating the physics of quantum systems—near-optimal algorithms have been developed for the simulation of Hamiltonian dynamics [4–6], Lindbladian dynamics [7–10], and steady state (finite temperature or ground state) properties of Hamiltonians [11–14]. Several techniques used for simulating quantum systems have been generalized to accelerate more fundamental linear algebra computational primitives—exponential quantum speedup in the solution of systems of linear equations have been obtained [15–17], and quantum speedups have also been shown in solving ordinary differential equations [18,19] and partial differential equations [20].

Another fundamental computation that can be accelerated on quantum computers is matrix powering i.e. computing a matrix-element $v^\dagger A^t u$ given access to the matrix $A \in \mathbb{C}^{N\times N}$, a positive integer power t and vectors $v, u \in \mathbb{C}^N$. This is a computational primitive which appears in various applications, including but not limited to solving linear differential equations, simulating discrete-time Markov chains, and matrix inversion and eigenvalue computation using Krylov subspace methods. Classically, this problem can be solved by repeated matrix multiplication in time $O[\text{poly}(N)Dt]$, where D is the sparsity of the matrix A. Without any assumptions on the matrix A, one approach to solve the matrix-powering problem on a quantum computer is to map it to a matrix inversion problem and use quantum algorithms for solving linear equations—this approach has been investigated in Refs. [18,19] in the context of solving linear time-homogeneous ordinary differential equations and has a run time of $O[\text{polylog}(N)\text{poly}(\varepsilon^{-1})\kappa_V \|v\|^2\|u\|^2 Dt]$, where κ_V is the condition number of the eigenvector matrix of A, to obtain $v^\dagger A^t u$ to a precision ε for stable matrices, thereby providing an exponential speedup in the matrix size over classical algorithms.

Furthermore, several authors have studied the problem of powering a stochastic matrix which arises in the context of simulating the dynamics of a discrete-time Markov chain [21–23]. Classically, the problem of powering a stochastic matrix can be solved efficiently to a precision ε with the Monte Carlo algorithm in time $O(Dt\|v\|^2\|u\|^2/\varepsilon^2)$—using the Monte Carlo algorithm is thus exponentially faster than using repeated matrix multiplication. While the quantum algorithms based on linear equation solvers do not provide an exponential speedup for stochastic matrix powering when compared to the classical Monte Carlo algorithm, there have been two proposals for achieving polynomial quantum speedups for this specific problem. One of the proposed algorithms is to use a reversible implementation of the classical Monte Carlo algorithm together with quantum amplitude estimation to achieve

[*]Both authors contributed equally.
[†]rahul.trivedi@mpq.mpg.de

a quadratic improvement in the dependence of the run time on precision as compared to the classical Monte Carlo algorithm [24]. This idea has been applied to propose solutions to the heat equation [25] and stochastic differential equations [26]. A different quantum speedup can be obtained for symmetric stochastic matrices by employing quantum walks [27–30]. In particular, Ref. [31] prepares a quantum state within an ε radius of $A^t u/\|A^t u\|$ in $O[D\|A^t u\|^{-1}\sqrt{t}\ \ln^{1/2}(\varepsilon^{-1}\|A^t u\|^{-1})]$. The same author generalized this algorithm to arbitrary Hermitian matrices A in Ref. [32]. While this algorithm obtains a quantum speedup over classical methods (an exponential speedup in N over repeated matrix multiplication, and quadratic speedup in t over the Monte Carlo algorithm), the dependence of the run time on $\|A^t u\|^{-1}$ can often make it polynomially slow in N.

In this paper we introduce two algorithms to compute $v^\dagger A^t u$ for stable Hermitian matrices A, i.e., Hermitian matrices all of whose eigenvalues have magnitudes less than 1. The first algorithm, which combines the construction of Ref. [31] with a Hadamard test [33], has a run time of $\tilde{O}(D\sqrt{t}\|v\|\|u\|\varepsilon^{-1}\|A\|_1^t)$[1]. For matrices where it is known that $\|A\|_1 \leqslant 1$, this provides a quantum speedup over repeated matrix multiplication, since its run time does not scale polynomially with the size of the matrix. Furthermore, it provides a quadratic speedup in t over the classical Monte Carlo algorithm for symmetric stochastic matrices (in which case $\|A\|_1 = 1$). For problems such as the simulation of diffusive discrete-time Markov chains, where $\|A^t u\|^{-1} = O(\sqrt{N})$ at large t, this algorithm provides an exponential speedup in the size of the matrix A over Ref. [31]. The second algorithm, based on Fourier series expansion, has a run time of $\tilde{O}[Dt^2\text{poly}(\|v\|\|u\|\varepsilon^{-1})]$ to compute $v^\dagger A^\tau u$ for all $\tau \in \{0, 1, 2, \dots, t\}$. While this is slower than the quantum-walk-based algorithm, it only uses Hamiltonian simulation as a primitive and thus is more suitable for near-term quantum hardware. A similar run time is achieved in Ref. [34] employing only Hamiltonian simulation, wherein the authors instead of using a Fourier series identified a matrix power with a derivative of the matrix exponential and used a finite difference approximation for it. Furthermore, our algorithm achieves a run time comparable to that of the quantum algorithms based on linear equation solvers [18,19]. For matrices that are not stochastic and consequently cannot be classically powered with the Monte Carlo algorithm, this algorithm achieves a quantum speedup over repeated matrix multiplication since its run time does not scale polynomially with the size of the matrix. Finally, following a construction similar to that of Ref. [35], we provide no-go theorems that limit the speedups achievable with a quantum computer for powering non-Hermitian matrices.

II. PROBLEM DEFINITION, PRELIMNARIES, AND SUMMARY OF RESULTS

We consider the problem of powering a Hermitian matrix $A \in \mathbb{C}^{N\times N}$ that is stable; i.e., all of its eigenvalues have a magnitude less than 1, or equivalently $\|A\|_2 \leqslant 1$. Furthermore, we assume the matrix to be D-sparse; i.e., every row or column of the stochastic matrix has at most D nonzero elements. Hermitian matrices arising in practice will typically have $D = O(1)$ or $O[\text{polylog}(N)]$. The matrix-powering problem that we consider is precisely defined below.

Problem (Matrix powering). Given a D-sparse stable Hermitian matrix $A \in \mathbb{C}^{N\times N}$, a positive integer power t, and vectors $v, u \in \mathbb{C}^N$, compute $v^\dagger A^t u$ to a specified precision of $\varepsilon > 0$.

We point out that previous works that solve the matrix-powering problem in various contexts adopt a different problem definition wherein they aim to prepare a quantum state encoding $A^t u$. Since in many applications of matrix powering we are finally interested in computing its inner product, $v^\dagger A^t u$, with another vector v which is typically known beforehand, the algorithms proposed in this paper directly compute this expectation value without ever explicitly preparing a quantum state encoding $A^t u$. We make two further notes about this problem definition.

(i) The precision of the output of this algorithm is assumed to be in a probabilistic sense; i.e., the algorithm is said to produce an estimate X of a quantity x with a precision ε if $\text{Prob}[|X - x| \leqslant \varepsilon]$ is large enough. The value of this probability, often referred to as the *confidence level* of the algorithm, is assumed to be a prespecified constant close to 1 throughout this paper and we suppress it in the complexity results.

(ii) We assume a black-box query model for the sparse Hermitian matrix A [36]; i.e., we assume access to two oracles O_F and O_A which allow us to access the elements of the Hermitian matrix. The oracle O_F provides access to the indices of the nonzero elements of each column of the Hermitian matrix via the implementation of a unitary that satisfies

$$O_F|j, k\rangle = |j, f(j, k)\rangle\ \forall j \in [N], k \in [D], \tag{1}$$

where $f(j, k)$ is the index of the kth nonzero element in the jth row or column. The oracle O_A provides access to the nonzero elements of the matrix A via the implementation of a unitary that satisfies

$$O_A|j, k\rangle|z\rangle = |j, k\rangle|z \oplus A_{j,k}\rangle\ \forall j, k \in [N], \tag{2}$$

where $A_{j,k}$ are the complex elements of the matrix A that are represented by a bit-string up to some specified precision δ. On a quantum computer, these oracles can be implemented with quantum circuits of depth $O[D\text{polylog}(1/\delta)]$, e.g., with a quantum random access memory (qRAM) [37]. On near-term hardware, there might be alternative more efficient ways of implementing these oracles for specific matrices A (for instance, the matrices corresponding to local Hamiltonians of a

[1]*Notation for norms.* Throughout this paper, for a vector $v \in \mathbb{C}^N$, $\|v\|_k$, $k \in \{1, 2, \dots\}$, will refer to the standard ℓ^k norm of the vector. Furthermore, for convenience, we will use $\|v\|$ to denote the ℓ^2 norm of v. For matrices $A \in \mathbb{C}^{N\times N}$, $\|A\|_k$ denotes the operator norm induced by the ℓ^k vector norm, i.e., $\|A\|_k = \sup_v \|Av\|_k/\|v\|_k$. In particular, $\|A\|_2$ will be the largest singular value of A, which coincides with the largest magnitude of eigenvalue if A is Hermitian. Additionally, for Hermitian matrices A, $\|A\|_\infty = \|A\|_1$ will be the maximum absolute row (or column) sum of the matrix A.

lattice of classical spins). In this paper, for clarity, we express our complexity results in terms of the number of calls to the oracles O_F and O_A, and these can easily be translated to the circuit depths for various hardware-specific implementations.

In the remainder of this paper, we provide several quantum algorithms to solve the matrix-powering problem and achieve speedups over classical algorithms. The first algorithm builds on Ref. [31] and combines a quantum walk together with a Hadamard test and a classical sampling algorithm to obtain the following result. Furthermore, by employing the linear combination of unitaries (LCU) technique along with quantum amplitude amplification [24], we can obtain a quadratic improvement in the scaling of the run time with the precision ε.

Theorem 1. Given a constant $C > 0$ such that $\|A\|_1 < C$, the matrix-powering problem can be solved with a quantum algorithm in $\tilde{O}(C^t D\sqrt{t}\|v\|\|u\|\varepsilon^{-1})$ calls to the oracles O_F and O_A.

Here $\tilde{O}$ hides any polylog complexity factors. We point out that these algorithms suffer from an exponential scaling with the power t when $\|A\|_1 > 1$—this is due to the fact that the quantum walk construction we employ can only be used if the sum of magnitude of the elements of each row (or column) of A is smaller than 1. For a number of matrix-powering problems, such as simulation of discrete-time Markov chains, $\|A\|_1 = 1$, and the run time of these scales sublinearly with t. The above results improve the fast-forwarding algorithm presented in Refs. [31], whose run time scales inversely with $\|A^t u\|$—our approach avoids this scaling at the expense of scaling with $\|u\|^2$ and $\|v\|^2$. This could be of relevance in problems such as the simulation of diffusive discrete-time Markov chains, where $\|A^t u\|^{-1} = O(\sqrt{N})$ at large t. Furthermore, compared to the quantum algorithms based on linear equation solvers, this result has a quadratic speedup in t. We also remark that when compared to classical algorithms, we obtain an exponential speedup in N over the matrix multiplication algorithm and a quadratic speedup in t over the Monte Carlo algorithm when the matrix A is stochastic.

While the quantum-walk-based algorithms provided above are able to achieve "fast-forwarding," i.e., a sublinear run time with respect to the matrix power t, they are difficult to implement on near-term quantum hardware. Given experimental constraints, it is widely believed that Hamiltonian simulation [4–6] will be one of the first problems to be solved on practical hardware. Furthermore, simulation of several classes of Hamiltonians can also be implemented on analog quantum simulators [38–40] which are significantly easier to experimentally build as compared to fully programmable quantum computers. Based on a truncated Fourier series expansion of the function $f(x) = x^t$, we provide a quantum algorithm to solve the matrix-powering problem with only the ability to use Hamiltonian simulation.

Theorem 2. The matrix-powering problem can be solved simultaneously for all powers from 0 to t using an efficient Hamiltonian simulator in time $\tilde{O}[t^2\text{poly}(\|v\|\|u\|\varepsilon^{-1})D]$.

We note that this result has a worse run time not only when compared to the quantum-walk algorithms but also with the classical Monte Carlo algorithm if the matrix A is stochastic. However, it achieves an exponential speedup in N over the classical repeated matrix multiplication algorithm although at an expense of quadratically worse scaling with t, for matrices that are not stochastic. Furthermore, it achieves the same run time as quantum algorithms based on the linear equation solvers if they are employed to compute matrix powers from 0 to t. The key advantage of this algorithm over other quantum algorithms is its feasibility of being implemented on near-term quantum hardware.

Finally, all the algorithms provided above assume the matrix A to be Hermitian, in which case it was possible to obtain a fast-forwarding speedup using quantum walks, i.e., compute A^t in time $\Theta(\sqrt{t})$. A natural question to ask is if fast-forwarding is possible for non-Hermitian matrices as well. By utilizing a construction similar to the no-go theorems for Hamiltonian simulation [35] and relying on the result that even a quantum computer cannot speedup the calculation of parity of N-bits [41,42], we provide the following no-go theorem.

Theorem 3 (No-go theorem). There cannot exist a quantum algorithm that solves the matrix-powering problem in $\tilde{O}[t^\alpha\text{poly}(\|u\|, \|v\|, \varepsilon^{-1})]$ calls to the oracles O_F and O_A, with $\alpha < 1$, for any arbitrary irreducible sparse matrix A.

We point out that while these no-go theorems rigorously show that it is not possible to fast-forward the matrix-powering problem for generic non-Hermitian matrices, it does not prohibit an improvement of the run time's dependence on the size of the matrix. Indeed, matrix-powering methods based on quantum linear equation solvers [18,19] obtain an exponential improvement over classical algorithms even for non-Hermitian matrices if the matrix is not stochastic.

The remainder of this paper contains proofs of the theorems stated above. In Sec. III we describe the matrix-powering algorithms presented in this paper and prove Theorems 6, 1, and 2. In Sec. IV, we prove the no-go Theorem 3. We only provide proofs of the most important theorems in the main text, and details are relegated to the appendices.

III. MATRIX MULTIPLICATION ALGORITHM

Before detailing the matrix-powering algorithm, we provide the following lemma that maps the computation of $v^\dagger A^t u$ to the overlap of A^t, $\langle\psi|A^t|\psi\rangle$, with quantum states $|\psi\rangle$ that depend on u and v. This transformation is useful since the Hadamard test naturally allows for the computation of such overlaps.

Lemma 1. Given a Hermitian matrix A and vectors v and u, it follows that

$$\text{Re}[v^\dagger A^t u] = \tfrac{1}{2}\big(\lambda_1^R\langle\psi_1^R|A^t|\psi_1^R\rangle + \lambda_2^R\langle\psi_2^R|A^t|\psi_2^R\rangle\big),$$
$$\text{Im}[v^\dagger A^t u] = \tfrac{1}{2}\big(\lambda_1^I\langle\psi_1^I|A^t|\psi_1^I\rangle + \lambda_2^I\langle\psi_2^I|A^t|\psi_2^I\rangle\big),$$

where $|\psi_i^R\rangle$ and λ_i^R, $i \in \{1, 2\}$, are the eigenvectors and eigenvalues of the Hermitian matrix $uv^\dagger + vu^\dagger$, and $|\psi_i^I\rangle$ and λ_i^I, $i \in \{1, 2\}$, are the eigenvectors and eigenvalues of the Hermitian matrix $i(vu^\dagger - uv^\dagger)$.

Proof. It follows immediately from the Hermiticity of A that $2\text{Re}[v^\dagger A^t u] = \text{Tr}[A^t(uv^\dagger + vu^\dagger)]$ and $2\text{Im}[v^\dagger A^t u] = \text{Tr}[iA^t(vu^\dagger - uv^\dagger)]$. Using $uv^\dagger + vu^\dagger = \lambda_1^R|\psi_1^R\rangle\langle\psi_1^R| + \lambda_2^R|\psi_2^R\rangle\langle\psi_2^R|$ and $i(vu^\dagger - uv^\dagger) = \lambda_1^I|\psi_1^I\rangle\langle\psi_1^I| + \lambda_2^I|\psi_2^I\rangle\langle\psi_2^I|$, we obtain the result in the lemma. ■

Consequently, we focus on developing methods to compute the overlap $\langle\psi|A^t|\psi\rangle$ efficiently. It can also be noted that for problems where u and v are sparse, the eigenvectors $|\psi_{1,2}\rangle$ introduced above are also sparse and consequently efficiently preparable on quantum computers. Furthermore, we note that the eigenvalues λ_1 and λ_2 are bounded by the norms u and v, which is concretely stated in the following lemma.

Lemma 2. All eigenvalues λ of $uv^\dagger + vu^\dagger$ and $i(vu^\dagger - uv^\dagger)$ satisfy $|\lambda| \leqslant 2\|u\|\|v\|$.

Proof. Denoting by $|\psi\rangle$ the normalized eigenvector corresponding to the eigenvalue λ, it follows that $|\lambda| = \|(uv^\dagger + vu^\dagger)|\psi\rangle\| \leqslant \|u\|\ |v^\dagger|\psi\rangle| + \|v\|\ |u^\dagger|\psi\rangle| \leqslant 2\|u\|\|v\|$. A similar proof holds for the eigenvalues of $i(vu^\dagger - uv^\dagger)$. ■

A. Fast-forwarding with quantum walks

One of the key ingredients in the quantum-walk-based algorithms for the matrix-powering problem is expressing A^t as a linear combination of Chebyshev polynomials of A,

$$A^t = \sum_{m=0}^{t} p_m T_m(A), \tag{3}$$

where p_m is a probability distribution given by

$$p_m = \begin{cases} \frac{1}{2^{t-1}}\binom{t}{(t-m)/2} & \text{for } m > 0,\ t = m \bmod 2, \\ \frac{1}{2^t}\binom{t}{t/2} & \text{for } m = 0,\ t = 0 \bmod 2, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Consequently, a quantum circuit to compute the overlap $\langle\psi|A^t|\psi\rangle$ can be constructed from a quantum circuit that can compute the overlap $\langle\psi|T_m(A)|\psi\rangle$ for a specified $m \in \{0, 1, \ldots, t\}$. As is shown below, this can be done with a quantum walk provided that the 1-norm of A is smaller than 1. Since this is not necessary for stable Hermitian matrices, we assume that we have access to an upper bound C on this norm, i.e., $\|A\|_1 \leqslant C$ and compute $\langle\psi|(A/C)^t|\psi\rangle$, albeit to a precision C^t higher than that required in $\langle\psi|A^t|\psi\rangle$. Therefore, in the remainder of this section, unless otherwise mentioned, we assume $\|A\|_1 \leqslant 1$.

A quantum walk construction similar to that used in Refs. [27,28,31] together with a Hadamard test allows us to compute these overlaps. However, since the elements of the matrix A can be complex, it is important to design the quantum walk with care so as to account for the phase of the complex matrix elements [36]. For $A \in \mathbb{C}^{N\times N}$, we consider a Hilbert space $\mathbb{C}^{N+1} \otimes \mathbb{C}^{N+1} \otimes \mathbb{C}^2$ and assume access to a unitary V that satisfies

$$V|i,0,0\rangle = \sum_{k=1}^{N} \sqrt{|A_{k,i}|}e^{i\varphi_{k,i}/2}|i,k,1\rangle + \left(1 - \sum_{k=1}^{N}|A_{k,i}|\right)^{1/2}|i,N+1,1\rangle \text{ if } i \neq N+1, \tag{5a}$$

$$V^\dagger|i,j,1\rangle = \sqrt{|A_{j,i}|}e^{-i\varphi_{j,i}/2}|i,0,0\rangle + |\phi^\perp\rangle|1\rangle \text{ for some } |\phi^\perp\rangle \text{ if } i \neq N+1, \tag{5b}$$

$$V|N+1,j,b\rangle = |N+1,j,b\rangle, \tag{5c}$$

where if $A_{i,j} = |A_{i,j}|e^{i\angle A_{i,j}}$ for $\angle A_{i,j} \in (-\pi,\pi]$, then $\varphi_{i,j} = \angle A_{i,j}$ for $i \geqslant j$ and $-\angle A_{i,j}$ for $i < j$. Furthermore, we introduce the operator S given by

$$S|i,j,b\rangle = \begin{cases} |i,j,0\rangle \text{ if } b = 0, \\ |j,i,1\rangle \text{ if } b = 1 \text{ and } i \neq j, \\ \text{sgn}(A_{i,i})|i,i,1\rangle \text{ if } b = 1 \text{ and } i = j. \end{cases} \tag{6}$$

We remark that this operator is different from that used in Ref. [31]—in particular, we have modified this operator to account for possibly negative on-diagonal elements of the matrix A which Apers and Sarlette [31] did not handle since they were dealing with a stochastic matrix. Finally, the quantum walk operator W can then be constructed using the operators V and S and a reflection about the last qubit,

$$W = -(I \otimes I \otimes \sigma_z)V^\dagger S V. \tag{7}$$

We then obtain the following lemma, similar to that obtained in Refs. [31,32] stating that m applications of the quantum walk operator effectively apply $T_m(A)$ on an input state conditioned on the state of the last qubit.

Lemma 3 [Quantum walk for $T_m(A)$]. The unitary operator W defined in Eq. (7) satisfies

$$W^m|\psi\rangle|0\rangle|0\rangle = T_m(A)|\psi\rangle|0\rangle|0\rangle + |\psi^\perp\rangle|1\rangle, \tag{8}$$

for some $|\psi^\perp\rangle \in \mathbb{C}^{N+1} \otimes \mathbb{C}^{N+1}$.

Proof. Denote by Π_0 the projector that projects the third register to the state $|0\rangle$. Then we have $2\Pi_0 - I = -\sigma_z$. Using the fact that $\Pi_0\sigma_z = -\Pi_0$, and that U is unitary, we can write

$$\begin{aligned} \Pi_0 W^m &= -\Pi_0\sigma_z U(2\Pi_0 - I)UW^{m-2} \\ &= 2\Pi_0 U\Pi_0 UW^{m-2} - \Pi_0 U^2 W^{m-2} \\ &= 2\Pi_0 U\Pi_0 W^{m-1} - \Pi_0 W^{m-2}. \end{aligned} \tag{9}$$

Recall that the Chebyshev polynomials satisfy the recursion $T_m(x) = 2xT_{m-1}(x) - T_{m-2}(x)$. Using $x = \Pi_0 U$ and $T_0(\Pi_0 U) = \Pi_0$, it follows that $\Pi_0 W^m$ fulfills the same recursion. We can therefore write $\Pi_0 W^m = T_m(\Pi_0 U)$. Furthermore, since $(\Pi_0 U)^m|\psi\rangle|0,0\rangle = M^m|\psi\rangle|0,0\rangle$ (this is shown in Ref. [31]), we obtain

$$\Pi_0 W^m|\psi\rangle|0,0\rangle = T_m(M)|\psi\rangle|0,0\rangle. \tag{10}$$

Since W is a unitary operator, this implies that

$$W^m|\psi\rangle|0,0\rangle = T_m(M)|\psi\rangle|0,0\rangle + |\psi^\perp\rangle|1\rangle, \tag{11}$$

which proves the lemma. ■

As is shown in Appendix A, the quantum walk operator A can be implemented with $O(D)$ calls to the oracles O_F and O_A that access the matrix A. In order to estimate the overlap $\langle\psi|T_m(A)|\psi\rangle$ using the walk operator W, we introduce a controlled version of W and W^c via

$$W^c = I \otimes |0\rangle\langle 0| + W \otimes |1\rangle\langle 1|. \quad (12)$$

We then have the following lemma to compute the overlap $\langle\psi|T_m(A)|\psi\rangle$ using a Hadamard test with the controlled operator W^c.

Lemma 4 (Chebyshev polynomial overlap). Consider the state $(W^c)^m|\psi\rangle|0,0,+\rangle$, and measure the last two qubits on the basis $\{|0,+\rangle, |0,-\rangle, |1,+\rangle, |1,-\rangle\}$. Define a random variable X_m based on the measurement outcome μ via

$$X_m = \begin{cases} +1 & \text{if } \mu = (0,+), \\ -1 & \text{if } \mu = (0,-), \\ 0 & \text{otherwise,} \end{cases}$$

then $E(X_m) = \langle\psi|T_m(A)|\psi\rangle$.

Proof. It is straightforward to evaluate the expectation of X_m,

$$\begin{aligned} E(X_m) &= \sum_{m=0}^{t} p_m(p_{0+} - p_{0-}) \\ &= \sum_{m=0}^{t} p_m\langle\psi|T_m(A)|\psi\rangle \\ &= \langle\psi|T_m(A)|\psi\rangle, \end{aligned} \quad (13)$$

which completes the proof. ■

While this overlap estimation procedure can be used together with the Chebyshev polynomial expansion in Eq. (3) to compute $\langle\psi|A^t|\psi\rangle$, this would not provide any fast-forwarding since computing $T_t(A)$ would require t applications of the operator W^c. However, an important insight into the nature of the coefficients p_m in the expansion in Eq. (4) is that they concentrate around $m \sim \sqrt{t}$. One possible approach to exploit this property is to sample m from the probability distribution given by the coefficients p_m in Eq. (4) and compute the overlap $\langle\psi|T_m(A)|\psi\rangle$ of the corresponding Chebyshev polynomial using Lemma 4—this would allow us to reduce, on an average, the number of times the walk operator W^c is applied. This is formalized in the lemma below.

Lemma 5 [Matrix power overlap with classical sampling]. The overlap $\langle\psi|A^t|\psi\rangle$ can be computed by estimating the mean of a random variable X which is generated by first sampling $m \in \{0, 1, \dots, t\}$ from the probability distribution in Eq. (4), followed by drawing a sample of X_m defined in Lemma 4 using the state $|\psi\rangle$. Furthermore, this estimation can be done in expectation with a precision ε with $O(\varepsilon^{-2}\sqrt{t})$ calls to the controlled walk operator W^c or equivalently with $O(D\varepsilon^{-2}\sqrt{t})$ calls to the oracles O_F and O_A.

Proof. We can immediately see that

$$\begin{aligned} E(X) &= \sum_{m=0}^{t} p_m E(X_m) \\ &= \sum_{m=0}^{t} p_m\langle\psi|T_m(M)|\psi\rangle = \langle\psi|M^t|\psi\rangle. \end{aligned} \quad (14)$$

Furthermore, noting that $X^2 \in \{0, 1\}$, it follows that $\mathrm{var}(X) \leqslant 1$. Consequently, $E(X)$ can be estimated to a precision ε with $N = O(1/\varepsilon^2)$ samples. Furthermore, the average number of calls to the controlled walk operator W^c, $\langle m\rangle$, is given by

$$\langle m\rangle = \sum_{m=0}^{t} m p_m = \frac{1}{2^t}\sum_{n=0}^{t/2} n\binom{t}{t/2-n} = \frac{2+t}{2^{t-2}}\binom{t}{t/2-1}. \quad (15)$$

For large t, using Stirling's approximation this is

$$\binom{t}{t/2-1} = \frac{t/2}{t/2+1}\binom{t}{t/2} \sim \binom{t}{t/2} \sim \frac{2^t}{\sqrt{t\pi/2}}. \quad (16)$$

Therefore $\langle m\rangle = O(\sqrt{t})$, and consequently the number of calls to the W^c operator to achieve a precision ε is given by $N\langle m\rangle = O(\varepsilon^{-2}\sqrt{t})$. ■

Lemma 6. Given a constant $C > 0$ such that $\|A\|_1 < C$, the matrix-powering problem can be solved with a quantum algorithm in $O(C^{2t}D\sqrt{t}\|v\|^2\|u\|^2\varepsilon^{-2})$ calls, in expectation, to the oracles O_F and O_A.

Proof. Combining Lemma 5 with Lemma 1, we obtain a procedure for solving the matrix-powering problem. The complexity result in Theorem 6 can be obtained as follows: Given an upper bound C on $\|A\|_1$, we note that to compute $v^\dagger A^t u$ to a precision ε, we need to compute $\langle\psi_{1,2}^{L,R}|(A/C)^t|\psi_{1,2}^{L,R}\rangle$ to a precision of at most $\varepsilon/4\|u\|\|v\|C^t$ which can be done using the algorithm in Lemma 1 with $O(D\varepsilon^{-2}\sqrt{t}\|v\|^2\|u\|^2C^{2t})$ calls to the oracles O_F and O_A. ■

While the algorithm described above allows a quadratic fast-forwarding for the matrix-powering problem, the dependence of the run time on the precision ε can also be improved by using amplitude amplification, which is precisely stated in the following lemma from Ref. [24].

Lemma 7 (Overlap estimation, Theorem 2.5 of Ref. [24]). Given a state $|\psi\rangle$ in terms of its preparation unitary U from a known state $|0\rangle$: $|\psi\rangle = U|0\rangle$, an observable V and an estimate σ of its variance satisfying $\langle\psi|V^2|\psi\rangle - (\langle\psi|V|\psi\rangle)^2 \leqslant \sigma^2$, $\langle\psi|V|\psi\rangle$ can be estimated on a quantum computer with a precision ε with $\tilde{O}(\varepsilon^{-1}\sigma)$ calls to the unitary U.

In order to use amplitude amplification and achieve fast-forwarding in the same algorithm, we approximate the problem of computing the overlap $\langle\psi|A^t|\psi\rangle$ for a given $|\psi\rangle$ to computing an overlap of the form $\langle\phi_t|V|\phi_t\rangle$ where the operator V is independent of t and the state $|\phi_t\rangle$ can be prepared in $\Theta(\sqrt{t})$ calls to the oracles O_F and O_A. This is achieved by using a combination of quantum walks with the Hadamard test and the LCU technique similar to that of Ref. [31]. The quadratic fast-forwarding in this approach is also obtained due to the concentration of the coefficients p_m in Eq. (4) around $m = \sqrt{t}$. This is made concrete by the following lemma, according to which the sum in Eq. (3) can be truncated after $\sim O[\sqrt{t}\ln(\varepsilon^{-1})]$ terms while incurring a specified additive error ε.

Lemma 8. If A is a stable Hermitian matrix and $|\psi\rangle$ is a normalized state, then $\forall\varepsilon > 0$ and $C \geqslant 2\ln(2/\varepsilon)$

$$\left|\langle\psi|A^t|\psi\rangle - \langle\psi|\sum_{m=0}^{\sqrt{Ct}} p_m T_m(A)|\psi\rangle\right| \leqslant \varepsilon.$$

Proof. Using Lemma 3 from Ref. [31], we obtain that $\forall\varepsilon > 0, C \geqslant 2\ln(2/\varepsilon)$

$$\left| x^t - \sum_{m=0}^{\sqrt{Ct}} p_m T_m(x) \right| \leqslant \varepsilon \; \forall x \in [-1, 1]. \tag{17}$$

Since A is a stable matrix, its eigenvalues will have a magnitude less than equal to 1. Furthermore, since A is also Hermitian, its eigenvalues are real and lie in $[-1, 1]$ and hence satisfy Eq. (17). Denoting by $\lambda_i, |\phi_i\rangle$ the eigenvalues and eigenvectors of M and using its spectral decomposition $A = \sum_i \lambda_i |\phi_i\rangle\langle\phi_i|$, we obtain

$$\left| \langle\psi|A^t|\psi\rangle - \langle\psi| \sum_{m=0}^{\sqrt{Ct}} p_m T_m(A)|\psi\rangle \right|$$
$$\leqslant \sum_k |\langle\phi_k|\psi\rangle|^2 \left| \lambda_k^t - \sum_{m=0}^{\sqrt{Ct}} p_m T_m(\lambda_k) \right| \leqslant \varepsilon. \tag{18}$$

■

Consequently, we can effectively approximate A^t as a weighted linear combination of $O(\sqrt{t})$ Chebyshev polynomials of A—while the quantum walk operator introduced in Eq. (7) can be used to individually implement the Chebyshev polynomials, in order to implement their linear combination we use the LCU technique [6]. Below, we show the construction of an operator to effectively apply $\sum_{m=0}^{\tau} p_m W^m$ to a given quantum state. We do this by introducing auxillary qubits and implementing the following unitary V_P depending on the coefficients p_m:

$$V_P|\phi\rangle|0, 0\rangle = \sum_{m=0}^{\tau} \sqrt{p_m}|\phi\rangle|m, 0\rangle$$
$$+ \left[1 - \sum_{m=0}^{\tau} p_m \right]^{1/2} |\phi\rangle|0, 1\rangle. \tag{19}$$

Furthermore, we assume access to the controlled quantum walk operator W_τ defined by

$$W_\tau = \sum_{m=0}^{\tau} W^m \otimes |m\rangle\langle m| \otimes I. \tag{20}$$

The operator W_τ requires τ calls to the quantum walk operator W. Therefore, following the result in Appendix A, it can be constructed with $O(D\tau)$ calls to the oracles O_F and O_A. The operator $V_P^\dagger W_\tau V_P$ then effectively applies the linear combination $\sum_{m=0}^{\tau} p_\tau W^\tau$ to an input state.

Lemma 9 (LCU adapted from Refs. [6,31]). The unitary operator $V_P^\dagger W_\tau V_P$, with V_P and W_τ defined in Eqs. (19) and (20), respectively, satisfies

$$V_P^\dagger W_\tau V_P|\phi\rangle|0, 0\rangle = \sum_{m=0}^{\tau} p_m W^m|\phi\rangle|0, 0\rangle + |\phi^\perp\rangle|1\rangle, \tag{21}$$

for some $|\phi_\perp\rangle$.

In order to compute $\langle\psi| \sum_{m=0}^{\tau} p_m T_m(A)|\psi\rangle$, we consider a controlled version of the operator defined in Lemma 9: $W_\tau^c = I \otimes |0\rangle\langle 0| + V_P^\dagger W_\tau V_P \otimes |1\rangle\langle 1|$. It then follows from Lemmas 3 and 9 that computing the expectation value of the operator $|0\rangle\langle 0| \otimes \sigma_z$ on the last two qubits in the circuit of W_τ^c on a state prepared by application of $HW_\tau^c H$ (where H is a Hadamard gate on the last qubit) to $|\psi\rangle|0, 0, \ldots, 0\rangle$ allows us to evaluate $\langle\psi| \sum_{m=0}^{\tau} p_m T_m(A)|\psi\rangle$. By truncating the linear combination to an appropriate number of terms using Lemma 7 and using amplitude estimation (Lemma 8), we obtain the following lemma for estimating the overlap of the matrix with a given state.

Lemma 10 (Matrix power overlap with LCU). For a Hermitian stable matrix A with $\|A\|_1 \leqslant 1$ and a state $|\psi\rangle$, $\langle\psi|A^t|\psi\rangle$ can be computed on a quantum computer to a precision ε with $\tilde{O}(D\sqrt{t}\varepsilon^{-1})$ calls to the oracles O_F and O_A.

Proof. The overlap estimation can be done using the Hadamard test described above—as shown in Lemma 8, it is sufficient to use $\tau = O[\sqrt{t}\ln(\varepsilon^{-1})]$ in the LCU construction described in Lemma 9. Furthermore, the outcome of the Hadamard test has a variance bounded by 1 and consequently a direct application of amplitude amplification (Lemma 7) allows us to obtain an estimate of $\langle\psi|A^t|\psi\rangle$ with $\tilde{O}(D\sqrt{t}/\varepsilon)$ calls to the oracles O_F and O_A. ■

Proof of Theorem 2. Combining Lemma 10 with Lemma 1, we obtain a procedure for solving the matrix-powering problem. The complexity result in Theorem 1 can be obtained as follows. Given an upper bound C on $\|A\|_1$, we note that to compute $v^\dagger A^t u$ to a precision ε, we need to compute $\langle\psi_{1,2}^{L,R}|(A/C)^t|\psi_{1,2}^{L,R}\rangle$ to a precision of at most $\varepsilon/4\|u\|\|v\|C^t$ which can be done using the algorithm in Lemma 1 with $\tilde{O}(\sqrt{t}\|v\|\|u\|C^t\varepsilon^{-1})$ calls to the oracles O_F and O_A. ■

B. Matrix powering with Hamiltonian simulation

In this section, we describe an approach to solve the matrix-powering problem using Hamiltonian simulation as a primitive and prove the complexity result in Theorem 2. Formally for our purposes, a Hamiltonian simulation can be considered to be the problem of computing $\langle\psi|e^{-iHt}|\psi\rangle$ to a precision ε given access to the sparse Hamiltonian H and the states $|\psi\rangle$. A Hamiltonian simulator (implemented on a quantum computer or an analog quantum simulator) is said to be efficient if it can solve this problem in $\tilde{O}[\text{poly}(\varepsilon^{-1})D\|H\|_{\max}t]$ time, where D is the sparsity of the Hamiltonian and $\|H\|_{\max}$ is its maximum magnitude element. State of the art algorithms for Hamiltonian simulation on quantum computers achieve such run times for general sparse Hamiltonians [5,6], while such run times can be achieved on quantum simulators for local Hamiltonians.

We again restrict ourselves to stable Hermitian matrices A. In order to compute an overlap $\langle\psi|A^t|\psi\rangle$, we expand A^t into a Fourier series—as is shown in the following two lemmas, this can be done to a precision of ε while retaining only $N_h = O(t/\varepsilon)$ harmonics.

Lemma 11. $\forall\varepsilon \in (0, 2/\pi)$, $N_h \geqslant 4t/\pi^2\varepsilon$, and $\exists a \in \mathbb{C}^{2N_h+1}$ such that

$$\left| x^t - \sum_{n=-N_h}^{N_h} a_n e^{in\pi x/2} \right| \leqslant \varepsilon \; \forall\, x \in [-1, 1], \tag{22}$$

where a_n are the elements of the vector a and a can be computed classically in $O(N_h t)$ time. Furthermore, $\|a\|_1 \leqslant 1 \; \forall t > 0$ and $N_h > 0$.

A detailed proof of this lemma, as well as an explicit calculation of the coefficient vector a, is provided in Appendix B. Employing this result, we can now compute the overlap of the power $\langle\psi|A^t|\psi\rangle$ using an efficient Hamiltonian simulator.

Lemma 12. (Matrix overlap with Hamiltonian simulation). Given a D-sparse stable Hermitian matrix A, $\langle\psi|A^\tau|\psi\rangle$ can be computed for all $\tau \in \{0, 1, \dots, t\}$ using an efficient Hamiltonian simulator in time $\tilde{O}[\text{poly}(\varepsilon^{-1})Dt^2]$.

Proof. Since the matrix A is stable and Hermitian, all of its eigenvalues are real and lie in $[-1, 1]$, thus satisfying Eq. (22). Denoting by $\lambda_k, |\phi_k\rangle$ the eigenvalues and eigenvectors of A and using Lemma 11, we obtain that

$$\left|\langle\psi|A^\tau|\psi\rangle - \sum_{n=-N_h}^{n=N_h} a_n\langle\psi|e^{i\pi nA/2}|\psi\rangle\right| = \left|\sum_k |\langle\phi_k|\psi\rangle|^2\left(\lambda_k^\tau - \sum_{n=-N_h}^{n=N_h} a_n e^{i\pi n\lambda_k/2}\right)\right| \leqslant \frac{\varepsilon}{2}, \quad (23)$$

for an appropriately chosen $N_h = O(\tau/\varepsilon)$. Furthermore, we note that since A is stable, the magnitude of all of its elements is at most 1, i.e., $\|A\|_{\max} \leqslant 1$. Using an efficient Hamiltonian simulator, we can estimate $\langle\psi|e^{i\pi nA/2}|\psi\rangle$ to a precision of $\varepsilon/2$ in time $O[\text{poly}(\varepsilon^{-1})Dn]$. Since $\|a\|_1 \leqslant 1$, we can then compute $\sum_{n=-N_h}^{N_h} a_n\langle\psi|e^{i\pi nA/2}|\psi\rangle$ to a precision ε on a classical computer, thus determining $\langle\psi|A^\tau|\psi\rangle$ to a precision ε—since we need to compute $\langle\psi|e^{i\pi nA/2}|\psi\rangle$ for $n \in \{-N_h, -N_h + 1, \dots, N_h - 1, N_h\}$, the total time taken for this computation is $O[\text{poly}(\varepsilon^{-1})DN_h^2] = O[\text{poly}(\varepsilon^{-1}D)\tau^2]$. Furthermore, we note that since we propose to compute the overlaps $\langle\psi|e^{in\pi A/2}|\psi\rangle$ individually for all n, we can compute $\langle\psi|A^\tau|\psi\rangle$ for all $\tau \in \{0, 1, \dots, t\}$ with the same set of Hamiltonian simulations in time $O[\text{poly}(\varepsilon^{-1})Dt^2]$. ■

Proof of Theorem 2. Combining Lemma 12 with Lemma 1, we obtain a procedure for solving the matrix-powering problem. To obtain the complexity result in Theorem 2, we note that computing $v^\dagger A^\tau u$ to a precision ε, we need to compute $\langle\psi_{1,2}^{L,R}|A^\tau|\psi_{1,2}^{L,R}\rangle$ to a precision of at most $\varepsilon/4\|u\|\|v\|$. This can be done using the algorithm in Lemma 12 for $\tau \in \{0, 1, 2, \dots, t\}$ simultaneously in $O[\text{poly}(\varepsilon^{-1}\|v\|\|u\|)Dt^2]$ time. ■

IV. NO-GO THEOREMS FOR FAST-FORWARDING

In this section, we provide the no-go theorems stated in Sec. II, showing that fast-forwarding the matrix-powering problem is not possible for a generic non-Hermitian matrix. These no-go theorems utilize a construction similar to that of the no-go theorems for Hamiltonian simulation [35], and they rely on the fact that even a quantum computer cannot speed up the calculation of parity of N-bits [41,42], a result concretely stated as the following lemma.

Lemma 13 (N-bit parity problem, Refs. [41,42]). Consider N-bits $b_1, b_2, \dots, b_N$ which can be accessed as an oracle O_B: $O_B|i, b\rangle = |i, b \oplus b_i\rangle$. There cannot exist a quantum algorithm that can determine the parity $b_1 \oplus b_2 \oplus \cdots \oplus b_N$ with success probability greater than $1/2$ with fewer than $N/2$ calls to the oracle O_B.

We note that this result rules out even an approximate solution of the N-bit parity problem on a quantum computer with a run time less than $O(N)$. In particular, since the parity is known to be an integer, if an algorithm can estimate this parity to a precision of $\varepsilon < 1/2$ with a confidence level greater than $1/2$, then it would have solved the N-bit parity problem—consequently, in the rest of this analysis we can consider the precision $\varepsilon = O(1)$. As is shown in the no-go theorem below, we can construct a matrix such that computing its Nth power, in the sense specified in Sec. II, allows us to solve the N-bit parity problem, thereby ruling out the possibility of designing a quantum algorithm that can achieve a run time scaling sublinearly with the matrix power. We first provide a proof of Lemma 14, which rules out the possibility of fast-forwarding the powering of a general matrix, and then strengthen it to obtain Theorem 3, which rules out the possibility of fast-forwarding even the powering of irreducible matrices.

Lemma 14 (No-go theorem for an arbitrary matrix). There cannot exist a quantum algorithm that solves the matrix-powering problem in $\tilde{O}[t^\alpha\text{poly}(\|u\|, \|v\|, \varepsilon^{-1})]$ calls to the oracles O_F and O_A, with $\alpha < 1$, for any sparse matrix A.

Proof. Given N bits $b_1, b_2, \dots, b_N$, we can construct the following matrix-powering problem that determines the parity of $b_1 \oplus b_2 \cdots \oplus b_N$ on computing its Nth power.

(i) The matrix $A \in \mathbb{C}^{2N\times 2N}$ with the rows (or columns) being indexed by (i, σ), where $i \in [N]$ and $\sigma \in \{0, 1\}$, and matrix elements being given by

$$A_{i,\sigma;i',\sigma'} = \begin{cases} 1 & \text{if } i' = i - 1 \text{ and } \sigma \oplus \sigma' = b_{i-1}, \\ 0 & \text{otherwise.} \end{cases} \quad (24)$$

(ii) The vectors u and v are given by

$$u_{i,\sigma} = \begin{cases} 1 & \text{if } i = 0 \text{ and } \sigma = 0, \\ 0 & \text{otherwise,} \end{cases}$$
$$v_{i,\sigma} = \begin{cases} 1 & \text{if } i = N \text{ and } \sigma = 0, \\ -1 & \text{if } i = N \text{ and } \sigma = 1, \\ 0 & \text{otherwise.} \end{cases} \quad (25)$$

The matrix A is a stochastic matrix corresponding to a discrete-time Markov chain (DTMC) shown in Fig. 1(a)—the states of this DTMC can be grouped as per their σ index, and for every bit b_i is identified with a flip of the "σ" index at i. Consequently, on computing $v^\dagger A^t u$, we can count the number of bits that are 1, thereby computing $b_1 \oplus b_2 \oplus \cdots \oplus b_N$. Furthermore, we can note that the oracles O_F and O_A can be constructed with $O(1)$ calls to the oracle O_B since each bit determines the positions of the nonzero elements in at most two columns. Furthermore, we note that $\|u\|, \|v\| = O(1)$ by construction. Consequently, if there existed a quantum algorithm to solve the DTMC simulation problem with $\tilde{O}[t^\alpha\text{poly}(\|u\|, \|v\|, 1/\varepsilon)]$ queries to O_F and O_A with $\alpha < 1$, it could solve the N-bit parity problem in $\tilde{O}(N^\alpha)$ queries to O_B. This contradicts Lemma 13, thus proving that no such quantum algorithm can exist. ■

While this argument proves that no quantum algorithm can exist to fast-forward the matrix-powering problem for a general matrix, we note that the specific matrix A used in this argument is reducible. Consequently, this raises the question of whether their exists a quantum algorithm that can fast-forward the powering of arbitrary irreducible matrices.

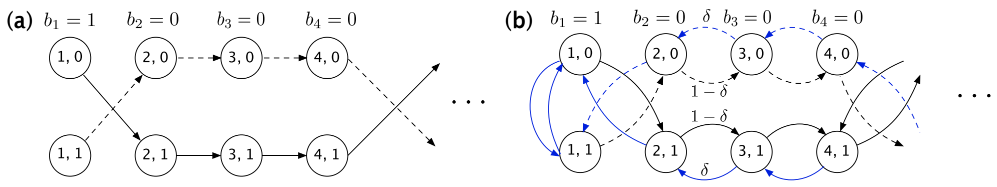

FIG. 1. Schematic representation of (a) a reducible Markov chain and (b) an irreducible Markov chain that solves the N-bit parity problem. In panel (b), the blue (gray) lines indicate the stochastic matrix element δ and the black lines indicate the stochastic matrix element $1 - \delta$.

We show that this too is not possible by constructing an irreducible stochastic matrix that is very close to the reducible stochastic matrix constructed above and thus approximately solves the N-bit parity problem.

Proof of Theorem 3. We consider an instance of the matrix-powering problem with u and v as defined in Eq. (25) and a matrix $A_\delta = A + B\delta$, where A is defined in Eq. (24), $\delta \in (0, 1)$, and matrix B has elements given by

$$B_{i,\sigma;i',\sigma'} = \begin{cases} -1 & \text{if } i' = i-1 \text{ and } \sigma \oplus \sigma' = b_{i-1}, \\ 1 & \text{if } i' = i+1 \text{ and } \sigma \oplus \sigma' = b_i, \\ 1 & \text{if } i = i' \in \{1, N\}, \sigma \neq \sigma', \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

The matrix A_δ is the stochastic matrix corresponding to the discrete-time Markov chain shown in Fig. 1(b). It is easy to see that A_δ is irreducible for $\delta \neq 0$. Furthermore, we can easily bound the difference between the result of powering A_δ and A:

$$\begin{aligned} \left|v^\dagger A_\delta^t u - v^\dagger A^t u\right| &\leqslant \|v\|\|u\|\left[(\|A\|_2 + \|B\|_2\delta)^t - \|A\|_2^t\right] \\ &\leqslant \|v\|\|u\|\|A\|_2^t(e^{\|B\|_2\delta t/\|A\|_2} - 1). \end{aligned} \tag{27}$$

For the choice of u and v under consideration, $\delta = \ln(1 + \varepsilon/2\sqrt{2})/8N^2$ and $t = N$, it follows that

$$\left|v^\dagger A_\delta^N u - v^\dagger A^N u\right| \leqslant \frac{\varepsilon}{2}, \tag{28}$$

wherein we have used $\|v\| = \sqrt{2}$ and $\|u\| = 1$, and $1/\sqrt{2N} \leqslant \|A\|_2 \leqslant 1$ and $\|B\|_2 \leqslant 4\sqrt{2N}$. This shows that being able to compute $v^\dagger A_\delta^t u$ to a precision of $\varepsilon/2$ allows us to determine $v^\dagger A^t u = b_1 \oplus b_2 \oplus \cdots \oplus b_N$ to precision ε. Consequently, from Lemma 14, the no-go theorem follows for irreducible matrices as well. ■

V. CONCLUSION

This paper studied the problem of computing the power of a stable Hermitian matrix. Following the construction of Ref. [31], we show that fast-forwarding is possible while powering stable Hermitian matrices, and we present algorithms based on quantum walks that improve their results. We also present a complementary algorithm to solve the matrix-powering problem using only Hamiltonian simulators which could potentially be used on near-term quantum hardware. Finally, by establishing a map between the the N-bit parity determination problem to a matrix-powering problem, we show that quantum computers cannot fast-forward powering of non-Hermitian matrices.

ACKNOWLEDGMENTS

We acknowledge support from the ERC Advanced Grant QUENOCOBA under the EU Horizon 2020 program (Grant No. 742102) and from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Project No. 414325145 in the framework of the Austrian Science Fund (FWF), Grant No. SFB F7104.

APPENDIX A: COSTS FOR QUANTUM WALK OPERATORS IN TERMS OF MATRIX ORACLES

In this Appendix we show how the coin operator W can be implemented with $O(D)$ calls to the oracles O_F and O_A that access the matrix A, where D is the sparsity. The coin operator is $W = V^\dagger SV$. We have assumed access to a unitary operator V such that

$$V|i,0,0\rangle = \sum_{k=1}^{N} \sqrt{|A_{k,i}|}e^{i\varphi_{k,i}/2}|i,k,1\rangle + \left(1 - \sum_{k=1}^{N}|A_{k,i}|\right)^{1/2}|i,N+1,1\rangle \text{ if } i \neq N+1, \tag{A1a}$$

$$V^\dagger|i,j,1\rangle = \sqrt{|A_{j,i}|}e^{-i\varphi_{j,i}/2}|i,0,0\rangle + \left|\phi^\perp\right\rangle|1\rangle \text{ for some } |\phi^\perp\rangle \text{ if } i \neq N+1, \tag{A1b}$$

$$V|N+1,j,b\rangle = |N+1,j,b\rangle. \tag{A1c}$$

We can write V as

$$V = |N+1\rangle\langle N+1| \otimes I \otimes I + \sum_{i=1}^{N}|i\rangle\langle i| \otimes V_i. \tag{A2}$$

Therefore, V applies V_i depending on the second and third registers depending on the state of the first qubit. The V_i act as

$$V_i = |\psi_i\rangle\langle 0,0| + \sum_{j=1}^{N}|j,0\rangle\langle j,0| + \sum_{j=1}^{N+1}|\psi_{i,j}^\perp\rangle\langle j,1|, \tag{A3}$$

where $|\psi_{i,j}^{\perp}\rangle$, along with $|\psi_i\rangle$, form an orthonormal basis. The expression for $|\psi_i\rangle$ is

$$\begin{aligned}|i\rangle|\psi_i\rangle &= V|i,0,0\rangle\\ &= \sum_{k=1}^{N}\sqrt{A_{k,i}^{*}}|i,k,1\rangle + \sqrt{1-\sum_{k=1}^{N}|A_{k,i}|}|i,N+1,1\rangle,\end{aligned} \tag{A4}$$

where the square root is defined as in Eq. (A1a). Therefore, we need to show that we can apply the operator U that prepares $|\psi_i\rangle$ with $O(D)$ queries to the oracles. This can be proved with a slight modification of the procedure in Ref. [43].

To do this, we start with the state $|i\rangle|0\rangle$. We then prepare a list with all the neighbors of i: $|i\rangle|f(i,1)\rangle\cdots|f(i,D)\rangle$. This can be done with $O(D)$ calls to O_F. Now we can prepare a list containing all the nonzero probabilities in the ith row $|i\rangle|A_{i,f(i,1)}\rangle\cdots|A_{i,f(i,D)}\rangle$. This can be done with $O(D)$ calls to O_A. Appending an extra register to this list, we can compute $|i\rangle|A_{i,f(i,1)}\rangle|A_{i,f(i,D)}\rangle|1-\sum_{k=1}^{D}|A_{i,f(i,k)}|\rangle$ Using both lists, following the procedure in Ref. [43], we can prepare the state

$$\sum_{j=1}^{D}\sqrt{A_{i,f(i,j)}^{*}}|i\rangle|j\rangle + \sqrt{1-\sum_{k=1}^{D}|A_{i,f(i,k)}|}|i,N+1\rangle. \tag{A5}$$

Querying again O_F we obtain the desired state. Therefore, it follows that the quantum walk operator W can be implemented with $O(D)$ calls to O_F and O_A.

APPENDIX B: PROOF OF LEMMA 11

In this Appendix, we provide a proof of Lemma 11 introduced and used in the main text. Our goal is to calculate a Fourier series expansion of x^t, which converges pointwise when $x\in[-1,1]$, and estimate the number of terms of the Fourier series that we need to retain to achieve a certain precision in this expansion. We point out that for odd t, a Fourier series expansion of x^t in the interval $[-1,1]$ could exhibit a Gibb's phenomena at $x=\pm 1$ since the periodic extension of x^t is not continuous. Consequently, we instead consider the function $f(x)$ on the interval $[-2,2]$ defined below:

$$f_t(x)=\begin{cases} x^t & \text{for } |x|\leqslant 1,\\ (2-x)^t & \text{for } 1<x\leqslant 2,\\ (-2-x)^t & \text{for } -2\leqslant x<-1.\end{cases} \tag{B1}$$

The periodic extension of this function is continuous, since $f_t(2)=f_t(-2)\ \forall t$ and $f_t(x)$ is continuous within the interval $(-2,2)$. Furthermore, for $|x|\leqslant 1$ this function coincides with x^t. We can now write down a Fourier series expansion for this function which converges pointwise for all $x\in[-2,2]$—for ease of analysis, we treat the cases when t is even and odd separately:

$$f_t(x)=\begin{cases}\sum_{p=0}^{\infty}c_p(t)\cos(p\pi x) & \text{if } t \text{ is even},\\ \sum_{p=0}^{\infty}s_p(t)\sin[(2p+1)\pi x/2] & \text{if } t \text{ is odd},\end{cases} \tag{B2}$$

where

$$\begin{aligned}c_p(t) &= \int_0^2 f_t(x)\cos(p\pi x)dx\\ &= 2\int_0^1 x^t\cos(p\pi x)dx,\end{aligned} \tag{B3a}$$

$$\begin{aligned}s_p(t) &= \int_0^2 f_t(x)\sin[(2p+1)\pi x/2]dx\\ &= 2\int_0^1 x^t\sin[(2p+1)\pi x/2]dx.\end{aligned} \tag{B3b}$$

We point out that Eq. (B2) can be rewritten in terms of complex exponentials to obtain a Fourier series of the form used in Lemma 11. Furthermore, $c_p(t)$ and $s_p(t)$ can be explicitly evaluated to obtain

$$c_p(t)=2(-1)^p\sum_{k=1}^{t/2}\frac{(-1)^{k+1}}{(p^2\pi^2)^k}\prod_{i=0}^{2k-2}(t-i), \tag{B4a}$$

$$s_p(t)=2(-1)^p\sum_{k=1}^{(t+1)/2}\frac{(-1)^{k+1}}{[(p+1/2)^2\pi^2]^k}\prod_{i=0}^{2k-2}(t-i). \tag{B4b}$$

We point out that $s_p(t)$ and $c_p(t)$ can be computed in $O(t)$ time on a classical computer using a recursive implementation of the summations in Eq. (B4). We now consider a truncated Fourier series expansion; i.e., we construct the function $\hat{f}_t^N(x)$ from the coefficients $c_p(t)$ and $s_p(t)$ where

$$\hat{f}_t^N(x)=\begin{cases}\sum_{p=0}^{N}c_p(t)\cos(p\pi x) & \text{if } t \text{ is even},\\ \sum_{p=0}^{N}s_p(t)\sin[(2p+1)\pi x/2] & \text{if } t \text{ is odd}.\end{cases} \tag{B5}$$

We then obtain that $\forall x\in[-1,1]$, $|x^t-\hat{f}_t^N(x)|\leqslant e_N$, where

$$e_N(t)=\begin{cases}\sum_{p=N+1}^{\infty}|c_p(t)| & \text{if } t \text{ is even},\\ \sum_{p=N+1}^{\infty}|s_p(t)| & \text{if } t \text{ is odd}.\end{cases} \tag{B6}$$

It now remains to provide bounds on $e_N(t)$ in terms of t and N. We note from Eq. (B4) that

$$\begin{aligned}\forall p>t/\pi,\ |c_p(t)| &\leqslant 2\sum_{k=1}^{t/2}\frac{1}{(p^2\pi^2)^k}\prod_{i=0}^{2k-2}(t-i)\\ &\leqslant 2\sum_{k=1}^{\infty}\frac{t^{2k-1}}{(p^2\pi^2)^k}=\frac{2t}{p^2\pi^2-t^2}.\end{aligned} \tag{B7}$$

A similar bound holds for $|s_p(t)|$:

$$\begin{aligned}\forall p>t/\pi,\ |s_p(t)| &\leqslant 2\sum_{k=1}^{(t+1)/2}\frac{1}{[(p+1/2)^2\pi^2]k}\prod_{i=0}^{2k-2}(t-i)\\ &\leqslant 2\sum_{k=1}^{\infty}\frac{t^{2k-1}}{(p^2\pi^2)^k}\leqslant\frac{2t}{p^2\pi^2-t^2}.\end{aligned} \tag{B8}$$

Consequently, it then follows that

$$\begin{aligned}\forall N>t/\pi,\ e_N(t) &\leqslant 2\sum_{p=N+1}^{\infty}\frac{t}{p^2\pi^2-t^2}\\ &\leqslant\int_N^{\infty}\frac{2t}{x^2\pi^2-t^2}dx=\frac{1}{\pi}\ln\left(\frac{N\pi+t}{N\pi-t}\right).\end{aligned} \tag{B9}$$

To ensure that $e_N(t)$ is smaller than a given precision ε, we can then choose N to be

$$N = \frac{t}{\pi \tanh(\pi\varepsilon/2)} \geqslant \frac{2t}{\pi^2\varepsilon}. \tag{B10}$$

We point out that for this estimate to be correct, the chosen N should also be larger than t/π [Eq. (B9)], which is implied by Eq. (B10) if the precision ε to be smaller than $2/\pi$ and we obtain the estimate provided in Lemma 11.

Finally, we compute the l_1 norm of the coefficients $c_p(t)$ and $s_p(t)$. We note that $(-1)^p c_p(t) \geqslant 0$ and $(-1)^p s_p(t) \geqslant 0$ for all $p \geqslant 0$. This is easily seen as follows—from Eq. (B3), using integration by parts it follows that

$$c_p(t) = 2(-1)^p \frac{t}{p^2\pi^2} - \frac{t(t-1)}{p^2\pi^2} c_p(t-2), \tag{B11a}$$

$$s_p(t) = 2(-1)^p \frac{t}{(p+1/2)^2\pi^2} - \frac{t(t-1)}{(p+1/2)^2\pi^2} s_p(t-2). \tag{B11b}$$

Furthermore, from Eq. (B3), it also follows that

$$|c_p(t)|, |s_p(t)| \leqslant 2\int_0^1 x^t dx = \frac{2}{t+1} \quad \forall t \geqslant 0, \tag{B12}$$

from which it follows that $2t \geqslant |t(t-1)c_p(t-2)|$ and $2t \geqslant |t(t-1)s_p(t-2)|$. Together with Eq. (B11a), it follows that $(-1)^p c_p(t) \geqslant 0$ and $(-1)^p s_p(t) \geqslant 0$ for all $p \geqslant 0$. Therefore,

$$\sum_{p=0}^{\infty} |c_p(t)| = \sum_{p=0}^{\infty} c_p(t)(-1)^p = \sum_{p=0}^{\infty} c_p(t)\cos p\pi = 1 \text{ and}$$

$$\sum_{p=0}^{\infty} |s_p(t)| = \sum_{p=0}^{\infty} s_p(t)(-1)^p = \sum_{p=0}^{\infty} s_p(t)\sin\left(\frac{(2p+1)\pi}{2}\right) = 1.$$

Therefore, the 1-norm of $[c_0(t), c_1(t), \dots, c_N(t)]$ and $[s_0(t), s_1(t), \dots, s_N(t)]$ is less than 1.

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. Bardin, R. Barends, R. Biswas, S. Boixo, F. Brandao, D. Buell, B. Burkett, Y. Chen, J. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler *et al.*, Nature (London) **574**, 505 (2019).

[2] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, P. Hu, X.-Y. Yang, W.-J. Zhang, H. Li, Y. Li, X. Jiang, L. Gan, G. Yang, L. You, Z. Wang *et al.*, Science **370**, 1460 (2020).

[3] J. Preskill, Quantum **2**, 79 (2018).

[4] S. Lloyd, Science **273**, 1073 (1996).

[5] D. W. Berry, A. M. Childs, and R. Kothari, in *Proceedings of the 2015 IEEE 56th Annual Symposium on Foundations of Computer Science* (IEEE, New York, 2015), pp. 792–809.

[6] A. M. Childs and N. Wiebe, Quantum Info. Comput. **12**, 901 (2012).

[7] M. Kliesch, T. Barthel, C. Gogolin, M. Kastoryano, and J. Eisert, Phys. Rev. Lett. **107**, 120501 (2011).

[8] R. Di Candia, J. S. Pedernales, A. Del Campo, E. Solano, and J. Casanova, Sci. Rep. **5**, 9981 (2015).

[9] A. Chenu, M. Beau, J. Cao, and A. del Campo, Phys. Rev. Lett. **118**, 140403 (2017).

[10] R. Cleve and C. Wang, in *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017), Leibniz International Proceedings in Informatics (LIPIcs)*, Vol. 80 (Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2017), pp. 17:1–17:14.

[11] Y. Ge, J. Tura, and J. I. Cirac, J. Math. Phys. **60**, 022202 (2019).

[12] S. Oh, Phys. Rev. A **77**, 012326 (2008).

[13] R. Schützhold and G. Schaller, Phys. Rev. A **74**, 060304(R) (2006).

[14] S. Lu, M. C. Bañuls, and J. I. Cirac, PRX Quantum **2**, 020321 (2021).

[15] A. W. Harrow, A. Hassidim, and S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009).

[16] A. M. Childs, R. Kothari, and R. D. Somma, SIAM J. Comput. **46**, 1920 (2017).

[17] A. Ambainis, in *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)* (LIPIcs, Dagstuhl, Germany, 2012), Vol. 14, pp. 636–647.

[18] D. W. Berry, J. Phys. A: Math. Theor. **47**, 105301 (2014).

[19] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang, Commun. Math. Phys. **356**, 1057 (2017).

[20] A. M. Childs, J.-P. Liu, and A. Ostrander, arXiv:2002.07868.

[21] N. Van Kampen, *Stochastic Processes in Physics and Chemistry*, North-Holland Personal Library (Elsevier, Amsterdam, 1992).

[22] H. Risken and T. Frank, *The Fokker-Planck Equation: Methods of Solution and Applications*, Springer Series in Synergetics (Springer, Berlin, 1996).

[23] P. Gagniuc, *Markov Chains: From Theory to Implementation and Experimentation* (Wiley & Sons, New York, 2017).

[24] A. Montanaro, Proc. R. Soc., Ser. A **471**, 20150301 (2015).

[25] N. Linden, A. Montanaro, and C. Shao, arXiv:2004.06516.

[26] D. An, N. Linden, J.-P. Liu, A. Montanaro, C. Shao, and J. Wang, arXiv:2012.06283.

[27] J. Watrous, J. Comput. Syst. Sci. **62**, 376 (2001).

[28] M. Szegedy, in *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, New York, 2004), pp. 32–41.

[29] S. Subramanian, S. Brierley, and R. Jozsa, J. Phys. Commun. **3**, 065002 (2019).

[30] D. Aharonov, A. Ambainis, J. Kempe, and U. Vazirani, in *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, STOC '01 (Association for Computing Machinery, New York, 2001), pp. 50–59.

[31] S. Apers and A. Sarlette, Quantum Inf. Comput. **19**, 181 (2019).

[32] S. Apers, Quantum walks: Speed limits on mixing and fast-forwarding classical walks, Ph.D. thesis, Ghent University, 2019.

[33] D. Aharonov, V. Jones, and Z. Landau, Algorithmica **55**, 395 (2009).
[34] K. Seki and S. Yunoki, PRX Quantum **2**, 010333 (2021).
[35] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, Commun. Math. Phys. **270**, 359 (2007).
[36] D. W. Berry and A. M. Childs, Quantum Inf. Comput. **12**, 29 (2012).
[37] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett. **100**, 160501 (2008).
[38] I. M. Georgescu, S. Ashhab, and F. Nori, Rev. Mod. Phys. **86**, 153 (2014).
[39] A. Aspuru-Guzik and P. Walther, Nat. Phys. **8**, 285 (2012).
[40] I. Buluta and F. Nori, Science **326**, 108 (2009).
[41] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Phys. Rev. Lett. **81**, 5442 (1998).
[42] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. De Wolf, J. Assoc. Comput. Mach. **48**, 778 (2001).
[43] C.-F. Chiang, D. Nagaj, and P. Wocjan, Quantum Inf. Comput. **10**, 420 (2010).