Time-marching based quantum solvers for time-dependent linear differential equations

Di Fang[1,2,3], Lin Lin[1,4,3], and Yu Tong[5,1]

[1]Department of Mathematics, University of California, Berkeley, CA 94720, USA
[2]Simons Institute for the Theory of Computing, University of California, Berkeley, CA 94720, USA
[3]Challenge Institute for Quantum Computation, University of California, Berkeley, CA 94720, USA
[4]Applied Mathematics and Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[5]Institute for Quantum Information and Matter, California Institute of Technology, Pasadena, CA 91125, USA

The time-marching strategy, which propagates the solution from one time step to the next, is a natural strategy for solving time-dependent differential equations on classical computers, as well as for solving the Hamiltonian simulation problem on quantum computers. For more general homogeneous linear differential equations $\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = A(t)|\psi(t)\rangle, |\psi(0)\rangle = |\psi_0\rangle$, a time-marching based quantum solver can suffer from exponentially vanishing success probability with respect to the number of time steps and is thus considered impractical. We solve this problem by repeatedly invoking a technique called the uniform singular value amplification, and the overall success probability can be lower bounded by a quantity that is independent of the number of time steps. The success probability can be further improved using a compression gadget lemma. This provides a path of designing quantum differential equation solvers that is alternative to those based on quantum linear systems algorithms (QLSA). We demonstrate the performance of the time-marching strategy with a high-order integrator based on the truncated Dyson series. The complexity of the algorithm depends linearly on the amplification ratio, which quantifies the deviation from a unitary dynamics. We prove that the linear dependence on the amplification ratio attains the query complexity lower bound and thus cannot be improved in the worst case. This algorithm also surpasses existing QLSA based solvers in three aspects: (1) $A(t)$ does not need to be diagonalizable. (2) $A(t)$ can be non-smooth, and is only of bounded variation. (3) It can use fewer queries to the initial state $|\psi_0\rangle$. Finally, we demonstrate the time-marching strategy with a first-order truncated Magnus series, which simplifies the implementation compared to high-order truncated Dyson series approach, while retaining the aforementioned benefits. Our analysis also raises some open questions concerning the differences between time-marching and QLSA based methods for solving differential equations.

Contents

1 Introduction

In modern science and engineering, mathematical descriptions of "real-world" problems often lead to differential equations, which play a prominent role in many disciplines such as physics, chemistry, biology and economics. Efficient simulation of large scale differential equations has therefore served as one of the core tasks in many scientific applications. Recent advances of quantum algorithms indicate that quantum computers may significantly accelerate the simulation of such differential equations, particularly for those defined in high dimensional spaces. Let N be the number of degrees of freedom (e.g., the number of discretized points in a high dimensional space) which can be very large. For certain tasks, quantum computers can store and manipulate vectors of size N at a cost that scales only as polylog(N), which leads to potentially significant advantages over classical computers.

In this paper, we focus on the initial value problem of the system of homogeneous linear ordinary differential equations (ODEs)

$$\frac{\mathrm{d}}{\mathrm{d}t}\left|\psi(t)\right\rangle = A(t)\left|\psi(t)\right\rangle, \quad \left|\psi(0)\right\rangle = \left|\psi_0\right\rangle, \tag{1}$$

where $t \in [0,T] \subset \mathbb{R}^+$ is the independent variable with T as the final time, the vector $|\psi(t)\rangle \in \mathbb{C}^N$ is the dependent variable, the coefficient matrix $A(t) \in \mathbb{C}^{N\times N}$ is a matrix-valued function in t, and $|\psi_0\rangle \in \mathbb{C}^N$ is the initial condition. We assume the differential equation has a well-posed solution, which can be implied by, e.g., $A(t)$ being piecewise continuous. The coefficients can have jump discontinuity and are not required to be smooth in t. More precisely, we only need to assume that $A(t)$ is of bounded variation, i.e., the total variation $V_0^T(A)$ (see Eq. (10) for definition) is finite. The task for quantum differential equation solvers is to prepare a quantum state that is proportional to the final solution $|\psi(T)\rangle$ with certain precision.

One prominent example is the Hamiltonian simulation problem, which is a homogeneous linear differential equation governed by $A(t) = -iH(t)$ and $H(t)$ is a Hermitian matrix. Recent years have witnessed remarkable progresses on designing new algorithms as well as establishing improved theoretical complexity estimate of existing algorithms for both time-independent Hamiltonian simulation [9–13, 15, 20, 23, 27–30, 53, 55, 57, 60, 69] and time-dependent ones [3, 4, 7, 24, 47, 57, 66, 67]. These quantum algorithms can be applied, when the underlying dynamics is unitary or can be converted into a unitary dynamics (see e.g., [33]). Compared to the Hamiltonian simulation problem whose dynamics is unitary, quantum algorithms for the general linear differential equations are considerably less explored. Such algorithms produce a quantum state representing the solution of the differential equation. This is different from having the solution stored in classical memory, but we can still extract information from the quantum state that may not be efficiently obtainable from classical computation (see e.g., [26]). Unless otherwise specified, we do not consider the special case when $A(t) = A$ is time-independent, in which setting we may directly encode the propagator (i.e., the matrix function e^{AT}) using the quantum singular value transformation (QSVT) [38] when A is Hermitian or anti-Hermitian, or using a contour integral based strategy for a general A (see [59, 61], as well as Section 5.3).

The *time-marching* strategy is a natural strategy solving time-dependent differential equations, and is adopted by nearly all classical differential equation solvers (see e.g., [42, 43]). The idea is to divide the entire time interval $[0, T]$ into a number of short line segments separated by the temporal mesh points $0 = t_0 < t_1 < \cdots < t_L = T$, and to calculate the solution information at the next time step using the solution information from previous k time steps. Methods with $k = 1$ are called one-step methods (e.g., Runge-Kutta methods). Methods with $k > 1$ are called multi-step methods (e.g., Adams methods). Multi-step methods require multiple copies of the solution from previous steps, and cannot be directly implemented on quantum computers due to the obstruction of the no-cloning theorem. For non-unitary dynamics, even one-step methods lead to severe challenges in the design of quantum algorithms, mainly due to potentially diminishing success probability. In Section 1.3, we use an illustrative example with the simplest one-step integrator (the forward Euler method) to demonstrate such challenges. As a result, the time-marching strategy has not been viewed as a practical route for designing quantum differential equation solvers beyond the Hamiltonian simulation problem.

1.1 Related works

The prevailing strategy for designing quantum differential equation solvers to date is to construct a large linear system recording states during the entire history of the evolution, and then to apply the quantum linear systems algorithms (QLSA) [1, 5, 25, 32, 44, 49, 58] to solve the resulting linear systems of equations. One may perform certain amplification procedures to boost the success probability of getting the final solution. This QLSA-based strategy was first proposed by Berry [8], which successfully avoids the pitfall in Section 1.3. It has been adopted by various quantum linear differential equation solvers [14, 26, 48, 50], and has been applied to solve nonlinear differential equations using linearization techniques [2, 34, 36, 45, 46, 51, 52, 64].

The work [8] uses multi-step integrators, and the analysis is applicable to time-independent A that is diagonalizable as $A = VDV^{-1}$ with eigenvalues $\lambda_j = D_{jj}$ satisfying

$$|\arg(-\lambda_j)| \leq \theta_0, \quad 0 < \theta_0 < \pi/2, \quad \forall j. \tag{2}$$

The query complexity of the algorithms scales polynomially in T, the spectral norm $\|A\|$, the inverse precision ϵ^{-1}, and the condition number of the eigenvector matrix $\kappa_V := \|V\| \, \|V^{-1}\|$. For time-independent A, the work [14] combines a QLSA-based solver with the truncated Taylor series method, and the assumption in Eq. (2) was relaxed to A being dissipative (i.e., $\mathrm{Re}(\lambda_j) \leq 0, \forall j$). The complexity of this algorithm is also improved to be $\widetilde{\mathcal{O}}(Td \, \|A\| \, \kappa_V q \mathrm{polylog}(\epsilon^{-1}))$, where d is the sparsity of the matrix A, κ_V is the condition number of V, and $q = \sup_{t\in[0,T]} \||\psi(t)\rangle\| \, / \, \||\psi(T)\rangle\|$. For time-dependent linear differential equations, [26] combines a QLSA-based solver with a Chebyshev pseudospectral method. It assumes that $A(t)$ is diagonalizable as $A(t) = V(t)D(t)V(t)^{-1}$ and dissipative for all $t \in [0, T]$, and the underlying solution is sufficiently smooth in t. The complexity of the algorithm is $\widetilde{\mathcal{O}}(T\alpha\kappa_V q d \mathrm{polylog}(g' g^{-1} \epsilon^{-1}))$, where d is the sparsity of $A(t)$, $\alpha = \sup_{t\in[0,T]} \|A(t)\|$, $\kappa_V = \sup_{t\in[0,T]} \|\kappa_V(t)\|$, $g = \||\psi(T)\rangle\|$, $g' = \max_{t\in[0,T]} \max_{n\in\mathbb{N}} \left\||\psi^{(n+1)}(t)\rangle\right\|$, and $q = \sup_{t\in[0,T]} \||\psi(t)\rangle\| \, / \, \||\psi(T)\rangle\|$. The smoothness of the solution implicitly requires that $A(t)$ should be sufficiently smooth (e.g., analytic in t). When the coefficient matrix $A(t)$ is not sufficiently smooth, the polylogarithmic dependence on g' no longer holds, and the complexity of the algorithm depends polynomially on ϵ^{-1}.

In all the analysis above, the complexity depends explicitly on the condition number of the eigenvector matrix κ_V, which can be difficult to estimate and may lead to significant overestimation of the cost. The recent work by Krovi [48] replaces the dependence on κ_V by the dependence on $\sup_{t\in[0,T]} \left\|e^{At}\right\|$. This is a significant improvement due to the inequality

$$\sup_{t\in[0,T]} \left\|e^{At}\right\| \leq \sup_{t\in[0,T]} \left\|e^{Dt}\right\| \kappa_V.$$

For instance, consider

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1+\delta \end{pmatrix} = VDV^{-1}, \quad \text{with} \quad V = \begin{pmatrix} 1 & \frac{1}{\delta} \\ 0 & 1 \end{pmatrix}, \quad D = \begin{pmatrix} 1 & 0 \\ 0 & 1+\delta \end{pmatrix}. \tag{3}$$

Then $\|e^A\| = \mathcal{O}(1)$, but κ_V is $\Omega(\delta^{-2})$ which diverges as $\delta \to 0$. However, the technique in [48] is only applicable when A is time-independent.

1.2 Contribution

In this work, we propose that the time-marching method *can* become an efficient strategy for solving linear differential equations. Our main technical tool (Theorem 4) is a method to implement a sequence of non-unitary operations without incurring an exponential overhead. Theorem 4 has two main ingredients. The first is the uniform singular value amplification procedure, which was first developed in [54] and was refined in [38]. It allows us to amplify the success probability in each time step in a way that is oblivious to the quantum state. We find that in our context, the degree of the polynomial used by the uniform singular value amplification procedure in [38] exhibits a very large preconstant. We use a convex optimization based method to reduce the preconstant by orders of magnitude. Theorem 4 also proposes a useful tool called the compression gadget (which improves upon the result developed in [57]) to coherently combine short-time evolution operators into a long-time one without duplicating ancilla qubits. Combined with amplitude amplification [17], this leads to a further quadratic speedup in terms of the query complexity.

Consider a temporal mesh $0 = t_0 < t_1 < \cdots < t_L = T$. Using the time-marching strategy and the high-order truncated Dyson series algorithms [12, 47, 57] for short time evolution, we propose an algorithm to solve Eq. (1), which requires $\widetilde{\mathcal{O}}(Q(\alpha T)^2 \log(\epsilon^{-1}))$ queries to the coefficient matrix $A(t)$, and $\mathcal{O}(Q)$ queries to the initial state $|\psi_0\rangle$ (Theorem 8). Here $\alpha = \sup_{t\in[0,T]} \|A(t)\|$, and

$$Q = \frac{\prod_{l=1}^{L} \|\mathcal{T} e^{\int_{t_{l-1}}^{t_l} A(t)\mathrm{d}t}\|}{\| \, |\psi(T)\rangle \, \|}, \tag{4}$$

as a central quantity of this work, is the *amplification ratio*. It quantifies the non-unitarity of the dynamics (e.g., $Q = 1$ for unitary dynamics).

Compared to the state-of-the-art results based on the QLSA in [26] and [48], our algorithm has several advantages, which we summarize in Table 1. First, our method does not require the diagonalizability of $A(t)$, and the complexity is independent of the condition number κ_V. This generalizes the result of [48] to equations with time-dependent matrix coefficients.

Second, our algorithm also has lower regularity requirement for the coefficient matrix $A(t)$ and the solution $|\psi(t)\rangle$. In [26] the query complexity depends on the high-order derivatives of

the solution ($g' = \max_{t\in[0,T]} \max_{n\in\mathbb{N}} \left\| |\psi^{(n+1)}(t)\rangle \right\|$ in [26, Theorem 1]), and if the solution is not smooth, the spectral method loses the desirable exponential accuracy. Remarkably, in our method, exponential accuracy is achieved even when the coefficient matrix $A(t)$ is not smooth, thus allowing $|\psi(t)\rangle$ to be non-smooth. The results in both Theorem 8 and Theorem 14 are insensitive to the roughness in the coefficients $A(t)$. Note that $A(t)$ being of bounded variation is a very weak regularity condition, and in particular $V_0^T(A) = \int_0^T \|A'(t)\| \mathrm{d}t$ when A is differentiable. This also allows for the existence of jump discontinuities in the coefficients.

Third, our algorithm may use fewer queries to the initial state, which is advantageous if the initial state preparation is an important factor. To simplify the discussion, let us focus on the dependence on T and assume all other quantities such as α, Q to be constants. In order to achieve the scaling in [26, Theorem 1] (for other QLSA based differential equation solvers discussed in Section 1.1, the situation is similar), the quantum solver must employ a quantum linear system solver with near optimal query complexities. In other words, the complexity of the linear system solver should scale as $\widetilde{\mathcal{O}}(\kappa \mathrm{polylog}(\epsilon^{-1}))$, and κ is the condition number of the linear system. Moreover, the near optimal quantum algorithms also need to query the initial state for $\widetilde{\mathcal{O}}(\kappa)$ times. Such a dependence is most clearly seen from the perspective of adiabatic based near-optimal quantum linear system solvers [5, 32, 49]. This is because the construction of the adiabatic Hamiltonian corresponding to the linear system uses the initial state, and thus each query to the adiabatic Hamiltonian also queries the initial state. As a result, the quantum differential equation solvers in both [26, 48] need to query the initial state for $\widetilde{\mathcal{O}}(\kappa) = \widetilde{\mathcal{O}}(T)$ times. The relation between κ and T is rooted in the no-fast-forwarding theorem and cannot be generally improved. Our time-marching based algorithm does not rely on such adiabatic constructions, and the query complexity to the initial states does not explicitly depend on T. In the case where the time evolution is almost unitary, i.e., $Q = \mathcal{O}(1)$, and where T is large, this feature can offer significant advantage.

Our method (Theorem 8) also has two drawbacks. The first is that the T dependence in the number of queries to $A(t)$ is sub-optimal. The direct reason is that the uniform singular value amplification procedure becomes increasingly costly as T increases. The second is that the cost of our algorithm depends on Q defined in Eq. (4), while previous QLSA based algorithms depends on $q = \max_{t\in[0,T]} \frac{\||\psi(t)\rangle\|}{\||\psi(T)\rangle\|}$, which satisfies $q \leq Q$. We prove in Theorem 10 that the $\mathcal{O}(Q)$ dependence attains the query lower bound and cannot be improved in the worst case. There exists instances that Q can significantly (even exponentially) overestimate q, as will be discussed at the end of Section 4. However, $q\kappa_V$ and Q may not be directly related in general.

Finally, the implementation of the high-order truncated Dyson series algorithm requires complicated quantum control logic for handling time-ordering operators. To simplify the implementation, we combine the time-marching strategy with a first-order truncated Magnus series, whose implementation does not require the complex time-clocking quantum control logics. Though the cost of the resulting algorithm depends on higher powers of T and ϵ^{-1} comparing to the high-order integrators, it retains the aforementioned advantages compared to QLSA based solvers. Interestingly, this algorithm also exhibits a commutator scaling for differential equations in the high precision limit (see Theorem 14), which can be desirable when the norm of the commutator $\alpha_{\mathrm{comm}} = \sup_{s,\tau\in[0,T]} \|[A(s), A(\tau)]\|$ is much smaller than α.

Algorithms	κ_V dependence	Requiring smooth $A(t)$	Queries to A	Queries to $\|\psi_0\rangle$
This work (Theorem 8)	No	No	$\widetilde{\mathcal{O}}(T^2Q\alpha^2)$	$\mathcal{O}(Q)$
[26, Theorem 1]	Yes	Yes	$\widetilde{\mathcal{O}}\left(Tq\kappa_V\alpha\log\left(g'/g\right)\right)$	$\widetilde{\mathcal{O}}\left(Tq\kappa_V\alpha\log\left(g'/g\right)\right)$
[48, Theorem 10]	No	-	$\widetilde{\mathcal{O}}(Tq\alpha\sup_{t\in[0,T]}\left\|e^{At}\right\|)$	$\widetilde{\mathcal{O}}(Tq\alpha\sup_{t\in[0,T]}\left\|e^{At}\right\|)$

Table 1: Comparison with state-of-the-art high-order algorithms. We compare the high-order algorithms in terms of whether they require smooth $A(t)$, have κ_V dependence, and the T scaling in the number of queries to $A(t)$ and to the initial state in the case when $A(t)$ is smooth. Here $\alpha = \sup_{t\in[0,T]}\|A(t)\|$, Q is defined as (4), $\kappa_V = \max_{t\in[0,T]}\kappa_V(t)$ is a uniform upper bound of the condition number of $V(t)$ diagonalizing $A(t) = V(t)\Lambda(t)V^{-1}(t)$ when $A(t)$ is diagonalizable for all time $t \in [0,T]$, $q = \max_{t\in[0,T]}\frac{\||\psi(t)\rangle\|}{\||\psi(T)\rangle\|}$, $g = \||\psi(T)\rangle\|$, $g' = \max_{t\in[0,T]}\max_{n\in\mathbb{N}}\left\||\psi^{(n+1)}(t)\rangle\right\|$, where $|\psi^{(n+1)}(t)\rangle$ denotes the $(n+1)$-th derivative of $|\psi(t)\rangle$ and $\mathbb{N}$ is the set of natural numbers. The algorithm in [48, Theorem 10] on the last row is designed only for the time-independent case.

1.3 Challenges in designing time marching based quantum solvers

The most straightforward way of solving the ODE (1) is arguably the (forward) Euler method. For simplicity we consider the time-independent case, i.e., $A(t) = A$, and the time step sizes are chosen to be uniform: $t_l - t_{l-1} = T/L$. We further assume A is a normal matrix and can be unitarily diagonalized. Starting from $|\psi_0\rangle$, at each time step l, we go from $|\psi_{l-1}\rangle \approx |\psi(t_{l-1})\rangle$ to $|\psi_l\rangle \approx |\psi(t_l)\rangle$ via

$$|\psi_l\rangle = (I + A(t_l - t_{l-1}))\,|\psi_{l-1}\rangle\,. \tag{5}$$

We will show that a direct implementation of this method on a quantum computer leads to severe challenges despite its simple appearance. Let us first look at how we should implement $\bar{\Xi}_l = I + A(t_l - t_{l-1})$ on a quantum computer. Generally we can assume that A is given through a block encoding (see Appendix B for a short introduction of block encoding and QSVT), with which we can construct a block encoding of $\bar{\Xi}_l$ denoted by U_l through a linear combination of unitaries. This construction, if performed directly using [38, Lemma 29], involves a subnormalization factor of $1 + \|A\|(t_l - t_{l-1}) = 1 + \|A\|T/L$. Going from $|\psi_{l-1}\rangle$ to $|\psi_l\rangle$, we apply the block encoding U_l, and measure the ancilla qubits. The success of the procedure depends on the measurement outcome, and the success probability is

$$\frac{1}{(1+\|A\|T/L)^2} \times \frac{\|\,|\psi_l\rangle\,\|^2}{\|\,|\psi_{l-1}\rangle\,\|^2}, \tag{6}$$

where the first factor comes from the subnormalization factor discussed above. Since the success at each time step is independent, the total success probability of implementing Euler's method for L steps is

$$\frac{1}{(1+\|A\|T/L)^{2L}} \times \prod_{l=1}^{L}\frac{\|\,|\psi_l\rangle\,\|^2}{\|\,|\psi_{l-1}\rangle\,\|^2} \approx e^{-2\|A\|T}\frac{\|\,|\psi_L\rangle\,\|^2}{\|\,|\psi_0\rangle\,\|^2}.$$

For this method to yield a meaningful result, we need $|\psi_L\rangle \approx |\psi(T)\rangle$, and consequently $\|\,|\psi_L\rangle\,\| \approx \|\,|\psi(T)\rangle\,\|$. The success probability is therefore approximately $e^{-2\|A\|T}\frac{\||\psi(T)\rangle\|^2}{\||\psi(0)\rangle\|^2}$,

and it takes

$$e^{2\|A\|T} \frac{\| \, |\psi(0)\rangle \, \|^2}{\| \, |\psi(T)\rangle \, \|^2}$$

trials for the procedure to succeed with $\Omega(1)$ probability.

To see why this is not a reasonable scaling, let us consider the case where A is anti-Hermitian, which yields the Schrödinger equation, and we have $\| \, |\psi(T)\rangle \, \| = \| \, |\psi(0)\rangle \, \|$. The number of trials needed is therefore $e^{2\|A\|T}$, despite the fact that the usual Hamiltonian simulation algorithms generally succeed in one run!

The problem becomes even worse if we want to implement the time step $e^{AT/L}$ with high accuracy, rather than approximating it with $I + AT/L$. For example, we may consider implementing $e^{AT/L}$ through QSVT, if A is either Hermitian or anti-Hermitian. But the subnormalization factor that comes from QSVT has an extra factor of 2, becoming $2\|e^{AT/L}\|$, due to the summation of the even and odd parts [37, Theorem 56] or the real and imaginary parts [37, Theorem 58]. In the end the number of trials required becomes $4^L \|e^{AT}\|^2 \frac{\||\psi(0)\rangle\|^2}{\||\psi(T)\rangle\|^2}$, which increases exponentially with respect to the number of segments L even for a finite T. In the anti-Hermitian case, we can use the oblivious amplitude amplification (OAA) [38, Theorem 15] to solve this problem, or use the algorithm in [55] to avoid this problem entirely, but both methods work only because of the unitarity of the exact time evolution. For non-unitary dynamics, as OAA is not applicable, we need a different strategy to implement a time-marching based method.

1.4 Organization

The rest of the paper is organized as follows: in Section 2 we provide an overview of the method, presenting our method (Theorem 4) to link up short-time evolutions into a long-time evolution while keeping the success probability from decaying faster than necessary. In Section 3 we will discuss how to implement short-time evolution using the truncated Dyson series algorithm, leading to our first algorithm in Theorem 8. The amplification ratio Q dependence in this algorithm is shown to be optimal in Section 4. In Section 5 we demonstrate the performance of the time-marching based method with a first-order truncated Magnus series, which simplifies the implementation and also exhibits a commutator scaling in the high precision limit.

2 Overview of the method

2.1 Main idea

Let us first revisit the challenge of implementing the Euler method in Section 1.3. The reason that we end up with the exponential overhead $e^{2\|A\|T}$ is that each time step involves a subnormalization factor $1 + \|A\|T/L$. Now let us consider, what if the subnormalization factor is $\|I + AT/L\|$ rather than $1 + \|A\|T/L$? The subnormalization factor cannot be better than this because we need to encode $I + AT/L$ into a unitary matrix that has spectral norm 1. Again assume A is a normal matrix. If the subnormalization factor is indeed $\|I + AT/L\|$, then the $e^{2\|A\|T}$ overhead is replaced by

$$\|I + AT/L\|^{2L} = \|(I + AT/L)^L\|^2 \approx \|e^{TA}\|^2.$$

This is a far more reasonable scaling. For instance, if A is anti-Hermitian, then $\|e^{TA}\| = 1$, rather than exponentially growing with time.

From the above discussion, we can see that the seemingly subtle difference between the subnormalization factors $1 + \|A\|T/L$ and $\|I + AT/L\|$ is in fact crucial. To implement $(I + AT/L)\,|\phi_{l-1}\rangle$ using linear combination of unitaries as discussed in Section 1.3 involves a success probability as described in (6), which can be much smaller than the *intrinsic success probability*:

$$\frac{1}{\|I + AT/L\|^2} \times \frac{\| \, |\psi_l\rangle \, \|^2}{\| \, |\psi_{l-1}\rangle \, \|^2}. \tag{7}$$

The ratio between the intrinsic success probability and Eq. (6) is

$$\gamma^2 = \left(\frac{1 + \|A\|T/L}{\|I + AT/L\|}\right)^2, \tag{8}$$

which comes from excessive subnormalization due to the construction of the block encoding. This excessive factor can be removed through a technique called the uniform singular value amplification [38, Theorem 17]. In a nutshell, the uniform singular value amplification uses an odd polynomial $P(x)$ to approximate a linear function $f(x) = \gamma x$ in an interval $[-\gamma^{-1}, \gamma^{-1}]$ for γ defined in Eq. (8), and satisfies the norm constraint $|P(x)| \leq 1$ for all $x \in [-1, 1]$. The norm constraint is a crucial requirement for applying QSVT with the polynomial P. The effect of the uniform singular value amplification is that it approximately multiplies a factor γ to the encoded operator, which nearly exactly cancels the excessive subnormalization factor. One important feature of the uniform singular value amplification is that it is oblivious to the quantum state we want to act on, i.e., $|\psi_{l-1}\rangle$. This means we do not need to repeatedly prepare quantum states from previous time steps in this amplification procedure, and this is the key to avoiding an exponential overhead. After repeated usage of the uniform singular value amplification, the subnormalization factor scales as $\|e^{TA}\|$ instead of $e^{T\|A\|}$. The same technique also solves the problem with the subnormalization factor being much larger than 1 as discussed in Section 1.3.

As discussed above, for the time-independent case, assuming $\| \, |\psi(0)\rangle \, \| = 1$, we need to run the algorithm for $\|e^{AT}\|^2/\| \, |\psi(T)\rangle \, \|^2$ times to achieve $\Omega(1)$ overall success probability. In the time-dependent case, this factor takes a slightly more complicated form,

$$Q^2 = \frac{\prod_{l=1}^{L} \|\mathcal{T} e^{\int_{t_{l-1}}^{t_l} A(t)\mathrm{d}t}\|^2}{\| \, |\psi(T)\rangle \, \|^2},$$

where $0 = t_0 < t_1 < \cdots < t_L = T$ are determined by our choice of the temporal mesh. This gives the definition of Q in Eq. (4)

Because this Q dependence comes from the number of trials needed in order to complete the whole procedure successfully with high probability, a natural question is whether the dependence can be improved from $\mathcal{O}(Q^2)$ to $\mathcal{O}(Q)$ using amplitude amplification [17]. However, a direct application of amplitude amplification comes at a price. As mentioned above, at each time step, we need to perform measurements on the ancilla qubits, and we need to ensure that the measurement results from all the time steps are correct. The direct application of amplitude amplification requires the usage of different ancilla qubits for each time step, and the number of ancilla qubits needed will scale linearly with respect to the number of time

steps. To reduce the number of ancilla qubits, we employ a simplified version of the compression gadget introduced in [57] to coherently record the success or failure of each time step. This method ensures that the number of ancilla qubits needed to implement amplitude amplification scales only logarithmically in the number of time steps.

2.2 Input Model

To solve the ODE (1), we need to store the information of the coefficient matrix $A(t)$ and the initial state $|\psi(0)\rangle$ in the quantum computer. We assume that we have access to a unitary circuit U_{init} to prepare the initial state, i.e., $U_{\text{init}}|0^n\rangle = |\psi(0)\rangle$, where n is the number of qubits and $2^n = N$. For $A(t)$ this requires some explanation, in particular because we need not just a single matrix but a family of matrices on the time interval $[0, T]$.

A similar problem is encountered in the setting of time-dependent Hamiltonian simulation problem, where a *time-dependent matrix encoding* was proposed in [57] to encode the time-dependent Hamiltonian. We adopt essentially the same idea, and extend it to the non-Hermitian case.

Definition 1 (Time-dependent matrix encoding)**.** *An $(n_q, m, a, b, \alpha, \epsilon)$-*MAT *is a unitary that acts on three registers, each containing n_q, m, and n qubits respectively. It satisfies*

$$(I_{n_q} \otimes \langle 0^m| \otimes I_n)\text{MAT}(I_{n_q} \otimes |0^m\rangle \otimes I_n) = \sum_{\gamma=0}^{2^{n_q}-1} |\gamma\rangle\langle\gamma| \otimes \frac{\widetilde{A}\left((b-a)\frac{\gamma}{2^{n_q}} + a\right)}{\alpha}, \tag{9}$$

where $\|\widetilde{A}(t) - A(t)\| \leq \epsilon$ for $t \in [a, b]$. This unitary MAT *is called the time-dependent matrix encoding of $A(t)$ on the time interval $[a, b]$.*

Here n is the number of qubits corresponding to the system ($2^n = N$), m is the number of ancilla qubits for block encoding, and n_q qubits are used to store the index of the quadrature points. There are 2^{n_q} quadrature points in total.

The total variation of $A(t)$ on the interval $[a, b]$, denoted $V_a^b(A)$, is defined as follows:

$$V_a^b(A) = \sup_{R\in\mathbb{N}} \sup_{a=t_0<t_1<\cdots<t_R=b} \sum_{j=0}^{R-1} \|A(t_{j+1}) - A(t_j)\|. \tag{10}$$

The set of all functions of bounded variation is denoted by

$$BV([a, b]) := \{\, A \mid V_a^b(A) < \infty \,\}. \tag{11}$$

Our algorithm only requires that the total variation $V_0^T(A)$ of the coefficient matrix $A(t)$ is finite, i.e., $A \in BV([0, T])$.

We need to choose n_q to achieve the required accuracy for performing numerical quadrature, and this will be discussed in detail in Section 3.1. We note that for the sparse matrix input model, which is used in other quantum differential equation solvers [8, 14, 26, 48] and time-dependent Hamiltonian simulation algorithms [7], one can construct an efficient time-dependent matrix encoding. Thus, our algorithms also apply to the case of sparse matrices, which will be discussed in more details in Section 3.4.

2.3 Uniform singular value amplification

For a given temporal mesh $0 = t_0 < t_1 < \cdots < t_L = T$, the short time integrator at step l can be abstractly written as

$$|\psi_l\rangle = \bar{\Xi}_l |\psi_{l-1}\rangle, \quad l = 1, \ldots, L, \tag{12}$$

where $\bar{\Xi}_l$ is an operator approximating the exact evolution operator $\Xi_l = \mathcal{T} e^{\int_{t_{l-1}}^{t_l} A(t)\mathrm{d}t}$. We require that $\bar{\Xi}_l$ is consistent with Ξ_l to precision $\epsilon_l \|\Xi_l\|$, i.e.,

$$\|\Xi_l - \bar{\Xi}_l\| \leq \epsilon_l \|\Xi_l\|. \tag{13}$$

We assume that $\bar{\Xi}_l$ is implemented with its $(\alpha_l, m, 0)$-block encoding denoted by U_l, i.e.,

$$\alpha_l(\langle 0^m| \otimes I_n) U_l (|0^m\rangle \otimes I_n) = \bar{\Xi}_l. \tag{14}$$

Due to Eq. (13), U_l can also be viewed as an $(\alpha_l, m_l, \epsilon_l\|\Xi_l\|)$-block encoding of Ξ_l.

In this section we discuss how to approximately implement a series of non-unitary operations Ξ_1, Ξ_2, …, Ξ_L sequentially, and boost the success probability using uniform singular value amplification, which is developed in [38, Theorem 17]. The idea is to linearly amplify the singular values of $\bar{\Xi}_l/\alpha_l$ by a factor that is approximately $\gamma' = \alpha_l / \left\|\bar{\Xi}_l\right\|$. For simplicity, we first assume $\bar{\Xi}_l = \Xi_l$ (i.e., U_l block encodes the exact short time integrator), and study how this technique can help us boost the success probability of a single operation.

Lemma 2 (Uniform singular value amplification, [38, Theorem 17])**.** *Let U be an $(\alpha, m, 0)$-block encoding of Ξ. We can construct a $(\frac{\|\Xi\|}{1-\delta}, m+1, \epsilon\|\Xi\|)$-block encoding $\widetilde{U}$ of Ξ, using $\mathcal{O}(\frac{\alpha}{\delta\|\Xi\|}\log(\frac{\alpha}{\|\Xi\|\epsilon}))$ applications of (controlled-) U and its inverse.*

Proof. Consider the singular value decomposition $\Xi = W\Sigma V^\dagger$, where $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_N)$. Applying QSVT with an odd polynomial gives us the block encoding $\widetilde{U}$ of a new matrix, which up to rescaling is $\widetilde{\Xi} = W\widetilde{\Sigma}V^\dagger$, where $\widetilde{\Sigma} = \mathrm{diag}(\widetilde{\sigma}_1, \widetilde{\sigma}_2, \ldots, \widetilde{\sigma}_N)$. We choose the odd polynomial in the same way as in [38, Theorem 17], and we choose $\gamma = \frac{\alpha(1-\delta)}{\|\Xi\|}$. By these choices all singular values of Ξ are contained in the interval $[0, \gamma'^{-1}] = [0, \frac{1-\delta}{\gamma}]$. If we choose the rescaling factor of $\widetilde{\Xi}$ so that

$$\widetilde{\Xi} = \frac{\|\Xi\|}{1-\delta}(\langle 0^{m+1}| \otimes I_n)\widetilde{U}(|0^{m+1}\rangle \otimes I_n),$$

then by [38, Theorem 17],

$$\left|\frac{\widetilde{\sigma}_j}{\sigma_j} - 1\right| \leq \epsilon,$$

which implies

$$\|\widetilde{\Xi} - \Xi\| = \|\widetilde{\Sigma} - \Sigma\| \leq \epsilon\|\Sigma\| = \epsilon\|\Xi\|.$$

Therefore $\widetilde{U}$ is a $(\frac{\|\Xi\|}{1-\delta}, m+1, \epsilon\|\Xi\|)$-block encoding $\widetilde{U}$ of Ξ. By [38, Theorem 17] it requires $\mathcal{O}(\frac{\alpha}{\delta\|\Xi\|}\log(\frac{\alpha}{\|\Xi\|\epsilon}))$ applications of (controlled-) U and its inverse. □

We can now use Lemma 2 to address the problem discussed in Section 1.3, and for now neglect all errors in the block encodings. The error analysis with the block encoding errors taken into account will be analyzed in Theorem 4. If we directly apply U_l and post-select the measurement results, through the procedure described in Figure 1 without the uniform singular value amplification, then the success probability will be

$$\frac{\|\Xi_L \cdots \Xi_2 \Xi_1 |\psi(0)\rangle\|^2}{(\alpha_1 \alpha_2 \cdots \alpha_L)^2}. \tag{15}$$

For each l, $\|\Xi_l\| \leq \alpha_l$, and the cumulative difference between $\alpha_1 \alpha_2 \cdots \alpha_L$ and $\|\Xi_L\| \cdots \|\Xi_2\| \|\Xi_1\|$ can be quite significant, as discussed for the case of Euler's method in Section 1.3.

With the block encodings $\widetilde{U}_l$ given by the uniform singular value amplification, we can implement $\widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1$ as depicted in Figure 1. We apply each $\widetilde{U}_l$ sequentially, measuring the ancilla qubits after each application, only proceeding when the measurement result is all 0, and otherwise aborting the procedure. The operator $\widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1$ thus implemented will approximate the operator $\Xi_L \cdots \Xi_2 \Xi_1$, which is our goal. With with the choice $\delta = 1/L$, we have

$$(1-\delta)^L \geq e^{-\frac{\delta L}{1-\delta}} = \Omega(1).$$

Therefore the success probability is, up to a constant factor,

$$\frac{\|\Xi_L \cdots \Xi_2 \Xi_1 |\psi(0)\rangle\|^2}{\|\Xi_L\|^2 \cdots \|\Xi_2\|^2 \|\Xi_1\|^2}. \tag{16}$$

By turning the success probability from (15) to (16) using Lemma 2, we address the problem of the vanishing success probability discussed in Section 1.3.

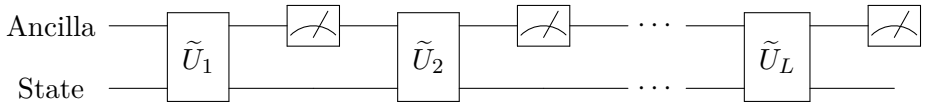
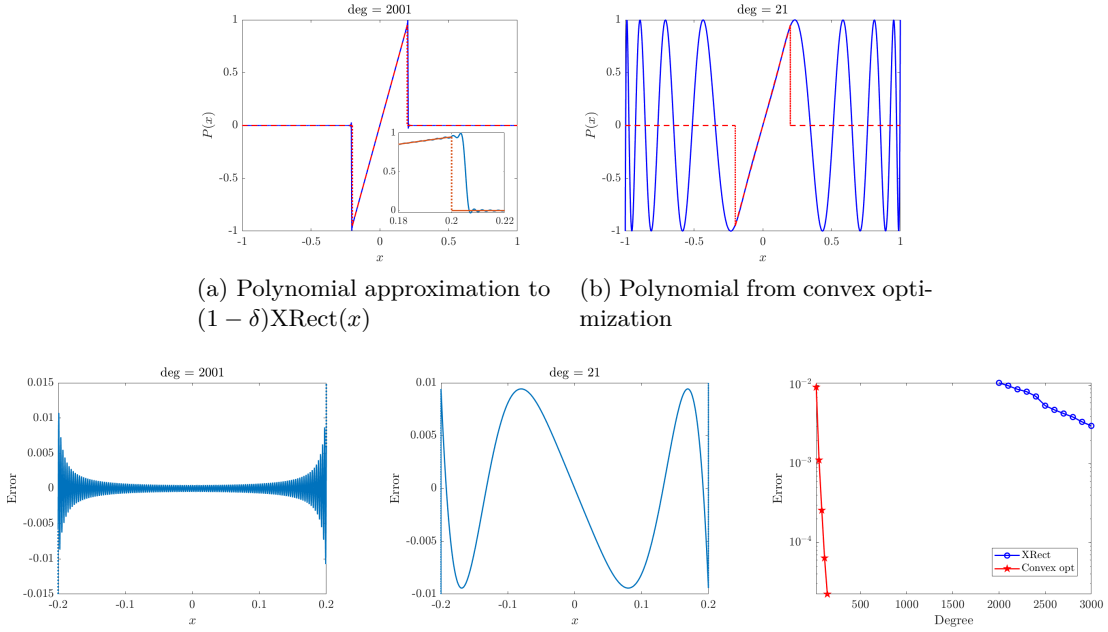


Figure 1: Implementing $\widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1$. After we apply each $\widetilde{U}_l$, we measure the ancilla qubits, and only proceed when the measurement result is all 0, and otherwise abort the procedure.

The uniform singular value amplification procedure was first proposed in [54, Theorem 5]. However, it only constructs an $(2\|\Xi\|, m+1, \epsilon\|\Xi\|)$-block encoding of Ξ, and the extra factor 2 means that the overall success probability of the procedure in Fig. 1 will still decrease exponentially fast as $\mathcal{O}(4^{-L})$. This is not acceptable in the context of this paper. Therefore we adopt the version in [38, Theorem 17], which refines the analysis so that the subnormalization factor is $\|\Xi\|/(1-\delta)$.

In the abstract form, one needs a polynomial that approximates $\gamma' x$ on the desired interval $\mathcal{I} = [-\gamma'^{-1}, \gamma'^{-1}]$. Such a polynomial has been constructed by [38, Theorem 17], which we examine in more detail. It first constructs an even polynomial $q(x)$ that approximates an even "rectangular function" defined as

$$\mathrm{Rect}(x) = \frac{1}{2}(\mathrm{sgn}(\gamma'^{-1} - |x|) + 1), \quad \mathrm{sgn}(x) = \begin{cases} 1, & x > 0 \\ -1, & x < 0 \\ 0, & x = 0 \end{cases}.$$



(a) Polynomial approximation to $(1-\delta)\mathrm{XRect}(x)$

(b) Polynomial from convex optimization

(c) Error of polynomial approximation to $(1-\delta)\mathrm{XRect}(x)$

(d) Error of polynomial from convex optimization

(e) Comparison of convergence speed

Figure 2: Comparison of the polynomials used for uniform singular value amplification procedure. (a) Polynomial approximating $(1-\delta)\mathrm{XRect}(x)$ used in [38, Theorem 17]; (b) Near-optimal polynomial approximation obtained via convex optimization (see Appendix C); (c) (d) Errors of the two methods on the interval $\mathcal{I} = [-\gamma'^{-1}, \gamma'^{-1}]$ with $\gamma' = 5, \delta = 0.05$; (e) Comparison of the convergence speed of the two methods, measured by the L^∞ error $\max_{x\in\mathcal{I}} |p(x) - (1-\delta)\mathrm{XRect}(x)|$.

The desired odd polynomial is $p(x) = \gamma' x q(x)$, which approximates $\mathrm{XRect}(x) := \gamma' x \cdot \mathrm{Rect}(x)$, and agrees with the linear function $\gamma' x$ on the interval $[-\gamma'^{-1}, \gamma'^{-1}]$. The polynomial $p(x)$ should also satisfy the *norm constraint*: $|p(x)| \leq 1$ for all $x \in [-1, 1]$ due to the requirement of QSVT (see [38, Corollary 5]). However, the function $\mathrm{XRect}(x)$ is discontinuous at $x = \gamma'^{-1}$, and therefore a polynomial approximation to $\mathrm{XRect}(x)$ exhibits the Gibbs phenomenon, which states that $p(x)$ always overshoots around $x = \gamma'^{-1}$, even as the polynomial degree increases to infinity (see e.g., [39] and [62, Chapter 9]). An example of the Gibbs phenomenon is shown in the inset of Fig. 2 (a), where the polynomial approximation overshoots $\mathrm{XRect}(x)$. As a result, instead of approximating $\mathrm{XRect}(x)$, we can only find a polynomial that approximates the function $(1 - \delta)\mathrm{XRect}(x)$ to satisfy the norm constraint. It is worth mentioning that although the construction above leads to the desired asymptotic scaling in Lemma 2, the preconstant can be quite large, which leads to high polynomial degrees even for moderate values of the parameters α, δ, ϵ. Fig. 2 (a) shows that for $\gamma' = 5, \delta = 0.05, \epsilon = 0.01$, the required polynomial degree is already as large as 2001. Reducing to a smaller value $\delta = 0.01$ would require a polynomial degree of around 10^4, which may be too large to be practically useful.

To address this issue, we notice that the uniform singular value amplification only requires us to find a polynomial $p(x)$ that approximates $\gamma' x$ on the desired interval $\mathcal{I} = [-\gamma'^{-1}, \gamma'^{-1}]$. Outside this interval $\mathcal{I}$, the value of $p(x)$ can be arbitrary, as long as the norm constraint is satisfied. In particular, $p(x)$ does not need to approximate $\mathrm{XRect}(x)$, which vanishes outside $\mathcal{I}$. This allows us to construct a convex optimization based procedure to numerically identify the near-optimal polynomial approximation for the uniform singular value amplification. The procedure is detailed in Appendix C. For the same parameter setting $\gamma' = 5, \delta = 0.05, \epsilon = 0.01$, the polynomial approximation is given in Fig. 2 (b) and the polynomial degree is merely 21. Fig. 2 (e) further shows that as fixing γ', δ, both methods converge exponentially with respect to the increase of the polynomial degrees. However, the convergence rate of the convex optimization based method is significantly faster, which reduces the number of queries to U_l by orders of magnitude.

2.4 Amplitude amplification using compression gadget

Since the main concern of running the algorithm in Fig. 1 is its success probability, it is natural to consider the usage of amplitude amplification [17] to reduce the number of repetitions needed to obtain a successful outcome. With the current procedure in Fig. 1, however, this results in a large space overhead. Directly applying $\widetilde{U}_l$ (and hence $\widetilde{\Xi}_l$) sequentially involves intermediate measurements to determine whether each $\widetilde{\Xi}_l$ is applied successfully. We need to record the measurement outcome of each of the L steps, and this means we need to duplicate the ancilla register L times to implement amplitude amplification. To avoid this overhead, we need to replace the procedure with a fully coherent one, with measurement performed only at the end. This allows us to reduce the Q dependence from $\mathcal{O}(Q^2)$ to $\mathcal{O}(Q)$.

Let us first formulate the problem in a more abstract way. We have unitaries $V_1, V_2, \ldots, V_L$, each of which is a $(\alpha'_l, m'_l, 0)$-block encoding of a potentially non-unitary operation Γ_l. The goal is to implement $\Gamma_L \cdots \Gamma_2 \Gamma_1$ with amplitude amplification, and without duplicating the ancilla registers.

This goal can be achieved using the *compression gadget* in Fig. 3, following the idea in Ref. [57]. In fact we are using a simplified version of the compression gadget in Ref. [57], as the

problem we are trying to solve is in some way easier. The main idea is to use a counter register to keep track of how many Γ_l's have been applied successfully in a coherent way. This allows us to post-select on the counter register to ensure that all Γ_l's have been applied successfully. This result is summarized in the following lemma. Its proof is given in Appendix D.

Lemma 3 (Compression gadget)**.** *Suppose we are given unitaries $V_1, V_2, \ldots, V_L$, each of which is a $(\alpha_l', m_l', 0)$-block encoding of Γ_l. Then we can construct a $(\alpha_{\mathrm{comp}}, m_{\mathrm{comp}}, 0)$-block encoding of $\Gamma_L \cdots \Gamma_2 \Gamma_1$, where*

$$\alpha_{\mathrm{comp}} = \alpha_1' \alpha_2' \cdots \alpha_L', \quad m_{\mathrm{comp}} = \max_l m_l' + \lceil \log_2(L) \rceil + 1,$$

using one application of each V_l.

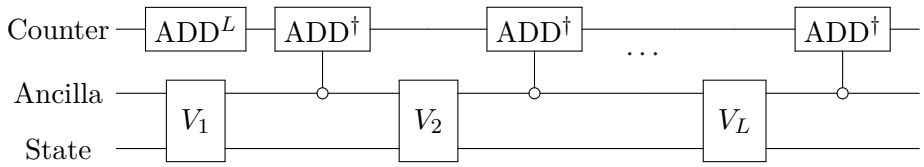


Figure 3: The simplified compression gadget for coherently applying $\Gamma_L \cdots \Gamma_2 \Gamma_1$. The counter register, containing $\lceil \log_2(L) \rceil + 1$ qubits, is used for keeping track of whether each Γ_l has been applied successfully; the ancilla register, containing $\max_l m_l'$ qubits, is for the ancilla qubits needed in V_l's; the state register stored the quantum state on which we want to apply $\Gamma_L \cdots \Gamma_2 \Gamma_1$. ADD implements addition by 1 modulo the smallest power of 2 that is larger than or equal to $2L$. Here each controlled ADD† is controlled on the state $|0^{m_{\max}}\rangle$.

We can now use Lemma 3 to obtain the following result, taking into account the uniform singular value amplification procedure:

Theorem 4 (Coherent implementation of long-time integrator)**.** *Suppose we are given Ξ_l through its $(\alpha_l, m_l, \epsilon_l \|\Xi_l\|)$-block encoding U_l, for $l = 1, 2, \ldots, L$, and $\sum_l \epsilon_l \leq 1/2$, then for any $0 < \epsilon' \leq 1/(2L)$, we can construct an $(\alpha_{\mathrm{comp}}, m_{\mathrm{comp}}, \epsilon_{\mathrm{comp}})$-block encoding of $\Xi_L \cdots \Xi_2 \Xi_1$, where*

$$\frac{\prod_{l=1}^L \|\Xi_l\|}{2(1-\delta)^L} \leq \alpha_{\mathrm{comp}} \leq \frac{e^{1/2} \prod_{l=1}^L \|\Xi_l\|}{(1-\delta)^L}, \tag{17}$$

and

$$m_{\mathrm{comp}} = \max_l m_l + \lceil \log_2(L) \rceil + 2, \quad \epsilon_{\mathrm{comp}} = e^{1/2} \left(L\epsilon' + \sum_l \epsilon_l \right) \prod_{l'=1}^L \|\Xi_{l'}\|. \tag{18}$$

using $\mathcal{O}(\frac{\alpha_l}{\delta \|\Xi_l\|} \log(\frac{\alpha_l}{\|\Xi_l\| \epsilon'}))$ applications of each (controlled-) U_l and its inverse.

Proof. We denote by $\bar{\Xi}_l$ the matrix that is exactly encoded in each U_l as in Eq. (14). Using Lemma 2, we first construct a $(\frac{\|\bar{\Xi}_l\|}{1-\delta}, m_l + 1, \epsilon' \|\bar{\Xi}_l\|)$-block encoding $\widetilde{U}_l$ of each $\bar{\Xi}_l$. Each $\bar{\Xi}_l$ uses U_l $\mathcal{O}(\frac{\alpha_l}{\delta \|\Xi_l\|} \log(\frac{\alpha_l}{\|\Xi_l\| \epsilon'}))$ times. Here we have used the fact that $\|\bar{\Xi}_l\| = \Theta(\|\Xi_l\|)$ because of Eq. (13). We denote by $\widetilde{\Xi}_l$ the matrix that is exactly encoded in $\widetilde{U}_l$, i.e.,

$$\frac{\|\bar{\Xi}_l\|}{1-\delta} (\langle 0^{m_l+1}| \otimes I_n) \widetilde{U}_l (|0^{m_l+1}\rangle \otimes I_n) = \widetilde{\Xi}_l.$$

Then

$$\|\bar{\Xi}_l - \widetilde{\Xi}_l\| \leq \epsilon' \|\bar{\Xi}_l\|. \tag{19}$$

Next, we use Lemma 3 to combine $\widetilde{U_l}$'s into a block encoding of $\Xi_L \cdots \Xi_2 \Xi_1$. Directly applying Lemma 3 yields an $(\alpha_{\text{comp}}, m_{\text{comp}}, 0)$-block encoding of $\widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1$, which we denote by U_{comp}, where

$$\alpha_{\text{comp}} = \frac{\prod_{l=1}^{L} \|\bar{\Xi}_l\|}{(1-\delta)^L}, \quad m_{\text{comp}} = \max_l m_l + \lceil \log_2(L) \rceil + 2.$$

Noting the fact that

$$\prod_{l=1}^{L}(1-\epsilon_l) \geq 1 - \sum_l \epsilon_l \geq 1/2, \quad \prod_{l=1}^{L}(1+\epsilon_l) \leq e^{\sum_l \epsilon_l} \leq e^{1/2},$$

we can get the upper and lower bounds for α_{comp} through

$$\frac{\prod_{l=1}^{L} \|\Xi_l\| \prod_{l=1}^{L}(1-\epsilon_l)}{(1-\delta)^L} \leq \alpha_{\text{comp}} \leq \frac{\prod_{l=1}^{L} \|\Xi_l\| \prod_{l=1}^{L}(1+\epsilon_l)}{(1-\delta)^L}$$

To bound the error between $\widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1$ and $\Xi_L \cdots \Xi_2 \Xi_1$, we have

$$\begin{aligned}
&\|\Xi_L \cdots \Xi_2 \Xi_1 - \widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1\| \\
\leq &\|\Xi_L \cdots \Xi_2 \Xi_1 - \bar{\Xi}_L \cdots \bar{\Xi}_2 \bar{\Xi}_1\| + \|\bar{\Xi}_L \cdots \bar{\Xi}_2 \bar{\Xi}_1 - \widetilde{\Xi}_L \cdots \widetilde{\Xi}_2 \widetilde{\Xi}_1\| \\
\leq &\sum_{l=1}^{L} \prod_{l'=l+1}^{L} \|\Xi_{l'}\| \prod_{r=1}^{l-1} \|\bar{\Xi}_r\| \|\Xi_l - \bar{\Xi}_l\| + \sum_{l=1}^{L} \prod_{l'=l+1}^{L} \|\bar{\Xi}_{l'}\| \prod_{r=1}^{l-1} \|\widetilde{\Xi}_r\| \|\bar{\Xi}_l - \widetilde{\Xi}_l\| \\
\leq &\sum_l \epsilon_l \prod_{l'=1}^{L} (\|\Xi_{l'}\|(1+\epsilon_l)) + L\epsilon' \prod_{l'=1}^{L} (\|\bar{\Xi}_l\|(1+\epsilon')) \\
\leq &e^{1/2} \left(\sum_l \epsilon_l \prod_{l'} \|\Xi_{l'}\| + L\epsilon' \prod_l \|\Xi_l\| \right),
\end{aligned}$$

where for the second inequality we have used Eqs. (13) and (19), and for the last inequality we have used $\prod_{l=1}^{L}(1+\epsilon_l) \leq e^{\sum_l \epsilon_l} \leq e^{1/2}$ as well as $(1+\epsilon')^L \leq e^{L\epsilon'} \leq e^{1/2}$. Therefore we can choose ϵ_{comp} as in the statement of the corollary.

□

With a coherent implementation of the long-time integrator given as a block encoding in Theorem 4, we can readily apply the standard amplitude amplification to boost the success probability and yield a quadratic speedup in terms of the query complexity.

3 High-order truncated Dyson series approach

In this section, we analyze the method described in Section 2, when the short time integrator is implemented using the high-order truncated Dyson series. In Section 3.1 we discuss how to implement the short time integrator developed in Ref. [57]. Then we use the tools developed in

Sections 2.3 and 2.4 to link different segments of short time evolution into long time evolution in Section 3.2. We analyze the success probability in Section 3.3. Our time-marching based strategy can be combined with any input models such as the sparse matrix model [7, 26, 48, 57] and the linear combination of unitaries (LCU) model [7, 57]. We discuss in particular the implementation with a sparse matrix input model in Section 3.4.

3.1 Short time evolution

The truncated Dyson series method implements $\Xi = \mathcal{T}e^{\int_a^b A(s)\mathrm{d}s}$ through

$$\mathcal{T}e^{\int_a^b A(s)\mathrm{d}s} \approx \sum_{k=0}^{K-1} \int_a^b \mathrm{d}s_1 \int_a^{s_1} \mathrm{d}s_2 \cdots \int_a^{s_{k-1}} \mathrm{d}s_k A(s_1)A(s_2)\cdots A(s_k).$$

This infinite series can be truncated at order K, and the error is upper bounded by $(\int_a^b \|A(s)\|\mathrm{d}s)^K/K!$. Therefore if we choose a and b so that $\int_a^b \|A(s)\|\mathrm{d}s = \mathcal{O}(1)$, we can achieve high accuracy with only a few terms.

In order to be robust against error in the coefficient matrix, we need an additional step in the error analysis of the algorithm. If $A(t)$ is accessed through an $(n_q, m, a, b, \alpha, \epsilon)$-MAT as discussed in Section 2.2, then the above approach will yield a block encoding of a different time evolution operator $\bar{\Xi} \approx \mathcal{T}e^{\int_a^b \widetilde{A}(t)\mathrm{d}t}$, where $\widetilde{A}(t)$ is the time-dependent matrix encoded in MAT exactly, as defined in Eq. (9). By Lemma 16, and assume that $(b-a)\alpha = \mathcal{O}(1)$, then

$$\|\Xi - \bar{\Xi}\| \leq e^{(b-a)\max_{u\in[a,b]}\{\|A(u)\|,\|\widetilde{A}(u)\|\}} \int_a^b \|\widetilde{A}(u) - A(u)\|\mathrm{d}u = \mathcal{O}(\epsilon(b-a)).$$

With this additional step, using the same algorithm as in [57, Theorem 3], we can encode Ξ with the following costs:

Lemma 5 (Short-time evolution through truncated Dyson series)**.** *Suppose $A \in BV([a,b])$, and is accessed through an $(n_q, m, a, b, \alpha, \epsilon)$-*MAT *as defined in Definition 1 for some $\epsilon \leq \epsilon'/(2(b-a))$. $b - a \leq (2\alpha)^{-1}$, $2^{n_q} = \Theta(\frac{1}{\epsilon'}((b-a)V_a^b(A) + 1))$, where $V_a^b(A)$ is the total variation of $A(t)$ on the interval $[a,b]$. Then we can construct an (α', m', ϵ')-block encoding of $\Xi = \mathcal{T}e^{\int_a^b A(t)\mathrm{d}t}$ using $\mathcal{O}(\frac{\log(\epsilon'^{-1})}{\log\log(\epsilon'^{-1})})$ queries to* MAT*. Here $\alpha' = \mathcal{O}(1)$ and $m' = \mathcal{O}(m + n_q)$. We also use $\mathcal{O}(m + n_q + \mathrm{polylog}(\alpha\epsilon'^{-1}))$ additional elementary gates.*

In [57, Theorem 3], n_q is chosen to satisfy

$$2^{n_q} = \Theta\left(\frac{1}{\epsilon'}\left((b-a)\int_a^b \|\dot{A}(t)\|\mathrm{d}t + (b-a)^2 \max_{t\in[a,b]} \|A(t)\|^2\right)\right).$$

The discussion on the numerical quadrature error in Appendix F shows that we can further relax the regularity condition so that $\int_a^b \|\dot{A}(t)\|\mathrm{d}t$ can be replaced by $V_a^b(A)$ defined in Eq. (10).

3.2 Block encoding of the long time evolution operator

We choose t_l's so that $t_l - t_{l-1} \leq (2\alpha)^{-1}$. Consequently the total number of segments are $L = \Theta(\alpha T)$. For each segment $[t_{l-1}, t_l]$, we construct a time-dependent matrix encoding of

$A(t)$ using the sparse matrix oracles in (27), and then implement the short time evolution operator $\Xi_l = \mathcal{T} e^{\int_{t_{l-1}}^{t_l} A(t)\mathrm{d}t}$. By Lemma 5, this procedure yields an $(\alpha_l, m_l, \epsilon_l \|\Xi_l\|)$-block encoding of Ξ_l, which we denote by U_l. Since

$$e^{-1/2} \leq e^{-\int_{t_{l-1}}^{t_l} \|A(t)\|\mathrm{d}t} \leq \|\Xi_l\| \leq e^{\int_{t_{l-1}}^{t_l} \|A(t)\|\mathrm{d}t} \leq e^{1/2},$$

the number of queries MAT is $\mathcal{O}\left(\frac{\log(\|\Xi_l\|^{-1}\epsilon_l^{-1})}{\log\log(\|\Xi_l\|^{-1}\epsilon_l^{-1})}\right) = \mathcal{O}\left(\frac{\log(\epsilon_l^{-1})}{\log\log(\epsilon_l^{-1})}\right)$. Each $\alpha_l = \mathcal{O}(1)$, and each m_l satisfies

$$\begin{aligned} m_l &= \mathcal{O}\left(m + \max_l \log\left(\frac{1}{\epsilon_l}\left((t_l - t_{l-1}) V_{t_{l-1}}^{t_l}(A) + 1\right)\right)\right) \\ &\leq \mathcal{O}\left(m + \max_l \log\left(\frac{1}{\epsilon_l}\left(V_0^T(A)/\alpha + 1\right)\right)\right). \end{aligned}$$

We need $\mathcal{O}\left(m + \max_l \log\left(\frac{1}{\epsilon_l}\left(V_0^T(A)/\alpha + 1\right)\right)\right)$ additional elementary gates and additional $m' = \mathcal{O}\left(m + \max_l \log\left(\frac{1}{\epsilon_l}\left(V_0^T(A)/\alpha + 1\right)\right)\right)$ ancilla qubits as a working register. The working register starts in state $|0^{m'}\rangle$ and will be returned to $|0^{m'}\rangle$ at each time step, and can therefore be reused.

We introduce the shorthand notation

$$P = \prod_{l=1}^{L} \|\Xi_l\|, \tag{20}$$

and use Theorem 4 to piece together the short time integrators into a block encoding approximating the long time evolution $\mathcal{T} e^{\int_0^T A(t)\mathrm{d}t} = \Xi_L \cdots \Xi_2 \Xi_1$. This yields an $(\alpha_{\mathrm{comp}}, m_{\mathrm{comp}}, \epsilon_{\mathrm{comp}})$-block encoding of $\mathcal{T} e^{\int_0^T A(t)\mathrm{d}t}$, where

$$\frac{P}{2(1-\delta)^L} \leq \alpha_{\mathrm{comp}} \leq \frac{e^{1/2} P}{(1-\delta)^L},$$

$$m_{\mathrm{comp}} = \lceil \log_2(L) \rceil + \mathcal{O}\left(m + \max_l \log\left(\frac{1}{\epsilon_l}\left(V_0^T(A)/\alpha + 1\right)\right)\right),$$

and

$$\epsilon_{\mathrm{comp}} = e^{1/2}\left(L\epsilon' + \sum_l \epsilon_l\right) P. \tag{21}$$

In this block encoding we use each U_l $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\epsilon'}\right)\right)$ times.

We now choose $\delta = 1/(2L)$, and $\epsilon_l = \mathcal{O}(\epsilon_{\mathrm{comp}}/(LP))$, $\epsilon' = \mathcal{O}(\epsilon_{\mathrm{comp}}/(LP))$. Also recall that $L = \mathcal{O}(\alpha T)$. With this choice of parameters we have

$$\alpha_{\mathrm{comp}} = \Theta(P) = \Theta\left(\prod_{l=1}^{L} \|\Xi_l\|\right), \tag{22}$$

and

$$m_{\mathrm{comp}} = \lceil \log_2(\alpha T) \rceil + \mathcal{O}\left(m + \log\left(\frac{(V_0^T(A) + \alpha) T P}{\epsilon_{\mathrm{comp}}}\right)\right). \tag{23}$$

Each U_l is used $\mathcal{O}\left(\alpha T \log\left(\frac{\alpha TP}{\epsilon_{\text{comp}}}\right)\right)$ times. Given the fact that each U_l uses queries to MAT $\mathcal{O}\left(\frac{\log(\epsilon_l^{-1})}{\log\log(\epsilon_l^{-1})}\right)$ times, and that there are $L = \mathcal{O}(\alpha T)$ of them, the total number of queries to MAT are

$$\mathcal{O}\left(\alpha^2 T^2 \frac{\log(\alpha T \prod_{l'=1}^{L} \|\Xi_{l'}\|/\epsilon_{\text{comp}})}{\log\log(\alpha T \prod_{l'=1}^{L} \|\Xi_{l'}\|/\epsilon_{\text{comp}})}\right). \tag{24}$$

We summarize the result of the construction of the block encoding for the long-time evolution using truncated Dyson series as follows.

Theorem 6 (Block encoding of long-time Dyson series evolution)**.** *We assume $A \in BV([0,T])$, and $V_0^T(A)$ is its total variation on $[0,T]$. Suppose A is accessed through an $(n_q, m, a, b, \alpha, \epsilon)$-*MAT *as defined in Definition 1. Let $0 = t_0 < t_1 < \cdots < t_L = T$ satisfy $t_l - t_{l-1} \leq 1/(2\alpha)$. We denote*

$$\Xi_l = \mathcal{T} e^{\int_{t_{l-1}}^{t_l} A(t)\mathrm{d}t}, \quad P = \prod_{l=1}^{L} \|\Xi_l\|.$$

Then we can construct an $(\alpha_{\text{comp}}, m_{\text{comp}}, \epsilon_{\text{comp}})$-block encoding of $\mathcal{T} e^{\int_0^T A(t)\mathrm{d}t}$, where α_{comp}, m_{comp} and ϵ_{comp} are given in Eqs. (21) to (23), respectively. In this block encoding the number of times we need to use:

- *queries to* MAT *as given by Eq. (24);*
- $m' = \mathcal{O}\left(m + \text{polylog}(V_0^T(A) dTP \epsilon_{\text{comp}}^{-1})\right)$ *additional ancilla qubits that start in, and will be returned to $|0^{m'}\rangle$;*
- $\mathcal{O}\left(\alpha^2 T^2 \left(m + \text{polylog}(V_0^T(A) dTP \epsilon_{\text{comp}}^{-1})\right)\right)$ *additional elementary gates.*

3.3 Success probability and main result for Dyson series approach

We can now apply the block encoding to an initial state $|\psi(0)\rangle$ to get the final state $|\psi(T)\rangle = \mathcal{T} e^{\int_0^T A(t)\mathrm{d}t} |\psi(0)\rangle$. We assume that $\| |\psi(0)\rangle \| = 1$. Directly applying the block encoded time evolution operator will introduce an error, and we want to control the resulting error in the final normalized state. This can be done through the following lemma:

Lemma 7. *If $\| |\psi\rangle - |\phi\rangle \| \leq \frac{1}{2} \| |\psi\rangle \|$, then*

$$\left\| \frac{|\psi\rangle}{\| |\psi\rangle \|} - \frac{|\phi\rangle}{\| |\phi\rangle \|} \right\| \leq \frac{4 \| |\psi\rangle - |\phi\rangle \|}{\| |\psi\rangle \|}.$$

Proof. Let $|R\rangle = |\psi\rangle - |\phi\rangle$. Then

$$\begin{aligned} \left\| \frac{|\psi\rangle}{\| |\psi\rangle \|} - \frac{|\phi\rangle}{\| |\phi\rangle \|} \right\| &\leq \frac{\| |\psi\rangle - |\phi\rangle \|}{\| |\psi\rangle \|} + \| |\phi\rangle \| \left| \frac{1}{\| |\psi\rangle \|} - \frac{1}{\| |\phi\rangle \|} \right| \\ &\leq \frac{\| |R\rangle \|}{\| |\psi\rangle \|} + \frac{(\| |\psi\rangle \| + \| |R\rangle \|) \| |R\rangle \|}{\| |\psi\rangle \| (\| |\psi\rangle \| - \| |R\rangle \|)} \\ &\leq \frac{4 \| |R\rangle \|}{\| |\psi\rangle \|}. \end{aligned} \tag{25}$$

□

We first use Theorem 6 to construct an $(\alpha_{\text{comp}}, m_{\text{comp}}, \epsilon_{\text{comp}})$-block encoding of $\Xi = \mathcal{T}e^{\int_0^T A(t)\mathrm{d}t}$. We denote this block encoding by W. Let $\widetilde{\Xi}$ be the time evolution operator that is exactly encoded in W. Then if we successfully prepare the state $|\widetilde{\psi}(T)\rangle = \widetilde{\Xi}\,|\psi(0)\rangle$ by applying the block encoding and measuring the ancilla qubits, the error will be

$$\| \, |\psi(T)\rangle - |\widetilde{\psi}(T)\rangle \, \| \leq \|\Xi - \widetilde{\Xi}\| \leq \epsilon_{\text{comp}}.$$

Therefore, by Lemma 7, in order to ensure that the error in the normalized state is upper bounded by ϵ, i.e.,

$$\left\| \frac{|\psi(T)\rangle}{\| \, |\psi(T)\rangle \, \|} - \frac{|\widetilde{\psi}(T)\rangle}{\| \, |\widetilde{\psi}(T)\rangle \, \|} \right\| \leq \epsilon, \tag{26}$$

it suffices to choose $\epsilon_{\text{comp}} = \mathcal{O}(\epsilon \| \, |\psi(T)\rangle \, \|)$.

Upon measuring the ancilla qubits, if all measurement outcomes are 0, then we have successfully prepared the state $|\widetilde{\psi}(T)\rangle$ that approximates the exact solution $|\psi(T)\rangle$. This happens with probability that is at least $\| \, |\widetilde{\psi}(T)\rangle \, \|^2/\alpha_{\text{comp}}^2$. Using the scaling of α_{comp} in Eq. (22), and the fact that $\| \, |\widetilde{\psi}(T)\rangle \, \| = (1+\mathcal{O}(\epsilon))\| \, |\psi(T)\rangle \, \|$, the success probability is $\Omega(Q^{-2})$, where

$$Q = \frac{\prod_{l=1}^{L} \|\Xi_l\|}{\| \, |\psi(T)\rangle \, \|}$$

is the same as that defined in Eq. (4). Suppose that the initial state $|\psi(0)\rangle$ can be prepared using a unitary circuit U_{init}, then we can boost the success probability to 2/3 with $\mathcal{O}(Q)$ rounds of amplitude amplification. With the above analysis, we can determine the cost of our algorithm, which we state in the following corollary.

Theorem 8 (Time-marching based solver using truncated Dyson series)**.** *Let $|\psi(t)\rangle$ be the solution to the problem in Eq. (1). Suppose we have a unitary circuit U_{init} that satisfies $U_{\text{init}}\,|0^n\rangle = |\psi(0)\rangle$. For coefficient matrix $A \in BV([0,T])$, suppose $0 = t_0 < t_1 < \ldots < t_L = T$. For each segment $[t_{l-1}, t_l]$ we have a time-dependent matrix encoding of $A(t)$ denoted as MAT_l that is an $(n_q, m, t_{l-1}, t_l, \alpha, \epsilon'')-\text{MAT}$ as defined in Definition 1 for some $\epsilon'' < \epsilon/(2TQ)$, and $t_l - t_{l-1} \leq (2\alpha)^{-1}$ for all l. Then we can prepare, with probability at least 2/3, a quantum state $|\widetilde{\psi}(T)\rangle$ that satisfies*

$$\left\| \frac{|\psi(T)\rangle}{\| \, |\psi(T)\rangle \, \|} - \frac{|\widetilde{\psi}(T)\rangle}{\| \, |\widetilde{\psi}(T)\rangle \, \|} \right\| = \mathcal{O}(\epsilon),$$

using

$$\mathcal{O}\left(\alpha^2 T^2 Q \log(\alpha T Q) \frac{\log(\alpha T Q \epsilon^{-1})}{\log\log(\alpha T Q \epsilon^{-1})}\right)$$

queries to all MAT_l, and $\mathcal{O}(Q)$ applications of (controlled-) U_{init} and its inverse. Here Q is defined in Eq. (4). In total we use $\mathcal{O}(n + m + \text{polylog}(V_0^T(A)\alpha T Q \epsilon^{-1}))$ qubits, and $\widetilde{\mathcal{O}}(\alpha^2 T^2 Q(m + \text{polylog}(V_0^T(A)\alpha T Q \epsilon^{-1})))$ additional elementary gates. Success is flagged by the measurement result of a qubit.

3.4 Application to sparse matrix input model

In this section, we discuss the complexity of our time-marching strategy using the sparse matrix input model. The same input model has also been used in other QLSA-based differential equation solvers [8, 14, 26, 48] as well as time-dependent Hamiltonian simulation algorithms [7]. We assume that $A(t)$ is a d-sparse matrix with $\|A(t)\| \leq 1$, and that the locations of non-zero elements are time-independent. The information of $A(t)$ is given through the following oracles:

$$\begin{aligned} U_{\text{row}} |j,s\rangle &= |j, \text{row}(j,s)\rangle , \\ U_{\text{col}} |j,s\rangle &= |j, \text{col}(j,s)\rangle , \\ U_{\text{val}} |t,j,k,z\rangle &= |t,j,k,z \oplus A_{jk}(t)\rangle . \end{aligned} \tag{27}$$

Here $\text{row}(j,s)$ is the row index of the sth nonzero element in the jth column, $\text{col}(j,s)$ is the column index of the sth nonzero element in the jth row. We can use [37, Lemma 48] to construct MAT in Definition 1 using the sparse matrix oracles in Eq. (27). To construct a $(n_q, m, a, b, \alpha, \epsilon)$-MAT we need a single query to both U_{row} and U_{col}, and two queries to U_{val}. The precision parameter ϵ can be made arbitrarily small, but to keep the error below ϵ we need $\mathcal{O}(n + \log^{5/2}(d\epsilon^{-1}))$ additional elementary gates and $\mathcal{O}(n_b + \log^{5/2}(d\epsilon^{-1}))$ additional ancilla qubits are needed, where n_b is the number of bits to encode the binary $A_{jk}(t)$. These additional ancilla qubits can be reused.

Corollary 9 (Time-marching based solver using truncated Dyson series with sparse matrix input model)**.** *Under the same assumptions in Theorem 8, together with the assumption that $A(t)$ is a d-sparse coefficient matrix given by unitaries U_{row}, U_{col}, and U_{val} in Eq. (27)we can prepare, with probability at least $2/3$, a quantum state $|\widetilde{\psi}(T)\rangle$ solving the problem in Eq. (1) that satisfies*

$$\left\| \frac{|\psi(T)\rangle}{\| |\psi(T)\rangle \|} - \frac{|\widetilde{\psi}(T)\rangle}{\| |\widetilde{\psi}(T)\rangle \|} \right\| = \mathcal{O}(\epsilon),$$

using $\mathcal{O}\left(d^2T^2Q \log(dTQ) \frac{\log(dTQ\epsilon^{-1})}{\log\log(dTQ\epsilon^{-1})}\right)$ applications of (controlled-) U_{row}, U_{col}, and U_{val} and their inverses, and $\mathcal{O}(Q)$ applications of (controlled-) U_{init} and its inverse. In total we use $\mathcal{O}(n+\text{polylog}(V_0^T(A)dTQ\epsilon^{-1}))$ qubits, and $\widetilde{\mathcal{O}}(d^2T^2Q(n+\text{polylog}(V_0^T(A)dTQ\epsilon^{-1})))$ additional elementary gates. Success is flagged by the measurement result of a qubit.

4 Optimality of the query complexity with respect to Q

In this section we show that the time-marching based solver using the high-order truncated Dyson series achieves the nearly optimal query complexity with respect to the amplification ratio Q. The optimality is guaranteed by the following lower bound result. Although we state this lower bound result in terms of a time-independent ODE, it automatically provides a lower bound for the harder problem of solving a time-dependent ODE in Eq. (1). The block encoding of the coefficient matrix A also automatically provides a time-dependent matrix encoding needed in Theorem 8.

Theorem 10. *For any given N, there exists a matrix $A \in \mathbb{C}^{N\times N}$ that can be accessed through its $(1, m, 0)$-block encoding U_A, a target time $T = \mathcal{O}(\log(N))$, and an initial state $|\psi(0)\rangle \in \mathbb{C}^{N\times N}$ with $\| |\psi(0)\rangle \| = 1$ such that the following statement holds: Let $|\psi(t)\rangle$ be the*

solution of the ODE $\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = A|\psi(t)\rangle$ *with* $0 \le t \le T$*, i.e.,* $|\psi(T)\rangle = e^{AT}|\psi(0)\rangle$*. Then for any* $\theta > 0$*, there is no quantum algorithm that can prepare a quantum state* ρ *(allowed to be a mixed state) with* $\mathcal{D}(|\psi(T)\rangle, \rho) \le 1/2$*, which uses* $\mathcal{O}(Q^{1-\theta}\mathrm{poly}(T))$ *queries to* U_A*. Here* $\mathcal{D}(\cdot,\cdot)$ *denotes the trace distance between two quantum states and*

$$Q = \sup_L \sup_{0=t_0<t_1<\cdots<t_L=T} \frac{\prod_{l=0}^{L-1} \|e^{A(t_{l+1}-t_l)}\|}{\| |\psi(T)\rangle \|}. \tag{28}$$

Proof. We assume towards contradiction that such an algorithm exists. Now we apply this hypothetical algorithm to solve the unstructured search problem, in which we are asked to find a marked binary string targ of length n. We denote $N = 2^n$. The target targ is marked by the following oracle U_{targ}:

$$U_{\mathrm{targ}}|\mathrm{targ}\rangle = -|\mathrm{targ}\rangle$$
$$U_{\mathrm{targ}}|x\rangle = |x\rangle, \ x \neq \mathrm{targ}.$$

Here x is any binary string of length n that is different from targ.

We let $A = -U_{\mathrm{targ}}$, then $-U_{\mathrm{targ}}$ is an $(1,0,0)$-block encoding of A. Solving the ODE up to time T yields us a (unnormalized) state

$$|\psi(T)\rangle = \frac{1}{\sqrt{N}}\left(\sum_{x\neq\mathrm{targ}} e^{-T}|x\rangle + e^{T}|\mathrm{targ}\rangle\right).$$

Note that in this quantum state, the amplitude corresponding to the marked element targ is amplified, while the amplitudes corresponding to all other elements are suppressed. Consequently, the probability getting targ when measuring the state in the computational basis increases with time. At time T, this probability is

$$\Pr_{|\psi(T)\rangle}(\mathrm{targ}) = \frac{|\langle\psi(T)|\mathrm{targ}\rangle|^2}{\langle\psi(T)|\psi(T)\rangle} = \frac{1}{(N-1)e^{-4T}+1}.$$

If we set $T = \frac{1}{4}\log(3(N-1))$, then the above probability will be $3/4$. Suppose we can prepare a state ρ with $\mathcal{D}(\rho, |\psi(T)\rangle) \le 1/2$, then

$$\Pr_\rho(\mathrm{targ}) \ge \Pr_{|\psi(T)\rangle}(\mathrm{targ}) - \mathcal{D}(\rho, |\psi(T)\rangle) \ge 3/4 - 1/2 = 1/4.$$

As a result, once we prepare a copy of ρ, we can measure in the computational basis, and with probability at least $1/4$ we will get the marked element targ. We can use one application of U_{targ} to check whether the measurement output is targ. Therefore in order to find targ with probability $2/3$, we only need $\mathcal{O}(1)$ copies of ρ.

Let us then look at the cost of preparing ρ. With the hypothetical algorithm for solving the ODE, we can prepare ρ using $\mathcal{O}(Q^{1-\theta}\mathrm{poly}(T))$ queries to U_A. Here $T = \frac{1}{4}\log(3(N-1))$ as chosen above, and in this scenario

$$Q = \sup_L \sup_{0=t_0<t_1<\cdots<t_L=T} \frac{\prod_{l=0}^{L-1} \|e^{A(t_{l+1}-t_l)}\|}{\| |\psi(T)\rangle \|} = \frac{\|e^{TA}\|}{\| |\psi(T)\rangle \|} = \sqrt{\frac{N}{(N-1)e^{-4T}+1}} = \sqrt{\frac{4N}{3}}.$$

Therefore, we need $\mathcal{O}(N^{\frac{1-\theta}{2}}\mathrm{polylog}(N)) = o(\sqrt{N})$ queries to prepare a single copy of ρ. Since only $\mathcal{O}(1)$ copies are needed, we would then be able to solve the unstructured search problem using only $o(\sqrt{N})$ queries to the oracle U_{targ}. This contradicts the lower bound for the unstructured search problem [6, Corollary 3.5]. □

It is worth noting that the lower bound result in Theorem 10 does not imply that the dependence on Q cannot be improved for specific instances of A. For example, consider the following 2×2 non-diagonalizable matrix

$$A = \begin{pmatrix} i & 1 \\ 0 & i \end{pmatrix}. \tag{29}$$

Direct calculation shows that $\left\| e^{AT} \right\|$, and therefore $\| |\psi(T)\rangle \|$ grows linearly in T. However,

$$\sup_{L} \sup_{0=t_0<t_1<\cdots<t_L=T} \prod_{l=0}^{L-1} \| e^{A(t_{l+1}-t_l)} \| \tag{30}$$

grows exponentially in T, and so is Q in Eq. (28). Applying the result of [48], we know that the cost of the optimal solver should grow as $\mathrm{poly}(T)$ for any T. Therefore our time-marching based algorithm is suboptimal in this case. Note that if A is a normal matrix, the quantity in Eq. (30) is equal to $\left\| e^{AT} \right\|$, and this issue does not arise.

5 Simplified implementation and first-order truncated Magnus series

The time-marching strategy can be paired with any reasonable short-time integrators. The implementation of the high-order truncated Dyson series algorithms requires complicated quantum control logic for handling time-ordering operators. In this section, we investigate the performance of time-marching based algorithms with low-order integrators that can be implemented without the explicit treatment of time-ordering operators. The simplest example is a first-order truncated Dyson series algorithm:

$$\Xi = \mathcal{T} e^{\int_{t_j}^{t_{j+1}} A(s)\mathrm{d}s} \approx I + \int_{t_j}^{t_{j+1}} A(s)\mathrm{d}s. \tag{31}$$

We can approximate the integral using numerical quadrature as in the implementation of the high-order truncated Dyson series.

A closely related first-order integrator takes the form

$$\Xi = \mathcal{T} e^{\int_{t_j}^{t_{j+1}} A(s)\mathrm{d}s} \approx e^{\int_{t_j}^{t_{j+1}} A(s)\mathrm{d}s}, \tag{32}$$

i.e., we perform the matrix exponentiation of the time-independent matrix $\int_{t_j}^{t_{j+1}} A(s)\mathrm{d}s$ directly without further Taylor expansion. This is the first-order truncated Magnus series for approximating the time-ordered integration. In the context of Hamiltonian simulation, this strategy gives rise to the quantum highly oscillatory protocol (qHOP) [4], which exhibits commutator scaling in high-precision limit, can be insensitive to the norm and the variation of $A(t)$, and can lead to second-order convergence (i.e., superconvergence) for certain $A(t)$. The rest of the section analyzes the scheme (32) for solving the ODE (1). We verify that this scheme also exhibits the desired commutator scaling. An extreme example where this is useful is when $A(t)$ commutes with itself at any time, then (32) becomes exact, which leads to improved accuracy and poly-logarithmic dependence in the query complexity on the precision. For general non-commuting case, we can use higher order truncation of the Magnus series to improve the accuracy dependence while keeping the commutator scaling. However, the implementation of the quantum circuit can be more complicated.

5.1 Short-time evolution description

For each short time evolution, the time-ordered evolution operator is approximated by truncating its first-order Magnus expansion (32), which is equivalent to directly ignoring the time-ordering operator. The integral can be further approximated using standard Riemann sum with M quadrature points ($M = 2^{n_q}$) as

$$\int_a^b A(s)\mathrm{d}s \approx \frac{b-a}{M} \sum_{k=0}^{M-1} A\left(a + k(b-a)/M\right). \tag{33}$$

The short-time first-order Magnus evolution operator, denoted as $\bar{\Xi}$ is thus given as

$$\bar{\Xi} = e^{\frac{b-a}{M}\sum_{k=0}^{M-1} A(a+k(b-a)/M)}. \tag{34}$$

The implementation of the short-time first-order Magnus evolution operator in two steps. The first step is to construct the block encoding of (33) via applying $\otimes_q$HAD on the n_q qubits where HAD represents the single qubit Hadamard gate, applying MAT and then uncomputing, namely

$$\begin{aligned}&\left(\langle 0|_m \otimes \langle 0|_q\right)\left(I_m \otimes (\otimes_q \mathrm{HAD}) \otimes I_n\right)\mathrm{MAT}_j\left(I_m \otimes (\otimes_q \mathrm{HAD}) \otimes I_n\right)\left(|0\rangle_m \otimes |0\rangle_q\right)\\ =&\frac{1}{M\alpha}\sum_{k=0}^{M-1} A\left(a + k(b-a)/M\right).\end{aligned} \tag{35}$$

The quantum circuit of implementing Eq. (35) is described in Fig. 4.

The second step is to implement the time-independent matrix exponential via a contour integral formulation combined with QSVT, as detailed in Appendix G. The observation is that for general A that are not normal, the singular value transformation no longer agrees with the eigenvalue transformation. Nevertheless, the matrix exponential can be applied by exploring the contour integral formulation proposed in [61]. The idea is to express e^A as a contour integral:

$$e^A = \frac{1}{2\pi i}\oint_\Gamma e^z (z - A)^{-1}\,\mathrm{d}z, \tag{36}$$

where Γ is a contour in the complex plane that contains all the eigenvalues of A in its interior. We then discretize the integral using numerical quadrature. This procedure turns the matrix exponential into a linear combination of $(z_j - A)^{-1}$, which can be efficiently calculated as matrix inversion problems that does not require the matrix being normal. Here z_j's are some constants determined by the numerical quadrature, which is detailed in Section 5.3. We also remark that there are certain cases of A where e^A admits a simpler implementation using QSVT via polynomial approximations of the exponential function [37, Corollary 64]. Ref. [59] also developed a method to apply a more general class of matrix functions using the contour integral, but they did not consider how to construct a block encoding, instead focusing on preparing the output state.

5.2 Time discretization error

We now analyze the time-discretization error for the short-time evolution.

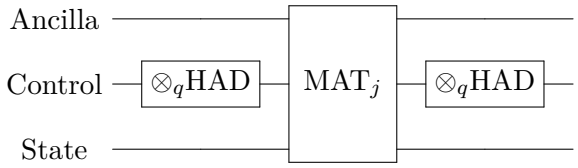


Figure 4: Quantum circuit of implementing a block encoding of the Hamiltonian formulated in Eq. (35). Here HAD represents the single qubit Hadamard gate.

Lemma 11 (Time discretization errors of the first-order Magnus integrator)**.** *Let* $\Xi = \mathcal{T}e^{\int_a^b A(t)\mathrm{d}t}$ *denote the exact evolution operator on the time interval* $[a,b]$*, and* $\bar{\Xi}$ *denote the first-order Magnus operator defined in Eq.* (34)*. Then we have*

$$\begin{aligned}\left\|\Xi - \bar{\Xi}\right\| \leq \frac{1}{2}\int_a^b \mathrm{d}\tau \left\|\left[\int_a^\tau A(s)\mathrm{d}s, A(\tau)\right]\right\| e^{\int_a^b \|A(s)\|\mathrm{d}s} \\ + \frac{b-a}{M} V_a^b(A) e^{\int_a^b \|A(s)\mathrm{d}s\| + \frac{b-a}{M}V_a^b(A)}.\end{aligned} \tag{37}$$

Proof. The errors come from two parts: one from dropping the time-ordering operator, while the other from the numerical quadrature. To estimate the error, we introduce the notations of the exact evolution from time a to t as

$$\mathcal{V}(t,a) := \mathcal{T}e^{\int_a^t A(s)\mathrm{d}s},$$

so that $\Xi = \mathcal{V}(b,a)$, and the matrix after dropping the time-ordering operator as

$$\widetilde{V}(t,a) = e^{\int_a^t A(s)\mathrm{d}s}. \tag{38}$$

We first study the approximation error between the two. For any $t \in [a,b]$, by differentiating $\widetilde{V}(t,a)$ with respect to t, we have

$$\partial_t \widetilde{V}(t,a) = \int_0^1 e^{\beta\int_{t_j}^t A(s)\mathrm{d}s} A(t) e^{(1-\beta)\int_{t_j}^t A(s)\mathrm{d}s}\mathrm{d}\beta. \tag{39}$$

The difference between $\widetilde{V}$ and $\mathcal{V}$ satisfies the differential equation

$$\partial_t\left(\widetilde{V} - \mathcal{V}\right) = A(t)\left(\widetilde{V} - \mathcal{V}\right) + \int_0^1 e^{\beta\int_a^t L(s)\mathrm{d}s} A(t) e^{(1-\beta)\int_a^t A(s)\mathrm{d}s}\mathrm{d}\beta - A(t)\widetilde{V}, \tag{40}$$

and by the variation of parameter formula [18], we have

$$\widetilde{V}(t,a) - \mathcal{V}(t,a) = \int_a^t \mathcal{V}(t,\tau) g(\tau)\mathrm{d}\tau, \tag{41}$$

where

$$\begin{aligned} g(\tau) &:= \int_0^1 e^{\beta\int_a^\tau A(s)\mathrm{d}s} A(\tau) e^{(1-\beta)\int_a^\tau A(s)\mathrm{d}s}\mathrm{d}\beta - A(\tau)e^{\int_a^\tau A(s)\mathrm{d}s} \\ &= \int_0^1 g_0(\beta;\tau) - g_0(0;\tau)\mathrm{d}\beta, \end{aligned}$$

and

$$g_0(\beta;\tau) := e^{\beta\int_a^\tau A(s)\mathrm{d}s}A(\tau)e^{(1-\beta)\int_a^\tau A(s)\mathrm{d}s}.$$

Applying the fundamental theorem of calculus in terms of β yields that

$$\begin{aligned}
\|g(\tau)\| =& \left\| \int_0^1 \int_0^\beta \partial_\gamma g_0(\gamma;\tau)\mathrm{d}\gamma\mathrm{d}\beta \right\| \\
=& \left\| \int_0^1 \int_0^\beta e^{\gamma\int_a^\tau L(s)\mathrm{d}s} \left[\int_a^\tau A(s)\mathrm{d}s, A(\tau)\right] e^{(1-\gamma)\int_a^\tau A(s)\mathrm{d}s}\mathrm{d}\gamma\mathrm{d}\beta \right\| \\
\leq& \int_0^1 \int_0^\beta e^{\gamma\|\int_a^\tau A(s)\mathrm{d}s\|} \left\| \left[\int_a^\tau A(s)\mathrm{d}s, A(\tau)\right] \right\| e^{(1-\gamma)\|\int_a^\tau A(s)\mathrm{d}s\|}\mathrm{d}\gamma\mathrm{d}\beta \\
\leq& \frac{1}{2} \left\| \left[\int_a^\tau A(s)\mathrm{d}s, A(\tau)\right] \right\| e^{\int_a^\tau \|A(s)\|\mathrm{d}s}.
\end{aligned}$$

Note that the exponential factor here is bounded quite loose, and in fact one can get a sharper bound by using the logarithmic norm. But here for our purposes, it is sufficient to estimate at the level of $\|A(s)\|$. Another useful fact is

$$\left\| \mathcal{T} e^{\int_{\beta_1}^{\beta_2} A(s)\mathrm{d}s} \right\| \leq e^{\int_{\beta_1}^{\beta_2} \|A(s)\|\mathrm{d}s}, \tag{42}$$

which can be shown by applying the Gronwall's inequality in terms of β_2 to

$$\|\Phi(\beta_2,\beta_1)\| \leq 1 + \int_{\beta_1}^{\beta_2} \|A(s)\| \, \|\Phi(s,\beta_1)\| \, \mathrm{d}s,$$

where $\Phi(\beta_2,\beta_1) := \mathcal{T} e^{\int_{\beta_1}^{\beta_2} A(s)\mathrm{d}s}$ is the fundamental matrix. Thus, the approximation error of the first part can be summarized as

$$\left\| \widetilde{V}(b,a) - \Xi \right\| = \left\| \widetilde{V}(b,a) - \mathcal{V}(b,a) \right\| \leq \int_a^b \mathrm{d}\tau \left\| \left[\int_a^\tau A(s)\mathrm{d}s, A(\tau)\right] \right\| e^{\int_a^b \|A(s)\|\mathrm{d}s}. \tag{43}$$

We now estimate the error induced by the numerical quadrature using the total variation. Following the result in Appendix F, we have

$$\|E(b,a)\| := \left\| \int_a^b A(s)\mathrm{d}s - \frac{b-a}{M} \sum_{k=0}^{M-1} L(a+k(b-a)/M) \right\| \leq \frac{b-a}{M} V_a^b(A), \tag{44}$$

where we recall that $V_a^b(A)$ is the total variation of $A(t)$ on the interval $[a,b]$. Furthermore, the variation of parameter formula gives that for square matrices B and E,

$$e^{B+E} - e^B = \int_0^1 \mathrm{d}\beta e^{B(1-\beta)} E e^{(B+E)\beta},$$

which implies that

$$\left\| e^{B+E} - e^B \right\| \leq \|E\| \, e^{\|B\|+\|E\|}. \tag{45}$$

Therefore,

$$\left\| \widetilde{V}(b,a) - \bar{\Xi} \right\| \leq \frac{b-a}{M} V_a^b(A) e^{\int_a^b \|A(s)\|\mathrm{d}s + \frac{b-a}{M} V_a^b(A)}. \tag{46}$$

Combining (43) and (46) yields the desired result. □

5.3 Algorithm for implementing e^A

In order to discuss the complexity of implementing the short-time first-order Magnus evolution operator, we first briefly discuss in this section how to implement a matrix exponential of a general time-independent matrix, namely to implement $f(A) = e^A$, when A is given through its $(\alpha, m, 0)$-block encoding. The details are laid out in Appendix G.

The main idea is to use the contour integral representation of a matrix function

$$f(A) = \frac{1}{2\pi i} \int_\Gamma e^z (z - A)^{-1} \mathrm{d}z, \tag{47}$$

where Γ is a unit circle with radius β: $\Gamma = \{z = \beta e^{i\theta} : \theta \in \mathbb{R}\}$. This contour integral can be discretized as

$$f_K(A) = \frac{1}{K} \sum_{k=0}^{K-1} e^{z_k} z_k (z_k - A)^{-1}, \tag{48}$$

for $z_k = \beta e^{i2\pi k/K}$. The error that comes from this discretization can be analyzed using Lemma 18, which is modified from [63, Theorem 18.1] and [59, Proposition 5]. We choose $\beta = 2\alpha$, $R = 4\alpha$, and by Lemma 18 we have

$$\|f(A) - f_K(A)\| = \mathcal{O}(e^{4\alpha} 2^{-K}). \tag{49}$$

Our goal is to construct a block encoding of e^A using (48). We will proceed as follows: we first construct a block encoding of

$$\Xi = \sum_{k=0}^{K-1} |k\rangle \langle k| \otimes (z_k - A), \tag{50}$$

where $z_k = \beta e^{i2\pi k/K}$. Inverting Ξ using QSVT gives us

$$\Xi^{-1} = \sum_{k=0}^{K-1} |k\rangle \langle k| \otimes (z_k - A)^{-1}. \tag{51}$$

Now suppose we can prepare quantum states $|\mathrm{COEF}_{\mathrm{int}}\rangle$, $|\mathrm{COEF}'_{\mathrm{int}}\rangle$ that satisfy

$$|\mathrm{COEF}_{\mathrm{int}}\rangle \propto \sum_k \sqrt{e^{z_k} z_k}\, |k\rangle\,, \quad |\mathrm{COEF}'_{\mathrm{int}}\rangle \propto \sum_k (\sqrt{e^{z_k} z_k})^* \, |k\rangle\,,$$

then we obtain the desired block encoding

$$(\langle \mathrm{COEF}'_{\mathrm{int}}| \otimes I_n) \Xi^{-1} (|\mathrm{COEF}_{\mathrm{int}}\rangle \otimes I_n) \propto f_K(A) \approx f(A). \tag{52}$$

Here for a complex number $z \in \mathbb{C}$, $\sqrt{z}$ can be any $w \in \mathbb{C}$ such that $w^2 = z$.

To construct this block encoding we need additional ancilla qubits. The entire circuit acts on five different registers, as shown in Figure 6. The first register contains one qubit and is used for matrix inversion through QSVT. The second register contains $\log_2(K)$ qubits to encode the k coefficients $e^{z_k} z_k$ in the amplitude. The third register contains one qubit and is used in the block encoding of each $z_k - A$. The fourth register contains m qubits, and this is the ancilla register used in the block encoding U_A. The fifth register contains n qubits and is the qubits that A acts on. We will use I_r to denote the identity operator acting on r qubits.

The cost of this implementation can be summarized in the following lemma, whose proof is laid out in Appendix G.

Lemma 12 (Matrix exponentiation)**.** *Given a $(\alpha, m, 0)$-block encoding U_A of A, we can construct a $(4\mathcal{A}/(3\alpha), m+\mathcal{O}(\log(\alpha)+\log\log(\epsilon^{-1})), \epsilon)$-block encoding of e^A, with $\mathcal{O}(\alpha+\log(\epsilon^{-1}))$ applications of (controlled-) U_A and its inverse, and $\mathcal{O}(\alpha^2+\log^2(\epsilon^{-1}))$ additional elementary gates. Here*

$$\mathcal{A} = \frac{2\alpha}{K}\sum_{k=0}^{K}|e^{z_k}|,$$

where $z_k = 2\alpha e^{i2\pi k/K}$.

5.4 Short-time complexity of the first-order integrator

The cost of constructing a block encoding of the short-time first-order Magnus evolution operator is given by the following lemma.

Lemma 13 (Short-time evolution through the first-order Magnus integrator)**.** *Suppose $A \in BV([a,b])$, and is accessed through an $(n_q, m, a, b, \alpha, \epsilon)$-*MAT *as defined in Definition 1 for some $\epsilon \leq \epsilon'/(3(b-a))$. Let*

$$\frac{1}{2}\int_a^b \mathrm{d}\tau \left\| \left[\int_a^\tau A(s)\mathrm{d}s, A(\tau) \right] \right\| \leq \beta_0,$$

and $b-a \leq \min\left\{(2\alpha)^{-1}, (2\epsilon)^{-1}\right\}$. Choose

$$M = 2^{n_q} = \Theta\left(\frac{(b-a)V_a^b(A)}{\beta_0}\right),$$

where $V_a^b(A)$ is the total variation of $A(t)$ on the interval $[a,b]$. Then we can construct an (α', m', δ')-block encoding of $\Xi = \mathcal{T}e^{\int_a^b A(t)\mathrm{d}t}$, with

$$\alpha' = \mathcal{O}\left(\frac{4\mathcal{A}}{3\alpha(b-a)}\right) = \mathcal{O}(1), \quad \mathcal{A} = \frac{2\alpha(b-a)}{K}\sum_{k=0}^{K} e^{2\alpha(b-a)\cos(2\pi k/K)} = \Theta(\alpha(b-a))$$

$$m' = m + n_q + \mathcal{O}\left(\log(\alpha(b-a)) + \log\log(1/\epsilon')\right), \quad \delta' = \epsilon' + 4\beta_0.$$

This requires

- $\mathcal{O}\left(\alpha(b-a)+\log(1/\epsilon')\right)$ *queries to* MAT*,*
- $\mathcal{O}(\alpha(b-a)n_q + n_q\log(1/\epsilon'))$ *additional elementary gates.*

Proof. MAT is a time-dependent matrix encoding of $A(t)$ with precision ϵ, and is hence a time-dependent matrix encoding of $\tilde{A}(t)$ with precision 0 by Definition 1.

We first consider the cost of constructing the block encoding of

$$\tilde{S} := \frac{b-a}{M}\sum_{k=0}^{M-1}\tilde{A}\left(a+k(b-a)/M\right).$$

Let $n_q = \log_2 M$. As depicted in (35), one can construct a $(\alpha(b-a), m+n_q, 0)$ block encoding of $\tilde{S}$, using 1 query to MAT and n_q additional one-qubit gates.

We then use QSVT and contour integral formulation Lemma 12 to implement $e^{\tilde{S}}$. Then we obtain the following

$$\left(\frac{4\mathcal{A}}{3\alpha(b-a)}, m+n_q+\mathcal{O}\left(\log(\alpha(b-a))+\log\log(1/\epsilon')\right), \frac{3-e}{3}\epsilon'\right)$$

block encoding of $e^{\tilde{S}}$, where

$$\mathcal{A}=\frac{2\alpha(b-a)}{K}\sum_{k=0}^{K} e^{2\alpha(b-a)\cos(2\pi k/K)}=\Theta(\alpha(b-a)).$$

This uses $\mathcal{O}\left(\alpha(b-a)\right)+\log(1/\epsilon'))$ queries to the block encoding of $\tilde{A}$ and hence $\mathcal{O}\left(\alpha(b-a)\right)+\log(1/\epsilon'))$ queries to MAT. The additional elementary gates needed for this procedure is

$$\mathcal{O}\left(\alpha^2(b-a)^2+n_q\log(1/\epsilon')+n_q\alpha(b-a)\right)=\mathcal{O}\left(n_q\log(1/\epsilon')+n_q\alpha(b-a)\right).$$

We now control the error between $e^{\tilde{S}}$ and $\Xi=\mathcal{T}e^{\int_a^b A(t)\mathrm{d}t}$. Denote

$$S:=\frac{b-a}{M}\sum_{k=0}^{M-1} A\left(a+k(b-a)/M\right).$$

We first consider the error between $e^{\tilde{S}}$ and e^{S}. Notice that

$$\left\|\tilde{S}-S\right\|\leq(b-a)\epsilon\leq 1/2,\quad \left\|\tilde{S}\right\|\leq(b-a)\alpha\leq 1/2.$$

By (45), we have

$$\left\|e^{\tilde{S}}-e^{S}\right\|\leq\left\|\tilde{S}-S\right\|e^{\|\tilde{S}\|+\|\tilde{S}-S\|}\leq e(b-a)\epsilon<\frac{e}{3}\epsilon'$$

Choose

$$M=\Theta\left(\frac{(b-a)V_a^b(A)}{\beta_0}\right)$$

so that

$$\frac{b-a}{M}V_a^b(A)=\Theta\left(\beta_0\right).$$

Thanks to Lemma 11, the error

$$\left\|e^{\tilde{A}}-\Xi\right\|\leq\beta_0 e^{\int_a^b\|A(s)\|\mathrm{d}s}\left(1+e^{\frac{1}{2}\int_a^b\mathrm{d}\tau\left\|\left[\int_a^\tau A(s)\mathrm{d}s,A(\tau)\right]\right\|}\right).$$

By our choice of a and b we have that the right-hand-side is bounded by $4\beta_0$, as

$$\frac{1}{2}\int_a^b\mathrm{d}\tau\left\|\left[\int_a^\tau A(s)\mathrm{d}s,A(\tau)\right]\right\|\leq\frac{1}{2}(b-a)^2\alpha^2\leq\frac{1}{8}.$$

Therefore, we get a

$$\left(\frac{4\mathcal{A}}{3\alpha(b-a)}, m+n_q+\mathcal{O}(\log(\alpha(b-a))+\log\log(1/\epsilon'), \epsilon'+4\beta_0\right)$$

block encoding of Ξ.

□

5.5 Block encoding of the long-time evolution operator

For simplicity, we choose a uniform temporal mesh. Let $t_l = lT/L$ so that $0 = t_0 < t_1 < \cdots < t_l < \cdots < t_L = T$ where L is the number of time steps. We perform the short-time first-order Magnus evolution on each interval $[t_l, t_{l+1}]$ and multiply all the evolution together.

By Lemma 13, each short-time first-order Magnus evolution on $[t_l, t_{l+1}]$ yields a $(\alpha_l, m_l, \epsilon_l \|\Xi_l\|)$-block encoding (denoted U_l) of $\Xi_l = \mathcal{T} e^{\int_{t_l}^{t_{l+1}} A(s)\mathrm{d}s}$, where

$$\alpha_l = \mathcal{O}(1), \quad m_l = m + n_{q,l} + \mathcal{O}\left(\log(\alpha(t_{l+1} - t_l)) + \log\log(1/\delta)\right), \quad \epsilon_l \|\Xi_l\| = \delta + 4\beta_l,$$

and β_l is an upper bound of

$$\frac{1}{2}\int_{t_l}^{t_{l+1}} \mathrm{d}\tau \left\| \left[\int_{t_l}^{\tau} A(s)\mathrm{d}s, A(\tau) \right] \right\|.$$

The construction uses $\mathcal{O}\left(\alpha(t_{l+1} - t_l) + \log(1/\delta)\right)$ queries to MAT_l and $\mathcal{O}(\alpha_l(t_{l+1} - t_l)n_{q,l} + n_{q,l}\log(1/\delta))$ additional elementary gates with $n_{q,l} = \mathcal{O}\left(\log(\frac{(t_{l+1}-t_l)V_{t_l}^{t_{l+1}}(A)}{\beta_l})\right)$.

Applying Theorem 4, we get a $(\alpha_{\mathrm{comp}}, m_{\mathrm{comp}}, \epsilon_{\mathrm{comp}})$-block encoding of $\mathcal{T} e^{\int_0^T A(t)\mathrm{d}t}$ with

$$\frac{P}{2(1-\delta)^L} \leq \alpha_{\mathrm{comp}} \leq \frac{e^{1/2}P}{(1-\delta)^L}.$$

Here

$$m_{\mathrm{comp}} = \lceil \log_2(L) \rceil + m + \max_l n_{q,l} + \max_l \mathcal{O}\left(\log(\alpha_l(t_{l+1} - t_l)) + \log\log(1/\epsilon_l)\right),$$

and

$$\epsilon_{\mathrm{comp}} = e^{1/2}\left(L\epsilon' + L\delta + 4\beta_{\mathrm{comp}}\right)P, \tag{53}$$

where $\beta_{\mathrm{comp}} = \sum_l \beta_l$ is an upper bound of

$$\frac{1}{2}\sum_{l=0}^{L-1}\int_{t_l}^{t_{l+1}} \mathrm{d}\tau \left\| \left[\int_{t_l}^{\tau} A(s)\mathrm{d}s, A(\tau) \right] \right\| \leq \frac{1}{4}\sup_{s,\tau\in[0,T]} \|[A(s), A(\tau)]\| \frac{T^2}{L}.$$

In this block encoding we use $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\epsilon'}\right)\right)$ applications of each U_l.

Denote the upper bound of the time average L^1-scaling $\int_0^T \|A(s)\|\,\mathrm{d}s/T$ as $\overline{\alpha}$. Thanks to our choice of t_l's, each β_l can be upper bounded by $\overline{\alpha}^2 T^2/L^2$. Choose

$$\epsilon' = \mathcal{O}\left(\frac{\epsilon_{\mathrm{comp}}}{LP}\right), \quad \delta = \mathcal{O}\left(\frac{\epsilon_{\mathrm{comp}}}{LP}\right), \quad L = \mathcal{O}\left(\max\left\{\frac{\alpha_{\mathrm{comm}}T^2P}{\epsilon_{\mathrm{comp}}}, \alpha T\right\}\right), \quad \delta = \frac{1}{2L},$$

where

$$\alpha_{\mathrm{comm}} := \sup_{s,\tau\in[0,T]} \|[A(s), A(\tau)]\|. \tag{54}$$

With this choice of parameters we have

$$\alpha_{\mathrm{comp}} = \Theta(P) = \Theta\left(\prod_{l=1}^{L} \|\Xi_l\|\right), \tag{55}$$

and

$$m_{\text{comp}} = m + \mathcal{O}\left(\log_2(V_0^T(A)\alpha TP\epsilon_{\text{comp}}^{-1})\right). \tag{56}$$

Each U_l is used

$$\mathcal{O}\left(L\log\left(\frac{LP}{\epsilon_{\text{comp}}}\right)\right) = \mathcal{O}\left(\max\left\{\frac{\alpha_{\text{comm}}T^2P}{\epsilon_{\text{comp}}}, \alpha T\right\}\log\left(\max\left\{\alpha_{\text{comm}}, \alpha\right\}\frac{TP}{\epsilon_{\text{comp}}}\right)\right)$$

times. Given the fact that each U_l uses queries to MAT_l $\mathcal{O}\left(\alpha(t_{l+1}-t_l) + \log\left(\max\left\{\alpha_{\text{comm}}, \alpha\right\}TP/\epsilon_{\text{comp}}\right)\right)$ times, and that there are L of them, the total number of queries to all MAT are

$$\mathcal{O}\left(\sum_{l=0}^{L-1} L\log\left(\frac{LP}{\epsilon_{\text{comp}}}\right)\left(\alpha(t_{l+1}-t_l) + \log\left(\max\left\{\alpha_{\text{comm}}, \alpha\right\}\frac{TP}{\epsilon_{\text{comp}}}\right)\right)\right)$$

$$= \mathcal{O}\Bigg(\left(\alpha T\max\left\{\frac{\alpha_{\text{comm}}T^2P}{\epsilon_{\text{comp}}}, \alpha T\right\} + \max\left\{\frac{\alpha_{\text{comm}}T^2P}{\epsilon_{\text{comp}}}, \alpha T\right\}^2\log\left(\max\left\{\alpha_{\text{comm}}, \alpha\right\}\frac{TP}{\epsilon_{\text{comp}}}\right)\right)\times \tag{57}$$

$$\times\log\left(\max\left\{\alpha_{\text{comm}}, \alpha\right\}\frac{TP}{\epsilon_{\text{comp}}}\right)\Bigg). \tag{58}$$

Note that when $\alpha_{\text{comm}} = 0$, this first-order truncated series implementation in fact has the same complexity scaling as the high-order truncated Dyson series. We remark that it is also possible to get the L^1 scaling $\overline{\alpha} = \frac{1}{T}\int_0^T \|A(s)\|\,\mathrm{d}s$, by varying time step sizes in the propagation according to the average performance of the Hamiltonian as in [4, 7]: Let $0 = t_0 < t_1 < \cdots < t_l < \cdots < t_L = T$ where L is the number of time steps and $t_1, \cdots, t_{L-1}$ are chosen such that

$$\int_0^{t_1}\|A(s)\|\,\mathrm{d}s = \cdots = \int_{t_l}^{t_{l+1}}\|A(s)\|\,\mathrm{d}s = \frac{1}{L}\int_0^T\|A(s)\|\,\mathrm{d}s, \quad 0 \le l \le L-1, \tag{59}$$

which we shall not discuss more details here.

5.6 Success probability and main result paired with first-order integrator

The success probability of the first-order Magnus method follows a similar argument as the Dyson series approach as detailed in Section 3.3. It suffices to choose $\epsilon_{\text{comp}} = \mathcal{O}(\epsilon\|\,|\psi(T)\rangle\,\|)$ and the success probability is at least $\Omega(Q^{-2})$. Suppose that the initial state $|\psi(0)\rangle$ can be prepared using a unitary circuit U_{init}, then we can boost the success probability to 2/3 with $\mathcal{O}(Q)$ rounds of amplitude amplification. We can conclude the result when pairing the time-marching strategy with the first-order Magnus-type integrator for long time evolution as follows.

Theorem 14 (The time-marching differential equation solver paired with the first-order Magnus integrator)**.** *Let $|\psi(t)\rangle$ be the solution to the ODE $\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = A(t)|\psi(t)\rangle$. Suppose we have a unitary circuit U_{init} that satisfies $U_{\text{init}}|0^n\rangle = |\psi(0)\rangle$. For a coefficient matrix $A \in BV([0,T])$, suppose $0 = t_0 < t_1 < \ldots < t_L = T$, and for each segment $[t_{l-1}, t_l]$ we have a time-dependent matrix encoding of $A(t)$ denoted as MAT_l that is an $(n_q, m, t_{l-1}, t_l, \alpha, \epsilon'') - \text{MAT}$ as defined in Definition 1 for some $\epsilon'' < \epsilon/(2TQ)$. Let $V_0^T(A)$ be the total variation of A on*

$[0,T]$ *as defined in* (10)*. We can prepare, with probability at least* $2/3$*, a quantum state* $|\widetilde{\psi}(T)\rangle$ *that satisfies*

$$\left\| \frac{|\psi(T)\rangle}{\| |\psi(T)\rangle \|} - \frac{|\widetilde{\psi}(T)\rangle}{\| |\widetilde{\psi}(T)\rangle \|} \right\| = \mathcal{O}(\epsilon),$$

using

$$\mathcal{O}\Bigg(\left(\alpha T \max\left\{ \frac{\alpha_{\mathrm{comm}} T^2 Q}{\epsilon}, \alpha T \right\} + \max\left\{ \frac{\alpha_{\mathrm{comm}} T^2 Q}{\epsilon}, \alpha T \right\}^2 \log\left(\max\left\{ \alpha_{\mathrm{comm}}, \alpha \right\} \frac{TQ}{\epsilon} \right) \right) \times \\ \times \log\left(\max\left\{ \alpha_{\mathrm{comm}}, \alpha \right\} \frac{TQ}{\epsilon} \right) \Bigg).$$

total number of queries to all MAT_l*, and* $\mathcal{O}(Q)$ *applications of (controlled-)* U_{init} *and its inverse. In particular, in the high-precision limit given a non-zero* α_{comm}*, the total number of queries to all* MAT_l *can be simplified as*

$$\mathcal{O}\left(\frac{\alpha_{\mathrm{comm}}^2 T^4 Q^3}{\epsilon^2} \log^2\left(\frac{\max\{\alpha_{\mathrm{comm}}, \alpha\} TQ}{\epsilon} \right) \right).$$

In total we use $\mathcal{O}(n + m + \mathrm{polylog}(V_0^T(A)\alpha TQ\epsilon^{-1}))$ *qubits. Here* Q *is defined as* (4)*, and* α_{comm} *is defined in Eq.* (54)*. Success is flagged by the measurement result of a qubit.*

6 Discussion

We revisit the time-marching strategy for solving the ODE (1), and demonstrate that it can be a useful alternative to QLSA based quantum differential equation solvers. On a technical level, our time-marching based algorithm achieves the following tasks for the first time: (1) It has provable performance guarantees for non-diagonalizable, and time-dependent dynamics. (2) It retains high-order accuracy for non-smooth $A(t)$. (3) It can use few queries to the initial state. For inhomogeneous linear differential equations $\frac{\mathrm{d}}{\mathrm{d}t} |\psi(t)\rangle = A(t) |\psi(t)\rangle + |b(t)\rangle$, we may use the variation of constants to express the solution as

$$|\psi(t)\rangle = \mathcal{T} e^{\int_0^t A(s)\mathrm{d}s} |\psi_0\rangle + \int_0^t \mathcal{T} e^{\int_{t'}^t A(s)\mathrm{d}s} |b(t')\rangle \, \mathrm{d}t'. \tag{60}$$

After discretization of the integration with respect to t', this is reduced to a series of homogeneous linear differential equations that can be implemented by the time-marching method. The analysis for the inhomogeneous case can thus be similar using the tools developed in this paper.

At a high level, time-marching based methods and QLSA based methods simply amount to two different ways of rearranging the same identities of the form $|\psi_l\rangle = \bar{\Xi}_l |\psi_{l-1}\rangle, l = 1, \ldots, L$. However, there are many differences between these two classes of algorithms, and we do not know whether the gaps can be closed with further improvements, or are intrinsic to each type of the method. For instance, it is possible to refine the complexity analysis of QLSA based algorithms so that the cost does not directly depend on the condition number κ_V, or to improve the algorithm so that it achieves high-order accuracy for non-smooth $A(t)$ [16]. However, it

is unclear whether the number of queries to the initial state of any QLSA based algorithm can be independent of T. For time-marching based algorithms, though the dependence on the precision is near-optimal and scales as polylog$(1/\epsilon)$, the dependence of the query complexity to the input matrix is $\mathcal{O}(T^2)$. We do not know whether the $\mathcal{O}(T^2)$ scaling in the query complexity to the matrix can be improved, or whether the dependence on Q can be weakened to $\bar{Q} = \frac{\|\mathcal{T}e^{\int_0^T A(t)\mathrm{d}t}\|}{\||\psi(T)\rangle\|}$. It is also possible that another class of algorithms is needed to provide a unifying perspective to the questions above, and to achieve near-optimal complexity with respect to all parameters.

Acknowledgments

This work was partially supported by the NSF Quantum Leap Challenge Institute (QLCI) program under Grant No. OMA-2016245 and NSF DMS-2208416 (D.F.), by the U.S. Department of Energy under the Quantum Systems Accelerator program under Grant No. DE-AC02-05CH11231 (Y.T.), and by a Google Quantum Research Award (L.L.). L.L. is a Simons Investigator. We thank Dong An, Dominic Berry, Andrew Childs and Jin-Peng Liu for discussions.

References

[1] A. Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *Leibniz Int. Proc. Informatics, LIPIcs*, volume 14, pages 636–647, 2012. doi:10.4230/LIPIcs.STACS.2012.636.

[2] D. An, D. Fang, S. Jordan, J.-P. Liu, G. H. Low, and J. Wang. Efficient quantum algorithm for nonlinear reaction-diffusion equations and energy estimation, 2022. URL: http://arxiv.org/abs/2205.01141, arXiv:2205.01141, doi:10.48550/ARXIV.2205.01141.

[3] D. An, D. Fang, and L. Lin. Time-dependent unbounded Hamiltonian simulation with vector norm scaling. *Quantum*, 5:1–49, may 2021. URL: https://doi.org/10.22331/q-2021-05-26-459, arXiv:2012.13105, doi:10.22331/Q-2021-05-26-459.

[4] D. An, D. Fang, and L. Lin. Time-dependent Hamiltonian Simulation of Highly Oscillatory Dynamics and Superconvergence for Schrödinger Equation. *Quantum*, 6:690, apr 2022. doi:10.22331/q-2022-04-15-690.

[5] D. An and L. Lin. Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm. *ACM Trans. Quantum Comput.*, 3(2):1–28, 2022. arXiv:1909.05500, doi:10.1145/3498331.

[6] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997. doi:10.1137/S0097539796300933.

[7] D. Berry, A. M. Childs, Y. Su, X. Wang, and N. Wiebe. Time-dependent Hamiltonian simulation with L^1-norm scaling. *Quantum*, 4:254, 2020. doi:10.22331/q-2020-04-20-254.

[8] D. W. Berry. High-order quantum algorithm for solving linear differential equations.

Journal of Physics A: Mathematical and Theoretical, 47(10):105301, feb 2014. doi:10.1088/1751-8113/47/10/105301.

[9] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders. Efficient quantum algorithms for simulating sparse hamiltonians. *Commun. Math. Phys.*, 270(2):359–371, 2007. arXiv:0508139, doi:10.1007/s00220-006-0150-x.

[10] D. W. Berry and A. M. Childs. Black-box hamiltonian simulation and unitary implementation. *Quantum Inf. Comput.*, 12(1-2):29–62, 2012. arXiv:0910.4157, doi:10.26421/QIC12.1-2.

[11] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma. Exponential improvement in precision for simulating sparse Hamiltonians. In *Proc. Annu. ACM Symp. Theory Comput.*, pages 283–292, 2014. arXiv:1312.1414, doi:10.1145/2591796.2591854.

[12] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma. Simulating hamiltonian dynamics with a truncated taylor series. *Phys. Rev. Lett.*, 114(9):90502, 2015. arXiv:1412.4687, doi:10.1103/PhysRevLett.114.090502.

[13] D. W. Berry, A. M. Childs, and R. Kothari. Hamiltonian Simulation with Nearly Optimal Dependence on all Parameters. *Proc. - Annu. IEEE Symp. Found. Comput. Sci. FOCS*, 2015-December:792–809, 2015. arXiv:1501.01715, doi:10.1109/FOCS.2015.54.

[14] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang. Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Communications in Mathematical Physics*, 356(3):1057–1081, 2017. doi:10.1007/s00220-017-3002-y.

[15] D. W. Berry, R. Cleve, and S. Gharibian. Gate-efficient discrete simulations of continuous-time quantum query algorithms. *Quantum Inf. Comput.*, 14(1-2):1–30, 2014. doi:10.26421/qic14.1-2-1.

[16] D. W. Berry and P. C. S. Costa. Quantum algorithm for time-dependent differential equations using dyson series, 2022. URL: https://arxiv.org/abs/2212.03544, doi:10.48550/ARXIV.2212.03544.

[17] G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. *Contemp. Math.*, 305:53–74, 2002. arXiv:0005055, doi:10.1090/conm/305/05215.

[18] F. Brauer and J. A. Nohel. *The Qualitative Theory of Ordinary Differential Equations: An Introduction*. Dover Books on Mathematics. Dover Publications, 2012. URL: https://books.google.com/books?id=9qPsbRl7hBkC.

[19] R. L. Burden, J. D. Faires, and A. C. Reynolds. *Numerical analysis*. Brooks Cole, 2000.

[20] E. Campbell. Random Compiler for Fast Hamiltonian Simulation. *Phys. Rev. Lett.*, 123(7):70503, 2019. arXiv:1811.08017, doi:10.1103/PhysRevLett.123.070503.

[21] S. Chakraborty, A. Gilyén, and S. Jeffery. The power of block-encoded matrix powers: Improved regression techniques via faster Hamiltonian simulation. *Leibniz Int. Proc. Informatics, LIPIcs*, 132, 2019. arXiv:1804.01973, doi:10.4230/LIPIcs.ICALP.2019.33.

[22] R. Chao, D. Ding, A. Gilyen, C. Huang, and M. Szegedy. Finding Angles for Quantum Signal Processing with Machine Precision. 2020. URL: http://arxiv.org/abs/2003.02831, arXiv:2003.02831, doi:10.48550/arXiv.2003.02831.

[23] C.-F. Chen, Hsin-Yuan, Huang, R. Kueng, and J. a. Tropp. Concentration for random product formulas. *arXiv:2008.11751*, page 25, 2020. URL: http://arxiv.org/abs/2008.11751, arXiv:2008.11751, doi:10.1103/PRXQuantum.2.040305.

[24] Y. H. Chen, A. Kalev, and I. Hen. Quantum Algorithm for Time-Dependent Hamiltonian Simulation by Permutation Expansion. *PRX Quantum*, 2(3):30342, sep 2021. URL: https://link.aps.org/doi/10.1103/PRXQuantum.2.030342, arXiv:2103.15334, doi:10.1103/PRXQuantum.2.030342.

[25] A. M. Childs, R. Kothari, and R. D. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM J. Comput.*, 46(6):1920–1950, 2017. arXiv:1511.02306, doi:10.1137/16M1087072.

[26] A. M. Childs and J. P. Liu. Quantum Spectral Methods for Differential Equations. *Commun. Math. Phys.*, 375(2):1427–1457, 2020. arXiv:1901.00961, doi:10.1007/s00220-020-03699-z.

[27] A. M. Childs, D. Maslov, Y. Nam, N. J. Ross, and Y. Su. Toward the first quantum simulation with quantum speedup. *Proc. Natl. Acad. Sci. U. S. A.*, 115(38):9456–9461, 2018. arXiv:1711.10980, doi:10.1073/pnas.1801723115.

[28] A. M. Childs, A. Ostrander, and Y. Su. Faster quantum simulation by randomization. *Quantum*, 3:182, 2019. arXiv:1805.08385, doi:10.22331/q-2019-09-02-182.

[29] A. M. Childs and Y. Su. Nearly Optimal Lattice Simulation by Product Formulas. *Phys. Rev. Lett.*, 123(5):50503, 2019. arXiv:1901.00564, doi:10.1103/PhysRevLett.123.050503.

[30] A. M. Childs, Y. Su, M. C. Tran, N. Wiebe, and S. Zhu. Theory of Trotter Error with Commutator Scaling. *Phys. Rev. X*, 11(1):11020, 2021. doi:10.1103/PhysRevX.11.011020.

[31] A. M. Childs and N. Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *Quantum Inf. Comput.*, 12(11-12):901–924, nov 2012. URL: http://dx.doi.org/10.26421/QIC12.11-12, arXiv:1202.5822, doi:10.26421/qic12.11-12-1.

[32] P. C. Costa, D. An, Y. R. Sanders, Y. Su, R. Babbush, and D. W. Berry. Optimal scaling quantum linear-systems solver via discrete adiabatic theorem. *PRX Quantum*, 3:040303, Oct 2022. URL: https://link.aps.org/doi/10.1103/PRXQuantum.3.040303, doi:10.1103/PRXQuantum.3.040303.

[33] P. C. S. Costa, S. Jordan, and A. Ostrander. Quantum algorithm for simulating the wave equation. *Physical Review A*, 99(1):012323, 2019. arXiv:1711.05394. doi:10.1103/PhysRevA.99.012323.

[34] I. Y. Dodin and E. A. Startsev. On applications of quantum computing to plasma simulations. *Physics of Plasmas*, 28(9):092101, 2021. doi:10.1063/5.0056974.

[35] Y. Dong, X. Meng, K. B. Whaley, and L. Lin. Efficient phase-factor evaluation in quantum signal processing. *Phys. Rev. A*, 103(4), 2021. arXiv:2002.11649, doi:10.1103/PhysRevA.103.042419.

[36] A. Engel, G. Smith, and S. E. Parker. Linear embedding of nonlinear dynamical systems and prospects for efficient quantum algorithms. *Physics of Plasmas*, 28(6):062305, 2021. arXiv:2012.06681. doi:10.1063/5.0040313.

[37] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. *arXiv preprint arXiv:1806.01838*, 2018. doi:10.48550/arXiv.1806.01838.

[38] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proc. 51st Annu. ACM SIGACT Symp. Theory Comput.*, pages 193–204, 2019. doi:10.1145/3313276.3316366.

[39] D. Gottlieb and C.-W. Shu. On the Gibbs phenomenon and its resolution. *SIAM Rev.*, 39(4):644–668, 1997. doi:10.1137/S0036144596301390.

[40] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, Mar. 2014.

[41] J. Haah. Product decomposition of periodic functions in quantum signal processing. *Quantum*, 3:190, 2019. arXiv:1806.10236, doi:10.22331/q-2019-10-07-190.

[42] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations (Springer Series in Computational Mathematics)*, volume 31. Springer, 2006.

[43] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations I. nonstiff problems*, volume 29. Springer, 1987. doi:10.1016/0378-4754(87)90083-8.

[44] A. W. Harrow, A. Hassidim, and S. Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103(15):150502, 2009. doi:10.1103/PhysRevLett.103.150502.

[45] S. Jin and N. Liu. Quantum algorithms for computing observables of nonlinear partial differential equations, 2022. arXiv:2202.07834. doi:10.48550/arXiv.2202.07834.

[46] I. Joseph. Koopman-von Neumann approach to quantum simulation of nonlinear classical dynamics. *Physical Review Research*, 2(4):043102, 2020. arXiv:2003.09980. doi:10.1103/PhysRevResearch.2.043102.

[47] M. Kieferová, A. Scherer, and D. W. Berry. Simulating the dynamics of time-dependent Hamiltonians with a truncated Dyson series. *Phys. Rev. A*, 99(4), apr 2019. URL: http://dx.doi.org/10.1103/PhysRevA.99.042314, arXiv:1805.00582, doi:10.1103/PhysRevA.99.042314.

[48] H. Krovi. Improved quantum algorithms for linear and nonlinear differential equations, 2022. URL: http://arxiv.org/abs/2202.01054, arXiv:2202.01054, doi:10.48550/ARXIV.2202.01054.

[49] L. Lin and Y. Tong. Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems. *Quantum*, 4:361, 2020. doi:10.22331/q-2020-11-11-361.

[50] N. Linden, A. Montanaro, and C. Shao. Quantum vs. classical algorithms for solving the heat equation. *Comm. Math. Phys.*, 395(2):601–641, 2022. doi:10.1007/s00220-022-04442-6.

[51] J.-P. Liu, H. Ø. Kolden, H. K. Krovi, N. F. Loureiro, K. Trivisa, and A. M. Childs. Efficient quantum algorithm for dissipative nonlinear differential equations. *Proceedings of the National Academy of Sciences*, 118(35), 2021. arXiv:2011.03185. doi:10.1073/pnas.2026805118.

[52] S. Lloyd, G. De Palma, C. Gokler, B. Kiani, Z.-W. Liu, M. Marvian, F. Tennie, and T. Palmer. Quantum algorithm for nonlinear differential equations, 2020. arXiv:2011.06571. doi:10.48550/arXiv.2011.06571.

[53] G. H. Low. Hamiltonian simulation with nearly optimal dependence on spectral norm. In *Proc. Annu. ACM Symp. Theory Comput.*, pages 491–502, 2019. arXiv:1807.03967, doi:10.1145/3313276.3316386.

[54] G. H. Low and I. L. Chuang. Hamiltonian simulation by uniform spectral amplification. *arXiv:1707.05391*, 2017. doi:10.48550/arXiv.1707.05391.

[55] G. H. Low and I. L. Chuang. Optimal Hamiltonian Simulation by Quantum Signal

Processing. *Phys. Rev. Lett.*, 118(1):10501, 2017. arXiv:1606.02685, doi:10.1103/PhysRevLett.118.010501.

[56] G. H. Low and I. L. Chuang. Hamiltonian simulation by qubitization. *Quantum*, 3:163, 2019. arXiv:1610.06546, doi:10.22331/q-2019-07-12-163.

[57] G. H. Low and N. Wiebe. Hamiltonian Simulation in the Interaction Picture. *arXiv:1805.00675*, 2018. URL: http://arxiv.org/abs/1805.00675, arXiv:1805.00675, doi:10.48550/arXiv.1805.00675.

[58] Y. Subaşl, R. D. Somma, and D. Orsucci. Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Phys. Rev. Lett.*, 122(6):60504, 2019. arXiv:1805.10549, doi:10.1103/PhysRevLett.122.060504.

[59] S. Takahira, A. Ohashi, T. Sogabe, and T. S. Usuda. Quantum algorithm for matrix functions by cauchy's integral formula. *Quantum Inf. Comput.*, 20(1-2):14–36, 2020. arXiv:2106.08075, doi:10.26421/qic20.1-2-2.

[60] Y. Tong, V. V. Albert, J. R. McClean, J. Preskill, and Y. Su. Provably accurate simulation of gauge theories and bosonic systems, 2021. URL: https://arxiv.org/abs/2110.06942, doi:10.48550/ARXIV.2110.06942.

[61] Y. Tong, D. An, N. Wiebe, and L. Lin. Fast inversion, preconditioned quantum linear system solvers, fast Green's-function computation, and fast evaluation of matrix functions. *Phys. Rev. A*, 104(3), 2021. doi:10.1103/PhysRevA.104.032422.

[62] L. N. Trefethen. *Approximation Theory and Approximation Practice, Extended Edition*, volume 164. SIAM, 2019. doi:10.1137/1.9781611975949.

[63] L. N. Trefethen and J. A. Weideman. The exponentially convergent trapezoidal rule. *SIAM Rev.*, 56(3):385–458, 2014. doi:10.1137/130932132.

[64] C. Tronci and I. Joseph. Koopman wavefunctions and Clebsch variables in Vlasov-Maxwell kinetic theory, 2021. doi:10.1017/S0022377821000805.

[65] J. Wang, Y. Dong, and L. Lin. On the energy landscape of symmetric quantum signal processing. *Quantum*, 6:850, Nov. 2022. doi:10.22331/q-2022-11-03-850.

[66] D. Wecker, M. B. Hastings, N. Wiebe, B. K. Clark, C. Nayak, and M. Troyer. Solving strongly correlated electron models on a quantum computer. *Phys. Rev. A - At. Mol. Opt. Phys.*, 92(6):62318, 2015. arXiv:1506.05135, doi:10.1103/PhysRevA.92.062318.

[67] N. Wiebe, D. Berry, P. Høyer, and B. C. Sanders. Higher order decompositions of ordered operator exponentials. *J. Phys. A Math. Theor.*, 43(6), 2010. arXiv:0812.0562, doi:10.1088/1751-8113/43/6/065203.

[68] L. Ying. Stable factorization for phase factors of quantum signal processing. *Quantum*, 6:842, Oct. 2022. doi:10.22331/q-2022-10-20-842.

[69] B. Şahinoğlu and R. D. Somma. Hamiltonian simulation in the low-energy subspace. *npj Quantum Inf.*, 7(1), 2021. arXiv:2006.02660, doi:10.1038/s41534-021-00451-w.

A Notations

Throughout the paper, $\|A\|$ denotes the spectral norm of a matrix A, $\|v\|$ denotes 2-norm of a vector v, and $\mathcal{D}(\cdot,\cdot)$ denotes the trace distance between two quantum states defined as

$$\mathcal{D}(\rho,\sigma) = \frac{1}{2}\operatorname{Tr}\left(\sqrt{(\rho-\sigma)^2}\right).$$

We use the following asymptotic notations besides the usual $\mathcal{O}$ notation: we write $f = \Omega(g)$ if $g = \mathcal{O}(f)$; $f = \Theta(g)$ if $f = \mathcal{O}(g)$ and $g = \mathcal{O}(f)$; $f = \widetilde{\mathcal{O}}(g)$ if $f = \mathcal{O}(g\text{polylog}(g))$. For two quantum states $|x\rangle$ and $|y\rangle$, we sometimes write $|x, y\rangle$ to denote $|x\rangle |y\rangle$. The notations regarding each short time evolution can be summarized as follows. Note that the same U_l can be viewed as a block encoding of Ξ_ℓ and $\bar{\Xi}_\ell$, and the difference is only in the error of the block encoding.

Notation	Meaning	Corresponding block encoding
Ξ_ℓ	the exact time evolution operator $\mathcal{T}e^{\int_{t_{l-1}}^{t_l} A(t)\mathrm{d}t}$	U_l is an $(\alpha_l, m_l, \epsilon_l \lVert\Xi_l\rVert)$-block encoding of Ξ_ℓ
$\bar{\Xi}_\ell$	an approximation of the exact evolution Ξ_ℓ	U_l is an $(\alpha_l, m_l, 0)$-block encoding of $\bar{\Xi}_\ell$
$\widetilde{\Xi}_\ell$	$\bar{\Xi}_\ell$ after uniform singular value amplification	$\widetilde{U}_l$ is a $(\frac{\lVert\bar{\Xi}_\ell\rVert}{1-\delta}, m_l + 1, 0)$-block encoding of $\widetilde{\Xi}_\ell$

B Block encoding and quantum singular value transformation

We call a matrix $A \in \mathbb{C}^{2^n \times 2^n}$ an n-qubit matrix or an n-qubit operator. In this work we extensively use the technique of block encoding [21, 38, 56], which is a way of embedding an arbitrary matrix as a sub-matrix of a larger unitary matrix. Using a unitary matrix U_A to encode A as a sub-matrix means that there exist a subnormalization factor $\alpha > 0$ such that

$$U_A = \begin{pmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix}, \tag{61}$$

where $\cdot$ denotes arbitrary matrix blocks of proper sizes. In general, the matrix that we block encode may only approximate, but is not exactly equal to, A/α. We use the following notation to describe such encodings.

Definition 15 (Block encoding, [38])**.** *An $(m+n)$-qubit unitary operator U_A is called an (α, m, ϵ)-block encoding of an n-qubit operator A, if $\lVert A - \alpha(\langle 0^m| \otimes I_n) U_A (|0^m\rangle \otimes I_n)\rVert \leq \epsilon$.*

Here m is the number of ancilla qubits for block encoding, and α is called the subnormalization factor. With the $(\alpha, m, 0)$-block encoding U_A, we can perform a quantum singular value transformation (QSVT) of A on a quantum computer using the technique developed in [38]. More precisely, suppose A has a singular value decomposition $A = U\sigma V^\dagger$, then for any odd polynomial $f(x)$ with degree d such that $|f(x)| \leq 1$ for $x \in [-1, 1]$, we can construct a block encoding of $Uf(\Sigma/\alpha)V^\dagger$, and this block encoding uses U_A d times. This is the main tool for implementing the uniform singular value amplification in Lemma 2. QSVT requires finding a sequence of phase factors corresponding to the polynomial we want to implement. There are classical algorithms capable of doing this [41] in polynomial time, and later works developed more efficient methods for high-degree polynomials with high precision requirements [22, 35, 65, 68].

C Convex optimization based method for uniform singular value amplification

In order to approximate an odd target function using an odd polynomial of degree d, we can express the target polynomial as the linear combination of Chebyshev polynomials with some

unknown coefficients $\{c_k\}$:

$$F(x) = \sum_{k=0}^{(d-1)/2} T_{2k+1}(x) c_k. \tag{62}$$

To formulate this as a discrete optimization problem, we first discretize $[-1, 1]$ using M grid points (e.g., roots of Chebyshev polynomials $\{x_j = -\cos\frac{j\pi}{M-1}\}_{j=0}^{M-1}$). We define the coefficient matrix, $A_{jk} = T_{2k+1}(x_j), \quad k = 0, \ldots, (d-1)/2$. Then the coefficients for approximating the polynomial used for uniform singular value amplification can be found by solving the following optimization problem

$$\begin{aligned} \min_{\{c_k\}} \quad & \max\left\{ \max_{x_j \in [0, \gamma'^{-1}]} \left| F(x_j) - (1-\delta)\gamma' x_j \right| \right\} \\ \text{s.t.} \quad & F(x_j) = \sum_k A_{jk} c_k, \quad |F(x_j)| \leq c, \quad \forall j = 0, \ldots, M-1. \end{aligned} \tag{63}$$

This is a convex optimization problem and can be solved using software packages such as CVX [40]. The norm constraint $|F(x)| \leq 1$ is relaxed to $|F(x_j)| \leq c$ to take into account that the constraint can only be imposed on the sampled points, and the values of $|F(x)|$ may slightly overshoot on $[-1, 1] \backslash \{x_j\}_{j=0}^{M-1}$. The effect of this relaxation can be negligible when we choose c to be sufficiently close to 1 (for instance, c can be $\max\{0.9999, 1 - 0.1\delta\}$). Since Eq. (63) approximately solves a min-max problem, it achieves the near-optimal solution (in the sense of the L^∞ norm) by definition both in the asymptotic and pre-asymptotic regimes. Once the polynomial $F(x)$ is given, the Chebyshev coefficients can be used as the input to find the phase factors using QSPPACK [1] with an optimization based method [35].

D The compression gadget

In this section we discuss how to construct the compression gadget that is used in Section 2.4 and prove Lemma 3. We suppose we are given unitaries $V_1, V_2, \ldots, V_L$, each of which is a $(\alpha_l', m_l', 0)$-block encoding of a potentially non-unitary operation Γ_l. Our goal is to construct a block encoding of $\Gamma_L \cdots \Gamma_2 \Gamma_1$ without duplicating the ancilla registers in each V_l.

To do this we introduce a counter register to count how many times Γ_l is applied successfully. This counter register contains $\lceil \log_2(L) \rceil + 1$ qubits, and its state encodes a binary that is used to track successful applications of Γ_l. We introduce a unitary operator ADD on this register defined through

$$\mathrm{ADD}\,|c\rangle = |c + 1 \mod 2^{\lceil \log_2(L) \rceil + 1}\rangle\,.$$

This operator performs addition modulo $2^{\lceil \log_2(L) \rceil + 1}$. Its inverse perform subtraction. We initialize the counter register in the state $|L\rangle$, and subtract from the number it encodes by applying $\mathrm{ADD}^\dagger$ each time Γ_l is applied successfully. We keep track of success/failure by applying controlled $\mathrm{ADD}^\dagger$ so that the whole process is coherent. In the end if all steps are successful the counter register will be in the state $|0\rangle$.

The circuit construction for the above coherent procedure is described in Figure 3. We denote $m_{\max} = \max_l m_l'$, and regard each V_l as acting on two registers, one is the ancilla

[1] https://github.com/qsppack/QSPPACK

register containing $m_{\max}$ qubits and the other the state register. If $m'_l < m_{\max}$, then we can simply let $m_{\max} - m'_l$ qubits in the ancilla register remain idle. The circuit in Fig. 3 is in fact a block encoding of $\Gamma_L \cdots \Gamma_2 \Gamma_1$. In this way we prove Lemma 3, which we restate here.

Lemma (Compression gadget)**.** *Suppose we are given unitaries $V_1, V_2, \ldots, V_L$, each of which is a $(\alpha'_l, m'_l, 0)$-block encoding of a potentially non-unitary operation Γ_l. Then we can construct an $(\alpha_{\mathrm{comp}}, m_{\mathrm{comp}}, 0)$-block encoding of $\Gamma_L \cdots \Gamma_2 \Gamma_1$, where*

$$\alpha_{\mathrm{comp}} = \alpha'_1 \alpha'_2 \cdots \alpha'_L, \quad m_{\mathrm{comp}} = \max_l m'_l + \lceil \log_2(L) \rceil + 1.$$

E Bounding the error due to the coefficient matrix

In this section, we show that small errors in the coefficient matrix can be controlled.

Lemma 16. *Let $\mathcal{V}_A(t,s) = \mathcal{T} e^{\int_s^t A(u)\mathrm{d}u}$, and $\mathcal{V}_B(t,s) = \mathcal{T} e^{\int_s^t B(u)\mathrm{d}u}$. Then*

$$\|\mathcal{V}_A(t,s) - \mathcal{V}_B(t,s)\| \leq e^{(t-s)\max_{u\in[s,t]}\{\|A(u)\|,\|B(u)\|\}} \int_s^t \|B(u) - A(u)\|\mathrm{d}u.$$

Proof. We observe that

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathcal{V}_A(t,s) - \mathcal{V}_B(t,s)) = A(t)(\mathcal{V}_A(t,s) - \mathcal{V}_B(t,s)) + (A(t) - B(t))\mathcal{V}_B(t).$$

Using Duhamel's principle we have

$$\mathcal{V}_A(t,s) - \mathcal{V}_B(t,s) = \int_s^t \mathcal{V}_A(t,u)(A(t) - B(u))\mathcal{V}_B(u,s)\mathrm{d}u. \tag{64}$$

Because

$$\|\mathcal{V}_A(t,u)\| \leq e^{\int_u^t \|A(v)\|\mathrm{d}v} \leq e^{(t-u)\max_{v\in[u,t]}\|A(v)\|},$$
$$\|\mathcal{V}_B(t,u)\| \leq e^{\int_s^u \|B(v)\|\mathrm{d}v} \leq e^{(u-s)\max_{v\in[s,u]}\|B(v)\|},$$

Eq. (64) implies

$$\|\mathcal{V}_A(t,s) - \mathcal{V}_B(t,s)\| \leq \int_s^t e^{(t-u)\max_{v\in[u,t]}\|A(v)\| + (u-s)\max_{v\in[s,u]}\|B(v)\|}\|B(u) - A(u)\|\mathrm{d}u.$$

The lemma can then be proved by observing that

$$(t-u)\max_{v\in[u,t]}\|A(v)\| + (u-s)\max_{v\in[s,u]}\|B(v)\| \leq (t-s)\max_{u\in[s,t]}\{\|A(u)\|, \|B(u)\|\}.$$

□

F Bounding numerical integration error using the total variation

In this section, we bound the numerical integration error. The standard numerical quadrature results typically bound the error by the derivative of the matrix $\|\dot{A}\|$, and hence the matrix A needs to be differentiable (see, e.g., [57, Eq. (A11)] or the textbook [19]), Here we bound

the numerical integration error by the total variation instead to account for non-differentiable cases.

For large interval $[0,T]$, we can partition the interval into segments $0 = x_0 < x_1 < \cdots x_K = T$, $x_k - x_{k-1} = \Delta$, and the difference between the integral and the Riemann sum can be bounded through

$$\begin{aligned}\left\|\int_0^T A(s)\mathrm{d}s - \Delta\sum_{k=0}^{K-1} A(x_k)\right\| &\leq \sum_{k=0}^{K-1}\left\|\int_0^\Delta A(x_k+y)\mathrm{d}y - \Delta A(x_k)\right\| \\ &\leq \Delta\sum_{k=0}^{K-1}\int_{x_k}^{x_{k+1}}\|\dot{A}(y)\|\mathrm{d}y \\ &= \Delta\int_0^T \|\dot{A}(y)\|\mathrm{d}y\end{aligned}$$

Now we obtain similar bounds using the total variation. First for short time

$$\left\|\int_0^\Delta A(x_k+s)\mathrm{d}s - \Delta A(x_k)\right\| \leq \int_0^\Delta \|A(x_k+s)-A(x_k)\|\mathrm{d}s \leq \Delta V_{x_k}^{x_k+\Delta}(A),$$

where the second inequality is because $\|A(x_k+s)-A(x_k)\| \leq V_{x_k}^{x_k+\Delta}(A)$. For long time

$$\begin{aligned}\left\|\int_0^T A(s)\mathrm{d}s - \Delta\sum_{k=0}^{K-1} A(x_k)\right\| &\leq \sum_{k=0}^{K-1}\left\|\int_0^\Delta A(x_k+y)\mathrm{d}y - \Delta A(x_k)\right\| \\ &\leq \Delta\sum_{k=0}^{K-1} V_{x_k}^{x_{k+1}}(A) \\ &= \Delta V_0^T(A),\end{aligned}$$

where we have used the fact that

$$V_0^T(A) = \sum_{k=0}^{K-1} V_{x_k}^{x_{k+1}}(A).$$

G Circuit construction of the contour integral formulation

In this section, we construct the circuit to implement e^A using QSVT and the contour integral formulation. We devote Appendix G.1 and Appendix G.2 to the preliminary discussions of the block encoding of the inverse of a matrix and the quadrature discretization errors of the contour integral formulation, respectively. Appendix G.3-Appendix G.5 discuss the circuit construction in details and prove Lemma 12.

G.1 Block encoding the inverse of a matrix

This section follows [61, Appendix B]. We will discuss how to build a block encoding of the inverse of a matrix A, given a block encoding of A.

For an odd function f we define the singular value transformation $f^\diamond$ in the following way: if $M = W_M\Sigma_M V_M^\dagger$, then $f^\diamond(M) = W_M f(\Sigma_M)V_M^\dagger$. This transformation can be implemented

on a quantum computer using the quantum singular value transformation (QSVT) method developed in Ref. [38]. The matrix inversion can be implemented as a singular value transformation in the following way: when $A = W\Sigma V^\dagger$ is invertible, $A^{-1} = V\Sigma^{-1}W^\dagger$. Therefore

$$(A/\alpha)^{-1} = (f^\diamond(A/\alpha))^\dagger \tag{65}$$

where $f(x) = x^{-1}$. However $f(\cdot)$ here is not bounded by 1 and is in fact singular at $x = 0$. Therefore instead of approximating $f(x) = x^{-1}$ on the whole interval $[-1, 1]$ we consider an odd polynomial $p(x)$ such that

$$\left| p(x) - \frac{3\delta}{4x} \right| \leq \epsilon', \quad \forall x \in [-1, -\delta] \cup [\delta, 1].$$

and $|p(x)| \leq 1$ for all $x \in [-1, 1]$. The existence of such an odd polynomial of degree $\mathcal{O}(\frac{1}{\delta}\log(\frac{1}{\epsilon'}))$ is guaranteed by [38, Corollary 69].

Then [38, Theorem 2] enables us to implement $(p^\diamond(A/\alpha))^\dagger = Vp(\Sigma/\alpha)W^\dagger$. We have

$$\|(p^\diamond(A/\alpha))^\dagger - (3\delta/4)(A/\alpha)^{-1}\| = \|p(\Sigma/\alpha) - (3\delta/4)(\Sigma/\alpha)^{-1}\| \leq \epsilon', \tag{66}$$

if all diagonal elements of Σ/α, i.e. the singular values of A/α, are in the interval $[\delta, 1]$. Therefore we want all singular values of A/α to be at least δ distance away from the origin. We then use QSVT to block encode $(p^\diamond(A/\alpha))^\dagger$ given a block encoding of A.

We assume A can be accessed by its $(\alpha, m, 0)$-block encoding U_A. The singular values of A/α are contained in $[1/(\alpha\|A^{-1}\|), \|A\|/\alpha]$. Therefore we choose $\delta = 1/(\alpha\|A^{-1}\|)$. Using QSVT, a $(1, m+1, 0)$-block encoding of $p^\diamond(A/\alpha)$ can be implemented [38, Theorem 2]. We denote this block encoding by $\mathcal{U}$. Then by Eq. (66)

$$\left\| \frac{4}{3\delta\alpha}(\langle 0^{m+1} \otimes I|)\mathcal{U}^\dagger(|0^{m+1} \otimes I\rangle) - A^{-1} \right\| = \left\| \frac{4}{3\delta\alpha}(p^\diamond(A/\alpha)^\dagger) - A^{-1} \right\| \leq \frac{4\epsilon'}{3\delta\alpha}.$$

Consequently, by our choice of δ, $\mathcal{U}^\dagger$ is a $(4\|A^{-1}\|/3, m+1, \epsilon)$-block encoding of A^{-1} where $\epsilon = 4\|A^{-1}\|\epsilon'/3$. Because the cost of QSVT scales linearly with respect to the degree of the polynomial $p(x)$, the total number of queries to to U_A and its inverse is

$$\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\epsilon'}\right)\right) = \mathcal{O}\left(\alpha\|A^{-1}\|\log\left(\frac{\|A^{-1}\|}{\epsilon}\right)\right).$$

We summarize the result in the following Lemma:

Lemma 17. *We assume A can be accessed by its $(\alpha, m, 0)$-block encoding U_A. Then a $(4\|A^{-1}\|/3, m+1, \epsilon)$-block encoding of A^{-1} can be constructed using $\mathcal{O}\left(\alpha\|A^{-1}\|\log\left(\frac{\|A^{-1}\|}{\epsilon}\right)\right)$ applications of (controlled-) U_A and its inverse.*

Note that in the case where $\|A^{-1}\|$ is unknown, it can be replaced with an upper bound of $\|A^{-1}\|$.

G.2 Evaluating contour integrals using the trapezoidal rule

In this section we analyze the error that comes from evaluating contour integrals using the trapezoidal rule. We want to evaluate a matrix function $f(A)$. Because of the fact that A may not be a Hermitian matrix, we cannot directly apply QSVT. As a workaround, we use contour integral and linear combination of unitaries (LCU) [31] to implement this matrix function. By Cauchy's formula we have

$$f(A) = \frac{1}{2\pi i}\int_\Gamma f(z)(z-A)^{-1}\mathrm{d}z, \tag{67}$$

where Γ is a circle with radius β: $\Gamma = \{z = \beta e^{i\theta} : \theta \in \mathbb{R}\}$, and $\|A\| < \beta$. We need to discretize this integral in actual implementation. To do this we use the trapezoidal rule. If we use K grid points on the unit circle we have

$$f_K(A) = \frac{1}{K}\sum_{k=0}^{K-1} f(z_k)z_k(z_k - A)^{-1}, \tag{68}$$

for $z_k = \beta e^{i2\pi k/K}$.

We directly use the result in [63, Theorem 18.1], which has been modified in Ref. [59] to a form more suitable for our purposes:

Lemma 18. *[63, Theorem 18.1], [59, Proposition 5] For a matrix function $f(A)$ and its discretized version $f_K(A)$ in (68), if $f(z)$ is analytic in the disc $\{z : |z| < R\}$, then*

$$\|f(A) - f_K(A)\| \le \frac{\sup_{|z|<R}|f(z)|}{1-\frac{\|A\|}{R}}\left(\frac{1}{1-(\frac{\|A\|}{\beta})^K}\left(\frac{\|A\|}{\beta}\right)^K + \frac{1}{1-(\frac{\beta}{R})^K}\left(\frac{\beta}{R}\right)^K\right). \tag{69}$$

G.3 Block encoding of Ξ

In this section we will only use the second to the fifth registers. We first define

$$|\mathrm{COEF}_k\rangle = \frac{1}{\sqrt{\beta+\alpha}}(\sqrt{z_k}\,|0\rangle + \sqrt{\alpha}\,|1\rangle), \quad |\mathrm{COEF}'_k\rangle = \frac{1}{\sqrt{\beta+\alpha}}((\sqrt{z_k})^*\,|0\rangle - \sqrt{\alpha}\,|1\rangle).$$

Because $|z_k| = \beta$ these are normalized quantum states. We can then implement using $\mathcal{O}(K)$ gates unitaries PREP and PREP$'$ such that

$$\mathrm{PREP}\,|k\rangle\,|0\rangle = |k\rangle\,|\mathrm{COEF}_k\rangle\,, \quad \mathrm{PREP}'\,|k\rangle\,|0\rangle = |k\rangle\,|\mathrm{COEF}'_k\rangle\,. \tag{70}$$

These two unitaries will give us the coefficients z_k we need in Ξ. Now we can construct a unitary SEL:

$$\mathrm{SEL} = (\mathrm{PREP}'\otimes I_{m+n})^\dagger(I_{\log_2(K)}\otimes cU_A)(\mathrm{PREP}\otimes I_{m+n}), \tag{71}$$

where $cU_A = |0\rangle\langle 0|\otimes I + |1\rangle\langle 1|\otimes U_A$ is controlled-U_A. The circuit is given in Fig. 5. One can verify that

$$(I_{\log_2(K)}\otimes\langle 0|\otimes\langle 0^m|\otimes I_n)\mathrm{SEL}(I_{\log_2(K)}\otimes|0\rangle\otimes|0^m\rangle\otimes I_n) = \frac{1}{\beta+\alpha}\Xi.$$

Therefore SEL is a $(\beta+\alpha, m+1, 0)$-block encoding of Ξ. Note that SEL uses U_A only once and $\mathcal{O}(K)$ additional elementary gates.

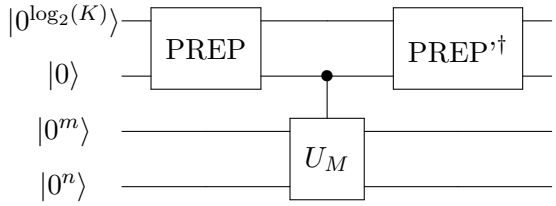


Figure 5: Quantum circuit of implementing the block encoding of the linear combination for each matrix inversion problem $(z_k - U)$. Here PREP and PREP' are defined as (70).

G.4 Block encoding of Ξ^{-1}

Now that we have a block encoding of Ξ we will directly apply Lemma 17 to get a block encoding of Ξ^{-1}. We need to upper bound $\|\Xi^{-1}\|$. First we have

$$\|\Xi^{-1}\| = \max_k \|(z_k - A)^{-1}\| \leq \max_k |z_k|^{-1} \sum_{r=0}^{\infty} \left(\frac{\|A\|}{|z_k|}\right)^r = \max_k \frac{1}{|z_k| - \|A\|} \leq \alpha^{-1},$$

where we have used the fact that $\|A\| \leq \alpha$ and $|z_k| = \beta = 2\alpha$. Using this result, and direct calculation of the other parameters in Lemma 17, it can be seen that we can get a $(4/(3\alpha), m+2, \epsilon')$-block encoding of Ξ^{-1} using $\mathcal{O}(\log(\alpha^{-1}\epsilon'^{-1}))$ applications of (controlled-) SEL and its inverse, which translates to the same number of applications of (controlled-) U_A and its inverse. We denote this block encoding of Ξ^{-1} by $\mathrm{SEL}_{\mathrm{inv}}$. It acts on all 5 registers of the circuit, and the first register contains the ancilla qubit needed for QSVT.

G.5 Using LCU to block encode e^A

We have now come to the final step in which we construct a block encoding of e^A using LCU. As discussed at the beginning of Section 5.3, we need quantum states $|\mathrm{COEF}_{\mathrm{int}}\rangle$, $|\mathrm{COEF}'_{\mathrm{int}}\rangle$ to encode the coefficients. We specify the normalization factor below:

$$|\mathrm{COEF}_{\mathrm{int}}\rangle = \frac{1}{\mathcal{A}K} \sum_k \sqrt{e^{z_k} z_k}\, |k\rangle\,, \quad |\mathrm{COEF}'_{\mathrm{int}}\rangle = \frac{1}{\mathcal{A}K} \sum_k (\sqrt{e^{z_k} z_k})^* \, |k\rangle\,, \tag{72}$$

where

$$\mathcal{A} = \frac{1}{K} \sum_{k=0}^{K-1} |e^{z_k} z_k| = \frac{\beta}{K} \sum_{k=0}^{K-1} |e^{z_k}|.$$

One can see that $0 < \mathcal{A} \leq 2\alpha e^{2\alpha}$. Now we can construct unitaries $\mathrm{PREP}_{\mathrm{int}}$ and $\mathrm{PREP}'_{\mathrm{int}}$ that satisfy

$$\mathrm{PREP}_{\mathrm{int}} |0^{\log_2(K)}\rangle = |\mathrm{COEF}_{\mathrm{int}}\rangle\,, \quad \mathrm{PREP}'_{\mathrm{int}} |0^{\log_2(K)}\rangle = |\mathrm{COEF}'_{\mathrm{int}}\rangle\,. \tag{73}$$

These two unitaries each uses $\mathcal{O}(K)$ elementary gates. Then we let

$$\mathcal{U}_{\mathrm{LCU}} = (I_1 \otimes \mathrm{PREP}'_{\mathrm{int}} \otimes I_{m+n+1})^\dagger \mathrm{SEL}_{\mathrm{inv}} (I_1 \otimes \mathrm{PREP}_{\mathrm{int}} \otimes I_{m+n+1}). \tag{74}$$

The circuit is given by Fig. 6. We can then verify that

$$\begin{aligned}
&(\langle 0| \otimes \langle 0^{\log_2(K)}| \otimes \langle 0| \otimes \langle 0^m| \otimes I_n) \mathcal{U}_{\mathrm{LCU}} (|0\rangle \otimes |0^{\log_2(K)}\rangle \otimes |0\rangle \otimes |0^m\rangle \otimes I_n) \\
&= \frac{3\alpha}{4\mathcal{A}} f_K(A) + \mathcal{O}(\alpha\epsilon') = \frac{3\alpha}{4\mathcal{A}} (f_K(A) + \mathcal{O}(\mathcal{A}\epsilon')).
\end{aligned}$$

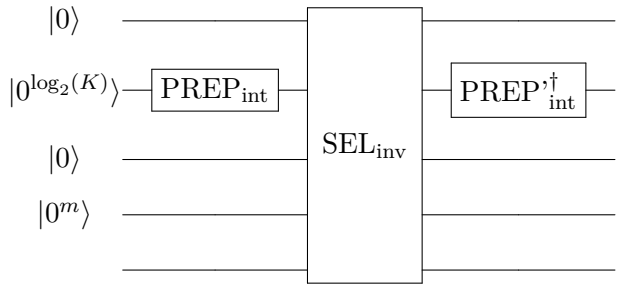


Figure 6: Quantum circuit of implementing the block encoding of the linear combination of the preconditioned matrix inversions. Here $\mathrm{PREP}_{\mathrm{int}}$ and $\mathrm{PREP'}_{\mathrm{int}}$ are defined as (73) and the select oracle $\mathrm{SEL}_{\mathrm{inv}}$ is the standard QSVT circuit for the matrix function as discussed in Appendix G.4.

Therefore $\mathcal{U}_{\mathrm{LCU}}$ is a $(4\mathcal{A}/(3\alpha), m + \log_2(K) + 2, \mathcal{O}(\mathcal{A}\epsilon'))$-block encoding of $f_K(A)$. By (49), it is also a $(4\mathcal{A}/(3\alpha), m + \log_2(K) + 2, \mathcal{O}(\mathcal{A}\epsilon' + e^{4\alpha}2^{-K}))$-block encoding of $f(A)$. In order to get the error to be below ϵ we can choose $\epsilon' = \Theta(\epsilon/\mathcal{A})$, and $K = \mathcal{O}(\alpha + \log(\epsilon^{-1}))$. In the whole process (controlled-) U_A and its inverse are used $\mathcal{O}(\log(\alpha^{-1}\epsilon'^{-1})) = \mathcal{O}(\log(\mathcal{A}\alpha^{-1}\epsilon^{-1})) = \mathcal{O}(\alpha + \log(\epsilon^{-1}))$ times. In sum, we get Lemma 12.