SIAM J. COMPUT.
Vol. 55, No. 2, pp. 376–409




# LAPLACE TRANSFORM–BASED QUANTUM EIGENVALUE TRANSFORMATION VIA LINEAR COMBINATION OF HAMILTONIAN SIMULATION*

DONG AN†, ANDREW M. CHILDS‡, LIN LIN§, AND LEXING YING¶

**Abstract.** Eigenvalue transformations, which include solving time-dependent differential equations as a special case, have a wide range of applications in scientific and engineering computation. While quantum algorithms for singular value transformations are well studied, eigenvalue transformations are distinct, especially for nonnormal matrices. We propose an efficient quantum algorithm for performing a class of eigenvalue transformations that can be expressed as a certain type of matrix Laplace transformation. This allows us to significantly extend the recently developed linear combination of Hamiltonian simulation method [D. An, J.-P. Liu, and L. Lin, *Phys. Rev. Lett.,* 131 (2023), 150603; D. An, A. M. Childs, and L. Lin, Commun. Math. Phys. 407, 19 (2026)] to represent a wider class of eigenvalue transformations, such as powers of the matrix inverse, $A^{-k}$, and the exponential of the matrix inverse, $e^{-A^{-1}}$. The latter can be interpreted as the solution of a mass-matrix differential equation of the form $Au'(t) = -u(t)$. We demonstrate that our eigenvalue transformation approach can solve this problem without explicitly inverting $A$, thereby reducing the computational complexity.

**Key words.** quantum algorithms, eigenvalue transformation, Laplace transform, linear combination of Hamiltonian simulation

**MSC codes.** 81P68, 68Q12, 65F60

**DOI.** 10.1137/24M1720160

## 1. Introduction.

Quantum computers are expected to solve certain computational problems much more efficiently than classical computers, such as factoring large integers [35] and simulating the dynamics of quantum systems [7, 30, 20, 14]. Large scale and high dimensionality appear ubiquitously in scientific and engineering com-

*Received by the editors December 30, 2024; accepted for publication (in revised form) December 2, 2025; published electronically March 17, 2026.
https://doi.org/10.1137/24M1720160

**Funding:** The work of the first author was supported by the United States Department of Defense through the Hartree Postdoctoral Fellowship at QuICS. The work of the second author was supported by the United States Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Accelerated Research in Quantum Computing program, grant DE-SC0025341, and by the National Science Foundation through QLCI grant OMA-2120757. The work of the third and fourth authors was supported by the United States Department of Energy, Office of Science, Accelerated Research in Quantum Computing Centers, Quantum Utility through Advanced Computational Quantum Algorithms through grant DE-SC0025572. In addition, the work of the third author was supported by the National Science Foundation Challenge Institute for Quantum Computation (CIQC) grant OMA-2016245, and by the Simons Investigator Award, which is a grant from the Simons Foundation through grant 825053. The work of the fourth author was supported by the National Science Foundation through grant DMS-2208163.

†Beijing International Center for Mathematical Research, Peking University, and Joint Center for Quantum Information and Computer Science, University of Maryland, College Park, College Park, MD 20742 USA (dongan@pku.edu.cn).

‡Joint Center for Quantum Information and Computer Science, University of Maryland, College Park, and Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, College Park, MD 20742 USA (amchilds@umd.edu).

§Department of Mathematics, University of California, Berkeley, Applied Mathematics and Computational Research Division, Lawrence Berkeley National Laboratory, and Challenge Institute for Quantum Computation, University of California, Berkeley, Berkeley, CA 94720 USA (linlin@math.berkeley.edu).

¶Department of Mathematics and Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA 94305-2125 USA (lexing@stanford.edu).

putation and have posed significant challenges for classical computers, motivating the development of efficient quantum algorithms for such problems.

Many scientific computing tasks can be expressed as eigenvalue transformations. Suppose $A \in \mathbb{C}^{N\times N}$ is diagonalizable as $A = \mathcal{V}D\mathcal{V}^{-1}$, where $\mathcal{V}$ is an invertible matrix and $D$ is a diagonal matrix. Let $h\colon \mathbb{C} \to \mathbb{C}$ be a function defined on the spectrum of $A$. The task of quantum eigenvalue transformation (QEVT) is to encode the matrix

$$h(A) := \mathcal{V}h(D)\mathcal{V}^{-1} \tag{1}$$

using a unitary matrix that can be efficiently implemented on a quantum computer, or to prepare a quantum state proportional to $h(A)|\psi\rangle$ for an input state $|\psi\rangle$. The most well-known instance of this problem involves the linear time-independent differential equation

$$\frac{\mathrm{d}u(t)}{\mathrm{d}t} = -Au(t), \quad 0 \le t \le T. \tag{2}$$

The solution can be expressed as a matrix function $u(T) = e^{-AT}u(0)$, so (2) can be solved by performing the matrix exponential $e^{-AT} = \mathcal{V}h(D)\mathcal{V}^{-1}$, which is an eigenvalue transformation with $h(z) = e^{-zT}$.

Certain eigenvalue transformation problems, such as the matrix inverse $A \mapsto A^{-1}$, can be formulated as singular value transformation problems.[1] This enables the usage of efficient quantum algorithms such as the quantum singular value transformation (QSVT) [20]. However, most eigenvalue transformations, including matrix exponentials for general matrices, cannot be reformulated in such a fashion. Other examples include powers of the matrix inverse $A \mapsto A^{-k}$ ($k \in \mathbb{R}_+$), the exponential of the matrix inverse $A \mapsto e^{-A^{-1}}$, and the matrix sign function $A \mapsto \mathrm{sgn}(A - \mu I)$ for a given $\mu \in \mathbb{R}$, to name a few. Two primary resources should be accounted for in the cost of preparing $h(A)|\psi\rangle$: the number of queries to the matrix $A$ and the number of queries to the input state $|\psi\rangle$.

**1.1. Related works.** We are aware of two strategies for implementing general QEVTs. The first strategy is the contour integral–based formulation [37, 38]. When the function $h$ is analytic within a region enclosing all eigenvalues of $A$ in the complex plane, the contour integral technique can be described as an integral over matrix inverses. After discretization of the contour, the eigenvalue transformation $h(A)$ can be implemented using a linear combination of matrix inverses, where each matrix inverse can be implemented by solving a quantum linear system problem (QLSP), for which many algorithms including QSVT are available [22, 1, 12, 36, 29, 3, 16, 18, 32].

The second strategy for a general eigenvalue transformation uses Taylor expansion of $h(z)$ into a linear combination of monomials and implements each matrix monomial using products of block encodings [20]. However, the monomial coefficients can be arbitrarily large even for well-behaved $h(z)$ such as Chebyshev polynomials, making the Taylor expansion approach highly inefficient. The recent development of QEVT by Low and Su [31] overcomes this problem via a history state polynomial using *stable* polynomial expansions. For instance, for a nonnormal matrix $A$ with real eigenvalues within $[-1, 1]$, this approach introduces a Chebyshev history state to encode all Chebyshev polynomials up to a specified order simultaneously. When the eigenvalues are not real but lie within the unit circle, the method constructs a Faber polynomial

[1] Given a matrix $A$ with singular value decomposition $A = W\Sigma V^\dagger$, the singular value transformation with function $h$ produces the matrix $Wh(\Sigma)V^\dagger$. The matrix inverse can be implemented by applying a singular value transformation on the Hermitian conjugate of $A$ with function $h(x) = 1/x$, since $(A^\dagger)^{-1} = W\Sigma^{-1}V^\dagger$.

approximation via a Faber polynomial history state. Using these history state polynomials, the eigenvalue transformation can be implemented again by solving a QLSP.

Both the contour integral method and the history state polynomial method described above can be used to implement the matrix exponential $e^{-AT}$. However, the connection between the matrix exponential and the differential equation (2) allows us to employ special techniques for this particular matrix eigenvalue problem. One type of algorithm, called the linear system approach [5, 8, 13, 28, 6], involves discretizing the time interval $[0,T]$ into $L$ shorter intervals of length $\Delta t = T/L$. This method uses a short-time propagator to propagate from one time interval to another and formulates a dilated linear system to encode the history state across the interval $[0,T]$. While the best existing algorithm of this type [6] may offer a near-optimal number of queries to $A$, it does not optimize the initial state preparation cost since the construction of such an algorithm uses the quantum linear system algorithm in [16] as a subroutine, which requires multiple copies of the initial state. Recently, [32] proposed a quantum linear system algorithm based on an improved version of the variable-time amplitude amplification, which can achieve optimal query complexity to the initial state. Combined with the block-preconditioning technique, [32] designs a differential equation solver which achieves the optimal state preparation cost, with a slight polylogarithmic overhead in the number of queries to $A$.

The time-marching method [19] also employs a short-time propagator as in the linear system approach and directly encodes the matrix exponential $e^{-AT}$ by performing repeated postselections of the short-time propagators. It can achieve optimal state preparation cost, but the complexity with respect to the number of queries to $A$ is suboptimal. Furthermore, to our knowledge, neither the linear system approach nor the time-marching approach is well-suited for general eigenvalue transformations since they heavily rely on discretizing differential equations in the form of (2).

The linear combination of Hamiltonian simulation (LCHS) method [4] significantly simplifies the process of constructing a block encoding $e^{-AT}$ when the matrix $-A$ is dissipative, in the sense that the Hermitian part $L=(A+A^\dagger)/2$ of the matrix $A$ is a positive semidefinite matrix.[2] The core of LCHS is an identity expressing $e^{-AT}$ as a linear combination of a continuously parameterized family of Hamiltonian simulation problems, which can then be implemented using any quantum Hamiltonian simulation algorithm without resorting to specific short-time integrators. It also achieves optimal state preparation cost for matrix exponentiation. A closely related approach for implementing $e^{-AT}$ is the Schrödingerization method [27], which converts a nonunitary differential equation into a dilated Schrödinger equation with an additional momentum dimension, subject to specific initial conditions. The approaches in [4, 27] are first-order schemes, and the matrix query complexity scales linearly with $1/\epsilon$, where $\epsilon$ is the target precision. Recently, LCHS has been generalized into a family of identities for expressing $e^{-AT}$ with exponentially improved accuracy [2]. This led to the first quantum algorithm to block encode matrix exponentials with both optimal state preparation cost and near-optimal scaling in matrix queries across all parameters, including the target precision $\epsilon$ and simulation time $T$.

The central question of this work is as follows:

*Can the LCHS method be generalized to efficiently represent matrix eigenvalue transformations beyond matrix exponentiation?*

[2]Note that in the analysis of some quantum differential equation algorithms [5, 8, 28], it is assumed that all the eigenvalues of $A$ have nonnegative real parts. Our assumption that the Hermitian part $L$ is positive semidefinite is stronger, which can be shown as follows. Let $\lambda$ be an eigenvalue of $A$ with eigenstate $|v\rangle$. Then, if the Hermitian part $L$ of $A$ is positive semidefinite, we have $0 \le \langle v|(A+A^\dagger)|v\rangle = \langle v|A|v\rangle + \langle v|A^\dagger|v\rangle = 2\mathrm{Re}(\lambda)$.

**1.2. Contribution and main idea.** In this work, we propose the following type of eigenvalue transformation. Let $h(z)$ be a function represented as the Laplace transform of a function $g(t)$:

$$h(z) = \int_0^\infty g(t)e^{-zt}\,\mathrm{d}t, \quad \operatorname{Re} z \geq 0. \tag{3}$$

If $-A$ is dissipative (i.e., all the eigenvalues of $A$ are in the right half-plane), then the eigenvalue transformation $h(A)$ takes the form

$$h(A) = \int_0^\infty g(t)e^{-At}\,\mathrm{d}t. \tag{4}$$

This is a linear combination of matrix exponentials, and each matrix exponential $e^{-At}$ can be further written as an LCHS problem using the technique in [2]. As a result, $h(A)$ is formulated as a double integral of continuously parameterized Hamiltonian simulation problems. We call this the *Laplace transform–based linear combination of Hamiltonian simulations* (Lap-LCHS).

Result 1 (informal version of Theorem 4). *Let $A = L + iH$, where $L = \frac{A+A^\dagger}{2} \succeq 0$ and $H = \frac{A-A^\dagger}{2i}$ are both Hermitian matrices. For a function $h(z) = \int_0^\infty g(t)e^{-zt}\,\mathrm{d}t$ that can be expressed by a Laplace transformation with $g(t)$ being its inverse Laplace transform, we have*

$$h(A) = \int_0^\infty \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} e^{-it(kL+H)}\,\mathrm{d}k\,\mathrm{d}t. \tag{5}$$

*Here*[3] $f(k) = \frac{1}{2\pi e^{-2^\beta} e^{(1+ik)^\beta}}$, $\beta \in (0,1)$.

Using Result 1, we construct an efficient quantum algorithm to implement the matrix function $h(A)$. We truncate and discretize the double integral in (5), approximating $h(A)$ by a weighted sum of Hamiltonian simulation operators $e^{-it(kL+H)}$ for some specific choices of $t$ and $k$. Then our quantum Lap-LCHS algorithm implements each Hamiltonian simulation operator by the optimal time-independent Hamiltonian simulation algorithm based on QSVT [20] and performs the weighted sum using the linear combination of unitaries (LCU) technique [15, 12].

We establish a detailed complexity estimate for the Lap-LCHS algorithm. Here we present the cost for preparing the normalized state $\frac{h(A)|\psi\rangle}{\|h(A)|\psi\rangle\|}$. We also analyze the cost of block encoding $h(A)$ in Theorem 6.

Result 2 (informal version of Corollary 7). *Let $A = L + iH$, where $L = \frac{A+A^\dagger}{2} \succeq 0$ and $H = \frac{A-A^\dagger}{2i}$ are both Hermitian matrices. For a given function $h(z)$, we can prepare the state $h(A)|\psi\rangle / \|h(A)|\psi\rangle\|$ with error at most $\epsilon$ using $\widetilde{\mathcal{O}}(\frac{\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|}(\alpha_A T (\log(\frac{1}{\epsilon}))^{1+o(1)}))$ queries to the matrix $A$ and $\mathcal{O}(\frac{\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|})$ queries to the input state $|\psi\rangle$. Here $\alpha_A \geq \|A\|$, $g(t)$ is the inverse Laplace transform of $h(z)$, $T$ is a truncation parameter such that $\|g\|_{L^1((T,\infty))} = \mathcal{O}(\epsilon\,\|h(A)|\psi\rangle\|)$, and $\|g\|_{L^1(\mathcal{I})} = \int_{\mathcal{I}} |g(t)|\,\mathrm{d}t$ denotes the integral of $|g(t)|$ over the interval $\mathcal{I} \subseteq \mathbb{R}_+$.*

In Result 2, the query complexity to the matrix $A$ depends linearly on the truncation parameter $T$, which depends implicitly on the error $\epsilon$. Specifically, $T$ is chosen

[3] In fact there exists a large family of kernel functions $f(k)$ that can make Result 1 hold (see Theorem 4). The one specified here is nearly optimal in the sense that it achieves almost the fastest possible asymptotic decay along the real axis [2, Proposition 7].

such that the integral $\int_T^\infty |g(t)|\,\mathrm{d}t$ is bounded by $\mathcal{O}(\epsilon\,\|h(A)|\psi\rangle\|)$. This may introduce an additional polynomial computational overhead with respect to $\epsilon$ if $g(t)$ decays only polynomially as $t \to \infty$. Consequently, the overall matrix query complexity with respect to the error should be carefully analyzed on a case-by-case basis.

The contour integral method, QEVT with history state polynomial, and Lap-LCHS each have different ranges of applicability where they are most efficient, so it is difficult to provide a succinct general comparison of these methods. As a rule of thumb, the contour integral method may require a part of the contour to be in the left half of the complex plane, so the cost *can* grow exponentially in $T$ for $h(z)$ of the form in (3). According to [31, Proposition 34], QEVT also requires the Hermitian part $L$ of the matrix $A$ to be positive semidefinite, just as Lap-LCHS does. However, it may be possible to implement a broader range of transformations using QEVT, as it only requires the complex analyticity of the function $h(z)$ over a bounded domain and does not rely on any assumptions regarding the existence of its inverse Laplace transform. Both Lap-LCHS and QEVT can achieve near-optimal complexity with respect to the number of queries to $A$, but Lap-LCHS has a lower state preparation cost. Specifically, Lap-LCHS achieves the optimal state preparation cost, while the cost of QEVT involves additional logarithmic overhead in terms of the polynomial degree, even when combined with the recently developed tunable variable-time amplitude amplification (VTAA) and block-preconditioning techniques [32]. Lap-LCHS is also simpler and might be more feasible in the early fault-tolerant regime by implementing the linear combination stochastically [39, 11], while it remains unclear how to implement QEVT with near-optimal scaling on intermediate-term quantum computers due to the complexity of VTAA.

To demonstrate the computational advantage of Lap-LCHS over other methods for performing eigenvalue transformations, we consider several specific instances. For all problems considered in this work, we find that Lap-LCHS achieves the best known complexity both in terms of the number of queries to $A$ and the initial state preparation oracle.

The first example is the matrix function

$$h(z) = \frac{1}{z}(1 - e^{-zT}) = \int_0^T e^{-zs}\,\mathrm{d}s, \tag{6}$$

so that $g(s)$ is an indicator function on the interval $[0, T]$. This corresponds to the simulation of a special inhomogeneous differential equation

$$\frac{\mathrm{d}u(t)}{\mathrm{d}t} = -Au(t) + |\psi\rangle, \quad u(0) = 0, \quad 0 \le t \le T. \tag{7}$$

Generalizing the homogeneous equation in (2), the inhomogeneous equation can model an external driving force. We apply the Lap-LCHS algorithm and find that in this case it is equivalent to combining the standard LCHS and variation of constants. This example is presented in subsection 4.1.

The second example is the power of matrix inverses

$$h(z) = (\eta + z)^{-p} \tag{8}$$

for some $\eta > 0, p > 0$. This transformation is a generalization of the standard matrix inverse, and the power $p$ does not need to be an integer. Fractional powers of matrices arise in certain Markov chain models and fractional differential equations [24]. Here the matrix function $h(A) = (\eta I + A)^{-p}$ involves a shift by $\eta$. Notice that we can implement $A^{-p}$ if all the eigenvalues of $A+A^\dagger$ are positive. Then we can choose $\eta$ to be the smallest eigenvalue of $(A+A^\dagger)/2$ and consider the matrix function $h(A-\eta I)$. This example is presented in subsection 4.2.

The third example is the simulation of a linear differential equation with a nondiagonal mass matrix:

$$A\frac{\mathrm{d}u(t)}{\mathrm{d}t} = -u(t), \quad 0 \le t \le T. \tag{9}$$

We consider two types of initial conditions, namely

$$u(0) = A^{-1}u_0 \quad \text{and} \quad u(0) = u_0. \tag{10}$$

In both cases, we are given an oracle for preparing the state $|u_0\rangle = u_0/\|u_0\|$. Equation (9) has applications to evolutionary partial differential equations with time-space mixed derivatives in the form of

$$\partial_t \mathcal{L}_x u(t,x) = -u(t,x), \tag{11}$$

where $\mathcal{L}_x = \sum_{k=0}^m a_k \partial_x^k$ is a differential operator with respect to the spatial variable $x$. We can discretize the spatial variable to obtain a semidiscretized partial differential equation in the form of (9), where $A$ is the discrete version of $\mathcal{L}_x$.

We apply the Lap-LCHS algorithm to (9) and analyze its complexity in subsection 4.3. Remarkably, Lap-LCHS can solve (9) (i.e., implement the transformation $e^{-TA^{-1}}$) without explicitly inverting the matrix $A$ and achieves the best known complexity in terms of both the number of queries to $A$ and the initial state preparation oracle.

The last example is the second-order differential equation

$$\frac{\mathrm{d}^2u}{\mathrm{d}t^2} = Au, \quad u(0) = |u_0\rangle, \quad u'(0) = v_0, \tag{12}$$

which has many applications, such as wave equations. These second-order differential equations have a branch of the solution that grows in time. We apply Lap-LCHS to simulate the *decaying* branch of the solution. Note that if we first convert the second-order differential equation into a set of first-order differential equations and apply the matrix exponential solver, the cost will be dominated by the growing solution. Similarly to the first-order case, we may also consider the second-order differential equation with a nondiagonal mass matrix

$$A\frac{\mathrm{d}^2u}{\mathrm{d}t^2} = u, \quad u(0) = |u_0\rangle, \quad u'(0) = v_0. \tag{13}$$

The results are presented in subsection 4.4.

**1.3. Discussion.** Despite significant recent advances in quantum algorithms for linear algebraic transformations, research on eigenvalue transformations for nonnormal matrices remains in a relatively early stage. Consider, for example, Hamiltonian simulation, which is obviously an eigenvalue transformation problem. However, an efficient implementation of $e^{-iHt}$ using the contour integral method or the history state polynomial method is far from obvious, while Lap-LCHS simply reduces to the Hamiltonian simulation problem itself by construction. To our knowledge, when $-A$ satisfies the dissipative condition, Lap-LCHS obtains the lowest complexity among existing approaches for eigenvalue transformations. However, a central unifying structure akin to the qubitization framework that underpins QSVT and prior developments in singular value transformations has yet to be clearly identified for the eigenvalue transformation of nonnormal matrices. Establishing such a unifying perspective could greatly enhance our understanding of quantum algorithms for matrix computation.

**1.4. Organization.** The rest of this paper is organized as follows. First, in section 2, we review two existing approaches for special eigenvalue transformations: QSVT for Hermitian matrix functions and LCHS for matrix exponentials. In section 3, we establish the framework of Lap-LCHS, discuss the construction of our quantum algorithm, and present its complexity analysis. Then we discuss the applications of the Lap-LCHS algorithm in section 4.

## 2. Preliminaries.

### 2.1. Singular value transformation and eigenvalue transformation of Hermitian matrices.

QSVT is a powerful tool for designing quantum algorithms. Given a square matrix $A \in \mathbb{C}^{N\times N}$ with operator norm $\|A\| \le 1$, its singular value decomposition can be expressed as $A = W\Sigma V^\dagger$. Here $W, V$ are $N\times N$ unitary matrices, and $\Sigma$ is a diagonal matrix with diagonal entries in $[0,1]$. QSVT enables an efficient implementation of the singular value transformation for a broad class of polynomials $h\colon [-1,1] \to \mathbb{C}$ of definite parity (i.e., $h$ is even or odd), expressed as

$$A \mapsto h^{\mathrm{SV}}(A) := \begin{cases} W h(\Sigma) V^\dagger & \text{if } h \text{ is odd}, \\ V h(\Sigma) V^\dagger & \text{if } h \text{ is even}. \end{cases} \tag{14}$$

This technique has unified a diverse range of tasks.

When $A$ is a Hermitian matrix and $A \succeq 0$, its eigenvalue decomposition and singular value decomposition coincide, and so do eigenvalue and singular value transformations of $A$.

When $A$ is an indefinite Hermitian matrix, its eigenvalue decomposition is $A = \mathcal{V} D \mathcal{V}^\dagger$ for some unitary matrix $\mathcal{V}$. Its singular value decomposition can be written as $A = W\Sigma V^\dagger$ with $W = \mathcal{V}\operatorname{sign}(D), V = \mathcal{V}$, and $\Sigma = |D|$ is obtained by taking the absolute values of the diagonal elements in $D$. In this case, if $f$ is an odd function, then

$$f^{\mathrm{SV}}(A) = W f(\Sigma) V^\dagger = V f(\operatorname{sign}(D)\Sigma) V^\dagger = \mathcal{V} f(D) \mathcal{V}^\dagger = f(A). \tag{15}$$

If $f$ is an even function, then

$$f^{\mathrm{SV}}(A) = V f(\Sigma) V^\dagger = \mathcal{V} f(D) \mathcal{V}^\dagger = f(A). \tag{16}$$

Therefore, as long as $f$ has definite parity, eigenvalue and singular value transformations of a Hermitian matrix $A$ are the same, so we can implement $f(A)$ by QSVT. If $f$ does not have definite parity, we can still implement $f(A)$ by using QSVT to implement its odd and even parts separately and then using LCU to combine them.

For a more general $A$, there is no longer a clear connection between eigenvalue transformations and singular value transformations. Furthermore, many useful functions of a general matrix $A$ are expressed as eigenvalue transformations rather than singular value transformations. For example, consider the matrix square $A^2$. If $A = PDP^{-1}$ is the eigenvalue decomposition for an invertible matrix $P$, then $A^2 = PD^2P^{-1}$ can be computed by taking the square function on the eigenvalues. However, if we consider the singular value decomposition $A = W\Sigma V^\dagger$, then

$A^2 = W\Sigma V^\dagger W\Sigma V^\dagger$ cannot be expressed as a singular value transformation because $W$ and $V$ are not the same in general. This argument also applies to general matrix polynomials. We provide more examples of eigenvalue transformations in section 4.

**2.2. Linear combination of Hamiltonian simulation for matrix exponentials.** The Cartesian decomposition of $A \in \mathbb{C}^{N\times N}$ is [9, Chapter I]

$$A = L + iH, \tag{17}$$

where the Hermitian matrices

$$L = \frac{A + A^\dagger}{2}, \quad H = \frac{A - A^\dagger}{2i} \tag{18}$$

are called the matrix real and imaginary parts of $A$, respectively. If $L \succeq 0$, then $-A$ is called dissipative. In particular, if $Av = \lambda v$, then

$$\frac{1}{2}(v^\dagger A v + v^\dagger A^\dagger v) = \operatorname{Re}\lambda \, \|v\|^2 = v^\dagger L v \geq 0. \tag{19}$$

Therefore, $\operatorname{Re}\lambda \geq 0$ for any eigenvalue $\lambda$ of the matrix $A$.

The recent generalization of the LCHS formula expresses $e^{-A}$ as an integral of Hamiltonian simulation problems weighted by a kernel.

THEOREM 3 ([2, Theorem 5]). *Let $f(z)$ be a function of $z \in \mathbb{C}$, such that*

1. *(Analyticity) $f(z)$ is analytic on the lower half-plane $\{z : \operatorname{Im}(z) < 0\}$ and continuous on $\{z : \operatorname{Im}(z) \leq 0\}$,*
2. *(Decay) there exists a parameter $\alpha > 0$ such that $|z|^\alpha |f(z)| \leq \widetilde{C}$ for a constant $\widetilde{C}$ when $\operatorname{Im}(z) \leq 0$,*
3. *(Normalization) $\int_{\mathbb{R}} \frac{f(k)}{1-ik} \, \mathrm{d}k = 1$.*

*Consider the Cartesian decomposition of $A$ given in* (17) *and suppose $L \succeq 0$. Then for $t \geq 0$,*

$$e^{-At} = \int_{\mathbb{R}} \frac{f(k)}{1-ik} e^{-it(kL+H)} \, \mathrm{d}k. \tag{20}$$

An asymptotically near-optimal choice of the kernel function is

$$f(z) = \frac{1}{2\pi e^{-2^\beta} e^{(1+iz)^\beta}}, \quad \beta \in (0,1). \tag{21}$$

This function decays at a near-exponential rate of $e^{-c|k|^\beta}$ on the real axis. Furthermore, $\beta$ can be arbitrarily close to (but not equal to) 1 [2, Proposition 7].

**3. Laplace transform–based linear combination of Hamiltonian simulations.** In this section, we present the Lap-LCHS method for QEVTs. We first show the theoretical foundation of Lap-LCHS for representing QEVTs in subsection 3.1. The corresponding quantum algorithm is constructed from the discretized version of the LCHS representation; the numerical discretization is discussed in subsection 3.2. Then, we list the oracles we assume for the quantum algorithm in subsection 3.3, walk through the steps of the quantum algorithm in subsection 3.4, and estimate its complexity in subsection 3.5.

**3.1. Formulation.** Theorem 3 shows that the matrix exponential function $h(z) = e^{-z}$ has an LCHS representation. For a more general $h(z)$, we can apply a matrix version of the Laplace transform given in (4) and represent this transform as a linear combination of matrix exponential functions, which in turn can be interpreted as an LCHS problem. Specifically, we have the following theorem.

THEOREM 4. *Let $f(z)$ satisfy the assumptions in Theorem 3. Define $g(t)$ and $h(z)$ according to the Laplace transform in (3). Consider the Cartesian decomposition of $A$ given in (17) and suppose $L \succeq 0$. Then*

$$h(A) = \int_0^\infty \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} e^{-it(kL+H)} \, \mathrm{d}k \, \mathrm{d}t. \tag{22}$$

*Proof.* By [25, Theorem 6.2.27], two continuous matrix functions are identical if and only if they are identical for all diagonalizable matrices in the domain. Assuming $A = VDV^{-1}$ is diagonalizable, $h(A) = Vh(D)V^{-1}$ is well-defined. By Theorem 3, the right-hand side is equal to

$$\int_0^\infty g(t)e^{-tA} \, \mathrm{d}t = V \left( \int_0^\infty g(t)e^{-tD} \, \mathrm{d}t \right) V^{-1} = Vh(D)V^{-1}. \tag{23}$$

This proves the theorem. ∎

Based on Theorem 4, we may design an algorithm that implements a block encoding of $h(A)$ and prepares a quantum state proportional to $h(A)|\psi\rangle$. We first discretize the integral (22) to approximate $h(A)$ by a discrete linear combination of multiple Hamiltonian simulation problems. The algorithm then implements each Hamiltonian simulation time-evolution operator by QSVT for optimal asymptotic scaling and combines the simulations using the LCU method.

**3.2. Discretization of the integral.** We first truncate (22) into a finite domain $[-K, K] \times [0, T]$, and then use the simplest Riemann sum to approximate the integral. Specifically, let

$$U(k,t) = e^{-it(kL+H)}. \tag{24}$$

We use the approximations

$$h(A) = \int_0^\infty \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} U(k,t) \, \mathrm{d}k \, \mathrm{d}t \tag{25}$$

$$\approx \int_0^T \int_{-K}^K \frac{f(k)g(t)}{1-ik} U(k,t) \, \mathrm{d}k \, \mathrm{d}t \tag{26}$$

$$\approx \sum_{j=0}^{M_k-1} \sum_{l=0}^{M_t-1} c_j \hat{c}_l U(k_j, t_l). \tag{27}$$

We denote by $M_k$ and $M_t$ the numbers of grid points used to discretize the variables $k$ and $t$, respectively. The corresponding step sizes are $h_k = 2K/M_k$ and $h_t = T/M_t$, and the grid nodes are defined by $k_j = -K + jh_k$ and $t_l = lh_t$. Here, with some slight abuse of notation, we use $h_k$, $h_t$, $M_k$, and $M_t$ as constants that are independent of the index variables $k_j$ and $t_l$. The associated coefficients are given by

$$c_j = h_k \frac{f(k_j)}{1-ik_j}, \quad \hat{c}_l = h_t g(t_l). \tag{28}$$

We bound the approximation errors in the following result, whose proof can be found in Appendix A.

LEMMA 5. *Suppose that* $g(t)\in C^1(\mathbb{R}_+)\cap L^1(\mathbb{R}_+)$. *Then*

1. *the truncation error can be bounded as*

$$\left\| h(A) - \int_0^T \int_{-K}^{K} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t \right\| \le \|f\|_{L^1(\mathbb{R})}\|g\|_{L^1((T,\infty))} + \|f\|_{L^1(\mathbb{R}\setminus[-K,K])}\|g\|_{L^1(\mathbb{R}_+)}; \tag{29}$$

2. *the quadrature error can be bounded as*

$$\begin{aligned}&\left\| \int_0^T \int_{-K}^{K} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t - \sum_{j=0}^{M_k-1}\sum_{l=0}^{M_t-1} c_j \hat{c}_l U(k_j,t_l) \right\| \\ &\le 2KTh_k\Big(\|f\|_{L^\infty(\mathbb{R})}\|tg\|_{L^\infty(\mathbb{R}_+)}\|A\| + \Big(\|f'\|_{L^\infty(\mathbb{R})}+\|f\|_{L^\infty(\mathbb{R})}\Big)\|g\|_{L^\infty(\mathbb{R}_+)}\Big) \\ &\quad + 2KTh_t\big(2\|f\|_{L^\infty(\mathbb{R})}\|g\|_{L^\infty(\mathbb{R}_+)}\|A\| \\ &\quad + \|f\|_{L^\infty(\mathbb{R})}\|g'\|_{L^\infty(\mathbb{R}_+)}\big);\end{aligned} \tag{30}$$

3. *in order to bound the overall approximation error* $\|h(A)-\sum_{j,l}c_j\hat{c}_lU(k_j,t_l)\|$ *by* $\epsilon$, *it suffices to choose* $K$ *and* $T$ *such that* $\|g\|_{L^1((T,\infty))}=\mathcal{O}(\epsilon/\|f\|_{L^1(\mathbb{R})})$ *and* $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])}=\mathcal{O}(\epsilon/\|g\|_{L^1(\mathbb{R}_+)})$, *and the numbers of grid points to be*

$$\begin{aligned}M_k = \mathcal{O}\Bigg(\frac{K^2T}{\epsilon}\Big(&\|f\|_{L^\infty(\mathbb{R})}\|tg\|_{L^\infty(\mathbb{R}_+)}\|A\| \\ &+ \Big(\|f'\|_{L^\infty(\mathbb{R})}+\|f\|_{L^\infty(\mathbb{R})}\Big)\|g\|_{L^\infty(\mathbb{R}_+)}\Big)\Bigg)\end{aligned} \tag{31}$$

*and*

$$M_t = \mathcal{O}\left(\frac{KT^2}{\epsilon}\left(\|f\|_{L^\infty(\mathbb{R})}\|g\|_{L^\infty(\mathbb{R}_+)}\|A\| + \|f\|_{L^\infty(\mathbb{R})}\|g'\|_{L^\infty(\mathbb{R}_+)}\right)\right). \tag{32}$$

Lemma 5 shows that the quadrature error and the number of grid points depend polynomially on the truncation parameters $K$ and $T$ such that the tail integrals of $f(k)$ and $g(t)$, respectively, are sufficiently small. If we use the asymptotically near-optimal kernel function $f(k)$ defined in (21), then it suffices to choose $K = \mathcal{O}((\log(\|g\|_{L^1(\mathbb{R}_+)}/\epsilon))^{1/\beta})$ for any $\beta\in(0,1)$, which scales only polylogarithmically in the inverse error. The scaling of $T$, however, depends on the decay rate of the inverse Laplace transform $g(t)$ as $t\to\infty$ and should be carefully analyzed on a case-by-case basis. For example, in computing the power of matrix inverse (see subsection 4.2) and solving linear differential equations with mass matrices (see subsection 4.3), the inverse Laplace transforms both decay exponentially in $t$, resulting in a polylogarithmic scaling of $T$ with respect to $1/\epsilon$.

In Lemma 5, we assume the inverse Laplace transform $g(t)$ to be in $L^1(\mathbb{R}_+)$. Notice that (22) in Theorem 4 does not need $g\in L^1(\mathbb{R}_+)$, but we will construct our quantum algorithms by discretizing (22) and applying the quantum LCU subroutine, which requires $g\in L^1(\mathbb{R}_+)$. Nevertheless, there are still some cases where $g(t)$ satisfies (22), is not in $L^1(\mathbb{R}_+)$, and allows an efficient quantum implementation. An example is

$$g(t) = \sum_j \delta_{t_j}(t) + \widetilde{g}(t), \tag{33}$$

where $\delta_{t_j}(t)$ is the Dirac delta function at $t_j \geq 0$ and $\widetilde{g}(t)$ is in $L^1(\mathbb{R}_+)$. In this case, by (22), the corresponding matrix function $h(A)$ can be written as

$$h(A) = \sum_j \int_{\mathbb{R}} \frac{f(k)}{1-ik} e^{-it_j(kL+H)} \,\mathrm{d}k + \int_0^\infty \int_{\mathbb{R}} \frac{f(k)\widetilde{g}(t)}{1-ik} e^{-it(kL+H)} \,\mathrm{d}k \,\mathrm{d}t. \tag{34}$$

Each integral in the summation is just the matrix exponential $e^{-At_j}$ according to Theorem 3 and can be efficiently implemented by the standard LCHS algorithm [2]. However, for technical simplicity, in this work we mainly focus on the case where the inverse Laplace transform is assumed to be in $L^1(\mathbb{R}_+)$.

**3.3. Oracles.** For a matrix $A$, we typically assume access to a block encoding $O_A$ such that

$$(\langle 0| \otimes I)\, O_A\, (|0\rangle \otimes I) = \frac{A}{\alpha_A}. \tag{35}$$

Here $\alpha_A \geq \|A\|$ is the block encoding factor. We may alternatively assume access to block encodings $O_L$ and $O_H$ of the matrix real and imaginary parts of $A$, such that

$$(\langle 0| \otimes I)\, O_L\, (|0\rangle \otimes I) = \frac{L}{\alpha_L}, \quad (\langle 0| \otimes I)\, O_H\, (|0\rangle \otimes I) = \frac{H}{\alpha_H}. \tag{36}$$

Notice that if we have $O_A$, then according to the equations $L = (A + A^\dagger)/2$ and $H = (A - A^\dagger)/(2i)$, we may apply LCU to construct $O_L$ and $O_H$ with $\alpha_L = \alpha_H = \alpha_A$, using one additional ancilla qubit and two queries to $O_A$. On the other hand, a block encoding of $A$ can also be constructed from block encodings of $L$ and $H$ by LCU with $\alpha_A = \alpha_L + \alpha_H$ and one query to each of $O_L$ and $O_H$.

If we are interested in approximating the state $h(A)|\psi\rangle / \|h(A)|\psi\rangle\|$, we assume a state preparation oracle $O_\psi$ for $|\psi\rangle$ such that $O_\psi|0\rangle = |\psi\rangle$.

In our algorithm, we use a few more unitaries to encode information about the quadrature formula (27) into quantum states. These unitaries can typically be constructed with much lower cost than the block encoding $O_A$ of the matrix $A$ and the state preparation oracle $O_\psi$ of the input state, as they are independent of the dimension of the eigenvalue transformation problem. We refer to them as *quadrature unitaries* and discuss their definitions and constructions here.

For the nodes $k_j$ and $t_l$, we can construct unitaries $O_k$ and $O_t$ to encode them in binary, i.e.,

$$O_k|j\rangle|0\rangle = |j\rangle|k_j\rangle, \quad O_t|l\rangle|0\rangle = |l\rangle|t_l\rangle. \tag{37}$$

For the coefficients, we construct a pair of state preparation unitaries. Specifically, for $c = (c_0, \ldots, c_{M_k-1})$, we assume $(O_{c,l}, O_{c,r})$ such that

$$O_{c,l}|0\rangle = \frac{1}{\sqrt{\|c\|_1}} \sum_{j=0}^{M_k-1} \overline{\sqrt{c_j}}|j\rangle, \quad O_{c,r}|0\rangle = \frac{1}{\sqrt{\|c\|_1}} \sum_{j=0}^{M_k-1} \sqrt{c_j}|j\rangle. \tag{38}$$

Here $\overline{z}$ denotes the complex conjugate of $z \in \mathbb{C}$, and $\sqrt{\cdot}$ denotes the principal branch of the square root. Similarly, for $\hat{c} = (\hat{c}_0, \ldots, \hat{c}_{M_t-1})$, we construct $(O_{\hat{c},l}, O_{\hat{c},r})$ such that

$$O_{\hat{c},l}|0\rangle = \frac{1}{\sqrt{\|\hat{c}\|_1}} \sum_{l=0}^{M_t-1} \overline{\sqrt{\hat{c}_l}}|l\rangle, \quad O_{\hat{c},r}|0\rangle = \frac{1}{\sqrt{\|\hat{c}\|_1}} \sum_{l=0}^{M_t-1} \sqrt{\hat{c}_l}|l\rangle. \tag{39}$$

Notice that the unitaries $O_{c,l}$, $O_{c,r}$, $O_{\hat{c},l}$, and $O_{\hat{c},r}$ prepare superpositions of $M$ basis states, where $M$ can be polynomial in terms of $K$, $T$, and $\epsilon$ according to Lemma 5. In general, we can prepare an $M$-dimensional quantum state with cost $\mathcal{O}(M)$ [34], but this might incur a gate complexity that is polynomial in the inverse error $1/\epsilon$, as $M$ scales polynomially in $1/\epsilon$ for low-order quadrature formulas (as shown in Lemma 5). However, since the amplitudes of the states are known integrable functions evaluated at discrete points, the state preparation circuits can be constructed more efficiently, in time only $\mathcal{O}(\operatorname{poly}\log M)$ [21, 33] (see Appendix B for a detailed discussion). Alternatively, we may use a high-order quadrature rule for integral discretization following [2], such as composite Gaussian quadrature, to improve the scaling of $M$ on $K$, $T$, and $\epsilon$, at the expense of introducing high-order derivative dependence.

**3.4. Algorithm.** The basic approach to implementing $h(A)$ is to use LCU to linearly combine Hamiltonian simulation operators $U(k_j, t_l)$. We first describe the construction of a select oracle for $U(k,t)$, a prerequisite for LCU. We can write

$$U(k,t) = e^{-it(kL+H)} = e^{-iT(K\alpha_L+\alpha_H)\widetilde{H}}, \quad \widetilde{H} = \frac{t}{T}\frac{1}{K\alpha_L+\alpha_H}(kL+H). \tag{40}$$

Thus, it suffices to construct a coherent block encoding of $\widetilde{H}$ and use it as the input model for the QSVT circuit for the time-evolution operator $e^{-iT(K\alpha_L+\alpha_H)\widetilde{H}}$.

Let us start with the state

$$|j\rangle|l\rangle|0\rangle_j|0\rangle_l|0\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a|\psi\rangle. \tag{41}$$

Here $|\psi\rangle$ is the input state in the system register, and $|j\rangle$ and $|l\rangle$ are indices for the $k$ and $t$ variables, respectively. We introduce six additional ancilla registers. Specifically, $|0\rangle_j$ and $|0\rangle_l$ encode in binary the nodes $k_j$ and $t_l$, respectively; $|0\rangle_R$, $|0\rangle_{R_k}$, and $|0\rangle_{R_t}$ are single qubits for rotations; and $|0\rangle_a$ represents the ancilla register of the block encoding. We first apply $O_k$ and $O_t$ to compute the nodes, giving

$$|j\rangle|l\rangle|k_j\rangle_j|t_l\rangle_l|0\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a|\psi\rangle. \tag{42}$$

Now, we construct a controlled block encoding of $kL+H$ by LCU. Applying a controlled rotation c-R acting as

$$\text{c-R}: |k\rangle|0\rangle \to |k\rangle\left(\frac{\sqrt{\alpha_L k}}{\sqrt{\alpha_L|k|+\alpha_H}}|0\rangle + \frac{\sqrt{\alpha_H}}{\sqrt{\alpha_L|k|+\alpha_H}}|1\rangle\right), \tag{43}$$

we obtain

$$|j\rangle|l\rangle|k_j\rangle_j|t_l\rangle_l\left(\frac{\sqrt{\alpha_L k_j}}{\sqrt{\alpha_L|k_j|+\alpha_H}}|0\rangle_R + \frac{\sqrt{\alpha_H}}{\sqrt{\alpha_L|k_j|+\alpha_H}}|1\rangle_R\right)|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a|\psi\rangle. \tag{44}$$

Applying the controlled block encodings $|0\rangle_R\langle 0|_R \otimes U_L$ and $|1\rangle_R\langle 1|_R \otimes U_H$ yields

$$|j\rangle|l\rangle|k_j\rangle_j|t_l\rangle_l\frac{\sqrt{\alpha_L k_j}}{\sqrt{\alpha_L|k_j|+\alpha_H}}|0\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a\frac{L}{\alpha_L}|\psi\rangle \tag{45}$$

$$+|j\rangle|l\rangle|k_j\rangle_j|t_l\rangle_l\frac{\sqrt{\alpha_H}}{\sqrt{\alpha_L|k_j|+\alpha_H}}|1\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a\frac{H}{\alpha_H}|\psi\rangle + |\perp_a\rangle. \tag{46}$$

Here, $|\perp_a\rangle$ is a possibly unnormalized state orthogonal to $|0\rangle_a$ in the ancilla register labeled by $a$. Later on, we will append more subscripts to indicate that it is orthogonal in more ancilla registers. Applying c-$\overline{\mathrm{R}}^\dagger$ where c-$\overline{\mathrm{R}}$ is the controlled rotation gate obtained by taking the conjugates of the coefficients in c-R, we obtain

$$|j\rangle|l\rangle|k_j\rangle_j|t_l\rangle_l|0\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a\frac{k_jL+H}{\alpha_L|k_j|+\alpha_H}|\psi\rangle+|\perp_{R,a}\rangle. \tag{47}$$

We further shrink the rescaling factor of $k_jL+H$ by applying two additional controlled rotations on $R_k$ and $R_t$, acting as

$$|k\rangle|0\rangle\to|k\rangle\left(\frac{\alpha_L|k|+\alpha_H}{\alpha_LK+\alpha_H}|0\rangle+\sqrt{1-\left|\frac{\alpha_L|k|+\alpha_H}{\alpha_LK+\alpha_H}\right|^2}|1\rangle\right) \tag{48}$$

and

$$|t\rangle|0\rangle\to|t\rangle\left(\frac{t}{T}|0\rangle+\sqrt{1-\left|\frac{t}{T}\right|^2}|1\rangle\right), \tag{49}$$

respectively. Then we obtain

$$|j\rangle|l\rangle|k_j\rangle_j|t_l\rangle_l|0\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a\frac{t_l(k_jL+H)}{T(\alpha_LK+\alpha_H)}|\psi\rangle+|\perp_{R,R_k,R_t,a}\rangle. \tag{50}$$

Uncomputing the $j$ and $l$ registers by applying $O_k^\dagger$ and $O_t^\dagger$ gives

$$|j\rangle|l\rangle|0\rangle_j|0\rangle_l|0\rangle_R|0\rangle_{R_k}|0\rangle_{R_t}|0\rangle_a\frac{t_l(k_jL+H)}{T(\alpha_LK+\alpha_H)}|\psi\rangle+|\perp_{R,R_k,R_t,a}\rangle. \tag{51}$$

We have shown how to apply a sequence of operators to map (42) to (51). By the definition of block encoding, this sequence of operators, denoted by $U_{t(kL+H)}$, is indeed a controlled block encoding of $t(kL+H)$, satisfying

$$(\langle 0|_{a'}\otimes I)\,U_{t(kL+H)}\,(|0\rangle_{a'}\otimes I)=\sum_{j=0}^{M_k-1}\sum_{l=0}^{M_t-1}|j\rangle\langle j|\otimes|l\rangle\langle l|\otimes\frac{t_l(k_jL+H)}{T(\alpha_LK+\alpha_H)}. \tag{52}$$

Here, the ancilla register labeled by $a'$ combines all the previous ancilla registers.

Now we use $U_{t(kL+H)}$ as the block encoding in the QSVT circuit for the time-evolution operator $e^{-iT(\alpha_LK+\alpha_H)\widetilde{H}}$. Then, by [20, Corollary 60], we obtain the select oracle

$$\mathrm{SEL}=\sum_{j=0}^{M_k-1}\sum_{l=0}^{M_t-1}|j\rangle\langle j|\otimes|l\rangle\langle l|\otimes W_{j,l}. \tag{53}$$

Here $W_{j,l}$ block encodes another matrix $V_{j,l}\approx U(k_j,t_l)$. Finally, by the LCU lemma (see, e.g., the version in [2, Lemma 22]), the operator

$$(O_{c,l}^\dagger\otimes O_{\hat{c},l}^\dagger\otimes I)\mathrm{SEL}(O_{c,r}\otimes O_{\hat{c},r}\otimes I) \tag{54}$$

gives a block encoding of $\frac{1}{\|c\|_1\|\hat{c}\|_1}\sum_{j,l}c_j\hat{c}_lV_{j,l}$, which is an approximation of $\frac{1}{\|c\|_1\|\hat{c}\|_1}h(A)$.

**3.5. Complexity analysis.**

THEOREM 6. *Let $f, g, h$ be functions satisfying the assumptions in Theorem 4 and Lemma 5, and let $A$ be a matrix with a positive semidefinite real part. Suppose that we are given the oracles in subsection 3.3. Then Lap-LCHS can implement an $(\alpha, \epsilon)$-block encoding*[4] *of $h(A)$ with the following properties.*

1. *The block encoding factor satisfies*

$$\alpha = \mathcal{O}(\|f\|_{L^1(\mathbb{R})} \|g\|_{L^1(\mathbb{R}_+)}). \tag{55}$$

2. *It suffices to choose $K$ and $T$ such that $\|g\|_{L^1((T,\infty))} = \mathcal{O}(\epsilon / \|f\|_{L^1(\mathbb{R})})$ and $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])} = \mathcal{O}(\epsilon / \|g\|_{L^1(\mathbb{R}_+)})$, and choose $M_k, M_t$ as in Lemma 5.*
3. *The algorithm uses*

$$\mathcal{O}\left(\alpha_A KT + \log\left(\frac{\|f\|_{L^1(\mathbb{R})} \|g\|_{L^1(\mathbb{R}_+)}}{\epsilon}\right)\right) \tag{56}$$

*queries to $O_A$, and $\mathcal{O}(1)$ queries to the oracles $O_k, O_t, O_c, O_{\hat{c}}$.*

*Proof.* Following the steps in subsection 3.4, we can construct a controlled $(T(\alpha_L K + \alpha_H), 0)$-block encoding of $t(kL + H)$, using $\mathcal{O}(1)$ queries to $O_A$. According to [20, Corollary 60], for $\epsilon_1 > 0$ to be determined later, we can implement

$$\text{SEL} = \sum_{j=0}^{M_k-1} \sum_{l=0}^{M_t-1} |j\rangle\langle j| \otimes |l\rangle\langle l| \otimes W_{j,l}, \tag{57}$$

where $W_{j,l}$ is a $(1,0)$-block encoding of $V_{j,l}$ such that $\|V_{j,l} - U(k_j, t_l)\| \le \epsilon_1$, using

$$\mathcal{O}\left(T(\alpha_L K + \alpha_H) + \log\left(\frac{1}{\epsilon_1}\right)\right) = \mathcal{O}\left(\alpha_A KT + \log\left(\frac{1}{\epsilon_1}\right)\right) \tag{58}$$

queries to $O_A$. Then the LCU lemma (see, e.g., [2, Lemma 22]) ensures that $(O_{c,l}^\dagger \otimes O_{\hat{c},l}^\dagger \otimes I)\text{SEL}(O_{c,r} \otimes O_{\hat{c},r} \otimes I)$ gives a $(\|c\|_1 \|\hat{c}\|_1, 0)$-block encoding of $\sum_{j,l} c_j \hat{c}_l V_{j,l}$. By the definition of the coefficients in (28), we can further bound the block encoding factor as $\|c\|_1 \|\hat{c}\|_1 = \mathcal{O}(\|f\|_{L^1(\mathbb{R})} \|g\|_{L^1(\mathbb{R}_+)})$.

We now estimate the approximation error between $\sum_{j,l} c_j \hat{c}_l V_{j,l}$ and the ideal operator $h(A)$ and determine the choices of the parameters. By the triangle inequality, we have

$$\begin{aligned} \left\|h(A) - \sum_{j,l} c_j \hat{c}_l V_{j,l}\right\| &\le \left\|\sum_{j,l} c_j \hat{c}_l V_{j,l} - \sum_{j,l} c_j \hat{c}_l U(k_j, t_l)\right\| \\ &\quad + \left\|h(A) - \sum_{j,l} c_j \hat{c}_l U(k_j, t_l)\right\| && (59) \\ &\le \sum_{j,l} |c_j||\hat{c}_l| \|V_{j,l} - U(k_j, t_l)\| + \left\|h(A) - \sum_{j,l} c_j \hat{c}_l U(k_j, t_l)\right\| && (60) \\ &\le \|c\|_1 \|\hat{c}\|_1 \epsilon_1 + \left\|h(A) - \sum_{j,l} c_j \hat{c}_l U(k_j, t_l)\right\|. && (61) \end{aligned}$$

[4] An $(\alpha, \epsilon)$-block encoding of a matrix $B$ is a unitary that block encodes a matrix $\widetilde{B}$ with $\|B - \alpha\widetilde{B}\| \le \epsilon$.

To bound this by $\epsilon$, it suffices to choose

$$\epsilon_1 = \mathcal{O}\left(\frac{\epsilon}{\|f\|_{L^1(\mathbb{R})}\|g\|_{L^1(\mathbb{R}_+)}}\right) \tag{62}$$

and bound the quadrature error $\|h(A) - \sum_{j,l} c_j \hat{c}_l U(k_j, t_l)\|$ by $\mathcal{O}(\epsilon)$. Lemma 5 gives sufficient choices of $K, T$ and $M_k, M_t$. ∎

The state $h(A)|\psi\rangle / \|h(A)|\psi\rangle\|$ can then be approximated by applying the block encoding of $h(A)$ onto the input state $|\psi\rangle$ and boosting the success probability by amplitude amplification.

Corollary 7. *Let $f, g, h$ be functions satisfying the assumptions in Theorem* 4 *and Lemma* 5, *let $A$ be a matrix with positive semidefinite real part, and let $|\psi\rangle$ be an input state. Suppose that we are given the oracles described in subsection* 3.3. *Then Lap-LCHS can prepare an $\epsilon$-approximation of the state $h(A)|\psi\rangle / \|h(A)|\psi\rangle\|$ with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left(\frac{\|f\|_{L^1(\mathbb{R})}\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|}\left(\alpha_A KT + \log\left(\frac{\|f\|_{L^1(\mathbb{R})}\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|\,\epsilon}\right)\right)\right) \tag{63}$$

*queries to $O_A$, and*

$$\mathcal{O}\left(\frac{\|f\|_{L^1(\mathbb{R})}\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|}\right) \tag{64}$$

*queries to $O_\psi$ and quadrature unitaries. Here $K$ and $T$ are truncation parameters such that $\|g\|_{L^1((T,\infty))} = \mathcal{O}(\epsilon\,\|h(A)|\psi\rangle\| / \|f\|_{L^1(\mathbb{R})})$ and $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])} = \mathcal{O}(\epsilon\,\|h(A)|\psi\rangle\| / \|g\|_{L^1(\mathbb{R}_+)})$.*

*Proof.* We first use Theorem 6 to construct an $(\alpha, \epsilon')$-block encoding of $h(A)$, where the precision parameter $\epsilon'$ will be determined later. Applying this block encoding to $|0\rangle|\psi\rangle$ gives

$$\frac{1}{\alpha}|0\rangle B|\psi\rangle + |\perp\rangle, \tag{65}$$

where $B$ is an operator such that $\|B - h(A)\| \le \epsilon'$. Upon projecting the ancilla onto $|0\rangle$, we obtain the state $B|\psi\rangle / \|B|\psi\rangle\|$. Furthermore,

$$\left\|\frac{B|\psi\rangle}{\|B|\psi\rangle\|} - \frac{h(A)|\psi\rangle}{\|h(A)|\psi\rangle\|}\right\| \le \frac{2}{\|h(A)|\psi\rangle\|}\|B|\psi\rangle - h(A)|\psi\rangle\| \le \frac{2\epsilon'}{\|h(A)|\psi\rangle\|}. \tag{66}$$

To bound this by $\epsilon$, it suffices to choose $\epsilon' = \|h(A)|\psi\rangle\|\,\epsilon/2$. According to Theorem 6, in each run of the algorithm, we use

$$\mathcal{O}\left(\alpha_A KT + \log\left(\frac{\|f\|_{L^1(\mathbb{R})}\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|\,\epsilon}\right)\right) \tag{67}$$

queries to $O_A$, and $\mathcal{O}(1)$ queries to the state preparation oracle $O_\psi$ and quadrature unitaries.

With amplitude amplification, the number of repetitions for constant success probability is

$$\mathcal{O}\left(\frac{\alpha}{\|B|\psi\rangle\|}\right) = \mathcal{O}\left(\frac{\|f\|_{L^1(\mathbb{R})}\|g\|_{L^1(\mathbb{R}_+)}}{\|h(A)|\psi\rangle\|}\right), \tag{68}$$

where we have used $\|B|\psi\rangle\| \ge \|h(A)|\psi\rangle\| - \epsilon' = (1 - \epsilon/2)\,\|h(A)|\psi\rangle\|$. This contributes to another multiplicative factor in the claimed complexity and completes the proof. ∎

**4. Applications.** Here, we discuss several applications of eigenvalue transformation problems and analyze the query complexity of the Lap-LCHS algorithm for them. These applications include linear inhomogeneous differential equations, powers of the matrix inverse, linear differential equations with nonnormal mass matrices, and second-order differential equations. Throughout this section, in the LCHS formula we always use the asymptotically optimal kernel function $f(k)$ defined in (21). We regard the parameter $\beta \in (0,1)$ in (21) as fixed; the big-$\mathcal{O}$ constants in this section may depend on $\beta$.

**4.1. Linear inhomogeneous differential equations.** Consider the linear inhomogeneous differential equation in (7). By variation of constants, the solution is

$$u(T) = \int_0^T e^{-A(t-s)}|\psi\rangle \, \mathrm{d}s = h(A)|\psi\rangle, \tag{69}$$

where

$$h(z) = \frac{1}{z}(1 - e^{-Tz}). \tag{70}$$

This function has the inverse Laplace transform $g(t) = \mathbf{1}_{[0,T]}(t)$, which is the indicator function. Lap-LCHS for evaluating $h(A)|\psi\rangle$ uses the integral representation in (69). The overall complexity can be estimated as a direct consequence of Corollary 7 by noticing that $\|g\|_{L^1(\mathbb{R}_+)} = T$ and the truncation parameter $K = \mathcal{O}((\log(T/(\|u(T)\|\,\epsilon)))^{1/\beta})$ for fixed $\beta \in (0,1)$.

Corollary 8. *Consider the linear inhomogeneous differential equation in* (7)*, where the coefficient matrix $A$ has a positive semidefinite matrix real part. Then, for any fixed $\beta \in (0,1)$, Lap-LCHS can prepare an $\epsilon$-approximation of the state $u(T)/\|u(T)\|$ with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left(\frac{1}{\|u(T)\|}\alpha_A T^2 \left(\log\left(\frac{T}{\|u(T)\|\,\epsilon}\right)\right)^{1/\beta}\right) \tag{71}$$

*queries to the block encoding of $A$ with block encoding factor $\alpha_A$, and*

$$\mathcal{O}\left(\frac{T}{\|u(T)\|}\right) \tag{72}$$

*queries to the state preparation oracle for $|\psi\rangle$. Here $\beta \in (0,1)$ is the order parameter in the kernel function of the LCHS formula.*

The best known quantum algorithm prior to LCHS was the truncated Taylor series method proposed in [8]. This method discretizes the ODE in (7) using the truncated Taylor series, formulates the discretized evolution as a dilated linear system of equations, and then solves it with the LCU-based quantum linear system algorithm [12]. Reference [8] gives a detailed complexity analysis using a sparse input oracle for the coefficient matrix. A recent work [6], besides generalizing this idea to the time-dependent case, establishes an improved complexity estimate with block encoding oracle access and an optimal quantum linear system algorithm [16].

A comparison between Lap-LCHS and the truncated Taylor series method is given in Table 1. We see that Lap-LCHS has a better state preparation cost by fully eliminating the explicit dependence on $\alpha_A$ and $\epsilon$. In terms of the matrix query complexity, Lap-LCHS is better in terms of $\epsilon$ as it has a lower degree of the $\log(1/\epsilon)$ term. However, for simulation up to fixed accuracy, by noticing that $\max_t \|u(t)\| \leq T$, Lap-LCHS only has at most comparable (and sometimes worse) matrix query complexity compared to the truncated Taylor series method.

Table 1

*Comparison between Lap-LCHS and the previous approach for solving the ODE* (7)*. Here, we assume the Cartesian decomposition $A = L + iH$ with $L \succeq 0$. $\alpha_A$ is the block encoding factor of $A$, $T$ is the evolution time, and $\epsilon$ is the tolerated error in the output state.*

| **Method** | **Queries to $A$** | **Queries to $\vert\psi\rangle$** |
|---|---|---|
| This work (Corollary 8) | $\widetilde{\mathcal{O}}(\frac{1}{\\|u(T)\\|}\alpha_A T^2(\log(\frac{1}{\epsilon}))^{1/\beta})$ | $\mathcal{O}(\frac{T}{\\|u(T)\\|})$ |
| Taylor [8, 6] | $\widetilde{\mathcal{O}}(\frac{\max_t\\|u(t)\\|}{\\|u(T)\\|}\alpha_A T(\log(\frac{1}{\epsilon}))^2)$ | $\widetilde{\mathcal{O}}(\frac{\max_t\\|u(t)\\|}{\\|u(T)\\|}\alpha_A T\log(\frac{1}{\epsilon}))$ |

**4.2. Power of matrix inverse.** Consider the problem of solving the shifted linear system of equations

$$(\eta I + A)x = |b\rangle. \tag{73}$$

Here $\eta > 0$ is the shifting parameter. The goal of a quantum algorithm for (73) is to approximately prepare the quantum state $(\eta I + A)^{-1}|b\rangle / \left\|(\eta I + A)^{-1}|b\rangle\right\|$. Here, we discuss a more general problem of preparing the state

$$|x\rangle = \frac{(\eta I + A)^{-p}|b\rangle}{\left\|(\eta I + A)^{-p}|b\rangle\right\|}, \tag{74}$$

where $p > 0$ is a real positive parameter. The task of computing powers of the inverse shifted matrix as in (74) appears in many applications, including fractional differential equations [10], ridge regression [23], and inverse iteration for the eigenvalue problem [26].

In the Lap-LCHS framework, we choose the function

$$h(z) = (\eta + z)^{-p}. \tag{75}$$

Its inverse Laplace transform is

$$g(t) = \frac{1}{\Gamma(p)} t^{p-1} e^{-\eta t}, \tag{76}$$

where $\Gamma(p)$ denotes the gamma function. The following result shows the complexity of our algorithm. Its proof can be found in subsection D.1.

Corollary 9. *Let $A$ be a matrix with positive semidefinite real part, and let $\eta > 0$ and $p > 0$. Then for any fixed $\beta \in (0,1)$, Lap-LCHS can prepare an $\epsilon$-approximation of the state $|x\rangle = x/\|x\|$ where $x = (\eta I + A)^{-p}|b\rangle$, with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left(\frac{\alpha_A}{\eta^{p+1}\|x\|}\left(\log\left(\frac{1}{\epsilon\eta\|x\|}\right)\right)^{1+1/\beta}\right) \tag{77}$$

*queries to the block encoding of $A$ with block encoding factor $\alpha_A$ and*

$$\mathcal{O}\left(\frac{1}{\eta^p\|x\|}\right) \tag{78}$$

*queries to the state preparation oracle for $|b\rangle$.*

When $p = 1$, Corollary 9 gives the complexity of our algorithm for solving a linear system of equations with matrix $\eta I + A$. Here we briefly compare our result with existing quantum linear system algorithms. For simplicity, let us assume $\|A\| = 1$ and $\alpha_A = \mathcal{O}(1)$. The first nuance is the different parameter dependence: the complexity of existing quantum linear system algorithms typically depends on the condition number $\kappa$ of the matrix $\eta I + A$, while our algorithm directly depends on $\eta$. Notice that since $\|\eta I + A\| = \mathcal{O}(1+\eta)$ and $\|(\eta I + A)^{-1}\| \leq 1/\eta$, we always have $\kappa = \mathcal{O}(1/\eta)$, so our algorithm scales in terms of a worse parameter.

Nevertheless, it is possible that $\kappa = \Theta(1/\eta)$ (for example, when $A$ is Hermitian and its smallest eigenvalue is $\mathcal{O}(\eta)$). In that case, our algorithm uses $\mathcal{O}(\kappa/\|x\|)$ queries to the state preparation oracle of $|b\rangle$, achieving optimal state preparation cost [32]. The overall query complexity is $\mathcal{O}((\kappa^2/\|x\|)(\log(1/\epsilon))^{1+1/\beta})$, which is almost the same (up to logarithmic factors) as the linear system algorithm based on QSVT [20]. In the worst case, since $\|x\| \geq \Omega(1)$, the overall query complexity of our algorithm becomes $\mathcal{O}(\kappa^2(\log(1/\epsilon))^{1+1/\beta})$. This is worse than the adiabatic-based algorithm [16], which has optimal query complexity $\mathcal{O}(\kappa \log(1/\epsilon))$.

**4.3. Linear differential equations with mass matrices.** Consider the differential equation

$$A\frac{\mathrm{d}u}{\mathrm{d}t} = -u, \tag{79}$$

where $A = L + iH$ for Hermitian matrices $L, H$, and we assume the matrix real part satisfies $L \succeq \gamma > 0$. The goal of a quantum algorithm for (79) is to prepare a quantum state approximating the normalized solution $u(T)/\|u(T)\|$ where $u(T) = e^{-TA^{-1}}u(0)$. We consider two types of initial conditions, namely

$$u(0) = A^{-1}u_0 \tag{80}$$

or

$$u(0) = u_0. \tag{81}$$

In both cases, we are given an oracle for preparing the state $|u_0\rangle = u_0/\|u_0\|$.

**4.3.1. Initial condition with matrix inverse.** First, consider the initial condition in (80). In this case, the solution can be represented as

$$u(T) = e^{-TA^{-1}}A^{-1}u_0, \tag{82}$$

so our goal is to implement the operator $e^{-TA^{-1}}A^{-1}$.

We start with the Laplace transform

$$\frac{1}{z'}e^{-T/z'} = \int_0^\infty e^{-z't'} J_0(2\sqrt{Tt'})\,\mathrm{d}t'. \tag{83}$$

Here, $J_0$ represents the Bessel function of the first kind of order 0. Since $J_0(2\sqrt{Tt'})$ is not in $L^1$, we cannot directly implement this formula based on LCHS. However, since we assume the real part of the matrix is uniformly bounded away from 0 by $\gamma$, we can consider the shifted version of (83) by choosing $z' = z + \gamma$ and obtain

$$\frac{1}{z+\gamma}e^{-T/(z+\gamma)} = \int_0^\infty e^{-zt'}e^{-\gamma t'} J_0(2\sqrt{Tt'})\,\mathrm{d}t'. \tag{84}$$

By replacing $z$ by $A - \gamma I = (L - \gamma I) + iH$ and using Theorem 4, we have

$$e^{-TA^{-1}}A^{-1} = \int_0^\infty \int_{\mathbb{R}} \frac{f(k)}{1-ik} e^{-\gamma t'} J_0(2\sqrt{Tt'}) e^{-it'(k(L-\gamma I)+H)} \,\mathrm{d}k\,\mathrm{d}t' \tag{85}$$

$$= \int_0^\infty \int_{\mathbb{R}} \frac{f(k)g(t';T)e^{it'k\gamma}}{1-ik} e^{-it'(kL+H)} \,\mathrm{d}k\,\mathrm{d}t'. \tag{86}$$

Here

$$g(t';T) = e^{-\gamma t'} J_0(2\sqrt{Tt'}), \tag{87}$$

and its $L^1$ norm is $\mathcal{O}(\gamma^{-1})$ and is independent of $T$.

We have represented the operator $e^{-TA^{-1}}A^{-1}$ as LCHS as in (86). Notice that (86) does not have the standard form specified in Theorem 4 since we have an extra phase factor $e^{it'k\gamma}$ involving both $t'$ and $k$ and the weight function is not separable in $t'$ and $k$. However, we can still implement (86) by the algorithm described in subsection 3.4 with a small modification. Specifically, after we construct the select oracle in (53), namely

$$\mathrm{SEL} = \sum_{j=0}^{M_k-1} \sum_{l=0}^{M_t-1} |j\rangle\langle j| \otimes |l\rangle\langle l| \otimes W_{j,l}, \tag{88}$$

where $W_{j,l}$ block encodes another matrix $V_{j,l} \approx e^{-it'_l(k_jL+H)}$ and $k_j, t'_l$ are the quadrature nodes, we apply an additional controlled phase gate that maps $|j\rangle|l\rangle$ to $e^{it'_lk_j\gamma}|j\rangle|l\rangle$ to append the phase factor. This results in a new select oracle

$$\mathrm{SEL}' = \sum_{j=0}^{M_k-1} \sum_{l=0}^{M_t-1} |j\rangle\langle j| \otimes |l\rangle\langle l| \otimes e^{it'_lk_j\gamma} W_{j,l}, \tag{89}$$

and we can replace (53) by this new select oracle to implement our algorithm. The complexity analysis in Theorem 6 and Corollary 7 still applies with this modification.

The overall complexity is given in the following result. Its proof can be found in subsection D.2.

COROLLARY 10. *Consider the differential equation* (79) *with the initial condition* (80)*, where the matrix real part of $A$ is positive definite and its eigenvalues are bounded from below by $\gamma > 0$. Then for any fixed $\beta \in (0,1)$, we can prepare an $\epsilon$-approximation of the state $u(T)/\|u(T)\|$ with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left( \frac{\|u_0\|}{\|u(T)\|} \frac{\alpha_A}{\gamma^2} \left( \log\left( \frac{\|u_0\|}{\epsilon\|u(T)\|\gamma} \right) \right)^{1+1/\beta} \right) \tag{90}$$

*queries to the block encoding of $A$ with block encoding factor $\alpha_A$, and*

$$\mathcal{O}\left( \frac{\|u_0\|}{\|u(T)\|} \frac{1}{\gamma} \right) \tag{91}$$

*queries to the state preparation oracle of $|u_0\rangle$. Here, $\beta \in (0,1)$ is the order parameter in the kernel function of the LCHS formula.*

**4.3.2. Initial condition without matrix inverse.** Now consider the initial condition in (81). In this case, our goal is to implement the operator $e^{-TA^{-1}}$. Writing

$$e^{-T/z'} - 1 = \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \frac{T^n}{z'^n}, \tag{92}$$

we can (formally) perform the inverse Laplace transform of each term (the inverse Laplace transform of $z'^{-\alpha}$ is $\frac{1}{\Gamma(\alpha)}t'^{\alpha-1}$), giving

$$\sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \frac{T^n t'^{n-1}}{\Gamma(n)} = -\sqrt{\frac{T}{t'}} J_1(2\sqrt{Tt'}), \tag{93}$$

where $J_1$ is the Bessel function of the first kind of order 1. Therefore (see Lemma 15 in Appendix C for a rigorous proof)

$$e^{-T/z'} - 1 = -\int_0^\infty e^{-z't'} \sqrt{\frac{T}{t'}} J_1(2\sqrt{Tt'})\,\mathrm{d}t'. \tag{94}$$

By changing the variable $z' = z + \gamma/2$, we have

$$e^{-T/(z+\gamma/2)} - 1 = -\int_0^\infty e^{-zt'} e^{-\gamma t'/2} \sqrt{\frac{T}{t'}} J_1(2\sqrt{Tt'})\,\mathrm{d}t'. \tag{95}$$

By replacing $z$ by $L - \gamma I/2 + iH$, which still has a positive definite real part, and using Theorem 4, we have

$$e^{-TA^{-1}} - I = -\int_0^\infty \int_{\mathbb{R}} \frac{f(k)}{1-ik} e^{-\gamma t'/2} \sqrt{\frac{T}{t'}} J_1(2\sqrt{Tt'}) e^{-it'(kL - k\gamma I/2 + H)}\,\mathrm{d}k\,\mathrm{d}t' \tag{96}$$

$$= \int_0^\infty \int_{\mathbb{R}} \frac{f(k) g(t';T) e^{it'k\gamma/2}}{1-ik} e^{-it'(kL+H)}\,\mathrm{d}k\,\mathrm{d}t', \tag{97}$$

where

$$g(t';T) = -e^{-\gamma t'/2} \sqrt{\frac{T}{t'}} J_1(2\sqrt{Tt'}). \tag{98}$$

Similarly to the previous case, (97) can also be implemented within our general framework despite an interacting phase factor. After obtaining the block encoding of $e^{-TA^{-1}} - I$, we may implement another LCU to add an identity matrix and construct $e^{-TA^{-1}} - I + I = e^{-TA^{-1}}$.

We give the complexity of this algorithm in the following result. Its proof can be found in subsection D.3.

COROLLARY 11. *Consider the differential equation* (79) *with initial condition* (81) *where the real part of $A$ is positive definite and its eigenvalues are bounded from below by $\gamma > 0$. Then for any fixed $\beta \in (0,1)$, we can prepare an $\epsilon$-approximation of the state $u(T)/\|u(T)\|$ with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left( \frac{\|u_0\|}{\|u(T)\|} \frac{\alpha_A}{\gamma} \left(1 + \sqrt{\frac{T}{\gamma}}\right) \left(\log\left(\frac{T\|u_0\|}{\epsilon\|u(T)\|\gamma}\right)\right)^{1+1/\beta} \right) \tag{99}$$

*queries to the block encoding of $A$ with block encoding factor $\alpha_A$, and*

$$\mathcal{O}\left( \frac{\|u_0\|}{\|u(T)\|} \left(1 + \sqrt{\frac{T}{\gamma}}\right) \right) \tag{100}$$

*queries to the state preparation oracle of $|u_0\rangle$. Here, $\beta \in (0,1)$ is the order parameter in the kernel function of the LCHS formula.*

**4.3.3. Comparison with previous algorithms.** Prior to LCHS, the quantum linear differential equation algorithm with the lowest query complexity was the truncated Dyson series method [6], which can approximate a quantum state $e^{-TB}|v_0\rangle$ for a matrix $B$ with a positive semidefinite real part. As the input model, this algorithm requires a block encoding of $B$ and a state preparation oracle for $|v_0\rangle$. In our setup, $B = A^{-1}$, and $v_0$ can be either $A^{-1}u_0$ or $u_0$. To apply the algorithm in [6], we need the matrix $B = A^{-1}$ to have positive semidefinite real part. This is guaranteed under our assumption that $A + A^\dagger \succeq 0$, because $A^{-1} + (A^{-1})^\dagger = (A^{-1})^\dagger(A + A^\dagger)A^{-1}$ (i.e., $A^{-1} + (A^{-1})^\dagger$ is a congruence transformation of $A + A^\dagger$), and congruence transformation preserves positive semidefiniteness. Then, a straightforward approach is first to construct the block encoding of $A^{-1}$ using QSVT and then construct its exponential using the truncated Dyson series method.

Reference [38, Appendix B] shows that we can construct a block encoding of $A^{-1}$ with block encoding factor $\mathcal{O}(\|A^{-1}\|)$ using $\mathcal{O}(\alpha_A \|A^{-1}\| \log(\|A^{-1}\|/\epsilon))$ queries to the block encoding of $A$. Then, [6, Theorem 1] shows that preparing an $\epsilon$-approximation of $e^{-TA^{-1}}|v_0\rangle/\|e^{-TA^{-1}}|v_0\rangle\|$ takes $\widetilde{\mathcal{O}}(\frac{\|v_0\|}{\|u(T)\|}\|A^{-1}\| T(\log(1/\epsilon))^2)$ queries to the block encoding of $A^{-1}$, so the query complexity to the block encoding of $A$ is

$$\widetilde{\mathcal{O}}\left(\frac{\|v_0\|}{\|u(T)\|}\alpha_A \|A^{-1}\|^2 T\left(\log\left(\frac{1}{\epsilon}\right)\right)^3\right). \tag{101}$$

We can then replace $\|v_0\|$ by $\|u_0\|$ or $\|A^{-1}u_0\| \le \|A^{-1}\|\|u_0\|$ to obtain the matrix query complexity with different initial conditions.

Reference [6, Theorem 1] also shows that the algorithm uses $\widetilde{\mathcal{O}}(\frac{\|v_0\|}{\|u(T)\|}\|A^{-1}\| T \log(1/\epsilon))$ queries to the state preparation oracle for $|v_0\rangle$. This is the final state preparation cost when $v_0 = u_0$. When $v_0 = A^{-1}u_0$, the state $|v_0\rangle$ can be constructed using a quantum linear system solver. The optimal algorithm for this [16] takes $\mathcal{O}(\alpha_A \|A^{-1}\| \log(1/\epsilon))$ queries to the block encoding of $A$ and the state preparation oracle for $|u_0\rangle$. This extra matrix query cost is not dominant compared to (101), but contributes to another multiplicative factor in the state preparation cost, which is $\widetilde{\mathcal{O}}(\frac{\|v_0\|}{\|u(T)\|}\alpha_A \|A^{-1}\|^2 T(\log(1/\epsilon))^2) = \widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\alpha_A \|A^{-1}\|^3 T(\log(1/\epsilon))^2)$.

A comparison is given in Table 2. In both cases, the Lap-LCHS algorithm has better matrix query complexity and state preparation cost.

TABLE 2

*Comparison between Lap-LCHS and the previous approach for solving the ODE* (79). *Here, we assume the Cartesian decomposition $A = L + iH$ with $L \succeq \gamma > 0$. $\alpha_A$ is the block encoding factor of $A$, $T$ is the evolution time (and for technical simplicity, we assume $T \ge \gamma$), and $\epsilon$ is the tolerated error in the output state.*

| With initial condition $u(0) = A^{-1}u_0$ | | |
|---|---|---|
| **Method** | **Queries to $A$** | **Queries to $\|u_0\rangle$** |
| This work (Corollary 10) | $\widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\alpha_A\gamma^{-2}(\log(\frac{1}{\epsilon}))^{1+1/\beta})$ | $\mathcal{O}(\frac{\|u_0\|}{\|u(T)\|}\frac{1}{\gamma})$ |
| QSVT [20] + Dyson [6] | $\widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\alpha_A\|A^{-1}\|^3 T(\log(\frac{1}{\epsilon}))^3)$ | $\widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\alpha_A\|A^{-1}\|^3 T(\log(\frac{1}{\epsilon}))^2)$ |
| With initial condition $u(0) = u_0$ | | |
| **Method** | **Queries to $A$** | **Queries to $\|u_0\rangle$** |
| This work (Corollary 11) | $\widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\alpha_A\gamma^{-3/2}\sqrt{T}(\log(\frac{1}{\epsilon}))^{1+1/\beta})$ | $\mathcal{O}(\frac{\|u_0\|}{\|u(T)\|}\sqrt{\frac{T}{\gamma}})$ |
| QSVT [20] + Dyson [6] | $\widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\alpha_A\|A^{-1}\|^2 T(\log(\frac{1}{\epsilon}))^3)$ | $\widetilde{\mathcal{O}}(\frac{\|u_0\|}{\|u(T)\|}\|A^{-1}\| T\log(\frac{1}{\epsilon}))$ |

**4.3.4. Application to evolutionary partial differential equations with time-space mixed derivatives.** As we describe in this section, linear differential equations with mass matrices as in (79) can describe certain evolutionary partial differential equations with time-space mixed derivatives so that the Lap-LCHS algorithm can be applied to such problems. We start with the example

$$\frac{\partial u(t,x)}{\partial t} = \frac{\partial^2 u(t,x)}{\partial t\,\partial x} - u(t,x), \quad t\in[0,T],\, x\in[0,1], \tag{102}$$

$$u(0,x) = u_0(x), \tag{103}$$

$$u(t,0) = u(t,1). \tag{104}$$

A standard technique for numerically solving partial differential equations is the method of lines, in which we first discretize all but the time variable to obtain an ODE system and then apply a numerical ODE solver. We apply the central difference formula $\partial_x v(t,x) \approx \frac{1}{2h}(v(t,x+h)-v(t,x-h))$ to discretize the spatial variable $x$ with step size $h$, giving the semidiscretized system

$$\frac{\partial u(t,x)}{\partial t} \approx \frac{1}{2h}\left(\frac{\partial u(t,x+h)}{\partial t} - \frac{\partial u(t,x-h)}{\partial t}\right) - u(t,x). \tag{105}$$

Let $[0,h,2h,\ldots,(N-1)h]$ be the grid points for discretizing $x$, where $N$ is the number of grid points and $h = 1/N$, and $\mathbf{u}(t) = [u(t,0); u(t,h); u(t,2h); \ldots; u(t,(N-1)h)]$. Then from (105) we have

$$\frac{d\mathbf{u}}{dt} \approx D\frac{d\mathbf{u}}{dt} - \mathbf{u}, \tag{106}$$

where

$$D = \frac{N}{2}\begin{pmatrix} 0 & 1 & & & & -1 \\ -1 & 0 & 1 & & & \\ & -1 & 0 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 0 & 1 \\ 1 & & & & -1 & 0 \end{pmatrix}. \tag{107}$$

Notice that all the eigenvalues of $D$ are imaginary, so (106) is exactly in the form of (79) with $A = L + iH$, $L = I \succ 0$, and $H = iD$.

More generally, we can consider the equation

$$\partial_t \mathcal{L}_x u(t,x) = -u(t,x), \tag{108}$$

where $\mathcal{L}_x = \sum_{k=0}^m a_k \partial_x^k$ is a differential operator with respect to the $x$ variable. Then, the spatially discretized system is also in the form of (79), where $A$ is the discrete version of $\mathcal{L}_x$. The stability condition $L \succ 0$ can be satisfied by imposing conditions on the coefficients $a_{2k}$ of the even-order derivatives. It is also possible to generalize this approach to the case where $x$ has a higher dimension.

**4.4. Second-order differential equations.** Consider the second-order differential equation

$$\frac{\mathrm{d}^2 u}{\mathrm{d}t^2} = Au, \quad u(0) = |u_0\rangle, \quad u'(0) = v_0, \tag{109}$$

where $A$ has a positive semidefinite real part. The solution of (109) has two branches, $e^{\pm\sqrt{A}t}$. Here we only consider a special scenario where the choice of $v_0$ forces the dynamics to have only a decaying branch, i.e., the solution is $u(T) = e^{-\sqrt{A}T}|u_0\rangle$.

In the Lap-LCHS framework, we have

$$h(z) = e^{-T\sqrt{z}} = \int_0^\infty e^{-zt'} \frac{T}{2\sqrt{\pi t'^3}} e^{-T^2/(4t')} \,\mathrm{d}t'. \tag{110}$$

Then the inverse Laplace transform of $h(z)$ is

$$g(t') = \frac{T}{2\sqrt{\pi t'^3}} e^{-T^2/(4t')}. \tag{111}$$

We can directly apply the Lap-LCHS algorithm to prepare $h(A)|u_0\rangle$. Its complexity can be estimated using Corollary 7, which is proven in subsection D.4.

Corollary 12. *Consider the second-order differential equation in* (109), *where the coefficient matrix $A$ has a positive semidefinite real part. Then for any fixed $\beta \in (0,1)$, Lap-LCHS can prepare an $\epsilon$-approximation of the decaying branch $|u(T)\rangle = u(T)/\|u(T)\|$, where $u(T) = e^{-\sqrt{A}T}|u_0\rangle$, with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left(\frac{\alpha_A T^2}{\|u(T)\|^3 \epsilon^2}\left(\log\left(\frac{1}{\|u(T)\|\,\epsilon}\right)\right)^{1/\beta}\right) \tag{112}$$

*queries to the block encoding of $A$ with block encoding factor $\alpha_A$, and*

$$\mathcal{O}\left(\frac{1}{\|u(T)\|}\right) \tag{113}$$

*queries to the state preparation oracle for $|u_0\rangle$.*

Corollary 12 shows that the matrix query complexity of Lap-LCHS depends poorly on several parameters, including $T$, $\epsilon$, and $\|u(T)\|$. This is mainly due to the slow decay of $g(t)$ in (111), which requires a large truncation parameter $T'$. It is possible to improve this scaling in a more restricted case where the real part of $A$ is positive definite, and its eigenvalues are lower bounded by $\gamma$, using the same shifting trick as in subsection 4.3. Specifically, by substituting the variable $z \to z+\gamma$ in (110), we have

$$e^{-T\sqrt{z+\gamma}} = \int_0^\infty e^{-zt'} e^{-\gamma t'} \frac{T}{2\sqrt{\pi t'^3}} e^{-T^2/4t'} \,\mathrm{d}t'. \tag{114}$$

Replacing $z$ by $A - \gamma I = (L-\gamma I) + iH$, we have

$$e^{-T\sqrt{A}} = \int_0^\infty \int_{\mathbb{R}} \frac{f(k) g(t';T) e^{it'k\gamma}}{1-ik} e^{-it'(kL+H)} \,\mathrm{d}k\,\mathrm{d}t'. \tag{115}$$

Now

$$g(t';T) = e^{-\gamma t'} \frac{T}{2\sqrt{\pi t'^3}} e^{-T^2/4t'}, \tag{116}$$

which has an exponential decay as $t' \to \infty$, so the truncation parameter can be significantly reduced to $\mathcal{O}((1/\gamma)\log(1/\epsilon'))$ in order to bound $\|g\|_{L^1((T',\infty))}$ by $\mathcal{O}(\epsilon')$. The complexity of Lap-LCHS in this special scenario is captured by the following result, which can be directly obtained from the proof of Corollary 12 and the new choice of $T'$.

COROLLARY 13. *Consider the second-order differential equation in* (109), *where the coefficient matrix $A$ has a positive definite matrix real part $L \succeq \gamma > 0$. Then for any fixed $\beta \in (0,1)$, Lap-LCHS can prepare an $\epsilon$-approximation of the decaying branch $|u(T)\rangle = u(T)/\|u(T)\|$, where $u(T) = e^{-\sqrt{A}T}|u_0\rangle$, with $\Omega(1)$ success probability and a flag indicating success, using*

$$\mathcal{O}\left(\frac{1}{\|u(T)\|}\frac{\alpha_A}{\gamma}\left(\log\left(\frac{1}{\|u(T)\|\,\epsilon}\right)\right)^{1+1/\beta}\right) \tag{117}$$

*queries to the block encoding of $A$ with block encoding factor $\alpha_A$, and*

$$\mathcal{O}\left(\frac{1}{\|u(T)\|}\right) \tag{118}$$

*queries to the state preparation oracle for $|u_0\rangle$.*

Although there is no explicit dependence on the evolution time $T$ in Corollary 13, the actual asymptotic scaling in $T$ is $\widetilde{\mathcal{O}}(e^{\gamma T})$. This is because we assume positive definiteness of the real part $L$ of the matrix $A$, so the solution norm $\|u(T)\|$ always decays exponentially in $T$. Nevertheless, Corollary 13 can still describe an effective method for solving (109) within a short or intermediate time period (e.g., when $T = \mathcal{O}(1/\gamma)$).

We present two extensions of this result below. Here we only discuss the design of the algorithms and leave detailed complexity analysis for future work.

**4.4.1. Wave equations.** A notable example of the second-order differential equation in (109) is the wave equation, where the coefficient matrix $A$ is the discretized Laplacian. In this case, the matrix $A$ is Hermitian negative semidefinite and does not satisfy the condition of our aforementioned results, so we cannot directly apply Corollary 12 or Corollary 13.

However, the range of applicability of (110) is in fact beyond $\mathrm{Re}\, z \geq 0$. By change of variable $T \to Te^{-i\theta/2}$ for any $0 \leq \theta < \pi/2$, (110) still holds as

$$h(z) = e^{-T\sqrt{e^{-i\theta}z}} = \int_0^\infty e^{-zt'} \frac{Te^{-i\theta/2}}{2\sqrt{\pi t'^3}} e^{-T^2 e^{-i\theta}/(4t')}\,\mathrm{d}t', \tag{119}$$

which still converges for all $\mathrm{Re}\, z \geq 0$. Therefore we can solve the decaying branch of

$$\frac{\mathrm{d}^2 u}{\mathrm{d}t^2} = e^{-i\theta} Bu, \quad u(0) = |u_0\rangle, \tag{120}$$

where $B$ has a positive semidefinite real part. We further write $B = iA$, where $A$ has a negative semidefinite *imaginary* part, so we can solve the decaying branch of

$$\frac{\mathrm{d}^2 u}{\mathrm{d}t^2} = ie^{-i\theta} Au, \quad u(0) = |u_0\rangle \tag{121}$$

using the formula

$$|u(T)\rangle = e^{-T\sqrt{ie^{-i\theta}A}}|u_0\rangle = \int_0^\infty \frac{Te^{-i\theta/2}}{2\sqrt{\pi t'^3}} e^{-T^2e^{-i\theta}/(4t')} e^{-iAt'}|u_0\rangle\,\mathrm{d}t'. \tag{122}$$

In particular, when the imaginary part of $A$ is 0 (i.e., $A$ is a Hermitian matrix), the right-hand side only involves Hamiltonian simulation of $A$. Taking the limit $\theta \to \pi/2$, we can get arbitrarily close to

$$\frac{\mathrm{d}^2 u}{\mathrm{d}t^2} = Au, \quad u(0) = |u_0\rangle. \tag{123}$$

When $A \preceq 0$, this corresponds to the scenario of the wave equation.

We also remark that existing algorithms, such as those in [17], achieve efficient quantum algorithms for the wave equation by leveraging a *known* factorization of the matrix $B$, which allows the wave equation to be reformulated directly as a Hamiltonian simulation problem. To the best of our knowledge, there are no efficient quantum algorithms for wave equations in the absence of such a factorization. Moreover, the matrix Laplace transform technique critically depends on the decay properties of the solution, and therefore, (121) represents the most general form that can be addressed using our current Lap-LCHS method.

**4.4.2. Second-order differential equations with mass matrices.** Analogous to the first-order case, we may also consider a second-order differential equation with a nondiagonal mass matrix

$$A\frac{\mathrm{d}^2u}{\mathrm{d}t^2} = u, \quad u(0) = |u_0\rangle, \quad u'(0) = v_0. \tag{124}$$

We focus on the decaying branch of the solution $u(T) = e^{-\sqrt{A^{-1}}T}|u_0\rangle$. Here, we briefly discuss the simulation strategy and omit the detailed complexity analysis for simplicity. Using the fact that

$$e^{-T/\sqrt{z}} - 1 = \sum_{n=1}^{\infty} \frac{(-1)^n}{n!}\frac{T^n}{z^{n/2}}, \tag{125}$$

we can perform the inverse Laplace transform of each term (in particular, the inverse Laplace transform of $z^{-\alpha}$ is $\frac{1}{\Gamma(\alpha)}t'^{\alpha-1}$) and obtain (see Lemma 16 in Appendix C)

$$e^{-T/\sqrt{z}} - 1 = \int_0^\infty e^{-zt'} g(t')\,\mathrm{d}t', \tag{126}$$

where

$$g(t') = \sum_{n=1}^{\infty} \frac{(-1)^n T^n}{n!}\frac{t'^{n/2-1}}{\Gamma(n/2)} = \frac{1}{2}T^2 \times {}_0F_2\left(;\frac{3}{2},2;\frac{T^2t'}{4}\right) - \frac{T}{\sqrt{\pi t'}} \times {}_0F_2\left(;\frac{1}{2},\frac{3}{2};\frac{T^2t'}{4}\right), \tag{127}$$

with ${}_pF_q(a_1,\ldots,a_p;b_1,\ldots,b_q;z)$ denoting the generalized hypergeometric function.

Suppose that the real part of $A$ is positive definite and its eigenvalues are lower bounded by $\gamma > 0$. Using the shifting trick $z \to z + \gamma$ again, we have

$$e^{-T/\sqrt{z+\gamma}} - 1 = \int_0^\infty e^{-zt'} e^{-\gamma t'} g(t')\,\mathrm{d}t'. \tag{128}$$

Then, by replacing $z$ by $A - \gamma I$, we have

$$e^{-T/\sqrt{A}} = \int_0^\infty \int_{\mathbb{R}} \frac{f(k) g(t';T) e^{it'k\gamma}}{1-ik} e^{-it'(kL+H)}\,\mathrm{d}k\,\mathrm{d}t' \tag{129}$$

with

$$g(t';T) = e^{-\gamma t'}\left(\sum_{n=1}^{\infty}\frac{(-1)^n T^n}{n!}\frac{t'^{n/2-1}}{\Gamma(n/2)}\right). \tag{130}$$

So $e^{-T/\sqrt{A}}$ can be implemented by the Lap-LCHS algorithm.

**Appendix A. Bounding the integral discretization error.**

*Proof of Lemma* 5. We focus on bounding two types of errors. Choices of the parameters can be directly derived from the error bounds. For the truncation error, we have

$$\left\|\int_0^\infty \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t - \int_0^T \int_{-K}^K \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t\right\| \tag{131}$$

$$\leq \left\|\int_0^\infty \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t - \int_0^T \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t\right\| \tag{132}$$

$$+ \left\|\int_0^T \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t - \int_0^T \int_{-K}^K \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t\right\| \tag{133}$$

$$= \left\|\int_T^\infty \int_{\mathbb{R}} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t\right\| + \left\|\int_0^T \int_{\mathbb{R}\setminus[-K,K]} \frac{f(k)g(t)}{1-ik} U(k,t)\,\mathrm{d}k\,\mathrm{d}t\right\| \tag{134}$$

$$\leq \left\|\frac{f}{1-ik}\right\|_{L^1(\mathbb{R})} \|g\|_{L^1((T,\infty))} + \left\|\frac{f}{1-ik}\right\|_{L^1(\mathbb{R}\setminus[-K,K])} \|g\|_{L^1(\mathbb{R}_+)} \tag{135}$$

$$\leq \|f\|_{L^1(\mathbb{R})} \|g\|_{L^1((T,\infty))} + \|f\|_{L^1(\mathbb{R}\setminus[-K,K])} \|g\|_{L^1(\mathbb{R}_+)}. \tag{136}$$

For the quadrature error, let $V(k,j) = \frac{f(k)g(t)}{1-ik} U(k,t)$. We first write

$$\int_0^T \int_{-K}^K V(k,t)\,\mathrm{d}k\,\mathrm{d}t - h_k h_t \sum_{j=0}^{2K/h_k-1} \sum_{l=0}^{T/h_t-1} V(k_j,t_l) \tag{137}$$

$$= \sum_{j=0}^{2K/h_k-1} \sum_{l=0}^{T/h_t-1} \left( \int_{t_l}^{t_l+h_t} \int_{k_j}^{k_j+h_j} V(k,t)\,\mathrm{d}k\,\mathrm{d}t - h_k h_t V(k_j,t_l) \right). \tag{138}$$

On each interval $[k_j, k_j+h_j] \times [t_l, t_l+h_t]$, we have

$$\|V(k,t) - V(k_j,t_l)\| \leq \sup \left\|\frac{\partial V}{\partial k}\right\| h_k + \sup \left\|\frac{\partial V}{\partial t}\right\| h_t, \tag{139}$$

so

$$\left\|\int_0^T \int_{-K}^K V(k,t)\,\mathrm{d}k\,\mathrm{d}t - h_k h_t \sum_{j=0}^{2K/h_k-1} \sum_{l=0}^{T/h_t-1} V(k_j,t_l)\right\| \tag{140}$$

$$\leq \sum_{j=0}^{2K/h_k-1} \sum_{l=0}^{T/h_t-1} h_k h_t \left( \sup \left\|\frac{\partial V}{\partial k}\right\| h_k + \sup \left\|\frac{\partial V}{\partial t}\right\| h_t \right) \tag{141}$$

$$= 2KT \left( \sup \left\|\frac{\partial V}{\partial k}\right\| h_k + \sup \left\|\frac{\partial V}{\partial t}\right\| h_t \right). \tag{142}$$

Straightforward computations give

$$\frac{\partial V}{\partial k} = \frac{f(k)g(t)}{1-ik}(-itL)e^{-it(kL+H)} + \frac{f'(k)(1-ik) + if(k)}{(1-ik)^2} g(t) e^{-it(kL+H)}, \tag{143}$$

$$\frac{\partial V}{\partial t} = \frac{f(k)g(t)}{1-ik}(-i)(kL+H)e^{-it(kL+H)} + \frac{f(k)g'(t)}{1-ik} e^{-it(kL+H)}, \tag{144}$$

and thus

$$\sup \left\| \frac{\partial V}{\partial k} \right\| \leq \|f\|_{L^\infty(\mathbb{R})} \|tg\|_{L^\infty(\mathbb{R}_+)} \|A\| + \left( \|f'\|_{L^\infty(\mathbb{R})} + \|f\|_{L^\infty(\mathbb{R})} \right) \|g\|_{L^\infty(\mathbb{R}_+)}, \tag{145}$$

$$\sup \left\| \frac{\partial V}{\partial t} \right\| \leq 2 \|f\|_{L^\infty(\mathbb{R})} \|g\|_{L^\infty(\mathbb{R}_+)} \|A\| + \|f\|_{L^\infty(\mathbb{R})} \|g'\|_{L^\infty(\mathbb{R}_+)}. \tag{146}$$

∎

**Appendix B. Construction of the preparation oracles.** Here we discuss the construction of the unitaries $O_{c,l}$, $O_{c,r}$, $O_{\hat{c},l}$, and $O_{\hat{c},r}$, which prepare quantum states encoding the quadrature coefficients in their amplitudes. Specifically, these unitaries prepare a quantum state of the form

$$\frac{1}{\sqrt{\sum_{j=0}^{M-1} |F(a+jh)|^2}} \sum_{j=0}^{M-1} F(a+jh)|j\rangle. \tag{147}$$

Here $F(k)$ is a known complex-valued function defined on a real interval $[a,b]$, $M$ is a positive integer, and $h = (b-a)/M$ is the step size.

In general, an $M$-dimensional quantum state can be prepared with cost $\mathcal{O}(M)$ [34]. However, the amplitudes of the quantum state in (147) are given by known functions evaluated at discrete points, and these functions in the Lap-LCHS algorithm have closed-form expressions and are integrable (e.g., in $O_{c,r}$ we have $F(k) = \frac{f(k)}{1-ik}$ where $f(k)$ is the kernel function in LCHS and can be chosen as in (21)). Then, the state preparation circuits can be constructed more efficiently, with cost only $\mathcal{O}(\operatorname{poly}\log M)$, by the Grover–Rudolph approach [21]. More specifically, we first apply Grover–Rudolph to prepare a state proportional to $\sum_{j=0}^{M-1} |F(a+jh)||j\rangle$ encoding the absolute values of $F(k)$ in its amplitudes, and then apply the unitary $\widetilde{U} : |j\rangle \to e^{i\arg(F(a+jh))}|j\rangle$ to introduce the correct phases. Notice that $\widetilde{U}$ can be efficiently constructed with the help of classical arithmetic, and the overall complexity of preparing the desired state is $\mathcal{O}(\operatorname{poly}\log M)$.

The implementation of the Grover–Rudolph approach requires coherent arithmetic, resulting in complicated quantum circuits due to its high qubit and gate costs in practice. A recent work [33] provides an alternative state preparation algorithm based on QSVT and avoids coherent arithmetic operations, provided the function $F(k)$ can be approximated by a degree-$d$ polynomial or Fourier series (which is also the case in Lap-LCHS). As shown in [33, Theorem 1], this approach only needs 4 ancilla qubits and $\mathcal{O}(\log(M)d/\mathcal{F})$ gates, where $\mathcal{F} = \sqrt{\frac{\sum_{j=0}^{M-1}|F(a+jh)|^2}{M\max_j |F(a+jh)|^2}}$. Notice that if the function $F(k)$ performs badly in the sense that it concentrates only on a few points, e.g., $F(a+jh)$ is nonzero only at $j=0$, then $\mathcal{F} \sim \sqrt{1/M}$ and such a state preparation approach still has cost $\widetilde{\mathcal{O}}(\sqrt{M})$. However, if the function $F(k)$ is continuously differentiable and in $L^1$ (which is satisfied for $O_{c,l}$ and $O_{c,r}$ and also holds for $O_{\hat{c},l}$ and $O_{\hat{c},r}$ for continuously differentiable inverse Laplace transforms), then $\mathcal{F} \approx \frac{1}{\sqrt{b-a}} \frac{\|F\|_{L^2(a,b)}}{\|F\|_{L^\infty(a,b)}}$ does not depend on the number $M$, so the gate complexity of the state preparation is $\mathcal{O}(\log(M))$.

**Appendix C. Laplace transform of series of functions.** In section 4, we compute the inverse Laplace transforms of some functions by formally taking the inverse Laplace transform of each term in its series expansion. Here, we rigorously validate such a procedure by analyzing the term-by-term computation of the Laplace transform. This also validates the same procedure for the inverse Laplace transform by definition of the inverse Laplace transform.

Let $\mathbb{C}_> = \{z \in \mathbb{C} \mid \operatorname{Re} z > 0\}$, $\overline{\mathbb{C}_>} = \{z \in \mathbb{C} \mid \operatorname{Re} z \geq 0\}$ denote the right half plane and its closure, respectively. We first state a general convergence result, which is a direct consequence of the dominated convergence theorem.

LEMMA 14. *Let $g(t) = \sum_{j=1}^{\infty} a_j q_j(t)$ be a pointwise absolutely convergent function series on $(0,\infty)$. Suppose that $\lim_{J\to\infty} \sum_{j=1}^{J} \int_0^\infty |a_j q_j(t) e^{-zt}|\, \mathrm{d}t < \infty$ for $z \in \mathcal{D} \subset \overline{\mathbb{C}_>}$. Then the Laplace transform of $g(t)$ exists on $\mathcal{D}$ and is given as*

$$h(z) = \sum_{j=1}^{\infty} a_j \int_0^\infty q_j(t) e^{-zt}\, \mathrm{d}t. \tag{148}$$

*Proof.* We fix the complex number $z$ in the domain of interest and drop the explicit dependence on it. Let $g_J(t) = \sum_{j=1}^{J} a_j q_j(t)$ and $G(t) = \sum_{j=1}^{\infty} |a_j q_j(t) e^{-zt}|$, which is well-defined due to the absolute convergence. Then $|g_J(t) e^{-zt}| \leq G(t)$, and by Beppo Levi's lemma we have

$$\int_0^\infty G(t)\, \mathrm{d}t = \lim_{J\to\infty} \sum_{j=1}^{J} \int_0^\infty |a_j q_j(t) e^{-zt}|\, \mathrm{d}t < \infty. \tag{149}$$

Therefore, by the dominated convergence theorem, we have that $g(t)e^{-zt}$ is integrable and

$$\begin{aligned} \int_0^\infty g(t) e^{-zt}\, \mathrm{d}t &= \int_0^\infty \lim_{J\to\infty} g_J(t) e^{-zt}\, \mathrm{d}t = \lim_{J\to\infty} \int_0^\infty g_J(t) e^{-zt}\, \mathrm{d}t \\ &= \sum_{j=1}^{\infty} a_j \int_0^\infty q_j(t) e^{-zt}\, \mathrm{d}t, \end{aligned} \tag{150}$$

where the last step is due to integrability of each $a_j q_j(t) e^{-zt}$ and linearity of the integral. □

Now, we can give a rigorous proof of the examples of the Laplace transform series we use in section 4.

LEMMA 15. *The Laplace transform of $g(t) = \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \frac{T^n t^{n-1}}{\Gamma(n)}$ on $\mathbb{C}_>$ is $h(z) = e^{-T/z} - 1$.*

*Proof.* Notice that the Laplace transform of $t^{n-1}$ is $\Gamma(n) z^{-n}$. We have

$$\lim_{N\to\infty} \sum_{n=1}^{N} \int_0^\infty \left| \frac{(-1)^n}{n!} \frac{T^n t^{n-1}}{\Gamma(n)} e^{-zt} \right| \mathrm{d}t = \lim_{N\to\infty} \sum_{n=1}^{N} \frac{1}{n!} \frac{T^n}{(\operatorname{Re}(z))^n} = e^{T/\operatorname{Re}(z)} - 1 < \infty. \tag{151}$$

Then, by Lemma 14, we can compute

$$\int_0^\infty g(t) e^{-zt}\, \mathrm{d}t = \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \frac{T^n}{\Gamma(n)} \int_0^\infty t^{n-1} e^{-zt}\, \mathrm{d}t = \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \frac{T^n}{z^n} = h(z) \tag{152}$$

as claimed. □

LEMMA 16. *The Laplace transform of* $g(t)=\sum_{n=1}^\infty \frac{(-1)^nT^n}{n!}\frac{t^{n/2-1}}{\Gamma(n/2)}$ *on* $\mathbb{C}_>$ *is* $h(z)=e^{-T/\sqrt{z}}-1$.

*Proof.* Notice that the Laplace transform of $t^{\alpha-1}$ is $\Gamma(\alpha)z^{-\alpha}$ for $\alpha>0$. We have

$$\begin{aligned}\lim_{N\to\infty}\sum_{n=1}^N\int_0^\infty\left|\frac{(-1)^nT^n}{n!}\frac{t^{n/2-1}}{\Gamma(n/2)}e^{-zt}\right|\mathrm{d}t \\ =\lim_{N\to\infty}\sum_{n=1}^N\frac{1}{n!}\frac{T^n}{(\mathrm{Re}(z))^{n/2}}=e^{T/\sqrt{\mathrm{Re}(z)}}-1<\infty.\end{aligned} \tag{153}$$

Then, by Lemma 14, we can compute

$$\begin{aligned}\int_0^\infty g(t)e^{-zt}\,\mathrm{d}t&=\sum_{n=1}^\infty\frac{(-1)^nT^n}{n!}\frac{1}{\Gamma(n/2)}\int_0^\infty t^{n/2-1}e^{-zt}\,\mathrm{d}t\\&=\sum_{n=1}^\infty\frac{(-1)^nT^n}{n!}\frac{1}{z^{n/2}}=h(z)\end{aligned} \tag{154}$$

as claimed. ∎

**Appendix D. Complexity estimates for applications of Lap-LCHS.** Here, we present the proofs of the complexity estimates for the various applications discussed in section 4. The ideas of the proofs are similar. In each specific application, we compute the $L^1$ norm of the functions $f(k)$ and $g(t)$ and estimate the integral truncation parameters $K$ and $T$ to satisfy the assumption in Theorem 6. Then, the claimed results directly follow from Theorem 6 or Corollary 7.

**D.1. Proof of Corollary 9.**

*Proof of Corollary* 9. In this application, we have $\|f\|_{L^1(\mathbb{R})}=\mathcal{O}(1)$ as usual. The $L^1$ norm of $g(t)$ is

$$\|g\|_{L^1(\mathbb{R}_+)}=\frac{1}{\Gamma(p)}\int_0^\infty t^{p-1}e^{-\eta t}\,\mathrm{d}t=\eta^{-p}. \tag{155}$$

To bound $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])}$ by $\mathcal{O}(\epsilon\|x\|/\|g\|_{L^1(\mathbb{R}_+)})=\mathcal{O}(\epsilon\|x\|\eta^p)$, we can choose $K=\mathcal{O}((\log(1/(\epsilon\eta\|x\|)))^{1/\beta})$. We still need to choose $T$ such that $\|g\|_{L^1((T,\infty))}=\mathcal{O}(\epsilon\|x\|/\|f\|_{L^1(\mathbb{R})})=\mathcal{O}(\epsilon\|x\|)$. Notice that

$$\begin{aligned}\|g\|_{L^1((T,\infty))}&=\frac{1}{\Gamma(p)}\int_T^\infty t^{p-1}e^{-\eta t}\,\mathrm{d}t && (156)\\&\le e^{-\eta T/2}\left(\frac{1}{\Gamma(p)}\int_T^\infty t^{p-1}e^{-\eta t/2}\,\mathrm{d}t\right) && (157)\\&\le e^{-\eta T/2}\left(\frac{1}{\Gamma(p)}\int_0^\infty t^{p-1}e^{-\eta t/2}\,\mathrm{d}t\right) && (158)\\&=e^{-\eta T/2}(\eta/2)^{-p}. && (159)\end{aligned}$$

Then, it suffices to choose

$$T=\mathcal{O}\left(\frac{1}{\eta}\log\left(\frac{1}{\epsilon\eta\|x\|}\right)\right). \tag{160}$$

Plugging these estimates back into Corollary 7 gives the claimed query complexity. ∎

**D.2. Proof of Corollary 10.**

*Proof of Corollary* 10. We first discuss the complexity of implementing a block encoding of $e^{-TA^{-1}}A^{-1}$. In its LCHS representation, we have

$$f(k) = \frac{1}{2\pi e^{-2^\beta} e^{(1+ik)^\beta}} \tag{161}$$

and

$$g(t';T) = e^{-\gamma t'} J_0(2\sqrt{Tt'}). \tag{162}$$

Notice that

$$|g(t';T)| \le e^{-\gamma t'}. \tag{163}$$

We can estimate the $L^1$ norms of the functions as

$$\|f\|_{L^1} = \mathcal{O}(1), \quad \|g\|_{L^1} = \mathcal{O}(1/\gamma) \tag{164}$$

and choose

$$T' = \mathcal{O}\left(\frac{1}{\gamma}\log\left(\frac{1}{\gamma\epsilon'}\right)\right), \quad K = \mathcal{O}\left(\left(\log\left(\frac{1}{\gamma\epsilon'}\right)\right)^{1/\beta}\right) \tag{165}$$

such that $\|g\|_{L^1([T',\infty])} = \mathcal{O}(\epsilon'/\|f\|_{L^1(\mathbb{R})})$ and $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])} = \mathcal{O}(\epsilon'/\|g\|_{L^1(0,\infty)})$. According to Theorem 6, we can construct an $(\alpha' = \mathcal{O}(1/\gamma), \epsilon')$-block encoding of $e^{-TA^{-1}}A^{-1}$ using

$$\mathcal{O}\left(\frac{\alpha_A}{\gamma}\left(\log\left(\frac{1}{\gamma\epsilon'}\right)\right)^{1+1/\beta} + \log\left(\frac{1}{\gamma\epsilon'}\right)\right) = \mathcal{O}\left(\frac{\alpha_A}{\gamma}\left(\log\left(\frac{1}{\gamma\epsilon'}\right)\right)^{1+1/\beta}\right) \tag{166}$$

queries to the block encoding of $A$.

Applying this block encoding to the input state $|0\rangle|u_0\rangle$ gives

$$\frac{1}{\alpha'}|0\rangle B|u_0\rangle + |\perp\rangle, \tag{167}$$

where $B$ is a matrix such that $\|B - e^{-TA^{-1}}A^{-1}\| \le \epsilon'$. Measuring the ancilla register onto 0 gives the state $B|u_0\rangle/\|B|u_0\rangle\|$. This is an approximation of the desired state

$$\frac{e^{-TA^{-1}}A^{-1}|u_0\rangle}{\left\|e^{-TA^{-1}}A^{-1}|u_0\rangle\right\|} \tag{168}$$

up to error

$$\begin{aligned}\left\|\frac{B|u_0\rangle}{\|B|u_0\rangle\|} - \frac{e^{-TA^{-1}}A^{-1}|u_0\rangle}{\left\|e^{-TA^{-1}}A^{-1}|u_0\rangle\right\|}\right\| &\le \frac{2}{\left\|e^{-TA^{-1}}A^{-1}|u_0\rangle\right\|}\left\|B - e^{-TA^{-1}}A^{-1}\right\| \\ &\le \frac{2\epsilon'\|u_0\|}{\|u(T)\|}.\end{aligned} \tag{169}$$

To bound the final error by $\epsilon$ and the success probability by $\Omega(1)$, it suffices to choose

$$\epsilon' = \mathcal{O}\left(\frac{\|u(T)\|}{\|u_0\|}\epsilon\right) \tag{170}$$

and implement $\mathcal{O}(\alpha'/\|B|u_0\rangle\|) = \mathcal{O}(\frac{1}{\gamma}\frac{\|u_0\|}{\|u(T)\|})$ rounds of amplitude amplification. Plugging these choices back into (166) gives the overall query complexity. ∎

**D.3. Proof of Corollary 11.**

*Proof of Corollary* 11. We start by estimating the complexity of block encoding $e^{-TA^{-1}} - I$ based on (97) by estimating the $L^1$ norm and truncation parameter of the functions

$$f(k) = \frac{1}{2\pi e^{-2^\beta} e^{(1+ik)^\beta}} \tag{171}$$

and

$$g(t';T) = -e^{-\gamma t'/2}\sqrt{\frac{T}{t'}}J_1(2\sqrt{Tt'}). \tag{172}$$

We have $\|f\|_{L^1} = \mathcal{O}(1)$ as usual. By bounding the Bessel function by 1 and substituting the variable $s = \sqrt{Tt'}$, we have

$$\|g\|_{L^1} \le \int_0^\infty e^{-\gamma t'/2}\sqrt{\frac{T}{t'}}\,\mathrm{d}t' \tag{173}$$

$$= 2\int_0^\infty e^{-\frac{s^2}{2T/\gamma}}\,\mathrm{d}s \tag{174}$$

$$= \mathcal{O}\left(\sqrt{\frac{T}{\gamma}}\right). \tag{175}$$

We choose $K$ and $T'$ so that $\|g\|_{L^1([T',\infty])} = \mathcal{O}(\epsilon'/\|f\|_{L^1(\mathbb{R})})$ and $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])} = \mathcal{O}(\epsilon'/\|g\|_{L^1(0,\infty)})$. Using similar techniques as in estimating $\|g\|_{L^1}$, we obtain

$$\|g\|_{L^1([T',\infty])} \le \int_{T'}^\infty e^{-\gamma t'/2}\sqrt{\frac{T}{t'}}\,\mathrm{d}t' \tag{176}$$

$$= 2\int_{\sqrt{TT'}}^\infty e^{-\frac{s^2}{2T/\gamma}}\,\mathrm{d}s \tag{177}$$

$$\le 2e^{-\frac{TT'}{4T/\gamma}}\int_0^\infty e^{-\frac{s^2}{4T/\gamma}}\,\mathrm{d}s \tag{178}$$

$$= \mathcal{O}\left(e^{-\gamma T'/4}\sqrt{\frac{T}{\gamma}}\right). \tag{179}$$

Therefore, it suffices to choose

$$T' = \mathcal{O}\left(\frac{1}{\gamma}\log\left(\frac{T}{\epsilon'\gamma}\right)\right), \quad K = \mathcal{O}\left(\left(\log\left(\frac{T}{\epsilon'\gamma}\right)\right)^{1/\beta}\right). \tag{180}$$

According to Theorem 6, we can construct an $(\mathcal{O}(\sqrt{T/\gamma}), \epsilon')$-block encoding of $e^{-TA^{-1}} - I$ using

$$\mathcal{O}\left(\frac{\alpha_A}{\gamma}\left(\log\left(\frac{T}{\epsilon'\gamma}\right)\right)^{1+1/\beta} + \log\left(\frac{T}{\epsilon'\gamma}\right)\right) = \mathcal{O}\left(\frac{\alpha_A}{\gamma}\left(\log\left(\frac{T}{\epsilon'\gamma}\right)\right)^{1+1/\beta}\right) \tag{181}$$

queries to the block encoding of $A$.

Now the block encoding of $e^{-TA^{-1}}$ can be directly constructed by using LCU for $e^{-TA^{-1}} - I$ and $I$. According to [20, Lemma 52], this is an $(\mathcal{O}(1 + \sqrt{T/\gamma}), \epsilon')$-block encoding with the same query complexity. Using the same analysis in the proof of Corollary 10, to bound the error in the final output state by $\epsilon$ and the failure probability by $1-\Omega(1)$, it suffices to choose

$$\epsilon' = \mathcal{O}\left(\frac{\|u(T)\|}{\|u_0\|}\epsilon\right) \tag{182}$$

and run $\mathcal{O}((1+\sqrt{T/\gamma})\frac{\|u_0\|}{\|u(T)\|})$ rounds of amplitude amplification. Plugging these estimates back into (181) yields the desired complexity estimates. ∎

### D.4. Proof of Corollary 12.

*Proof of Corollary* 12. To use Corollary 7, we only need to estimate the $L^1$ norms of $f$ and $g$ and determine the integral truncation parameters. We have $\|f\|_{L^1(\mathbb{R})} = \mathcal{O}(1)$ as usual, and by substituting the variable $s = T^2/(4t')$, we have

$$\|g\|_{L^1(\mathbb{R}_+)} = \int_0^\infty \frac{T}{2\sqrt{\pi t'^3}} e^{-T^2/(4t')}\,\mathrm{d}t' = \int_0^\infty \frac{1}{\sqrt{\pi s}} e^{-s}\,\mathrm{d}s = \mathcal{O}(1). \tag{183}$$

Then we choose $K$ and $T'$ such that $\|g\|_{L^1((T',\infty))} = \mathcal{O}(\epsilon\|u(T)\|)$ and $\|f\|_{L^1(\mathbb{R}\setminus[-K,K])} = \mathcal{O}(\epsilon\|u(T)\|)$. This can be achieved by choosing $K = \mathcal{O}((\log(1/(\|u(T)\|\,\epsilon)))^{1/\beta})$ as usual, and we take $T' = \mathcal{O}(T^2/(\epsilon\|u(T)\|)^2)$ because

$$\|g\|_{L^1((T',\infty))} = \int_{T'}^\infty \frac{T}{2\sqrt{\pi t'^3}} e^{-T^2/(4t')}\,\mathrm{d}t' \le \int_{T'}^\infty \frac{T}{2\sqrt{\pi t'^3}}\,\mathrm{d}t' = \frac{T}{\sqrt{\pi T'}}. \tag{184}$$ ∎

## REFERENCES


[1] A. Ambainis, *Variable time amplitude amplification and quantum algorithms for linear algebra problems*, in 29th Symposium on Theoretical Aspects of Computer Science (STACS'12), LIPIcs. Leibniz Int. Proc. Inform. 14, 2012, pp. 636–647, https://doi.org/10.4230/LIPIcs.STACS.2012.636.

[2] D. An, A. M. Childs, and L. Lin, *Quantum algorithm for linear non-unitary dynamics with near-optimal dependence on all parameters*, Commun. Math. Phys., 407 (2026), 19, https://doi.org/10.1007/s00220-025-05509-w.

[3] 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 (2022), pp. 1–28, https://doi.org/10.1145/3498331.

[4] D. An, J.-P. Liu, and L. Lin, *Linear combination of Hamiltonian simulation for nonunitary dynamics with optimal state preparation cost*, Phys. Rev. Lett., 131 (2023), 150603, https://doi.org/10.1103/PhysRevLett.131.150603.

[5] D. W. Berry, *High-order quantum algorithm for solving linear differential equations*, J. Phys. A, 47 (2014), 105301, https://doi.org/10.1088/1751-8113/47/10/105301.

[6] D. W. Berry and P. C. S. Costa, *Quantum algorithm for time-dependent differential equations using Dyson series*, Quantum, 8 (2024), 1369, https://doi.org/10.22331/q-2024-06-13-1369.

[7] D. W. Berry, A. M. Childs, and R. Kothari, *Hamiltonian simulation with nearly optimal dependence on all parameters*, in Proceedings of the 56th Annual Symposium on Foundations of Computer Science, IEEE, 2015, pp. 792–809, https://doi.org/10.1109/FOCS.2015.54.

[8] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang, *Quantum algorithm for linear differential equations with exponentially improved dependence on precision*, Commun. Math. Phys., 356 (2017), pp. 1057–1081, https://doi.org/10.1007/s00220-017-3002-y.

[9] R. Bhatia, *Matrix Analysis*, Grad. Texts in Math. 169, Springer, Berlin, 1997, https://doi.org/10.1007/978-1-4612-0653-8.

[10] A. Bonito and J. E. Pasciak, *Numerical approximation of fractional powers of elliptic operators*, Math. Comp., 84 (2015), pp. 2083–2110, https://doi.org/10.1090/S0025-5718-2015-02937-8.

[11] S. Chakraborty, *Implementing any linear combination of unitaries on intermediate-term quantum computers*, Quantum, 8 (2024), 1496, https://doi.org/10.22331/q-2024-10-10-1496.

[12] 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 (2017), pp. 1920–1950, https://doi.org/10.1137/16M1087072.

[13] A. M. Childs and J.-P. Liu, *Quantum spectral methods for differential equations*, Commun. Math. Phys., 375 (2020), pp. 1427–1457, https://doi.org/10.1007/s00220-020-03699-z.

[14] A. M. Childs, Y. Su, M. C. Tran, N. Wiebe, and S. Zhu, *Theory of trotter error with commutator scaling*, Phys. Rev. X, 11 (2021), 011020, https://doi.org/10.1103/PhysRevX.11.011020.

[15] A. M. Childs and N. Wiebe, *Hamiltonian simulation using linear combinations of unitary operations*, Quantum Inform. Comput., 12 (2012), pp. 901–924, https://doi.org/10.26421/QIC12.11-12-1.

[16] 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 (2022), 040303, https://doi.org/10.1103/PRXQuantum.3.040303.

[17] P. C. S. Costa, S. Jordan, and A. Ostrander, *Quantum algorithm for simulating the wave equation*, Phys. Rev. A, 99 (2019), https://doi.org/10.1103/PhysRevA.99.012323.

[18] A. M. Dalzell, *A Shortcut to an Optimal Quantum Linear System Solver*, https://arxiv.org/abs/2406.12086, 2024.

[19] D. Fang, L. Lin, and Y. Tong, *Time-marching based quantum solvers for time-dependent linear differential equations*, Quantum, 7 (2023), 955, https://doi.org/10.22331/q-2023-03-20-955.

[20] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, *Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics*, in Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC '19, ACM, 2019, https://doi.org/10.1145/3313276.3316366.

[21] L. Grover and T. Rudolph, *Creating Superpositions that Correspond to Efficiently Integrable Probability Distributions*, https://arxiv.org/abs/quant-ph/0208112, 2002.

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

[23] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York, 2009, https://doi.org/10.1007/978-0-387-84858-7.

[24] N. J. Higham and L. Lin, *A Schur–Padé algorithm for fractional powers of a matrix*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1056–1078, https://doi.org/10.1137/10081232X.

[25] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, 1991, https://doi.org/10.1017/CBO9780511840371.

[26] I. C. F. Ipsen, *Computing an eigenvector with inverse iteration*, SIAM Rev., 39 (1997), pp. 254–291, https://doi.org/10.1137/S0036144596300773.

[27] S. Jin, N. Liu, and Y. Yu, *Quantum simulation of partial differential equations via schrödingerization*, Phys. Rev. Lett., 133 (2024), 230602, https://doi.org/10.1103/PhysRevLett.133.230602.

[28] H. Krovi, *Improved quantum algorithms for linear and nonlinear differential equations*, Quantum, 7 (2023), 913, https://doi.org/10.22331/q-2023-02-02-913.

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

[30] G. H. Low and I. L. Chuang, *Optimal Hamiltonian simulation by quantum signal processing*, Phys. Rev. Lett., 118 (2017), 010501, https://doi.org/10.1103/PhysRevLett.118.010501.

[31] G. H. Low and Y. Su, *Quantum eigenvalue processing*, in Proceedings of the 65th IEEE Symposium on Foundations of Computer Science (FOCS 2024), 2024, pp. 1051–1062, https://doi.org/10.1109/FOCS61266.2024.00070.

[32] G. H. Low and Y. Su, *Quantum Linear System Algorithm with Optimal Queries to Initial State Preparation*, https://arxiv.org/abs/2410.18178, 2024.

[33] S. McArdle, A. Gilyén, and M. Berta, *Quantum State Preparation Without Coherent Arithmetic*, https://arxiv.org/abs/2210.14892, 2022.

[34] V. V. Shende, S. S. Bullock, and I. L. Markov, *Synthesis of quantum-logic circuits*, IEEE Trans. Comput. Aid. Design Integrated Circuits Syst., 25 (2006), pp. 1000–1010, https://doi.org/10.1109/TCAD.2005.855930.

[35] P. W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM J. Comput., 26 (1997), pp. 1484–1509, https://doi.org/10.1137/S0097539795293172.

[36] Y. Subaşi, R. D. Somma, and D. Orsucci, *Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing*, Phys. Rev. Lett., 122 (2019), 060504, https://doi.org/10.1103/PhysRevLett.122.060504.

[37] S. Takahira, A. Ohashi, T. Sogabe, and T. S. Usuda, *Quantum algorithm for matrix functions by Cauchy's integral formula*, Quantum Inform. Comput., 20 (2020), pp. 14–36, https://doi.org/10.26421/QIC20.1-2-2.

[38] 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 (2021), 032422, https://doi.org/10.1103/PhysRevA.104.032422.

[39] S. Wang, S. McArdle, and M. Berta, *Qubit-efficient randomized quantum algorithms for linear algebra*, PRX Quantum, 5 (2024), 020324, https://doi.org/10.1103/PRXQuantum.5.020324.