Physics Letters A 383 (2019) 2862–2868

Contents lists available at ScienceDirect

Physics Letters A

www.elsevier.com/locate/pla

An improved quantum principal component analysis algorithm based on the quantum singular threshold method

Jie Lin [a,b], Wan-Su Bao [a,b,*], Shuo Zhang [a,b], Tan Li [a,b], Xiang Wang [a,b]

[a] Henan Key Laboratory of Quantum Information and Cryptography, PLA SSF IEU, Zhengzhou, Henan 450001, China
[b] Synergetic Innovation Center of Quantum Information and Quantum Physics, University of Science and Technology of China, Hefei, Anhui 230026, China

ARTICLE INFO

Article history:
Received 30 April 2019
Received in revised form 14 June 2019
Accepted 15 June 2019
Available online 20 June 2019
Communicated by M.G.A. Paris

Keywords:
Principle component analysis
Singular value threshold method

ABSTRACT

Quantum principal component analysis (qPCA) is a dimensionality reduction algorithm for getting the eigenvectors corresponding to top several eigenvalues of the data matrix and then reconstructing. However, qPCA can only construct the quantum state contains all the eigenvectors and eigenvalues. In this paper, we present an improved quantum principal component analysis (Improved qPCA) algorithm with a fixed threshold. We can reduce the singular value less than the threshold to 0 and obtain a target quantum state which can be used to get an output similar to qPCA after phase estimation. Compared with qPCA, our algorithm has only the target eigenvalues and the probability that we get each target eigenvalue is greater. Furthermore, our algorithm can serve as an additional regularization method and a subroutine for subsequent data processing.

1. Introduction

The goal of dimensionality reduction algorithms [1–3] is to compress high dimensional data into lower dimensional data such that main properties of data are preserved. Principal component analysis (PCA) [4–6] is a widely used dimensionality reduction algorithm in many aspects which aims at preserving the global variance. The algorithm is used to solve an eigenvalue problem of the covariance matrix and select eigenvectors corresponding to the target eigenvalues to construct transformation matrix.

Quantum computing has received attention due to its parallel advantage which helps a range of classical algorithms to achieve a speedup increase [7–14]. qPCA [15] offers a speedup over the PCA by solving an eigenvalue problem of the unknown M-dimensional density matrix with rank $T \ll M$. Note that the matrix is sparse or low-rank. The algorithm outputs a quantum state that contains all the eigenvalues and the solution contains top-r eigenvalues is extracted by sampling. This quantum state can be useful in solving a particular problem about large eigenvalues. However, the cost of qPCA may be expensive when the eigenvalues are more average. For instance, the eigenvalues satisfy $\lambda_1 > \cdots > \lambda_T$ and $|\lambda_{r-i} - \lambda_T| < \zeta$ where $r, i \in Z^+$ and $\lambda_1, \cdots, \lambda_r$ is a solution. qPCA constructs a quantum state $\sum_{k=1}^{T} \lambda_k |u_k\rangle\langle u_k| \otimes |\tilde{\lambda_k}\rangle\langle\tilde{\lambda_k}|$ where $u_k, k = 1, \cdots, T$ are eigenvectors. In this case, qPCA may calculate all the eigenvalues to distinguish top-r eigenvalues in time $O(Tploy(logM))$ when ζ is small. Furthermore, for a sparse and high-rank matrix ($T \sim O(M)$ and $r \ll T \leq M$), qPCA may provide a method with the time complexity $O(Mploy(logM))$.

For this result, we propose the Improved qPCA algorithm based on QSVT algorithm. As in the case of quantum linear equations algorithm [16], our method yields a quantum state which contains only all eigenvectors corresponding to the target eigenvalues. Assume $R_{CCR} = (\sum_{k=1}^{r} \lambda_k / \sum_{k=1}^{T} \lambda_k)$, we usually have a cumulative contribution rate (CCR) condition: $R_{CCR} \geq l$ where $l \in [0, 1]$. For each target eigenvalue, the probability of our algorithm is increased by $\sum_{k=1}^{T} \lambda_k^2 / \sum_{k=1}^{r} \lambda_k^2$ times at one measurement. It allows us to perform Improved qPCA to reconstruct data with fewer samples. Suppose $r \ll T$, we perform Improved qPCA to get the solution in polynomial time $O(rpoly(logM))$. In addition, the part of our algorithm is actually an optimal matrix low-rank approximation in quantum form which helps scale kernel-based algorithms [17–19] to large-scale datasets. The quantum state generated by our algorithm has other function [20–23] such as acting on the Nyström approximation. The remainder of the paper is organized as follows: We describe the QSVT algorithm and give a brief overview of the Improved qPCA algorithm in Sect. 2, analyze the complexity and fidelity and discuss the Improved qPCA algorithm in Sect. 3, and present our conclusion in Sect. 4.

* Corresponding author at: Zhengzhou Information Science and Technology Institute, Zhengzhou, 450001, China.
E-mail address: bws@qiclab.cn (W.-S. Bao).

https://doi.org/10.1016/j.physleta.2019.06.026
0375-9601/

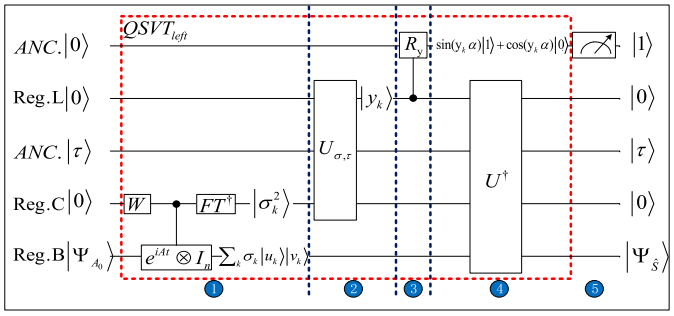

Fig. 1. Quantum circuit overview of QSVT algorithm [25]. The circuit model is divided into five parts. The first part is the phase estimation, the second part is the threshold judgment, the third part is to assign the quantum state to the amplitude, the fourth part is the inverse transformation of the first part, and the last part is the measurement operation. Parts 1 through 4 are named $QSVT_{left}$ and part 5 is named $QSVT_{right}$.

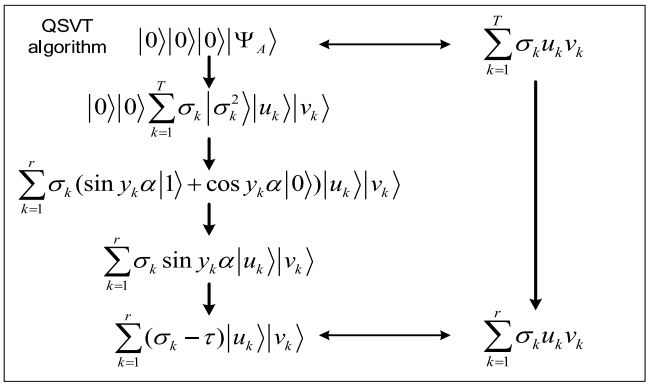

Fig. 2. Entire process of QSVT algorithm and classical input and output corresponding to the algorithm.

2. The QSVT and Improved qPCA algorithm

In this section, we briefly describe the algorithmic procedures for the QSVT algorithm [24–26] and design a modified quantum singular value thresholding algorithm (MQSVT). On this basis, we present a improved qPCA algorithm. We first show the connection of QSVT algorithm and our algorithm. Then we use the given parameters to introduce our algorithm.

2.1. QSVT algorithm and MQSVT algorithm

We now describe the basic notation. Give a matrix $A \in R^{M\times N}$ with rank $T \le min(M,N)$, $A_{i,j}$ represents the elements of the i-th row and the j-th column of A. We define the thin singular value decomposition (SVD) of A as $A = U\Sigma V = \sum_{k=1}^{T} \sigma_k u_k v_k$ where u_k, v_k represent left and right singular value vectors respectively. The optimal r-rank approximation of A is $A = \sum_{k=1}^{r} \sigma_k u_k v_k$ where $r < T$. Suppose $Ker = AA^T \in R^{M\times M}$ and $\lambda_k = \sigma_k^2$, $\lambda_k, k = 1, \cdots, T$ are eigenvalues of Ker.

As shown in Fig. 1 and Fig. 2, the QSVT algorithm takes $A = \sum_{k=1}^{T} \sigma_k u_k v_k$ as a input to get $\sum_{k=1}^{r} (\sigma_k - \tau) u_k v_k$ in quantum form where $d \ge r$ and τ is a threshold. In initialization, we store the matrix elements in register B via qRAM. The first step is to store the singular value information in register C through quantum phase estimation. In the second step, the algorithm uses register L to store $y_k = 1 - \tau/\sigma_k$ through judgment operation with a fixed threshold. The information on register L is assigned to the amplitude of the auxiliary bit through rotation transformation in the third step. In the fourth step the algorithm performs an unitary transformation $U^\dagger$ to eliminate unwanted registers. Finally, the measurement is carried out and register B collapses correspondingly to the target quantum state. The output of QSVT algorithm is very close to the desired result. The difference is the existence of threshold which gives us a revelation about qPCA algorithm. We can get the target quantum state simply by separating out the threshold term. To achieve this goal, we design an modified quantum circuit model about the second and third parts of the QSVT algorithm [27].

We modify the QSVT circuit model in Fig. 3. In the second part, we improve the adder of QSVT circuit model [25] and transform $y_k = 1 - \tau/\sigma_k$ to $y'_k = 1 + \tau/\sigma_k$ in register L. In the third part we change the parameter α of the rotation transformation to $\alpha/2$ and the information assigned on the auxiliary bit amplitude is $sin\frac{(1+\tau/\sigma_k)\alpha}{2}$. Finally, we perform the MQSVT algorithm to output a result $\sum_{k=1}^{r} \sigma_k sin(\frac{y'_k \alpha}{2}) |u_k\rangle |v_k\rangle$.

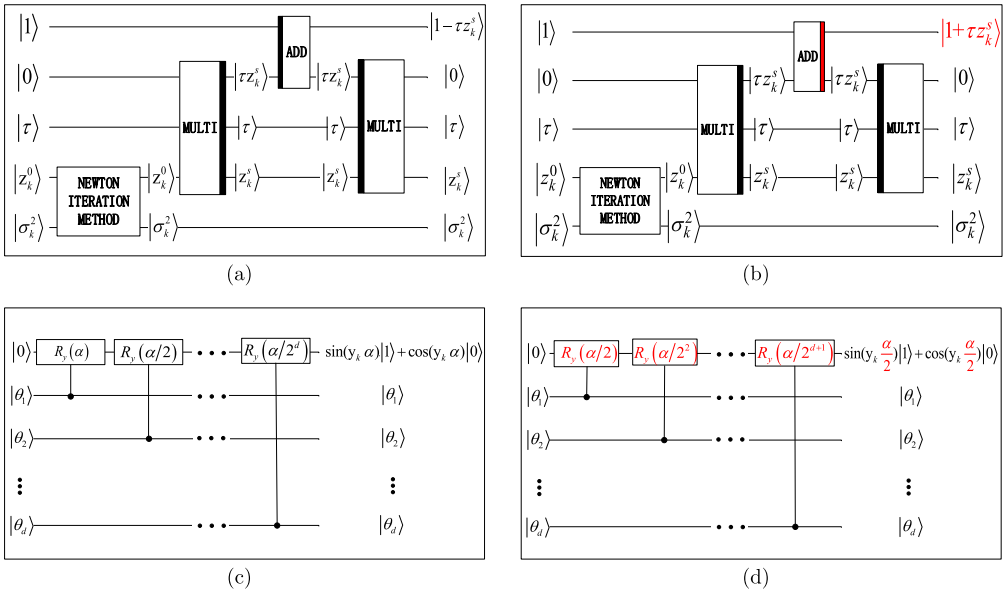

Fig. 3. (a)(c) Quantum circuit overview of QSVT algorithm in part 2 and part 3. (b)(d) Quantum circuit model of MQSVT algorithm in part 2 and part 3. Note that the rest of MQSVT algorithm is consistent with QSVT algorithm. Similarly, parts 1 through 4 are named $MQSVT_{left}$ and part 5 is named as $MQSVT_{right}$.

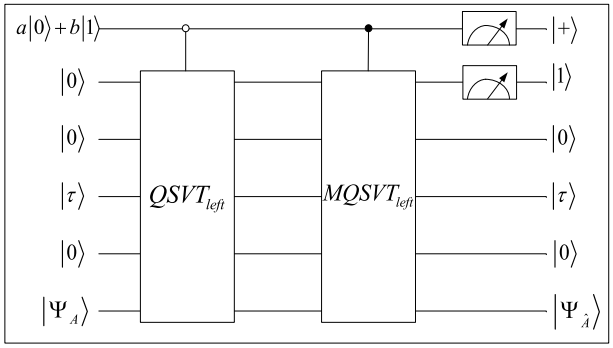

Fig. 4. Quantum circuit overview of Improved qPCA algorithm. The white node indicates that the $QSVT_{left}$ is performed when the control qubit is at 0. Otherwise, we perform the transformation $MQSVT_{left}$ when the control bit is at 1.

2.2. Improved qPCA algorithm

Based on algorithms QSVT and MQSVT, we propose a Improved qPCA algorithm with a fixed threshold and show the circuit model. It is seen from Fig. 4 that we construct an auxiliary bit to control these two quantum algorithms. We initialize the data via the method outlined in Ref. [25] in this algorithm. The matrix A is loaded a quantum state

$$|\Psi_A\rangle = \sum A_{i,j}|i\rangle|j\rangle = \sum_{k=1}^{T}\sigma_k|u_k\rangle|v_k\rangle. \tag{1}$$

A core operation is the construction of a unitary matrix in phase estimation. The matrix $tr_2|\Psi_A\rangle\langle\Psi_A|$ is constructed by $|\Psi_A\rangle$ where tr_2 is a partial trace and Ker is proportional to this matrix. And we use the methods of qPCA [16] to construct unitary operation e^{iKert}. Then we can construct the unitary operation [28] corresponding to the first part of QSVT algorithm

$$U_{PE}(Ker) = (F_T^{\dagger}\otimes I)(\sum_{\tau=0}^{T_d-1}|\tau\rangle\langle\tau|\otimes e^{iKer\tau t/T_d})(H^{\otimes t_d}\otimes I), \tag{2}$$

where t_d is the number of qubits with eigenvalues. In the third part of algorithms QSVT and MQSVT, when $\sigma_k > \tau$, the unitary $U_{\sigma,\tau}$ and $U'_{\sigma,\tau}$ can be constructed to get

$$\begin{aligned} U_{\sigma,\tau}|0\rangle|\tau\rangle|\sigma_k^2\rangle &= |y_k\rangle|\tau\rangle|\sigma_k^2\rangle \\ U'_{\sigma,\tau}|0\rangle|\tau\rangle|\sigma_k^2\rangle &= |y'_k\rangle|\tau\rangle|\sigma_k^2\rangle. \end{aligned} \tag{3}$$

Then we construct the unitary transformation

$$PO_{\sigma,\tau} = |0\rangle\langle 0|\otimes I\otimes U_{\sigma,\tau} + |1\rangle\langle 1|\otimes I\otimes U'_{\sigma,\tau}. \tag{4}$$

Our algorithm sets $a = 1/\sqrt{5}$ and $b = 2/\sqrt{5}$ for the auxiliary control bit as shown in Fig. 4. The overall procedure of Improved qPCA algorithm includes the following steps:

Step 1: Initialization. The initial quantum state is prepared for

$$|\psi_1\rangle = (\frac{1}{\sqrt{5}}|0\rangle + \frac{2}{\sqrt{5}}|1\rangle)|0\rangle|0\rangle|0\rangle|\Psi_A\rangle. \tag{5}$$

Step 2: Phase estimation. Perform unitary operation $U = I\otimes U_{PE}(Ker)$ on $|\psi_1\rangle$ which leads to

$$|\psi_2\rangle = (\frac{1}{\sqrt{5}}|0\rangle + \frac{2}{\sqrt{5}}|1\rangle)|0\rangle|0\rangle\sum_{k=1}^{T}\sigma_k\left|\sigma_k^2\right\rangle|u_k\rangle|v_k\rangle. \tag{6}$$

Step 3: Transformations on different values of auxiliary control qubit. If $\sigma_k > \tau$, we perform the unitary operation $PO_{\sigma,\tau}$ to obtain the state

$$\begin{aligned} |\psi_3\rangle =& \frac{1}{\sqrt{5}}|0\rangle|0\rangle\sum_{k=1}^{r}|y_k\rangle\sigma_k\left|\sigma_k^2\right\rangle|u_k\rangle|v_k\rangle + \frac{2}{\sqrt{5}}|1\rangle|0\rangle \\ &\times\sum_{k=1}^{r}\left|y'_k\right\rangle\sigma_k\left|\sigma_k^2\right\rangle|u_k\rangle|v_k\rangle \\ &+(\frac{1}{\sqrt{5}}|0\rangle + \frac{2}{\sqrt{5}}|1\rangle)|0\rangle|0\rangle\sum_{k=r+1}^{T}\sigma_k\left|\sigma_k^2\right\rangle|u_k\rangle|v_k\rangle. \end{aligned} \tag{7}$$

Step 4: Rotation. As shown in Fig. 3, the unitary $R_y(\alpha)$ can realize

$$R_y(\alpha)|0\rangle|y_k\rangle = (sin(y_k\alpha)|0\rangle + cos(y_k\alpha)|1\rangle)|y_k\rangle. \tag{8}$$

Perform unitary operation $R = R_y(\alpha)$ on $|\psi_3\rangle$ which leads to

$$\begin{aligned} |\psi_4\rangle =& \frac{1}{\sqrt{5}}[|0\rangle\sum_{k=1}^{r}(siny_k\alpha|1\rangle + cosy_k\alpha|0\rangle)|y_k\rangle \\ &+|1\rangle\sum_{k=1}^{r}(2sin\frac{y'_k\alpha}{2}|1\rangle + 2cos\frac{y'_k\alpha}{2}|0\rangle)\left|y'_k\right\rangle]|H_k\rangle \\ &+(\frac{1}{\sqrt{5}}|0\rangle + \frac{2}{\sqrt{5}}|1\rangle)|0\rangle|0\rangle\sum_{k=r+1}^{T}|H_k\rangle, \end{aligned} \tag{9}$$

where $|H_k\rangle = \sigma_k\left|\sigma_k^2\right\rangle|u_k\rangle|v_k\rangle$.

Step 5: Projection measurement. We perform $U^{\dagger} = (I\otimes U_{PE}^{\dagger}(Ker))(I\otimes PO_{\sigma,\tau}^{\dagger}\otimes I)$ to eliminate extra registers and project the quantum state $|\psi_4\rangle$ onto $|+\rangle|1\rangle$, collapse to

$$\left|\Psi_{\hat{A}}\right\rangle = \sum_{k=1}^{r}\sigma_k(siny_k\alpha + 2sin\frac{y'_k\alpha}{2})|u_k\rangle|v_k\rangle. \tag{10}$$

Step 6: The coefficient approximation. We get $\left|\Psi_{A_0}\right\rangle$ by approximating coefficient function $siny_k\alpha + 2sin\frac{y'_k\alpha}{2}$, so

$$\left|\Psi_{A_0}\right\rangle = \sum_{k=1}^{r}\sigma_k|u_k\rangle|v_k\rangle. \tag{11}$$

Finally, take PCA into consideration, we perform the same phase estimation as step 2 to get

$$\left|\Psi_{A_1}\right\rangle = \sum_{k=1}^{r}\sigma_k|u_k\rangle|v_k\rangle|\lambda_k\rangle, \tag{12}$$

where $\sigma_k = \sqrt{\lambda_k}$.

We introduce our algorithm in detail in this paragraph. Firstly, in each step of our algorithm, we normalize the data and omit the normalization factor. The selection of threshold and parameters is crucial to determine the quality of approximation in step 6. The amplitude of auxiliary qubit $a|0\rangle + b|0\rangle$ is referred as coefficient parameters and the system parameter is α. The detailed description about parameters selection is shown in Appendix A.

Next, in the previous algorithm, we can take low-rank matrix A as a input. For kernel PCA, we define a kernel map ϕ, the data x_i is mapped into $\phi(x_i)$ and $Ker_{ij} = \langle\phi(x_i)|\phi(x_j)\rangle$. We employ $|\hat{\psi_A}\rangle$ as a part of the initial state where $|\hat{\psi_A}\rangle = \sum_{k=i}^{M}|i\rangle|\phi(x_i)\rangle$. In this case we perform similar steps to obtain the target solution. For instance, we use copies of quantum states [24] to generate $|\phi(x_i)\rangle = |x_i\rangle|x_i\rangle$ and $Ker_{ij} = \langle\phi(x_i)|\phi(x_j)\rangle = \langle x_i|x_j\rangle^2$.

This algorithm can be used for sparse matrix when we adopt the unitary operation e^{iAt} instead of e^{iKert} to generate $|\sigma_k\rangle$ and

perform phase estimation where A is sparse. We design a simple quantum circuit using addition and multiplication as shown in Fig. 3 to obtain the state $|\sigma_k^2\rangle$. The process is as follows:

$$|0\rangle|0\rangle|\sigma_k\rangle \overset{ADD}{\rightleftharpoons} |0\rangle|\sigma_k\rangle|\sigma_k\rangle \overset{MULTI}{\rightleftharpoons} |\sigma_k^2\rangle|\sigma_k\rangle|\sigma_k\rangle.$$

3. The algorithm analysis

In this section, we briefly analyze the complexity of our algorithm and discuss fidelity problem for approximation process. Firstly, we show the complexity of performing each step and obtaining the classical solution about eigenvectors and eigenvalues. Then, we describe the fidelity problem and discuss the influence of parameters selection on fidelity. Finally, we discuss the application of our algorithm.

3.1. The complexity

We analyze the time complexity of our algorithm. As can be seen from Fig. 4, Improved QPCA algorithm consists of QSVT algorithm, MQSVT algorithm and an auxiliary control bit. In this paper MQSVT algorithm is consistent with QSVT algorithm in terms of time complexity. The preparation of initial state $|\Psi_A\rangle$ costs time $O(logMN)$ via qRAM. We define the condition number as κ and suppose $t = (\kappa/\epsilon)$ with accuracy ϵ where κ represents the ratio between $A's$ largest and smallest eigenvalues. In phase estimation, when A is a low-rank or s-sparse matrix, the time complexity of unitary operations e^{-iKert} and $e^{-iAt}(M = N)$ is $O(\epsilon^{-1}t^2 logMN)$ [15,28] and $O(log(M)s^2t)$ [16,29], respectively. Here we can not consider computational complexity of parameters selection. Finally, we need to run our algorithm multiple times to construct the eigenvectors corresponding eigenvalues by sampling. In a nutshell, all eigenvalues are of size $O(1/r)$ and the corresponding time complexity is $t = O(r)$.

Thus, the total time complexity of the algorithm is $O(r\epsilon^{-1}t^2 logMN)$ or $O(rlog(M)s^2t)$. If $\epsilon^{-1}t^2$ and s^2t are equivalent to a constant factor and suppose $N \leq M$, the total time complexity of the algorithm is $O(rploy(logM))$.

3.2. The success probability and fidelity

We now discuss the success probability of obtaining the target solution and the fidelity problem which is the degree of similarity between actual output and ideal output. Eqs. (9) and (10) show that we obtain the Eqs. (10) with a certain probability. Eqs. (11) is approximation of the Eqs. (10). Error of phase estimation depends on the preset accuracy, which is not discussed here. Therefore, we now analyze the relation of probability and fidelity and select parameters to preserve our algorithm efficiency contains high fidelity and high probability.

The probability of projecting into $|+\rangle|1\rangle$ in step 5 can be calculated via Eqs. (9) and (10):

$$P(|+\rangle|1\rangle) = \frac{\sum_{k=1}^{r}\sigma_k^2(siny_k\alpha + 2sin\frac{y_k'\alpha}{2})^2}{10\sum_{k=1}^{T}\sigma_k^2}. \tag{13}$$

We use inner product of Eqs. (9) and (10) to show the fidelity of the actual output and the ideal output:

$$\begin{aligned} &F(|\psi_{A_0}\rangle, |\psi_5\rangle) = \langle\psi_{A_0}|\psi_5\rangle/N_1 \\ &= \frac{\sum_{k=1}^{r}\sigma_k^2(siny_k\alpha + 2sin\frac{y_k'\alpha}{2})}{\sqrt{\sum_{k=1}^{r}\sigma_k^2\sum_{k=1}^{r}\sigma_k^2(siny_k\alpha + 2sin\frac{y_k'\alpha}{2})^2}}, \end{aligned} \tag{14}$$

where N_1 is product of normalization coefficient of $|\psi_{A_0}\rangle, |\psi_5\rangle$.

The goal of our algorithm is to achieve high fidelity and output the target solution with high probability. The QSVT algorithm uses function $G(\alpha)$ [20] to ensure high fidelity and high probability where $G(\alpha) = \sqrt{P} * F$. We introduce a function f instead of $G(\alpha)$:

$$f(\sigma_k) = siny_k\alpha + 2sin\frac{y_k'\alpha}{2}. \tag{15}$$

The function $f(\sigma_k)$ is a factor of Eqs. (13) and Eqs. (14). And if Eqs. (17) can satisfy the condition:

$$\forall k,\ f(\sigma_k) \approx B, \tag{16}$$

where F is a constant, the function of fidelity and probability is transformed to the following equation:

$$\begin{cases} P(|+\rangle|1\rangle) \approx \frac{B^2\sum_{k=1}^{r}\sigma_k^2}{10\sum_{k=1}^{T}\sigma_k^2}; \\ F(|\psi_{A_0}\rangle, |\psi_5\rangle) \approx 1. \end{cases} \tag{17}$$

We now consider a more general case involving coefficient parameters. Suppose

$$f(\sigma_k, \alpha, C) = siny_k\alpha + Csin\frac{y_k'\alpha}{2}, \tag{18}$$

where $C = b/a$ is used to replace the parameters a, b. We intend to select the parameters that minimize $\|f(\sigma_i)_{max} - f(\sigma_j)_{min}\|$. The problem of $\underset{\alpha}{argmax}\, G(\alpha)$ is transformed to solve this equation:

$$RA(\alpha, C) = \sum_{i,j=1}^{r}\left|f(\sigma_i, \alpha, C) - f(\sigma_j, \alpha, C)\right|. \tag{19}$$

We find optimal parameters α, C such that minimize $RA(\alpha, C)$, i.e. to solve the problem $\underset{\alpha,C}{argmin}\, RA(\alpha, C)$. Here we give a set of parameters $\alpha = \frac{\pi}{2}, C = 2$ (see the Appendix A for parameters discussion). We can determine the maximum and minimum of the closed interval by taking the derivative and then the range of function f is

$$2 < f(\sigma_k, \frac{\pi}{2}, 2) < 2.6. \tag{20}$$

Here the domain of this function is $[0, 1]$. The actual domain of this function is $[\frac{\tau}{\sigma_1}, \frac{\tau}{\sigma_r}]$ and Eqs. (20) is not a best result. It shows that we can further reduce the range by selecting parameters α, C and threshold τ.

The goal of modifying parameters and threshold is to obtain the ideal result with high probability and high fidelity. The rank of matrices is also a important factor to our final result. We now turn to this problem in the following paragraph and use MATLAB 2016 to simulate our algorithm with parameters $\alpha = \frac{\pi}{2}, C = 2$ and different rank.

As shown in Fig. 5, the algorithm achieves good results with fidelity above 99.92% and probability above 0.3. Fig. 5 (a), (b) and (c) show that the rank of matrices has little effect on the fidelity above 99.955%. We draw a conclusion that probability and threshold mainly affect the fidelity from Fig. 5 (e) and (f), especially when the threshold is small and the probability is in the interval [0.45, 0.6]. Fig. 5 (d) and (f) show the range of probability and the threshold impacts on probability. The probability and fidelity are sensitive to threshold by changing the $[\frac{\tau}{\sigma_1}, \frac{\tau}{\sigma_r}]$. Furthermore, assume $\sigma_{r+1} \leq \tau_1 < \tau_2 < \sigma_r$, we have $\frac{\tau_1}{\sigma_r} < \frac{\tau_2}{\sigma_r}$ and change the probability and fidelity. In summary, the algorithm can satisfy our demand with probability 0.3 and fidelity 99.92% in the worst case.

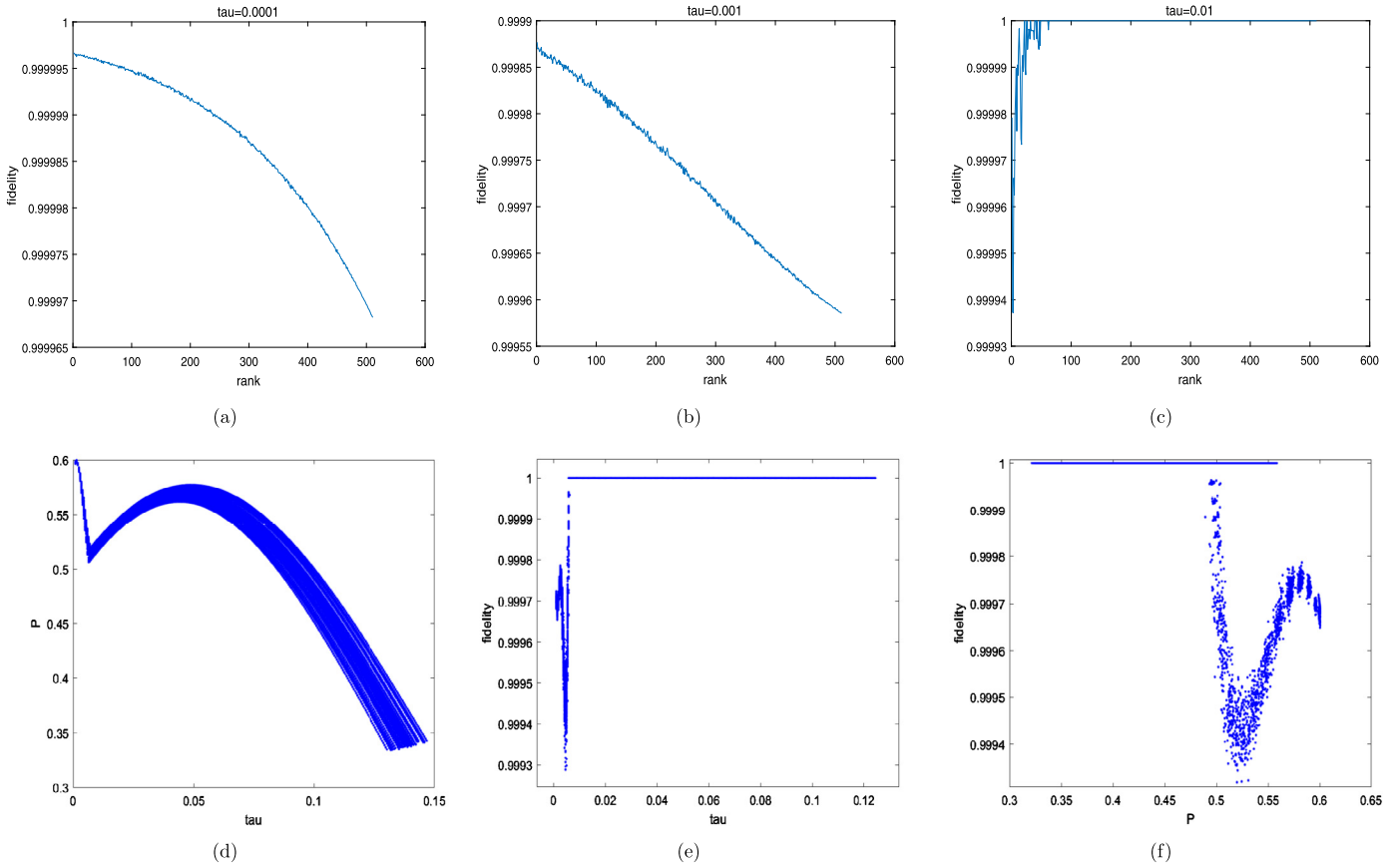

Fig. 5. (a) (b) (c) The function relation between fidelity and rank based on different thresholds. (d) The function relation between average probability and threshold under the condition of multiple matrices. (e) The function relation between average fidelity and threshold with multiple matrices. (f) The function relation between average fidelity and average probability with multiple matrices. All of the matrixes use 517×512 dimensional and these matrixes are normalized.

3.3. Discussion of our algorithm

qPCA algorithm can be performed in time O(r ploy(logM)) if a matrix A satisfy this condition: A is r-rank matrix and our goal is the eigenvectors corresponding to top r eigenvalues. Compared with qPCA, our algorithm first performs an optimal matrix approximation in time O(ploy(logM)) which transforms $A = \sum_{k=1}^{T} \sigma_k u_k v_k$ to $A' = \sum_{k=1}^{r} \sigma_k u_k v_k$. Here we take step1-step6 of our algorithm as a quantum optimal matrix approximation process. It shows the advantage that we can handle the high-rank matrix in time O(r ploy(logM)). And in this case, the final result of qPCA algorithm obtains the solution in time $O(Tploy(logM))$. In summary, our algorithm may implement an exponential acceleration compared to qPCA when $r \ll T < M$. In addition, we can use other method to increase the probability of some eigenvalues by matrix inversion and the QSVT(MQSVT) algorithm itself.

The goal of obtaining eigenvalues and eigenvectors can also be achieved via quantum annealing [30]. This method takes $AA^{\dagger}$ as the target Hamiltonian to output the ground state of $AA^{\dagger}$ with the large time scale T_A and we execute the method multiple times to yield target eigenvectors. The time scale T_A mainly depends on the inner product the ground state of the initial Hamiltonian and the ground state of the target Hamiltonian. Under specific conditions, this method can be more suitable for solving the target eigenvalues. In other cases, our algorithm can achieve better results. Both approaches complement each other.

The unitary transformation e^{iAt} can be constructed as a combination of unitary operators. Next we discuss that matrix A is the sum of sparse matrix and low-rank matrix. Define the matrix of A as

$$A = \begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix}, B = \begin{pmatrix} A_1 & 0 \\ 0 & A_4 \end{pmatrix} \tag{21}$$

where B is s-sparse matrix and $(A - B)$ is a low-rank matrix. The unitary transformation e^{iAt} is constructed by e^{iBt} and $e^{i(A-B)t}$. It means that we can use some matrices with special structure except for sparse or low-rank matrices.

We discuss several applications of our algorithm. Naturally, various machine learning applications involve PCA algorithm. Next, our matrix approximation process can achieve an exponential acceleration over classical or hybrid counterparts if $logM \ll r < T$. It allows us to perform our approximation method to construct the quantum state $|\Psi_{A_0}\rangle$ as a input of other quantum data analyze methods. Meanwhile we can employ this quantum state to achieve the data noise reduction which is used to side channel attack and a few actual applications. Finally, our algorithm can be useful for other matrix approximation techniques. For instance, Nyström approximation is helpful to achieve a sparse solution [31–34] of LSSVM. Apparently, it needs optimal low-rank approximation.

4. Conclusion

In this paper, an improved quantum principal component analysis algorithm (Improved qPCA) is proposed based on QSVT algorithm. We have shown that Improved qPCA can used to solve a general matrix, especially high-rank matrix. Note that we can solve kernel PCA with $A = Ker$ where Ker is a kernel matrix. The

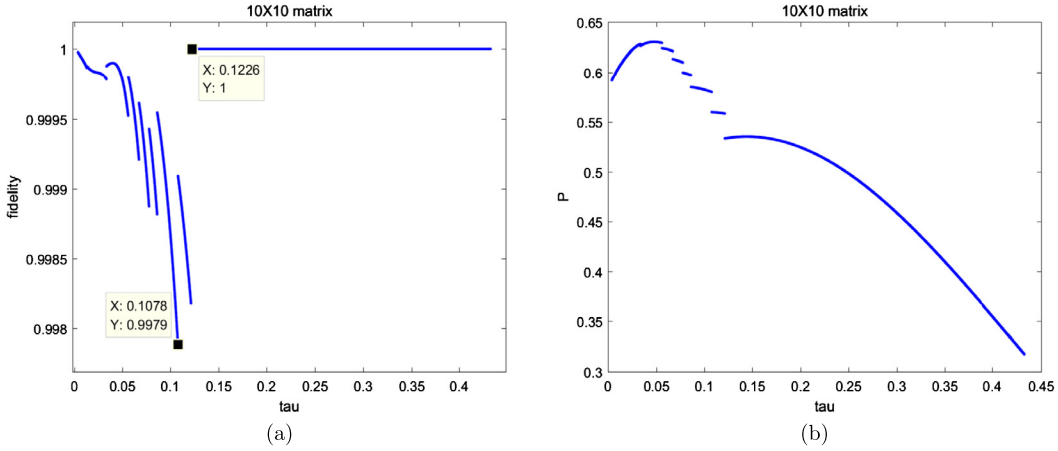

Fig. 6. (a) The function relation between fidelity and threshold in this 10×10 matrix. (b) The function relation between probability and threshold in this 10×10 matrix.

total complexity of Improved qPCA is $O(rpoly(logM))$ and the algorithmic complexity of two key subroutines is $O(poly(logM))$. The quantum-inspired classical algorithm [35,36] can also solve PCA problem in polynomial time with low-rank matrices. The core of our algorithm is an optimal matrix approximation in quantum form. Therefore, the QSVT algorithm can be used to estimate $\sum_{k=1}^{r}(\sigma_k - \tau)u_k v_k$ and the MQSVT algorithm is proposed for estimating $\sum_{k=1}^{r}(\sigma_k + \tau)u_k v_k$. Finally we achieve the matrix approximation with high fidelity and high probability by using the two state to eliminate the threshold τ.

Our analysis and discussion show the following opinions. Firstly, compared with qPCA, we give a speed up if the rank T of the matrix is greater than the target rank r. Second, our algorithm, especially the quantum optimal matrix approximation, can used to a subroutine in other quantum machine learning algorithms for solve the matrix approximation problem. And the complexity of this procedure is $O(poly(logM))$. Finally, we extend the construction of unitary transformation to a more general symmetric matrix.

There are also some viewpoints worth discussing. We can not accurately choose the threshold to retain the eigenvalues. The another shortcoming is that we have a fidelity problem to effect the algorithm accuracy. Here we think it is equal to perform a L_2 regularization operation [20] in some case.

Appendix A. Parameters selection and example

In this section, we show that the general matrix can be replaced by the normalization of matrix and give an example to illustrate the effect of parameters. Here we only discuss the case with system parameter $\alpha = \frac{\pi}{2}$. From Eqs. (13) and Eqs. (14) we can conclude

$$A, \tau \overset{F,P}{\Longleftrightarrow} A' = IA, \tau' = I\tau. \tag{A.1}$$

Then a general matrix A can be represented by $A/\|A\|_F$ where $\|\bullet\|_F$ is the Frobenius norm. Next we give an example to explain that the difference of actual eigenvalues and ideal eigenvalues. And the fidelity of our algorithm can be further enhanced by the selection of coefficient parameters and threshold. The 10×10 matrix with the rank 10 shows this result. The corresponding eigenvalues are

$$\sigma = (0.4328, 0.1215, 0.1079, 0.0863, 0.0779, 0.0674, 0.0562, 0.0336, 0.0133, 0.0032). \tag{A.2}$$

Then, Fig. 6 shows that we can get the fidelity above 99.79% and probability above 0.3. For the thresholds $\tau = 0.1078$ and $\tau = 0.0864$, the domain of Eqs. (13) is about [0.249,0.999] and [0.2,0.8] and the big interval of former leads to the lowest point of Fig. 6(a). This result can be used to explain that the different thresholds ($\tau : 0.1079 > 0.1078 > 0.864 > 0.863$) have different fidelity for obtaining a r-rank approximation matrix. We choose a threshold $\tau = 0.05$ with fidelity 99.976% and probability 0.633 and the eigenvalues of low-rank approximation are

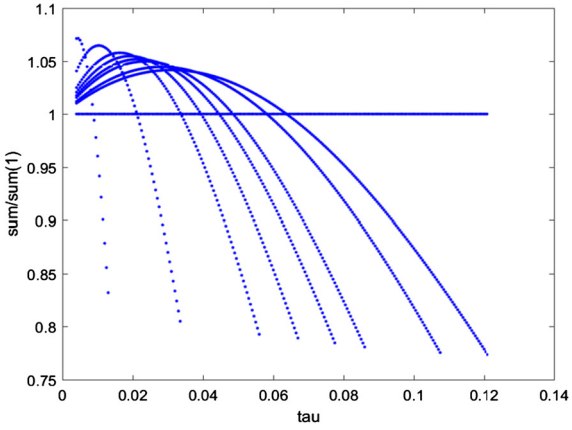

Fig. 7. The function relation σ/σ_1 between the threshold and the deformation degree of eigenvalues where σ_1 represents the first eigenvalue.

$$\sigma' = (0.4328, 0.1227, 0.1092, 0.0876, 0.0793, 0.0688, 0.0576, 0.0350, 0.0142). \tag{A.3}$$

A few eigenvalues can change dramatically in Fig. 7. And there exist an eigenvalue which is 77.4% of the original value when the threshold $\tau = 0.1078$. The fidelity can be further improved by changing the coefficient parameter C. For instance, we set the coefficient parameter $C = 1.3$, the eigenvalues corresponding to vector σ' are

$$\sigma'' = (0.4328, 0.1215, 0.1079, 0.0863, 0.0779, 0.0674, 0.0561, 0.0333, 0.0122). \tag{A.4}$$

Therefore, we can select system parameter α and construct a auxiliary qubit with coefficient parameter C or a, b before performing our algorithm. All the parameters we use satisfy high fidelity and certain probability. In addition, the main goal of threshold selection is to find the number of appropriate eigenvalues based on CCR condition. We do not actually spend extra cost to increase fidelity by changing the threshold slightly.

References

[1] Ming Li, Baozong Yuan, 2d-lda: a statistical linear discriminant analysis for image matrix, Pattern Recognit. Lett. 26 (5) (2005) 527–532.

[2] Yoshua Bengio, Jean-françcois Paiement, Pascal Vincent, Olivier Delalleau, Nicolas L. Roux, Marie Ouimet, Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering, in: Advances in Neural Information Processing Systems, 2004, pp. 177–184.

[3] Effrosyni Kokiopoulou, Yousef Saad, Orthogonal neighborhood preserving projections: a projection-based dimensionality reduction technique, IEEE Trans. Pattern Anal. Mach. Intell. 29 (12) (2007) 2143–2156.

[4] Svante Wold, Kim Esbensen, Paul Geladi, Principal component analysis, Chemom. Intell. Lab. Syst. 2 (1–3) (1987) 37–52.

[5] Michael E. Tipping, Christopher M. Bishop, Probabilistic principal component analysis, J. R. Stat. Soc., Ser. B, Stat. Methodol. 61 (3) (1999) 611–622.

[6] Jonathon Shlens, A tutorial on principal component analysis, arXiv preprint, arXiv:1404.1100, 2014.

[7] Patrick Rebentrost, Maria Schuld, Leonard Wossnig, Francesco Petruccione, Seth Lloyd, Quantum gradient descent and Newton's method for constrained polynomial optimization, arXiv preprint, arXiv:1612.01789, 2016.

[8] Guoming Wang, Quantum algorithm for linear regression, Phys. Rev. A 96 (1) (2017) 012335.

[9] Patrick Rebentrost, Masoud Mohseni, Seth Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. 113 (13) (2014) 130503.

[10] He-Liang Huang, Qi Zhao, Xiongfeng Ma, Chang Liu, Zu-En Su, Xi-Lin Wang, Li Li, Nai-Le Liu, Barry C. Sanders, Chao-Yang Lu, et al., Experimental blind quantum computing for a classical client, Phys. Rev. Lett. 119 (5) (2017) 050503.

[11] He-Liang Huang, Ashutosh K. Goswami, Wan-Su Bao, Prasanta K. Panigrahi, Demonstration of essentiality of entanglement in a deutsch-like quantum algorithm, Sci. China, Phys. Mech. Astron. 61 (6) (2018) 060311.

[12] Hoi-Kwan Lau, Raphael Pooser, George Siopsis, Christian Weedbrook, Quantum machine learning over infinite dimensions, Phys. Rev. Lett. 118 (8) (2017) 080501.

[13] He-Liang Huang, Xi-Lin Wang, Peter P. Rohde, Yi-Han Luo, You-Wei Zhao, Chang Liu, Li Li, Nai-Le Liu, Chao-Yang Lu, Jian-Wei Pan, Demonstration of topological data analysis on a quantum processor, Opt. 5 (2) (2018) 193–198.

[14] Adrian Steffens, Patrick Rebentrost Iman Marvian, Jens Eisert, Seth Lloyd, An efficient quantum algorithm for spectral estimation, New J. Phys. 19 (3) (2017) 033005.

[15] Seth Lloyd, Masoud Mohseni, Patrick Rebentrost, Quantum principal component analysis, Nat. Phys. 10 (9) (2014) 631.

[16] W. Aram Harrow, Avinatan Hassidim, Seth Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. 103 (15) (2009) 150502.

[17] Bernhard Schölkopf, Alexander Smola, Klaus-Robert Müller, Kernel principal component analysis, in: International Conference on Artificial Neural Networks, Springer, 1997, pp. 583–588.

[18] Alessandro Maria Rizzi, Support vector regression model for bigdata systems, arXiv preprint, arXiv:1612.01458, 2016.

[19] Senjian An, Wanquan Liu, Svetha Venkatesh, Face recognition using kernel ridge regression, in: 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–7.

[20] Dan-Bo Zhang, Zheng-Yuan Xue, Shi-Liang Zhu, Z.D. Wang, Realizing quantum linear regression with auxiliary qumodes, Phys. Rev. A 99 (1) (2019) 012331.

[21] Ameet Talwalkar, Matrix Approximation for Large-Scale Learning, PhD thesis, Citeseer, 2010.

[22] Petros Drineas, Michael W. Mahoney, S. Muthukrishnan, Subspace sampling and relative-error matrix approximation: column-based methods, in: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, Springer, 2006, pp. 316–326.

[23] Shusen Wang, Luo Luo, Zhihua Zhang, The modified Nyström method: theories, algorithms, and extension, Eprint Arxiv 33 (6) (2014) 65–89.

[24] Bojia Duan, Jiabin Yuan, Ying Liu, Dan Li, Quantum algorithm for support matrix machines, Phys. Rev. A 96 (3) (2017) 032301.

[25] Bojia Duan, Jiabin Yuan, Ying Liu, Dan Li, Efficient quantum circuit for singular-value thresholding, Phys. Rev. A 98 (1) (2018) 012308.

[26] Alex Parent, Martin Roetteler, Michele Mosca, Improved reversible and quantum circuits for Karatsuba-based integer multiplication, arXiv preprint, arXiv:1706.03419, 2017.

[27] Bojia Duan, Jiabin Yuan, Juan Xu, Dan Li, Quantum algorithm and quantum circuit for a-optimal projection: dimensionality reduction, Phys. Rev. A 99 (3) (2019) 032311.

[28] Patrick Rebentrost, Adrian Steffens, Seth Lloyd, Quantum singular value decomposition of non-sparse low-rank matrices, arXiv preprint, arXiv:1607.05404, 2016.

[29] Dominic W. Berry, Graeme Ahokas, Richard Cleve, Barry C. Sanders, Efficient quantum algorithms for simulating sparse Hamiltonians, Commun. Math. Phys. 270 (2) (2007) 359–371.

[30] Yoichiro Hashizume, Takashi Koizumi, Kento Akitaya, Takashi Nakajima, Soichiro Okamura, Masuo Suzuki, Singular-value decomposition using quantum annealing, Phys. Rev. E 92 (2) (2015) 023302.

[31] Shuisheng Zhou, Sparse lssvm in primal using Cholesky factorization for large-scale problems, IEEE Trans. Neural Netw. Learn. Syst. 27 (4) (2015) 783–795.

[32] Raghvendra Mall, Johan AK Suykens, Very sparse lssvm reductions for large-scale data, IEEE Trans. Neural Netw. Learn. Syst. 26 (5) (2015) 1086–1097.

[33] Chen Li, Zhou Shuisheng, Sparse algorithm for robust lssvm in primal space, Neurocomputing (2017).

[34] Mu Li, Wei Bi, James T. Kwok, Bao-Liang Lu, Large-scale Nyström kernel matrix approximation using randomized svd, IEEE Trans. Neural Netw. Learn. Syst. 26 (1) (2014) 152–164.

[35] András Gilyén, Seth Lloyd, Ewin Tang, Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension, arXiv preprint, arXiv:1811.04909, 2018.

[36] Ewin Tang, Quantum-inspired classical algorithms for principal component analysis and supervised clustering, arXiv preprint, arXiv:1811.00414, 2018.