Improved quantum algorithm for A-optimal projection

Shi-Jie Pan,[1,2,3] Lin-Chun Wan,[1] Hai-Ling Liu,[1] Qing-Le Wang,[4,5] Su-Juan Qin,[1,*] Qiao-Yan Wen,[1,†] and Fei Gao[1,3,‡]

[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]*Center for Quantum Computing, Peng Cheng Laboratory, Shenzhen 518055, China*

[4]*School of Control and Computer Engineering, North China Electric Power University, Beijing 102206, China*

[5]*CAS Key Laboratory of Quantum Information, University of Science and Technology of China, Hefei 230026, China*

(Received 11 June 2020; accepted 29 September 2020; published 4 November 2020)

Dimensionality reduction algorithms, which reduce the dimensionality of a given data set whereas preserving the information of the original data set as well as possible, play an important role in machine learning and data mining. Duan *et al.* proposed a quantum version of the A-optimal projection algorithm (AOP) for dimensionality reduction [Phys. Rev. A **99**, 032311 (2019)] and claimed that the algorithm has exponential speedups on the dimensionality of the original feature space n and the dimensionality of the reduced feature space k over the classical algorithm. In this paper, we correct the time complexity of the algorithm of Duan *et al.* to $O[\frac{\kappa^{4s}\sqrt{k^s}}{\epsilon^s}\mathrm{polylog}^s(\frac{mn}{\epsilon})]$, where κ is the condition number of a matrix that related to the original data set, s is the number of iterations, m is the number of data points, and ϵ is the desired precision of the output state. Since the time complexity has an exponential dependence on s, the quantum algorithm can only be beneficial for high-dimensional problems with a small number of iterations s. To get a further speedup, we propose an improved quantum AOP algorithm with time complexity $O[\frac{s\kappa^6\sqrt{k}}{\epsilon}\mathrm{polylog}(\frac{mn}{\epsilon}) + \frac{s^2\kappa^4}{\epsilon}\mathrm{polylog}(\frac{\kappa k}{\epsilon})]$ and space complexity $O[\log_2(nk/\epsilon) + s]$. With space complexity slightly worse, our algorithm achieves, at least, a polynomial speedup compared to the algorithm of Duan *et al.*. Also, our algorithm shows exponential speedups in n and m compared with the classical algorithm when κ, k, and $1/\epsilon$ are $O[\mathrm{polylog}(nm)]$.

DOI: 10.1103/PhysRevA.102.052402

I. INTRODUCTION

Quantum computing is more computationally powerful than classical computing in solving specific problems, such as the factoring problem [1], the unstructured data search problem [2], and the matrix computation problems [3,4]. In recent years, quantum machine learning (QML) has received wide attention as an emerging research area that successfully combines quantum physics and machine learning. An important part of the study of QML focuses on designing quantum algorithms to speed up the machine learning problems, such as data classification [5–9], linear regression [10–14], association rules mining [15], and anomaly detection [16].

In the big data era, most of the real-world data are high dimensional, which requires high computational performance and usually causes a problem called *curse of dimensionality* [17]. Since the high-dimensional real-world data are often confined to a region of the space having lower effective dimensionality [17], a technique called dimensionality reduction (DR) which reduces the dimensionality of the given data set whereas preserving the information of the original data set as well as possible was proposed. The DR algorithm often serves as a preprocessing step in data mining and machine learning.

Based on the feature space that the data lie on and the learning task that we want to handle, various DR algorithms have been developed. Generally, when the data lie on a linear embedded manifold, principal component analysis (PCA) [18], a DR algorithm maintaining the characteristics of the data set that contribute the most to the variance, is guaranteed to uncover the intrinsic dimensionality of the manifold. When the data lie on a nonlinearly embedded manifold, the manifold learning techniques, such as isomap [19], locally linear embedding [20], and the Laplacian eigenmap [21] can be used to discover the nonlinear structure of the manifold. Since the DR algorithms mentioned above aim to discover the geometrical or cluster structure of the training data, these algorithms are not directly related to the classification and regression tasks which are the two most important tasks in machine learning and data mining. For the classification task, a famous DR technique called linear discriminant analysis (LDA) was put forward, which maximizes the ratio of the between-class variance and the within-class variance of the training data [22]. For the regression task, He *et al.* proposed a novel DR algorithm called A-optimal projection (AOP) that aims to minimize the prediction error of a regression model whereas reducing the dimensionality [23]. Their algorithm improves the regression performance in the reduced space.

In the context of quantum computing, the quantum PCA was proposed by Lloyd *et al.* to reveal in quantum form the eigenvectors corresponding to the large eigenvalues of an unknown low-rank density matrix [24]. Later, Yu *et al.* proposed a quantum algorithm that compresses training data based on

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

PCA [25]. When the dimensionality of the reduced space is polylogarithmic in the training data, their quantum algorithm achieves an exponential speedup compared with the classical algorithm. Cong and Duan implemented a quantum LDA algorithm which has an exponential speedup in the scales of the original data set compared with the classical algorithm [7]. In Ref. [26], Duan *et al.* studied the AOP algorithm and proposed its quantum counterpart, called the Duan-Yuan-Xu-Li (DYXL) algorithm. The DYXL algorithm is iterative and was expected to have a time complexity $O[s\text{polylog}(nk/\epsilon)]$, where s is the number of iterations, n is the dimensionality of the original feature space, k is the dimensionality of the reduced feature space, and ϵ is the desired precision of the output state.

In this paper, we reanalyze the DYXL algorithm and correct the time complexity to $O[\frac{\kappa^{4s}\sqrt{k^s}}{\epsilon^s}\text{polylog}^s(\frac{mn}{\epsilon})]$, where κ is the condition number of a matrix that related to the original data set, and m is the number of data points. We find that in the DYXL algorithm multiple copies of the current candidate are consumed to improve the candidate by quantum phase estimation and postselection in each iteration, which results in the total time complexity having exponential dependence on the number of iterations s. Thus, the DYXL algorithm can only be beneficial for high-dimensional problems with a small s, which limits the practical application of the algorithm. To get a further speedup and reduce the dependence on s, we propose an improved quantum AOP algorithm with time complexity $O[\frac{s\kappa^6\sqrt{k}}{\epsilon}\text{polylog}(\frac{nm}{\epsilon}) + \frac{s^2\kappa^4}{\epsilon}\text{polylog}(\frac{\kappa k}{\epsilon})]$. Note that in the DYXL algorithm, one only changes the amplitude of the candidate in each iteration. In our algorithm, we process the amplitude information of the candidate in computational basis to reduce the consumption of the copies of the current candidate. Our algorithm has a quadratic dependence rather than an exponential dependence on s in time complexity, which achieves a significant speedup over the DYXL algorithm with the space complexity slightly worse. Also, it shows exponential speedups over the classical algorithm on n and m when κ, k, and $1/\epsilon$ are $O[\text{polylog}(nm)]$.

The rest of this paper is organized as follows. In Sec. II, we review the classical AOP algorithm in Sec. II A, its quantum version in Sec. II B, and analyze the complexity of the DYXL algorithm in Sec. II C. We then propose our quantum AOP algorithm and analyze the complexity in Sec. III. In Sec. IV, we discuss the number of iterations of the two quantum algorithms. The conclusion is given in Sec. V.

II. REVIEW OF THE CLASSICAL AND QUANTUM AOP ALGORITHM

In this section, we will briefly review the AOP algorithm in Sec. II A. The DYXL algorithm will be introduced in Sec. II B, and we will analyze its complexity in Sec. II C.

A. Review of the AOP algorithm

Suppose $X = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ is a data matrix with dimension $n \times m$, where n is the number of the features and m is the number of data points. The objective of the AOP is to find the optimal projection matrix $A \in n \times k$ which minimizes the trace of the covariance matrix of regression parameters to reduce the dimensionality of X.

In He *et al.* [23], a graph regularized regression model was chosen, and, thus, the optimal projection matrix A can be obtained by solving the following objective function:

$$\min_A \text{Tr}[(A^T X(I + \lambda_1 L)X^T A + \lambda_2 I)^{-1}], \tag{1}$$

where λ_1 and λ_2 are the regularized coefficients, $L = \text{diag}(S\mathbf{1}) - S$ is the *graph Laplacian* where S is the weight matrix of the data points and $\mathbf{1}$ is a vector of all ones. Let $\mathcal{N}_k(\mathbf{x})$ denote the k nearest neighbors of $\mathbf{x}$, a simple definition of S is as follows:

$$S_{i,j} = \begin{cases} 1, & \text{if } \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i), \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

To solve the optimization problem (1), He *et al.* introduced a variable B and proposed the following theorem [23]:

Theorem 1 (Theorem 4.3 in Ref. [23]). The optimization problem (1) is equivalent to the following optimization problem:

$$\min_{A,B} \|I - A^T \widetilde{X} B\|^2 + \lambda \|B\|^2, \tag{3}$$

where $\widetilde{X} = X\Sigma$ and Σ is defined by the equation $I + \lambda_1 L = \Sigma\Sigma^T$.

Then we can use the iterative method to find the optimal A. The procedure of computing the projection matrix A can be summarized as follows:

(1) Initialize the matrix A by computing the PCA of the data matrix X.

(2) Computing matrix B according to Eq. (4),

$$B = (\widetilde{X}^T A A^T \widetilde{X} + \lambda_2 I)^{-1} \widetilde{X}^T A. \tag{4}$$

(3) Computing matrix A according to Eq. (5),

$$A = (\widetilde{X} B B^T \widetilde{X}^T)^{-1} \widetilde{X} B. \tag{5}$$

Normalize A to satisfy $\|A\|_F \leqslant \rho$ (ρ is a constant and here we set it to 1).

(4) Repeat steps 2 and 3 until convergence.

Since the AOP algorithm involves matrix multiplication and inversion, the time complexity of the classical algorithm is $\Omega[s\text{poly}(nm)]$, where s is the number of iterations.

B. Review of the DYXL algorithm

In Ref. [26], the authors reformulated the iterative method of AOP to make the algorithm suitable for quantum settings. They adjusted the initialization of matrix A and combined steps 2 and 3 into one step to remove the variable B. Suppose the singular value decomposition of matrix $\widetilde{X}$ is $\widetilde{X} = \sum_{j=0}^{r-1} \sigma_j |\mathbf{u}_j\rangle\langle\mathbf{v}_j|$, where $r = O[\text{polylog}(m, n)]$ is the rank of $\widetilde{X}$, $|\mathbf{u}_j\rangle$ and $|\mathbf{v}_j\rangle$ are the left and right singular vectors with a corresponding singular value of $\sigma_j (1 = \sigma_0 \geqslant \sigma_1 \geqslant \cdots \geqslant \sigma_{r-1} > 0)$. The reformulated AOP algorithm can be summarized as follows:

(1) Initialize matrix $A^{(0)}$ by computing the PCA of the data matrix $\widetilde{X}$,

$$A^{(0)} = \text{PCA}(\widetilde{X}) = \sum_{j=0}^{k-1} |\mathbf{u}_j\rangle\langle\mathbf{j}|, \tag{6}$$

where $A^{(i)}$ is the matrix A of iteration i, k is the rank of $A^{(0)}$ and $|\mathbf{j}\rangle$ is the computational basis state.

(2) Update matrix A according to the following equation:

$$A^{(i)} = \sum_{j=0}^{k-1} \beta_j^{(i)} |\mathbf{u}_j\rangle\langle\mathbf{j}| = \sum_{j=0}^{k-1} \frac{\left(\sigma_j \beta_j^{(i-1)}\right)^2 + \lambda_2}{c^{(i)} \sigma_j^2 \beta_j^{(i-1)}} |\mathbf{u}_j\rangle\langle\mathbf{j}|, \quad (7)$$

where $\beta_j^{(i)}$ is the singular value of $A^{(i)}$ with corresponding left and right singular vectors $|\mathbf{u}_j\rangle$ and $|\mathbf{j}\rangle$, $c^{(i)}$ is a constant to ensure that $\|A^{(i)}\|_F \leqslant 1$.

(3) Repeat step 2 until convergence.

The DYXL algorithm can be summarized as follows:

(1) Initialize $i = 0$, and prepare state $|\psi_{A^{(0)}}\rangle$, where

$$|\psi_{A^{(0)}}\rangle = \frac{1}{\sqrt{k}} \sum_{j=0}^{k-1} |\mathbf{u}_j\rangle|\mathbf{j}\rangle. \quad (8)$$

(2) Suppose quantum state $|\psi_{A^{(i-1)}}\rangle$ is given, prepare the following state:

$$\begin{aligned} \left|\psi_0^{(i-1)}\right\rangle &= |0\rangle^D (|0\rangle \cdots |0\rangle)^C (|0\rangle \cdots |0\rangle)^B |\psi_{A^{(i-1)}}\rangle^A \\ &= |0\rangle^D \sum_{j=0}^{k-1} \beta_j^{(i-1)} (|0\rangle \cdots |0\rangle)^C (|0\rangle \cdots |0\rangle)^B (|\mathbf{u}_j\rangle|\mathbf{j}\rangle)^A, \end{aligned} \quad (9)$$

where the superscripts D, C, B, and A represent the registers D, C, B, and A, respectively.

(3) Perform phase estimation on $|\psi_0^{(i-1)}\rangle$ for the unitary $e^{i\widetilde{X}\widetilde{X}^\dagger t_0}$ and $e^{iA^{(i-1)}A^{(i-1)\dagger} t_0}$,

$$\left|\psi_1^{(i-1)}\right\rangle = |0\rangle^D \sum_{j=0}^{k-1} \beta_j^{(i-1)} \left|\sigma_j^2\right\rangle^C \left|\left(\beta_j^{(i-1)}\right)^2\right\rangle^B (|\mathbf{u}_j\rangle|\mathbf{j}\rangle)^A. \quad (10)$$

(4) Perform an appropriate controlled rotation on the registers B, C, and D and transform the system to

$$\begin{aligned} \left|\psi_2^{(i-1)}\right\rangle = &\sum_{j=0}^{k-1} \beta_j^{(i-1)} \left|\sigma_j^2\right\rangle^C \left|\left(\beta_j^{(i-1)}\right)^2\right\rangle^B (|\mathbf{u}_j\rangle|\mathbf{j}\rangle)^A \\ &\times \left[\sqrt{1 - \rho^2 f\left(\sigma_j, \beta_j^{(i-1)}\right)^2} |0\rangle + \rho f\left(\sigma_j, \beta_j^{(i-1)}\right)|1\rangle\right]^D, \end{aligned} \quad (11)$$

where ρ is a constant to ensure $|\rho f(\sigma_j, \beta_j^{(i-1)})| \leqslant 1$, $f(\sigma_j, \beta_j^{(i-1)}) = \frac{(\sigma_j \beta_j^{(i-1)})^2 + \lambda_2}{(\sigma_j \beta_j^{(i-1)})^2}$.

(5) Measure the register D, then uncompute the registers C, B, and A, and remove the registers C, B. Conditioned on seeing 1 in D, we have the state,

$$\begin{aligned} \left|\psi_3^{(i-1)}\right\rangle &= \frac{1}{\sqrt{N^{(i)}}} \sum_{j=0}^{k-1} \frac{\left(\sigma_j \beta_j^{(i-1)}\right)^2 + \lambda_2}{\sigma_j^2 \beta_j^{(i-1)}} |\mathbf{u}_j\rangle|\mathbf{j}\rangle \\ &= \sum_{j=0}^{k-1} \beta_j^{(i)} |\mathbf{u}_j\rangle|\mathbf{j}\rangle = |\psi_{A^{(i)}}\rangle, \end{aligned} \quad (12)$$

where $N^{(i)} = \sum_{j=0}^{k-1} \left(\frac{(\sigma_j \beta_j^{(i-1)})^2 + \lambda_2}{\sigma_j^2 \beta_j^{(i-1)}}\right)^2$. Thus, $\beta_j^{(i)} \in [0, 1]$ for $j \in \{0, 1, 2, \ldots, k-1\}$ and $i \geqslant 0$.

TABLE I. The time complexity of each step of the DYXL algorithm.

Steps[a]	Time complexity
Step 1	$O[\log_2(\epsilon^{-1}) \log_2(nk)]$
Step 3	$O[(G^{(i-1)}/\epsilon_1) \log_2(1/\epsilon_1)] +$ $O[(1/\epsilon_1)\text{polylog}(nm/\epsilon_1)]$
Step 4	$O[\text{polylog}(1/\epsilon)]$
Step 5	$O(\kappa^2)$ repetitions

[a]Here steps 3–5 are the steps of the ith iteration, and we neglect the runtime of step 2. $G^{(i-1)}$ is the time complexity to prepare state $|\psi_{A^{(i-1)}}\rangle$, κ is the condition number of $\widetilde{X}$, and ϵ is the desired precision of the output state $\epsilon_1 = O(\frac{\epsilon}{\kappa^2\sqrt{k}})$.

(6) For $i = 1$ to $s - 1$, repeat steps 2–5.

C. Complexity analysis of the DYXL algorithm

In Ref. [26], the authors analyzed the time complexity of each iteration (step 2–step 5 in this paper) and claimed that the total time complexity is the product of the number of iterations and the time complexity of each iteration. Actually, in the ith iteration, the algorithm has to prepare state $|\psi_{A^{(i-1)}}\rangle$ several times to perform phase estimation in step 3 and perform measurements to obtain an appropriate state in step 5, which means that the total time complexity is exponential on the number of iterations s. The time complexity of each step can be seen in Table I, and the proof details can be seen in Appendix A.

Putting it all together, the runtime of the ith iteration (i.e., preparing state $|\psi_{A^{(i)}}\rangle$) is

$$\begin{aligned} G^{(i)} &= O\left\{\left[\frac{G^{(i-1)}}{\epsilon_1} \log_2\left(\frac{1}{\epsilon_1}\right) + \frac{1}{\epsilon_1}\text{polylog}\frac{mn}{\epsilon_1} + \text{polylog}\left(\frac{1}{\epsilon}\right)\right]\kappa^2\right\} \\ &= O\left[\frac{\kappa^4\sqrt{k} G^{(i-1)}}{\epsilon}\text{polylog}\left(\frac{mn}{\epsilon}\right)\right] = O(T G^{(i-1)}), \end{aligned}$$

where $T = (\kappa^4\sqrt{k}/\epsilon)\text{polylog}(mn/\epsilon)$. Since $G^{(0)} = O[\log_2(\epsilon^{-1}) \log_2(nk)]$, the overall time complexity of the algorithm is

$$\begin{aligned} G^{(s)} &= O(T G^{(s-1)}) = O(T^s G^{(0)}) \\ &= O\left(\frac{\kappa^{4s}\sqrt{k^s}}{\epsilon^s}\text{polylog}^s(mn/\epsilon)\right). \end{aligned} \quad (13)$$

As for the space complexity, $O[\ln(nk/\epsilon)]$ qubits are used to prepare initial state $|\psi_{A^{(0)}}\rangle$. In steps 2 and 3, the quantum phase estimations require $O[\ln(1/\epsilon_1)]$ qubits. In step 4, the controlled rotation requires $O[\ln(1/\epsilon)]$ ancillary qubits. Note that the qubits in the current iteration can be reused in the next iteration, thus, the space complexity of the algorithm is $O[\ln(nk/\epsilon)]$. The details are shown in Appendix A.

III. AN IMPROVED QUANTUM AOP ALGORITHM

In this section, we present an improved quantum AOP algorithm.

In the DYXL algorithm, to get state $|\psi_{A^{(i)}}\rangle$ from state $|\psi_{A^{(i-1)}}\rangle$, one performs phase estimation and postselection, which consumes multiple copies of $|\psi_{A^{(i-1)}}\rangle$, thus, the number of copies of initial state $|\psi_{A^{(0)}}\rangle$ required depends exponentially on the number of iterations s. Note that, in each iteration of the DYXL algorithm, one only changes the singular value β_j for $j = 0, 1, \ldots k-1$. In our algorithm, we put the calculation into the computational basis to reduce the consumption of the copies of $|\psi_{A^{(i-1)}}\rangle$.

A. An improved quantum AOP algorithm

The specific process of our quantum algorithm is as follows:

(1) *Initialization*. Initialize $i = 0$, and prepare state $|\psi_{A^{(0)}}\rangle = \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle$.

(2) *Prepare state* $|\psi_0\rangle$. Perform phase estimation with precision parameter ϵ on state $|\psi_{A^{(0)}}\rangle$ for the unitary $e^{i\widetilde{X}\widetilde{X}^{\dagger}t_0}$, and then append state $|\frac{1}{\sqrt{k}}\rangle|0\rangle$, thus, we obtain

$$\begin{aligned}|\psi_0\rangle &= \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}(|\mathbf{u}_j\rangle|\mathbf{j}\rangle)^A\left|\sigma_j^2\right\rangle^B\left|\frac{1}{\sqrt{k}}\right\rangle^C|0\rangle^D\\ &= \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}(|\mathbf{u}_j\rangle|\mathbf{j}\rangle)^A\left|\sigma_j^2\right\rangle^B\left|\beta_j^{(0)}\right\rangle^C|0\rangle^D,\end{aligned} \tag{14}$$

where $\beta_j^{(0)} = \frac{1}{\sqrt{k}}$ for $j = 0, 1, \ldots, k-1$, the superscripts A, B, C, and D represent the registers A, B, C, and D, respectively (in the absence of ambiguity, we omit these superscripts below for the sake of simplicity).

Assuming that we can prepare state $|\psi_{i-1}\rangle$, where

$$|\psi_{i-1}\rangle = \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle\left|\beta_j^{(i-1)}\right\rangle|0\rangle. \tag{15}$$

Thus, we could perform quantum arithmetic operations to get

$$\left|\phi_1^{(i)}\right\rangle = \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle\left|\beta_j^{(i-1)}\right\rangle\left|c^{(i)}\beta_j^{(i)}\right\rangle, \tag{16}$$

where $c^{(i)}\beta_j^{(i)} = \frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}$ and $\sum_{j=0}^{k-1}(\beta_j^{(i)})^2 = 1$.

In order to obtain the information of $\beta_j^{(i)}$, we will estimate $c^{(i)}$ first, then we can prepare state $\frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle|\sigma_j^2\rangle|\beta_j^{(i-1)}\rangle|\beta_j^{(i)}\rangle := |\phi_3^{(i)}\rangle$ from state $|\phi_1^{(i)}\rangle$.

(3) *Estimate* $c^{(i)}$. Assuming that we can prepare state $|\psi_{i-1}\rangle$ in time G_{i-1}.

(i) Prepare state $|\phi_1^{(i)}\rangle$ from state $|\psi_{i-1}\rangle$.

(ii) Add an ancillary qubit (register E) and perform an appropriate controlled rotation on the registers D and E, transforms the system to

$$\begin{aligned}\left|\phi_2^{(i)}\right\rangle &= \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}(|\mathbf{u}_j\rangle|\mathbf{j}\rangle)^A\left|\sigma_j^2\right\rangle^B\left|\beta_j^{(i-1)}\right\rangle^C\left|c^{(i)}\beta_j^{(i)}\right\rangle^D\\ &\times\left[\sqrt{1-\left(\frac{c^{(i)}\beta_j^{(i)}}{c}\right)^2}|0\rangle+\frac{c^{(i)}\beta_j^{(i)}}{c}|1\rangle\right]^E\\ &:= \cos(\theta)|a\rangle|0\rangle^E + \sin(\theta)|b\rangle|1\rangle^E,\end{aligned} \tag{17}$$

where the parameter c is a constant to ensure $\frac{c^{(i)}\beta_j^{(i)}}{c} \leqslant 1$,

$$\begin{aligned}|a\rangle &= \sum_{j=0}^{k-1}\sqrt{\frac{c^2-\left(c^{(i)}\beta_j^{(i)}\right)^2}{kc^2-(c^{(i)})^2}}|\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle\left|\beta_j^{(i-1)}\right\rangle\left|c^{(i)}\beta_j^{(i)}\right\rangle,\\ |b\rangle &= \sum_{j=0}^{k-1}\beta_j^{(i)}|\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle\left|\beta_j^{(i-1)}\right\rangle\left|c^{(i)}\beta_j^{(i)}\right\rangle,\\ \sin(\theta) &= \frac{c^{(i)}}{c\sqrt{k}}.\end{aligned} \tag{18}$$

(iii) Perform quantum amplitude estimation to estimate $\sin(\theta)$. Then we can obtain the classical information of $c^{(i)}$ by $c^{(i)} = \sqrt{k}c\,\sin(\theta)$.

(4) *Prepare state* $|\psi_i\rangle$.

(i) Since we have the classical information of $c^{(i)}$, we can perform a quantum arithmetic operation to get

$$\left|\phi_3^{(i)}\right\rangle = \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle^B\left|\beta_j^{(i-1)}\right\rangle^C\left|\beta_j^{(i)}\right\rangle^D. \tag{19}$$

Note that by using the techniques from step 2 to stage (i) of step 4, we could prepare state $\frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle|\sigma_j^2\rangle|\beta_j^{(0)}\rangle|\beta_j^{(1)}\rangle\cdots|\beta_j^{(s)}\rangle$ from state $|\psi_{A^{(0)}}\rangle$. Then followed by controlled rotation, uncomputing, and measurement, we could obtain the desired state $|\psi_{A^{(s)}}\rangle$. However, it requires much more space resource than the DYXL algorithm. To reduce the space complexity, we transform the register C to $|0\rangle$ and only keep $|\beta_j^{(i)}\rangle$ after iteration i, i.e., obtain state $|\psi_i\rangle = \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle|\sigma_j^2\rangle^B|\beta_j^{(i)}\rangle^D|0\rangle^C$.

(ii) Perform a quantum arithmetic operation on registers B, C, and D, to get $|\psi_i\rangle$. Since $c^{(i)}\beta_j^{(i)} = \frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}$, we have

$$\sigma_j^2\left(\beta_j^{(i-1)}\right)^2 - c^{(i)}\sigma_j^2\beta_j^{(i)}\beta_j^{(i-1)} + \lambda_2 = 0, \tag{20}$$

which is a one-variable quadratic equation about the variable $\beta_j^{(i-1)}$. The solution of the equation is

$$\beta_{j\pm}^{(i-1)} = \frac{c^{(i)}\sigma_j^2\beta_j^{(i)} \pm \sqrt{\left(c^{(i)}\sigma_j^2\beta_j^{(i)}\right)^2 - 4\sigma_j^2\lambda_2}}{2\sigma_j^2}. \tag{21}$$

Two cases are considered here. In case 1, when $\lambda_2 \geqslant 1$, then for $x \leqslant 1$, the function $f(x) = \frac{(\sigma_j x)^2+\lambda_2}{\sigma_j^2 x}$ is a monotonic decreasing function, which means that $\beta_j^{(i-1)} = \beta_{j-}^{(i-1)}$. In case 2, when $\lambda_2 < 1$, we add a qubit (register F) to store the magnitude relationship between $\beta_j^{(i-1)}$ and $\sqrt{\frac{\lambda}{\sigma^2}}$ on the state in Eq. (19), i.e.,

$$\left|\phi_4^i\right\rangle = \frac{1}{\sqrt{k}}\sum_{j=0}^{k-1}|\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle^B\left|\beta_j^{(i-1)}\right\rangle^C\left|\beta_j^{(i)}\right\rangle^D\left|\gamma_j^{(i)}\right\rangle^F,$$

where

$$\left|\gamma_j^{(i)}\right\rangle = \begin{cases}|1\rangle, & \text{if } \beta_j^{(i-1)} \geqslant \sqrt{\frac{\lambda}{\sigma^2}},\\ |0\rangle, & \text{if } \beta_j^{(i-1)} < \sqrt{\frac{\lambda}{\sigma^2}}.\end{cases} \tag{22}$$

Then, according to the information in register F, we could get

$$\beta_j^{(i-1)} = \begin{cases} \beta_{j+}^{(i-1)}, & \text{if } \left|\gamma_j^{(i)}\right\rangle = |1\rangle, \\ \beta_{j-}^{(i-1)}, & \text{if } \left|\gamma_j^{(i)}\right\rangle = |0\rangle. \end{cases} \tag{23}$$

Thus, we could transform the state of register C to $|0\rangle$ by a simple quantum arithmetic operation on registers C and D. We should keep in mind that we need an ancillary qubit in each iteration for case 2.

(5) *Iteration*. For $i = 1$ to $s-2$, repeat steps 2–4. Thus, we obtain state,

$$|\psi_{s-1}\rangle = \frac{1}{\sqrt{k}} \sum_{j=0}^{k-1} |\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle\left|\beta_j^{(s-1)}\right\rangle|0\rangle. \tag{24}$$

(6) *Controlled rotation*. Add an ancillary qubit (register E) and perform an appropriate controlled rotation on the state $|\psi_{s-1}\rangle|0\rangle^E$, transform the system to

$$\left|\phi_2^{(s)}\right\rangle = \frac{1}{\sqrt{k}} \sum_{j=0}^{k-1} |\mathbf{u}_j\rangle|\mathbf{j}\rangle\left|\sigma_j^2\right\rangle\left|\beta_j^{(s-1)}\right\rangle\left|c^{(s)}\beta_j^{(s)}\right\rangle \times \left[\sqrt{1-\left(\frac{c^{(s)}\beta_j^{(s)}}{c}\right)^2}|0\rangle + \frac{c^{(s)}\beta_j^{(s)}}{c}|1\rangle\right]^E. \tag{25}$$

(7) *Uncomputing and measurement*. Uncompute registers B, C, and D, and measure register E to seeing 1, thus, obtain

$$|\psi_{A^{(s)}}\rangle = \sum_{j=0}^{k-1} \beta_j^{(s)}|\mathbf{u}_j\rangle|\mathbf{j}\rangle. \tag{26}$$

B. The complexity of the improved quantum AOP algorithm

We have described an improved quantum AOP algorithm above. In this subsection, we will analyze the time complexity and space complexity of the algorithm.

The time complexity and space complexity of step 1 are $O[\log_2(\epsilon^{-1})\log_2(nk)]$ and $O[\log_2(nk/\epsilon)]$, the same as the DYXL algorithm.

In step 2, similar to the complexity of the step 3 of the DYXL algorithm, the phase estimation stage is of time complexity $O[\frac{1}{\epsilon_1}\text{polylog}(\frac{nm}{\epsilon_1})]$ and space complexity $O[\log_2(1/\epsilon_1)]$ with error ϵ_1. The stage of appending registers $|\frac{1}{\sqrt{k}}\rangle^C|0\rangle^D$ is of time complexity $O[\log_2(1/\epsilon_1)]$ where the number of qubits in registers B, C, and D is $O[\log_2(1/\epsilon_1)]$. Thus, the time complexity of this step is $O[\frac{1}{\epsilon_1}\text{polylog}(\frac{nm}{\epsilon_1})]$.

In step 3, since the time complexity of preparing state $|\psi_{i-1}\rangle$ is much greater than the complexity of stages (i) and (ii) (which is $O[\text{polylog}(1/\epsilon_1)]$ and $O[\log_2(1/\epsilon_1)]$ respectively), we will neglect the complexity of these two stages. In stage (iii), define

$$\begin{aligned} U_{i-1}: U_{i-1}|0\rangle &= \left|\phi_2^{(i)}\right\rangle = \sin(\theta)|a\rangle|0\rangle + \cos(\theta)|b\rangle|1\rangle, \\ S: S_0 &= I - 2|0\rangle^{ABCDE}\langle 0|^{ABCDE}, \\ S_\chi: S_\chi &= I - 2|1\rangle^E\langle 1|^E. \end{aligned}$$

According to the quantum amplitude estimation algorithm [27,28], the unitary operator $Q = -U_{i-1}S_0U_{i-1}^\dagger S_\chi$ acts as a rotation on the two-dimensional space $\text{Span}\{|a\rangle|0\rangle, |b\rangle|1\rangle\}$ with eigenvalues $e^{\pm 2i\theta}$ and corresponding eigenvectors $\frac{|a\rangle|0\rangle \mp |b\rangle|1\rangle}{2}$, thus, the time complexity of U_{i-1} is $O(G_{i-1})$. The quantum amplitude estimation algorithm will generate θ within error ϵ_2, which means that $O[\log_2(1/\epsilon_2)]$ qubits are required to store θ. The corresponding time complexity is $O[\frac{1}{\epsilon_2}(2+\frac{1}{2\eta})G_{i-1}]$, where $1-\eta$ is the probability to success [we could simply choose $\eta = O(1)$] . Finally, we obtain the classical information of $c^{(i)}$ within relative error $O(\kappa^2\epsilon_2)$ (see Appendix D, here we use relative error to ensure that the estimation of $c^{(i)}$ will not be influenced by the scale of $c^{(i)}$.

In step 4, for stage (i), similar to the analysis of the DYXL algorithm (see Appendix A), we want to bind the relative error of $\beta_j^{(i)}$ (denoted as $\tilde{\epsilon}_\beta$) by $O(\epsilon)$. According to Appendix D, $\tilde{\epsilon}_\beta = O(\kappa^2\sqrt{k}\epsilon_1 + \tilde{\epsilon}_c) = O(\kappa^2\sqrt{k}\epsilon_1 + \kappa^2\epsilon_2)$, thus, we can choose $\epsilon_1 = O(\frac{\epsilon}{\kappa^2\sqrt{k}})$ and $\epsilon_2 = O(\frac{\epsilon}{\kappa^2})$ to ensure $\tilde{\epsilon}_\beta = O(\epsilon)$. Since the classical information of $c^{(i)}$ is given by step 3, this stage is of time complexity $O[G_{i-1} + \text{polylog}(\frac{\kappa k}{\epsilon})]$. As for stage (ii), the time complexity of the two cases is $O[\text{polylog}(\frac{\kappa k}{\epsilon})]$. In the worst case [case 2 of stage (ii)], we need an ancillary qubit (register F) in each iteration.

In step 6, the time complexities of the two operations are $O[\text{polylog}(\epsilon_1)] = O[\text{polylog}(\frac{\kappa k}{\epsilon})]$ and $O[\log_2(1/\epsilon)]$, which is similar with stage (i) and stage (ii) of step 3. Note that no extra qubit is needed here since register E is reused.

In step 7, the time complexity of the uncomputing stage is just the same as the time complexity to prepare state $|\phi_2^{(s)}\rangle$ [Eq. (25)] from state $|\psi_{A^{(0)}}\rangle$. For $c^{(s)} = \Omega(k)$ and $c = O(\sqrt{k}\kappa^2)$, the probability of seeing 1 is

$$p(1) = \frac{1}{k}\sum_{j=0}^{k-1}\left(\frac{c^{(s)}}{c}\beta_j^{(s)}\right)^2 = \Omega\left(\frac{1}{\kappa^4}\right). \tag{27}$$

We can use the quantum amplitude amplification [27,28] to reduce the repetition to $O(\kappa^2)$.

Now, we put all together. From the analysis of step 3, we know that the time complexity to estimate $c^{(i)}$, $i = 1, \ldots, s$ is $O_{c^{(i)}} = O(\frac{1}{\epsilon_2}G_{i-1})$. Also, from step 4, given $c^{(i)}$, the time complexity to prepare state $|\psi_i\rangle$ is

$$G_i = G_{i-1} + 2\text{polylog}\left(\frac{\kappa k}{\epsilon}\right). \tag{28}$$

Thus, given $c^{(i)}$ for $i = 1, 2, \ldots, s-1$,

$$G_{s-1} = G_0 + 2(s-1)\text{polylog}\left(\frac{\kappa k}{\epsilon}\right), \tag{29}$$

where $G_0 = \frac{\kappa^2\sqrt{k}}{\epsilon}\text{polylog}(\frac{nm}{\epsilon})$.

The time complexity to obtain state $|\psi_{A^{(s)}}\rangle$ is as follows:

$$\begin{aligned} O_{|\psi_{A^{(s)}}\rangle} &= O\left(\kappa^2 2 O_{|\phi_2^{(s)}\rangle}\right) \\ &= O\left\{\kappa^2\left[\text{polylog}\left(\frac{\kappa k}{\epsilon}\right) + G_{s-1} + \sum_{i=1}^{s-1} c^{(i)}\right]\right\} \\ &= O\left[\frac{s\kappa^6\sqrt{k}}{\epsilon}\text{polylog}\left(\frac{nm}{\epsilon}\right) + \frac{s^2\kappa^4}{\epsilon}\text{polylog}\left(\frac{\kappa k}{\epsilon}\right)\right]. \end{aligned} \tag{30}$$

As for the space complexity, $O[\log_2(nk/\epsilon)]$ qubits are required to obtain the classical information of $c^{(i)}$ in step 3.

Given state $|\psi_{i-1}\rangle$, an ancillary qubit is required to prepare state $|\psi_i\rangle$ in step 4 in iteration i when $\lambda_2 \leqslant 1$. If $\lambda_2 \geqslant 1$, no ancillary qubit is needed. Since this ancillary qubit cannot be reused by the other iterations, $O(s)$ ancillary qubits are required for all iterations. In step 5–step 7, no ancillary qubit is needed. Putting all together, the space complexity of the algorithm is $O[\log_2(nk/\epsilon) + s]$.

Since our algorithm and the DYXL algorithm are iterative algorithms, we compare the number of iterations of the two quantum algorithms with the same loss. In each iteration of the DYXL algorithm, given state $|\psi_{A^{(i-1)}}\rangle$, we can obtain state $|\psi_{A^{(i)}}\rangle$ within a relative error $O(\epsilon)$ in $\beta_j^{(i)}$ (Appendix A). And in each iteration of our algorithm, given $|\psi_{i-1}\rangle$, we obtain state $|\psi_i\rangle$ within a relative error $O(\epsilon)$ in $\beta_j^{(i)}$, the same as the DYXL algorithm. Thus, the two quantum algorithms will converge to the same loss with the same number of iterations.

IV. DISCUSSION

The exponential speedup claimed by Duan *et al.* [26] is based on the assumptions that κ, k, and $1/\epsilon$ are $O[\text{polylog}(nm)]$. We follow these assumptions to compare the two quantum algorithms. The advantage of our algorithm is that the time complexity is quadratically dependent on s, whereas the DYXL algorithm has an exponential dependence on s. If s is a constant, our algorithm has a polynomial speedup over the DYXL algorithm. If s grows linearly with $\log_2(mn)$, our algorithm has exponential speedups on n and m compared with the DYXL algorithm. Since the speedup of our algorithm is strongly dependent on s, we now analyze the value of s below.

Note that the steps of the two quantum algorithms are exactly the same as those of the reformulated AOP algorithm in Sec. II B. Thus, by controlling the precisions of each iteration to the same level, the number of iterations of the quantum algorithms is the same as the reformulated AOP algorithm. We estimate the number of iterations of the reformulated AOP algorithm in Appendix E by numerical experiments on randomly generated datasets since it is difficult to determine the value of s through theoretical analysis. The experimental results show that $s = \Omega[k + \kappa + \log_2(1/\epsilon)]$ may hold. If it holds, s grows linearly with $\log_2(mn)$, which means that the exponential speedups on n and m of our algorithm hold. Note that the experimental results are based on randomly generated original datasets, it cannot rule out the possibility that parameter s may be less in the practical datasets.

V. CONCLUSION

In this paper, we reanalyzed the DYXL algorithm in detail and corrected the complexity calculation. It was shown that the DYXL algorithm has an exponential dependence on the number of iterations s, thus, the quantum algorithm may lose its advantage as s increases. To get a further speedup, we presented an improved quantum AOP algorithm with a time complexity quadratic on s. Our algorithm achieves, at least, a polynomial speedup over the DYXL algorithm. When κ, k, and $1/\epsilon$ are $O[\text{polylog}(nm)]$, our algorithm achieves exponential speedups compared with the classical algorithm on n and m. As for the space complexity, our algorithm is slightly worse than the DYXL algorithm.

The speedups of our algorithm mainly come from the idea of putting the information to be updated into computational basis, which saves the consumption of the current candidates. We hope this idea could inspire more iterative algorithms to get a quantum speedup. We will explore the possibility in the future.

ACKNOWLEDGMENTS

We would like to thank the anonymous referees for their helpful comments. S.-J.P. thanks C.-G. Li, C.-T. Ding, and S.-J. Li for fruitful discussions. This work was supported by the National Natural Science Foundation of China (Grants No. 61672110, No. 61671082, No. 61976024, No. 61972048, and No. 61801126), the Fundamental Research Funds for the Central Universities (Grant No. 2019XDA01), the Open project of CAS Key Laboratory of Quantum Information, University of Science and Technology of China (Grant No. KQI201902) and the Beijing Excellent Talents Training Funding Project (Grant No. 201800002685XG356).

APPENDIX A: THE COMPLEXITY OF EACH STEP OF THE DYXL ALGORITHM

In this Appendix, we analyze the time complexity of each step of the DYXL algorithm in detail. Different from the original paper [26], on one hand, we use the best-known results on Hamiltonian simulation to get a tight bound of the time complexity, on the other hand, we estimate the parameters which have not been estimated in the original paper or need to be correct.

In step 1, $|\psi_{A^{(0)}}\rangle$ can be written in computational basis as $\sum_{x_1x_2\cdots x_l\in\{0,1\}^l} \alpha_{x_1x_2\cdots x_l}|x_1x_2\cdots x_l\rangle$, where $l = k\log_2 n$. By the assumption that the elements of $A^{(0)}$ and $\omega^{(i)}$ are given and stored in quantum random access memory, where $\omega^{(i)}$ is defined as $\cos^2(2\pi\omega_i) = (\frac{\alpha_{x_1x_2\cdots x_{l-1}0}}{x_1x_2\cdots x_{l-1}})^2 + O[\text{poly}(\epsilon)]$, and the time complexity and space complexity for preparing $|\psi_{A^{(0)}}\rangle$ are $O[\log_2(\epsilon^{-1})\log_2(nk)]$ and $O[\log_2(nk/\epsilon)]$ [26].

In step 3, assume the condition number of $\widetilde{X}$ and $A^{(i-1)}$ is κ and $\kappa^{(i-1)}$, then $\kappa^{(i-1)} = O(\kappa^2)$ (the proof is given in Appendix B. We should mention that, in the original paper [26], the author gave a wrong estimate of $\kappa^{(i-1)}$, which influenced the total complexity). Note that

$$\text{tr}_2(|\psi_{A^{(i-1)}}\rangle\langle\psi_{A^{(i-1)}}|) = \sum_{j=1}^{k}\left(\beta_j^{(i-1)}\right)^2|\mathbf{u}_j\rangle\langle\mathbf{u}_j| = A^{(i-1)}A^{(i-1)^\dagger}. \tag{A1}$$

According to Corollary 1 (Corollary 17 of Ref. [29]), the time complexity to simulate $e^{iA^{(i-1)}A^{(i-1)^\dagger}t_0}$ within error ϵ_0 is $O\{[t_0 + \ln(1/\epsilon_0)]G^{(i-1)}\}$, where $G^{(i-1)}$ is the time complexity to prepare state $|\psi_{A^{(i-1)}}\rangle$. For the simulation of $\widetilde{X}\widetilde{X}^\dagger$, assume there is a quantum circuit to prepare state $|\psi_{\widetilde{X}}\rangle = \frac{1}{|\widetilde{X}|_F}\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\widetilde{X}_{ij}|i\rangle|j\rangle = \frac{1}{|\widetilde{X}|_F}\sum_{j=0}^{r-1}\sigma_j|\mathbf{u}_j\rangle|\mathbf{v}_j\rangle$ in time $O[\text{polylog}(nm)]$. Note that $\widetilde{X}\widetilde{X}^\dagger = |\widetilde{X}|_F^2\text{tr}_2(|\psi_{\widetilde{X}}\rangle\langle\psi_{\widetilde{X}}|)$, where $|\widetilde{X}|_F = \sqrt{\sum_{j=0}^{r-1}\sigma_j^2} \leqslant \sqrt{r} = O[\text{polylog}(nm)]$. Thus, the time complexity to simulate $e^{i\widetilde{X}\widetilde{X}^\dagger t_0}$ within ϵ_0 is $O\{\text{polylog}(nm)[|\widetilde{X}|_F^2 t_0 + \ln(1/\epsilon_0)]\}$.

Corollary 1 (Ref. [29]). Given access to the oracle $\hat{G}$ specifying a Hamiltonian $\hat{H} = \hat{\rho}$ that is a density-matrix $\hat{\rho}$, where

$$\begin{aligned} \hat{G}|0\rangle_a = |G\rangle_a = \sum_j \sqrt{\alpha_j}|j\rangle_{a_1}|\chi_j\rangle_{a_2}, \\ \hat{\rho} = \mathrm{tr}|G\rangle\langle G|_{a_1} = \sum_j \alpha_j |\chi_j\rangle\langle\chi_j|, \end{aligned} \tag{A2}$$

time evolution by $\hat{H}$ can be simulated for time t and error ϵ with $O[t + \log_2(1/\epsilon)]$ queries.

According to Refs. [27,30,31], taking $O(1/\epsilon_1)$ times of controlled $e^{iA^{(i-1)}A^{(i-1)\dagger}t_0}$ to perform phase estimation ensures that the singular value $\beta_j^{(i-1)}$ being estimated within error $O(\epsilon_1)$, so as the phase estimation on $e^{i\widetilde{X}\widetilde{X}^\dagger t_0}$. Let $\epsilon_0 = \epsilon_1^2$ and $t_0 = O(1)$, the time complexity of the two phase estimations be $O[(G^{(i-1)}/\epsilon_1)\log_2(1/\epsilon_1)]$ and $O[(1/\epsilon_1)\mathrm{polylog}(nm/\epsilon_1)]$, respectively, whereas the space complexity of these two phase estimations are $O[\log_2(1/\epsilon_1)]$.

The implementation of controlled rotation of step 4 can be divided into two stages [26]. The first stage is a quantum circuit to compute $y_j = \rho\frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{(\sigma_j\beta_j^{(i-1)})^2}$ and store in an auxiliary register L with $O[\ln(1/\epsilon)]$ qubits, where $\rho = O(\frac{1}{2\lambda_2 k\kappa^4})$ (the proof is given in Appendix C, we should mention that in the original paper [26], the authors did not analyze parameter ρ which has a strong correlation with the total complexity). Since $\beta_j^{(i-1)}$ and σ_j is estimated with error $O(\epsilon_1)$, the relative error of estimating y_j is

$$\begin{aligned} \tilde{\epsilon}_y &= O\left(\frac{\lambda_2\left(\beta_j^{(i-1)}\right)^2\sigma_j + \lambda_2\beta_j^{(i-1)}\sigma_j^2}{\left(\beta_j^{(i-1)}\right)^4 + \sigma_j^4 + \lambda_2\left(\beta_j^{(i-1)}\right)^2\sigma_j^2}\epsilon_1\right) \\ &= O\left(\frac{\lambda_2\left(\beta_j^{(i-1)} + \sigma_j\right)\beta_j^{(i-1)}\sigma_j}{(\lambda_2+2)\left(\beta_j^{(i-1)}\right)^2\sigma_j^2}\epsilon_1\right) \\ &= O(\kappa^2\sqrt{k}\epsilon_1), \end{aligned} \tag{A3}$$

in the first equation, and we neglect the terms with the power of ϵ_1 greater than 1, and in the last equation, we use the conclusion of Eq. (C1). To ensure that the final error of this iteration is within $O(\epsilon)$, we could take $\tilde{\epsilon}_y = O(\epsilon)$, which means $\epsilon_1 = O(\frac{\epsilon}{\kappa^2\sqrt{k}})$. Following the result of Ref. [32], the time complexity of this stage is $O[\mathrm{polylog}(1/\epsilon)]$. The second stage is to perform controlled rotation CR on the state, where $CR|y_j\rangle^L|0\rangle^D = |y_j\rangle^L(\sqrt{1-y_j^2}|0\rangle + y_j|1\rangle)^D$. The time complexity of this stage is $O[\log_2(1/\epsilon)]$ [3,9–11,13,14].

In step 5, the probability of seeing 1 in register D is $p(1) = \rho^2\sum_j(\frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{\sigma_j^2\beta_j^{(i-1)}})^2 = O(\frac{1}{\kappa^4})$ as shown in Appendix C (we should mention that in the original paper [26], the authors did not analyze this probability which is directly related to the total complexity). Using amplitude amplification [28], we find that $O(\kappa^2)$ repetitions are sufficient.

APPENDIX B: ESTIMATE THE PARAMETER $\kappa^{(i)}$

In this Appendix, we analyze the condition number $\kappa^{(i)}$ of $A^{(i)}$. First, we give the following theorem:

Theorem 2. If $\beta_j^{(i-1)} > \beta_{j'}^{(i-1)}$, then $\beta_j^{(i)} > \beta_{j'}^{(i)}$.

Proof. (1) Note that $\beta_0^{(0)} = \beta_1^{(0)} = \cdots = \beta_{k-1}^{(0)}$ and $1 = \sigma_0 \geqslant \sigma_1 \geqslant \cdots \geqslant \sigma_{k-1} = \frac{1}{\kappa}$. For $j > j'$, the following inequalities hold:

$$\begin{aligned} \beta_j^{(0)} &\geqslant \beta_{j'}^{(0)}, \\ \sigma_j^2\beta_j^{(0)} &\leqslant \sigma_{j'}^2\beta_{j'}^{(0)}. \end{aligned} \tag{B1}$$

(2) Assuming that for $i \geqslant 1$, the following inequalities hold:

$$\begin{aligned} \beta_j^{(i-1)} &\geqslant \beta_{j'}^{(i-1)}, \\ \sigma_j^2\beta_j^{(i-1)} &\leqslant \sigma_{j'}^2\beta_{j'}^{(i-1)}. \end{aligned} \tag{B2}$$

Then,

$$\begin{aligned} \sigma_j^2\beta_j^{(i)} &= \sigma_j^2\frac{\left(\sigma_j\beta_j^{(i-1)}\right)^2 + \lambda_2}{c^{(i-1)}\sigma_j^2\beta_j^{(i-1)}} \\ &= \frac{1}{c^{(i-1)}}\sigma_j^2\beta_j^{(i-1)} + \frac{\lambda_2}{c^{(i-1)}\beta_j^{(i-1)}} \\ &\leqslant \frac{1}{c^{(i-1)}}\sigma_{j'}^2\beta_{j'}^{(i-1)} + \frac{\lambda_2}{c^{(i-1)}\beta_{j'}^{(i-1)}} \\ &= \sigma_{j'}^2\beta_{j'}^{(i)}. \end{aligned} \tag{B3}$$

Also,

$$\begin{aligned} \beta_j^{(i)} &= \frac{\left(\sigma_j\beta_j^{(i-1)}\right)^2 + \lambda_2}{c^{(i-1)}\sigma_j^2\beta_j^{(i-1)}} \\ &= \frac{1}{c^{(i-1)}}\left(\beta_j^{(i-1)} + \frac{\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}\right) \\ &\geqslant \frac{1}{c^{(i-1)}}\left(\beta_{j'}^{(i-1)} + \frac{\lambda_2}{\sigma_{j'}^2\beta_{j'}^{(i-1)}}\right) \\ &= \beta_{j'}^{(i)}. \end{aligned} \tag{B4}$$

Thus, the theorem holds. ∎

According to Theorem 2, we have as follows:

$$\begin{aligned} \max_j \beta_j^{(i)} &= \frac{\max_j\left(\beta_j^{(i-1)}\right)^2 + \lambda_2\kappa^2}{c^{(i-1)}\max_j\beta_j^{(i-1)}}, \\ \min_j \beta_j^{(i)} &= \frac{\min_j\left(\beta_j^{(i-1)}\right)^2 + \lambda_2}{c^{(i-1)}\min_j\beta_j^{(i-1)}}. \end{aligned} \tag{B5}$$

Let $\max_j(\beta_j^{(i)})^2 = a^{(i)} \leqslant 1$, thus,

$$\begin{aligned} \kappa^{(i)} &= \frac{\max_j\beta_j^{(i)}}{\min_j\beta_j^{(i)}} \\ &= \frac{\max_j\left(\beta_j^{(i-1)}\right)^2 + \lambda_2\kappa^2}{c^{(i-1)}\max_j\beta_j^{(i-1)}}\frac{c^{(i-1)}\min_j\beta_j^{(i-1)}}{\min_j\left(\beta_j^{(i-1)}\right)^2 + \lambda_2} \\ &= \frac{(a^{(i-1)} + \lambda_2\kappa^2)\kappa^{(i-1)}}{a^{(i-1)} + \lambda_2(\kappa^{(i-1)})^2}. \end{aligned} \tag{B6}$$

Note that $\beta_j^{(0)}$ takes the same value for $j = 0, 1, \ldots k-1$, thus, $\kappa^{(0)} = 1$, $\kappa^{(1)} = \frac{a^{(0)}+\lambda_2\kappa^2}{a^{(0)}+\lambda_2}$. According to Eq. (B6), the

following equation holds:

$$\kappa^{(i-1)}\kappa^{(i)} = \frac{(a^{(i-1)} + \lambda_2\kappa^2)(\kappa^{(i-1)})^2}{a^{(i-1)} + \lambda_2(\kappa^{(i-1)})^2}. \tag{B7}$$

According to Eqs. (B6) and (B7), we have as follows:

$$\begin{aligned} \kappa^{(i)}\kappa^{(i-1)} &\geqslant \kappa^2, \quad \text{if } \kappa^{(i-1)} \geqslant \kappa, \\ \kappa^{(i)}\kappa^{(i-1)} &\leqslant \kappa^2, \quad \text{if } \kappa^{(i-1)} \leqslant \kappa, \end{aligned} \tag{B8}$$

Since $\kappa = O[\text{polylog}(NM)]$, $\lambda_2 = O(1)$, we assume that $\lambda_2\kappa > 1$, thus, $\kappa^{(0)} = 1 \leqslant \kappa$, $\kappa^{(1)} = \frac{a^{(0)}+\lambda_2\kappa^2}{a^{(0)}+\lambda_2} \geqslant \kappa$. Then,

$$\begin{aligned} \kappa^{(2l)}\kappa^{(2l+1)} &\leqslant \kappa^2 \\ \kappa^{(2l+1)}\kappa^{(2l+2)} &\geqslant \kappa^2. \end{aligned} \tag{B9}$$

Furthermore,

$$\begin{aligned} 1 = \kappa^{(0)} &\leqslant \kappa^{(2)} \leqslant \cdots \leqslant \kappa^{(2l)} \leqslant \kappa \\ &\leqslant \kappa^{(2l+1)} \leqslant \cdots \leqslant \kappa^{(3)} \leqslant \kappa^{(1)} \\ &= \frac{a^{(0)+\lambda_2\kappa^2}}{a^{(0)} + \lambda_2} = O(\kappa^2). \end{aligned} \tag{B10}$$

The sequence $\kappa^{(2l)}$ ($l = 0, 1, 2, \ldots, \infty$) is monotonically increasing with upper bound κ, and the sequence $\kappa^{(2l+1)}$ ($l = 0, 1, 2, \ldots \infty$) is monotonically decreasing with lower bound κ, thus, the sequence $\kappa^{(i)}$ ($i = 0, 1, 2, \ldots, \infty$) converges on κ, i.e., $\lim_{i\to\infty} \kappa^{(i)} = \kappa$.

In conclusion, we get $\kappa^{(i)} = O(\kappa^2)$ for $i \geqslant 0$.

APPENDIX C: ESTIMATE THE PARAMETER ρ AND THE $p(1)$ OF STEP 5 IN THE DYXL ALGORITHM

In this Appendix, we analyze the value of parameter ρ which first appeared in step 4 of the DYXL algorithm.

It is obvious that $\sum_{j=1}^{k}(\beta_j^{i-1})^2 = 1$ for $\beta_j^{(i-1)}$ ($j = 0, 1, \ldots, k-1$) is the amplitude of quantum state $|\psi_A^{(i-1)}\rangle$. Thus,

$$\begin{aligned} \frac{1}{\sqrt{k}} &\leqslant \max_j \beta_j^{(i-1)} \leqslant 1, \\ \frac{1}{\kappa^{i-1}\sqrt{k}} &\leqslant \min_j \beta_j^{(i-1)} \leqslant \frac{1}{\sqrt{k}}. \end{aligned} \tag{C1}$$

Since $\rho\frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{(\sigma_j\beta_j^{(i-1)})^2} \leqslant 1$, $\rho \leqslant \min_j \frac{(\sigma_j\beta_j^{(i-1)})^2}{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}$. Note that

$$\begin{aligned} \min_j \frac{\left(\sigma_j\beta_j^{(i-1)}\right)^2}{\left(\sigma_j\beta_j^{(i-1)}\right)^2 + \lambda_2} &= \min_j \frac{\beta_j^{(i-1)}}{\beta_j^{(i-1)} + \frac{\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}} \geqslant \frac{\min_j \beta_j^{(i-1)}}{\max_j\left(\beta_j^{(i-1)} + \frac{\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}\right)} = \frac{\min_j \beta_j^{(i-1)} \max_j \beta_j^{(i-1)}}{\max_j\left(\beta_j^{(i-1)}\right)^2 + \lambda_2\kappa^2} \geqslant \frac{1/(\kappa^{(i-1)}k)}{1 + \lambda_2\kappa^2} \\ &= \frac{1}{\kappa^{(i-1)}k + \lambda_2 k\kappa^2\kappa^{(i-1)}} > \frac{1}{2\lambda_2 k\kappa^2\kappa^{(i-1)}} \end{aligned} \tag{C2}$$

for the $\kappa^{(i)} = O(\kappa^2)$, we could choose the parameter $\rho = O(\frac{1}{2\lambda_2 k\kappa^4})$. Then the probability of seeing 1 in register D in step 5 is

$$\begin{aligned} p(1) &= \rho^2 \sum_j \left(\frac{(\sigma_j\beta_j^{(i-1)})^2 + \lambda_2}{\sigma_j^2\beta_j^{(i-1)}}\right)^2 \leqslant \rho^2 k\left[\frac{\left(\frac{1}{\kappa}\max_j \beta_j^{(i-1)}\right)^2 + \lambda_2}{\frac{1}{\kappa^2}\max_j \beta_j^{(i-1)}}\right]^2 \leqslant \rho^2 k\left(\frac{1 + \lambda_2\kappa^2}{\max_j \beta_j^{(i-1)}}\right)^2 \leqslant \rho^2 k^2(1 + \lambda_2\kappa^2)^2 \\ &\leqslant \left(\frac{1}{2\lambda_2 k\kappa^4}\right)^2 k^2(1 + \lambda_2\kappa^2)^2 = O\left(\frac{1}{\kappa^4}\right). \end{aligned} \tag{C3}$$

APPENDIX D: ESTIMATE THE PARAMETER $c^{(i)}$ AND ANALYZE THE RELATIVE ERROR OF $c^{(i)}$ AND $\beta_j^{(i)}$

In this Appendix, we analyze the value of parameter $c^{(i)}$ and θ first and then analyze the relative errors of $c^{(i)}$ and $\beta_j^{(i)}$ of our algorithm.

Note that

$$c^{(i)}\beta_j^{(i)} = \frac{\left(\sigma_j\beta_j^{(i-1)}\right)^2 + \lambda_2}{\sigma_j^2\beta_j^{(i-1)}} = \beta_j^{(i-1)} + \frac{\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}. \tag{D1}$$

According to Theorem 2, we have

$$c^{(i)}\beta_j^{(i)} \geqslant \min_j c^{(i)}\beta_j^{(i)} = \min_j\left(\beta_j^{(i-1)} + \frac{\lambda_2}{\sigma_j^2\beta_j^{(i-1)}}\right) = \min_j \beta_j^{(i-1)} + \frac{\lambda_2}{\min_j \beta_j^{(i-1)}}. \tag{D2}$$

Similarly, we have $c^{(i)}\beta_j^{(i)} \leqslant \max_j \beta_j^{(i-1)} + \frac{\lambda_2\kappa^2}{\max_j \beta_j^{(i-1)}}$. Because $c > c^{(i)}\beta_j^{(i)}$, we can choose $c \geqslant \max_j \beta_j^{(i-1)} + \frac{\lambda_2\kappa^2}{\max_j \beta_j^{(i-1)}}$. Since

$$\max_j \beta_j^{(i-1)} + \frac{\lambda_2\kappa^2}{\max_j \beta_j^{(i-1)}} \leqslant 1 + \lambda_2\sqrt{k}\kappa^2, \tag{D3}$$

we can take $c = 1 + \lambda_2\sqrt{k}\kappa^2 = O(\sqrt{k}\kappa^2)$.

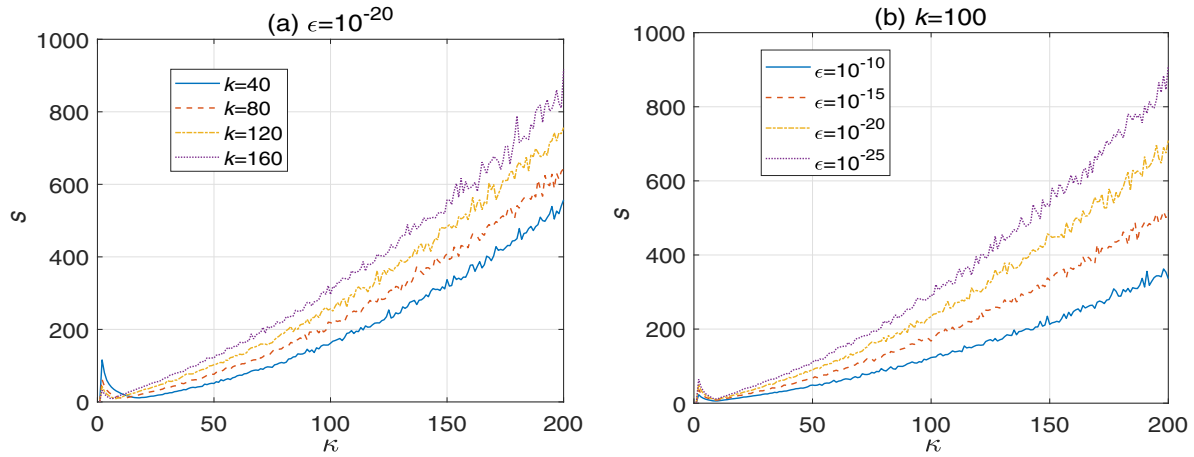

FIG. 1. The relationship of the condition number κ and the number of iterations s when $\epsilon = 10^{-20}$ (a) and $k = 100$ (b), respectively.

Based on Eq. (C1),

$$(c^{(i)})^2 = (c^{(i)})^2 \sum_j \left(\beta_j^{(i)}\right)^2 = \sum_j \left(\beta_j^{(i-1)} + \frac{\lambda_2}{\sigma_j^2 \beta_j^{(i-1)}}\right)^2$$
$$= \sum_j \left(\left(\beta_j^{(i-1)}\right)^2 + \frac{2\lambda_2}{\sigma_j^2} + \left(\frac{\lambda_2}{\sigma_j^2 \beta_j^{(i-1)}}\right)^2\right) \geqslant 1 + 2k\lambda_2 + \lambda_2^2 k^2 = \Omega(k^2), \tag{D4}$$

thus, $c^{(i)} = \Omega(k)$. Then according to Eq. (19),

$$\sin(\theta) = \frac{c^{(i)}}{c\sqrt{k}} = \Omega\left(\frac{k}{k\kappa^2}\right) = \Omega\left(\frac{1}{\kappa^2}\right). \tag{D5}$$

The relative error of $c^{(i)}$ is

$$\tilde{\epsilon}_c = \left|\frac{c^{(i)} - \tilde{c}^{(i)}}{c^{(i)}}\right| = \left|\frac{c\sqrt{k}\sin(\theta) - c\sqrt{k}\sin(\tilde{\theta})}{c\sqrt{k}\sin(\theta)}\right| \leqslant \left|\frac{\Delta\theta}{\sin(\theta)}\right| = O(\kappa^2\epsilon_2). \tag{D6}$$

Thus, the relative error of $\beta_j^{(i)}$ of step 4 is

$$\left|\frac{\beta_j^{(i)} - \tilde{\beta}_j^{(i)}}{\beta_j^{(i)}}\right| = \left|\frac{\frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{c^{(i)}\sigma_j^2\beta_j^{(i-1)}} - \frac{(\tilde{\sigma}_j\tilde{\beta}_j^{(i-1)})^2+\lambda_2}{\tilde{c}^{(i)}\tilde{\sigma}_j^2\tilde{\beta}_j^{(i-1)}}}{\frac{(\sigma_j\beta_j^{(i-1)})^2+\lambda_2}{c^{(i)}\sigma_j^2\beta_j^{(i-1)}}}\right| = O\left[\left|\frac{\sigma_j^2\lambda_2 + 2\sigma_j\beta_j^{(i-1)}\lambda_2 - \sigma_j^4\left(\beta_j^{(i-1)}\right)^2}{\sigma_j^4\left(\beta_j^{(i-1)}\right)^3 + \sigma_j\left(\beta_j^{(i-1)}\right)\lambda_2}\right|\epsilon_1 + \tilde{\epsilon}_c\right]$$
$$= O\left(\frac{\sigma_j\lambda_2 + 2\beta_j^{(i-1)}\lambda_2}{\sigma_j\left(\beta_j^{(i-1)}\right)\lambda_2}\epsilon_1 + \tilde{\epsilon}_c\right)$$
$$= O(\kappa^2\sqrt{k}\epsilon_1 + \tilde{\epsilon}_c), \tag{D7}$$

where we neglect the quadratic terms of ϵ_1 in the second equation and ϵ_1 is the absolute error of β_j and σ_j.

APPENDIX E: THE NUMBER OF ITERATIONS OF THE REFORMULATED AOP ALGORITHM

In this Appendix, we evaluate the number of iterations s of the reformulated AOP algorithm through numerical experiments on randomly generated datasets with respect to three parameters, i.e., the condition number κ of $\widetilde{X}$, the precision ϵ, and the number of the principal components k of $\widetilde{X}$ (or the dimensionality of the reduced feature space). By analyzing the steps of the algorithm, we find that the parameters n and m (the dimensionality of $\widetilde{X}$) will not influence the number of iteration directly.

We evaluate the relationship of s and κ first. Note that the first step of the reformulated AOP algorithm is computing the PCA of the data matrix $\widetilde{X}$ and then selecting the k largest eigenvalues (i.e., the $\sigma_0^2, \sigma_1^2, \ldots, \sigma_{k-1}^2$) to perform the later steps. To reduce the running time, we randomly generate k positive numbers as the k largest eigenvalues of $\widetilde{X}$ to remove the process of PCA. The experimental

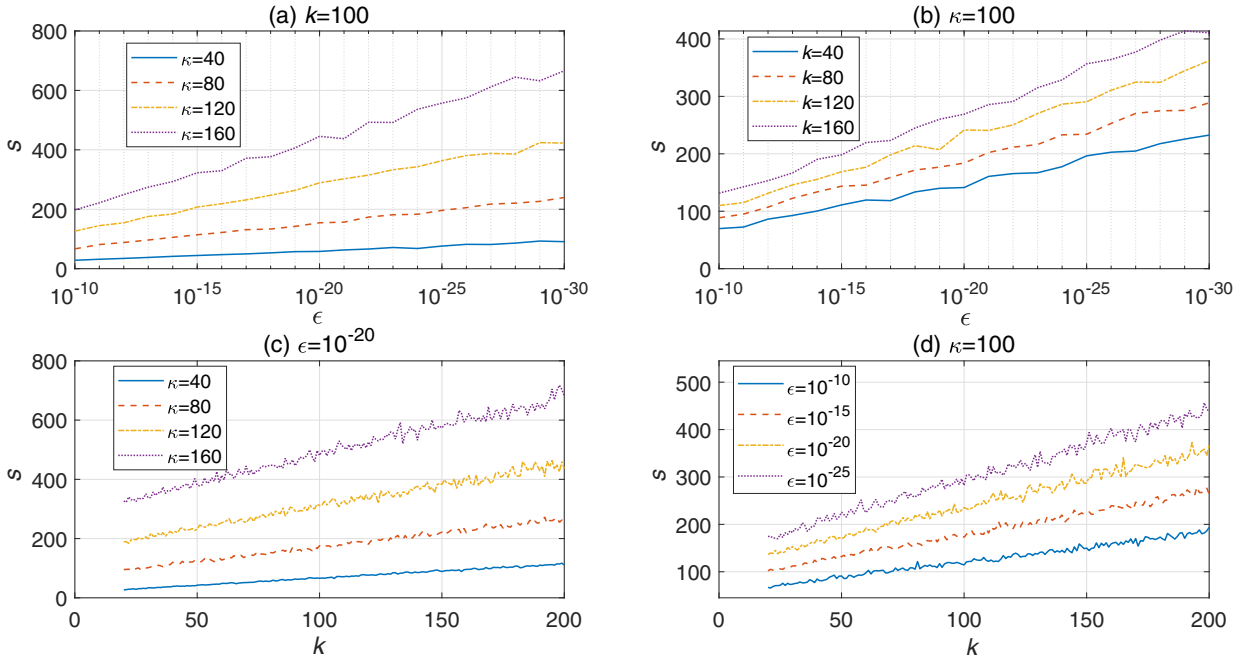

FIG. 2. Evolution of the number of iterations s with respect to the parameter ϵ [(a) and (b)] and k [(c) and (d)].

results are shown in Fig. 1. We set $\epsilon = 10^{-20}$ and $k = \{40, 80, 120, 160\}$ in Fig. 1(a) and set $k = 100$ and $\epsilon = \{10^{-10}, 10^{-15}, 10^{-20}, 10^{-25}\}$ in Fig. 1(b). The experimental results show that s may have a superlinear dependence on κ when $\kappa \geqslant 20$. Note that the exponential speedups of the two quantum algorithms are based on $\kappa = O[\text{polylog}(mn)]$ and the quantum algorithms have advantages when n and m are large. Thus, the situation that the value of κ is large, i.e., $\kappa \geqslant 20$, deserves more attention.

The relationship of ϵ and s is shown in Figs. 2(a) and 2(b), and the relationship of k and s is shown in Figs. 2(c) and 2(d). The experimental results in Fig. 2 show that s may be linearly dependent on $\log_2(1/\epsilon)$ and k. Put all the three parameters together, $s = \Omega[\kappa + \log_2(1/\epsilon) + k]$ may hold.

[1] P. W. Shor, *Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Sannta Fe, NM, 1994* (IEEE Computer Society, Los Alamitos, CA, 1994), pp. 124–134.

[2] L. K. Grover, *STOC 96: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, 1996* (ACM, New York, 1996), p.212.

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

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

[5] S. Lloyd, M. Mohseni, and P. Rebentrost, arXiv:1307.0411.

[6] P. Rebentrost, M. Mohseni, and S. Lloyd, Phys. Rev. Lett. **113**, 130503 (2014).

[7] I. Cong and L. Duan, New J. Phys. **18**, 073011 (2016).

[8] M. Schuld, M. Fingerhuth, and F. Petruccione, Europhys Lett. **119**, 60002 (2017).

[9] B. Duan, J. Yuan, Y. Liu, and D. Li, Phys. Rev. A **96**, 032301 (2017).

[10] N. Wiebe, D. Braun, and S. Lloyd, Phys. Rev. Lett. **109**, 050505 (2012).

[11] M. Schuld, I. Sinayskiy, and F. Petruccione, Phys. Rev. A **94**, 022342 (2016).

[12] G. Wang, Phys. Rev. A **96**, 012335 (2017).

[13] C.-H. Yu, F. Gao, and Q.-Y. Wen, IEEE Transactions on Knowledge and Data Engineering 1 (2019).

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

[15] C.-H. Yu, F. Gao, Q.-L. Wang, and Q.-Y. Wen, Phys. Rev. A **94**, 042311 (2016).

[16] N. Liu and P. Rebentrost, Phys. Rev. A **97**, 042315 (2018).

[17] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag, New York, 2006).

[18] H. Hotelling, Biometrika **28**, 321 (1936).

[19] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, Science **290**, 2319 (2000).

[20] S. T. Roweis and L. K. Saul, Science **290**, 2323 (2000).

[21] M. Belkin and P. Niyogi, *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, edited by T. G. Dietterich, S. Becker, and Z. Ghahramani (MIT, Cambridge, MA, 2002), p. 585.

[22] R. A. Fisher, Ann. Eugenics **7**, 179 (1936).

[23] X. He, C. Zhang, L. Zhang, and X. Li, IEEE Trans. Pattern Anal. Mach. Intell. **38**, 1009 (2015).

[24] S. Lloyd, M. Mohseni, and P. Rebentrost, Nat. Phys. **10**, 631 (2014).
[25] C.-H. Yu, F. Gao, S. Lin, and J. Wang, Quantum Inf. Process. **18**, 249 (2019).
[26] B. Duan, J. Yuan, J. Xu, and D. Li, Phys. Rev. A **99**, 032311 (2019).
[27] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2002).
[28] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp, Contemporary Mathematics **305**, 53 (2002).
[29] G. H. Low and I. L. Chuang, Quantum **3**, 163 (2019).
[30] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, Proc. R. Soc. London, Ser. A **454**, 339 (1998).
[31] A. Luis and J. Peřina, Phys. Rev. A **54**, 4564 (1996).
[32] Y. Cao, A. Papageorgiou, I. Petras, J. Traub, and S. Kais, New J. Phys. **15**, 013021 (2013).