Block-encoding-based quantum algorithm for linear systems with displacement structures

Lin-Chun Wan,[1,2] Chao-Hua Yu,[3] Shi-Jie Pan,[1] Su-Juan Qin,[1,*] Fei Gao,[1,†] and Qiao-Yan Wen[1,‡]

[1]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*

[2]*State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China*

[3]*School of Information Management, Jiangxi University of Finance and Economics, Nanchang 330032, China*

(Received 15 January 2020; revised 6 October 2021; accepted 3 November 2021; published 8 December 2021)

Matrices with the displacement structures of circulant, Toeplitz, and Hankel types as well as matrices with structures generalizing these types are omnipresent in computations of sciences and engineering. In this paper we present efficient and memory-reduced quantum algorithms for solving linear systems with such structures by devising an approach to implement the block-encodings of these structured matrices. More specifically, by decomposing $n \times n$ dense matrices into linear combinations of displacement matrices, we first deduce the parametrized representations of the matrices with displacement structures so that they can be treated similarly. With such representations, we then construct ϵ-approximate block-encodings of these structured matrices in two different data access models, i.e., the black-box model and the quantum random access memory (QRAM) data structure model. It is shown the quantum linear system solvers based on the proposed block-encodings provide a quadratic speedup with respect to the dimension over classical algorithms in the black-box model and an exponential speedup in the QRAM data structure model. In particular, these linear system solvers subsume known results with significant improvements and also can motivate new instances where there was no specialized quantum algorithm before. As an application, one of the quantum linear system solvers is applied to the linear prediction of time series, which justifies the claimed quantum speedup is achievable for problems of practical interest.

DOI: 10.1103/PhysRevA.104.062414

I. INTRODUCTION

Quantum technologies have shown their significant influence in communication and computing. On the one hand, many quantum cryptographic protocols have been proposed for protecting security and privacy [1–5]. On the other hand, quantum computing which makes use of quantum mechanical principles, such as superposition and entanglement, shows tremendous potential that outperforms the conventional computing in time complexity in solving many problems, including Boolean function computing [6,7], matrix computing [8–10], and machine learning [11–15].

Matrices encountered in practical computations often have some special structures and many problems can be transformed into solving linear systems with structures. Among various matrix structures, the circulant, Toeplitz, and Hankel types and their generalization called circulant-like, Toeplitz-like, and Hankel-like are the best known and well studied [16–18]. As of now, several efficient quantum algorithms have been proposed for solving linear systems with these displacement structures. The quantum algorithm for the Poisson equation is the earliest work on solving Toeplitz linear systems involving a specific kind of banded Toeplitz matrices [19]. A quantum algorithm for solving sparse circulant systems was proposed by Mahasinghe and Wang [20]. Thereafter, the quantum algorithm for solving circulant systems with the bounded spectral norm was presented in Ref. [21], where no assumption on the sparseness was demanded. By constructing associated circulant matrices, Wan *et al.* [22] proposed a quantum algorithm for solving Toeplitz systems in the Wiener class (defined in Sec. III C), which is an asymptotic quantum algorithm whose error is related to the dimension of the Toeplitz matrices.

The quantum algorithms introduced above can achieve excellent performance under certain circumstances and are powerful for solving large families of problems. However, these algorithms followed different ideas and employed different techniques that the quantum algorithm for linear systems with a specific displacement structure cannot be used for reference by other types. Moreover, it is intractable to generalize these algorithms to solve linear systems with the same type of generalized structures. Then an interesting question is whether we can design quantum algorithms for linear systems with various types of displacement structures in a unified way. A unified treatment of such structured matrices can often provide conceptual and computational benefits and may also give insight into finding some new solvable instances. Combined with the method of block-encoding, we answer the question in the affirmative.

A block-encoding of a matrix $\boldsymbol{M}$ is a unitary $\boldsymbol{U}$ that encodes $\boldsymbol{M}/\alpha$ as its top left block, where $\alpha \geqslant \|\boldsymbol{M}\|$ is a scaling factor. Given a way to implement block-encodings of some matrices, many operations on the matrices can be done,

*qsujuan@bupt.edu.cn
†gaof@bupt.edu.cn
‡wqy@bupt.edu.cn

including linear system solving [10,23,24]. Nevertheless, it is worth noting that the complexity of the quantum algorithm for linear systems based on block-encodings has a linear dependence on scaling factors and implementing block-encodings with preferred scale factors often requires ingenious design. Although there are some methods to implement the block-encodings for several specific matrices, such as sparse matrices [10,23], density operators [23,25], positive operator-valued measure operators [25], Gram matrices [10], and matrices stored in a quantum-accessible data structure [24,26], directly applying the methods mentioned cannot give rise to appealing quantum linear system solvers for the matrices with displacement structures. Exploiting the structures of such matrices to implement their block-encodings with favorable scaling factors and less cost of time, space, or memory deserves specialized study.

In this paper we devise an approach that decomposes $n \times n$ dense matrices into linear combinations of unitaries (LCU) and implement block-encodings of matrices with displacement structures following the idea of the LCU lemma [27]. The proposed block-encodings can give rise to efficient quantum algorithms for a set of linear systems with displacement structures, including the linear systems, such as Toeplitz systems in the Wiener class and circulant systems with the bounded spectral norm, whose quantum algorithms can be improved by our method, and the linear systems without specialized quantum algorithm before, such as some Toeplitz-like and Hankel-like linear systems. More specifically, the main contributions of this paper are as follows.

(a) We first deduce parametrized representations of $n \times n$ dense matrices by decomposing them into linear combinations of unitaries, which provide a way for the structured matrices of interest to be represented and treated similarly. The proposed LCU decompositions possess several desirable features for implementing block-encodings. (i) The elementary component unitaries are displacement matrices that can be easily implemented. (ii) The decomposition coefficients are the elements of the displacement of the decomposed matrices that can be easily calculated. (iii) For the structured matrices of interest, the number of decomposed items is roughly $O(n)$. In particular, this decomposition method provides a representation with $2n-1$ parameters for Toeplitz or Hankel matrices, which is optimal in terms of the number of parameters.

(b) Based on the proposed LCU decompositions, we then construct efficient quantum circuits in two different data access models commonly used in various quantum algorithms, i.e., the black-box model and the quantum random access memory (QRAM) data structured model, to implement the ϵ-approximate block-encodings of matrices with displacement structures. If a matrix is given in the QRAM data structure model, it will often lead to a low-complexity construction. Otherwise, the construction scheme in the black-box model may be adopted since it requires less pre-storage. In both models, we implement block-encodings whose scaling factors are proportional to the l_1-norm χ of the displacement of the structured matrices. For the structured matrices with small χ, the linear system solvers based on the proposed block-encodings provide a quadratic speedup with respect to the dimension over classical algorithms in the black-box model and an exponential speedup in the QRAM data structure model.

(c) We show that many matrices with displacement structures that are frequently encountered in various practical problems can harness the potential advantages of the proposed constructions. With the quantum linear system solver developed in the block-encoding framework, we obtain the following algorithms: (i) a quantum algorithm for Toeplitz linear systems in the Wiener class, which is an exact algorithm in which the error is independent of the dimension of the Toeplitz matrices and which positively answers the open question raised in [22]; (ii) a quantum algorithm for circulant linear systems with the bounded spectral norm, providing a quadratic improvement in the dependence on the condition number and an exponential improvement in the dependence on the precision over the quantum algorithm proposed in [21]; and (iii) an efficient quantum algorithm for linear systems with Toeplitz- or Hankel-like structures. In particular, we also obtain a quantum algorithm for banded Toeplitz or Hankel linear systems without the use of any black box or QRAM, which may be more convenient when constructing practical quantum circuits. Finally, we show that the proposed quantum algorithm for Toeplitz linear systems can be used in linear prediction of time series, which provides a concrete example to illustrate that the quantum speedup is practically achievable.

II. PRELIMINARIES

A. Matrices with displacement structure

The matrices with displacement structures arise pervasively in many contexts. Below we display three popular classes of such structured matrices and their generalizations, which are also our focus in this paper.

A Toeplitz matrix $\boldsymbol{T}_n$ is a matrix of size $n \times n$ whose elements along each diagonal are constants. More clearly,

$$\boldsymbol{T}_n = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & \cdots & t_{-(n-1)} \\ t_1 & t_0 & t_{-1} & \ddots & \vdots \\ t_2 & t_1 & t_0 & \ddots & t_{-2} \\ \vdots & \ddots & \ddots & \ddots & t_{-1} \\ t_{(n-1)} & \cdots & t_2 & t_1 & t_0 \end{pmatrix}, \tag{1}$$

where $t_{i,k} = t_{i-k}$ and $\boldsymbol{T}_n$ is determined by the sequence $\{t_j\}_{j=-(n-1)}^{n-1}$.

There is a common special case of Toeplitz matrix called a circulant matrix whose every row is a right cyclic shift of the row above it:

$$\boldsymbol{C}_n = \begin{pmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ c_2 & c_1 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & & c_{n-1} \\ c_{n-1} & \cdots & & c_1 & c_0 \end{pmatrix}. \tag{2}$$

Since the circulant matrix has some fantastic properties, it has been studied specifically in both classical and quantum settings.

Another representative class of matrices with displacement structures is the Hankel matrix. A matrix $\boldsymbol{H}_n$ is called a Hankel

matrix if it has the form

$$\boldsymbol{H}_n = \begin{pmatrix} h_0 & h_1 & h_2 & \cdots & h_{n-1} \\ h_1 & h_2 & & \ddots & h_n \\ h_2 & & \ddots & \ddots & \vdots \\ \vdots & h_{n-1} & \ddots & & h_{2n-3} \\ h_{n-1} & h_n & \cdots & h_{2n-3} & h_{2n-2} \end{pmatrix}. \quad (3)$$

The entry $h_{i,k}$ $(i,k=0,1,\ldots,n-1)$ of $\boldsymbol{H}_n$ is equal to h_{i+k} for given sequence $\{h_j\}_{j=0}^{2n-2}$. In other words, the skew diagonals of a Hankel matrix are constants.

There are some natural generalizations of these structured matrices called Toeplitz- or Hankel-like matrices (circulant-like matrices are regarded as a special case of Toeplitz-like matrices). A matrix is said to be Toeplitz- or Hankel-like if there are a few elements on some diagonals or skew diagonals of the matrix that are not equal to the others.

The computations with these structured matrices, both Toeplitz, Hankel matrices and Toeplitz-like, Hankel-like matrices, are widely applied in various areas of science and engineering. For example, in time-series analysis, the covariance matrices of weakly stationary processes are Toeplitz matrices (see [28]). The visual tracking framework of [29] requires a base sample image to generate multiple virtual samples which correspond to circulant matrices. The solvability of certain classical interpolation problems is connected with Hankel matrices (see [18]). The numerical solutions of some partial differential equations with mixed boundary conditions can be obtained by solving the Toeplitz-like linear systems generated by the discretization of the finite-difference method (see [30]). Other applications involve polynomial computations [17], image restoration [31], machine learning [32], compressed sensing [33], and so on.

Compared to general matrices, the structures in these matrices can be exploited to perform algebraic operations, such as matrix-vector multiplication, inversion, and matrix exponential, with much less running time and memory space. There are a number of classical methods that have been presented to solve the linear systems with such structures [31]. However, the time complexity of these methods is $\Omega(n)$ and it is still a hard task to tackle the problems with very large n on a classical computer.

B. Framework of block-encodings

In this section we review the framework of block-encodings introduced in [23,24].

Definition 1 (block-encoding). Suppose that $\boldsymbol{M}$ is an s-qubit operator, $\alpha,\epsilon\in\mathbb{R}^+$, and $a\in\mathbb{N}$. Then we say that the $(s+a)$-qubit unitary $\boldsymbol{U}$ is an $(\alpha;a;\epsilon)$-block-encoding of $\boldsymbol{M}$ if

$$\|\boldsymbol{M}-\alpha(\langle 0|^{\otimes a}\otimes \boldsymbol{I})\boldsymbol{U}(|0\rangle^{\otimes a}\otimes \boldsymbol{I})\|\leqslant\epsilon. \quad (4)$$

Given a block-encoding $\boldsymbol{U}$ of a matrix $\boldsymbol{M}$, one can produce the state $\boldsymbol{M}|\psi\rangle/\|\boldsymbol{M}|\psi\rangle\|$ by applying $\boldsymbol{U}$ to an initial state $|0\rangle|\psi\rangle$. Low and Chuang [23] presented a Hamiltonian simulation algorithm under the framework of block-encodings by combining the techniques of qubitization and quantum signal processing, which can simulate sparse Hamiltonians with optimal complexity. Taking this Hamiltonian simulation algorithm as a subroutine, Chakraborty *et al.* [24] developed several useful tools within the block-encoding framework such as singular-value estimation and a quantum linear system solver. In fact, they also pointed out that one can implement any smooth function of a Hamiltonian when given a block-encoding of this Hamiltonian by using the techniques developed in [34]. Furthermore, the method of block-encoding has been applied to the study of machine learning, and many quantum algorithms have been presented such as the quantum clustering algorithm [35], quantum classification algorithm [36], and quantum algorithms for semidefinite programming problems [25,37].

Although the block-encoding can be applied to various algorithms for various computational problems, we will narrow our goal to the detailed study of solving linear systems and then analyze the improvements brought about by the method proposed in this paper. Here we describe an informal version of the complexity result of the quantum algorithm for linear systems in the framework of block-encoding. Given a $(\alpha;a;\epsilon)$-block-encoding $\boldsymbol{U}$ of $\boldsymbol{M}$, there is a quantum algorithm that produces a state that is ϵ-close to $\boldsymbol{M}^{-1}|b\rangle/\|\boldsymbol{M}^{-1}|b\rangle\|$ in time $O(\kappa_{\boldsymbol{M}}[\alpha(a+T_U)\log\frac{1}{\epsilon}+T_b])$, where $\kappa_{\boldsymbol{M}}$ is the condition number of $\boldsymbol{M}$, T_U is the running time to implement $\boldsymbol{U}$, and T_b is the running time to prepare the state $|b\rangle$. This result suggests that one needs to efficiently construct block-encodings with small scaling factors.

C. Data access model

We now specify the data access model involved in the proposed quantum algorithms for solving a linear system $\boldsymbol{M}\boldsymbol{x}=\boldsymbol{b}$. For the right-hand-side vector $\boldsymbol{b}$, a unitary that produces the quantum state $|b\rangle=\sum_i b_i|i\rangle/\|\sum_i b_i|i\rangle\|$ is required in the quantum algorithm solving the linear system. It is shown that some specialized algorithms can be used to generate $|b\rangle$ efficiently under certain conditions [38,39], or our quantum algorithm may be used as a subroutine while $|b\rangle$ can be prepared by another part of a larger quantum algorithm. Alternatively, if an efficiently implementable state preparation procedure cannot be provided, we assume $\boldsymbol{b}$ can be accessed in the same way as the coefficient matrix, which will be introduced below.

For the coefficient matrix $\boldsymbol{M}$, we are given two different data access models which are most commonly used in various quantum algorithms. In the first data access model, the elements of the matrix $\boldsymbol{M}\in\mathbb{C}^{n\times n}$ are accessed by a black box $\mathcal{O}_{\boldsymbol{M}}$ acting as

$$\mathcal{O}_{\boldsymbol{M}}|i\rangle|k\rangle|0\rangle=|i\rangle|k\rangle|m_{i,k}\rangle,\quad i,k=0,1,\ldots,n-1. \quad (5)$$

This model is often referred to as the black-box model [40,41]. The access operation can be made efficiently when $m_{i,k}$ are efficiently computable or a QRAM is provided.

Kerenidis and Prakash [42] introduced a different data access model called the QRAM data structure model. This data access model stores data in QRAM with a binary tree structure and allows access in superposition. When the data are given as an $n\times n$ matrix, the matrix is stored in the binary trees by rows, and an additional binary tree is required to store the norms of the rows. Obviously, the memory requirement and the complexity of constructing this data structure must be $O(n^2)$. Although the quantum algorithms in this model do not

take the complexity of constructing data structure, reducing the memory requirement and thereby reducing the complexity of constructing data structure has many practical implications.

III. METHODS AND RESULTS

A. LCU decomposition of matrices

In this section we deduce parametrized representations of $n \times n$ matrices by decomposing them into linear combinations of unitaries, where the unitaries used as elementary components are easy to implement and the decomposition coefficients are easy to calculate. Without loss of generality, we assume that n is always a power of 2.

For a better understanding, we first introduce some necessary background information about matrix displacement.

Definition 2. For a given pair of operator matrices $(\boldsymbol{A}, \boldsymbol{B})$ and a matrix $\boldsymbol{M} \in \mathbb{C}^{n\times n}$, the linear displacement operator $\mathcal{L}(\boldsymbol{M}) : \mathbb{C}^{n\times n} \longmapsto \mathbb{C}^{n\times n}$ of Stein type is defined by

$$\mathcal{L}(\boldsymbol{M}) = \Delta_{\boldsymbol{A},\boldsymbol{B}}[\boldsymbol{M}] = \boldsymbol{M} - \boldsymbol{AMB} \tag{6}$$

and that of Sylvester type is defined by

$$\mathcal{L}(\boldsymbol{M}) = \nabla_{\boldsymbol{A},\boldsymbol{B}}[\boldsymbol{M}] = \boldsymbol{AM} - \boldsymbol{MB}. \tag{7}$$

The image $\mathcal{L}(\boldsymbol{M})$ of the operator $\mathcal{L}$ is called the displacement of the matrix $\boldsymbol{M}$. According to the specific structure of the matrix $\boldsymbol{M}$, one can instantiate the operator matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ with desirable properties. Here, for our purposes, we introduce one of the customary choices of $\boldsymbol{A}$ and $\boldsymbol{B}$, the unit f-circulant matrix $\boldsymbol{Z}_f$, which we will define next.

Definition 3 (unit f-circulant matrix). For a real-valued scalar f, an $n \times n$ unit f-circulant matrix is defined as

$$\boldsymbol{Z}_f = \begin{pmatrix} 0 & 0 & \cdots & f \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \end{pmatrix}. \tag{8}$$

It is easy to verify that $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_{-1}$ are unitary matrices, as well as $\boldsymbol{Z}_1^i$ and $\boldsymbol{Z}_{-1}^i$, $i = 0, 1, \ldots, n-1$. By inverting the displacement operators with operator matrices $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_{-1}$, we then demonstrate how to decompose an $n \times n$ matrix as linear combinations of unitaries.

Theorem 1. Let $\boldsymbol{M} \in \mathbb{C}^{n\times n}$, $m_{i,k}$ be the kth element of the ith row of $\boldsymbol{M}$,

$$\boldsymbol{J} = \begin{pmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{pmatrix}$$

be the reversal matrix, and

$$g(k) := \begin{cases} 0, & k = 0, 1, 2, \ldots, n-2 \\ 1, & k = n-1. \end{cases} \tag{9}$$

Then $\boldsymbol{M}$ can be decomposed as (i)

$$\boldsymbol{M} = \frac{1}{2} \sum_{i,k=0}^{n-1} \hat{m}_{i,k} \boldsymbol{Z}_1^i \boldsymbol{J} \boldsymbol{Z}_{-1}^{n-1-k}, \tag{10}$$

where $\hat{m}_{i,k} = m_{i,k} - (-1)^{g(k)} m_{(i-1)\bmod n,(k+1)\bmod n}$ is the kth element of the ith row of matrix $\Delta_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{M}]$, and (ii)

$$\boldsymbol{M} = \frac{1}{2} \sum_{i,k=0}^{n-1} \tilde{m}_{i,k} \boldsymbol{Z}_1^i \boldsymbol{Z}_{-1}^{n-1-k}, \tag{11}$$

where $\tilde{m}_{i,k} = m_{(i-1)\bmod n,k} - (-1)^{g(k)} m_{i,(k+1)\bmod n}$ is the kth element of the ith row of matrix $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{M}]$.

See Appendix A for a proof of Theorem 1.

We call these two decompositions Stein type and Sylvester type, respectively. From this theorem, using the displacement matrices $\{\boldsymbol{J}, \boldsymbol{Z}_1^i, \boldsymbol{Z}_{-1}^i, i = 0, 1, \ldots, n-1\}$ as the elementary components, one can decompose an $n \times n$ matrix into linear combinations of these simple unitaries, and the decomposition coefficients are the elements of the displacement of the matrix which can be easily calculated. The proposed LCU decompositions actually provide a way to parametrize the decomposed matrices, that is, we can use the elements of $\mathcal{L}(\boldsymbol{M})$ as a parametrized representation of $\boldsymbol{M}$.

B. Implementation of block-encodings of matrices with displacement structures

In this section we will show that the Toeplitz matrices, Hankel matrices and their generalizations can generate elegant parametrized representations with nearly $O(n)$ parameters when associated with operator matrices $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_{-1}$. Furthermore, we will illustrate how to implement the block-encodings of such matrices in detail.

Taking the Toeplitz matrices as an example, we compute their Sylvester displacement first:

$$\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_n] = \begin{pmatrix} t_{n-1} - t_{-1} & t_{n-2} - t_{-2} & \cdots & 2t_0 \\ 0 & 0 & \cdots & t_{-(n-1)} + t_1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & t_{-1} + t_{n-1} \end{pmatrix}. \tag{12}$$

Then $\boldsymbol{T}_n$ can be decomposed into a linear combination of unitaries as

$$\begin{aligned} \boldsymbol{T}_n = \tfrac{1}{2}\big[&2t_0\boldsymbol{I} + (t_1 + t_{-(n-1)})\boldsymbol{Z}_1^1 + \cdots + (t_{n-1} + t_{-1})\boldsymbol{Z}_1^{n-1} \\ &+ (t_1 - t_{-(n-1)})\boldsymbol{Z}_{-1}^1 + \cdots + (t_{n-1} - t_{-1})\boldsymbol{Z}_{-1}^{n-1}\big]. \end{aligned} \tag{13}$$

We would like to emphasize that since the Toeplitz matrices are represented with $2n-1$ parameters, this decomposition should be optimal for the number of items.

To implement the block-encodings of Toeplitz matrices from Eq. (13), we define two state preparation operators as

$$\begin{aligned} \boldsymbol{V}_{(\nabla[\boldsymbol{T}_n])}|0\rangle &= |V_{(\nabla[\boldsymbol{T}_n])}\rangle \\ &= \frac{1}{\sqrt{\chi_{\boldsymbol{T}_n}}} \sum_{j=0}^{n-1} \sqrt{t_j + t_{-[(n-j)\bmod n]}}|j\rangle \\ &\quad + \frac{1}{\sqrt{\chi_{\boldsymbol{T}_n}}} \sum_{j=n}^{2n-1} \sqrt{t_{(j-n)} - t_{-[(2n-j)\bmod n]}}|j\rangle \\ &\equiv \frac{1}{\sqrt{\chi_{\boldsymbol{T}_n}}} \sum_{j=0}^{2n-1} \sqrt{\tilde{t}_j}|j\rangle, \end{aligned} \tag{14}$$

$$\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n]^*)}|0\rangle = |V_{(\nabla[\boldsymbol{T}_n]^*)}\rangle = \frac{1}{\sqrt{\chi_{\boldsymbol{T}_n}}}\sum_{j=0}^{2n-1}\sqrt{\tilde{t}_j^*}|j\rangle, \tag{15}$$

where $\chi_{\boldsymbol{T}_n} = \sum_{j=0}^{2n-1}|\tilde{t}_j|$ and the square root operation takes the main square root of $\tilde{t}_j$ and $\tilde{t}_j^*$. Then we define a controlled unitary

$$\texttt{Select}\boldsymbol{U}_{T_n} = \sum_{j=0}^{n-1}|j\rangle\langle j|\otimes\boldsymbol{Z}_1^j + \sum_{j=n}^{2n-1}|j\rangle\langle j|\otimes\boldsymbol{Z}_{-1}^{j-n}. \tag{16}$$

Since $\sqrt{\tilde{t}_j}(\sqrt{\tilde{t}_j^*})^* = \tilde{t}_j$, it is easy to verify that

$$\boldsymbol{T}_n = \frac{\chi_{\boldsymbol{T}_n}}{2}[\langle 0|(\boldsymbol{V}^\dagger_{(\nabla[\boldsymbol{T}_n]^*)}\otimes\boldsymbol{I})\texttt{Select}\boldsymbol{U}_{T_n}(\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n])}\otimes\boldsymbol{I})|0\rangle], \tag{17}$$

which means that $(\boldsymbol{V}^\dagger_{(\nabla[\boldsymbol{T}_n]^*)}\otimes\boldsymbol{I})\texttt{Select}\boldsymbol{U}_{T_n}(V_{(\nabla[\boldsymbol{T}_n])}\otimes\boldsymbol{I})$ is a block-encoding of $\boldsymbol{T}_n$.

Implementing the block-encoding of $\boldsymbol{T}_n$ is to implement $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n])}$, $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n]^*)}$, and $\texttt{Select}\boldsymbol{U}_{T_n}$. For quantum state preparation operators, we need to construct a quantum circuit that prepares $|V_{(\nabla[\boldsymbol{T}_n])}\rangle$ reversibly. In the black-box model, this cannot be done by the traditional black-box quantum state preparation [43] because its success probability cannot be increased arbitrarily close to certainty, which will make the final error uncontrollable. We provide a reversible algorithm called steerable black-box quantum state preparation, based on fixed-point amplitude amplification [44], which can prepare a quantum state with the success probability increasing arbitrarily close to certainty. Since it will be referred to multiple times as a key subroutine of our algorithm and may be of independent interest for other quantum algorithms, we make a formal statement here.

Lemma 1. For a vector $\boldsymbol{x}\in\mathbb{C}^{n\times 1}$, $|\boldsymbol{x}|_{\max} = \max_i|\sqrt{x_i}|$ is a small constant, and the elements are given by a black box $\mathcal{O}_x$ acting as

$$\mathcal{O}_x|i\rangle|0\rangle \to |i\rangle|x_i\rangle. \tag{18}$$

Then the steerable black-box quantum state preparation algorithm generates a state, up to a global phase, that is an ϵ_p approximation of

$$|x\rangle = \frac{1}{\sqrt{\|\boldsymbol{x}\|_1}}\sum_{i=0}^{n-1}\sqrt{x_i}|i\rangle \tag{19}$$

with a success probability of at least $1-\delta^2$, using $O(\frac{\sqrt{n}\log(1/\delta)}{\sqrt{\|\boldsymbol{x}\|_1}})$ queries of $\mathcal{O}_x$ and additional $O(\frac{\sqrt{n}\log(1/\delta)}{\sqrt{\|\boldsymbol{x}\|_1}}\text{polylog}(\frac{n}{\epsilon_p\sqrt{\|\boldsymbol{x}\|_1}}))$ elementary gates.

See Appendix B for a proof of Lemma 1.

The operators defined by Eqs. (14)–(16) are actually the operators of the LCU circuit [27] which has been used in many quantum algorithms [24,34,45,46]. Here we implement this circuit with two different data access models introduced in Sec. II C. The method in the QRAM data structure model is especially useful for the structured matrices whose displacements have been stored in the data structure, while the method in the black-box model will have a wider range of applications because of the flexibility of its implementation. We summarize the results as follows.

Theorem 2. Let $\boldsymbol{T}_n\in\mathbb{C}^{n\times n}$ be a Toeplitz matrix. (i) If the elements of $\boldsymbol{T}_n$ are provided by a black box $\mathcal{O}_{\boldsymbol{T}_n}$, i.e.,

$$\mathcal{O}_{\boldsymbol{T}_n}|i\rangle|k\rangle|0\rangle = |i\rangle|k\rangle|t_{i,k}\rangle,$$

one can implement a $(\chi_{\boldsymbol{T}_n}/2;\log(n)+2;\epsilon)$-block-encoding of $\boldsymbol{T}_n$ with $O(\frac{\sqrt{n}\log(\chi_{\boldsymbol{T}_n}/\epsilon)}{\sqrt{\chi_{\boldsymbol{T}_n}}})$ uses of $\mathcal{O}_{\boldsymbol{T}_n}$ and additionally using $O(\frac{\sqrt{n}}{\sqrt{\chi_{\boldsymbol{T}_n}}}\text{polylog}(\frac{n\chi_{\boldsymbol{T}_n}}{\epsilon}))$ elementary gates. (ii) If the nonzero elements of Sylvester displacement of $\boldsymbol{T}_n$, i.e., $\{\tilde{t}_j\}_{j=0}^{2n-1}$, are stored in the QRAM data structure as shown in Lemma 6, one can implement a $(\chi_{\boldsymbol{T}_n}/2;\log(n)+1;\epsilon)$-block-encoding of $\boldsymbol{T}_n$ with gate complexity $O(\text{polylog}(n\chi_{\boldsymbol{T}_n}/\epsilon))$ and memory cost $O(n)$.

See Appendix C for a proof of Theorem 2.

Moreover, we show that how to implement the block-encoding of the Toeplitz-like matrices. Let $\boldsymbol{T}_L\in\mathbb{C}^{n\times n}$ be a Toeplitz-like matrix, $(\boldsymbol{T}_L)_{i,k} = \tau_{i,k}$, $(\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L])_{i,k} = \tilde{\tau}_{i,k}$. We first define two state preparation operators as

$$\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L])}|0\rangle|0\rangle = |V_{(\nabla[\boldsymbol{T}_L])}\rangle = \frac{1}{\sqrt{\chi_{\boldsymbol{T}_L}}}\sum_{i=0}^{n-1}\sum_{k=0}^{n-1}\sqrt{\tilde{\tau}_{i,k}}|i\rangle|k\rangle, \tag{20}$$

$$\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L]^*)}|0\rangle|0\rangle = |V_{(\nabla[\boldsymbol{T}_L]^*)}\rangle = \frac{1}{\sqrt{\chi_{\boldsymbol{T}_L}}}\sum_{i=0}^{n-1}\sum_{k=0}^{n-1}\sqrt{\tilde{\tau}_{i,k}^*}|i\rangle|k\rangle, \tag{21}$$

where $\chi_{\boldsymbol{T}_L} = \sum_{i=0}^{n-1}\sum_{k=0}^{n-1}|\tilde{\tau}_{i,k}|$ and the square root operation takes the main square root of $\tilde{\tau}_{i,k}$ and $\tilde{\tau}_{i,k}^*$. Then we define

$$\texttt{Select}\boldsymbol{U}_{T_L} = \left(\sum_{i=0}^{n-1}|i\rangle\langle i|\otimes\boldsymbol{I}\otimes\boldsymbol{Z}_1^i\right) \times\left(\sum_{k=0}^{n-1}\boldsymbol{I}\otimes|k\rangle\langle k|\otimes\boldsymbol{Z}_{-1}^{n-1-k}\right). \tag{22}$$

Since $\sqrt{\tilde{\tau}_{i,k}}(\sqrt{\tilde{\tau}_{i,k}^*})^* = \tilde{\tau}_{i,k}$, it is easy to verify that

$$\boldsymbol{T}_L = \frac{\chi_{\boldsymbol{T}_L}}{2}[\langle 0|(\boldsymbol{V}^\dagger_{(\nabla[\boldsymbol{T}_L]^*)}\otimes\boldsymbol{I})\texttt{Select}\boldsymbol{U}_{T_L}(\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L])}\otimes\boldsymbol{I})|0\rangle], \tag{23}$$

which means that $(\boldsymbol{V}^\dagger_{(\nabla[\boldsymbol{T}_L]^*)}\otimes\boldsymbol{I})\texttt{Select}\boldsymbol{U}_{T_L}(\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L])}\otimes\boldsymbol{I})$ is a block-encoding of $\boldsymbol{T}_L$.

It seems difficult to implement the state preparation operators $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L])}$ and $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L]^*)}$ with complexity less than $O(n)$ in the black-box model. However, as shown in Eq. (12), the Sylvester displacement of a Toeplitz matrix has nonzero elements only along its first row and last column. For a Toeplitz-like matrix $\boldsymbol{T}_L$, there are a few elements on some diagonals of the matrix that are not equal to the others. Then it can be directly verified that the submatrix left by deleting the first row and last column of the Sylvester displacement of the Toeplitz-like matrix is a sparse matrix. Based on this observation, we construct the block-encodings of the Toeplitz-like matrices, and the results are summarized as follows.

Corollary 1. Let $\boldsymbol{T}_L\in\mathbb{C}^{n\times n}$ be a Toeplitz-like matrix. Suppose that the submatrix left by deleting the first row and last column of $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L]$ is $(d-1)$-row sparse, i.e., there are at

most $d-1$ nonzero elements in each row. (i) If the elements of $\boldsymbol{T}_L$ are provided by a black box $\mathcal{O}_{\boldsymbol{T}_L}$, i.e.,

$$\mathcal{O}_{\boldsymbol{T}_L}|i\rangle|k\rangle|0\rangle = |i\rangle|k\rangle|\tau_{i,k}\rangle,$$

and a black box that computes the positions of the distinct elements on diagonals of the Toeplitz-like matrices is provided, one can implement a $(\chi_{\boldsymbol{T}_L}/2; 2\log(n)+2; \epsilon)$-block-encoding of $\boldsymbol{T}_L$ with $O(\frac{\sqrt{nd}\log(\chi_{\boldsymbol{T}_L}/\epsilon)}{\sqrt{\chi_{\boldsymbol{T}_L}}})$ uses of $\mathcal{O}_{\boldsymbol{T}_L}$ and additionally using $O(\frac{\sqrt{nd}}{\sqrt{\chi_{\boldsymbol{T}_L}}}\text{polylog}(\frac{nd\chi_{\boldsymbol{T}_L}}{\epsilon}))$ elementary gates. (ii) If the nonzero elements of Sylvester displacement of $\boldsymbol{T}_L$, i.e., $\{\tilde{\tau}_{i,k}\}_{i,k=0}^{n-1}$, are stored in the QRAM data structure as shown in Lemma 7, one can implement a $(\chi_{\boldsymbol{T}_L}/2; 2\log(n); \epsilon)$-block-encoding of $\boldsymbol{T}_L$ with gate complexity $O(\text{polylog}(n\chi_{\boldsymbol{T}_L}/\epsilon))$ and memory cost $O(dn\log n)$.

See Appendix D for a proof of Corollary 1.

Remark 1. One might be confused about the QRAM data structure used in this paper, which stores $\tilde{m}_{i,k}$ instead of $m_{i,k}$ for a matrix $\boldsymbol{M}$. In fact, in most quantum algorithms using this data structure, such as those in [24,26], the stored entries are $m_{i,k}^p$, $p \in [0,2]$. Since $\tilde{m}_{i,k}$, as defined below Eq. (11), can be calculated as efficiently as $m_{i,k}^p$, $p \in [0,2]$, our assumption about such a data structure is not stronger than the assumption in the previous algorithms.

Remark 2. Results (i) and (ii) in Theorem 2, as well as Corollary 1, use different data access models, where the black-box model queries the elements of structured matrices, while the QRAM data structure model requires that the elements of their displacements have been stored. Due to the different data access models, a direct comparison of the complexity of these results is inappropriate, and it is unwise to claim which result is more advantageous based on the complexity. In practical applications, one should choose the appropriate method according to the pattern from which the data can be obtained.

For a circulant matrix $\boldsymbol{C}_n$, computing its Sylvester displacement, the LCU decomposition of $\boldsymbol{C}_n$ is $\boldsymbol{C}_n = \sum_{j=0}^{n-1} c_j \boldsymbol{Z}_1^j$. Similar to the implementation of the block-encodings of the Toeplitz matrices, we can implement block-encodings of $\boldsymbol{C}_n$. Since the number of decomposed items of the circulant matrices is less than that of the Toeplitz matrices, fewer resources are required to implement the block-encodings than stated in Theorem 2. The same conclusion holds for circulant-like matrices.

For a Hankel matrix $\boldsymbol{H}_n$, it can be decomposed as follows, by computing their Stein displacements,

$$\begin{aligned}\boldsymbol{H}_n = \tfrac{1}{2}\big[&2h_{n-1}\boldsymbol{J} + (h_n + h_0)\boldsymbol{Z}_1^1\boldsymbol{J} \\ &+ \cdots + (h_{2n-2} + h_{n-2})\boldsymbol{Z}_1^{n-1}\boldsymbol{J} + (h_{n-2} - h_{2n-2})\boldsymbol{J}\boldsymbol{Z}_{-1}^1 \\ &+ \cdots + (h_0 - h_n)\boldsymbol{J}\boldsymbol{Z}_{-1}^{n-1}\big]. \end{aligned} \tag{24}$$

Note that $\boldsymbol{J}\boldsymbol{Z}_{-1}^i = -\boldsymbol{Z}_{-1}^{n-i}\boldsymbol{J}$, and this decomposition is equivalent to

$$\begin{aligned}\boldsymbol{H}_n = \tfrac{1}{2}\big[&2h_{n-1}\boldsymbol{J} + (h_n + h_0)\boldsymbol{Z}_1^1\boldsymbol{J} \\ &+ \cdots + (h_{2n-2} + h_{n-2})\boldsymbol{Z}_1^{n-1}\boldsymbol{J} + (h_{2n-2} - h_{n-2})\boldsymbol{Z}_{-1}^{n-1}\boldsymbol{J} \\ &+ \cdots + (h_n - h_0)\boldsymbol{Z}_{-1}^1\boldsymbol{J}\big]. \end{aligned} \tag{25}$$

Since $\boldsymbol{J} = \boldsymbol{\sigma}_x^{\otimes \log n}$ ($\boldsymbol{\sigma}_x$ is a Pauli-X operator), we can implement an ϵ-approximate block-encoding of $\boldsymbol{H}_n$ by constructing a quantum circuit similar to the block-encoding implementation of $\boldsymbol{T}_n$, where the scaling factor is $\chi'_{\boldsymbol{H}_n}/2 = \sum_{i=0}^{n-1}\sum_{k=0}^{n-1}|\hat{h}_{i,k}|/2$. Also, the block-encodings of the Hankel-like matrices can be implemented similarly to those of the Toeplitz-like matrices.

In many cases, such as visual tracking [47], we need to extend the non-Hermitian matrices with displacement structures to Hermitian. For the extended matrices

$$\overline{\boldsymbol{M}} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{M} \\ \boldsymbol{M}^\dagger & \boldsymbol{0} \end{pmatrix}, \tag{26}$$

let $\boldsymbol{U}$ be a $(\chi; a; \epsilon)$-block-encoding of $\boldsymbol{M}$; then we can implement a $(\chi; a; \epsilon)$-block-encoding of $\overline{\boldsymbol{M}}$ by using the method of complementing block-encoded matrices [24]. The cost of implementing this block-encoding is nearly twice the cost of implementing $\boldsymbol{U}$. Therefore, without a loss of generality, we can assume that the matrices studied in the following sections are Hermitian.

C. Quantum algorithm for linear systems with displacement structures

As mentioned, given a block-encoding $\boldsymbol{U}$ of a matrix $\boldsymbol{M}$, one can perform a number of useful operations on $\boldsymbol{M}$. In particular, combining the variable-time amplitude amplification technique [48] and the idea of implementing smooth functions of block-encoded Hamiltonians [34], Chakraborty *et al.* [24] presented a quantum algorithm for linear systems within the block-encoding framework. We invoke the complexity of this algorithm as follows.

Lemma 2 (variable-time quantum linear systems algorithm [24]). Let $\boldsymbol{H}$ be an $n \times n$ Hermitian matrix and λ_i be the nonzero eigenvalues of $\boldsymbol{H}$ such that $\lambda_i \in [-1, -1/\kappa_{\boldsymbol{H}}] \cup [1/\kappa_{\boldsymbol{H}}, 1]$, where $\kappa_{\boldsymbol{H}} > 2$ is the condition number of $\boldsymbol{H}$. Suppose that there is an $(\alpha; a; \delta)$-block-encoding $\boldsymbol{U}$ of $\boldsymbol{H}$, where $\delta = o(\epsilon/\kappa_{\boldsymbol{H}}^2 \log^3(\frac{\kappa_{\boldsymbol{H}}}{\epsilon}))$, and $\boldsymbol{U}$ can be implemented in time T_U. Also suppose that the state $|b\rangle$ can be prepared in time T_b. Then there exists a quantum algorithm that produces a state that is ϵ-close to $\boldsymbol{H}^{-1}|b\rangle/\|\boldsymbol{H}^{-1}|b\rangle\|$ in time

$$O\left(\kappa_{\boldsymbol{H}}\left[\alpha(a+T_U)\log^2\left(\frac{\kappa_{\boldsymbol{H}}}{\epsilon}\right) + T_b\right]\log(\kappa_{\boldsymbol{H}})\right).$$

As mentioned in Sec. II C, there are some methods that can prepare right-hand-side state $|b\rangle$ in time $O(\text{polylog}n)$ under certain conditions. Even if such an efficient state preparation procedure cannot be provided, the complexity of preparing $|b\rangle$ will not exceed the complexity of implementing block-encodings of structured matrices in the same data access model. More specifically, in the black-box model, the query complexity of preparing an n-dimensional quantum state is $O(\sqrt{n})$ [43]. In the QRAM data structured model, one can prepare the quantum state $|b\rangle$ with complexity $O(\text{polylog}n)$ [42]. Here, following the assumption of previous quantum algorithms [8,27], we neglect the error in producing $|b\rangle$ since this error is independent of the design of the quantum algorithm.

Therefore, according to Lemma 2, the method proposed in Theorem 2 can induce a quantum algorithm to

solve the structured linear systems with complexity (i) $\tilde{O}(\kappa_{\boldsymbol{H}}\sqrt{\chi}\sqrt{n}\text{polylog}(1/\epsilon))$ in the black-box model (we use the symbol $\tilde{O}$ to hide redundant polylogarithmic factors, and since the elementary gate requirement in the black-box model is larger than the query complexity by logarithmic factors, we will not describe them individually from now on) and (ii) $\tilde{O}(\kappa_{\boldsymbol{H}}\chi\,\text{polylog}(n/\epsilon))$ in the QRAM data structure model. Obviously, this algorithm is expected to be efficient for matrices whose χ is small.

Many matrices with displacement structures encountered in a diverse range of applications satisfy this criterion. One of the typical examples should be Toeplitz matrices in the Wiener class [28,31]. This kind of matrix is usually obtained by the discretization of some continuous problems. More specifically, let $C_{2\pi}$ be the set of all 2π-periodic continuous real-valued functions defined on $[0, 2\pi]$. Let $\boldsymbol{T}_n$ be the $n \times n$ Toeplitz matrices whose elements of every diagonal are given by the Fourier coefficients of a function $f \in C_{2\pi}$, i.e.,

$$t_j = \frac{1}{2\pi}\int_0^{2\pi} f(\lambda)e^{-ij\lambda}d\lambda, \quad j = 0, \pm1, \pm2, \ldots. \tag{27}$$

The function f is called the generating function of the sequence of Toeplitz matrices $\boldsymbol{T}_n(1 \leqslant n < \infty)$. The sequence of Toeplitz matrices $\boldsymbol{T}_n(1 \leqslant n < \infty)$ whose element sequence $\{t_j\}$ is absolutely summable is said to be in the Wiener class. That is to say, for Toeplitz matrices in the Wiener class, there must be a constant ρ such that

$$\sum_{j=-\infty}^{\infty} |t_j| < \rho. \tag{28}$$

Thus, for Toeplitz matrices in the Wiener class, we have

$$\begin{aligned}\chi_{\boldsymbol{T}_n} &= 2|t_0| + |t_1 + t_{-(n-1)}| + \cdots + |t_{n-1} + t_{-1}| \\ &\quad + |t_1 - t_{-(n-1)}| + \cdots + |t_{n-1} - t_{-1}| \\ &\leqslant 2\sum_{j=-(n-1)}^{n-1} |t_j| < 2\rho.\end{aligned} \tag{29}$$

The complexity of the quantum algorithm for solving the Toeplitz systems in the Wiener class is (i) $\tilde{O}(\kappa_{\boldsymbol{T}_n}\sqrt{n}\text{polylog}(1/\epsilon))$ in the black-box model and (ii) $\tilde{O}(\kappa_{\boldsymbol{T}_n}\text{polylog}(n/\epsilon))$ in the QRAM data structure model. When the Toeplitz matrices are well conditioned [we call a matrix $\boldsymbol{M}$ well conditioned when $\kappa_{\boldsymbol{M}} \in O(\text{polylog}\, n)$] and $1/\epsilon \in O(\text{poly}\, n)$, the quantum algorithm is (i) quadratically faster than the classical methods in the black-box model and (ii) exponentially faster than the classical methods in the QRAM data structure model.

As of now, some work regarding Toeplitz matrices has been studied in the quantum setting. Wan *et al.* [22] adopted associated circulant matrices to approximate the Toeplitz matrices in the Wiener class and solved the circulant linear systems by accessing the values of the generating function at specific points in parallel. It is an asymptotic quantum algorithm whose error is related to the dimension of the Toeplitz matrices. An open question raised in [22] was whether there is an exact quantum algorithm where the error is independent of the dimension. The algorithm suggested in this section gives the answer, and it is more advantageous when rigorous precision is required or the Toeplitz matrices and their associated circulant matrices do not approach quickly as the dimension increases. Additionally, for the cases where no generating function is provided, our algorithm can improve the dependence on the condition number and precision since the complexity of the quantum algorithm proposed in [22] has a quadratic dependence on the condition number and a linear dependence on the precision.

Besides the Toeplitz matrices in the Wiener class, for the circulant matrices $\boldsymbol{C}_n$, it is often the case in practical applications that c_j are non-negative for all j and the spectral norms $\|\boldsymbol{C}_n\| = \sum_{j=0}^{n-1} c_j$ of $\boldsymbol{C}_n$ are constants. Thus, $\chi_{\boldsymbol{C}_n}$ will be bounded by some constants, and the quantum algorithm based on the proposed block-encodings can solve these circulant linear systems with complexity (i) $\tilde{O}(\kappa_{\boldsymbol{C}_n}\sqrt{n}\text{polylog}(1/\epsilon))$ in the black-box model and (ii) $\tilde{O}(\kappa_{\boldsymbol{C}_n}\text{polylog}(n/\epsilon))$ in the QRAM data structure model.

For the circulant matrices described above, based on the observation of LCU decomposition of $\boldsymbol{C}_n$, Zhou and Wang [21] used the method of simulating the Hamiltonian with a truncated Taylor series [45] and the algorithm of Harrow *et al.* [8] to solve the associated linear systems. Under the assumption that there is an oracle that can prepare the state $\frac{1}{\sqrt{\chi_{c_n}}}\sum_{j=0}^{n-1}\sqrt{c_j}|j\rangle$ in time $O(\text{polylog}n)$, the complexity of the quantum algorithm proposed in [21] is $\tilde{O}(\kappa_{\boldsymbol{C}_n}^2\,\text{polylog}(n)/\epsilon)$. However, it is not always feasible to prepare this initial state with such a running time, especially in the black-box model. In this paper we show in detail how to implement the preparation of this state in two different data access models. In particular, when using the same data access model, the algorithm proposed in this paper provides complexity improvement on $\kappa_{\boldsymbol{C}_n}$ and $1/\epsilon$, which comes from the use of updated techniques for the Hamiltonian simulation and linear system solver.

There are some Hankel matrices of which $\sum_{j=0}^{\infty}|h_j|$ are convergent, such as

$$\boldsymbol{H}_{i,k} = \frac{1}{(i+k+1)!}, \tag{30}$$

which arise in determining the covariance structure of an iterated Kolmogorov diffusion [49]. In addition, there are also some Hankel matrices generated by the discretization of some functions [50], just like the Toeplitz matrices in the Wiener class. Then we can implement block-encodings of these Hankel matrices with bounded scaling factors, which will derive a quantum algorithm that solves the Hankel linear systems with significant speedup (same as that for Toeplitz systems in the Wiener class) in both data access models.

It immediately follows that, for the Toeplitz- or Hankel-like matrices, if they satisfy a constraint similar to one of the above forms, we can then solve the linear systems with such structures efficiently by using the block-encodings constructed in Corollary 1. For example, finding a greatest common divisor of univariate polynomials involves solutions of Toeplitz-like systems and in some cases, as shown in [51], the elements of the displacement of the coefficient matrices are absolutely summable. We would like to emphasize the method proposed in this paper will result in efficient quantum linear system solvers for the structured matrices whose χ is small, not only for the matrices introduced in this section.

IV. APPLICATION TO TIME-SERIES ANALYSIS

Note that the quantum algorithm introduced in the preceding section always outputs a state encoding the solution of the linear system in its amplitudes. Reading out all the classical information of the solution is time consuming. To illustrate that the quantum speedup is practically achievable, we provide a concrete example where the coefficient matrix satisfies the specification and some useful information can be extracted from the output.

More specifically, we apply the quantum algorithm for the Toeplitz systems in the Wiener class to solve the linear prediction problem of time series. Predicting the future value of a discrete-time stochastic process with a set of past samples of the process is one of the most important problems in time-series analysis. For linear prediction, we need to estimate the predicted value by a linear combination of the past samples.

To present the problem clearly, we first introduce some terminology used in signal processing (for details, see [52]). Let $u(k)$ be a discrete-time stationary zero-mean complex-valued process. A finite-impulse-response linear filter of order n is of the form

$$\hat{u}(i) = \sum_{k=1}^{n} w_k^* u(i-k), \tag{31}$$

where $\hat{u}(i)$ is the filter output based on the data $\{u(k)\}_{k=i-n}^{i-1}$ and $\{w_k\}_{k=1}^{n}$ are the impulse responses of the filter. For the situation of linear prediction, the desired response is $u(i)$, representing the actual sample of the input process at time i. The difference between the desired response $u(i)$ and the filter output $\hat{u}(i)$ is called the estimation error. To estimate the desired response, we should choose the impulse responses $\{w_k\}_{k=1}^{n}$ by making the estimation error as small as possible in some statistical sense.

According to the Wiener filter theory, when the estimation error is optimized in the mean-square-error sense, the impulse responses $\{w_k\}_{k=1}^{n}$ are given by the solution of the linear system

$$\boldsymbol{R}\boldsymbol{w} = \boldsymbol{r}. \tag{32}$$

Here

$$\boldsymbol{R} = \begin{pmatrix} r(0) & r(1) & \cdots & r(n-1) \\ r^*(1) & r(0) & \cdots & r(n-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(n-1) & r^*(n-2) & \cdots & r(0) \end{pmatrix}, \tag{33}$$

$$\boldsymbol{r} = \begin{pmatrix} r^*(1) \\ r^*(2) \\ \vdots \\ r^*(n) \end{pmatrix}, \tag{34}$$

where $r(k) = \mathbb{E}[u(j)u^*(j-k)]$ ($\mathbb{E}$ is the expectation operator) are the autocovariances of the input process for lag k. This linear system is commonly called the Wiener-Hopf equations.

Note that the covariance matrix $\boldsymbol{R}$ is an $n \times n$ Hermitian Toeplitz matrix and is almost always positive definite. For a discrete-time stationary process, if the autocovariances of the process are absolutely summable, i.e., $\sum_{k=-\infty}^{\infty} |r(k)| < \infty$, then the function $\tilde{f}(\lambda)$ that takes $r(k)$ as its Fourier coefficients is called the power spectral density function of the process. The power spectral density functions ordinarily exist for the stochastic processes encountered in the physical sciences and engineering. Thus, $\boldsymbol{R}$ is a Toeplitz matrix generated by $\tilde{f}(\lambda)$ and in the Wiener class. Moreover, the eigenvalues λ_k of a Hermitian Toeplitz matrix satisfy

$$f_{\min} \leqslant \lambda_k \leqslant f_{\max}, \tag{35}$$

where $f_{\min}$ and $f_{\max}$ represent the smallest value and the largest value of the generating function, respectively. When the spectral density function is bounded (which can be guaranteed by the continuity of $\tilde{f}(\lambda)$ on $[0, 2\pi]$), the condition number of $\boldsymbol{R}$ will also be bounded.

For the case of known statistics, i.e., the autocovariances of the stationary process are known, one can query the elements of the covariance matrix $\boldsymbol{R}$ by the "black box." Alternatively, the covariance matrix $\boldsymbol{R}$ can be stored in the QRAM data structure as shown in Lemma 6 in advance. Similarly, the vector $\boldsymbol{r}$ can also be provided with two different data access models. Then we can prepare a quantum state $|r\rangle$ with complexity (i) $O(\sqrt{n}/\|\boldsymbol{r}\|_2)$ in the black-box model [41] and (ii) $O(\text{polylog}(n/\epsilon))$ in the QRAM data structure model [42]. Note that $\|\boldsymbol{r}\|_2 = O(\chi_{\boldsymbol{R}})$ will be a constant. By using the quantum algorithm for solving the Toeplitz systems, we can get a quantum state $|w\rangle$ proportional to the solution of Eq. (32) with complexity (i) $\tilde{O}(\sqrt{n}\text{polylog}(1/\epsilon))$ in the black-box model and (ii) $\tilde{O}(\text{polylog}(n/\epsilon))$ in the QRAM data structure model.

Given the vector $\boldsymbol{u} = [u(i-1), \ldots, u(i-n)]^T$ with the appropriate data access model, the quantum state $|u\rangle$ can also be prepared by the methods described in Sec. III C. Then the filter output $\hat{u}(i) = \sum_{k=1}^{n} w_k^* u(i-k)$ can be approximately computed up to some factor by evaluating the inner product of $|u\rangle$ and $|w\rangle$ using the Hadamard test [53,54]. Since there is no need to read out all the values of the obtained quantum state, the quantum speedup of solving the linear system is preserved. This process in fact provides an example showing that quantum algorithms can yield significant speedup for problems of practical interest.

V. DISCUSSION

There are some special cases of matrices with displacement structures that can be decomposed into linear combinations of displacement matrices with a few items. The simplest case is banded Toeplitz matrices $\bar{\boldsymbol{T}}_n$, i.e., $t_k = 0$ for $|k| > \varrho$, where ϱ is a constant. The linear systems of banded Toeplitz matrices occur in many applications, involving the numerical solution of certain differential equations, the modeling of queueing problems, digital filtering, and so on.

Computing the Sylvester displacements of banded Toeplitz matrices, the matrices can be decomposed as

$$\begin{aligned} \bar{\boldsymbol{T}}_n = \tfrac{1}{2}\big[&2t_0\boldsymbol{I} + t_1\boldsymbol{Z}_1^1 + \cdots + t_\varrho \boldsymbol{Z}_1^\varrho \\ &+ t_{-\varrho}\boldsymbol{Z}_1^{n-\varrho} + \cdots + t_{-1}\boldsymbol{Z}_1^{n-1} \\ &+ t_1\boldsymbol{Z}_{-1}^1 + \cdots + t_\varrho \boldsymbol{Z}_{-1}^\varrho \\ &+ (-t_{-\varrho})\boldsymbol{Z}_{-1}^{n-\varrho} + \cdots + (-t_{-1})\boldsymbol{Z}_{-1}^{n-1}\big]. \end{aligned} \tag{36}$$

Similarly to Eqs. (14)–(16), we can define two state preparation operators $\boldsymbol{V}_{(\nabla[\bar{\boldsymbol{T}}])}$ and $\boldsymbol{V}_{(\nabla[\bar{\boldsymbol{T}}]^*)}$ and a controlled unitary

operator $\texttt{Select}(\boldsymbol{U}_{\bar{\boldsymbol{T}}})$. Then the unitaries $\boldsymbol{V}_{(\nabla[\bar{\boldsymbol{T}}])}$ and $\boldsymbol{V}_{(\nabla[\bar{\boldsymbol{T}}]^*)}$ can be implemented by using the generic state preparation algorithm described in [55], which requires a gate cost of $O(\varrho)$. Also, the controlled unitary operator $\texttt{Select}(\boldsymbol{U}_{\bar{\boldsymbol{T}}})$ can be implemented with $O(\varrho \log n)$ primitive gates, as the quantum circuit of each $\boldsymbol{Z}_1^j$ or $\boldsymbol{Z}_{-1}^j$ only requires $O(\log n)$ primitive gates. Thus, we can efficiently implement a $(\chi_{\bar{\boldsymbol{T}}_n}/2; \lceil \log(4\varrho + 1) \rceil; \epsilon)$-block-encoding of $\bar{\boldsymbol{T}}_n$, where $\chi_{\bar{\boldsymbol{T}}_n} = \sum_{j=-\varrho}^{\varrho} |t_j|$. It should be noted that the implementation scheme proposed here may further facilitate construction of practical circuits of the block-encodings of such matrices since it does not require any oracle or QRAM. Combined with the quantum algorithm for linear systems within the block-encoding framework, it can offer an exponential improvement in the dimension of the linear systems over classical methods.

In the black-box model, one can use the method of [23] to efficiently implement a block-encoding for a sparse matrix, where the scaling factor linearly depends on the sparsity (the maximum number of nonzero entries in any row or column) of the matrix. Since the structured matrices studied in this paper are not sparse, the scaling factor of this block-encoding will be $O(n)$. Then the quantum algorithm for structured linear systems based on such block-encoding cannot provide speedup compared with the classical algorithm. We note that [10] provided a way to improve the scaling factor; however, it required that the upper bound on the p-norm of the rows of the matrix is known, which is not the circumstance considered in this paper.

In the QRAM data structure model, the method stated in [24,26] also implements block-encodings based on the assumption that the powers of the elements of a matrix are stored in the quantum-accessible data structure beforehand. For a matrix with a displacement structure, the scaling factor produced by the method of [24,26] can be of the same order of magnitude as that in this paper. However, it should be noted that constructing the data structure that stores some entries about the matrices may constitute the main restriction of this data access model. With respect to this, our method is more advantageous. More specifically, the method of [24,26] would require a QRAM with data structure storing $O(n^2)$ entries for the matrices with displacement structures. In our method, since we represent these structured matrices with $O(n)$ entries of their displacements, a QRAM with data structure storing $O(n)$ entries is required. Obviously, the data structure in our method can be constructed more rapidly and uses less memory space, so our method will be more favorable when solving problems involving matrices with displacement structures.

As analyzed above, the origin of the advantages of our algorithm is the succinct parametrized representations of the matrices with displacement structures which are attributed to the Stein and Sylvester types of LCU decompositions. There are also some intuitive methods to perform an LCU decomposition. Typically, we specify a set of unitaries that are easy to implement as the basis and then calculate the decomposition coefficients by solving a linear system with n^2 unknown parameters. Alternatively, one can decompose the matrix into a sum of tensor products of Pauli operators, while the number of decomposition items will be considerably larger than ours. These decompositions are highly complex to construct and may not even fit to implement block-encodings. In general, it is not easy to find a desirable decomposition.

In particular, in this paper, by the proposed LCU decompositions, we make the implementation of block-encodings of matrices with displacement structure closely related to the task of preparing $O(n)$-dimensional quantum states. Since there are $\Omega(n)$ parameters when defining an $n \times n$ matrix with displacement structure, the query complexity in the black-box model and the memory cost in the QRAM data structure model are nearly optimal. It is an open question whether one can bypass the preparation of n-dimensional quantum states and implement block-encodings of the matrices with displacement structures with fewer resources.

It should be noted that although the proposed decompositions in this paper are also available for general dense matrices, the quantum algorithms based on these decompositions cannot significantly improve the known results due to the number of decomposition items of n^2. For the same reason, even if other unitaries are used as the basis, a universal LCU decomposition generally cannot give rise to quantum algorithms that surpass existing methods for all dense matrices. Nevertheless, it is still worth exploring specialized decomposition for some specific structured matrices to implement favorable block-encodings, which can result in a significant reduction in the complexity of the quantum linear system solver.

Following Tang's breakthrough work [56], there is a large class of classical algorithms whose running time is polylogarithmic in the dimension. However, this type of speedup is only achievable when the matrices involved are of low rank. It is not applicable to use the dequantization method to solve linear systems with displacement structures since these matrices are not low rank in general. In addition, these dequantized algorithms have higher overhead than the quantum algorithms in practice due to their large polynomial dependence on the rank and the other parameters. For the computation of displacement structured matrices, it is an interesting open problem to explore classical algorithms with similar overheads to quantum algorithms.

VI. CONCLUSION

In this paper we demonstrated that several important classes of matrices with displacement structures can be represented and treated similarly by decomposing them into linear combinations of displacement matrices. Based on the devised decompositions, we implemented block-encodings of these structured matrices in two different data access models and introduced efficient quantum algorithms for solving the linear system with such structures. The obtained quantum linear system solvers improved the known results and also motivated some other instances (see Table I for a brief summary). In particular, we provided a concrete example to illustrate that these quantum algorithms can be used to solve problems of practical interest with significant speedup.

The presented methods can actually be extended to solve many important computational problems having ties to the structured matrices studied in this paper, such as structured

TABLE I. Summary of quantum algorithms for solving linear systems with displacement structures.

Coefficient matrix	Algorithm	Remark	Comparison of complexity
Toeplitz matrices in the Wiener class	Wan *et al.* [22]	Asymptotic algorithm	Improve the dependence on the condition number and precision when the generating function is unknown
	Theorem 2 based	Nonasymptotic algorithm	
Circulant matrices with bounded spectral norm	Zhou and Wang [21]	An oracle of preparing the n-dimensional state is required	Improve the dependence on the condition number and precision when using the same data access model
	Theorem 2 based	Two different data access models	
Discretized Laplacian (specific banded Toeplitz matrices)	Cao *et al.* [19]	Finite-difference discretization of the Poisson equation	Same order of magnitude
	Discussion based	Applicable to general banded Toeplitz matrices	
Toeplitz- or Hankel-like matrices with small χ	No specialized quantum algorithm		
	Corollary 1 based		

least-squares problems and computation of the structured matrices exponential. Also, we hope our work can inspire the study of matrices with displacement structures on near-term quantum devices (for example, see [57]) since we have shown that these matrices can be decomposed into some easy-to-implement unitaries. Finally, there are many other structured matrices such as Cauchy, Bezout, Vandermonde, Loewner, and Pick matrices which are widely employed in various areas. Designing quantum algorithms for these structured matrices with a dramatic computational acceleration and a major memory-space decrease is worthy of further study.

ACKNOWLEDGMENTS

The authors would like to thank Yi-Jie Shi for her constructive comments on an early version of this paper. This work was supported by the Fundamental Research Funds for the Central Universities (Grant No. 2019XD-A01), the National Natural Science Foundation of China (Grants No. 61972048, No. 61976024, and No. 62006105), the Jiangxi Provincial Natural Science Foundation (Grant No. 20202BABL212004), and the China Scholarship Council (Grant No. 201806470057).

APPENDIX A: PROOF OF THEOREM 1

In this Appendix we prove the conclusion in Theorem 1. There are some well-known fundamental results and the proof of these results can be found in [17]. For completeness, we restate them here.

Lemma 3 (from [17]). For matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{M} \in \mathbb{C}^{n\times n}$ and $k \geqslant 1$ we have

$$\boldsymbol{M} = \boldsymbol{A}^k \boldsymbol{M} \boldsymbol{B}^k + \sum_{i=0}^{k-1} \boldsymbol{A}^i \Delta_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{M}) \boldsymbol{B}^i. \tag{A1}$$

Proof. It is trivial when $k = 1$. We assume the identity is true for k. Then, multiplying the identity on the left by $\boldsymbol{A}$ and right by $\boldsymbol{B}$, we obtain

$$\begin{aligned} \boldsymbol{A}\boldsymbol{M}\boldsymbol{B} &= \boldsymbol{A}^{k+1}\boldsymbol{M}\boldsymbol{B}^{k+1} + \sum_{i=0}^{k-1} \boldsymbol{A}^{i+1} \Delta_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{M}) \boldsymbol{B}^{i+1} \\ &= \boldsymbol{A}^{k+1}\boldsymbol{M}\boldsymbol{B}^{k+1} + \sum_{i=0}^{k} \boldsymbol{A}^{i} \Delta_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{M}) \boldsymbol{B}^{i} - \Delta_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{M}), \\ \boldsymbol{M} &= \boldsymbol{A}^{k+1}\boldsymbol{M}\boldsymbol{B}^{k+1} + \sum_{i=0}^{k} \boldsymbol{A}^{i} \Delta_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{M}) \boldsymbol{B}^{i}. \end{aligned} \tag{A2}$$

Thus, the identity is true for $k + 1$. According to mathematical induction, it is true for all natural numbers. ■

Lemma 4 (from [17]). If $\boldsymbol{A}$ is an a-potent matrix of order n and $\boldsymbol{B}$ is a b-potent matrix of order n, i.e., $\boldsymbol{A}^n = a\boldsymbol{I}$ and $\boldsymbol{B}^n = b\boldsymbol{I}$, then

$$\boldsymbol{M} = \frac{1}{1-ab} \sum_{i=0}^{n-1} \boldsymbol{A}^i \Delta_{\boldsymbol{A},\boldsymbol{B}}(\boldsymbol{M}) \boldsymbol{B}^i. \tag{A3}$$

Proof. This conclusion is a direct inference of Lemma 3, when $k = n, \boldsymbol{A}^n = a\boldsymbol{I}$, and $\boldsymbol{B}^n = b\boldsymbol{I}$. ■

Lemma 5 (from [17]). If the operator matrix $\boldsymbol{A}$ is nonsingular, then $\nabla_{\boldsymbol{A},\boldsymbol{B}} = \boldsymbol{A}\Delta_{\boldsymbol{A}^{-1},\boldsymbol{B}}$; if the operator matrix $\boldsymbol{B}$ is nonsingular, then $\nabla_{\boldsymbol{A},\boldsymbol{B}} = -\Delta_{\boldsymbol{A},\boldsymbol{B}^{-1}}\boldsymbol{B}$.

Proof. Note that if $\boldsymbol{A}$ is nonsingular, then $\boldsymbol{A}\boldsymbol{M} - \boldsymbol{M}\boldsymbol{B} = \boldsymbol{A}(\boldsymbol{M} - \boldsymbol{A}^{-1}\boldsymbol{M}\boldsymbol{B})$; if $\boldsymbol{B}$ is nonsingular, then $\boldsymbol{A}\boldsymbol{M} - \boldsymbol{M}\boldsymbol{B} = -(\boldsymbol{M} - \boldsymbol{A}\boldsymbol{M}\boldsymbol{B}^{-1})\boldsymbol{B}$. ■

Definition 4 (the f-circulant matrix). The f-circulant matrix $\boldsymbol{Z}_f(\boldsymbol{v})$ generated by a unit f-circulant matrix and a given

vector $\boldsymbol{v} = [v_0, \ldots, v_{n-1}]^T$ is defined as follows:

$$\begin{aligned}\boldsymbol{Z}_f(\boldsymbol{v}) &= \left(\boldsymbol{v}\, Z_f \boldsymbol{v}\, Z_f^2 \boldsymbol{v} \cdots \boldsymbol{Z}_f^{n-1}\boldsymbol{v}\right) \\ &= \begin{pmatrix} v_0 & f v_{n-1} & \cdots & f v_1 \\ v_1 & v_0 & \cdots & f v_2 \\ \vdots & \vdots & \vdots & f v_{n-1} \\ v_{n-1} & \cdots & v_1 & v_0 \end{pmatrix}. \end{aligned} \tag{A4}$$

It turns out that a matrix $\boldsymbol{M}$ can be expressed as the sum of the products of f-circulant matrices and the reversal matrix, by inverting the displacement operators.

Theorem 3 (from [17]). If a matrix $\boldsymbol{M} \in \mathbb{C}^{n\times n}$ satisfies $\mathcal{L}(\boldsymbol{M}) = \boldsymbol{G}\boldsymbol{H}^T$, where $\boldsymbol{G} = [\boldsymbol{g}_1 \cdots \boldsymbol{g}_r]$, $\boldsymbol{H} = [\boldsymbol{h}_1 \cdots \boldsymbol{h}_r] \in \mathbb{C}^{n\times r}$, and e and f are constants, then $\boldsymbol{M}$ can be expressed as (i)

$$\boldsymbol{M} = \frac{1}{1-ef}\sum_{j=1}^{r} \boldsymbol{Z}_e(\boldsymbol{g}_j)\boldsymbol{Z}_f(\boldsymbol{J}\boldsymbol{h}_j)^T\boldsymbol{J}, \tag{A5}$$

where $\mathcal{L}(\boldsymbol{M}) = \Delta_{\boldsymbol{Z}_e,\boldsymbol{Z}_f}[\boldsymbol{M}]$ and $ef \neq 1$, and (ii)

$$\boldsymbol{M} = \frac{1}{e-f}\sum_{j=1}^{r} \boldsymbol{Z}_e(\boldsymbol{g}_j)\boldsymbol{Z}_f(\boldsymbol{J}\boldsymbol{h}_j), \tag{A6}$$

where $\mathcal{L}(\boldsymbol{M}) = \nabla_{\boldsymbol{Z}_e,\boldsymbol{Z}_f}[\boldsymbol{M}]$ and $e \neq f$.

Proof. From Lemma 4, let $\boldsymbol{A} = \boldsymbol{Z}_e$, $\boldsymbol{B} = \boldsymbol{Z}_f$, and $ef \neq 1$. Then we have

$$\begin{aligned}\boldsymbol{M} &= \frac{1}{1-ef}\sum_{i=0}^{n-1} \boldsymbol{Z}_e^i \Delta_{\boldsymbol{Z}_e,\boldsymbol{Z}_f}(\boldsymbol{M})\boldsymbol{Z}_f^i \\ &= \frac{1}{1-ef}\sum_{j=1}^{r}\sum_{i=0}^{n-1} \boldsymbol{Z}_e^i \boldsymbol{g}_j \boldsymbol{h}_j^T \boldsymbol{Z}_f^i \\ &= \frac{1}{1-ef}\sum_{j=1}^{r} \left(\boldsymbol{g}_j\boldsymbol{h}_j^T + \boldsymbol{Z}_e\boldsymbol{g}_j\boldsymbol{h}_j^T\boldsymbol{Z}_f + \boldsymbol{Z}_e^2\boldsymbol{g}_j\boldsymbol{h}_j^T\boldsymbol{Z}_f^2 \right. \\ &\quad \left. + \cdots + \boldsymbol{Z}_e^{n-1}\boldsymbol{g}_j\boldsymbol{h}_j^T\boldsymbol{Z}_f^{n-1}\right) \\ &= \frac{1}{1-ef}\sum_{j=1}^{r} \left[\boldsymbol{g}_j\, \boldsymbol{Z}_e\boldsymbol{g}_j\, \boldsymbol{Z}_e^2\boldsymbol{g}_j \cdots \boldsymbol{Z}_e^{n-1}\boldsymbol{g}_j\right] \\ &\quad \times \left[\boldsymbol{h}_j\, \boldsymbol{Z}_f^T\boldsymbol{h}_j\, \left(\boldsymbol{Z}_f^T\right)^2\boldsymbol{h}_j \cdots \left(\boldsymbol{Z}_f^T\right)^{n-1}\boldsymbol{h}_j\right]^T \\ &= \frac{1}{1-ef}\sum_{j=1}^{r} \boldsymbol{Z}_e(\boldsymbol{g}_j)\big[\boldsymbol{J}\boldsymbol{J}\boldsymbol{h}_j\, \boldsymbol{J}\boldsymbol{Z}_f\boldsymbol{J}\boldsymbol{h}_j \\ &\quad \boldsymbol{J}(\boldsymbol{Z}_f)^2\boldsymbol{J}\boldsymbol{h}_j \cdots \boldsymbol{J}(\boldsymbol{Z}_f)^{n-1}\boldsymbol{J}\boldsymbol{h}_j\big]^T \\ &= \frac{1}{1-ef}\sum_{j=1}^{r} \boldsymbol{Z}_e(\boldsymbol{g}_j)[\boldsymbol{J}\cdot\boldsymbol{Z}_f(\boldsymbol{J}\boldsymbol{h}_j)]^T \\ &= \frac{1}{1-ef}\sum_{j=1}^{r} \boldsymbol{Z}_e(\boldsymbol{g}_j)\boldsymbol{Z}_f(\boldsymbol{J}\boldsymbol{h}_j)^T\boldsymbol{J} \end{aligned} \tag{A7}$$

by using the facts $\boldsymbol{J}^2 = \boldsymbol{I}$ and $\boldsymbol{Z}_f = \boldsymbol{J}\boldsymbol{Z}_f^T\boldsymbol{J}$.

Furthermore, according to Lemma 5, $\Delta_{\boldsymbol{Z}_{1/e}^T,\boldsymbol{Z}_f}[\boldsymbol{M}] = \boldsymbol{Z}_{1/e}^T \nabla_{\boldsymbol{Z}_e,\boldsymbol{Z}_f}[\boldsymbol{M}]$, where $\boldsymbol{Z}_f^{-1} = \boldsymbol{Z}_{1/f}^T$. Then we can deduce the conclusion for the Sylvester type. ■

Now we demonstrate how to decompose an $n \times n$ matrix into linear combinations of unitaries. For our purposes, we choose $(\boldsymbol{Z}_1, \boldsymbol{Z}_{-1})$ as the operator matrices. Note that

$$\begin{aligned}\boldsymbol{Z}_1(\boldsymbol{g}_j) &= \begin{pmatrix} g_j^0 & g_j^{n-1} & \cdots & g_j^1 \\ g_j^1 & g_j^0 & \cdots & g_j^2 \\ \vdots & \ddots & \ddots & \vdots \\ g_j^{n-1} & \cdots & g_j^1 & g_j^0 \end{pmatrix} \\ &= g_j^0\boldsymbol{Z}_1^0 + g_j^{n-1}\boldsymbol{Z}_1^{n-1} + \cdots + g_j^1\boldsymbol{Z}_1^1, \end{aligned} \tag{A8}$$

$$\begin{aligned}\boldsymbol{Z}_{-1}(\boldsymbol{h}_j) &= \begin{pmatrix} h_j^0 & -h_j^{n-1} & \cdots & -h_j^1 \\ h_j^1 & h_j^0 & \cdots & -h_j^2 \\ \vdots & \ddots & \ddots & \vdots \\ h_j^{n-1} & \cdots & h_j^1 & h_j^0 \end{pmatrix} \\ &= h_j^0\boldsymbol{Z}_{-1}^0 + h_j^{n-1}\boldsymbol{Z}_{-1}^{n-1} + \cdots + h_j^1\boldsymbol{Z}_{-1}^1, \end{aligned} \tag{A9}$$

where

$$\boldsymbol{g}_j = \left(g_j^0, g_j^1, \ldots, g_j^{n-1}\right)^T, \quad \boldsymbol{h}_j = \left(h_j^0, h_j^1, \ldots, h_j^{n-1}\right)^T. \tag{A10}$$

Then, on the one hand,

$$\begin{aligned}\boldsymbol{M} &= \frac{1}{2}\sum_{j=1}^{r} \boldsymbol{Z}_1(\boldsymbol{g}_j)\boldsymbol{Z}_{-1}(\boldsymbol{J}\boldsymbol{h}_j) \\ &= \frac{1}{2}\sum_{j=1}^{r} \left(g_j^0\boldsymbol{Z}_1^0 + g_j^1\boldsymbol{Z}_1^1 + \cdots + g_j^{n-1}\boldsymbol{Z}_1^{n-1}\right) \\ &\quad \times \left(h_j^{n-1}\boldsymbol{Z}_{-1}^0 + h_j^{n-2}\boldsymbol{Z}_{-1}^1 + \cdots + h_j^0\boldsymbol{Z}_{-1}^{n-1}\right) \\ &= \frac{1}{2}\sum_{j=1}^{r} \left(g_j^0 h_j^{n-1}\boldsymbol{Z}_1^0\boldsymbol{Z}_{-1}^0 + g_j^0 h_j^{n-2}\boldsymbol{Z}_1^0\boldsymbol{Z}_{-1}^1 \right. \\ &\quad + \cdots + g_j^0 h_j^0 \boldsymbol{Z}_1^0\boldsymbol{Z}_{-1}^{n-1} + g_j^1 h_j^{n-1}\boldsymbol{Z}_1^1\boldsymbol{Z}_{-1}^0 \\ &\quad + g_j^1 h_j^{n-2}\boldsymbol{Z}_1^1\boldsymbol{Z}_{-1}^1 + \cdots + g_j^1 h_j^0 \boldsymbol{Z}_1^1\boldsymbol{Z}_{-1}^{n-1} + \cdots \\ &\quad + g_j^{n-1} h_j^{n-1}\boldsymbol{Z}_1^{n-1}\boldsymbol{Z}_{-1}^0 + g_j^{n-1} h_j^{n-2}\boldsymbol{Z}_1^{n-1}\boldsymbol{Z}_{-1}^1 \\ &\quad \left. + \cdots + g_j^{n-1} h_j^0 \boldsymbol{Z}_1^{n-1}\boldsymbol{Z}_{-1}^{n-1}\right) \\ &= \frac{1}{2}\sum_{j=1}^{r}\sum_{i,k=0}^{n-1} g_j^i h_j^k \boldsymbol{Z}_1^i \boldsymbol{Z}_{-1}^{n-1-k} \\ &= \frac{1}{2}\sum_{i,k=0}^{n-1}\sum_{j=1}^{r} g_j^i h_j^k \boldsymbol{Z}_1^i \boldsymbol{Z}_{-1}^{n-1-k}. \end{aligned} \tag{A11}$$

On the other hand, since

$$\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{M}] = \boldsymbol{G}\boldsymbol{H}^T = \sum_{j=1}^{r} \boldsymbol{g}_j\boldsymbol{h}_j^T, \tag{A12}$$

it is immediately verified that

$$\tilde{m}_{i,k} = \sum_{j=1}^{r} g_j^i h_j^k, \quad i,k = 0, 1, \ldots, n-1, \tag{A13}$$

where $\tilde{m}_{i,k}$ is the kth element of the ith row of matrix $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{M}]$. Therefore,

$$\boldsymbol{M} = \frac{1}{2}\sum_{i,k=0}^{n-1} \tilde{m}_{i,k}\boldsymbol{Z}_1^i \boldsymbol{Z}_{-1}^{n-1-k}. \tag{A14}$$

The decomposition for the Stein type can be proved in the same way.

APPENDIX B: STEERABLE BLACK-BOX QUANTUM STATE PREPARATION

The scenario for steerable black-box state preparation is as follows. For a vector $\boldsymbol{x} = (x_0, x_1, \ldots, x_{n-1})^T$, we are provided a black box that returns target elements, i.e.,

$$\mathcal{O}_x|i\rangle|0\rangle \to |i\rangle|x_i\rangle.$$

The task is to prepare a quantum state that is ϵ-close to

$$|x\rangle = \frac{1}{\sqrt{\|\boldsymbol{x}\|_1}}\sum_{i=0}^{n-1}\sqrt{x_i}|i\rangle$$

with an adjustable bound on the success probability $1-\delta^2$. The quantum algorithm for preparing this state contains two steps.

(1) Prepare the initial state. (a) Start with a uniform superposition state and perform the black box to have

$$\sum_{i=0}^{n-1}\frac{1}{\sqrt{n}}|i\rangle_1|x_i\rangle_a. \tag{B1}$$

(b) Add a qubit and perform controlled rotation to yield

$$\sum_{i=0}^{n-1}\frac{1}{\sqrt{n}}|i\rangle_1|x_i\rangle_a\left(\frac{\sqrt{x_i}}{|\boldsymbol{x}|_{\max}}|0\rangle_2 + \sqrt{1-\frac{|x_i|}{|\boldsymbol{x}|_{\max}^2}}|1\rangle_2\right), \tag{B2}$$

where $|\boldsymbol{x}|_{\max} = \max_i|\sqrt{x_i}|$ is a small constant. (c) Uncompute the black box to obtain

$$\sum_{i=0}^{n-1}\frac{1}{\sqrt{n}}|i\rangle_1\left(\frac{\sqrt{x_i}}{|\boldsymbol{x}|_{\max}}|0\rangle_2 + \sqrt{1-\frac{|x_i|}{|\boldsymbol{x}|_{\max}^2}}|1\rangle_2\right). \tag{B3}$$

Denoting this state by $|\psi\rangle_{1,2}$, this step can be regarded as a unitary operator $\boldsymbol{U}_a$ such that $\boldsymbol{U}_a|0\rangle_{1,2} = |\psi\rangle_{1,2}$. Note that $|\psi\rangle_{1,2}$ can be rewritten as

$$|\psi\rangle_{1,2} = \sqrt{P_0}|\alpha\rangle_{1,2} + \sqrt{1-P_0}|\beta\rangle_{1,2}, \tag{B4}$$

where

$$P_0 = \frac{\|\boldsymbol{x}\|_1}{n|\boldsymbol{x}|_{\max}^2}, \tag{B5}$$

$$|\alpha\rangle_{1,2} = \sum_{i=0}^{n-1}\frac{\sqrt{x_i}}{\sqrt{\|\boldsymbol{x}\|_1}}|i\rangle_1|0\rangle_2, \tag{B6}$$

$$|\beta\rangle_{1,2} = \sum_{i=0}^{n-1}\frac{\sqrt{1-\frac{|x_i|}{|\boldsymbol{x}|_{\max}^2}}}{\Upsilon}|i\rangle_1|1\rangle_2, \tag{B7}$$

$$\Upsilon = \sqrt{\sum_{i=0}^{n-1}\left|1-\frac{|x_i|}{|\boldsymbol{x}|_{\max}^2}\right|}. \tag{B8}$$

(2) Amplify the amplitude of getting $|\alpha\rangle$. In this step, we apply the fixed-point quantum search algorithm proposed in [44] to amplify the success probability with an adjustable bound. More specifically, define conditional phase shift operators

$$\boldsymbol{S}_t^{\varphi} = \boldsymbol{I}_{1,2} + (e^{i\varphi}-1)\boldsymbol{I}_1 \otimes |0\rangle\langle 0|_2 \tag{B9}$$

and

$$\boldsymbol{S}_a^{\phi} = \boldsymbol{I}_{1,2} + (e^{i\phi}-1)|0\rangle\langle 0|_{1,2}. \tag{B10}$$

The algorithm performs the sequence of the generalized Grover operator

$$\boldsymbol{G}(\phi_l,\varphi_l)\boldsymbol{G}(\phi_{l-1},\varphi_{l-1})\cdots\boldsymbol{G}(\phi_1,\varphi_1), \tag{B11}$$

where $\boldsymbol{G}(\phi_j,\varphi_j) = -\boldsymbol{U}_a\boldsymbol{S}_a^{\phi_j}\boldsymbol{U}_a^{\dagger}\boldsymbol{S}_t^{\varphi_j}$. The condition on the phases $\{\varphi_j,\phi_j, 1\leqslant j\leqslant l\}$ was indicated in [44],

$$\phi_j = \varphi_{l-j+1} = -2\,\mathrm{arccot}[\sqrt{1-\gamma^2}\tan(2\pi\, j/L)], \tag{B12}$$

where $L = 2l+1$, $\gamma = T_{1/L}^{-1}(1/\delta)$, $\delta\in(0,1)$, and $T_L(x)$ is the Lth Chebyshev polynomial of the first kind. After l iterations, the final state, up to a global phase, will be

$$|\psi_l\rangle = \sqrt{P_L}|\alpha\rangle + \sqrt{1-P_L}|\beta\rangle, \tag{B13}$$

where $P_L = 1-\delta^2 T_L^2[T_{1/L}(1/\delta)\sqrt{1-P_0}]$ is the success probability. It was shown that for a given δ and a known lower bound $P_{\min}$ of P_0, the condition of L:

$$L \geqslant \frac{\log(2/\delta)}{\sqrt{P_{\min}}}, \tag{B14}$$

can ensure $P_L \geqslant 1-\delta^2$.

We now show that the $P_{\min}$ can be provided by using amplitude estimation. More specifically, for any $\epsilon_0 > 0$, amplitude estimation can approximate the probability P_0 up to an additive error $P_0\epsilon_0$ with $O(1/\epsilon_0\sqrt{P_0})$ uses of the standard Grover operator. Let the output of amplitude estimation be P_0'; then $P_0 \geqslant \frac{P_0'}{1+\epsilon_0}$. Thus, we can take $\frac{P_0'}{1+\epsilon_0}$ as a low bound of P_0. Note that ϵ_0 is the relative error of estimated P_0 and we only need the low bound of P_0, so we can set ϵ_0 to be a small constant like $\frac{1}{2}$. Then the query complexity of this step is $O(\sqrt{n}/\sqrt{\|\boldsymbol{x}\|_1})$.

Reviewing the cost of amplitude estimation and fixed-point amplitude amplification, we now analyze the complexity of this approach. Clearly, the query complexity is $O(\frac{\log(1/\delta)\sqrt{n}}{\sqrt{\|\boldsymbol{x}\|_1}})$. The gate complexity of this approach is dominated by the gate complexity of fixed-point amplitude amplification, which is given by the gate complexity of the generalized Grover operator multiplied by the number of iterations. The conditional phase shift operators $\boldsymbol{S}_a^{\phi_j}$ and $\boldsymbol{S}_t^{\varphi_j}$ can be implemented by using $O(\log n)$ elementary gates. The gate complexity of $\boldsymbol{U}_a$ depends on the precision of controlled rotation. Note that performing the controlled rotation with error ϵ_r will generate a state that is $O(\sqrt{n}\epsilon_r)$-close to $|\psi\rangle$. Then, after performing the fixed-point amplitude amplification, we can get a state that is $O(\sqrt{n}\epsilon_r/\sqrt{P_0})$-close to $|\psi_l\rangle$, since the error is also amplified by the fixed-point amplitude amplification. To obtain overall error $O(\epsilon_p)$, ϵ_r should be $O(\epsilon_p\sqrt{P_0}/\sqrt{n})$. Also it is shown that the controlled rotation can be performed with error ϵ_r using $O(\mathrm{polylog}(\frac{1}{\epsilon_r}))$ elementary gates. Thus, the gate complexity of $\boldsymbol{U}_a$ is $O(\mathrm{polylog}(\frac{\sqrt{n}}{\epsilon_p\sqrt{P_0}}))$. Putting these all together, we can

obtain a state, up to a global phase, that is ϵ_p-close to $|x\rangle$ with a success probability of at least $1-\delta^2$, using $O(\frac{\sqrt{n}\log(1/\delta)}{\sqrt{\|x\|_1}})$ queries of $\mathcal{O}_x$ and $O(\frac{\sqrt{n}\log(1/\delta)}{\sqrt{\|x\|_1}}\mathrm{polylog}(\frac{n}{\epsilon_p\sqrt{\|x\|_1}}))$ elementary gates.

APPENDIX C: PROOF OF THEOREM 2

Now we show how to implement the block-encoding of $\boldsymbol{T}_n$, i.e., the quantum state preparation operators and the controlled unitary, in the two data access models introduced in Sec. II C.

1. Black-box model

In the black-box model, we are given the black box

$$\mathcal{O}_{\boldsymbol{T}_n}|i\rangle|k\rangle|0\rangle = |i\rangle|k\rangle|t_{i,k}\rangle, \quad i,k=0,1,\ldots,n-1.$$

Then we can construct black box $\mathcal{O}_1$ satisfying

$$\mathcal{O}_1|j\rangle|0\rangle = |j\rangle|t_j + t_{-[(n-j)\mathrm{mod}n]}\rangle, \quad j=0,1,\ldots,n-1,$$

by using the black box $\mathcal{O}_{\boldsymbol{T}_n}$ twice to query the elements in the sites $|j,0\rangle$ and $|0,(n-j)\mathrm{mod}n\rangle$ and following an addition operation [58,59]. More specifically, $\mathcal{O}_1$ can be constructed as follows.

(1) Compute the index of the site by using X gates and the quantum modular subtractor [58,59]:

$$\begin{aligned}&|j\rangle_{a_1}|0\rangle_{a_2}|0\rangle_{b_1}|0\rangle_{b_2}|0\rangle_{a_3}|0\rangle_{b_3}\\ &\quad\rightarrow |j\rangle_{a_1}|0\rangle_{a_2}|0\rangle_{b_1}|(n-j)\mathrm{mod}n\rangle_{b_2}|0\rangle_{a_3}|0\rangle_{b_3}.\end{aligned} \tag{C1}$$

(2) Perform $\mathcal{O}_{\boldsymbol{T}_n}$ on registers $\{a_1,a_2,a_3\}$ and $\{b_1,b_2,b_3\}$, respectively:

$$\begin{aligned}&|j\rangle_{a_1}|0\rangle_{a_2}|0\rangle_{b_1}|(n-j)\mathrm{mod}n\rangle_{b_2}|0\rangle_{a_3}|0\rangle_{b_3}\\ &\quad\rightarrow |j\rangle_{a_1}|0\rangle_{a_2}|0\rangle_{b_1}|(n-j)\mathrm{mod}n\rangle_{b_2}|t_j\rangle_{a_3}|t_{-[(n-j)\mathrm{mod}n]}\rangle_{b_3}.\end{aligned} \tag{C2}$$

(3) Perform quantum addition operation on registers $\{a_3,b_3\}$ to yield

$$|j\rangle_{a_1}|0\rangle_{a_2}|0\rangle_{b_1}|(n-j)\mathrm{mod}n\rangle_{b_2}|t_j\rangle_{a_3}|t_j + t_{-[(n-j)\mathrm{mod}n]}\rangle_{b_3}. \tag{C3}$$

(4) Reverse the computation on registers $\{a_3,b_2\}$:

$$|j\rangle_{a_1}|0\rangle_{a_2}|0\rangle_{b_1}|0\rangle_{b_2}|0\rangle_{a_3}|t_j + t_{-[(n-j)\mathrm{mod}n]}\rangle_{b_3}. \tag{C4}$$

The mapping on registers a_1 and b_3 is actually $\mathcal{O}_1$.

Similarly, we can construct the black box $\mathcal{O}_2$ satisfying

$$\mathcal{O}_2|j\rangle|0\rangle = |j\rangle|t_j - t_{-[(n-j)\mathrm{mod}n]}\rangle, \quad j=0,1,\ldots,n-1. \tag{C5}$$

Since the gates required for constructing $\mathcal{O}_1$ and $\mathcal{O}_2$ are negligible compared to other subroutines of the quantum algorithm, we did not consider their complexity in this paper. With these two black boxes $\mathcal{O}_1$ and $\mathcal{O}_2$, it is feasible to generate a controlled black box $\mathcal{O}_{1\wedge 2}$ of the form

$$|0\rangle\langle 0|\otimes\mathcal{O}_1 + |1\rangle\langle 1|\otimes\mathcal{O}_2. \tag{C6}$$

Note the black box $\mathcal{O}_{1\wedge 2}$ query in superposition acting as

$$\mathcal{O}_{1\wedge 2}\frac{1}{\sqrt{2n}}\Big(\sum_{j=0}^{n-1}|0\rangle|j\rangle|0\rangle + \sum_{j=0}^{n-1}|1\rangle|j\rangle|0\rangle\Big) = \frac{1}{\sqrt{2n}}\sum_{j=0}^{2n-1}|j\rangle|\tilde{t}_j\rangle. \tag{C7}$$

Thus, using this black box, we can approximatively implement $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n])}$ by the steerable black-box quantum state preparation algorithm. Similarly, we can approximatively implement $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n]^*)}$ by constructing a black box $\mathcal{O}^*_{1\wedge 2}$ that returns $\tilde{t}^*_j$.

When implementing $\mathtt{Select}\boldsymbol{U}_{T_n}$, directly using the controlled circuit may take $O(n\log n)$ elementary gates. To make the implementation more efficient, we use an idea similar to that in Ref. [21]. More specifically, note that

$$\boldsymbol{Z}_1^j = \sum_{a=0}^{n-1}|(a+j)\mathrm{mod}n\rangle\langle a|, \tag{C8}$$

$$\begin{aligned}\boldsymbol{Z}_{-1}^j = &\left(\sum_{b=0}^{n-1}|(b+j)\mathrm{mod}n\rangle\langle b|\right)\\ &\times\left(\sum_{b=0}^{n-1-j}|b\rangle\langle b| - \sum_{b=n-j}^{n-1}|b\rangle\langle b|\right).\end{aligned} \tag{C9}$$

Thus, the action of $\mathtt{Select}\boldsymbol{U}_{T_n}$ on the basis states is

$$\mathtt{Select}\,\boldsymbol{U}_{T_n}|j\rangle|e\rangle = \begin{cases}|j\rangle|(e+j)\mathrm{mod}n\rangle, & 0\leqslant j\leqslant n-1,\ 0\leqslant e\leqslant n-1\\ |j\rangle|(e+j)\mathrm{mod}n\rangle, & n\leqslant j\leqslant 2n-1,\ 0\leqslant e\leqslant 2n-1-j\\ -|j\rangle|(e+j)\mathrm{mod}n\rangle, & n\leqslant j\leqslant 2n-1,\ 2n-j\leqslant e\leqslant n-1.\end{cases} \tag{C10}$$

Then, on the one hand, let

$$f_1(j,e) = \begin{cases}0, & 0\leqslant j\leqslant n-1,\ 0\leqslant e\leqslant n-1\\ 0, & n\leqslant j\leqslant 2n-1,\ 0\leqslant e\leqslant 2n-1-j\\ 1, & n\leqslant j\leqslant 2n-1,\ 2n-j\leqslant e\leqslant n-1.\end{cases} \tag{C11}$$

To compute this classical function with a quantum circuit, we use the quantum comparator [58,59] which comprises $O(\log n)$ elementary gates. Suppose that a and b are two natural numbers. Then the quantum comparator outputs the result c of the comparison of the two numbers, i.e., if $b\geqslant a$, $c=0$; otherwise, $c=1$. The specific quantum circuit of $\boldsymbol{U}_{f_1}$ is as follows.

(i) Prepare an initial quantum state

$$\begin{aligned}&|j\rangle_{a_1}|e\rangle_{a_2}|0\rangle_{b_1}|0\rangle_{b_2}|0\rangle_{c_1}|0\rangle_{c_2}|-\rangle_{c_3}\\ &\quad\rightarrow |j\rangle_{a_1}|e\rangle_{a_2}|n-1\rangle_{b_1}|2n-1-j\rangle_{b_2}|0\rangle_{c_1}|0\rangle_{c_2}|-\rangle_{c_3}.\end{aligned}$$

(ii) Perform a quantum comparator on registers $\{a_1, b_1, c_1\}$ and $\{a_2, b_2, c_2\}$, respectively.

(iii) Perform a Toffoli gate on the registers $\{c_1, c_2, c_3\}$.

(iv) Reverse the computation on registers $\{c_2, c_1, b_2, b_1\}$ in order.

The mapping on registers a_1, a_2, and c_3 is

$$\boldsymbol{U}_{f_1}|j\rangle|e\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}=(-1)^{f_1(j,e)}|j\rangle|e\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}. \tag{C12}$$

On the other hand, using a quantum modular adder [58,59], which requires $O(\log n)$ elementary gates, we can implement

$$\boldsymbol{U}_{\text{add1}}|j\rangle|e\rangle=|j\rangle|(e+j)\text{mod}n\rangle. \tag{C13}$$

Therefore, by using $\boldsymbol{U}_{\text{add1}}$ and $\boldsymbol{U}_{f_1}$, we can implement $\texttt{Select}\boldsymbol{U}_{T_n}$ equivalently (due to linearity, the implementation is available for any state). In summary, $\texttt{Select}\boldsymbol{U}_{T_n}$ can be implemented in time $O(\text{polylog}n)$.

Now we analyze the error incurred due to imperfect state preparation. Let $\boldsymbol{U}_{\tilde{V}}$ and $\boldsymbol{U}_{\tilde{V}^*}$ be unitaries that perform the steerable black-box quantum state preparation with the black boxes $\mathcal{O}_{1\wedge 2}$ and $\mathcal{O}^*_{1\wedge 2}$. Define $\boldsymbol{U}_V$ and $\boldsymbol{U}_{V^*}$ as

$$\boldsymbol{U}_V|0\rangle_{a_1}|0\rangle_{a_2}=\sqrt{1-\delta^2}|V_{(\nabla[\boldsymbol{T}_n])}\rangle_{a_1}|0\rangle_{a_2}+\delta|\zeta\rangle_{a_1}|1\rangle_{a_2}, \tag{C14}$$

$$\boldsymbol{U}_{V^*}|0\rangle_{a_1}|0\rangle_{a_2}=\sqrt{1-\delta^2}|V_{(\nabla[\boldsymbol{T}_n]^*)}\rangle_{a_1}|0\rangle_{a_2}+\delta|\zeta\rangle_{a_1}|1\rangle_{a_2}, \tag{C15}$$

where $|\zeta\rangle=\sum_{j=0}^{2n-1}\zeta_j|j\rangle$ is a quantum state similar in form to Eq. (B7). Obviously, $\boldsymbol{U}_{\tilde{V}}$ and $\boldsymbol{U}_{\tilde{V}^*}$ are ϵ_p approximations of $\boldsymbol{U}_V$ and $\boldsymbol{U}_{V^*}$, respectively. To simplify the notation, we rewrite Toeplitz matrices as $\boldsymbol{T}_n=\frac{1}{2}\sum_{j=0}^{2n-1}\tilde{t}_j\boldsymbol{U}_j$ by using the symbol $\boldsymbol{U}_j$ to represent $\boldsymbol{Z}_1^i$ and $\boldsymbol{Z}_{-1}^i$ in the order of Eq. (13). Similarly, the operator defined by Eq. (16) can be rewritten as $\texttt{Select}\boldsymbol{U}=\sum_{j=0}^{2n-1}|j\rangle\langle j|_{a_1}\otimes\boldsymbol{U}_j$, where $\boldsymbol{U}_j$ perform on the register s. Let $|0\rangle_a\equiv|0\rangle_{a_1}|0\rangle_{a_2}$, we have

$$\begin{aligned}\left\|\boldsymbol{M}-\frac{\chi}{2}(\langle 0|_a\boldsymbol{U}_{\tilde{V}^*}^\dagger\otimes\boldsymbol{I}_s)(\texttt{Select}\boldsymbol{U}\otimes\boldsymbol{I}_{a_2})(\boldsymbol{U}_{\tilde{V}}|0\rangle_a\otimes\boldsymbol{I}_s)\right\|&\leqslant\left\|\boldsymbol{M}-\frac{\chi}{2}(\langle 0|_a\boldsymbol{U}_{V^*}^\dagger\otimes\boldsymbol{I}_s)(\texttt{Select}\boldsymbol{U}\otimes\boldsymbol{I}_{a_2})(\boldsymbol{U}_V|0\rangle_a\otimes\boldsymbol{I}_s)\right\|\\&+\left\|\frac{\chi}{2}(\langle 0|_a\boldsymbol{U}_{V^*}^\dagger\otimes\boldsymbol{I}_s)(\texttt{Select}\boldsymbol{U}\otimes\boldsymbol{I}_{a_2})(\boldsymbol{U}_V|0\rangle_a\otimes\boldsymbol{I}_s)\right.\\&\left.-\frac{\chi}{2}(\langle 0|_a\boldsymbol{U}_{\tilde{V}^*}^\dagger\otimes\boldsymbol{I}_s)(\texttt{Select}\boldsymbol{U}\otimes\boldsymbol{I}_{a_2})(\boldsymbol{U}_{\tilde{V}}|0\rangle_a\otimes\boldsymbol{I}_s)\right\|.\end{aligned} \tag{C16}$$

Note that

$$\begin{aligned}&(\langle 0|_a\boldsymbol{U}_{V^*}^\dagger\otimes\boldsymbol{I}_s)(\texttt{Select}\boldsymbol{U}\otimes\boldsymbol{I}_{a_2})(\boldsymbol{U}_V|0\rangle_a\otimes\boldsymbol{I}_s)\\&=(\langle 0|_{a_1}\langle 0|_{a_2}\boldsymbol{U}_{V^*}^\dagger\otimes\boldsymbol{I}_s)\left(\sqrt{1-\delta^2}\sum_{j=0}^{2n-1}\sqrt{\frac{\tilde{t}_j}{\chi}}|j\rangle_{a_1}|0\rangle_{a_2}\otimes\boldsymbol{U}_j+\delta\sum_{j=0}^{2n-1}\zeta_j|j\rangle_{a_1}|1\rangle_{a_2}\otimes\boldsymbol{U}_j\right)\\&=\left(\sqrt{1-\delta^2}\sum_{j=0}^{2n-1}\sqrt{\frac{\tilde{t}_j}{\chi}}\langle j|_{a_1}\langle 0|_{a_2}+\delta\sum_{j=0}^{2n-1}\zeta_j\langle j|_{a_1}\langle 1|_{a_2}\right)\left(\sqrt{1-\delta^2}\sum_{j=0}^{2n-1}\sqrt{\frac{\tilde{t}_j}{\chi}}|j\rangle_{a_1}|0\rangle_{a_2}\otimes\boldsymbol{U}_j+\delta\sum_{j=0}^{2n-1}\zeta_j|j\rangle_{a_1}|1\rangle_{a_2}\otimes\boldsymbol{U}_j\right)\\&=(1-\delta^2)\sum_{j=0}^{2n-1}\frac{\tilde{t}_j}{\chi}\boldsymbol{U}_j+\delta^2\sum_{j=0}^{2n-1}\zeta_j^2\boldsymbol{U}_j.\end{aligned} \tag{C17}$$

Then, computing the first term on the right-hand side of the inequality,

$$\begin{aligned}\left\|\boldsymbol{M}-\frac{\chi}{2}(\langle 0|_a\boldsymbol{U}_{V^*}^\dagger\otimes\boldsymbol{I}_s)(\texttt{Select}\boldsymbol{U}\otimes\boldsymbol{I}_{a_2})(\boldsymbol{U}_V|0\rangle_a\otimes\boldsymbol{I}_s)\right\|&=\left\|\frac{1}{2}\sum_{j=0}^{2n-1}\tilde{t}_j\boldsymbol{U}_j-\frac{1}{2}(1-\delta^2)\sum_{j=0}^{2n-1}\tilde{t}_j\boldsymbol{U}_j-\frac{1}{2}\chi\delta^2\sum_{j=0}^{2n-1}\zeta_j^2\boldsymbol{U}_j\right\|\\&=\left\|\frac{1}{2}\delta^2\sum_{j=0}^{2n-1}\tilde{t}_j\boldsymbol{U}_j-\frac{1}{2}\chi\delta^2\sum_{j=0}^{2n-1}\zeta_j^2\boldsymbol{U}_j\right\|\\&\leqslant\chi\delta^2.\end{aligned} \tag{C18}$$

In addition, since $\boldsymbol{U}_{\tilde{V}}$ and $\boldsymbol{U}_{\tilde{V}^*}$ are ϵ_p approximations of $\boldsymbol{U}_V$ and $\boldsymbol{U}_{V^*}$, respectively, the second term on the right-hand side of Eq. (C16) can be bounded by $\chi\epsilon_p$. Let $\chi\delta^2$ and $\chi\epsilon_p$ not be larger than $\epsilon/2$, we can get the result (i) of Theorem 2.

2. QRAM data structure model

For the QRAM data structure model, the quantum state preparation operators can be implemented using the method of [42]. More specifically, we state the following lemma.

Lemma 6 (from [42]). Suppose that $\boldsymbol{x} \in \mathbb{C}^{n\times 1}$ is stored in a QRAM data structure, i.e., the entry x_i is stored in the ith leaf of a binary tree, and the internal node of the tree stores the sum of the modulus of elements in the subtree rooted in it. Then there is a quantum algorithm that can generate an ϵ_p approximation of $|x\rangle = \frac{1}{\sqrt{\|\boldsymbol{x}\|_1}} \sum_{i=0}^{n-1} \sqrt{x_i}|i\rangle$ with gate complexity $O(\text{polylog}(n/\epsilon_p))$.

Obviously, if $\{\tilde{t}_j\}_{j=0}^{n-1}$ and $\{\tilde{t}_j\}_{j=n}^{2n-1}$ are stored in such a data structure, respectively, there are two unitaries that generate the states

$$\boldsymbol{U}_1|0\rangle = \frac{1}{\sqrt{\chi_1}} \sum_{j=0}^{n-1} \sqrt{\tilde{t}_j}|j\rangle, \quad \boldsymbol{U}_2|0\rangle = \frac{1}{\sqrt{\chi_2}} \sum_{j=0}^{n-1} \sqrt{\tilde{t}_{j+n}}|j\rangle, \tag{C19}$$

where $\chi_1 = \sum_{j=0}^{n-1} |\tilde{t}_j|$ and $\chi_2 = \sum_{j=n}^{2n-1} |\tilde{t}_j|$.

Since χ_1 and χ_2 are known, which are stored in the root of the binary trees, we can prepare a state

$$\frac{\sqrt{\chi_1}}{\sqrt{\chi_{\boldsymbol{T}_n}}}|0\rangle|0\rangle + \frac{\sqrt{\chi_2}}{\sqrt{\chi_{\boldsymbol{T}_n}}}|1\rangle|0\rangle.$$

Then, performing a controlled unitary $|0\rangle\langle 0| \otimes \boldsymbol{U}_1 + |1\rangle\langle 1| \otimes \boldsymbol{U}_2$, we can get the state $\frac{1}{\sqrt{\chi_{\boldsymbol{T}_n}}} \sum_{j=0}^{2n-1} \sqrt{\tilde{t}_j}|j\rangle$. Thus, $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n])}$ can be implemented in the QRAM data structure model with complexity $O(\text{polylog}(n/\epsilon_p))$. Similarly, we can implement $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_n]^*)}$ with the same cost.

In addition, the $\texttt{Select}\boldsymbol{U}_{T_n}$ can be implemented in the same way as in the black-box model. Taking into account the amplification of the error, we can implement the block-encoding with complexity $O(\text{polylog}(n\chi_{\boldsymbol{T}_n}/\epsilon))$ in the QRAM data structure model. Moreover, according to the constructed data structure, the memory cost in this data access model is $O(n)$.

APPENDIX D: PROOF OF COROLLARY 1

1. Black-box model

For a Toeplitz-like matrix $\boldsymbol{T}_L$, given a black box $O_{\boldsymbol{T}_L}$ that queries the kth nonzero element of the ith row of $\boldsymbol{T}_L$,

$$\mathcal{O}_{\boldsymbol{T}_L}|i,k\rangle|0\rangle = |i,k\rangle|\tau_{i,k}\rangle,$$

the following map can be performed by querying the black box $O_{\boldsymbol{T}_L}$ twice:

$$\mathcal{O}_{\tilde{\boldsymbol{T}}_L}|i,k\rangle|0\rangle = |i,k\rangle|\tau_{(i-1)\bmod n,k} - (-1)^{g(k)}\tau_{i,(k+1)\bmod n}\rangle.$$

Here $\mathcal{O}_{\tilde{\boldsymbol{T}}_L}$ actually returns the kth nonzero element of the ith row of the Sylvester displacements of Toeplitz-like matrices, i.e., $\tilde{\tau}_{i,k}$.

In addition, if a black box that computes the positions of the distinct elements (the element is different from the previous element on the same diagonal) on diagonals of the Toeplitz-like matrices is provided, we can construct a black box that computes the positions of nonzero elements of the Sylvester displacements of Toeplitz-like matrices, i.e.,

$$\mathcal{O}^p_{\tilde{\boldsymbol{T}}_L}|i,k\rangle = |i, f(i,k)\rangle,$$

where the function $f(i,k)$ gives the column index of the kth nonzero element in row i of $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L]$.

When implementing the state preparation operators, if the steerable black-box quantum state preparation is directly used, the query complexity should be $O(n)$. To overcome this obstacle, we first prepare a uniform superposition state that only represents the position of the nonzero elements of $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L]$. The specific process is as follows. Prepare an initial state as

$$\frac{1}{\sqrt{2}}(|0\rangle_1 + |1\rangle_1)|0\rangle_2|0\rangle_3. \tag{D1}$$

Apply Hadamard gates to registers 2 and 3 controlled by register 1:

$$\frac{1}{\sqrt{2}}(|0\rangle_1 + |1\rangle_1)|0\rangle_2|0\rangle_3 \xrightarrow{|0\rangle\langle 0|_1\otimes \boldsymbol{I}_2\otimes \boldsymbol{H}_3^{\otimes \log n} + |1\rangle\langle 1|_1\otimes \boldsymbol{H}_2^{\otimes \log n}\otimes \boldsymbol{I}_3^{\otimes \log(n/d)}\otimes \boldsymbol{H}_3^{\otimes \log d}} \frac{1}{\sqrt{2n}}|0\rangle_1|0\rangle_2 \sum_{k=0}^{n-1}|k\rangle_3 + \frac{1}{\sqrt{2nd}}|1\rangle_1 \sum_{i=0}^{n-1}|i\rangle_2 \sum_{k=0}^{d-1}|k\rangle_3. \tag{D2}$$

Using a controlled-$\mathcal{O}^p_{\tilde{\boldsymbol{T}}_L}$, i.e., $|0\rangle\langle 0|_1 \otimes \boldsymbol{I}_{2,3} + |1\rangle\langle 1|_1 \otimes \mathcal{O}^p_{\tilde{\boldsymbol{T}}_L}$, we can prepare

$$\frac{1}{\sqrt{2n}}|0\rangle_1|0\rangle_2 \sum_{k=0}^{n-1}|k\rangle_3 + \frac{1}{\sqrt{2nd}}|1\rangle_1 \sum_{i=0}^{n-1}|i\rangle_2 \sum_{\{f(i,k)|\tilde{m}_{i,f(i,k)}\neq 0\}}|f(i,k)\rangle_3. \tag{D3}$$

Add an ancillary qubit and perform a controlled rotation

$$\frac{1}{\sqrt{2n}}|0\rangle_1|0\rangle_2 \sum_{k=0}^{n-1}|k\rangle_3\left(\frac{1}{\sqrt{d}}|0\rangle_4 + \sqrt{1-\frac{1}{d}}|1\rangle_4\right) + \frac{1}{\sqrt{2nd}}|1\rangle_1 \sum_{i=0}^{n-1}|i\rangle_2 \sum_{\{f(i,k)|\tilde{m}_{i,f(i,k)}\neq 0\}}|f(i,k)\rangle_3|0\rangle_4. \tag{D4}$$

Amplify the amplitude of $|0\rangle_4$. Since the amplitude is known, it can be amplified to exactly 1 by using Long's amplitude amplification with a zero theoretical failure rate [60]. The obtained state is denoted by $|\Phi_{\text{intm}}\rangle$:

$$|\Phi_{\text{intm}}\rangle = \frac{1}{\sqrt{n(d+1)}}|0\rangle_1|0\rangle_2 \sum_{k=0}^{n-1}|k\rangle_3|0\rangle_4 + \frac{1}{\sqrt{n(d+1)}}|1\rangle_1 \sum_{i=0}^{n-1}|i\rangle_2 \sum_{\{f(i,k)|\tilde{m}_{i,f(i,k)}\neq 0\}}|f(i,k)\rangle_3|0\rangle_4. \tag{D5}$$

We run Long's amplitude amplification again to get the quantum state $|\Phi_{\text{init}}\rangle$:

$$|\Phi_{\text{init}}\rangle = \frac{1}{\sqrt{n+(n-1)d}}|0\rangle_1|0\rangle_2\sum_{k=0}^{n-1}|k\rangle_3|0\rangle_4 + \frac{1}{\sqrt{n+(n-1)d}}|1\rangle_1\sum_{i=1}^{n-1}|i\rangle_2 \otimes \sum_{\{f(i,k)|\tilde{m}_{i,f(i,k)}\neq 0\}}|f(i,k)\rangle_3|0\rangle_4. \tag{D6}$$

Note that the success probability of getting $|\Phi_{\text{intm}}\rangle$ is $\frac{d+1}{2d} \geqslant \frac{1}{2}$ and the success probability of getting $|\Phi_{\text{init}}\rangle$ is $\frac{(n-1)d+n}{n(d+1)} \geqslant \frac{1}{2}$; thus only a few iterations are required for the amplitude amplifications.

With the quantum state $|\Phi_{\text{init}}\rangle$ and the black box $O_{\tilde{T}_L}$, we can approximatively implement $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L])}$ and $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L]^*)}$ by the steerable black-box quantum state preparation algorithm. The query complexity is $O(\frac{\sqrt{nd}\log(1/\delta)}{\sqrt{\chi_{T_L}}})$, and $O(\frac{\sqrt{nd}\log(1/\delta)}{\sqrt{\chi_{T_L}}}\text{polylog}(\frac{nd}{\sqrt{\chi_{T_L}}\epsilon_p}))$ elementary gates are required.

To implement $\texttt{Select}\boldsymbol{U}_{T_L}$, we first observe its action on the basis states. Notice that

$$\texttt{Select}\boldsymbol{U}_{T_L}|i\rangle|k\rangle|e\rangle = \begin{cases} |i\rangle|k\rangle|(i+e-k-1)\text{mod}n\rangle \\ \qquad \text{where } 0 \leqslant e \leqslant k, \\ -|i\rangle|k\rangle|(i+e-k-1)\text{mod}n\rangle \\ \qquad \text{where } k < e \leqslant n-1. \end{cases} \tag{D7}$$

Then, on the one hand, let

$$f_2(k,e) = \begin{cases} 0, & 0 \leqslant e \leqslant k \\ 1, & k < e \leqslant n-1. \end{cases} \tag{D8}$$

Similar to the calculation of f_1, we can construct a quantum circuit to implement

$$\boldsymbol{U}_{f_2}|k\rangle|e\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}} = (-1)^{f_2(k,e)}|k\rangle|e\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}. \tag{D9}$$

On the other hand, using quantum adders [58,59], which requires $O(\log n)$ elementary gates, we can implement

$$\boldsymbol{U}_{\text{add2}}|i\rangle|k\rangle|e\rangle = |i\rangle|k\rangle|(i+e-k-1)\text{mod}n\rangle. \tag{D10}$$

Therefore, $\texttt{Select}\boldsymbol{U}_{T_L}$ can be implemented by $\boldsymbol{U}_{\text{add2}}$ and $\boldsymbol{I}\otimes\boldsymbol{U}_{f_2}$ in time $O(\text{polylog}n)$.

Based on the above conclusions and following the error analysis in Appendix C 1, we can infer the result (i) of Corollary 1.

2. QRAM data structure model

For the QRAM data structure model, the quantum state preparation operators can be implemented as follows.

Lemma 7. Let $\boldsymbol{T}_L \in \mathbb{C}^{n\times n}$ and $\|\tilde{\tau}_{i,\cdot}\|_1$ be the 1-norm of the ith row of $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L]$. Suppose that $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L]$ is stored in a QRAM data structure. More specifically, for the ith row of $\nabla_{\boldsymbol{Z}_1,\boldsymbol{Z}_{-1}}[\boldsymbol{T}_L]$, the entry $\tilde{\tau}_{i,k}$ is stored in the kth leaf of a binary tree and the internal node of the tree stores the sum of the modulus of elements in the subtree rooted at it, besides an additional binary tree whose ith leaf stores $\|\tilde{\tau}_{i,\cdot}\|_1$. Then there is a quantum algorithm that can perform the following maps with ϵ_p precision in time $O(\text{polylog}(n/\epsilon_p))$:

$$\boldsymbol{P}: |i\rangle|0\rangle \mapsto \frac{\sum_{k=0}^{n-1}\sqrt{\tilde{\tau}_{i,k}}|i\rangle|k\rangle}{\sqrt{\|\tilde{\tau}_{i,\cdot}\|_1}}, \tag{D11}$$

$$\boldsymbol{P}': |i\rangle|0\rangle \mapsto \frac{\sum_{k=0}^{n-1}\sqrt{\tilde{\tau}^*_{i,k}}|i\rangle|k\rangle}{\sqrt{\|\tilde{\tau}_{i,\cdot}\|_1}}, \tag{D12}$$

$$\boldsymbol{Q}: |0\rangle|k\rangle \mapsto \frac{\sum_{i=0}^{n-1}\sqrt{\|\tilde{\tau}_{i,\cdot}\|_1}|i\rangle|k\rangle}{\sqrt{\chi_{\boldsymbol{T}_L}}}. \tag{D13}$$

This conclusion can be directly derived from the results in [42]. Obviously,

$$\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L])}|0\rangle|0\rangle = \boldsymbol{P}\boldsymbol{Q}|0\rangle|0\rangle = \frac{1}{\sqrt{\chi_{\boldsymbol{T}_L}}}\sum_{i=0}^{n-1}\sum_{k=0}^{n-1}\sqrt{\tilde{\tau}_{i,k}}|i\rangle|k\rangle. \tag{D14}$$

Similarly, we can efficiently implement $\boldsymbol{V}_{(\nabla[\boldsymbol{T}_L]^*)}$ with $\boldsymbol{P}'$ and $\boldsymbol{Q}$.

Note that $\texttt{Select}\boldsymbol{U}_{T_L}$ can be implemented in the same way as in Appendix D 1. Taking into account the amplification of the error, we can implement the block-encoding with complexity $O(\text{polylog}(n\chi_{\boldsymbol{T}_L}/\epsilon))$ in the QRAM data structure model. Moreover, the memory cost in this data access model is easy to calculate as $O(dn\log n)$.

[1] C. H. Bennett and G. Brassard, Theor. Comput. Sci. **560**, 7 (2014).
[2] C.-Y. Wei, X.-Q. Cai, T.-Y. Wang, S.-J. Qin, F. Gao, and Q.-Y. Wen, IEEE J. Sel. Areas Commun. **38**, 517 (2020).
[3] G.-B. Xu and D.-H. Jiang, Quantum Inf. Process. **20**, 128 (2021).
[4] F. Gao, S. Qin, W. Huang, and Q. Wen, Sci. China Phys. Mech. **62**, 70301 (2019).
[5] J. F. Fitzsimons, npj Quantum Inf. **3**, 23 (2017).
[6] X. He, X. Sun, G. Yang, and P. Yuan, arXiv:1801.05717.
[7] W. Chen, Z. Ye, and L. Li, Phys. Rev. A **101**, 022325 (2020).
[8] A. W. Harrow, A. Hassidim, and S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009).
[9] L. Zhao, Z. Zhao, P. Rebentrost, and J. Fitzsimons, Quantum Mach. Intell. **3**, 21 (2021).
[10] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, edited by M. Charikar and E. Cohen (ACM Press, New York, 2019), pp. 193–204.
[11] C.-H. Yu, F. Gao, Q.-L. Wang, and Q.-Y. Wen, Phys. Rev. A **94**, 042311 (2016).
[12] C.-H. Yu, F. Gao, S. Lin, and J. Wang, Quantum Inf. Process. **18**, 249 (2019).
[13] C.-H. Yu, F. Gao, and Q.-Y. Wen, IEEE Trans. Knowl. Data Eng. **33**, 858 (2021).
[14] S.-J. Pan, L.-C. Wan, H.-L. Liu, Q.-L. Wang, S.-J. Qin, Q.-Y. Wen, and F. Gao, Phys. Rev. A **102**, 052402 (2020).

[15] A. Montanaro, npj Quantum Inf. **2**, 15023 (2016).

[16] T. Kailath and A. H. Sayed, *Fast Reliable Algorithms for Matrices with Structure* (SIAM, Philadelphia, 1999).

[17] V. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms* (Springer Science+Business Media, New York, 2001).

[18] V. Peller, *Hankel Operators and Their Applications* (Springer Science+Business Media, New York, 2012).

[19] Y. Cao, A. Papageorgiou, I. Petras, J. Traub, and S. Kais, New J. Phys. **15**, 013021 (2013).

[20] A. Mahasinghe and J. B. Wang, J. Phys. A: Math. Theor. **49**, 275301 (2016).

[21] S. S. Zhou and J. B. Wang, R. Soc. Open Sci. **4**, 160906 (2017).

[22] L.-C. Wan, C.-H. Yu, S.-J. Pan, F. Gao, Q.-Y. Wen, and S.-J. Qin, Phys. Rev. A **97**, 062322 (2018).

[23] G. H. Low and I. L. Chuang, Quantum **3**, 163 (2019).

[24] S. Chakraborty, A. Gilyén, and S. Jeffery, in *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming*, edited by C. Baier, I. Chatzigiannakis, P. Flocchini, and S. Leonardi (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Saarbrücken, 2019).

[25] J. van Apeldoorn and A. Gilyén, in *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)* (Ref. [24]).

[26] I. Kerenidis and A. Prakash, Phys. Rev. A **101**, 022316 (2020).

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

[28] R. M. Gray, Found. Trends Commun. Inf. Theory **2**, 155 (2006).

[29] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, IEEE Trans. Pattern Anal. **37**, 583 (2014).

[30] G. D. Smith, G. D. Smith, and G. D. S. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods* (Oxford University Press, Oxford, 1985).

[31] R. H. Chan and M. K. Ng, SIAM Rev. **38**, 427 (1996).

[32] J. C. Ye, Y. Han, and E. Cha, SIAM J. Imag. Sci. **11**, 991 (2018).

[33] J. Haupt, W. U. Bajwa, G. Raz, and R. Nowak, IEEE Trans. Inf. Theory **56**, 5862 (2010).

[34] J. Van Apeldoorn, A. Gilyén, S. Gribling, and R. de Wolf, in *Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS)* (IEEE, Piscataway, NJ, 2017), pp. 403–414.

[35] I. Kerenidis, J. Landman, A. Luongo, and A. Prakash, in *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (ACM Press, New York, 2019), pp. 4136–4146.

[36] C. Shao, J. Phys. A: Math. Theor. **53**, 045301 (2020).

[37] I. Kerenidis and A. Prakash, ACM Trans. Quantum Comput. **1**, 1 (2020).

[38] L. Grover and T. Rudolph, arXiv:quant-ph/0208112.

[39] A. N. Soklakov and R. Schack, Phys. Rev. A **73**, 012307 (2006).

[40] D. W. Berry and A. M. Childs, Quantum Inf. Comput. **12**, 29 (2012).

[41] Y. R. Sanders, G. H. Low, A. Scherer, and D. W. Berry, Phys. Rev. Lett. **122**, 020502 (2019).

[42] I. Kerenidis and A. Prakash, in *Proceedings of the 8th Innovations in Theoretical Computer Science Conference*, edited by C. H. Papadimitriou (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Saarbrücken, 2017), Vol. 67, pp. 49:1–49:21.

[43] L. K. Grover, Phys. Rev. Lett. **85**, 1334 (2000).

[44] T. J. Yoder, G. H. Low, and I. L. Chuang, Phys. Rev. Lett. **113**, 210501 (2014).

[45] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma, Phys. Rev. Lett. **114**, 090502 (2015).

[46] G.-L. Long, Y. Liu, and C. Wang, Commun. Theor. Phys. **51**, 65 (2009).

[47] C.-H. Yu, F. Gao, C. Liu, D. Huynh, M. Reynolds, and J. Wang, Phys. Rev. A **99**, 022301 (2019).

[48] A. Ambainis, in *Proceedings of the 29th Symposium on Theoretical Aspects of Computer Science*, edited by C. Dürr and T. Wilke (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Saarbrücken, 2012), Vol. 14, pp. 636–647.

[49] K. Habermann, Australas. J. Combinat. **79**, 250 (2021).

[50] H. Widom, Trans. Am. Math. Soc. **121**, 1 (1966).

[51] A. Schönhage, J. Complexity **1**, 118 (1985).

[52] S. S. Haykin, *Adaptive Filter Theory* (Pearson Education India, Delhi, 2005).

[53] D. Aharonov, V. Jones, and Z. Landau, Algorithmica **55**, 395 (2009).

[54] B. Wu, M. Ray, L. Zhao, X. Sun, and P. Rebentrost, Phys. Rev. A **103**, 042422 (2021).

[55] V. V. Shende, S. S. Bullock, and I. L. Markov, IEEE Trans. Comput.-Aid. Design **25**, 1000 (2006).

[56] E. Tang, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (Ref. [10]), pp. 217–228.

[57] X. Wang, Z. Song, and Y. Wang, Quantum **5**, 483 (2021).

[58] S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, arXiv:quant-ph/0410184.

[59] H.-S. Li, P. Fan, H. Xia, H. Peng, and G.-L. Long, Sci. China Phys. Mech. **63**, 280311 (2020).

[60] G.-L. Long, Phys. Rev. A **64**, 022307 (2001).