Quantum Information and Computation, Vol. 12, No. 1&2 (2012) 0029–0062
© Rinton Press

BLACK-BOX HAMILTONIAN SIMULATION AND UNITARY IMPLEMENTATION

DOMINIC W. BERRY
Institute for Quantum Computing, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1
Department of Physics and Astronomy, Macquarie University, Sydney, NSW 2109, Australia

ANDREW M. CHILDS
Department of Combinatorics & Optimization and Institute for Quantum Computing
University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

Received July 8, 2011
Revised November 1, 2011

We present general methods for simulating black-box Hamiltonians using quantum walks. These techniques have two main applications: simulating sparse Hamiltonians and implementing black-box unitary operations. In particular, we give the best known simulation of sparse Hamiltonians with constant precision. Our method has complexity linear in both the sparseness D (the maximum number of nonzero elements in a column) and the evolution time t, whereas previous methods had complexity scaling as D^4 and were superlinear in t. We also consider the task of implementing an arbitrary unitary operation given a black-box description of its matrix elements. Whereas standard methods for performing an explicitly specified $N \times N$ unitary operation use $\tilde{O}(N^2)$ elementary gates, we show that a black-box unitary can be performed with bounded error using $O(N^{2/3}(\log\log N)^{4/3})$ queries to its matrix elements. In fact, except for pathological cases, it appears that most unitaries can be performed with only $\tilde{O}(\sqrt{N})$ queries, which is optimal.

Keywords: Quantum computation, quantum query complexity, Hamiltonian simulation, quantum walk
Communicated by: R Jozsa & B Terhal

1 Introduction

One of the major applications of quantum computation is the simulation of Hamiltonian dynamics. Hamiltonian simulation is the basis for simulating quantum systems—the original motivation for quantum computers [1]—and also has applications to quantum algorithms [2, 3, 4, 5].

An explicit procedure for simulating local Hamiltonians on a quantum computer was given by Lloyd [6]. This result was later substantially generalized to the simulation of sparse Hamiltonians by Aharonov and Ta-Shma [7]. References [8, 9] improved these results by providing a simulation scheme with complexity that scales close to linearly in the evolution time t and as the fourth power of the sparseness parameter D, the maximum number of nonzero elements in a column.

In this paper, we consider the general task of simulating Hamiltonians, without necessarily assuming sparsity. As particular applications, we provide improved methods for simulating sparse Hamiltonians and implementing unitary transformations. Similar to previous work on Hamiltonian simulation [7, 8, 9], we assume throughout that the Hamiltonian is specified by an oracle. Specifically, in our model, a black-box function computes the matrix element H_{jk} for any desired row index $j \in \{1, 2, \ldots, M\}$ and column index $k \in \{1, 2, \ldots, M\}$. To simultaneously treat the case of sparse Hamiltonians, we also consider a black box that computes the positions of the nonzero matrix elements. (See Sec. 2.1 for a detailed discussion of our model and its relationship to prior work.)

An algorithm for computing the matrix elements of H can be used to construct such a black box. In particular, this means that the black-box model applies to common physical Hamiltonians such as those considered by Lloyd [6]. Such Hamiltonians are a sum of local terms, each acting on a limited number of subsystems. Thus they are sparse, and there is an efficient method for computing the matrix elements of the overall Hamiltonian. An advantage of the black-box model of Hamiltonian simulation is that it can be applied not only to physical Hamiltonians, but as a basis for designing algorithms for other problems [2, 4, 5].

In contrast to previous work on Hamiltonian simulation [6, 7, 2, 8, 9], most of which is based on Lie-Trotter-Suzuki formulae, we use a new approach based on a quantum walk [10]. A limitation of the Lie-Trotter-Suzuki approach is that it relies on limiting the error by using many short time steps. As a result, the complexity of the simulation always scales superlinearly in t. The quantum walk approach provides scaling that is strictly linear in t [10], which is known to be optimal [9]. Another limitation of the Lie-Trotter-Suzuki approach is that it relies on decomposing the Hamiltonian into 1-sparse matrices, which results in poor scaling in the sparseness D. In Ref. [9] the scaling was $\tilde{O}(D^4)$, which was recently improved to $\tilde{O}(D^3)$ [11].[a] In contrast, by using quantum walks we improve the scaling to linear in D. This represents the best known constant-precision simulation of sparse Hamiltonians. Furthermore, if we are willing to accept superlinear scaling in t, the scaling in D may be improved to $\tilde{O}(D^{2/3})$ in general, and in many cases to $\tilde{O}(D^{1/2})$.

The quantum walk approach to Hamiltonian simulation was proposed in Ref. [10], though without providing an explicit method to implement the steps of the quantum walk in the general case. Here we present a complete method for Hamiltonian simulation by showing how to implement the steps of the quantum walk for a general Hamiltonian that may or may not be sparse. We use the method of Ref. [10] together with a range of other tools, which we combine and improve on in nontrivial ways to obtain our final result. In particular, our approach introduces the following techniques.

1. In Sec. 4, we modify the method of Ref. [10] by using phase estimation to correct a lazy quantum walk, giving a more efficient simulation.

2. In Sec. 5, we describe how to perform steps of the quantum walk using state preparation by amplitude amplification (similar to a method proposed in Ref. [12]), improving the efficiency in the non-sparse case.

3. We modify the state to be prepared by using an ancilla qubit to satisfy an orthogonality condition for the lazy quantum walk (compare Eqs. (11) and (24)). This facilitates more

[a]We use a tilde to indicate that subpolynomial scaling is ignored—that is, $f = \tilde{O}(g)$ if $f = O(g^{1+\eta})$ for any $\eta > 0$.

efficient state preparation by amplitude amplification in Sec. 5 (outperforming direct application of Ref. [12]), and even allows us to prepare the state in only $O(1)$ queries in the sparse case described in Sec. 4.

4. In Sec. 7, we further improve the simulation in the non-sparse case by decomposing the Hamiltonian as a sum of terms and recombining these terms using Lie-Trotter-Suzuki formulae. The decomposition depends on the magnitudes of the matrix elements, giving an approach that is fundamentally different from previous applications of Lie-Trotter-Suzuki formulae.

While our results on simulating non-sparse Hamiltonians may be of interest in their own right, additional motivation for studying this problem comes from the related task of implementing general unitary transformations. Standard methods for implementing an arbitrary $N \times N$ unitary transformation on a quantum computer work by decomposing it into a product of two-level unitary matrices [13, 14] and performing the two-level unitaries via the Solovay-Kitaev theorem [15, 16, 17]. This method uses $N^2 \operatorname{poly}(\log N)$ gates. Since counting arguments show that $\Omega(N^2)$ elementary gates are required even to approximate a general unitary transformation [14, 18, 17], such an implementation is nearly optimal.

Instead of considering an explicit unitary operation, we study the problem of performing a unitary transformation specified by a black box for its matrix elements, similar to the black box for a non-sparse Hamiltonian described above. In such an oracle model, counting arguments for unitary implementation no longer apply, since the black box depends on the unitary. While the question of how many queries are required to implement a general unitary in this model seems quite natural, to the best of our knowledge it has not been studied previously.

Implementation of unitaries is closely connected to Hamiltonian simulation, because one can implement a unitary by simulating a related Hamiltonian. Reference [19] used this idea to provide a method for implementing sparse unitaries. However, their approach relies on a decomposition into 1-sparse Hamiltonians, so it performs poorly in the non-sparse case.

Using Hamiltonian simulation via quantum walks, we show that the complexity of implementing a general (non-sparse) $N \times N$ unitary scales with N as $\tilde{O}(N^{2/3})$. We also present numerical evidence that typical unitaries can be implemented in only $\tilde{O}(\sqrt{N})$ queries (although it is possible to construct unitaries for which our method uses more queries). This is much less than the $\Omega(N^2)$ elementary gates required to implement a general unitary given an explicit description instead of a black box. The best lower bound we are aware of is $\Omega(\sqrt{N})$, because implementation of a black-box unitary operation can be used to solve a search problem [20].

Implementation of black-box unitary transformations is closely related to the task of preparing an N-dimensional quantum state given a black box for its amplitudes in a fixed basis. Grover showed that the query complexity of this task is $\Theta(\sqrt{N})$ [12]. As mentioned above, we build on his technique in order to implement black-box unitaries. It is an open question whether black-box unitaries can be implemented in $O(\sqrt{N})$ queries in general, or if there is a fundamental separation between the query complexity of implementing unitaries and the query complexity of preparing states.

Since implementing unitary transformations is a basic task in quantum computation, we expect this result to have applications to quantum algorithms. For example, our approach could serve as an alternative to previous methods for efficiently implementing general unitary transformations on a logarithmic number of qubits, with improved performance provided the matrix elements can be computed quickly.

The remainder of this article is organised as follows. In Sec. 2 we give a technical summary of our main contributions. Then, in Sec. 3, we summarise the method of Ref. [10] for simulating Hamiltonian evolution. Our main result, that a black-box Hamiltonian can be simulated with $\tilde{O}(D^{2/3})$ queries to its matrix elements, is proven in Sec. 7, building on a foundation established in Secs. 4 through 6. Some of these intermediate results may be of interest in their own right; in particular, in Sec. 4, we present a simple method to perform the steps of the quantum walk of Ref. [10] that is especially suitable for sparse Hamiltonians. We explain how Hamiltonian simulation can be used to implement black-box unitaries in Sec. 8. In Sec. 9 we give some examples of the simulation as applied to particular unitary operations. We conclude in Sec. 10 with a summary of the results and a discussion of some open problems.

2 Model and results

2.1 Model

We formulate the Hamiltonian simulation and unitary implementation problems using an oracle model, in which a description of the Hamiltonian or unitary is provided by a black box. For Hamiltonian simulation, the matrix elements of some Hermitian matrix $H \in \mathbb{C}^{M\times M}$ are given by a black box O_H acting as

$$O_H|j,k\rangle|z\rangle = |j,k\rangle|z \oplus H_{jk}\rangle, \tag{1}$$

where $j,k \in \{1,2,\dots,M\}$. Here the matrix element H_{jk} is represented by its real and imaginary parts written in binary, and $\oplus$ denotes the bitwise XOR of such representations. Similarly, for the problem of implementing a unitary $U \in \mathbb{C}^{N\times N}$, we are given a black box O_U acting as

$$O_U|j,k\rangle|z\rangle = |j,k\rangle|z \oplus U_{jk}\rangle, \tag{2}$$

where $j,k \in \{1,2,\dots,N\}$. The error of the simulation or implementation must be no greater than δ as quantified by the trace distance. In practice, the black box O_H or O_U provides the matrix elements to some finite precision, ζ, using $O(\log\frac{1}{\zeta})$ qubits. We assume that $\zeta \ll \delta$, so the imprecision in this approximation does not affect the analysis.

We can also take advantage of sparsity if it is possible to compute the positions of nonzero matrix elements. Specifically, suppose there are at most D nonzero elements in each row or column. For Hamiltonian simulation, suppose that in addition to the black box O_H, we are given a black box O_F acting as

$$O_F|j,k\rangle = |j, f(j,k)\rangle \tag{3}$$

for any $j \in \{1,2,\dots,M\}$ and $k \in \{1,2,\dots,D\}$, where the function $f(j,k)$ gives the row index of the kth nonzero element in column j (or the row index of any zero element when there are fewer than k nonzero elements in column j).

Note that O_F computes the row index in place. In contrast, some previous work on simulating sparse Hamiltonians [9, 11] assumes that a single query only computes $f(j,k)$ given j and k, i.e., performs the isometry O_F' acting as $O_F'|j,k\rangle = |j,k,f(j,k)\rangle$. The oracle

O_F can be used to produce such an oracle in one query, simply by copying k to a third register before calling O_F. Therefore, all the upper bounds for the complexity in [9, 11] hold for the oracle O_F. Furthermore, the algorithms in [9, 11] do not depend on this aspect of the oracle, so changing the oracle in those algorithms would not lead to improved upper bounds. Thus our results may be directly compared with those of [9, 11].

To construct the oracle O_F, it suffices to first compute $f(j,k)$ for a given j, and then compute k given j and $f(j,k)$ in order to erase the register encoding k. In contrast, O'_F does not uncompute k. For realistic cases such as the local Hamiltonians considered by Lloyd [6], determining k given j and $f(j,k)$ is not difficult, so O_F is a realistic representation of the resources used.

If desired, one can quantify the resources used by our simulations in terms of queries to the black box O'_F instead of to O_F. Even if k cannot be computed directly, one can implement O_F with O'_F using additional queries to uncompute k. If the function f provides the nonzero elements in sorted order, one can find k by binary search, increasing the number of queries by a factor of only $\log D$. In general, one can use Grover's algorithm to find k, increasing the number of queries by a factor of $O(\sqrt{D})$.

Another model used by some papers on sparse Hamiltonian simulation is that for any given j, a single query reveals all the nonzero entries in the jth column, i.e., the values $f(j,1),\ldots,f(j,D)$ [7, 2, 8]. With that model, the black box O_F can be implemented using two queries, whereas D calls to O_F are required to compute all the values $f(j,1),\ldots,f(j,D)$. As Refs. [7, 2, 8] are primarily concerned with showing polynomial scaling in D, such a difference is unimportant.

We emphasise that in the present work, we do not require $D = \mathrm{poly}(\log M)$; our methods apply for any $D \leq M$. If the Hamiltonian is not sparse, or if it is sparse but the nonzero elements are in unknown positions, then we can simply take $D = M$ and let O_F be the identity operation. Thus, all the results of the paper hold for non-sparse cases, with $D = M$. In particular, when considering the problem of unitary implementation, we do not assume sparsity, so the black box O_F is not required.

We assume that information about the Hamiltonian or unitary can only be obtained by querying the oracle, so in particular we do not know the norms of H or U (except for the trivial fact that $\|U\| = 1$). However, we assume that we do have upper bounds on various norms: we are given constants $\Lambda, \Lambda_1, \Lambda_{\max}$ satisfying $\Lambda \geq \|H\|$, $\Lambda_1 \geq \|H\|_1$, and $\Lambda_{\max} \geq \|H\|_{\max}$, where $\|H\|$ denotes the spectral norm of H, $\|H\|_1 := \max_j \sum_{k=1}^{M} |H_{jk}|$, and $\|H\|_{\max} := \max_{j,k} |H_{jk}|$.

2.2 Results

Our first main result, proved in Sec. 4, is that a Hamiltonian can be simulated with scaling linear in both $\|H\|t$ and D.

Theorem 1. *For a given Hamiltonian H, let $\Lambda \geq \|H\|$ and $\Lambda_{\max} \geq \|H\|_{\max}$. Then the evolution under H for time t can be simulated with error at most $\delta \in (0,1]$ using*

$$O\left(\frac{\Lambda t}{\sqrt{\delta}} + D\Lambda_{\max}t + 1\right) \tag{4}$$

queries to O_H and O_F.

This result is suitable for simulation of sparse Hamiltonians. The next main result, proved in Sec. 7.3, gives improved scaling in D at the expense of worse scaling in $\|H\|t$. This result may be preferable for non-sparse Hamiltonians.

Theorem 2. *Let $\Lambda \geq \|H\|$. The evolution under the Hamiltonian H for time t can be simulated with error at most $\delta \in (0,1]$ using*

$$O\left(D^{2/3}[(\log\log D)\Lambda t]^{4/3}\delta^{-1/3}\right) \tag{5}$$

queries to O_H and O_F, provided $\delta D > \Lambda t > \sqrt{\delta}$.

Using the correspondence between Hamiltonian simulation and unitary implementation described in Sec. 8, this easily implies our main result on the implementation of black-box unitaries (see Sec. 8 for the proof).

Corollary 3. *A black-box unitary operation U can be implemented with error at most $\delta \in (0,1]$ using*

$$O\left(N^{2/3}(\log\log N)^{4/3}\delta^{-1/3}\right). \tag{6}$$

queries to O_U.

Although the above results are the best we are able to show for general non-sparse Hamiltonians and unitaries, we believe that our methods are typically more efficient. Theorem 2 and Corollary 3 are based on a decomposition of the Hamiltonian into a sum of terms, where the nonzero matrix elements of each term have comparable size. In the worst-case analysis of Sec. 7.3, we must take into account the possibility that in this decomposition, the spectral norms of the individual terms could be much larger than the spectral norm of the total Hamiltonian. Numerically, we find that this does not occur when selecting matrices at random, only when matrices are specifically designed to cause this behaviour (see Sec. 7.2). Assuming that all terms in the decomposition have comparable norms, we show in Sec. 7.1 that the number of queries to simulate a Hamiltonian with $\|H\| \leq \Lambda$ is

$$O\left((\Lambda t)^{3/2}\sqrt{D/\delta}(\log D)^{7/4}\right); \tag{7}$$

correspondingly, a black-box unitary can be implemented using

$$O\left(\sqrt{N/\delta}(\log N)^{7/4}\right) \tag{8}$$

queries.

3 Review of Hamiltonian simulation

In this section we summarise an approach to Hamiltonian simulation based on discrete-time quantum walks [10]. Throughout, M denotes the dimension of the Hilbert space that a black-box Hamiltonian acts on, and N the dimension of the space that a black-box unitary transformation acts on. To construct a discrete-time quantum walk from a given Hamiltonian H, the Hilbert space is expanded from $\mathbb{C}^M$ to $\mathbb{C}^{M+1} \otimes \mathbb{C}^{M+1}$. A step of the discrete-time quantum walk is described by a unitary operator

$$V := iS(2TT^\dagger - \mathbb{1}). \tag{9}$$

Here the operator S swaps the two registers, i.e., $S|j,k\rangle = |k,j\rangle$ for all $j,k \in \{1,2,\ldots,M+1\}$. The operator T is the isometry

$$T := \sum_{j=1}^{M} |\eta_j\rangle\langle j| \tag{10}$$

mapping $|j\rangle$ to $|\eta_j\rangle := |j\rangle|\varphi_j\rangle$, where

$$|\varphi_j\rangle := \sqrt{\frac{\epsilon}{\|H\|_1}} \sum_{k=1}^{M} \sqrt{H_{jk}^*}|k\rangle + \sqrt{1 - \frac{\epsilon\sigma_j}{\|H\|_1}}|M+1\rangle \tag{11}$$

with

$$\sigma_j := \sum_{k=1}^{M} |H_{jk}| \tag{12}$$

(cf. Eq. (27) of Ref. [10]). Here $\epsilon \in (0,1]$ is a parameter that can be made small to obtain a *lazy quantum walk*, and $\|H\|_1 := \max_j \sum_{k=1}^{M} |H_{jk}|$.

The state $|\varphi_j\rangle$ is chosen so that $\langle\eta_j|S|\eta_k\rangle$ is proportional to H_{jk}. We have

$$\begin{aligned}\langle\eta_j|S|\eta_k\rangle &= \langle j|\varphi_k\rangle\langle\varphi_j|k\rangle \\ &= \frac{\epsilon}{\|H\|_1}\sqrt{H_{kj}^*}\left(\sqrt{H_{jk}^*}\right)^*.\end{aligned} \tag{13}$$

Note that caution is needed when choosing the sign of the square root. Provided H_{jk} is not a negative real number, it suffices to take the principal square root of both H_{kj}^* and H_{jk}^* (i.e., if $z = re^{i\theta}$ for some $r \geq 0$ and $\theta \in (-\pi, \pi)$, define $\sqrt{z} := \sqrt{r}e^{i\theta/2}$, so that $\sqrt{z}(\sqrt{z^*})^* = z$). This choice ensures that Eq. (13) gives $\epsilon H_{jk}/\|H\|_1$. However, if H_{jk} is a negative real number, this choice does not suffice. Instead, for $H_{jk} \in (-\infty, 0)$ with $j \neq k$, we take

$$\sqrt{H_{jk}^*} = \operatorname{sign}(j-k)\, i\sqrt{|H_{jk}|}. \tag{14}$$

By taking a different sign above and below the diagonal, we ensure that Eq. (13) is negative, as required.

The above prescription does not handle the case where the Hamiltonian has negative diagonal elements. To ensure that the diagonal entries are nonnegative, we simply add a multiple of the identity: given an upper bound $\Lambda_{\max}$ on $\|H\|_{\max}$, we replace H with $H + \Lambda_{\max}\mathbb{1}$. This only changes the Hamiltonian evolution for time t by a global phase of $e^{-i\Lambda_{\max}t}$, and the relevant norms of H are increased by at most a factor of 2.

The eigenvalues and eigenvectors of V are closely related to those of H [21]. If we define $\tilde{H}$ to be the operator with matrix elements

$$\tilde{H}_{jk} := \langle\eta_j|S|\eta_k\rangle, \tag{15}$$

then Eq. (13) gives

$$\tilde{H} = \frac{\epsilon H}{\|H\|_1}. \tag{16}$$

The operators H and $\tilde{H}$ have common eigenstates $|\lambda\rangle$. The corresponding eigenvalues for H and $\tilde{H}$ are denoted λ and $\tilde{\lambda}$, and are related by

$$\tilde{\lambda} = \frac{\epsilon\lambda}{\|H\|_1}. \tag{17}$$

Each eigenstate $|\lambda\rangle$ corresponds to two eigenvectors of V,

$$|\mu_\pm^\lambda\rangle := \frac{1 - e^{\pm i \arcsin \tilde\lambda} S}{\sqrt{2(1-\tilde\lambda^2)}} T|\lambda\rangle, \tag{18}$$

with eigenvalues

$$\mu_\pm^\lambda := \pm e^{\pm i \arcsin \tilde\lambda}. \tag{19}$$

Reference [10] describes simulations of H based on the quantum walk V. One approach is to use phase estimation to (coherently) determine the value of $\tilde\lambda$. Introducing a phase of $\exp(-i\lambda t) = \exp(-i\tilde\lambda t \|H\|_1/\epsilon)$ for each eigenvector $|\lambda\rangle$ simulates evolution under H for time t. More specifically, a general initial state has the form

$$|\psi\rangle = \sum_{\lambda,k} \psi_{\lambda,k} |\lambda, k\rangle, \tag{20}$$

where the index k accounts for degenerate eigenvalues of H. Applying the operation T yields

$$\begin{aligned} T|\psi\rangle &= \sum_{\lambda,k} \psi_{\lambda,k} T|\lambda,k\rangle \\ &= \sum_{\lambda,k} \frac{\psi_{\lambda,k}}{\sqrt{2(1-\tilde\lambda^2)}} [(1 - \tilde\lambda e^{-i \arccos \tilde\lambda})|\mu_+^\lambda, k\rangle + (1 - \tilde\lambda e^{i \arccos \tilde\lambda})|\mu_-^\lambda, k\rangle], \end{aligned} \tag{21}$$

where the $|\mu_\pm^\lambda, k\rangle$ are eigenvectors of V, with the index k labeling an orthonormal basis for each eigenspace. Applying the correct phase factor for each eigenspace gives

$$T e^{-iHt}|\psi\rangle = \sum_{\lambda,k} e^{-i\lambda t} \psi_{\lambda,k} T|\lambda,k\rangle. \tag{22}$$

Applying $T^\dagger$ then gives $e^{-iHt}|\psi\rangle$, the desired time-evolved state.

Phase estimation can provide an estimate of $\mu_\pm^\lambda$ with variance approximately $(\pi/d)^2$ using d applications of V. The variance in λ is then $O(\|H\|_1^2/\epsilon^2 d^2)$, which translates to an error in the state of $O(\|H\|_1 t/\epsilon d)$. If the allowed error is δ, then the simulation can be achieved with $d = O(\|H\|_1 t/\delta)$ by taking $\epsilon = 1$. (Note that the δ used here is the square root of that used in Ref. [10], because that paper considered a lower bound on the fidelity of $1-\delta$, whereas we take δ to be an upper bound on the trace distance.)

4 Sparse Hamiltonian simulation

The simulation scheme presented in Ref. [10] quantifies the complexity in terms of the number of quantum walk steps. To simulate a black-box Hamiltonian, we require a method to perform these steps. In this section we describe a simple approach to this problem, thereby providing a Hamiltonian simulation method suitable for the sparse case.

The walk step is composed of two operators, the swap S and a reflection $2TT^\dagger - \mathbb{1}$. The operation S is easy to implement; the difficulty lies in implementing the reflection. It is given explicitly by

$$2TT^\dagger - \mathbb{1} = \sum_{j=1}^{M} |j\rangle\langle j| \otimes (2|\varphi_j\rangle\langle\varphi_j| - \mathbb{1}). \tag{23}$$

That is, it is a reflection about $|\varphi_j\rangle$ conditional on the state $|j\rangle$ in the first register. To perform this reflection, it suffices to give a procedure for preparing $|\varphi_j\rangle$ from the $|0\rangle$ state: by performing inverse state preparation, reflecting about $|0\rangle$, and then performing state preparation, we effectively reflect about $|\varphi_j\rangle$.

Black-box preparation of an M-dimensional quantum state can clearly be performed in M queries, but this would introduce an overall multiplicative factor of M in the complexity of the implementation. The overall query complexity of the simulation would then be $O(M\|H\|_1 t/\delta)$. Taking advantage of sparsity reduces this to $O(D\|H\|_1 t/\delta)$, but Theorem 1 uses even fewer queries.

Our improved simulation uses the following insight. We modify the state $|\varphi_j\rangle$ to

$$|\phi_j\rangle := \sqrt{\frac{\epsilon}{\|H\|_1}} \sum_{k=1}^{M} \sqrt{H_{jk}^*}|k\rangle|0\rangle + \sqrt{1-\frac{\epsilon\sigma_j}{\|H\|_1}}|\zeta_j\rangle|1\rangle, \tag{24}$$

where $|\zeta_j\rangle$ is some superposition of the $|k\rangle$. That is, we append an ancilla qubit, and replace $|M+1\rangle$ with $|\zeta_j\rangle|1\rangle$. The second term of Eq. (24), flagged by a $|1\rangle$ state in the ancilla qubit, takes the place of the $|M+1\rangle$ state in Eq. (11). Thus the discrete-time quantum walk takes place in $\mathbb{C}^{2M} \otimes \mathbb{C}^{2M}$, although it is effectively confined to a subspace of dimension $(M+1)^2$.

To take account of the fact that $\|H\|_1$ may not be known exactly, we replace ϵ with $\varepsilon = \epsilon\Lambda_1/\|H\|_1$, where Λ_1 is a known upper bound on $\|H\|_1$. Then we can alternatively express the definition of $|\phi_j\rangle$ as

$$|\phi_j\rangle := \sqrt{\frac{\varepsilon}{\Lambda_1}} \sum_{k=1}^{M} \sqrt{H_{jk}^*}|k\rangle|0\rangle + \sqrt{1-\frac{\varepsilon\sigma_j}{\Lambda_1}}|\zeta_j\rangle|1\rangle. \tag{25}$$

Note that the restriction $\epsilon \le 1$ implies that $\varepsilon \le \Lambda_1/\|H\|_1$. The relation between the eigenvalues λ and $\tilde{\lambda}$ can be expressed in terms of ε as

$$\lambda = \tilde{\lambda}\Lambda_1/\varepsilon. \tag{26}$$

Provided ε is sufficiently small, we can prepare this state using a constant number of queries. In particular:

Lemma 4. *The state $|\phi_j\rangle$ in Eq. (25) can be prepared in $O(1)$ calls to the oracles O_H and O_F provided $\varepsilon \in (0, \Lambda_1/D\Lambda_{\max}]$, where $\Lambda_{\max} \ge \|H\|_{\max}$ and $\Lambda_1 \ge \|H\|_1$.*

Proof. First, prepare an equal superposition over $|1\rangle$ to $|D\rangle$ in the first register and initialise the ancilla qubit to $|0\rangle$, giving

$$\frac{1}{\sqrt{D}} \sum_{k=1}^{D} |k\rangle|0\rangle. \tag{27}$$

Querying the black box O_F changes this to

$$|\phi_j^a\rangle := \frac{1}{\sqrt{D}} \sum_{k\in F_j} |k\rangle|0\rangle, \tag{28}$$

where F_j is the set of indices given by O_F on input j.

Next we transform $|\phi_j^a\rangle$ to

$$\frac{1}{\sqrt{D}}\sum_{k\in F_j}|k\rangle\left[\sqrt{\frac{H_{jk}^*}{X}}|0\rangle+\sqrt{1-\frac{|H_{jk}|}{X}}|1\rangle\right] \tag{29}$$

where $X=\Lambda_1/\varepsilon D$. The most important requirement on X is that $X\geq\Lambda_{\max}$, so $X\geq\|H\|_{\max}$ and the amplitude for the $|0\rangle$ state has magnitude at most 1. Because of the requirement that $\varepsilon\leq\Lambda_1/D\Lambda_{\max}$, taking $X=\Lambda_1/\varepsilon D$ ensures that $X\geq\Lambda_{\max}$. In addition, for this value of X, Eq. (29) has the form of $|\phi_j\rangle$ for some choice of $|\zeta_j\rangle$.

The state $|\phi_j^a\rangle$ can be transformed to Eq. (29) by computing H_{jk} in an ancilla register with the black box O_H and using this value to perform a controlled rotation on the qubit. Applying O_H again uncomputes the ancilla storing H_{jk}. Note that the rotation can be performed with error at most δ using poly$(\log\frac{1}{\delta})$ operations [15, 16], but this factor is not included in the analysis since we focus on the query complexity. □

Next, we improve the Hamiltonian simulation method reviewed in Sec. 3 by combining a lazy quantum walk with phase estimation.

Lemma 5. *Let $\Lambda_1 t/\varepsilon\in\mathbb{Z}$ and $\varepsilon\in(0,1]$. Evolution under H for time t can be simulated using $O(\Lambda_1 t/\varepsilon)$ steps of the quantum walk defined by the states $|\phi_j\rangle$ from Eq.* (25)*, with error $O(\Lambda^2\varepsilon^2/\Lambda_1^2)$.*

Proof. Given an arbitrary input state, first we (coherently) determine the sign $\pm$ of $\mu_\pm^\lambda$ (recall Eq. (19)). The phase of μ_+^λ is $\arcsin\tilde{\lambda}$, whereas the phase of μ_-^λ is $\pi-\arcsin\tilde{\lambda}$. Performing phase estimation with one bit of precision on V gives probabilities of measuring $+$ or $-$, given that the eigenvalue is μ_+^λ or μ_-^λ, of

$$\Pr(+|\pm)=\frac{1\pm\sqrt{1-\tilde{\lambda}^2}}{2}, \tag{30}$$

$$\Pr(-|\pm)=\frac{1\mp\sqrt{1-\tilde{\lambda}^2}}{2}. \tag{31}$$

The probability of error is therefore $O(\tilde{\lambda}^2)$. Since $\tilde{\lambda}=\varepsilon\lambda/\Lambda_1$ and $\lambda\leq\Lambda$, the error due to misidentification of the sign is $O(\Lambda^2\varepsilon^2/\Lambda_1^2)$.

Having estimated the sign, we can apply a lazy quantum walk more accurately than in Ref. [10]. If the sign of $\mu_\pm^\lambda$ is $-$, we apply V^d, where $d=\Lambda_1 t/\varepsilon$ is an integer (chosen so that $\lambda t=\tilde{\lambda}d$), giving a phase factor of

$$(\mu_-^\lambda)^d=(-e^{-i\arcsin\tilde{\lambda}})^d=(-1)^d e^{-i\lambda t}+O(d\tilde{\lambda}^3). \tag{32}$$

(The sign can be corrected if d is odd.) Similarly, if the sign of $\mu_\pm^\lambda$ is $+$, we apply $(V^\dagger)^d$, giving a phase factor

$$[(\mu_+^\lambda)^*]^d=(e^{-i\arcsin\tilde{\lambda}})^d=e^{-i\lambda t}+O(d\tilde{\lambda}^3). \tag{33}$$

In either case, the error in the final state is

$$O(d\tilde{\lambda}^3)=O(\lambda^3 t(\varepsilon/\Lambda_1)^2)\leq O(\Lambda^3 t(\varepsilon/\Lambda_1)^2), \tag{34}$$

considerably less than that for the method given in Ref. [10, Theorem 2] for small ε.

We further reduce the error by using an estimate of $\tilde{\lambda}$ to correct the lazy quantum walk. After the lazy quantum walk implements the phase factor $e^{-id \arcsin \tilde{\lambda}}$, the estimate of $\tilde{\lambda}$ is used to correct the difference between $\tilde{\lambda}$ and $\arcsin \tilde{\lambda}$. With d applications of V, we obtain an estimate of $\arcsin \tilde{\lambda}$ with standard deviation $O(1/d)$ (see for example [10, Theorem 5]). Since $\tilde{\lambda} - \arcsin \tilde{\lambda} = O(\tilde{\lambda}^3)$, this estimate only improves the accuracy if $1/d$ is small compared to $\tilde{\lambda}$. If $\tilde{\lambda} d < 1$, we do not perform a correction. In that case the error is $O(d\tilde{\lambda}^3) \le O(\tilde{\lambda}^2)$. On the other hand, if $\tilde{\lambda} d \ge 1$, then the standard deviation in the estimate of $\tilde{\lambda} - \arcsin \tilde{\lambda}$ is $O(\tilde{\lambda}^2/d)$, and again the error in the final phase is $O(\tilde{\lambda}^2)$. So in both cases, the error in the final state is $O(\tilde{\lambda}^2) \le O(\Lambda^2 \varepsilon^2/\Lambda_1^2)$. The overall number of steps of the quantum walk used is $O(d) = O(\Lambda_1 t/\varepsilon)$. □

Combining the results of Lemmas 4 and 5 gives the improved Hamiltonian simulation method described by Theorem 1.

Proof of Theorem 1. We apply the Hamiltonian simulation described in Lemma 5, with the method described in Lemma 4 to perform the steps of the quantum walk.

Using

$$X = \frac{1}{Dt} \max \left\{ \lceil \Lambda t/\sqrt{\delta} \rceil, \lceil \Lambda_{\max} D t \rceil \right\}, \tag{35}$$

we take $\varepsilon = \Lambda_1 / DX$. This value of X satisfies $X \ge \Lambda_{\max}$ (as we always require for X), which implies that the condition $\varepsilon \le \Lambda_1 / D\Lambda_{\max}$ of Lemma 4 is satisfied. Thus, by Lemma 4, the steps of the quantum walk can be performed using $O(1)$ queries.

With this value of ε, $\Lambda_1 t/\varepsilon = XDt$ is an integer, and we can use Lemma 5. Then the error is

$$O(\Lambda^2 \varepsilon^2/\Lambda_1^2) = O(\Lambda^2/D^2 X^2) \le O(\delta). \tag{36}$$

The total number of oracle calls is $O(\Lambda_1 t/\varepsilon) = O(XDt)$, and is therefore

$$O(\lceil \Lambda t/\sqrt{\delta} \rceil + \lceil \Lambda_{\max} D t \rceil) = O(\Lambda t/\sqrt{\delta} + \Lambda_{\max} D t + 1) \tag{37}$$

as claimed. □

Comparing this to the simulation of sparse Hamiltonians using high-order integrators [9], the scaling is better in terms of all parameters except δ. (We assume that the upper bounds on norms of H have the same order as the norms themselves, so for example $\Lambda = O(\|H\|)$.) The number of queries is only linear in $\|H\|t$, as opposed to slightly superlinear. The scaling is particularly improved in terms of D, as it is only linear, whereas the scaling in Ref. [9] was as D^4. The scaling in δ is as $1/\sqrt{\delta}$, as opposed to an arbitrarily small power in Ref. [9]. However, there is an advantage in that for $\delta = O(1/D^2)$ there is no further explicit dependence on D.

This also improves over the method of Ref. [10], which uses $O(\|H\|_1 t/\delta)$ steps of the quantum walk. Using a naive method for state preparation—simply querying all nonzero elements—uses $O(D\|H\|_1 t/\delta)$ queries. Theorem 1 improves on this as $\|H\|_1$ is typically larger than both $\|H\|_{\max}$ and $\|H\|$, and because the scaling with δ is improved. In particular, the bound $\|H\|_1 \le D\|H\|_{\max}$ shows that $D\|H\|_1 t/\delta \le O(D^2 \|H\|_{\max} t/\delta)$, which is worse than

$D\|H\|_{\max}t$, and the bound $\|H\|_1 \le \sqrt{D}\|H\|$ shows that $D\|H\|_1 t/\delta \le O(D^{3/2}\|H\|t/\delta)$, which is worse than $\|H\|t/\sqrt{\delta}$.

We conclude this section by describing in more detail how the isometry T is used in each part of the simulation. It is used in three different ways:

1. At the beginning, to map the initial state from $\mathbb{C}^M$ into the tensor product space $\mathbb{C}^{2M} \otimes \mathbb{C}^{2M}$.
2. To implement each application of V.
3. At the end, to map the final state from $\mathbb{C}^{2M} \otimes \mathbb{C}^{2M}$ back to $\mathbb{C}^M$.

For the first step, we can directly implement T as defined in Eq. (10). The initial state is a superposition of states $|j\rangle$ for $j \in \{1, 2, \dots, M\}$ used to control the state preparation. We simply introduce another register in which $|\phi_j\rangle$ is prepared.

When T is used to implement V, we need to specify the action on the extra qubit introduced in the state preparation procedure. We only require the correct eigenvalue and eigenvector relations, namely, that

$$T^\dagger S T|\lambda\rangle = \tilde{\lambda}|\lambda\rangle. \tag{38}$$

On the expanded space, the isometry T should have the form

$$\sum_{j=1}^{M} \left[|j,0\rangle|\phi_j\rangle\langle j,0| + |j,1\rangle|\Omega_j\rangle\langle j,1|\right] \tag{39}$$

for some states $|\Omega_j\rangle$. Because $|\lambda\rangle$ is orthogonal to $|j,1\rangle$ (as the initial state has the qubit initialised as $|0\rangle$),

$$T|\lambda\rangle = \sum_{j=1}^{M} |j,0\rangle|\phi_j\rangle\langle j,0|\lambda\rangle, \tag{40}$$

giving

$$T^\dagger S T|\lambda\rangle = \sum_{j,k=1}^{M} \big[\langle j,0|\phi_k\rangle\langle\phi_j|k,0\rangle|j,0\rangle\langle k,0|\lambda\rangle + \langle j,1|\phi_k\rangle\langle\Omega_j|k,0\rangle|j,1\rangle\langle k,0|\lambda\rangle\big]. \tag{41}$$

To obtain Eq. (38), we simply need the second term above to vanish. This can be ensured by taking $|\Omega_j\rangle = |\Omega,1\rangle$ for any state $|\Omega\rangle$. Note that it is important to properly apply the operator V to states where the ancilla qubit is in the state $|1\rangle$, since although the individual $T|\lambda\rangle$ have the ancilla in the $|0\rangle$ state, the eigenstates $|\mu_\pm^\lambda\rangle$ of V have a component of $ST|\lambda\rangle$, and therefore have a component with the ancilla in the $|1\rangle$ state.

For the final use of T, we wish to map the state $|j,0\rangle|\phi_j\rangle$ to $|j,0\rangle$ for each $j \in \{1, 2, \dots, M\}$. In general, this can only be carried out approximately, because the final state will not be exactly a superposition of states of the form $|j,0\rangle|\phi_j\rangle$. First, if the ancilla qubit is in the state $|1\rangle$, this may be regarded as a failure, because the ideal final state has the ancilla in the state $|0\rangle$. Otherwise, we perform inverse state preparation conditional on the index j. Starting from $|j,0\rangle|\phi_j\rangle$, the second register should ideally be mapped to the initial state used for the state preparation; any other state can be regarded as failure. In general, the total failure probability is proportional to the error in the inverse state preparation procedure.

5 Improved state preparation

To further improve our simulations, especially in the non-sparse case, we consider state preparation techniques based on amplitude amplification [22, 23, 24].

Our techniques draw from work on black-box state preparation. In that problem, we are given an oracle O_ψ acting as

$$O_\psi|j\rangle|z\rangle = |j\rangle|z \oplus \psi_j\rangle \tag{42}$$

for some quantum state $|\psi\rangle = \sum_{j=1}^M \psi_j|j\rangle$; the goal is to prepare a copy of $|\psi\rangle$. Grover showed how to prepare a black-box quantum state with only $O(\sqrt{M})$ queries [12]. By the lower bound for search [20], preparation of an M-dimensional black-box quantum state requires $\Omega(\sqrt{M})$ queries, so this state preparation scheme is optimal.

We can use Grover's technique to prepare the states from Eq. (25) and thereby implement the quantum walk. This is favorable for large D, in which case state preparation based on Lemma 4 alone is suboptimal. By combining the approach of Lemma 4 with amplitude amplification, we improve on both these approaches, as follows.

Lemma 6. *Let $\Lambda_1 \geq \|H\|_1$, $\Lambda_{\max} \geq \|H\|_{\max}$, and $\varepsilon \in (0,1]$. Then*

$$|\phi_j\rangle = \sqrt{\frac{\varepsilon}{\Lambda_1}}\sum_{k=1}^M \sqrt{H_{jk}^*}|k\rangle|0\rangle + \sqrt{1-\frac{\varepsilon\sigma_j}{\Lambda_1}}|\zeta_j\rangle|1\rangle \tag{43}$$

can be approximately prepared using

$$O\left(\sqrt{\frac{\varepsilon\Lambda_{\max}D}{\Lambda_1}}+1\right) \tag{44}$$

queries to O_H and O_F. The approximation has relative error in the weighting of the first term of $O(\varepsilon)$.

Proof. As in the proof of Lemma 4, we can prepare

$$|\phi_j^b\rangle := \frac{1}{\sqrt{D}}\sum_{k\in F_j}|k\rangle\left[\sqrt{\frac{H_{jk}^*}{X}}|0\rangle + \sqrt{1-\frac{|H_{jk}|}{X}}|1\rangle\right] \tag{45}$$

using one query to O_F and two queries to O_H, where X is a real number satisfying $X \geq \Lambda_{\max} \geq \|H\|_{\max}$. Let B_j denote a unitary operation that prepares $|\phi_j^b\rangle$ from $|0\rangle|0\rangle$.

We now use a form of amplitude amplification similar to that introduced by Grover [12]. We define two reflection operators. The first reflects about the $|0\rangle$ state for the ancilla qubit,

$$R^f := \mathbb{1}\otimes(\mathbb{1}-2|0\rangle\langle 0|), \tag{46}$$

and the second reflects about the state $|\phi_j^b\rangle$,

$$R_j^b := 2|\phi_j^b\rangle\langle\phi_j^b| - \mathbb{1}. \tag{47}$$

The latter reflection can be performed by applying $B_j^\dagger$, reflecting about $|0\rangle|0\rangle$, and then applying B_j. Using an appropriate number of these reflections, we could obtain a final state close to

$$|\phi_j^f\rangle := \frac{1}{\sqrt{\sigma_j}}\sum_{k=1}^M \sqrt{H_{jk}^*}|k\rangle|0\rangle. \tag{48}$$

However, the key point is that we do not rotate all the way towards this state, but instead prepare

$$|\phi_j\rangle = \sqrt{\frac{\varepsilon\sigma_j}{\Lambda_1}}|\phi_j^f\rangle + \mathcal{N}_j \sum_{k\in F_j} \sqrt{1-\frac{|H_{jk}|}{X_j}}|k\rangle|1\rangle, \tag{49}$$

where $\mathcal{N}_j$ is a normalisation constant. This expression corresponds to the definition of $|\phi_j\rangle$ with

$$|\zeta_j\rangle \propto \sum_{k\in F_j} \sqrt{1-\frac{|H_{jk}|}{X_j}}|k\rangle. \tag{50}$$

According to Eq. (15), the normalised Hamiltonian corresponding to the discrete-time quantum walk defined by these states is

$$\tilde{H} = \frac{\varepsilon H}{\Lambda_1}. \tag{51}$$

We prepare a state close to $|\phi_j\rangle$ using amplitude amplification. Let

$$|\phi_j(r)\rangle := (R_j^b R^f)^r |\phi_j^b\rangle \tag{52}$$

denote the state as a function of the number of steps, r. We have

$$|\phi_j(r)\rangle = \sin[(2r+1)\theta_j]|\phi_j^f\rangle + \mathcal{N}_j \cos[(2r+1)\theta_j] \sum_{k\in F_j} \sqrt{1-\frac{|H_{jk}|}{X}}|k\rangle|1\rangle, \tag{53}$$

where

$$\sin\theta_j = \langle\phi_j^f|\phi_j^b\rangle = \sqrt{\frac{\sigma_j}{DX}}. \tag{54}$$

By Eqs. (48) and (49), $\langle\phi_j^f|\phi_j\rangle = \sqrt{\varepsilon\sigma_j/\Lambda_1}$, so the value of r that gives the desired outcome is

$$\begin{aligned} r_j^{\text{opt}} &:= \frac{1}{2}\left(\frac{1}{\theta_j}\arcsin\sqrt{\frac{\varepsilon\sigma_j}{\Lambda_1}} - 1\right) \\ &= \frac{1}{2}\left(\frac{\arcsin\sqrt{\frac{\varepsilon\sigma_j}{\Lambda_1}}}{\arcsin\sqrt{\frac{\sigma_j}{DX}}} - 1\right). \end{aligned} \tag{55}$$

There are several reasons why we cannot perform exactly r_j^{opt} queries. This value may not be an integer, and it is j-dependent. Furthermore, since σ_j is not known in general, the exact value of r_j^{opt} is unknown. However, if ε is small, then the arcsin function can be linearised, and we can take

$$r \approx \frac{1}{2}\sqrt{\frac{\varepsilon XD}{\Lambda_1}} - \frac{1}{2}. \tag{56}$$

Specifically, we choose

$$r = \left\lceil \frac{1}{2}\sqrt{\frac{\varepsilon\Lambda_{\max}D}{\Lambda_1}} - \frac{1}{2} \right\rceil \tag{57}$$

and

$$X = (2r+1)^2\frac{\Lambda_1}{\varepsilon D}. \tag{58}$$

Since the number of queries per step is $O(1)$, the total number of queries is $O(\sqrt{\varepsilon\Lambda_{\max}D/\Lambda_1}+1)$ as claimed.

Now we analyse the error incurred due to imperfect state preparation. First consider the deviation of r from $r_j^{\rm opt}$. This deviation results from linearisation of both the numerator and denominator of Eq. (55). The argument of the arcsin function in the denominator is smaller than that in the numerator, so to determine the scaling of the error, it suffices to consider the error in the linearisation of the numerator. The relative error is thus

$$\frac{|r_j^{\rm opt}-r|}{r}=O(\varepsilon\sigma_j/\Lambda_1)\le O(\varepsilon) \tag{59}$$

since $\sigma_j\le\Lambda_1$ for all j.

The effect of the difference between r and $r_j^{\rm opt}$ is a slightly incorrect weighting of $|\phi_j^f\rangle$ in the final state:

$$\begin{aligned}\langle\phi_j^f|\phi_j(r)\rangle&=\sin[(2r+1)\theta_j]\\&=\sqrt{\frac{\varepsilon\sigma_j}{\Lambda_1}}(1+x_j)\end{aligned} \tag{60}$$

where

$$\begin{aligned}x_j&:=\sqrt{\frac{\Lambda_1}{\varepsilon\sigma_j}}\sin[(2r+1)\theta_j]-1\\&=\sqrt{\frac{\Lambda_1}{\varepsilon\sigma_j}}\left\{\sin[(2r_j^{\rm opt}+1)\theta_j]+2\theta_j\cos[(2r_j^{\rm int}+1)\theta_j](r-r_j^{\rm opt})\right\}-1\\&=\sqrt{\frac{\Lambda_1}{\varepsilon\sigma_j}}2\theta_j\cos[(2r_j^{\rm int}+1)\theta_j](r-r_j^{\rm opt})\end{aligned} \tag{61}$$

for some $r_j^{\rm int}\in[r,r_j^{\rm opt}]$, where in the second line we have used Taylor's theorem. Hence

$$\begin{aligned}|x_j|&\le\sqrt{\frac{\Lambda_1}{\varepsilon\sigma_j}}2\theta_j|r-r_j^{\rm opt}|\\&\le\sqrt{\frac{\Lambda_1}{\varepsilon\sigma_j}}\pi\sqrt{\frac{\sigma_j}{DX}}|r-r_j^{\rm opt}|\\&=\frac{\pi r}{2r+1}\frac{|r-r_j^{\rm opt}|}{r}\\&=O(\varepsilon).\end{aligned} \tag{62}$$

In the next to last line, we have used Eq. (58). Hence the error in the weighting of the first term in Eq. (43) is $O(\varepsilon)$, as claimed. □

The state preparation scheme described in Lemma 6 introduces additional error in the Hamiltonian simulation, but this error is well bounded. In particular, we have the following.
Lemma 7. *The error in the state preparation scheme of Lemma 6 results in an error in the Hamiltonian simulation described in Sec. 3 of* $O(\|H\|t\varepsilon)$.

Proof. The actual Hamiltonian being simulated, $\tilde{H}'$, has matrix elements

$$\begin{aligned}
\tilde{H}'_{jk} &= \langle j,0|\langle\phi_j(r)|S|k,0\rangle|\phi_k(r)\rangle \\
&= \langle\phi_j(r)|k,0\rangle\langle j,0|\phi_k(r)\rangle \\
&= \sin[(2r+1)\theta_j]\sin[(2r+1)\theta_k]\frac{H_{jk}}{\sqrt{\sigma_j\sigma_k}} \\
&= \frac{\varepsilon H_{jk}}{\Lambda_1}(1+x_j)(1+x_k). \qquad (63)
\end{aligned}$$

Defining a diagonal matrix $\mathbf{x} := \mathrm{diag}(x_1, x_2, \ldots, x_M)$, the error in the Hamiltonian is

$$\begin{aligned}
\|\tilde{H}' - \tilde{H}\| &= \left\|\frac{\varepsilon}{\Lambda_1}(\mathbf{x}H + H\mathbf{x} + \mathbf{x}H\mathbf{x})\right\| \\
&\le \frac{\varepsilon\|H\|}{\Lambda_1}(2x_{\max} + x_{\max}^2) \\
&= \frac{\varepsilon\|H\|}{\Lambda_1}O(\varepsilon), \qquad (64)
\end{aligned}$$

where $x_{\max} := \max_j |x_j|$. In evolving the Hamiltonian over time t, we multiply this by a factor of $t\Lambda_1/\varepsilon$, so the resulting error is $O(\|H\|t\varepsilon)$. □

6 Non-sparse Hamiltonians

Now we examine the overall performance of the Hamiltonian simulation algorithm with improved state preparation. Multiplying the number of steps of the quantum walk by the number of queries required to implement each step, we find the following.

Lemma 8. *Given a black-box Hamiltonian H, let $\Lambda \ge \|H\|$, $\Lambda_1 \ge \|H\|_1$, and $\Lambda_{\max} \ge \|H\|_{\max}$. Then H can be simulated for time t with error at most $\delta \in (0,1]$ using*

$$O\left(t^{3/2}\sqrt{\frac{\Lambda_{\max}D\Lambda_1\Lambda}{\delta}}\right) \qquad (65)$$

queries to O_H and O_F, provided that

$$\begin{aligned}
\Lambda t &\ge \sqrt{\delta}, \qquad (66) \\
\Lambda t &\ge \frac{\Lambda^2}{\Lambda_{\max}\Lambda_1 D}, \text{ and} \qquad (67) \\
\Lambda &\le \Lambda_1. \qquad (68)
\end{aligned}$$

The restriction (68) simply means that Λ is not unnecessarily large. Because $\|H\| \le \|H\|_1$, we can decrease any given Λ to be at most Λ_1, provided (66) still holds. This Lemma provides improved performance in cases where D is large. This may mean that $D = M$, but we continue to perform the analysis in terms of the sparseness parameter D for generality.

Proof. We take

$$\varepsilon = \frac{\Lambda_1 t}{\lceil\Lambda_1\Lambda t^2/\delta\rceil}. \qquad (69)$$

This ensures that $\varepsilon \le \delta/\Lambda t$, so $\|H\|t\varepsilon \le \delta$. The restriction (66) then ensures that $\varepsilon \le 1$. In addition, (66) and (68) ensure that $\Lambda_1\Lambda t^2/\delta \ge 1$, so the ceiling function does not affect the scaling, and

$$1/\varepsilon = O(\Lambda t/\delta). \tag{70}$$

With this value of ε, $\Lambda_1 t/\varepsilon$ is an integer, and therefore Lemma 5 shows that $O(\Lambda_1 t/\varepsilon)$ quantum walk steps suffice for the simulation. Then, using Lemma 6, the number of oracle queries for each step of the quantum walk is $O(\sqrt{\varepsilon\Lambda_{\max}D/\Lambda_1})$, unless this quantity is less than 1, in which case the state preparation proceeds without amplitude amplification.

That case does not alter the result, because the total number of queries for the simulation as given by Eq. (4) in Theorem 1 is less than Eq. (65) given the restrictions in Lemma 8. This can be shown as follows. First, assuming $\sqrt{\varepsilon\Lambda_{\max}D/\Lambda_1} = O(1)$, we have

$$\begin{aligned} D\Lambda_{\max}t &= \frac{\sqrt{\delta}D\Lambda_{\max}t}{\sqrt{\delta}} \\ &= O\left(\frac{\sqrt{\varepsilon\Lambda t}D\Lambda_{\max}t}{\sqrt{\delta}}\right) \\ &\le O\left(t^{3/2}\sqrt{\frac{\Lambda_{\max}D\Lambda_1\Lambda}{\delta}}\right). \end{aligned} \tag{71}$$

In the second line we have used Eq. (70), and in the third line we have used the condition that $\sqrt{\varepsilon\Lambda_{\max}D/\Lambda_1} = O(1)$. Next,

$$\frac{\Lambda t}{\sqrt{\delta}} \le t^{3/2}\sqrt{\frac{\Lambda_{\max}D\Lambda_1\Lambda}{\delta}} \tag{72}$$

using the restriction (67). Finally, combining Eqs. (66) and (67) shows that the number of queries in Eq. (65) is at least constant. Thus we find that Eq. (4) is less than Eq. (65), as required.

For the case where state preparation proceeds via amplitude amplification, we multiply the number of steps of the quantum walk (from Lemma 5) by the number of oracle calls for each step (from Lemma 6). Thus the total number of queries is

$$O\left(t\sqrt{\frac{\Lambda_{\max}D\Lambda_1}{\varepsilon}}\right) \le O\left(t^{3/2}\sqrt{\frac{\Lambda_{\max}D\Lambda_1\Lambda}{\delta}}\right). \tag{73}$$

where we have used Eq. (70).

Finally, we consider the error in the simulation. Because $\varepsilon \le \delta/\Lambda t$, Lemma 7 implies that the error due to imperfect state preparation is $O(\delta)$. Using Lemma 5, the error due to the quantum walk simulation is $O(\Lambda^2\varepsilon^2/\Lambda_1^2)$. Using $\varepsilon \le \delta/\Lambda t$ and $\sqrt{\delta} \le \Lambda t$, this contribution to the error is also $O(\delta)$.

The statement of the Lemma requires that the error is less than δ, rather than $O(\delta)$. However, any multiplying factor for the error can be absorbed into the big-O notation of Eq. (65). $\square$

We are interested in improving the scaling with D beyond the linear scaling in Theorem 1. The number of queries in Lemma 8 contains $\sqrt{D}$, but also depends on several other quantities. For simplicity, in this discussion we assume that Λ can be replaced with $\|H\|$, and so forth. In the worst case we can have $\|H\|_{\max} \propto \|H\|$ and $\|H\|_1 \propto \|H\|\sqrt{D}$. This would yield overall scaling of $O((\|H\|t)^{3/2}D^{3/4}/\sqrt{\delta})$. However, it should be noted that this worst case arises from two different factors.

1. To have $\|H\|_{\max} \propto \|H\|$, the distribution of the magnitudes of the matrix elements should have a sharp peak, so there is a row with most of the weight on one of the elements.
2. To have $\|H\|_1 \propto \|H\|\sqrt{D}$, the magnitudes of the matrix elements should be relatively evenly distributed.

If we could ensure that all the nonzero elements had magnitudes within some constant factor (so there is no sharp peak), then we would obtain $\|H\|_{\max} \propto \|H\|/\sqrt{D}$, giving a scaling of $O((\|H\|t)^{3/2}\sqrt{D/\delta})$.

7 Breaking up the Hamiltonian

We now consider how the simulation can be improved by breaking up the Hamiltonian into a sum of terms. Although the matrix elements of the Hamiltonian may differ over a wide range, the Hamiltonian can be broken up into terms, each of which has matrix elements of similar magnitude. By combining the evolution under these Hamiltonians via a Lie-Trotter-Suzuki formula, we can expect scaling close to $O((\|H\|t)^{3/2}\sqrt{D/\delta})$. The only problem is that the spectral norms of the individual Hamiltonians may be large. First we present a derivation showing that, provided the norms of the individual terms are not large, then the expected scaling is obtained. Next we present numerical results showing that typical spectral norms are small, although there are pathological cases with large norms. Finally, we present a general method using a number of queries roughly proportional to $D^{2/3}$ even when the spectral norms are large.

7.1 Small norms

In order to present our result, we define the function “break”, which quantifies how much the norm can be increased by breaking up the Hamiltonian into parts. Let

$$\mathrm{break}(H) := \max_{a,b\in\mathbb{R}} \|H^{ab}\|/\|H\|, \tag{74}$$

where the matrix H^{ab} is defined by

$$H^{ab}_{jk} := \begin{cases} H_{jk} & \text{if } a < |H_{jk}| \le b, \\ 0 & \text{if } |H_{jk}| \le a \text{ or } b < |H_{jk}|. \end{cases} \tag{75}$$

In this subsection we suppose that break(H) is small. We present numerical evidence in Sec. 7.2 that break$(H) \le 1.5$ in most cases. From the definition, it is clear that break$(H) \ge 1$. In addition, because $\|H^{ab}\| \le \|H^{ab}\|_1 \le \|H\|_1 \le \|H\|\sqrt{D}$, we have break$(H) \le \sqrt{D}$. If break$(H)$ can be upper bounded by a constant, we obtain a simulation with scaling close to $\sqrt{D}$.

Theorem 9. *Let $\Lambda \geq \|H\|$ and $\Upsilon \in [\operatorname{break}(H), \sqrt{D}]$. The evolution under the Hamiltonian H for time t can be simulated with error at most $\delta \in (0,1]$ using*

$$O\left(\sqrt{\Upsilon D/\delta}(\log D)^{7/4}(\Lambda t)^{3/2}\right) \tag{76}$$

queries to O_H and O_F, provided $\delta D > \Lambda t > \sqrt{\delta}$.

Proof. We split the Hamiltonian into L terms, each with nonzero elements of approximately the same magnitude:

$$H = \sum_{\ell=1}^{L} H_\ell. \tag{77}$$

We take the Hamiltonians H_ℓ to include elements with decreasing magnitudes: H_1 contains elements with the largest magnitudes, H_2 contains elements with the next largest magnitudes, and so forth. We denote the cutoff values A_ℓ, so $H_\ell = H^{A_\ell A_{\ell-1}}$ for $\ell < L$ and $H_L = H^{0 A_{L-1}}$. We take $A_0 = \Lambda$, $A_L = \Lambda/\sqrt{D}$, and $A_0 > A_1 > \cdots > A_L$. In examining H_ℓ, let $\Lambda^{(\ell)}$ denote an upper bound on $\|H_\ell\|$, $\Lambda_1^{(\ell)}$ an upper bound on $\|H_\ell\|_1$, $\Lambda_{\max}^{(\ell)}$ an upper bound on $\|H_\ell\|_{\max}$, and τ_ℓ the time interval for simulation of H_ℓ. We also let δ_ℓ denote the error allowed for simulating H_ℓ over a time step of length τ_ℓ.

Because $|[H_\ell]_{jk}| \leq A_{\ell-1}$, we can take $\Lambda_{\max}^{(\ell)} = A_{\ell-1}$. To choose a value of $\Lambda_1^{(\ell)}$ for $\ell < L$, we use

$$\begin{aligned} \|H_\ell\|_1 &\leq \max_j \sum_{k=1}^{M} |[H_\ell]_{jk}|^2/A_\ell \\ &\leq \max_j \sum_{k=1}^{M} |H_{jk}|^2/A_\ell \\ &\leq \|H\|^2/A_\ell. \end{aligned} \tag{78}$$

Therefore we can take $\Lambda_1^{(\ell)} = \Lambda^2/A_\ell$ for $\ell < L$. For $\ell = L$, we have

$$\|H_\ell\|_1 \leq \|H\|_1 \leq \|H\|\sqrt{D}. \tag{79}$$

Since we set $A_L = \Lambda/\sqrt{D}$, we have $\Lambda_1^{(\ell)} = \Lambda^2/A_\ell$ for $\ell = L$ as well. This is why we define a value for A_L, even though it is not used to bound matrix elements.

The success of the simulation depends crucially on the scaling of the norms $\|H_\ell\|$. By assumption, $\Upsilon \geq \operatorname{break}(H)$, so $\|H_\ell\| \leq \Upsilon\|H\|$. Because $\|H_\ell\| \leq \|H_\ell\|_1$, we can take $\Lambda^{(\ell)} = \min\{\Upsilon\Lambda, \Lambda^2/A_\ell\}$.

Using Lemma 8, the number of queries to simulate H_ℓ for time τ_ℓ is

$$O\left(\tau_\ell^{3/2}\sqrt{\frac{\Lambda_{\max}^{(\ell)} D \Lambda_1^{(\ell)} \Lambda^{(\ell)}}{\delta_\ell}}\right) \leq O\left(\Lambda\tau_\ell^{3/2}\sqrt{\frac{D\Upsilon\Lambda A_{\ell-1}}{\delta_\ell A_\ell}}\right). \tag{80}$$

In this proof we take τ_ℓ and δ_ℓ to be independent of ℓ. To ensure that the number of queries is independent of ℓ (so no one term dominates the scaling), we take constant ratios $A_{\ell-1}/A_\ell$. To satisfy $A_0 = \Lambda$ and $A_L = \Lambda/\sqrt{D}$, we can take $A_\ell = \Lambda D^{-\ell/2L}$. Then $\Lambda_{\max}^{(\ell)} =$

$A_{\ell-1} = \Lambda D^{(1-\ell)/2L}$, $\Lambda_1^{(\ell)} = \Lambda^2/A_\ell = \Lambda D^{\ell/2L}$, and the ratio between successive cutoffs is $A_{\ell-1}/A_\ell = D^{1/2L}$.

Next we ensure that the conditions of Lemma 8 hold. Condition (68) follows immediately from $\Lambda^{(\ell)} = \min\{\Upsilon\Lambda, \Lambda^2/A_\ell\}$. To satisfy conditions (66) and (67), τ_ℓ cannot be too small, but it must be small enough that the Trotter error is $O(\delta)$. To achieve this, we choose τ_ℓ to satisfy

$$\tau_\ell \geq \max\left\{\frac{\delta}{L^{3/2}\Lambda^2 t}, \frac{\Upsilon}{\Lambda D^{1+1/2L}}\right\}. \tag{81}$$

In addition, to apply the Trotter formula, t/τ_ℓ must be an *even* integer. Thus we take

$$\tau_\ell = \frac{t}{2\left\lfloor \min\left\{\frac{L^{3/2}\Lambda^2 t^2}{2\delta}, \frac{\Lambda D^{1+1/2L} t}{2\Upsilon}\right\}\right\rfloor}. \tag{82}$$

For this expression to be well-defined, the denominator must be nonzero. For the first term of the minimum, we find that

$$\frac{L^{3/2}\Lambda^2 t^2}{2\delta} > \frac{L^{3/2}}{2} > 1. \tag{83}$$

The first inequality uses the condition $\Lambda t > \sqrt{\delta}$ and the second uses $L \geq 2$ (since otherwise we are not breaking up the Hamiltonian at all). For the second term to be at least 1, we require

$$\Lambda D t \geq 2\Upsilon D^{-1/2L}. \tag{84}$$

If this does not hold, then we perform the simulation with Theorem 1 instead of Lemma 8. Since $\Lambda \geq \|H\|_{\max}$, we can take $\Lambda_{\max} = \Lambda$. The condition $\delta D > \Lambda t > \sqrt{\delta}$ implies that Eq. (4) is $O(D\Lambda t)$. Provided Eq. (84) is violated, we find that we can simulate the Hamiltonian with $O(D^{1/2L})$ queries. We will take $L \propto \log D$, so the simulation uses $O(1)$ queries, which is no more than Eq. (76). Thus, for the remainder of this proof, we assume that Eq. (84) holds, so Eq. (82) is well-defined.

Using Eq. (82), we find that Eq. (81) is satisfied, and

$$\Lambda^{(\ell)}\tau_\ell \geq \Lambda\tau_\ell \geq \Lambda\sqrt{\tau_\ell}\sqrt{\frac{\delta}{L^{3/2}\Lambda^2 t}} = \sqrt{\frac{\delta\tau_\ell}{L^{3/2}t}}. \tag{85}$$

The first inequality uses $\Lambda^{(\ell)} \geq \Lambda$ and the second uses Eq. (81). Taking

$$\delta_\ell = \frac{\delta\tau_\ell}{L^{3/2}t}, \tag{86}$$

we obtain $\Lambda^{(\ell)}\tau_\ell \geq \sqrt{\delta_\ell}$, so Eq. (66) is satisfied. Equation (67) follows from

$$\Lambda^{(\ell)}\tau_\ell \geq \Lambda^{(\ell)}\Upsilon/\Lambda D^{1+1/2L} \geq (\Lambda^{(\ell)})^2/\Lambda_{\max}^{(\ell)}\Lambda_1^{(\ell)} D. \tag{87}$$

Here the first inequality holds due to the second term of the maximum in Eq. (81).

Now we use a Kth order Lie-Trotter-Suzuki integrator to combine the simulations of the H_ℓ into a simulation of H. The Strang splitting formula [26] corresponds to $K = 1$; larger values of K correspond to higher-order Lie-Trotter-Suzuki formulae. In this proof we simply take $K = 1$; in Section 7.3 we will consider the case $K = 2$. The simulation resulting from a Kth order integrator is approximate, introducing error [27, 28, 9]

$$O\left(\left[2L5^{K-1}\tau \max_{\ell} \Lambda^{(\ell)}\right]^{2K+1} \frac{t}{\tau}\right). \tag{88}$$

Here τ is the time interval over which the integrator is repeated. That is, the time is broken up into t/τ intervals, and the same integrator is used on each of those intervals. By Eq. (A2) of Ref. [29], $\tau_\ell \geq \tau \times (3/2)3^{-K}$. Thus, for a fixed value of K, $\tau = O(\tau_\ell)$. Also, the number of queries is increased by a factor of $5^K L$ due to the number of terms in the integrator.

To bound the error, we need to take account of the error due to the individual simulations and the error due to the Trotter formula. There are $O(Lt/\tau)$ terms in the Trotter formula for $K = 1$, so the total error in performing the individual simulations (neglecting only the error introduced by the Trotter formula) is $O(\delta_\ell Lt/\tau)$. Because we take $\delta_\ell = \delta\tau_\ell/L^{3/2}t$, the total error due to the simulations is $O(\delta/L^{1/2})$, which is $O(\delta)$.

With $K = 1$ and $L \propto \log D$, the Trotter error from Eq. (88) is $O(L^3\Lambda^3\tau_\ell^2 t)$. If $\delta/L^{3/2}\Lambda t \geq \Upsilon/D^{1-1/2L}$, so that Eq. (82) gives $\tau_\ell = O(\delta/L^{3/2}\Lambda^2 t)$, then this Trotter error is $O(\delta^2/\Lambda t)$, which is $O(\delta)$ because $\Lambda t > \delta$. Alternatively, if $\delta/L^{3/2}\Lambda t < \Upsilon/D^{1+1/2L}$, then the Trotter error is $O(\Lambda t L^3 \Upsilon^2/D^{2+1/L})$, which is $O(\delta L^3\Upsilon^2/D^{1+1/L})$. With $L \propto \log D$ and $\Upsilon \leq \sqrt{D}$, the Trotter error is $O(\delta)$.

The total number of queries is given by (80) multiplied by Lt/τ, so we obtain a simulation using

$$O\left(L^{7/4}(\Lambda t)^{3/2} D^{1/2+1/(4L)}\sqrt{\Upsilon/\delta}\right) \tag{89}$$

queries. Now taking $L \propto \log D$, $D^{1/L} = O(1)$, so the number of queries is as given in Eq. (76). The condition $\delta D > \Lambda t > \sqrt{\delta}$ ensures that this is at least 1. □

This theorem holds regardless of whether the norms are small. In the worst case we can have $\Upsilon = \sqrt{D}$, in which case the $D^{3/4}$ scaling is again obtained (as at the end of Sec. 6). On the other hand, if breaking up the Hamiltonian does not significantly increase the norm, then we obtain $\sqrt{D}$ scaling.

7.2 Norms of components

Now we present numerical results suggesting that for typical matrices, the spectral norms of the components are small, and break(H) can be upper bounded by a constant. If we consider general Hamiltonians, then the norms of the components are almost always smaller than the norm of the original Hamiltonian. (In this subsection, we use "norm" to mean the spectral norm.) We tested general Hamiltonians by generating random Hermitian matrices with normally distributed elements. In no case was break(H) more than 1.2, as shown in Fig. 1. For large dimension, break(H) approached 1.

We expect larger norms for the components when breaking up a Hamiltonian derived from a unitary matrix as discussed in Sec. 8 below (see Eq. (123)). This is because unitaries can have a large difference between the spectral norm and the 1-norm, and the spectral norms of the individual components are bounded by the 1-norm of H. To test this class of Hamiltonians, we generated random unitaries according to the Haar measure. The values of break(H) were

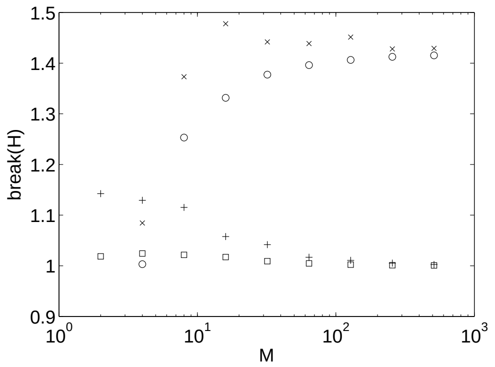

Fig. 1. The function break(H) for random Hamiltonians. The plusses and squares are the maximum and mean values, respectively, obtained for 100 randomly generated Hermitian matrices. The crosses and circles are the maximum and mean values, respectively, for sets of 100 Hamiltonians composed of random unitaries.

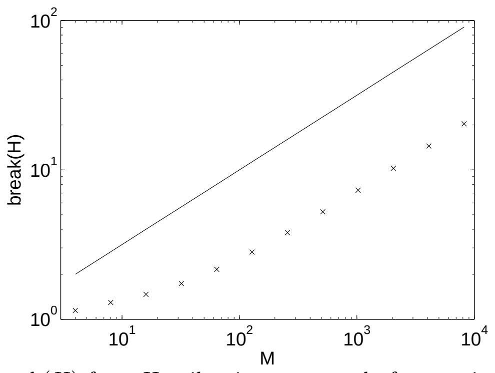

Fig. 2. The function break(H) for a Hamiltonian composed of a matrix that has been produced by perturbing a quantum Fourier transform. The solid line is $\sqrt{M}$ for comparison.

larger than for random Hamiltonians, but were still no larger than 1.5, as also shown in Fig. 1.

One way to generate matrices that do have components with large norms is to perturb the quantum Fourier transform. We considered increasing the magnitude of the elements with positive real part by 0.01% and decreasing the rest by 0.01%. The value of break(H) (with H constructed from this matrix as in Eq. (123)) was then proportional to $\sqrt{M}$ (see Fig. 2).

Although this example yields large norms for one splitting, the norms of the components are well-behaved with respect to other splittings. For example, a different threshold could be used, or we could introduce a smooth transition between the two components (i.e., for values in some transition region, part of the matrix element could go to one component and part to the other). However, we suspect that for any particular splitting, one can find examples that result in large norms for that splitting.

7.3 Large norms

In this subsection we establish the improved simulation described in Theorem 2, without relying on the assumption that the spectral norms of components remain small. We begin with an intuitive description of the method before giving the proof.

We again break the Hamiltonian into components H_ℓ according to the magnitudes of the matrix elements. In this case, the best available upper bound on the spectral norms of the components is Λ^2/A_ℓ. Using Lemma 8, the number of queries to simulate each component involves a ratio $A_{\ell-1}/A_\ell^2$, in contrast to the corresponding ratio $A_{\ell-1}/A_\ell$ when the norms are assumed to be small (compare Eqs. (80) and (99)). As a result, the cutoff values should be chosen to make $A_{\ell-1}/A_\ell^2$ constant, rather than to make $A_{\ell-1}/A_\ell$ constant.

In addition, the simulation of H_L should not be performed via Lemma 8, because that would result in an overall scaling no better than that provided by Lemma 8. Instead, we use Theorem 1 to simulate H_L. By comparing the number of queries required to simulate H_L and H_{L-1}, this means that (ignoring scaling in quantities other than D) we should have

$$\sqrt{\frac{DA_{L-2}}{A_{L-1}^2}} \approx DA_{L-1}. \tag{90}$$

Here the expression on the left comes from using Lemma 8 to simulate H_{L-1}, and the expression on the right comes from using Theorem 1 for H_L. This expression means that $A_{L-2} \approx DA_{L-1}^4$. The restriction $A_{L-2} > A_{L-1}$ then means that $A_{L-1} \gtrapprox D^{-1/3}$. Therefore the number of queries is minimised for $A_{L-1} \approx A_{L-2} \approx D^{-1/3}$. Then $A_{\ell-1}/A_\ell^2 \approx D^{1/3}$, and the number of queries is roughly $D^{2/3}$.

To ensure that A_0 is independent of D, we must modify the above choices slightly. We choose a small constant ξ, and take $A_{L-1} \propto D^{\xi-1/3}$ and $A_{\ell-1}/A_\ell^2 \propto D^{1/3+2\xi}$. Iterating gives $A_{L-2} \propto D^{4\xi-1/3}$, $A_{L-3} \propto D^{10\xi-1/3}$, and so forth. The sequence needs to give A_0 independent of D, but because the coefficient of ξ increases exponentially, L need vary only logarithmically in ξ.

This approach results in a number of queries to simulate each H_ℓ proportional to $D^{2/3+\xi}$. By choosing $\xi \propto 1/\log D$, D^ξ is $O(1)$. In addition, L varies doubly logarithmically in D, which gives a double-logarithmic factor in the overall scaling in Theorem 2. In the proof below, scaling in all quantities is considered, so it is convenient to define a quantity $\aleph$ that includes D together with the other quantities we have omitted here. The scaling is then given in terms of $\aleph$, rather than explicitly in terms of D.

In order to show the result rigorously, we need to carefully choose the time intervals, because these are lower bounded by the conditions (66) and (67) of Lemma 8, and upper bounded by the need to ensure that the error in the Trotter formula is sufficiently small. This is challenging, because the bounds for the different components differ significantly. The bounds on the Trotter error for H_ℓ decrease with ℓ, so the lower bound on the time interval for H_1 is greater than the upper bound on the time interval for H_L. Thus it is not possible to combine these elements in the same Trotter formula while adequately bounding the error. To overcome this problem, we use nested Trotter formulae. We use a higher-order Lie-Trotter-Suzuki formula for H_2 through H_L, in order to obtain sufficiently small error despite the large upper bound on the norm of H_L. Then we use the Strang splitting to combine this product formula with H_1.

Proof of Theorem 2. The Hamiltonian H is again broken into L pieces as in Eq. (77), again with $H_\ell = H^{A_\ell A_{\ell-1}}$ for $\ell < L$ and $H_L = H^{0A_{L-1}}$. For $\ell < L$, the norms are upper bounded as

$$\|H_\ell\| \le \|H_\ell\|_1 \le \|H\|^2/A_\ell. \tag{91}$$

The spectral norm of H_L can be bounded more strongly:

$$\begin{aligned}\|H_L\| &= \left\| H - \sum_{\ell=1}^{L-1} H_\ell \right\| \\ &\le \|H\| + \|H\|^2/A_{L-1}.\end{aligned} \tag{92}$$

Therefore, we can take $\Lambda^{(\ell)} = \Lambda^2/A_\ell$ for $\ell < L$ and $\Lambda^{(L)} = \Lambda + \Lambda^2/A_{L-1}$. We can also take $\Lambda_1^{(\ell)} = \Lambda^2/A_\ell$ for $\ell < L$, but for $\ell = L$ the best available bound gives $\Lambda_1^{(L)} = \Lambda\sqrt{D}$. We have $\Lambda_{\max}^{(\ell)} = A_{\ell-1}$.

For $k \ge 1$, let

$$A_{L-k} = \Lambda/\aleph^{1/3-(3\times 2^{k-1}-2)\xi} \tag{93}$$

where

$$\aleph := \frac{\delta D}{L\Lambda t}, \tag{94}$$

$$\xi := \frac{1}{6(3\times 2^{L-2}-1)}. \tag{95}$$

With this choice, $A_0 = \Lambda$ and $A_{L-1} = \Lambda/\aleph^{1/3-\xi}$; unlike in Section 7.1, the ratio between successive cutoffs is not constant. We break H into

$$L := \left\lceil \log_2 \left[\frac{2}{9} \log\left(\frac{\delta D}{\Lambda t}\right) + \frac{4}{3} \right] \right\rceil \tag{96}$$

pieces.

Note that if $\delta D/\Lambda t \le e^3$, then $L = 1$, and we do not break up the Hamiltonian; we simply simulate H using Theorem 1. Recall that by assumption, $\delta D > \Lambda t > \sqrt{\delta}$. Therefore, $D\Lambda t > \Lambda t/\sqrt{\delta} > 1$. Since $\Lambda \ge \|H\|_{\max}$, this shows that Theorem 1 uses $O(D\Lambda t)$ queries. Assuming $\delta D/\Lambda t \le e^3$, we have

$$\begin{aligned}D\Lambda t &\le D\Lambda t \left(\frac{e^3 \Lambda t}{D\delta}\right)^{1/3} \\ &= eD^{2/3}\frac{(\Lambda t)^{4/3}}{\delta^{1/3}} \\ &= O(D^{2/3}[(\log\log D)\Lambda t]^{4/3}\delta^{-1/3}).\end{aligned} \tag{97}$$

This establishes Theorem 2 when $\delta D/\Lambda t \le e^3$. In the remainder of the proof, we assume that $\delta D/\Lambda t > e^3$, so $L > 1$. It can also be shown that this implies $\aleph > 1$.

For $\ell < L$, Lemma 8 lets us simulate the Hamiltonian H_ℓ for time τ_ℓ using

$$O\left(\tau_\ell^{3/2}\sqrt{\frac{\Lambda_{\max}^{(\ell)} D \Lambda_1^{(\ell)} \Lambda^{(\ell)}}{\delta_\ell}}\right) \tag{98}$$

queries. Conditions (66) and (67) of Lemma 8 are satisfied provided τ_ℓ is sufficiently small; we verify this below when choosing τ_ℓ in the analysis of the Trotter error. The condition (68) is trivial for $\ell < L$. We set $\delta_\ell = \delta\tau_\ell/Lt$ to ensure that the contribution to the error from the simulations is $O(\delta)$. Thus for $\ell < L$, the number of queries used to simulate H_ℓ for time τ_ℓ is

$$O\left(\Lambda^2 \tau_\ell \sqrt{\frac{LDA_{\ell-1}t}{\delta A_\ell^2}}\right). \tag{99}$$

A simple calculation shows that

$$A_{\ell-1}/A_\ell^2 = \aleph^{1/3+2\xi}/\Lambda. \tag{100}$$

Thus the query complexity of simulating H_ℓ for $\ell < L$ is

$$O\left(D^{2/3}\Lambda\tau_\ell \aleph^\xi \left(\frac{L\Lambda t}{\delta}\right)^{1/3}\right). \tag{101}$$

To simulate H_L for time τ_L, we apply Theorem 1, at a cost of

$$O\left(\frac{\Lambda^2\sqrt{Lt\tau_L}}{A_{L-1}\sqrt{\delta}} + DA_{L-1}\tau_L + 1\right) \tag{102}$$

queries. By a simple calculation,

$$DA_{L-1}\tau_L = D^{2/3}\Lambda\tau_L \aleph^\xi \left(\frac{L\Lambda t}{\delta}\right)^{1/3}, \tag{103}$$

so the query complexity of simulating H_L is also given by (101) provided the second term of Eq. (102) is dominant. We verify this after choosing τ_L below.

Now we analyze the Trotter error. We use a two-step process to combine the terms of H. First we use a Trotter formula for the two components H_1 and $\sum_{\ell=2}^L H_\ell$. Then we combine the terms of $\sum_{\ell=2}^L H_\ell$ using another Trotter formula. We do this because large time steps are needed for H_1, but its norm is small, whereas the time steps for the remaining H_ℓ can be smaller, but the norms are larger.

To combine H_1 and $\sum_{\ell=2}^L H_\ell$, the minimum time step is set by the restrictions (66) ($\Lambda^{(\ell)}\tau_\ell \geq \sqrt{\delta_\ell}$) and (67) ($\tau_\ell \geq \Lambda^{(\ell)}/\Lambda_{\max}^{(\ell)}\Lambda_1^{(\ell)}D$) for H_1. For general ℓ, using the choice $\delta_\ell = \delta\tau_\ell/Lt$, we see that these restrictions are satisfied provided

$$\tau_\ell \geq \max\left\{\frac{\delta}{L(\Lambda^{(\ell)})^2 t}, \frac{\Lambda^{(\ell)}}{\Lambda_{\max}^{(\ell)}\Lambda_1^{(\ell)}D}\right\}. \tag{104}$$

For $\ell < L$, a simple calculation shows that

$$\begin{aligned}\frac{\Lambda^{(\ell)}}{\Lambda_{\max}^{(\ell)}\Lambda_1^{(\ell)}D} &= \frac{1}{A_{\ell-1}D} \\ &= \frac{\delta}{L\Lambda^2 t}\aleph^{-2/3-(3\times 2^{L-\ell}-2)\xi} \\ &= \frac{\delta}{L(\Lambda^{(\ell)})^2 t}\aleph^{-6(2^{L-\ell}-1)\xi},\end{aligned} \tag{105}$$

where in the third line we have used

$$\frac{\Lambda^{(\ell)}}{\Lambda} = \frac{\Lambda}{A_\ell} = \aleph^{1/3-(3\times 2^{L-\ell-1}-2)\xi}. \tag{106}$$

Therefore, since $\aleph > 1$, the first term of Eq. (104) is larger than the second, and it suffices to take

$$\tau_\ell \ge \frac{\delta}{L(\Lambda^{(\ell)})^2 t} = \frac{\delta A_\ell^2}{L\Lambda^4 t}. \tag{107}$$

Since $\aleph > 1$ implies $A_\ell < A_{\ell-1}$, this lower bound decreases with increasing ℓ. Thus it suffices to ensure that τ_1 is sufficiently large. Here we use the $K = 1$ integrator, so the ratio t/τ_1 must be an even integer. We can achieve this, and ensure $\tau_1 \ge \delta A_1^2/L\Lambda^4 t$, by taking

$$\tau_1 = \frac{t}{2\left\lfloor \frac{t^2 L\Lambda^4}{2\delta A_1^2} \right\rfloor}. \tag{108}$$

This expression is finite because

$$\frac{t^2 L\Lambda^4}{2\delta A_1^2} = \frac{(\Lambda t)^2 L\aleph^{1/3+2\xi}}{2\delta} > \frac{L\aleph^{1/3+2\xi}}{2} > 1. \tag{109}$$

The equality uses Eq. (100) to compute A_1 in terms of $A_0 = \Lambda$, the first inequality uses the assumption $\Lambda t > \sqrt{\delta}$, and the last inequality uses $\aleph > 1$ and $L \ge 2$.

The norms of the two components in the Trotter formula are bounded as

$$\|H_1\| \le \|H\|^2/A_1 \le \Lambda^2/A_1, \tag{110}$$

$$\left\| \sum_{\ell=2}^{L} H_\ell \right\| \le \|H\| + \|H\|^2/A_1 \le 2\Lambda^2/A_1, \tag{111}$$

where the second line uses $A_1 \le A_0 = \Lambda$. Thus, by Eq. (88), the Trotter error for combining H_1 and $\sum_{\ell=2}^{L} H_\ell$ with a $K = 1$ integrator is

$$\begin{aligned} O\left(\frac{\tau_1^2 \Lambda^6 t}{A_1^3}\right) &= O\left(\frac{\delta^2 A_1}{L^2\Lambda^2 t}\right) \\ &\le O\left(\frac{\delta^2}{L^2\Lambda t}\right) \\ &\le O(\delta), \end{aligned} \tag{112}$$

where in the last step we have used $\delta < \Lambda t$, which follows from $\sqrt{\delta} < \Lambda t$ and $\delta \le 1$.

Next we combine the H_ℓ with $\ell > 1$, giving a simulation for time τ_1. We assume that $L \ge 3$ so there are at least two such terms to combine; then $\delta D/\Lambda t > e^{12}$. By Eq. (107) for $\ell = 2$, the conditions of Lemma 8 are satisfied if we use time intervals of at least $\delta A_2^2/L\Lambda^4 t$. However, we must choose time intervals that are compatible with the form of the integrator. In this case, we use the $K = 2$ integrator (see Section 7.1), which involves using two different intervals denoted $\tau_2^{(1)}$ and $\tau_2^{(2)}$. We use the same pair of intervals for all $\ell \ge 2$.

The integrator requires intervals of the form

$$\tau_2^{(1)} = p_2\tau_1/2\nu \tag{113}$$

$$\tau_2^{(2)} = (4p_2-1)\tau_1/2\nu \tag{114}$$

for some positive integer ν, where $p_2 := 1/(4-4^{1/3})$. Since $p_2 < 4p_2 - 1$, $\tau_2^{(2)} > \tau_2^{(1)}$, so to satisfy the conditions of Lemma 8, it suffices to ensure that $\tau_2^{(1)} \geq \delta A_2^2/L\Lambda^4 t$. We enforce this by choosing

$$\nu := \left\lfloor \frac{p_2\tau_1 L\Lambda^4 t}{2\delta A_2^2} \right\rfloor. \tag{115}$$

This is a positive integer because

$$\frac{p_2\tau_1 L\Lambda^4 t}{2\delta A_2^2} \geq \frac{p_2 A_1^2}{2A_2^2} = \frac{\aleph^{1/6+\xi}}{2(4-4^{1/3})} > 1. \tag{116}$$

The first inequality uses $\tau_1 \geq \delta A_1^2/L\Lambda^4 t$. The final inequality holds since $\delta D/\Lambda t > e^{12}$, as discussed above.

With this choice in hand, we can now verify that the second term of Eq. (102) is dominant. Henceforth we omit the superscripts on the time intervals, as they only differ by a multiplicative constant. Since $A_2 = \Lambda/\aleph^{1/4+3\xi/2}$, $\tau_L = \tau_2 = \Theta(\aleph^{1/2-3\xi}/D\Lambda)$, and the second term of Eq. (102) is

$$DA_{L-1}\tau_L = \Theta(A_{L-1}\aleph^{1/2-3\xi}/\Lambda) = \Theta(\aleph^{1/6-2\xi}). \tag{117}$$

In comparison, the first term is

$$\begin{aligned}\frac{\Lambda^2\sqrt{Lt\tau_L}}{A_{L-1}\sqrt{\delta}} &= \Theta\left(\Lambda\aleph^{1/3-\xi}\frac{1}{\sqrt{\Lambda\aleph}}\sqrt{\frac{\aleph^{1/2-3\xi}}{\Lambda}}\right) \\ &= \Theta(\aleph^{1/12-5\xi/2}),\end{aligned} \tag{118}$$

which is smaller than (117) since $\aleph > 1$. We claim that the third term of Eq. (102) can also be neglected. To see this, first note that the choice of L in Eq. (96) ensures that $\xi \leq 1/\log\aleph$, so $\aleph^\xi \leq e$. By Eq. (117), this implies that $DA_{L-1}\tau_L = \Theta(\aleph^{1/6-2\xi}) = \Omega(1)$. It follows that Eq. (101) also gives an upper bound on the number of queries needed to simulate H_L for time τ_L.

Now we analyze the error in the Trotter formula for $\sum_{\ell=2}^{L} H_\ell$. The norm of the H_ℓ for $\ell \geq 2$ is largest for $\ell = L$, in which case we have the bound

$$\begin{aligned}\|H_L\| &\leq \|H\| + \|H\|^2/A_{L-1} \\ &= O\left(\Lambda\aleph^{1/3-\xi}\right).\end{aligned} \tag{119}$$

By Eq. (88), the error in the $K = 2$ integrator is

$$\begin{aligned}
O\left(\left[\frac{L\tau_2\Lambda^2}{A_{L-1}}\right]^5 \frac{t}{\tau_2}\right) &= O\left(\frac{L\delta^4 A_2^8 \aleph^{5/3-5\xi}}{\Lambda^{11}t^3}\right) \\
&= O\left(\frac{\delta^4 L}{(\Lambda t)^3 \aleph^{1/3+17\xi}}\right) \\
&\le O\left(\delta\left[\frac{\delta}{\Lambda t}\right]^{8/3}\frac{L^{4/3}}{D^{1/3}}\right) \\
&\le O(\delta). \qquad (120)
\end{aligned}$$

In the last line we have used the assumption $\delta < \Lambda t$ and Eq. (96) for L, which shows that $L = O(\log\log D)$.

So far we have only considered the number of queries to simulate the individual H_ℓ. For the complete simulation, there is an additional factor of L to take account of the integrators. Therefore, the total number of queries is

$$O\left(\frac{D^{2/3}(L\Lambda t)^{4/3}\aleph^\xi}{\delta^{1/3}}\right). \qquad (121)$$

As discussed above, $\aleph^\xi \le e$, so this factor can be ignored. Overall, we find that

$$O\left(\frac{D^{2/3}[(\log\log D)\Lambda t]^{4/3}}{\delta^{1/3}}\right) \qquad (122)$$

queries suffice for the simulation, as claimed. □

8 Implementation of unitaries

Next we explain how to implement a unitary transformation using the results for simulation of Hamiltonians. A simple way to implement a unitary transformation U, as proposed by Jordan and Wocjan [19] (and independently observed by one of us), is to simulate the Hamiltonian

$$H = \begin{bmatrix} 0 & U \\ U^\dagger & 0 \end{bmatrix}. \qquad (123)$$

The Hilbert space consists of a qubit tensored with the target space. Since $H^2 = \mathbb{1}$, we have

$$e^{-iHt} = \cos(t)\mathbb{1} - i\sin(t)H, \qquad (124)$$

and applying this Hamiltonian for time $t = \pi/2$ yields the evolution

$$e^{-iH\pi/2}|1\rangle|\psi\rangle = -i|0\rangle U|\psi\rangle, \qquad (125)$$

which is sufficient to implement U.

Properties of the unitary U and its associated Hamiltonian in Eq. (123) are closely related. The dimension of the Hamiltonian, M, is simply twice the dimension of the unitary, N. In addition, we have

$$\|H\| = \|U\| = 1, \qquad (126)$$

$$\|H\|_1 = \max\{\|U\|_1, \|U^\dagger\|_1\}, \qquad (127)$$

$$\|H\|_{\max} = \|U\|_{\max}. \qquad (128)$$

We assume that the matrix elements of U are given by an oracle O_U as in Eq. (2). This oracle can trivially be used to construct an oracle O_H for the Hamiltonian as in Eq. (1). Each call to O_H uses one call to O_U, so black-box Hamiltonian simulation results can be applied directly to black-box unitary implementation. However, for unitary implementation we can take advantage of the fact that the eigenvalues of the Hamiltonian are restricted.

Lemma 10. *Suppose H has eigenvalues ± 1 and $\pi/[2\arcsin(\varepsilon/\Lambda_1)]$ is an odd integer, for $\varepsilon \in (0,1]$. Using a quantum walk with states $|\phi_j\rangle$ as in Eq. (25), evolution for time $\pi/2$ can be simulated exactly using $O(\Lambda_1/\varepsilon)$ queries.*

Proof. Since the eigenvalues of H are $\lambda = \pm 1$, the relationship between $\arcsin\tilde{\lambda}$ and $\tilde{\lambda}$ is simple: taking

$$d = \frac{\pi}{2\arcsin(\varepsilon/\Lambda_1)}, \tag{129}$$

the eigenvalues of V^d are $+i$ for $\lambda = 1$ and $-i$ for $\lambda = -1$. These eigenvalues are equivalent (up to the minus sign) to evolution under the Hamiltonian H for time $\pi/2$. □

This result can be used to exactly implement unitary operators via a quantum walk. The scaling is as follows:

Theorem 11. *Given a black-box unitary U, let $\Lambda_{\max} \geq \|U\|_{\max}$. Then U can be implemented exactly with $O(N\Lambda_{\max})$ queries to O_U.*

Since we are primarily concerned with implementation of general unitaries, which are not sparse, we express unitary implementation results in terms of the dimension N rather than the sparseness parameter D.

Proof. This implementation proceeds by simulating the Hamiltonian given in Eq. (123) for time $\pi/2$ using Lemma 10, with the steps of the quantum walk implemented using Lemma 4. The Hamiltonian has no more than N nonzero elements in any row of column, so we can take $D = N$.

Take $\varepsilon = \Lambda_1/NX$, where

$$X = \frac{1}{N\sin[\pi/(2d)]}, \tag{130}$$

$$d = 2\left\lceil \frac{\pi}{4\arcsin[1/(\Lambda_{\max}N)]} - \frac{1}{2} \right\rceil + 1. \tag{131}$$

It is easily shown that $X \geq \Lambda_{\max}$, so $\varepsilon \leq \Lambda_1/D\Lambda_{\max} \leq 1$. In addition, $\pi/[2\arcsin(\varepsilon/\Lambda_1)]$ is an odd integer, so the conditions of Lemma 10 are satisfied. Then, using Lemma 10, the Hamiltonian can be simulated for time $\pi/2$ using $O(\Lambda_1/\varepsilon) = O(NX)$ steps of the quantum walk.

Because $\varepsilon \leq \Lambda_1/D\Lambda_{\max}$, we can use Lemma 4, and each step of the quantum walk can be implemented using $O(1)$ queries. Thus the total number of queries is $O(NX)$. Because $\Lambda_{\max} \geq \|H\|_{\max} \geq 1/\sqrt{N}$, $X \leq 2\Lambda_{\max}$, and the number of queries is $O(N\Lambda_{\max})$. □

Our other results on Hamiltonian simulation can also be used to implement unitaries, although in these cases there are other sources of error, so the simulation can no longer be performed exactly. In each case we take $t = \pi/2$, $\|H\| = 1$, and $D = N$. Lemma 8 yields the following corollary for unitary implementation.

Corollary 12. *Given a black-box unitary U, let $\Lambda_1 \geq \max\{\|U\|_1, \|U^\dagger\|_1\}$ and $\Lambda_{\max} \geq \|U\|_{\max}$. Then U can be implemented with error at most $\delta \in (0,1]$ using*

$$O\left(\sqrt{\Lambda_{\max} N \Lambda_1/\delta}\right) \tag{132}$$

queries to O_U.

Proof. We apply Lemma 8 together with the norm bounds in Eqs. (126) to (128). We use $t = \pi/2$ and $\|H\| = 1$ to obtain Eq. (132). We omit conditions (66) to (68) of Lemma 8 as they are automatically satisfied. First, the condition (66) holds because $\delta \leq 1$. Second, (67) holds because $\Lambda_{\max} \geq 1/\sqrt{N}$ and $\Lambda_1 \geq \|U\|_1 \geq 1$. Third, (68) holds because $\Lambda_1 \geq 1 = \Lambda$. □

Similarly, Theorem 9 yields the following.

Corollary 13. *Let $\Upsilon \in [\mathrm{break}(U), \sqrt{N}]$. The unitary operation U can be implemented with error at most $\delta \in (0,1]$ using*

$$O\left(\sqrt{\Upsilon N/\delta}(\log N)^{7/4}\right) \tag{133}$$

queries to O_H and O_F, provided $\delta N > \pi/2$.

Proof. We use Theorem 9 with $\Lambda = \|H\| = 1$, $t = \pi/2$, and $D = N$. Then the number of queries is as in Eq. (133). Since $\|H\|t = \pi/2$, the condition $\delta D > \Lambda t > \sqrt{\delta}$ in Theorem 9 becomes $\delta N > \pi/2 > \sqrt{\delta}$, and since $\delta \leq 1$, the latter inequality is trivial. □

Finally, Theorem 2 yields Corollary 3, which can be proven as follows.

Proof of Corollary 3. For unitaries, $\Lambda = \|H\| = 1$, $t = \pi/2$, and $D = N$, so Eq. (5) gives an upper bound of

$$O\left(N^{2/3}(\log\log N)^{4/3}\delta^{-1/3}\right) \tag{134}$$

queries, as claimed. The condition $\delta D > \Lambda t > \sqrt{\delta}$ becomes $\delta N > \pi/2$ for the same reason as in the proof of Corollary 13 above. If $\delta N \leq \pi/2$, then we instead implement the unitary using Theorem 11. This takes $O(N)$ queries, which is smaller than the claimed upper bound. □

In the worst case, Corollary 12 yields query complexity of $O(N^{3/4}/\sqrt{\delta})$. This is because $\Lambda_{\max}$ could be as large as 1 and Λ_1 could be as large as $\sqrt{N}$. On the other hand, if the nonzero matrix elements are of similar magnitude, then $\Lambda_{\max} \propto 1/\Lambda_1$, so the scaling will be $O(\sqrt{N/\delta})$. Alternatively, if it is possible to break the unitary into components without otaining large spectral norms, then $\mathrm{break}(U) = O(1)$, and Corollary 13 yields scaling of $\tilde{O}(\sqrt{N/\delta})$. Those results are not sufficient to prove this scaling for all unitaries, because $\mathrm{break}(U)$ may be large. However, in general we can use Corollary 3 to implement any unitary with $\tilde{O}(N^{2/3}\delta^{-1/3})$ queries.

9 Examples

We now consider some simple examples of unitaries and discuss the query complexity of implementing them by the methods of the previous section.

First, consider the unitary with matrix elements $U_{jk} = g(j + k \bmod N)$, where g is a black-box function for a search problem with a unique marked item $j^\star$. The function $g\colon \{1, 2, \ldots, N\} \to \{0, 1\}$ satisfies $g(j^\star) = 1$ and $g(j) = 0$ for $j \neq j^\star$. Since $U|0\rangle = |j^\star\rangle$, implementing U solves the search problem; thus it requires $\Omega(\sqrt{N})$ queries [20]. This unitary has $\|U\|_{\max} = 1$ and $\|U\|_1 = 1$, so Corollary 12 gives a complexity of $O(\sqrt{N/\delta})$, which is optimal. In fact, simply implementing the isometry T solves the search problem, because it can prepare the state $T|0\rangle = |0\rangle|j^\star\rangle$. The implementation of T in this case is in fact equivalent to the standard Grover search algorithm [25].

In this case, σ_j is known, so the implementation can be performed exactly. Using Lemma 10, unitaries may be implemented exactly using a quantum walk, so the only remaining source of error is in performing the steps of the quantum walk. From the proof of Lemma 6, the steps of the quantum walk may be performed exactly if r_j^{opt} from Eq. (55) is a known integer. Because we have $\sigma_j = \varepsilon = \Lambda_1 = 1$, we can easily adjust X to ensure that this is the case, and therefore that the simulation is performed exactly. More generally, whenever U is a permutation matrix, it can be implemented in only $O(\sqrt{N})$ queries in a similar fashion.

Another simple example is the quantum Fourier transform, the unitary with $U_{jk} = e^{2\pi ijk/N}/\sqrt{N}$. For this unitary, $\|U\|_{\max} = 1/\sqrt{N}$ and $\|U\|_1 = \sqrt{N}$. Therefore, Corollary 12 again gives a complexity of $O(\sqrt{N/\delta})$. In fact, for this case we can take $\varepsilon = 1$, $\sigma_j = \Lambda_1 = \sqrt{N}$ and $X = 1/\sqrt{N}$, so Eq. (55) gives $r_j^{\mathrm{opt}} = 0$. Therefore no amplitude amplification is required, and the implementation is again exact.

These two examples illustrate the two extremal cases where Corollary 12 gives scaling of $\sqrt{N}$. First, if all the weight is on one matrix element in each row, then $\|U\|_1 = 1$. At the other extreme, if the weight is evenly distributed between the matrix elements, then $\|U\|_1 = \sqrt{N}$, but $\|U\|_{\max} = 1/\sqrt{N}$. These correspond to the two points listed at the end of Sec. 6. In either case, the nonzero matrix elements have the same magnitude.

Note that to take advantage of sparsity, the locations of the nonzero elements must be known (or more precisely, their locations must be accessible via the oracle O_F). Effectively, the quantity D measures how many matrix elements are not known to be zero. For the search problem, the locations of the nonzero elements are not known in advance (finding those positions would in itself solve the search problem), so $D = N$. In contrast, for the norms, it does not matter if the nonzero elements are in known positions. If there are m nonzero matrix elements, then $\|U\|_1 \le \sqrt{m}$ regardless of the positions of those elements.

For Corollary 12 to yield scaling worse than $\sqrt{N}$, the distribution of magnitudes of matrix elements of U must have a sharp peak in combination with a relatively broad distribution for the remaining elements. As a natural example of this, consider the unitary given by $U = \exp(-i\pi J_x/2)$, where J_x is the x-rotation operator for a spin-J system, with dimension $N = 2J + 1$, and we use the basis of J_z eigenstates. The first column of $\exp(-i\pi J_x/2)$ has a relatively narrow peak, whereas for columns towards the middle the elements are more spread out (see Fig. 3). The maximum element of U has absolute value $\sqrt{(2\lceil J\rceil)!}/2^{\lceil J\rceil}\lceil J\rceil!$, which is $O(J^{-1/4})$ by Stirling's formula. Since $\|U\|_1 = O(\sqrt{J})$, Corollary 12 yields an overall number of black-box queries of $O(N^{5/8}/\delta^{1/2})$.

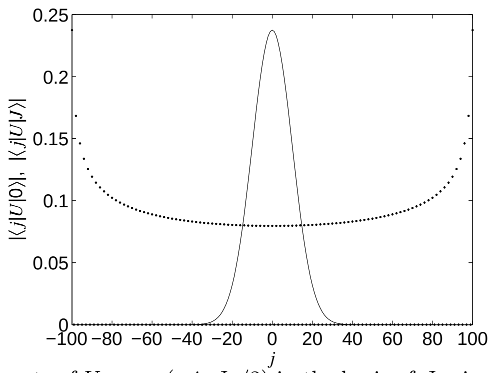

Fig. 3. The matrix elements of $U = \exp(-i\pi J_x/2)$ in the basis of J_z eigenstates for $J = 100$. The separate points are $|\langle j|U|0\rangle|$, and the solid curve is $|\langle j|U|J\rangle|$.

Thus Corollary 12 does not provide $\sqrt{N}$ scaling in this case, though the scaling is better than the worst-case $N^{3/4}$ scaling. Using Corollary 3 would not yield improved scaling in this example, because $2/3 > 5/8$. However, numerical testing indicates that breaking this unitary into components does not increase the spectral norms, so using the approach given in Sec. 7.1 would yield $\sqrt{N}$ scaling. In this example, calculating the matrix elements of U is nontrivial, and consequently the overall complexity of the algorithm in terms of elementary gates would be greater than $\sqrt{N}$.

We emphasise that the motivation to implement unitaries is not as a shortcut to simulation of Hamiltonians via $U = e^{-iHt}$. In general, calculating the matrix elements of $U = e^{-iHt}$ given the matrix elements of H may be difficult. Rather, the motivation for implementing unitaries is to provide a tool to develop other algorithms. As discussed above, the search problem may be encoded as a unitary operation. The algorithm for implementing unitaries may be regarded as a new generalisation of the Grover algorithm.

10 Conclusion

We have shown how to use quantum walks to simulate black-box Hamiltonians. In particular, we showed that these techniques can be used to implement an arbitrary $N \times N$ unitary transformation using $\tilde{O}(N^{2/3}/\delta^{1/3})$ queries to a black box for its matrix elements, with error at most δ as quantified by the trace distance.

Our approach is based on simulating Hamiltonian dynamics via discrete-time quantum walk [10], combined with state preparation via amplitude amplification [12] and integrators to break up the Hamiltonian. In many cases the implementation can be performed even faster, with $\tilde{O}(\sqrt{N/\delta})$ black-box calls. This scaling can be achieved except when breaking the Hamiltonian into a sum of terms yields components with large spectral norms, and numerical testing suggests that such cases are rare.

For many applications, our work provides the best known simulation of sparse Hamiltonians. The number of queries is strictly linear in $\|H\|t$, rather than slightly superlinear, as when higher-order integrators are used [9, 8]. In addition, the scaling in the sparseness parameter D is at worst linear, in contrast with the $O(D^4)$ scaling of Ref. [9].

The best lower bound we know for black-box unitary implementation is $\Omega(\sqrt{N})$ queries, because implementing an $N \times N$ unitary suffices to solve unstructured search with N items.

It remains an open problem to determine whether it is possible to perform the simulation using $O(\sqrt{N})$ queries in general.

Our results also apply to more general Hamiltonian simulation problems. It might be interesting to investigate the extent to which the general black-box Hamiltonian simulation described by Theorem 2 can be improved. Simulations using $O(\|Ht\|)$ queries are not possible in general [30], but the tradeoff between quantities such as D, $\|Ht\|$, $\|Ht\|_1$, $\|Ht\|_{\max}$, and δ is poorly understood.

Acknowledgements

We thank Richard Cleve, Aram Harrow, Stephen Jordan, Robin Kothari, and John Watrous for helpful discussions. DWB is funded by an Australian Research Council Future Fellowship (FT100100761). AMC received support from MITACS, NSERC, the Ontario Ministry of Research and Innovation, QuantumWorks, and the US ARO/DTO.

References

1. R. P. Feynman, *Simulating physics with computers*, International Journal of Theoretical Physics **21**, 467 (1982).
2. A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, and D. A. Spielman, *Exponential algorithmic speedup by quantum walk*, in *Proceedings of the 35th ACM Symposium on Theory of Computing* (ACM, New York, 2003), pp. 59–68; arXiv:quant-ph/0209131.
3. E. Farhi, J. Goldstone, and S. Gutmann, *A quantum algorithm for the Hamiltonian NAND tree*, Theory of Computing **4**, 169 (2008); arXiv:quant-ph/0702144.
4. A. M. Childs, R. Cleve, S. P. Jordan, and D. Yonge-Mallo, *Discrete-query quantum algorithm for NAND trees*, Theory of Computing **5**, 119 (2009); quant-ph/0702160.
5. A. W. Harrow, A. Hassidim, and S. Lloyd, *Quantum algorithm for linear systems of equations*, Phys. Rev. Lett. **103**, 150502 (2009).
6. S. Lloyd, *Universal quantum simulators*, Science **273** 1073 (1996).
7. D. Aharonov and A. Ta-Shma, *Adiabatic quantum state generation and statistical zero knowledge*, in *Proceedings of the 35th ACM Symposium on Theory of Computing, 2003* (ACM, New York, 2003), pp. 20–29; arXiv:quant-ph/0301023.
8. A. M. Childs, *Quantum information processing in continuous time*, Ph.D. thesis, Massachusetts Institute of Technology, 2004.
9. D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, *Efficient quantum algorithms for simulating sparse Hamiltonians*, Commun. Math. Phys. **270**, 359 (2007); arXiv:quant-ph/0508139.
10. A. M. Childs, *On the relationship between continuous- and discrete-time quantum walk*, Commun. Math. Phys. **294**, 581 (2009); arXiv:0810.0312.
11. A. M. Childs and R. Kothari, *Simulating sparse Hamiltonians with star decompositions*, Theory of Quantum Computation, Communication, and Cryptography (TQC 2010), Lecture Notes in Computer Science **6519**, 94 (2011); arXiv:1003.3683.
12. L. K. Grover, *Synthesis of quantum superpositions by quantum computation*, Phys. Rev. Lett. **85**, 1334 (2000).
13. M. Reck, A. Zeilinger, H. J. Bernstein, and P. Bertani, *Experimental realization of any discrete unitary operator*, Phys. Rev. Lett. **73**, 58 (1994).
14. A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, *Elementary gates for quantum computation*, Phys. Rev. A **52**, 3457 (1995); quant-ph/9503016.
15. A. Y. Kitaev, *Quantum computations: Algorithms and error correction*, Russ. Math. Surveys **52**, 1191 (1997).

16. A. Y. Kitaev, A. H. Shen, and M. N. Vyalyi, *Classical and Quantum Computation*, Graduate Studies in Mathematics Vol. 47 (American Mathematical Society, Providence, RI, 2002).
17. A. W. Harrow, B. Recht, and I. L. Chuang, *Efficient discrete approximations of quantum gates*, J. Math. Phys. **43**, 4445 (2002).
18. E. Knill, *Approximation by quantum circuits*, Technical Report LAUR-95-2225, Los Alamos National Laboratory, 1995; arXiv:quant-ph/9508006.
19. S. P. Jordan and P. Wocjan, *Efficient quantum circuits for arbitrary sparse unitaries*, Phys. Rev. A **80**, 062301 (2009); arXiv:0904.2211.
20. C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, *Strengths and weaknesses of quantum computing*, SIAM J. Comput. **26**, 1510 (1997); arXiv:quant-ph/9701001.
21. M. Szegedy, *Quantum speed-up of Markov chain based algorithms*, Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (IEEE, Los Alamitos, CA, 2004), pp. 32–41; arXiv:quant-ph/0401053.
22. G. Brassard and P. Høyer, *An exact quantum polynomial-time algorithm for Simon's problem*, in *Proceedings of Fifth Israeli Symposium on Theory of Computing and Systems* (IEEE, Los Alamitos, CA, 1997), pp. 12–23; arXiv:quant-ph/9704027.
23. G. Brassard, P. Høyer, M. Mosca, and A. Tapp, in *Quantum Computation and Information*, edited by S. J. Lomonaco and H. E. Brandt (AMS, Providence, 2002); arXiv:quant-ph/0005055.
24. L. K. Grover, *Quantum computers can search rapidly by using almost any transformation*, Phys. Rev. Lett. **80**, 4329 (1998).
25. L. K. Grover, *A fast quantum mechanical algorithm for database search*, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (ACM, New York, 1996), pp. 212–219; arXiv:quant-ph/9605043.
26. G. Strang, *On the construction and comparison of difference schemes*, SIAM J. Numer. Anal. **5**, 506 (1968).
27. M. Suzuki, *Fractal decomposition of exponential operators with applications to many-body theories and Monte Carlo simulations*, Phys. Lett. A **146**, 319 (1990).
28. M. Suzuki, *General theory of fractal path integrals with applications to many-body theories and statistical physics*, J. Math. Phys. **32**, 400 (1991).
29. N. Wiebe, D. W. Berry, P. Høyer, and B. C. Sanders, *Higher order decompositions of ordered operator exponentials*, J. Phys. A: Math. Theor. **43**, 065203 (2010); arXiv:0812.0562.
30. A. M. Childs and R. Kothari, *Limitations on the simulation of non-sparse Hamiltonians*, Quantum Inform. Comput. **10**, 669 (2010); arXiv:0908.4398.