QUANTUM ALGORITHM FOR FINDING THE NEGATIVE CURVATURE DIRECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

We present an efficient quantum algorithm aiming to find the negative curvature direction for escaping the saddle point, which is a critical subroutine for many second-order non-convex optimization algorithms. We prove that our algorithm could produce the target state corresponding to the negative curvature direction with query complexity $\tilde{\mathcal{O}}(\mathrm{polylog}(d)\epsilon^{-1})$, where d is the dimension of the optimization function. The quantum negative curvature finding algorithm is exponentially faster than any known classical method which takes time at least $\mathcal{O}(d\epsilon^{-1/2})$. Moreover, we propose an efficient algorithm to achieve the classical read-out of the target state. Our classical read-out algorithm runs exponentially faster on the degree of d than existing counterparts.

1 INTRODUCTION

Algorithms for finding the minima of functions have attracted significant attention due in part to their prevalent applications in machine learning, deep learning and robust statistics; in particular, those with good complexity guarantees that can converge to the local minima. Numerous algorithms have been proposed in recent years for finding points that satisfying

$$\|\nabla f(\boldsymbol{x})\| \leq \epsilon_g, \text{and } \lambda_{\min}\left(\nabla^2 f(\boldsymbol{x})\right) \geq -\epsilon_H,$$

where $\epsilon_g, \epsilon_H \in (0,1)$. Recent proposals (Nesterov & Polyak, 2006; Conn et al., 2000; Agarwal et al., 2017) based on second-order Newton-type and first-order methodology have been analyzed from such a perspective. However, those methods normally deal with the situations that the iterations may be trapped in the saddle points, since in many cases, such as deep neural networks (Dauphin et al., 2014; Choromanska et al., 2015), existence of many saddle points is the main bottleneck.

In general non-convex optimization, many algorithms have been proposed to escape the saddle points. These algorithms can be divided into the following two categories: the first-order gradient-based algorithms and the second-order Hessian-based algorithms. Generally, second-order algorithms have better iteration complexity than first-order algorithms (cf. Jin et al. (2017)). However, each iteration in the second-order method involves the computation of the **negative curvature direction**, namely, the eigenvector of a Hessian matrix $\boldsymbol{H} = \nabla^2 f(\boldsymbol{x})$ with negative eigenvalue. This computation could take time $\mathcal{O}(d^2)$ when the Hessian matrix is given, or $\mathcal{O}(d/\sqrt{\epsilon})$ when Lanczos method is used with Gradient information to approximate the Hessian-vector product.

Quantum algorithms have shown great potential to become faster alternatives than classical algorithms for many kinds of problems in the field of linear algebra, including principal component analysis (Lloyd et al., 2014), support-vector machine (Rebentrost et al., 2014), singular value decomposition (Rebentrost et al., 2018). These works encourage us to develop an efficient quantum algorithm for finding the negative curvature. To begin with, we formally define the negative curvature finding problem as follows.

Negative Curvature Finding (NCF) problem: *Given a function $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ which has L-Lipschitz continuous gradient and the corresponding Hessian , we aim to build a quantum algorithm that could efficiently provide the unit vector $\boldsymbol{u}$ with the condition:*

$$\boldsymbol{u}^T \boldsymbol{H} \boldsymbol{u} \leq -\alpha + \epsilon, \tag{1}$$

where $0 < \alpha < L$ and $0 < \epsilon < \alpha$; or make the non-vector statement that with high probability there is no unit vector $\boldsymbol{u}$ satisfying the following condition:

$$\boldsymbol{u}^T \boldsymbol{H} \boldsymbol{u} < -\alpha. \tag{2}$$

1.1 RELATED WORK

Optimization methods for non-convex problems can be roughly divided into first-order methods and second-order methods, depending on the order of the derivative to the objective function they used. Generally, to find the local minima, the second-order methods (Carmon et al., 2018; Agarwal et al., 2017) are exploited to find the effective direction to escape the saddle point. Specifically, finding the Negative Curvature is considered as a critical subroutine to analyze the characteristic of the saddle point.

First-order algorithms: For the non-convex problem, the first-order method (Gradient-based method) can find the stationary point, which could be a global minima, local minima or saddle point. However, standard analysis by gradient descent cannot distinguish between saddle points and local minima, leaving open the possibility that gradient descent may get stuck at saddle points. Recently Ge et al. (2015); Jin et al. (2017; 2019) showed that by adding noise at each step, gradient descent can escape all saddle points in a polynomial number of iterations. Lee et al. (2016) proved that under similar conditions, gradient descent with random initialization avoids saddle points even without adding noise. However, each step of Gradient-based methods requires $\mathcal{O}(d)$ operations and their iteration complexity is higher than second-order algorithms (Jin et al., 2017).

Second-order algorithms: Traditionally, second-order Newton-based methods can converge to local minima, which use the Hessian information to distinguish between first-order and second-order stationary points. There are two kinds of methods that make use of Hessian information. 1) Hessian-based: trust-region (Conn et al., 2000) and cubic regularization (Nesterov & Polyak, 2006) are two methods, in which the sub-problem is to find the decrease direction based on the given Hessian. The calculation of each iteration involves performing Hessian-vector production, which takes time at least $\mathcal{O}(d^2)$. 2) Hessian-free: The Hessian-free methods use Lanczos method to calculate the negative curvature direction and use gradient to approximate the Hessian-vector product (Agarwal et al., 2017; Carmon et al., 2018; Carmon & Duchi, 2016). The Hessian-free method involves $\mathcal{O}(d\epsilon^{-1/2})$ complexity per iteration. The advantage of the second-order algorithm is the superior iteration complexity than the first-order algorithm. However, using Hessian information usually increases computation time per iteration.

1.2 OUR CONTRIBUTION

The contribution of this work can be briefly divided into two parts: 1) an efficient quantum algorithm to generate the required quantum state, which corresponds to the negative curvature direction, and 2) an efficient quantum algorithm to obtain the description of the target state $|\boldsymbol{u}_t\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle$, where $\{\boldsymbol{s}_i\}_{i=1}^{r}$ is an independent vector set selected from columns of Hessian $\boldsymbol{H}$ with rank r.

Negative Curvature Finding: We develop an efficient quantum algorithm to produce the target state $|\boldsymbol{u}_t\rangle$ (for case (1)) or make the non-vector statement (for case (2)). We provide Proposition 1 as the main result of this part, which guarantees the time complexity of our NCF algorithm:

Proposition 1. *There exists a quantum algorithm which could solve the Negative Curvature Finding problem in time $\tilde{\mathcal{O}}(\mathrm{polylog}(d)\mathrm{poly}(r)\epsilon^{-1})$, by providing the target state $|\boldsymbol{u}_t\rangle$ (for case (1)), or making the non-vector statement (for case (2)).*

Classical Read-out: The classical read-out problem is one bottleneck for many quantum machine learning algorithms whose results are quantum states. Generally, the read-out of a d-dimensional quantum state takes time at least $\mathcal{O}(d)$ (Aaronson, 2015), and could offset the claimed quantum speed-up. In order to solve this dilemma, we develop an efficient quantum algorithm for the classical read-out of the target state. We notice that the target state $|\boldsymbol{u}_t\rangle$ can be written as the linear combination form $|\boldsymbol{u}_t\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle$, where $\{\boldsymbol{s}_i\}_{i=1}^{r}$ is a linearly independent basis sampled from column vectors $\{\boldsymbol{h}_j\}_{j=1}^{d}$. The algorithm suits the case when the result quantum state lies in the span of several given states, and may give rise to independent interest.

One advantage of generating the form $|\boldsymbol{u}_t\rangle = \sum_{i=1}^r x_i|\boldsymbol{s}_i\rangle$ is that the updating operation for the l-th iteration $\boldsymbol{z}^{(l+1)} = \boldsymbol{z}^{(l)} + \eta\boldsymbol{u}_t$ in general non-convex algorithms could be implemented in quantum form $|\boldsymbol{z}^{(l+1)}\rangle \propto |\boldsymbol{z}^{(l)}\rangle + \eta\sum_{i=1}^r x_i|\boldsymbol{s}_i\rangle$ by Linear-Sum-of-States method (Shao, 2018) efficiently, which may inspire efficient quantum counterparts for these non-convex algorithms. Note that we could not perform the operation $|\boldsymbol{z}^{(l+1)}\rangle \propto |\boldsymbol{z}^{(l)}\rangle + \eta|\boldsymbol{u}_t\rangle$ directly since the target state $|\boldsymbol{u}_t\rangle$ is generated by post-selection instead of standard unitary operations. Our state read-out algorithm contains two subroutines named as the Complete Basis Selection and the State Overlap Estimation. The main results about the Complete Basis Selection and the Classical Read-out are briefly summarized as following theorems:

Theorem 1. *There exists a quantum algorithm which takes time $\tilde{\mathcal{O}}(\text{poly}(r)\epsilon^{-2}r^c)$ to find an index set $\{g(i)\}_{i=1}^r$, where r is the rank of $\boldsymbol{H}$, $c = 2\log\frac{4r\|\boldsymbol{H}\|_F}{\epsilon}$ and $\{g(i)\}_{i=1}^r$ forms a complete basis $\{|\boldsymbol{h}_{g(i)}\rangle\}_{i=1}^r$ with probability at least 3/4.*

Theorem 2. *The classical description of the target state $|\boldsymbol{u}_t\rangle = \sum_{i=1}^r x_i|\boldsymbol{s}_i\rangle$ could be presented in time $\tilde{\mathcal{O}}(\text{polylog}(d)\text{poly}(r)\epsilon^{-5})$ with error bounds in $\frac{\epsilon}{2}$, when the basis set $\{\boldsymbol{s}_j\}_{j=1}^r$ is given.*

The rest of this paper is organized as follows. Some preliminaries about quantum information are introduced in Section 2. In Section 3, we develop an quantum algorithm to solve the NCF problem. In Section 4, we develop an quantum algorithm which aims to read out the target state. We summarize our results and contributions in Section 5.

2 PRELIMINARY

In this section we present some preliminary concepts. Some basic quantum knowledge will be introduced in Section 2.1. Some quantum technics will be introduced in Section 2.2.

2.1 NOTATIONS AND DEFINITIONS

In this section, we introduce some useful notations and definitions about quantum computing. The dirac notation is a standard notation in quantum mechanics to describe the quantum states. The form $|\boldsymbol{x}\rangle$ is the state which corresponds to the vector $\boldsymbol{x}$, and the form $\langle\boldsymbol{y}|$ is the state which corresponds to the vector $\boldsymbol{y}^T$. The notation $\langle\boldsymbol{y}|\boldsymbol{x}\rangle$ denotes the value $\boldsymbol{y}^T\boldsymbol{x}/(\|\boldsymbol{y}\|\|\boldsymbol{x}\|)$. The notation $|\boldsymbol{y}\rangle\langle\boldsymbol{x}|$ denotes the matrix $\boldsymbol{y}\boldsymbol{x}^T/(\|\boldsymbol{y}\|\|\boldsymbol{x}\|)$. Quantum state is unitary, which means $\||\boldsymbol{x}\rangle\|^2 = \langle\boldsymbol{x}|\boldsymbol{x}\rangle = 1$. Thus for vector $\boldsymbol{x} \in \mathbb{R}^d$, the state $|\boldsymbol{x}\rangle$ is defined as $\sum_{j=1}^d x_j/\|\boldsymbol{x}\| |j\rangle$, where x_j is the j-th component of vector $\boldsymbol{x}$ and $\{|j\rangle\}_{j=1}^d$ is the state basis which acts like $\{\boldsymbol{e}_j\}_{j=1}^d$ in classical case. One could obtain information from the quantum state by performing measurement. For example, the measurement of $|\boldsymbol{x}\rangle$ on the basis $\{|j\rangle\}_{j=1}^d$ could randomly produce different index j with probability $x_j^2/\|\boldsymbol{x}\|^2$.

We use $[n]$ to denote the set $\{1, 2, \cdots, n\}$. We denote the norm $\|\cdot\|$ as the $\|\cdot\|_2$ norm for vector and the spectral norm for matrix, if there is no more explanation. $\|\boldsymbol{A}\|_F = (\sum_{i=1}^m \sum_{j=1}^n a_{ij}^2)^{1/2}$ is the Frobenius norm of matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$. The lowercase form $\boldsymbol{h}_i$ is defined as the i-th column vector of matrix $\boldsymbol{H} \in \mathbb{R}^{d\times d}$. x_i is defined as the i-th component of vector $\boldsymbol{x}$. The tensor product of two matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{p\times q}$ is defined as $\boldsymbol{C} = \boldsymbol{A} \otimes \boldsymbol{B}$. The tensor product operation could be performed between vectors, since vector is one special kind of matrix. The tensor product could be defined between quantum states $|\boldsymbol{x}_1\rangle$ and $|\boldsymbol{x}_2\rangle$, for example, $|\boldsymbol{x}\rangle = |\boldsymbol{x}_1\rangle \otimes |\boldsymbol{x}_2\rangle$. The form $|\boldsymbol{x}_1\rangle \otimes |\boldsymbol{x}_2\rangle$ could also be written as $|\boldsymbol{x}_1\rangle|\boldsymbol{x}_2\rangle$.

We present definitions of smoothness and γ-separation here.

Definition 1. (smoothness) *A function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if it has L-Lipschitz continuous gradient, that is $\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, where $\mathcal{X}$ is the domain of $f(\boldsymbol{x})$.*

Definition 2. (γ-separation) *The set $G = \{a_1, a_2, \cdots, a_n\}$ is said to be γ-separated if $|a_i - a_j| > \gamma, \forall i, j \in [n]$ and $i \neq j$.*

Based on these definitions, we assume that the Hessian $\boldsymbol{H}$ in this article has two properties:

1. Hessian $\boldsymbol{H} \in \mathbb{R}^{d\times d}$ is a r-rank matrix;
2. The absolute value of $\boldsymbol{H}$'s non-zero eigenvalue is ϵ-separated.

The first property is directly derived from the assumption of previous classical non-convex optimization method (Carmon et al., 2018), and the low-rank Hessian case has been observed in neural networks (Gur-Ari et al., 2018). The second property is assumed such that we could distinguish different eigenvalues by their absolute value. We further assume that the Hessian matrix $\boldsymbol{H}$ has the eigen-decomposition $\boldsymbol{H} = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^T$ for the convenience of following discussion.

2.2 TECHNIQUES

The motivation idea behind our approach is to perform the quantum singular value estimation model and then generate eigen-states by the post-selection on the output state. Here we introduce two techniques including oracle models and critical conclusions in previous work.

Quantum Oracle Models (Kerenidis & Prakash, 2016): For the whole paper, we assume the existence of following quantum oracles, and discuss the query complexity of our algorithms to these oracles. Given Hessian $\boldsymbol{H} \in \mathbb{R}^{d \times d}$, we assume that $\boldsymbol{H}$ is stored in a classical data structure such that the following quantum oracles could be implemented:

$$U_H : |i\rangle|0\rangle \rightarrow |i\rangle|\boldsymbol{h}_i\rangle = \frac{1}{\|\boldsymbol{h}_i\|} \sum_{j=1}^{d} h_{ij}|i\rangle|j\rangle, \forall i \in [d], \tag{3}$$

$$V_H : |0\rangle|j\rangle \rightarrow |\tilde{\boldsymbol{h}}\rangle|j\rangle = \frac{1}{\|\boldsymbol{H}\|_F} \sum_{i=1}^{d} \|\boldsymbol{h}_i\| |i\rangle|j\rangle, \forall j \in [d], \tag{4}$$

where $\tilde{\boldsymbol{h}}$ stands for the d-dimensional vector whose i-th component is $\|\boldsymbol{h}_i\|/\|\boldsymbol{H}\|_F$.

The required data structure has a binary tree form. The sign and square value for each entry are stored in different leaves and the value stored in each parent node is the sum of its children's value. A detail description about this data structure can be referred to (Kerenidis & Prakash, 2016). Denote T_H as the time complexity of these oracles.

Quantum Singular Value Estimation (SVE): Given matrix $\boldsymbol{H} \in \mathbb{R}^{d \times d}$ which has the eigenvalue decomposition $\boldsymbol{H} = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j$, previous work (Kerenidis & Prakash, 2016) provided a quantum singular value estimation algorithm, which could be used for estimating singular value or generating eigenstate. Here we briefly introduce their conclusion about the time complexity of their algorithm:

Theorem 3. *(Kerenidis & Prakash, 2016) Suppose quantum accesses to oracles (3) and (4) exist. There is an algorithm which could perform the mapping $\sum_j \beta_j|\boldsymbol{u}_j\rangle \rightarrow \sum_j \beta_j|\boldsymbol{u}_j\rangle||\hat{\lambda}_j|\rangle$ with time complexity $\mathcal{O}(T_H \text{polylog}(d)\epsilon^{-1})$, where $\hat{\lambda}_j \in [\lambda_j - \epsilon\|\boldsymbol{H}\|_F, \lambda_j + \epsilon\|\boldsymbol{H}\|_F]$ with probability at least $1 - 1/\text{poly}(d)$.*

3 QUANTUM NEGATIVE CURVATURE FINDING ALGORITHM

Our main contribution in this section is the quantum Negative Curvature Finding (quantum NCF) algorithm presented in Algorithm 1. The quantum NCF algorithm solves the NCF problem by providing the target state $|\boldsymbol{u}_t\rangle$ (for case (1)) or making the non-vector statement (for case (2)). The target state $|\boldsymbol{u}_t\rangle$ corresponds to the eigenvector $\boldsymbol{u}_t$ which satisfies the condition $\boldsymbol{u}_t^T \boldsymbol{H} \boldsymbol{u}_t \leq -\alpha + \epsilon/2$. Here we present a tighter restrict on the target state $|\boldsymbol{u}_t\rangle$ to keep a $\epsilon/2$ redundancy for the classical read-out of the quantum state. The quantum NCF Algorithm uses the Proper Eigenvalue Labelling (Algorithm 2) and the Target State Generating Algorithm (Algorithm 5 in Appendix) as subroutines proposed in Section 3.2 and Section 3.3, respectively.

3.1 CHALLENGES TO DEVELOP QUANTUM NCF ALGORITHM

The core technical component of our quantum algorithm for the NCF problem is the quantum SVE algorithm. However, there are three major challenges that we have to overcome.

Firstly, the positive-negative eigenvalue problem. In the negative curvature finding problem, we are interested in obtaining eigenvectors with negative eigenvalues. Hence, we cannot directly apply the

---

Algorithm 1 Quantum Negative Curvature Finding (Quantum NCF) Algorithm

---

Input: Quantum access to oracles U_H and V_H. The parameter ϵ and α in the **NCF problem**.
Output: The target state $|u_t\rangle$ whose corrsponding classical unit vector u_t satisfies the condition $u_t^T H u_t \leq -\alpha + \epsilon/2$; or a statement with high probability that there is no such kind of unit vector u which satisfies the condition $u^T H u \leq -\alpha$.
1: Label the **proper**(less than $-\alpha + \epsilon/2$) eigenvalue of H (Proper Eigenvalue Labelling).
2: **if** the least eigenvalue of H is less than $-\alpha + \epsilon/2$, **then**
3:     generate the target state (Target State Generating Algorithm) and output the state;
4: **else**,
5:     claim that there is no such kind of unit vector u which satisfies the condition $u^T H u \leq -\alpha$.
6: **end if**

---

quantum SVE algorithm since it only gives the estimation on $|\lambda_j|$. In order to overcome this critical issue, we develop Algorithm 4 in Appendix to label negative eigenvalues.

Secondly, since the quantum SVE Algorithm presents ϵ-estimation on singular values with time complexity $\mathcal{O}(T_H \|H\|_F \mathrm{polylog}(d)\epsilon^{-1})$(Theorem 3), we need to provide a tight upper bound for the Frobenius norm $\|H\|_F$, which is shown in Lemma 1:

Lemma 1. *Suppose $H \in \mathbb{R}^{d \times d}$ is the Hessian matrix derived from the function $f : \mathbb{R}^d \to \mathbb{R}$ which has the L-Lipschitz continuous gradient. Thus the Frobenius norm of H has the upper bound $\|H\|_F \leq \sqrt{r}L$, where r is the rank of H.*

Finally, the input-state problem. For the general superposition state $\sum_j \beta_j |u_j\rangle$, the output state of quantum SVE algorithm has the form $\sum_j \beta_j |u_j\rangle ||\hat{\lambda}_j|\rangle$. We could generate different pure state $|u_j\rangle$ with probability $|\beta_j|^2$ by the measurement on eigenvalue register. Thus in order to guarantee a small time complexity, we need to prepare a special input state such that the overlap between the input and the target state is relatively large. We briefly summarize our conclusion on the time complexity of Algorithm 1 in Proposition 2.

Proposition 2. *Algorithm 1 takes time $\mathcal{O}(T_H \|H\|_F^5 \mathrm{polylog}(d)\epsilon^{-1})$ to solve the negative curvature finding problem by providing the target state $|u_t\rangle$ or making the statement that there is no unit vector satisfies the condition $u^T H u \leq -\alpha$.*

3.2 Positive-Negative Eigenvalue Discrimination

In this section, we propose an algorithm aiming to label the target eigenvalue which is less than $-\alpha + \epsilon/2$. This algorithm helps verifying the existence of solution to the NCF problem and generating the target state. Since the eigenvalue information of H is unknown to us, we need to build the Algorithm 2 to label the **proper** eigenvalue, which would benefit the target state generation task in the following section. The **proper** eigenvalue means the eigenvalue is less than $-\alpha + \epsilon/2$. We view this kind of eigenvalue as our target eigenvalue.

The mean idea of Algorithm 2 is to use the input state $\frac{1}{\|H\|_F} \sum_{j=1}^r \lambda_j |u_j\rangle |u_j\rangle$ for the quantum SVE model and obtain the state:

$$\frac{1}{\|H\|_F} \sum_{j=1}^r \lambda_j |u_j\rangle |u_j\rangle ||\tilde{\lambda}_j|\rangle.$$

The measurement on the eigenvalue register would lead this entangled state collapse to different states $|u_j\rangle |u_j\rangle$ for $j \in [r]$. We could obtain the state $|u_j\rangle$ by neglecting the state in any other register. Using state $|u_j\rangle$ to apply the PNED algorithm (Algorithm 4 in appendix) could provide a discrimination on the positive and negative of the corresponding eigenvalue λ_j. Thus, we could label the **proper** eigenvalue, for the case that the least eigenvalue is less than $-\alpha + \epsilon/2$; or make the non-vector statement, for the case that all of eigenvalues are greater than $-\alpha$. We present the time complexity of Algorithm 2 in Theorem 4.

Theorem 4. *Algorithm 2 could label the **proper** eigenvalue of H with probability $1 - 1/\mathrm{poly}(d)$, or claim with high probability that there is no unit vector u which satisfies $u^T H u < -\alpha$, with time complexity $\mathcal{O}(T_H \|H\|_F^5 \mathrm{polylog}(d)\epsilon^{-1})$.*

Algorithm 2 Proper Eigenvalue Labelling

Input: Quantum access to oracles U_H and V_H. The parameter ϵ and α in the NCF problem.
Output: A **proper** label to the singularvalue $|\lambda_j|$ such that $\lambda_j \leq -\alpha + \epsilon/2$ with probability $1 - \delta$, or a non-vector statement that there is no unit vector $\boldsymbol{u}$ which satisfies $\boldsymbol{u}^T \boldsymbol{H} \boldsymbol{u} < -\alpha$.

1: **for** $k = 1$ to $\frac{4\|\boldsymbol{H}\|_F^2}{\alpha^2}(2\frac{4\|\boldsymbol{H}\|_F^2}{\alpha^2}\log\frac{1}{\delta} + 3)$ **do**
2:     Create the state $\frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle$.
3:     Apply the quantum SVE model to obtain the state $\frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle||\tilde{\lambda}_j|\rangle$, where $|\tilde{\lambda}_j| \in |\lambda_j| \pm \epsilon/4$ with probability $1 - 1/\text{poly}(d)$.
4:     Measure the eigenvalue register and mark the result.
5:     Use the rest state in the first register as the input to apply the PNED algorithm.
6: **end for**
7: Count the result in step 4 and step 5 to obtain the sequence $\{(|\tilde{\lambda}_j|, n_j, m_j)\}_{j=1}^r$. n_j is the number of resulting $|\tilde{\lambda}_j|$ in step 4, and m_j is the number of resulting 1 in step 5 for different $|\tilde{\lambda}_j|$.
8: **if** $\frac{m_j}{n_j} < \frac{1}{2}$ for all $j \in [r]$, **then** make the non-vector statement;
9: **else**, choose the largest $|\tilde{\lambda}_j|$ which satisfies the condition $\frac{m_j}{n_j} > \frac{1}{2}$.
10:     **if** $|\tilde{\lambda}_j| < \alpha - \epsilon/4$, **then** make the no-vector statement;
11:     **else**, label eigenvalue λ_j as the **proper** eigenvalue.
12:     **end if**
13: **end if**

3.3 Target State Generating

Suppose the result of Algorithm 2 implies the existence of the target eigenvector $\boldsymbol{u}_t$, which satisfies $\boldsymbol{u}_t^T \boldsymbol{H} \boldsymbol{u}_t \leq -\alpha + \epsilon/2$. Then in order to give a solution to the NCF problem, we need to obtain the vector $\boldsymbol{u}_t$. In this section we construct a quantum procedure to generate the state $|\boldsymbol{u}_t\rangle$. The classical read-out of $|\boldsymbol{u}_t\rangle$, which means to estimate vector $\boldsymbol{u}_t$ from quantum state $|\boldsymbol{u}_t\rangle$, will be discussed in the following section. The idea is very similar to Algorithm 2. We still use state $\frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle$ as the input of quantum SVE algorithm to obtain state:

$$\frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle||\tilde{\lambda}_j|\rangle.$$

Suppose λ_t denotes the eigenvalue of $|\boldsymbol{u}_t\rangle$ that $\lambda_t \leq -\alpha + \epsilon/2$. The measurement on the eigenvalue register could generate state $|\boldsymbol{u}_t\rangle|\boldsymbol{u}_t\rangle$ with probability $P_t = \frac{\lambda_t^2}{\|\boldsymbol{H}\|_F^2} \geq \frac{\alpha^2}{4\|\boldsymbol{H}\|_F^2}$. Thus the probability of generating at least one state $|\boldsymbol{u}_t\rangle$ in $N = \left\lceil 4\frac{\|\boldsymbol{H}\|_F^2}{\alpha^2}\log\frac{1}{\delta} \right\rceil + 1$ times of measurement is $1 - (1 - P_t)^N$. There is: $1 - (1 - P_t)^N \geq 1 - e^{-NP_t} \geq 1 - e^{-\log(1/\delta)} = 1 - \delta$. So state $|\boldsymbol{u}_t\rangle$ could be generated in N times of measurement with probability at least $1 - \delta$. By considering the time complexity to run the quantum SVE algorithm ($\mathcal{O}(T_H\|\boldsymbol{H}\|_F\text{polylog}(d)\epsilon^{-1})$) and setting the probability error bound $\delta = 1/\text{poly}(d)$, we could derive the time complexity of generating target state in Theorem 5. The detail of Target State Generating algorithm is in Appendix (Algorithm 5).

Theorem 5. *Suppose that Hessian $\boldsymbol{H}$ has the eigenvector $\boldsymbol{u}_t$ with eigenvalue less than $-\alpha + \frac{\epsilon}{2}$. Then state $|\boldsymbol{u}_t\rangle$ could be generated in time $\mathcal{O}(T_H\|\boldsymbol{H}\|_F^3\text{polylog}(d)\epsilon^{-1})$ with probability at least $1 - 1/\text{poly}(d)$.*

4 State Read-out

In this section, we propose an efficient algorithm to read-out the classical vector $\boldsymbol{u}_t$ from the quantum state $|\boldsymbol{u}_t\rangle$. Generally, the classical read-out of a d-dimensional quantum state takes at least $\tilde{\mathcal{O}}(d)$ times of measurement to obtain the form $|\boldsymbol{x}\rangle = \sum_{i=1}^d x_i\boldsymbol{e}_i$. Thus the classical read-out of the required state could offset the exponential speed-up (Aaronson, 2015) provided in many quantum machine learning algorithms. In order to avoid this problem, we propose a quantum-classical hybrid procedure to rewrite the target state $|\boldsymbol{u}_t\rangle$ as the linear combination of r states, which is selected from column vectors of Hessian $\boldsymbol{H}$. The main result is stated in Proposition 3.

Proposition 3. *The classical description of the target state $|\boldsymbol{u}_t\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{h}_{g(i)}\rangle$[1] could be presented in time $\mathcal{O}(T_H \mathrm{polylog}(d) \mathrm{poly}(r) \epsilon^{-5})$ with error bounds in $\epsilon/2$, when the complete basis set $\{\boldsymbol{h}_{g(j)}\}_{j=1}^{r}$ is given.*

Recall that our Hessian matrix $\boldsymbol{H} \in \mathbb{R}^{d \times d}$ has the eigendecomposition $\boldsymbol{H} = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^T$, where λ_j is the eigenvalue of H with respect to eigenvector $\boldsymbol{u}_j$. Then, any eigenvector of $\boldsymbol{H}$ that corresponds to a non-zero eigenvalue could be represented as the linear combination of vectors in $\{\boldsymbol{h}_j\}_{j=1}^{d}$ because $\boldsymbol{H}\boldsymbol{u}_j = \lambda_j \boldsymbol{u}_j$ can be written as $\sum_{i=1}^{d} \boldsymbol{h}_i u_j^{(i)} = \lambda_j \boldsymbol{u}_j$. Since $\boldsymbol{H}$ has the rank of r, there exists a subset of complete basis $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$ of the column space, which is sampled from the set $\{\boldsymbol{h}_j\}_{j=1}^{d}$. Thus, any eigenvector $\boldsymbol{u}_j$ could also be represented as the linear combination of vectors in $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$.

Back to the state read-out problem, denote $\boldsymbol{s}_i$ as $\boldsymbol{h}_{g(i)}$ for simplicity. Suppose the target state $|\boldsymbol{u}_t\rangle$ that we generated in the previous section can be written as $|\boldsymbol{u}_t\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle$, where $\{x_i\}_{i=1}^{r}$ are coordinates of the state $|\boldsymbol{u}_t\rangle$ under the basis $\{|\boldsymbol{s}_i\rangle\}_{i=1}^{r}$. Thus, instead of simply reading out components of vector $\boldsymbol{u}_t$, we could get the classical description of $|\boldsymbol{u}_t\rangle$ by calculating each x_i. Note that the complete basis $\{|\boldsymbol{s}_i\rangle\}_{i=1}^{r}$ is not unique and we only need to identify one of them.

Note that the updating operation for the l-th iteration $\boldsymbol{z}^{(l+1)} = \boldsymbol{z}^{(l)} + \eta \boldsymbol{u}_t$ in general non-convex algorithms does not have a straightforward quantum generalization. In other words, we could not perform the operation $|\boldsymbol{z}^{(l+1)}\rangle \propto |\boldsymbol{z}^{(l)}\rangle + \eta |\boldsymbol{u}_t\rangle$ directly because the target state $|\boldsymbol{u}_t\rangle$ is generated by post-selection instead of standard unitary operations. However, we can use the linear-combination-of-states method (Shao, 2018) to generate the form $\sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle$ and circumvent the problem caused by the post-selection.

We also notice some recent breakthroughs about quantum-inspired algorithms (Tang, 2019; Arrazola et al., 2019), which are based on sampling technics and FKV Algorithm (Frieze et al., 2004). These quantum-inspired algorithms could perform approximate SVD which also outputs eigenvector as the linear-sum on a group of column vectors. However, in order to cover the whole column space, the quantum-inspired algorithm need to sample at least $\mathcal{O}(\frac{r^2}{\epsilon^2})$ number of columns as basis, while our method generate the linear-sum form exactly on r columns.

4.1 Complete Basis Selection

In this section, we develop a quantum algorithm to select a subset $S_I = \{g(1), g(2), \cdots, g(r)\}$ from $[d]$, which corresponds to the complete basis $\{|\boldsymbol{h}_{g(i)}\rangle\}_{i=1}^{r}$. The quantum complete basis selection algorithm can be viewed as the quantum version of Gram-Schmidt orthogonalization: firstly we choose $|\boldsymbol{t}_1\rangle = |\boldsymbol{h}_{g(1)}\rangle$ from the set $\{|\boldsymbol{h}_j\rangle\}_{j=1}^{d}$; then given state set $\{|\boldsymbol{t}_m\rangle\}_{m=1}^{l}$, we choose $|\boldsymbol{t}_{l+1}\rangle \propto |\boldsymbol{h}_{g(l+1)}\rangle - \sum_{m=1}^{l} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_{g(l+1)}\rangle$ from the state set $\{|\boldsymbol{h}_j\rangle - \sum_{m=1}^{l} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\}_{j=1}^{d}$. Since the chosen $|\boldsymbol{t}_{l+1}\rangle$ is orthogonal to states $\{|\boldsymbol{t}_m\rangle\}_{m=1}^{l}$ for each iteration $l \in [r-1]$, the state set $\{|\boldsymbol{t}_m\rangle\}_{m=1}^{r}$ forms an orthonormal basis. Note that the state $|\boldsymbol{t}_m\rangle$ is generated along with an index $g(m)$ for $m \in [r]$, so we would obtain a complete basis index set $\{g(m)\}_{m=1}^{r}$ after the implementation of this quantum algorithm. The quantum complete basis selection algorithm is provided in Algorithm 3, in which we denote:

$$|\phi_1^{(l)}\rangle \equiv \frac{1}{\|\boldsymbol{H}\|_F} \sum_{j=1}^{d} \|\boldsymbol{h}_j\| |j\rangle \left\{ \left[|\boldsymbol{h}_j\rangle - \sum_{m=1}^{l} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle \right] |0\rangle - \sum_{m=1}^{l} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle |1\rangle \right\}, \quad (5)$$

as the initial state in each iteration. The time complexity of Algorithm 3 is analyzed in Theorem 6. The detail about Algorithm 3 is in Appendix. We also provide Lemma 2 which provides the time complexity of confirming whether a given set $\{\boldsymbol{s}_i\}_{i=1}^{r}$ are linearly independent.

Theorem 6. *The Algorithm 3 takes time $\mathcal{O}(T_H \mathrm{poly}(r) \epsilon^{-2} r^{2\log(4r\|\boldsymbol{H}\|_F/\epsilon)})$ to find an index set $\{g(i)\}_{i=1}^{r}$, which forms a complete basis $\{|\boldsymbol{h}_{g(i)}\rangle\}_{i=1}^{r}$ with probability at least 3/4.*

Lemma 2. *It takes $\mathcal{O}(r^3)$ time to check whether the vector set $\{\boldsymbol{s}_i\}_{i=1}^{r}$ is linearly independent when the classical access to Hessian $\boldsymbol{H}$ is given, where $\boldsymbol{s}_i$ is sampled from column vectors of matrix $\boldsymbol{H}$.*

---

[1] $g(i)$ is the index of the i-th column vector in the complete basis.

Algorithm 3 Complete Basis Selection

Input: Quantum access to oracle U_H and V_H.
Output: The index set of the complete basis: $S_I = \{g(i)\}_{i=1}^r$.
 1: Initialize the index set $S_I = \varnothing$.
 2: **for** $l = 0$ to $r - 1$ **do**
 3:     Create the state $|\phi_1^{(l)}\rangle$ defined in Equation (5).
 4:     Measure the third register of state $|\phi_1^{(l)}\rangle$ multiple times to get the state $|\phi_2^{(l)}\rangle$ which is pro-
        portional to $\frac{1}{\|H\|_F} \sum_{j=1}^d |j\rangle \|h_j\| \left[|h_j\rangle - \sum_{m=1}^l |t_m\rangle\langle t_m|h_j\rangle \right]$.
 5:     Measure the first register and record the result as $g(l+1)$.
 6:     Denote $|t_{l+1}\rangle$ as the state proportional to $|h_{g(l+1)}\rangle - \sum_{m=1}^l |t_m\rangle\langle t_m|h_{g(l+1)}\rangle$.
 7:     Update the index set $S_I = S_I \cup \{g(l+1)\}$.
 8: **end for**

4.2 COORDINATES ESTIMATION

Assume the complete basis $\{|s_1\rangle, |s_2\rangle, \cdots, |s_r\rangle\}$ has been selected out in Section 4.1. Thus the read-out problem could be viewed as solving the equation $|u_t\rangle = \sum_{i=1}^r x_i|s_i\rangle$, where $x_i \in \mathbb{R}$ are unknown variables. The coordinate $\{x_i\}_{i=1}^r$ could be obtained by solving the r-dimensional linear equation system $Cx = b$, where $b_i = \langle u_t|s_i\rangle$ and $c_{ij} = \langle s_i|s_j\rangle$ for $i, j \in [r]$. The equation $Cx = b$ could be solved classically in at most $\mathcal{O}(r^3)$ time. Note that we can only get the approximation to c_{ij} or b_i instead of the exact value. Theorem 7 verifies the impact of the approximate error to c_{ij} or b_i on the read-out of the target state.

Theorem 7. *Suppose $\tilde{c}_{jk}$ is the ϵ_1-approximation to $c_{jk} = \langle s_j|s_k\rangle$ and $\tilde{b}_j$ is the ϵ_2-approximation to $b_j = \langle u_t|s_j\rangle$, $\forall j, k \in [r]$, where $\epsilon_1 = \frac{\epsilon}{6r^2\|C^{-1}\|^2}$ and $\epsilon_2 = \frac{\epsilon}{6r\|C^{-1}\|}$. Denote vector $\tilde{x} \in \mathbb{R}^r$ as the solution of $\tilde{C}x = \tilde{b}$. Then $\tilde{x}$ could lead an approximate eigenvector $\tilde{u}_t = \sum_{j=1}^r \tilde{x}_j s_j$, such that $\|\tilde{u}_t - u_t\| \leq \epsilon/2$.*

We propose several methods in Appendix to estimate overlap $b_i = \langle u_t|s_i\rangle$ and $c_{ij} = \langle s_i|s_j\rangle$, which are based on the Quantum SWAP Test (Buhrman et al., 2001) and the Hadamard Test. Our proposed quantum algorithms could present ϵ_1-estimation to $c_{ij} = \langle s_i|s_j\rangle$ in time $\mathcal{O}(T_H\epsilon_1^{-2})$ and ϵ_2-estimation to $b_i = \langle u_t|s_i\rangle$ in time $\mathcal{O}((T_{Input} + T_H)\epsilon_2^{-4})$, where T_{Input} is the time to generate state $|u_t\rangle$. Since the time complexity to generate the target state is $\mathcal{O}(T_H\|H\|_F^3\text{polylog}(d)\epsilon^{-1})$ as proposed in Theorem 5, we could derive Proposition 3.

Considering the time complexity $\mathcal{O}(T_H\|H\|_F^5\text{polylog}(d)\epsilon^{-1})$ to label the proper eigenvalue and the time complexity $\mathcal{O}(T_H\text{poly}(r)\epsilon^{-2}r^{2\log(4r\|H\|_F/\epsilon)})$ to generate the complete basis set, we could solve the NCF problem in time $\mathcal{O}(T_H\text{polylog}(d)\text{poly}(r)\epsilon^{-2}(\epsilon^{-3} + r^{2\log(4r\|H\|_F/\epsilon)}))$ by providing the target vector in the form $u_t = \sum_{i=1}^r x_i h_{g(i)}/\|h_{g(i)}\|$ with error bounded by ϵ or making the non-vector statement.

5 CONCLUSION

We propose an efficient quantum algorithm for the Negative Curvature Finding problem, which is a critical subroutine in many second-order methods for non-convex optimization. The proposed quantum algorithm could produce the target state in time $\mathcal{O}(T_H\epsilon^{-1}\text{poly}(r)\text{polylog}(d))$ with probability $1 - 1/\text{poly}(d)$, which runs exponentially faster than existing classical methods. Moreover, we propose an efficient hybrid quantum-classical algorithm for the efficient classical read-out of the target state with time complexity $\mathcal{O}(T_H\text{poly}(r)\text{polylog}(d)\epsilon^{-2}(\epsilon^{-3} + r^{2\log(4r\|H\|_F/\epsilon)}))$, which is exponentially faster on the degree of d than existing general quantum state read-out methods.

REFERENCES

Scott Aaronson. Quantum machine learning algorithms : Read the fine print. 2015.

Naman Agarwal, Zeyuan Allen-Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima faster than gradient descent. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1195–1199. ACM, 2017.

Juan Miguel Arrazola, Alain Delgado, Bhaskar Roy Bardhan, and Seth Lloyd. Quantum-inspired algorithms in practice. *arXiv preprint arXiv:1905.10415*, 2019.

Harry Buhrman, Richard Cleve, John Watrous, and Ronald De Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001.

Yair Carmon and John C Duchi. Gradient descent efficiently finds the cubic-regularized non-convex newton step. *arXiv preprint arXiv:1612.00547*, 2016.

Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pp. 192–204, 2015.

Andrew R Conn, Nicholas IM Gould, and Ph L Toint. *Trust region methods*, volume 1. Siam, 2000.

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pp. 2933–2941, 2014.

Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *J. ACM*, 51(6):1025–1041, November 2004. ISSN 0004-5411. doi: 10.1145/1039488.1039494. URL http://doi.acm.org/10.1145/1039488.1039494.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle pointsonline stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pp. 797–842, 2015.

Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*, pp. 409–426. Springer, 1994.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1724–1732. JMLR. org, 2017.

Chi Jin, Praneeth Netrapalli, Rong Ge, Sham M Kakade, and Michael I Jordan. Stochastic gradient descent escapes saddle points efficiently. *arXiv preprint arXiv:1902.04811*, 2019.

Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. *arXiv preprint arXiv:1603.08675*, 2016.

Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257, 2016.

Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10(9):631, 2014.

Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical review letters*, 113(13):130503, 2014.

Patrick Rebentrost, Adrian Steffens, Iman Marvian, and Seth Lloyd. Quantum singular-value decomposition of nonsparse low-rank matrices. *Physical review A*, 97(1):012327, 2018.

Alex Schwarzenberg-Czerny. On matrix factorization and efficient least squares solution. *Astronomy and Astrophysics Supplement Series*, 110:405, 1995.

Changpeng Shao. From linear combination of quantum states to grover's searching algorithm. *arXiv preprint arXiv:1807.09693*, 2018.

Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pp. 217–228. ACM, 2019.

A APPENDIX

A.1 THE PROOF OF THEOREM 2

Proof. The time complexity of Algorithm 1 could be directly obtained by the time complexity of Algorithm 2 and Algorithm 5, whose complexity analysis are presented in Theorem 4 and Theorem 5, respectively. □

A.2 THE PROOF OF LEMMA 1

Proof. Assume $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_d$ are eigenvalues of H, we have:

$$\min_{\|v\|=1} v^T H v \leq \lambda_j \leq \max_{\|v\|=1} v^T H v, \ \forall j \in [d].$$

By the definition of the Hessian matrix, for unit vector v, we have:

$$H v = \nabla^2 f(x) v = \lim_{h \to 0} \frac{\nabla f(x + hv) - \nabla f(x)}{h}.$$

From above equation, we can obtain:

$$v^T H v \leq \|v\| \cdot \|H v\| \leq \lim_{h \to 0} \frac{\|\nabla f(x+hv) - \nabla f(x)\|}{h} \leq \lim_{h \to 0} \frac{L\|hv\|}{h} = L,$$

and $v^T H v \geq -\|v\| \cdot \|H v\| \geq -\lim_{h \to 0} \frac{\|\nabla f(x+hv) - \nabla f(x)\|}{h} \geq -\lim_{h \to 0} \frac{L\|hv\|}{h} = -L.$

Thus, the eigenvalue λ_j is bounded in $[-L, L]$ for all $j \in [d]$.

We have $\|H\|_F^2 = \sum_i \sum_j h_{ij}^2 = Tr(H \cdot H) = \sum_j \lambda_j(H^2) \leq rL^2$, so $\|H\|_F \leq \sqrt{r}L$. □

Lemma 3. Hoeffding's inequality (Hoeffding, 1994)

Suppose $X_1, X_2, \cdots, X_n$ are independent random variables with bounds $X_i \in [a_i, b_i], \forall i \in [n]$. Define $\overline{X} = \frac{1}{n} \sum_{i=1}^n X_i$, then $\forall \epsilon > 0$, we have:

$$P(\overline{X} - E[\overline{X}] \geq \epsilon) \leq \exp\left(-\frac{2n^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\right), \tag{6}$$

and

$$P(\overline{X} - E[\overline{X}] \leq -\epsilon) \leq \exp\left(-\frac{2n^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\right). \tag{7}$$

A.3 THE DETAIL ABOUT ALGORITHM 4

In Algorithm 4, $P \in \mathbb{R}^{d^2 \times d}$ is the matrix whose column vector $p_i = e_i \otimes \frac{h_i}{\|h_i\|}$ for $i \in [d]$, and $Q \in \mathbb{R}^{d^2 \times d}$ is the matrix whose column vector $q_j = \frac{\tilde{h}}{\|H\|_F} \otimes e_j$ for $j \in [d]$. h_i is the i-th column vector of matrix H and $\tilde{h}$ is a d-dimensional vector whose i-th component is $\|h_i\|$. It can be directly obtained that the matrix P and Q satisfy the decomposition $H/\|H\|_F = P^T Q$ and have property $P^T P = Q^T Q = I$. Mappings $|x\rangle|0\rangle \to |Px\rangle$ and $|0\rangle|x\rangle \to |Qx\rangle$ can be performed by the quantum oracle U_H and V_H respectively.

Theorem 8. Positive-negative eigenvalue discrimination. *For eigenvalue λ of H with property $|\lambda| \geq a$, one could run Algorithm 4 for $n = 2[\frac{\|H\|_F^2}{a^2} \log \frac{1}{\delta} - \frac{1}{2}] + 3$ times, to make a statement that λ is positive or negative, with probability $1 - \delta$.*

Algorithm 4 Positive-Negative Eigenvalue Discrimination(PNED) Algorithm

---

Input: Quantum access to oracles U_H and V_H. Eigenstate $|u\rangle$ which corresponds to eigenvalue λ.
Output: A random boolean variable with distribution $P(0) = \frac{1+\lambda/\|H\|_F}{2}$ and $P(1) = \frac{1-\lambda/\|H\|_F}{2}$.
 1: Create state $|u\rangle|0\rangle|0\rangle$.
 2: Apply the Hadmard gate on the third register to obtain the state $\frac{1}{\sqrt{2}}(|u\rangle|0\rangle|0\rangle + |u\rangle|0\rangle|1\rangle)$.
 3: Apply the Controlled-SWAP gate to obtain the state $\frac{1}{\sqrt{2}}(|u\rangle|0\rangle|0\rangle + |0\rangle|u\rangle|1\rangle)$.
 4: Apply gate $U_H \otimes |0\rangle\langle0| + V_H \otimes |1\rangle\langle1|$ on the state to obtain $\frac{1}{\sqrt{2}}(|Pu\rangle|0\rangle + |Qu\rangle|1\rangle)$.
 5: Apply the Hadmard gate on the third register to obtain the state $\frac{|Pu\rangle+|Qu\rangle}{2}|0\rangle + \frac{|Pu\rangle-|Qu\rangle}{2}|1\rangle$.
 6: Measure the third register and output the result.

Proof. The measurement in step 6 of Algorithm 4 outputs 1 with probability:

$$P(1) = \|\frac{|Pu\rangle - |Qu\rangle}{2}\|^2 = \frac{1}{4}((\langle Pu| - \langle Qu|)(|Pu\rangle - |Qu\rangle). \tag{8}$$

Note that $\langle Pu|Qu\rangle = u^T P^T Q u = \frac{1}{\|H\|_F} u^T H u = \frac{\lambda}{\|H\|_F}$, so $P(1) = \frac{1-\lambda/\|H\|_F}{2}$. Similarly we have $P(0) = \frac{1+\lambda/\|H\|_F}{2}$.

Suppose that we need $2x+1$ times of measurement to give an $1-\delta$ correct statement about whether $\lambda > 0$ or $\lambda < 0$. The problem can be viewed as the biased coin problem. Define random variables X_i such that $P(X_i = 1) = p$ and $P(X_i = 0) = 1 - p$ and $S_n = \sum_{i=1}^{n} X_i$. Then there has the Hoeffding's inequality $P(S_n/n - p \le -\epsilon) \le e^{-2n\epsilon^2}$ and $P(S_n/n - p \ge \epsilon) \le e^{-2n\epsilon^2}$.

Back to the problem, suppose $\lambda < 0$, by setting $n = 2x+1$, $n(p - \epsilon) = x$ and $P = \frac{1-\lambda/\|H\|_F}{2}$, we have:

$$P(S_{2x+1} \le x) \le \exp(-2(2x+1)[\frac{1 - \lambda/\|H\|_F}{2} - \frac{x}{2x+1}]^2) < e^{-\frac{2x+1}{2}\frac{\lambda^2}{\|H\|_F^2}} \le e^{-\frac{2x+1}{2}\frac{a^2}{\|H\|_F^2}}.$$

Similarly for $\lambda > 0$, there is $P(S_{2x+1} \ge x) \le e^{-\frac{2x+1}{2}\frac{a^2}{\|H\|_F^2}}$.

Let $e^{-\frac{2x+1}{2}\frac{a^2}{\|H\|_F^2}} \le \delta$, we have $x \ge [\frac{\|H\|_F^2}{a^2} \log \frac{1}{\delta} - \frac{1}{2}] + 1$. $\qquad\square$

A.4 THE PROOF OF THEOREM 4

Proof. The input state $\frac{1}{\|H\|_F} \sum_{j=1}^{r} \lambda_j |u_j\rangle|u_j\rangle$ could be generated with oracles U_H and V_H:

$$|0\rangle|0\rangle \xrightarrow{V_H} \frac{1}{\|H\|_F} \sum_{i=1}^{d} \|h_i\||i\rangle|0\rangle \xrightarrow{U_H} \frac{1}{\|H\|_F} \sum_{i=1}^{d} \sum_{j=1}^{d} h_{ij}|i\rangle|j\rangle. \tag{9}$$

Since H has the eigen-decomposition $H = \sum_{k=1}^{r} \lambda_k u_k u_k^T$, we could rewrite entry h_{ij} as $h_{ij} = \sum_{k=1}^{r} \lambda_k u_k^{(i)} u_k^{(j)}$, where $u_k^{(i)}$ is the i-th component of vector u_k. Thus the state $\frac{1}{\|H\|_F} \sum_{i=1}^{d} \sum_{j=1}^{d} h_{ij}|i\rangle|j\rangle$ could be written as:

$$\frac{1}{\|H\|_F} \sum_{i=1}^{d} \sum_{j=1}^{d} \sum_{k=1}^{r} \lambda_k u_k^{(i)} u_k^{(j)}|i\rangle|j\rangle = \frac{1}{\|H\|_F} \sum_{k=1}^{r} \lambda_k |u_k\rangle|u_k\rangle.$$

Then we apply the quantum SVE model on this state. In order to give $\epsilon/4$-estimation on the singular value, the time complexity to run the quantum SVE algorithm is $\mathcal{O}(T_H\|H\|_F\mathrm{polylog}(d)\epsilon^{-1})$ by Theorem 3.

Suppose there are eigenvalues λ_j which are less than $-\alpha + \epsilon/2$. We denote the least one as λ_t and label it as the **proper** eigenvalue. By Theorem 8, we need to generate $n_t = 2[\frac{\|H\|_F^2}{\lambda_t^2} \log \frac{1}{\delta} - \frac{1}{2}] + 3$

numbers of state $|\boldsymbol{u}_t\rangle$ in order to guarantee that $\lambda_t < 0$ with probability $1 - \delta$. Note that the probability of generating state $|\boldsymbol{u}_t\rangle$ in each iteration of step 2-4 in Algorithm 2 is $P_t = \frac{\lambda_t^2}{\|\boldsymbol{H}\|_F^2}$. So averagely we need to perform step 2-4 in Algorithm 2 for $n = \frac{\|\boldsymbol{H}\|_F^2}{\lambda_t^2}\{2[\frac{\|\boldsymbol{H}\|_F^2}{\lambda_t^2} \log \frac{1}{\delta} - \frac{1}{2}] + 3\}$ times. The number n can be roughly upper bounded by $\frac{4\|\boldsymbol{H}\|_F^2}{\alpha^2}(2\frac{4\|\boldsymbol{H}\|_F^2}{\alpha^2} \log \frac{1}{\delta} + 3)$, since for negative curvature case $\epsilon < \alpha$, we have $|\lambda_t| = \alpha - \epsilon/2 > \alpha/2$.

By considering the time complexity to run the quantum SVE algorithm $(\mathcal{O}(T_H\|\boldsymbol{H}\|_F \text{polylog}(d)\epsilon^{-1}))$ and setting the probability error bound $\delta = 1/\text{poly}(d)$, we could derive that the time complexity of Algorithm 2 is $\mathcal{O}(T_H\|\boldsymbol{H}\|_F^5 \text{polylog}(d)\epsilon^{-1})$. $\qquad\square$

A.5 TARGET STATE GENERATING ALGORITHM

Algorithm 5 Target State Generating

Input: Quantum access to oracles U_H and V_H. Parameters α and ϵ in the **NCF problem**.
Output: The target state $|\boldsymbol{u}_t\rangle$ with property $\langle\boldsymbol{u}_t|H|\boldsymbol{u}_t\rangle = \lambda_t \leq -\alpha + \epsilon/2$.
1: **for** $k = 1$ to $\left[4\frac{\|\boldsymbol{H}\|_F^2}{\alpha^2} \log \frac{1}{\delta}\right] + 1$ **do**
2:    Create the state $\frac{1}{\|\boldsymbol{H}\|_F} \sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle$.
3:    Apply the quantum SVE model to obtain the state $\frac{1}{\|\boldsymbol{H}\|_F} \sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle||\tilde{\lambda}_j|\rangle$, where $|\tilde{\lambda}_j| \in |\lambda_j| \pm \epsilon/4$ with probability $1 - 1/\text{poly}(d)$.
4:    Measure the eigenvalue register and mark the result.
5:    **if** the eigenvalue measured in in step 4 is labelled to be **proper** in Algorithm 2, **then**
6:     output the state in the first register as the target state.
7:    **end if**
8: **end for**

A.6 THE DETAIL ABOUT ALGORITHM 3

Consider the state:

$$\left[\prod_{j=1}^l |g(j)\rangle\right] \otimes |\phi_1^{(l)}\rangle \equiv \frac{1}{\|\boldsymbol{H}\|_F}\left[\prod_{j=1}^l |g(j)\rangle\right]\sum_{j=1}^d \|\boldsymbol{h}_j\|\,|j\rangle\left\{\left[|\boldsymbol{h}_j\rangle - \sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right]|0\rangle - \sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle|1\rangle\right\},$$
(10)

we note that the state $|\phi_1^{(l)}\rangle$ in Step 3 of Algorithm 3 can be generated by the following procedure:

$$\left[\prod_{j=1}^l |g(j)\rangle\right]|0\rangle|0\rangle \xrightarrow{(a)} \frac{1}{\|\boldsymbol{H}\|_F}\left[\prod_{j=1}^l |g(j)\rangle\right]\sum_{j=1}^d \|\boldsymbol{h}_j\|\,|j\rangle|\boldsymbol{h}_j\rangle \tag{11}$$

$$\xrightarrow{(b)} \frac{1}{\|\boldsymbol{H}\|_F}\left[\prod_{j=1}^l |g(j)\rangle\right]\sum_{j=1}^d \|\boldsymbol{h}_j\|\,|j\rangle|\boldsymbol{h}_j\rangle\frac{|0\rangle + |1\rangle}{\sqrt{2}} \tag{12}$$

$$\xrightarrow{(c)} \frac{1}{\|\boldsymbol{H}\|_F}\left[\prod_{j=1}^l |g(j)\rangle\right]\sum_{j=1}^d \|\boldsymbol{h}_j\|\,|j\rangle\left\{\left[|\boldsymbol{h}_j\rangle - 2\sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right]\frac{|0\rangle}{\sqrt{2}} + |\boldsymbol{h}_j\rangle\frac{|1\rangle}{\sqrt{2}}\right\} \tag{13}$$

$$\xrightarrow{(d)} \frac{1}{\|\boldsymbol{H}\|_F}\left[\prod_{j=1}^l |g(j)\rangle\right]\sum_{j=1}^d \|\boldsymbol{h}_j\|\,|j\rangle\left\{\left[|\boldsymbol{h}_j\rangle - \sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right]|0\rangle - \sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle|1\rangle\right\} \tag{14}$$

$$\xrightarrow{(e)} \frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^d \|\boldsymbol{h}_j\|\,|j\rangle\left\{\left[|\boldsymbol{h}_j\rangle - \sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right]|0\rangle - \sum_{m=1}^l |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle|1\rangle\right\}. \tag{15}$$

In step (a), we employ the quantum oracles $U_H V_H$ on ancillas $|0\rangle|0\rangle$. In step (b), an auxiliary register $|0\rangle$ is appended, followed by a Hadamard gate. In step (c), we apply operation $\prod_{m=1}^{l}[R_m \otimes |0\rangle\langle 0| + I \otimes |1\rangle\langle 1|]$. The operation R_m is defined on the register $|g(1)\rangle, \cdots |g(m)\rangle, |h_j\rangle$:

$$R_m = I^{\otimes(m+1)} - 2\left\{\left[\prod_{j=1}^{m}|g(j)\rangle\right]|t_m\rangle\right\}\left\{\left[\prod_{j=1}^{m}\langle g(j)|\right]\langle t_m|\right\}.$$

In step (d), we apply the Hadamard gate on the auxiliary register. In step (e), we trace out registers $\prod_{j=1}^{l}|g(j)\rangle$ to generate state $|\phi_1^{(l)}\rangle$.

The crucial part in Algorithm 3 is to implement the reflection R_m. For the $m+1$ case, there is:

$$|t_{m+1}\rangle = \frac{1}{Z_{m+1}}(|s_{m+1}\rangle - \sum_{i=1}^{m}|t_i\rangle\langle t_i|s_{m+1}\rangle), \tag{16}$$

Note that $\{|t_i\rangle\}$ forms the orthogonal basis: $\langle t_i|t_j\rangle = 0, \forall i \neq j$, and $Z_{m+1} = \langle t_{m+1}|s_{m+1}\rangle = \||s_{m+1}\rangle - \sum_{i=1}^{m}|t_i\rangle\langle t_i|s_{m+1}\rangle\| = \sqrt{1 - \sum_{i=1}^{m}\langle t_i|s_{m+1}\rangle^2}$.

Denote $|t_i\rangle \equiv \sum_{j=1}^{i}x_{ij}|s_j\rangle$. The restriction that $|t_{m+1}\rangle$ is normalized and orthogonal to states $|s_1\rangle, |s_2\rangle, \cdots |s_m\rangle$ could yield Equation (17):

$$\begin{cases} \sum_{i=1}^{m+1}x_{m+1,i}\langle s_j|s_i\rangle = 0, \ \forall j \in [m], \\ \sum_{j=1}^{m+1}\sum_{i=1}^{m+1}x_{m+1,j}x_{m+1,i}\langle s_j|s_i\rangle = 1. \end{cases} \tag{17}$$

Note that $x_{m+1,m+1} = 1/Z_{m+1}$ by Equation (16). Define the m-dimensional vectors x, b, and the $m \times m$ matrix C_m to be:

$$x = \sum_{i=1}^{m}x_{m+1,i}e_i \ , b = \sum_{j=1}^{m}\langle s_j|s_{m+1}\rangle e_j \ , C_m = \{c_{ij} \equiv \langle s_i|s_j\rangle\}_{i,j}^{m,m},$$

where each value $c_{ij} = \langle s_i|s_j\rangle$ could be estimated by Hadamard Test. Thus, we could derive the following linear equations about x from Equation (17):

$$\begin{cases} C_m x = -\frac{1}{Z_{m+1}}b, \\ x^T b = \frac{Z_{m+1}^2 - 1}{Z_{m+1}}. \end{cases} \tag{18}$$

We could obtain the coordinate $\{x_{m+1,i}\}_{i=1}^{m+1}$ by solving Equation (18). There is:

$$\begin{cases} x_{m+1,m+1} = \frac{1}{Z_{m+1}} = \sqrt{\frac{|C_m|}{|C_{m+1}|}}, \\ x_{m+1,i} = -\frac{1}{Z_{m+1}}\frac{|C_m^{(i)}|}{|C_m|}, \ \forall i \in [m], \end{cases} \tag{19}$$

where matrix $C_m^{(i)}$ denotes the matrix generated from C_m by replacing the i-th column with b.

Suppose now we have obtained the linear combination form $|t_{m+1}\rangle = \sum_{i=1}^{m+1}x_{m+1,i}|s_i\rangle$. Thus, we could construct a unitary to generate the state $\left[\prod_{j=1}^{m+1}|g(j)\rangle\right]|t_{m+1}\rangle$ from the state $\left[\prod_{j=1}^{m+1}|g(j)\rangle\right]|0\rangle$, by the linear-combination-of-states method (Shao, 2018).

The idea of linear combination of states was introduced in Shao (2018), which focuses on the following problem: given quantum states $|a\rangle$ and $|b\rangle$, to prepare the state $|c\rangle = \frac{1}{Z_c}(x|a\rangle + y|b\rangle)$. The

method is based on the fact that $R_{2\theta} = (I - 2|b\rangle\langle b|)(I - 2|a\rangle\langle a|)$ can be viewed as the clockwise rotation in the plane spanned by $|a\rangle$ and $|b\rangle$ with angle 2θ, where $\theta = \arccos\langle a|b\rangle$ is the angle between $|a\rangle$ and $|b\rangle$. Thus any clockwise rotation in space $SPAN\{|a\rangle, |b\rangle\}$ with angle ϕ could be written as $R_\phi = R_{2\theta}^{\phi/2\theta}$. For the case $|c\rangle = \frac{1}{Z_c}(x|a\rangle + y|b\rangle)$, there is $|c\rangle = R_\phi|a\rangle$, where $\phi = \arccos\frac{x+y\langle a|b\rangle}{Z_c} = \arccos\frac{x+y\langle a|b\rangle}{\sqrt{x^2+y^2+2xy\langle a|b\rangle}}$. The linear sum of 2 states could be generalized to n case:

Theorem 9. (Shao, 2018) *Assume state $|\phi_i\rangle$ could be prepared by given unitary operation in time T_{in}, for $i \in [n]$. Then there is a unitary which could prepare the state $|\phi\rangle = \sum_{i=1}^{n} \alpha_i|\phi_i\rangle$ in time $\mathcal{O}(T_{in}n^{\log(n/\epsilon)})$ with error ϵ.*

Note that here we actually perform the state $|t'_{m+1}\rangle = |g(1)\rangle|g(2)\rangle\cdots|g(m+1)\rangle|t_{m+1}\rangle$ from states $\{|g(1)\rangle|g(2)\rangle\cdots|g(m+1)\rangle|s_i\rangle\}_{i=1}^{m+1}$, and each state $(\prod_{j=1}^{m+1}|g(j)\rangle)|s_i\rangle$ could be performed by oracle U_H on state $(\prod_{j=1}^{m+1}|g(j)\rangle)|0\rangle$. Denote U_{m+1} as the unitary which generates the state $\left[\prod_{j=1}^{m+1}|g(j)\rangle\right]|t_{m+1}\rangle$ from the state $\left[\prod_{j=1}^{m+1}|g(j)\rangle\right]|0\rangle$. The operation in step (c) of procedure (11): $\prod_{m=1}^{l}[R_m \otimes |0\rangle\langle 0| + I \otimes |1\rangle\langle 1|]$ could be performed by $\prod_{m=1}^{l}\left\{\left[U_{m+1}^\dagger(I - 2|0\rangle\langle 0|)U_{m+1}\right] \otimes |0\rangle\langle 0| + I \otimes |1\rangle\langle 1|\right\}$.

A.7 THE PROOF OF THEOREM 6

Denote P_l as the probability of resulting 0 after the measurement in Step 4 of Algorithm 3. In order to generate the required state, the measurement in Step 4 needs to be performed for $\mathcal{O}(1/P_l)$ times. The Lemma 4 provides a lower bound on P_l.

Lemma 4. *The probability of resulting 0 after the measurement in Step 4 of Algorithm 3 as the lower bound $P_l \geq \frac{(r-l)\epsilon^2}{4\|\mathbf{H}\|_F^2}$.*

Proof. Suppose $\lambda_1^2 \geq \lambda_2^2 \geq \cdots \geq \lambda_r^2$, where λ_i is the eigenvalue of $\mathbf{H}$. Since state $|t_m\rangle$ is the linear sum of $\{|h_j\rangle\}_{j=1}^{d}$, we can assume that $|t_m\rangle$ has the decomposition $|t_m\rangle = \sum_{i=1}^{r} w_{mi}|u_i\rangle$, for all $m = 1, 2, \cdots, l$, where $\sum_{i=1}^{r} w_{mi}w_{ni} = \delta_{mn}$. There is:

$$
\begin{aligned}
P_l &= \frac{1}{\|\mathbf{H}\|_F^2} \sum_{j=1}^{d} \left[\|h_j\|^2\|\,|h_j\rangle - \sum_{m=1}^{l}|t_m\rangle\langle t_m|h_j\rangle\|^2\right] \\
&= \frac{1}{\|\mathbf{H}\|_F^2} \sum_{j=1}^{d} \left[\|h_j\|^2 - \sum_{m=1}^{l}\|h_j\|^2|\langle t_m|h_j\rangle|^2\right] \\
&= 1 - \frac{1}{\|\mathbf{H}\|_F^2} \sum_{j=1}^{d}\sum_{m=1}^{l} \left[\sum_{i=1}^{r} w_{mi}\lambda_i u_i^{(j)}\right]^2 \\
&= 1 - \frac{1}{\|\mathbf{H}\|_F^2} \sum_{j=1}^{d}\sum_{m=1}^{l} \left[\sum_{i=1}^{r} w_{mi}^2\lambda_i^2(u_i^{(j)})^2 + \sum_{i\neq k}^{r} w_{mi}w_{mk}\lambda_i\lambda_k u_i^{(j)}u_k^{(j)}\right] \\
&= 1 - \frac{1}{\|\mathbf{H}\|_F^2} \sum_{m=1}^{l}\sum_{i=1}^{r} w_{mi}^2\lambda_i^2 \\
&= 1 - \frac{1}{\|\mathbf{H}\|_F^2} \sum_{i=1}^{r} c_i\lambda_i^2,
\end{aligned}
$$

where $c_i = \sum_{m=1}^{l} w_{mi}^2$.

Consider the r-dimensional vector $\mathbf{w}_m = \sum_{i=1}^{r} w_{mi}\mathbf{e}_i$. The vector set $\{\mathbf{w}_m\}_{m=1}^{l}$ forms an orthogonal basis in a l-dimensional subspace. Note that we can add $\mathbf{w}_{l+1}, \cdots \mathbf{w}_r$ such that

$\{\boldsymbol{w}_m\}_{m=1}^r$ forms an orthonormal basis in the whole r-dimensional space. Denote matrix $W = (\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_r)$. Since $W^T W = I$. Since W is unitary, there is:

$$\sum_{m=1}^r w_{mi}^2 = 1, \forall i \in [r]. \tag{20}$$

Thus we have the upper bound: $c_i = \sum_{m=1}^l w_{mi}^2 \le \sum_{m=1}^r w_{mi}^2 = 1$. Note that $\sum_{i=1}^r c_i = \sum_{i=1}^r \sum_{m=1}^l w_{mi}^2 = \sum_{m=1}^l \sum_{i=1}^r w_{mi}^2 = l$, so there is:

$$P_l \ge 1 - \frac{1}{\|\boldsymbol{H}\|_F^2} \sum_{i=1}^l \lambda_i^2 = \frac{\sum_{i=l+1}^r \lambda_i^2}{\|\boldsymbol{H}\|_F^2}. \tag{21}$$

Note that for the case $|\lambda_i| \le \epsilon/2$, $\tilde{\lambda}_i = 0$ is a good estimation for the NCF problem, so we could further assume $|\lambda_i| > \epsilon/2$ for the general case and bound the inequality (21) as $P_l \ge \frac{(r-l)\epsilon^2}{4\|\boldsymbol{H}\|_F^2}$.

$\square$

The error of implementing $|\boldsymbol{t}_{m+1}\rangle$ comes from the imperfect implementing of the linear-combination-of-states operation and the error of calculating $\boldsymbol{x}$. The former has been analyzed in Theorem 9. The error of calculating $\boldsymbol{x}$ is more complex.

Define vector $\boldsymbol{y} = -Z_{m+1}\boldsymbol{x}$. Note that all parameters $c_{ij} = \langle \boldsymbol{s}_i | \boldsymbol{s}_j \rangle, i, j \in [m]$ and $b_j = \langle \boldsymbol{s}_{m+1} | \boldsymbol{s}_j \rangle, j \in [m]$ are estimated by Hadamard test, which takes time $\mathcal{O}(T_H \epsilon'^{-2})$ with error bounded in ϵ' (see Appendix A.10 for more information). Thus the calculation on vector $\boldsymbol{y} = C_m^{-1} \boldsymbol{b}$ would have an error. Define matrix $\tilde{C}_m = C_m + \Delta C_m$ and $\tilde{\boldsymbol{b}} = \boldsymbol{b} + \Delta \boldsymbol{b}$ which are estimations on C_m and $\boldsymbol{b}$. Suppose $|\tilde{c}_{ij} - c_{ij}| \le \epsilon_1$ and $|\tilde{b}_i - b_i| \le \epsilon_1$ are error bounds for c_{ij} and b_i, respectively, $\forall i, j \in [m]$. Denote $\tilde{\boldsymbol{y}} = \tilde{C}_m^{-1} \boldsymbol{b}$ as the solution to the approximate linear equation and $\Delta \boldsymbol{y} = \tilde{\boldsymbol{y}} - \boldsymbol{y}$ as the error to $\boldsymbol{y}$. We have:

$$\begin{cases} C_m \boldsymbol{y} = \boldsymbol{b}, \\ (C_m + \Delta C_m)(\boldsymbol{y} + \Delta \boldsymbol{y}) = (\boldsymbol{b} + \Delta \boldsymbol{b}). \end{cases}$$

So there is:

$$\begin{aligned} \|\Delta \boldsymbol{y}\| &= \|(C_m + \Delta C_m)^{-1}(\Delta \boldsymbol{b} - \Delta C_m \cdot C_m^{-1} \boldsymbol{b})\| \\ &\le \|C_m^{-1}\| \cdot \|(I + C_m^{-1}\Delta C_m)^{-1}\| \cdot (\|\Delta \boldsymbol{b}\| + \|\Delta C_m \cdot C_m^{-1}\boldsymbol{b}\|) \\ &\le \|C_m^{-1}\| \cdot \frac{1}{1 - \|C_m^{-1}\Delta C_m\|} \cdot (\|\Delta \boldsymbol{b}\| + \|\Delta C_m\|\|C_m^{-1}\|\|\boldsymbol{b}\|) \\ &\le \frac{\|C_m^{-1}\|}{1 - \|C_m^{-1}\|m\epsilon_1} \cdot (\sqrt{m}\epsilon_1 + m^{3/2}\epsilon_1\|C_m^{-1}\|) \le \frac{2m^{3/2}\|C_m^{-1}\|^2\epsilon_1}{1 - \|C_m^{-1}\|m\epsilon_1}. \end{aligned}$$

The norm $\|C_m^{-1}\|$ here denotes the largest norm of eigenvalues of matrix C_m^{-1}. Thus, for $\epsilon_2 = 3m^{3/2}\|C_m^{-1}\|^2\epsilon_1$, we have bound $\|\Delta \boldsymbol{y}\| \le \epsilon_2$.

For $|\boldsymbol{t}_{m+1}\rangle = \sum_{i=1}^{m+1} x_{m+1,i}|\boldsymbol{s}_i\rangle$ and $|\tilde{\boldsymbol{t}}_{m+1}\rangle = \sum_{i=1}^{m+1} \tilde{x}_{m+1,i}|\boldsymbol{s}_i\rangle$, there is:

$$\begin{aligned} \langle \boldsymbol{t}_{m+1} | \tilde{\boldsymbol{t}}_{m+1} \rangle &= \sum_{i=1}^{m+1} \sum_{j=1}^{m+1} x_{m+1,i} c_{ij} \tilde{x}_{m+1,j} = \frac{1}{Z_{m+1}\tilde{Z}_{m+1}} [\sum_{i=1,j=1}^{m,m} y_i c_{ij} \tilde{y}_j - \sum_{i=1}^m y_i b_i - \sum_{j=1}^m b_j \tilde{y}_j + 1] \\ &= \frac{1}{Z_{m+1}\tilde{Z}_{m+1}} [1 - \sum_{i=1}^m y_i b_i] = \frac{Z_{m+1}}{\tilde{Z}_{m+1}}. \end{aligned}$$

15

Note that the form $|\tilde{\boldsymbol{t}}_{m+1}\rangle = \sum_{i=1}^{m+1} \tilde{x}_{m+1,i}|\boldsymbol{s}_i\rangle$ is not a normalized state:

$$\||\tilde{\boldsymbol{t}}_{m+1}\rangle\|^2 = \sum_{i=1}^{m+1}\sum_{j=1}^{m+1} \tilde{x}_{m+1,i}c_{ij}\tilde{x}_{m+1,j} = \frac{1}{\tilde{Z}_{m+1}^2}[\sum_{i=1,j=1}^{m,m} \tilde{y}_i c_{ij}\tilde{y}_j + 1 - 2\sum_{i=1}^{m}\tilde{y}_i b_i]$$

$$= \frac{1}{\tilde{Z}_{m+1}^2}[\sum_{i=1,j=1}^{m,m} \Delta y_i c_{ij}\Delta y_j + 2\sum_{i=1,j=1}^{m,m} \Delta y_i c_{ij}y_j + \sum_{i=1,j=1}^{m,m} y_i c_{ij}y_j + 1 - 2\sum_{i=1}^{m}\Delta y_i b_i - 2\sum_{i=1}^{m} y_i b_i]$$

$$= \frac{1}{\tilde{Z}_{m+1}^2}[Z_{m+1}^2 + \sum_{i=1,j=1}^{m,m} \Delta y_i c_{ij}\Delta y_j].$$

Thus, the overlap between state $|\boldsymbol{t}_{m+1}\rangle$ and $|\tilde{\boldsymbol{t}}_{m+1}\rangle$ is:

$$\frac{\langle \boldsymbol{t}_{m+1}|\tilde{\boldsymbol{t}}_{m+1}\rangle}{\||\tilde{\boldsymbol{t}}_{m+1}\rangle\|} = \frac{Z_{m+1}}{\sqrt{Z_{m+1}^2 + \Delta\boldsymbol{y}^T \boldsymbol{C}_m \Delta\boldsymbol{y}}}. \tag{22}$$

There is:

$$\||\boldsymbol{t}_{m+1}\rangle - |\tilde{\boldsymbol{t}}_{m+1}\rangle/\||\tilde{\boldsymbol{t}}_{m+1}\rangle\|\| = \sqrt{2 - 2\frac{Z_{m+1}}{\sqrt{Z_{m+1}^2 + \Delta\boldsymbol{y}^T \boldsymbol{C}_m \Delta\boldsymbol{y}}}} \tag{23}$$

$$\leq \sqrt{2 - 2\frac{Z_{m+1}}{\sqrt{Z_{m+1}^2 + \|\Delta\boldsymbol{y}\|^2\|\boldsymbol{C}_m\|}}} \tag{24}$$

$$\leq \sqrt{2 - 2\frac{Z_{m+1}}{Z_{m+1} + \frac{\|\boldsymbol{C}_m\|\|\Delta\boldsymbol{y}\|^2}{2Z_{m+1}}}} \tag{25}$$

$$\leq \frac{\|\boldsymbol{C}_m\|^{1/2}\|\Delta\boldsymbol{y}\|}{Z_{m+1}} \tag{26}$$

$$\leq \frac{m^{1/2}\epsilon_2}{Z_{m+1}} = \frac{3m^2\|\boldsymbol{C}_m^{-1}\|^2\epsilon_1}{Z_{m+1}}. \tag{27}$$

Let $\epsilon_3/2 = \frac{3m^2\|\boldsymbol{C}_m^{-1}\|^2\tilde{\epsilon}_1}{Z_{m+1}}$. It is clear that to obtain the linear combination form $|\tilde{\boldsymbol{t}}_{m+1}\rangle = \sum_{i=1}^{m+1}\tilde{x}_{m+1,i}|\boldsymbol{s}_i\rangle$ takes time $\mathcal{O}(m^3 + m^2 T_H\epsilon_1^{-2})$, where $\||\tilde{\boldsymbol{t}}_{m+1}\rangle/\||\tilde{\boldsymbol{t}}_{m+1}\rangle\| - |\boldsymbol{t}_{m+1}\rangle\| \leq \epsilon_3/2$. By Theorem 9, the implementation of state $|\tilde{\boldsymbol{t}}'_{m+1}\rangle = [\prod_{j=1}^{m+1}|g(j)\rangle]|\tilde{\boldsymbol{t}}_{m+1}\rangle$ takes time $\mathcal{O}(T_H(m+1)^{\log(2(m+1)/\epsilon_3)})$ with error bounds $\epsilon_3/2$. Thus, we can implement state $|\boldsymbol{t}'_{m+1}\rangle = [\prod_{j=1}^{m+1}|g(j)\rangle]|\boldsymbol{t}_{m+1}\rangle$ by unitary in time $\mathcal{O}(m^3 + m^2 T_H\epsilon_1^{-2} + T_H(m+1)^{\log(2(m+1)/\epsilon_3)})$ with error bounds in ϵ_3.

Now we consider the influence of imperfect implementation of state $|\tilde{\boldsymbol{t}}'_m\rangle$ to Algorithm 3. For simplicity, we neglect the term $[\prod_{j=1}^{m}|g(j)\rangle]$ in state $|\boldsymbol{t}'_m\rangle = [\prod_{j=1}^{m}|g(j)\rangle]|\boldsymbol{t}_m\rangle$. Denote $\Pi_l = \prod_{i=1}^{l}(I - 2|\boldsymbol{t}_i\rangle\langle\boldsymbol{t}_i|)$ and $\tilde{\Pi}_l = \prod_{i=1}^{l}(I - 2|\tilde{\boldsymbol{t}}_i\rangle\langle\tilde{\boldsymbol{t}}_i|)$. There is $\|\Pi_l - \tilde{\Pi}_l\| \leq 2l\epsilon_3$.

Note that the state

$$|\phi_1^{(l)}\rangle = \frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^{d}\|\boldsymbol{h}_j\|\||j\rangle\left\{\left[|\boldsymbol{h}_j\rangle - \sum_{m=1}^{l}|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right]|0\rangle - \sum_{m=1}^{l}|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle|1\rangle\right\}$$

in step 3 of Algorithm 3 can be written as:

$$\frac{1}{\|\boldsymbol{H}\|_F}\sum_{j=1}^{d}\|\boldsymbol{h}_j\|\||j\rangle[\frac{\Pi_l + I}{2}|\boldsymbol{h}_j\rangle|0\rangle + \frac{\Pi_l - I}{2}|\boldsymbol{h}_j\rangle|1\rangle].$$

The probability of generating 0 after the measurement on the last register is:

$$P_l = \frac{1}{\|\boldsymbol{H}\|_F^2}\sum_{j=1}^{d}\|\boldsymbol{h}_j\|^2\|\frac{\Pi_l + I}{2}|\boldsymbol{h}_j\rangle\|^2.$$

16

Similarly we define $|\tilde{\phi}_1^{(l)}\rangle = \frac{1}{\|\boldsymbol{H}\|_F} \sum_{j=1}^d \|\boldsymbol{h}_j\| |j\rangle [\frac{\tilde{\Pi}_l + I}{2} |\boldsymbol{h}_j\rangle |0\rangle + \frac{\tilde{\Pi}_l - I}{2} |\boldsymbol{h}_j\rangle |1\rangle]$,

and $\tilde{P}_l = \frac{1}{\|\boldsymbol{H}\|_F} \sum_{j=1}^d \|\boldsymbol{h}_j\|^2 \|\frac{\tilde{\Pi}_l + I}{2} |\boldsymbol{h}_j\rangle\|^2$ for the approximate case.

Since the objective of step 5 is to obtain the index $g(l+1)$ such that the column state $|\boldsymbol{h}_{g(l+1)}\rangle = |\boldsymbol{s}_{l+1}\rangle$ is linearly independent from basis $\{|\boldsymbol{s}_i\rangle\}_{i=1}^l$, we define P_l^{false} as the probability of selecting out the state $|\boldsymbol{s}_{l+1}\rangle \in \{|\boldsymbol{s}_i\rangle\}_{i=1}^l$ in the approximate case. Note that for state $|\boldsymbol{h}_j\rangle \in \{|\boldsymbol{s}_i\rangle\}_{i=1}^l$, $(\Pi_l + I)|\boldsymbol{h}_j\rangle = 0$, so there is:

$$
\begin{aligned}
P_l^{false} &= \frac{1}{\tilde{P}_l} \frac{1}{\|\boldsymbol{H}\|_F^2} \sum_{j:|\boldsymbol{h}_j\rangle \in \{|\boldsymbol{s}_i\rangle\}_{i=1}^l} \|\boldsymbol{h}_j\|^2 \|\frac{\tilde{\Pi}_l + I}{2} |\boldsymbol{h}_j\rangle\|^2 \\
&= \frac{\sum_{j:|\boldsymbol{h}_j\rangle \in \{|\boldsymbol{s}_i\rangle\}_{i=1}^l} \|\boldsymbol{h}_j\|^2 \|\frac{\tilde{\Pi}_l + I}{2} |\boldsymbol{h}_j\rangle\|^2}{\sum_{j=1}^d \|\boldsymbol{h}_j\|^2 \|\frac{\tilde{\Pi}_l + I}{2} |\boldsymbol{h}_j\rangle\|^2} \\
&= \frac{\sum_{j:|\boldsymbol{h}_j\rangle \in \{|\boldsymbol{s}_i\rangle\}_{i=1}^l} \|\boldsymbol{h}_j\|^2 \|\frac{\tilde{\Pi}_l - \Pi_l}{2} |\boldsymbol{h}_j\rangle\|^2}{\sum_{j=1}^d \|\boldsymbol{h}_j\|^2 (1/2 + \langle \boldsymbol{h}_j| \tilde{\Pi}_l - \Pi_l |\boldsymbol{h}_j\rangle/2 + \langle \boldsymbol{h}_j| \Pi_l |\boldsymbol{h}_j\rangle/2)} \\
&\leq \frac{l^2 \epsilon_3^2}{P_l - l\epsilon_3} \leq \frac{l^2 \epsilon_3^2}{\frac{(r-l)\epsilon^2}{4\|\boldsymbol{H}\|_F^2} - l\epsilon_3}.
\end{aligned}
$$

Let $\epsilon_3 = \frac{\epsilon^2}{8(r-1)\|\boldsymbol{H}\|_F^2}$, there is:

$$
\sum_{l=0}^{r-1} P_l^{false} \leq \sum_{l=0}^{r-1} \frac{l^2 \epsilon_3^2}{2(r-1)(r-l)\epsilon_3 - l\epsilon_3} \leq \sum_{l=0}^{r-1} l\epsilon_3 = \frac{r(r-1)}{2} \frac{\epsilon^2}{8(r-1)\|\boldsymbol{H}\|_F^2} = \frac{r\epsilon^2}{16\|\boldsymbol{H}\|_F^2} \leq \frac{1}{4}. \tag{28}
$$

Thus, by choosing $\epsilon_3 = \frac{\epsilon^2}{8(r-1)\|\boldsymbol{H}\|_F^2}$, Algorithm 3 could select out a complete basis $\{|\boldsymbol{s}_i\rangle\}_{i=1}^r$ with probability at least $\frac{3}{4}$.

Note that in Algorithm 3 we need to perform operations $R_i = I - 2|\boldsymbol{t}_i'\rangle\langle \boldsymbol{t}_i'|$ for $i = 1, 2, \cdots, r-1$, which needs the information of parameters $\{c_{ij}\}_{i=1,j=1}^{r-1,r-1}$. In order to guarantee the success probability of Algorithm 3 (Equation (28)), the estimation on each $c_{ij} = \langle \boldsymbol{s}_i | \boldsymbol{s}_j \rangle$ should have error bound $\epsilon_1 = \min_{m \in [r-2]} \frac{Z_{m+1}}{6m^2 \|\boldsymbol{C}_m^{-1}\|^2} \epsilon_3 \geq \frac{\min_{m \in [r-1]} Z_m}{48r^3 \|\boldsymbol{C}_r^{-1}\|^2 \|\boldsymbol{H}\|_F^2} \epsilon^2$. Thus, the estimation on each c_{ij} takes time $\mathcal{O}(T_H r^6 \|\boldsymbol{H}\|_F^4 \epsilon^{-4})$ and the estimation on the parameter group $\{c_{ij}\}_{i=1,j=1}^{r-1,r-1}$ takes time $\mathcal{O}(T_H r^8 \|\boldsymbol{H}\|_F^4 \epsilon^{-4})$. In order to obtain the description $|\boldsymbol{t}_m\rangle = \sum_{i=1}^m x_{m,i} |\boldsymbol{s}_i\rangle$, additional time is required for solving (m-1)-dimensional equations for $m = 2, 3, \cdots r-1$, which results the time complexity $\mathcal{O}(r^4)$ in total. With given parameters $\{x_{m,i}\}_{i=1}^m$, the implementation of operation $R_m = I - 2|\boldsymbol{t}_m'\rangle\langle \boldsymbol{t}_m'|$ takes time $\mathcal{O}(T_H(m+1)^{\log(2(m+1)/\epsilon_3)}) \leq \mathcal{O}(T_H \cdot r^{2\log(4r\|\boldsymbol{H}\|_F/\epsilon)})$.

Denote T_{basis} as the required time to implement Algorithm 3 and T_{R_i} as the required time to implement operation R_i. Since in each iteration of $l \in [r-1]$, Algorithm 3 refers operation $R_1, R_2, \cdots, R_l$ for $1/P_l$ times, there is:

$$
\begin{aligned}
\mathcal{O}(T_{basis}) &= \mathcal{O}(T_H r^8 \|\boldsymbol{H}\|_F^4 \epsilon^{-4}) + \sum_{l=0}^{r-1} \frac{1}{P_l} \sum_{m=1}^l \mathcal{O}(T_{R_i}) \\
&\leq \mathcal{O}(T_H r^8 \|\boldsymbol{H}\|_F^4 \epsilon^{-4}) + \frac{4\|\boldsymbol{H}\|_F^2}{\epsilon^2} \mathcal{O}(T_H r^{2\log(4r\|\boldsymbol{H}\|_F/\epsilon)}) \sum_{l=0}^{r-1} \frac{l}{r-l} \\
&= \mathcal{O}(T_H \|\boldsymbol{H}\|_F^2 \epsilon^{-2}(r^8 \|\boldsymbol{H}\|_F^2 \epsilon^{-2} + r^{1+2\log(4r\|\boldsymbol{H}\|_F/\epsilon)})) \\
&\leq \mathcal{O}(T_H \text{poly}(r) \epsilon^{-2} r^{2\log(4r\|\boldsymbol{H}\|_F/\epsilon)}).
\end{aligned}
$$

A.8 THE PROOF OF LEMMA 2

Proof. Define the index function $g : [r] \rightarrow [d]$ such that $s_i = h_{g(i)}, \forall i \in [r]$. Consider the eigen-decomposition of matrix H:

$$H = \sum_{j=1}^{r} \lambda_j u_j u_j^T. \tag{29}$$

It is natural to generate the decomposition:

$$h_j = \sum_{i=1}^{r} \lambda_i u_i u_i^{(j)}, \tag{30}$$

$$h_{jk} = \sum_{i=1}^{r} \lambda_i u_i^{(j)} u_i^{(k)}. \tag{31}$$

Define the $r \times r$ dimensional matrix $C = (h_{g(1)}^T, h_{g(2)}^T, \cdots, h_{g(r)}^T)^T (h_{g(1)}, h_{g(2)}, \cdots, h_{g(r)})$. There is:

$$\{h_{g(i)}\}_{i=1}^{r} \ is \ linear \ independent \Leftrightarrow det(C) \neq 0. \tag{32}$$

Denote the jk-th element of C as c_{jk}. Since $c_{jk} = h_j^T h_k = \sum_{i=1}^{r} \lambda_i^2 u_i^{(j)} u_i^{(k)}$, there is:

$$det(C) = \begin{vmatrix} \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(1))} u_i^{(g(1))} & \cdots & \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(1))} u_i^{(g(r))} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(r))} u_i^{(g(1))} & \cdots & \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(r))} u_i^{(g(r))} \end{vmatrix} \tag{33}$$

$$= \sum_{i_1=1}^{r} \sum_{i_2=1}^{r} \cdots \sum_{i_r=1}^{r} \begin{vmatrix} \lambda_{i_1}^2 u_{i_1}^{(g(1))} u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r}^2 u_{i_r}^{(g(1))} u_{i_r}^{(g(r))} \\ \vdots & \ddots & \vdots \\ \lambda_{i_1}^2 u_{i_1}^{(g(r))} u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r}^2 u_{i_r}^{(g(r))} u_{i_r}^{(g(r))} \end{vmatrix} \tag{34}$$

$$= \sum_{i_1=1}^{r} \sum_{i_2=1}^{r} \cdots \sum_{i_r=1}^{r} (\prod_{j=1}^{r} \lambda_{i_j}^2)(\prod_{j=1}^{r} u_{i_j}^{(g(j))}) \begin{vmatrix} u_{i_1}^{(g(1))} & \cdots & u_{i_r}^{(g(1))} \\ \vdots & \ddots & \vdots \\ u_{i_1}^{(g(r))} & \cdots & u_{i_r}^{(g(r))} \end{vmatrix}. \tag{35}$$

On the other hand, construct the matrix H' whose jk-th element is $h'_{jk} = h_{g(j),g(k)}$. There is:

$$det(H') = \begin{vmatrix} \sum_{i=1}^{r} \lambda_i u_i^{(g(1))} u_i^{(g(1))} & \cdots & \sum_{i=1}^{r} \lambda_i u_i^{(g(1))} u_i^{(g(r))} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{r} \lambda_i u_i^{(g(r))} u_i^{(g(1))} & \cdots & \sum_{i=1}^{r} \lambda_i u_i^{(g(r))} u_i^{(g(r))} \end{vmatrix} \tag{36}$$

$$= \sum_{i_1=1}^{r} \sum_{i_2=1}^{r} \cdots \sum_{i_r=1}^{r} \begin{vmatrix} \lambda_{i_1} u_{i_1}^{(g(1))} u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r} u_{i_r}^{(g(1))} u_{i_r}^{(g(r))} \\ \vdots & \ddots & \vdots \\ \lambda_{i_1} u_{i_1}^{(g(r))} u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r} u_{i_r}^{(g(r))} u_{i_r}^{(g(r))} \end{vmatrix} \tag{37}$$

$$= \sum_{i_1=1}^{r} \sum_{i_2=1}^{r} \cdots \sum_{i_r=1}^{r} (\prod_{j=1}^{r} \lambda_{i_j})(\prod_{j=1}^{r} u_{i_j}^{(g(j))}) \begin{vmatrix} u_{i_1}^{(g(1))} & \cdots & u_{i_r}^{(g(1))} \\ \vdots & \ddots & \vdots \\ u_{i_1}^{(g(r))} & \cdots & u_{i_r}^{(g(r))} \end{vmatrix}. \tag{38}$$

Note that the determinant in eq(35) and eq(38) is non-zero only if $i_m \neq i_n$ for any different $m, n \in [r]$. Consider the summation of i_j for all $j \in [r]$ over $\{1, 2, \cdots, r\}$, there is:

$$det(C)/\prod_{i=1}^{r} \lambda_i^2 = det(H')/\prod_{i=1}^{r} \lambda_i \tag{39}$$

Thus the problem about whether group $\{h_{g(i)}\}_{i=1}^{r}$ is linearly independent could be solved by calculating the determinant of matrix H'. Since H' is a $r \times r$ dimensional matrix, $det(H')$ could be calculated in $\mathcal{O}(r^3)$ time (Schwarzenberg-Czerny, 1995). We could claim that the group $\{h_{g(i)}\}_{i=1}^{r}$ is linearly independent if $det(H') \neq 0$, or $\{h_{g(i)}\}_{i=1}^{r}$ is linear dependent if $det(H') = 0$. □

A.9 THE PROOF OF THEOREM 7

Proof. For $|\Delta c_{ij}| \leq \epsilon_1$ and $|\Delta b_j| \leq \epsilon_2$, there is:

$$\|\Delta C\| \leq r\epsilon_1 \text{ and } \|\Delta b\| \leq \sqrt{r}\epsilon_2.$$

The matrix norm $\|\cdot\|$ here denotes the largest singular value of the matrix. Note that elements of matrix C and vector b are overlap of quantum states, which are bounded in $[-1, 1]$, so there is:

$$\|C\| \leq r \text{ and } \|b\| \leq \sqrt{r}.$$

We have:

$$
\begin{aligned}
\|\Delta x\| &= \|(C + \Delta C)^{-1}(\Delta b - \Delta C \cdot C^{-1}b)\| \\
&\leq \|C^{-1}\| \cdot \|(I + C^{-1}\Delta C)^{-1}\| \cdot (\|\Delta b\| + \|\Delta C \cdot C^{-1}b\|) \\
&\leq \|C^{-1}\| \cdot \frac{1}{1 - \|C^{-1}\Delta C\|} \cdot (\|\Delta b\| + \|\Delta C\|\|C^{-1}\|\|b\|) \\
&\leq \frac{\|C^{-1}\|}{1 - \|C^{-1}\|r\epsilon_1} \cdot (\sqrt{r}\epsilon_2 + r^{3/2}\epsilon_1\|C^{-1}\|) \leq \frac{\epsilon}{2\sqrt{r}}.
\end{aligned}
$$

Thus, for $u_t = \sum_{j=1}^r x_j s_j$ and $\tilde{u}_t = \sum_{j=1}^r \tilde{x}_j s_j$, there is:

$$\|u_t - \tilde{u}_t\| = \sqrt{\Delta x^T C \Delta x} \leq \|\Delta x\| \cdot \|C\|^{1/2} \leq \frac{\epsilon}{2\sqrt{r}} \cdot \sqrt{r} = \frac{\epsilon}{2}.$$

$\square$

A.10 THE ESTIMATION OF $c_{ij} = \langle s_i | s_j \rangle$

The overlap $c_{ij} = \langle s_i | s_j \rangle = \langle h_{g(i)} | h_{g(j)} \rangle$ can be estimated by the Hadamard Test. We provide the detail in Algorithm 6:

Algorithm 6 c_{ij} estimation

Input: Quantum access to oracle U_H. The index number i and j. The precision parameter ϵ. The probability error bound δ.

Output: An estimation $\tilde{c}_{ij}$ to the value $c_{ij} = \langle s_i | s_j \rangle$, such that $\tilde{c}_{ij} \in [c_{ij} - \epsilon, c_{ij} + \epsilon]$ with probability at least $1 - \delta$.

1: **for** $k = 1$ to $n = \lceil \frac{2}{\epsilon^2} \log(\frac{2}{\delta}) \rceil + 1$ **do**
2:    Create state $[|s_i\rangle|0\rangle + |s_j\rangle|1\rangle]/\sqrt{2}$.
3:    Apply the Hadamard gate on the second register to obtain the state $\frac{|s_i\rangle + |s_j\rangle}{2}|0\rangle + \frac{|s_i\rangle - |s_j\rangle}{2}|1\rangle$.
4:    Measure the second register and record the result.
5: **end for**
6: Count the number of resulting 0 in step 4 as m. Output $2m/n - 1$ as the estimation to c_{ij}.

Proposition 4. *Algorithm 6 present the ϵ-estimation to the overlap $c_{ij} = \langle s_i | s_j \rangle$ with probability at least $1 - \delta$ with running time $\mathcal{O}(T_H \epsilon^{-2} \log(1/\delta))$.*

State $\frac{|s_i\rangle|0\rangle + |s_j\rangle|1\rangle}{\sqrt{2}} = \frac{|h_{g(i)}\rangle|0\rangle + |h_{g(j)}\rangle|1\rangle}{\sqrt{2}}$ in step 2 could be generated by performing the following procedure on state $|g(i)\rangle|g(j)\rangle|0\rangle|0\rangle$:

$$|g(i)\rangle|g(j)\rangle|0\rangle|0\rangle \xrightarrow{H} |g(i)\rangle|g(j)\rangle|0\rangle \frac{|0\rangle + |1\rangle}{\sqrt{2}} \tag{40}$$

$$\xrightarrow{U_H \otimes |0\rangle\langle 0|} |g(i)\rangle|g(j)\rangle \frac{|h_{g(i)}\rangle|0\rangle + |0\rangle|1\rangle}{\sqrt{2}} \tag{41}$$

$$\xrightarrow{U_H \otimes |1\rangle\langle 1|} |g(i)\rangle|g(j)\rangle \frac{|h_{g(i)}\rangle|0\rangle + |h_{g(j)}\rangle|1\rangle}{\sqrt{2}} \tag{42}$$

$$\xrightarrow{\text{trace out the first two registers}} \frac{|h_{g(i)}\rangle|0\rangle + |h_{g(j)}\rangle|1\rangle}{\sqrt{2}}. \tag{43}$$

The Hadamard gate in (40) acts on the 4-th register. The gate $U_H \otimes |0\rangle\langle 0|$ in (41) acts on the 1-st, 3-rd and 4-th registers. The gate $U_H \otimes |1\rangle\langle 1|$ in (42) acts on the 2-nd, 3-rd and 4-th registers.

A.11 THE ESTIMATION OF $b_i = \langle u_t | s_i \rangle$

The estimation to the overlap $b_i = \langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle$ is more complicated. Technics like Algorithm 6 is infeasible, due to the post-selection method for generating target state $|\boldsymbol{u}_t\rangle$. Here we introduce another standard quantum algorithm named as Quantum Swap Test (Buhrman et al., 2001), which could estimate the square overlap between two quantum states $|\phi\rangle$ and $|\psi\rangle$. The circuit of the Quantum Swap Test is illustrated in Figure 1.
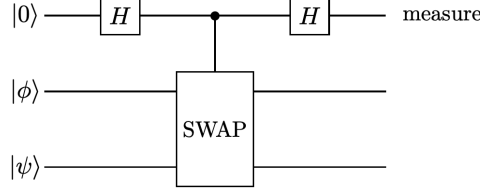

Figure 1: Circuit of the Quantum Swap Test

As shown in Figure 1, Quantum Swap Test performs the operation:

$$|0\rangle|\phi\rangle|\psi\rangle \rightarrow (H \otimes I)(|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U_{SWAP})(H \otimes I)|0\rangle|\phi\rangle|\psi\rangle. \tag{44}$$

The final state could be written as:

$$\frac{1}{2}|0\rangle(|\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle) + \frac{1}{2}|1\rangle(|\phi\rangle|\psi\rangle - |\psi\rangle|\phi\rangle). \tag{45}$$

The U_{SWAP} gate could be implemented in time $\mathcal{O}(\mathrm{polylog}(d))$, which performs the swap transformation $|\phi\rangle|\psi\rangle \rightarrow |\psi\rangle|\phi\rangle$ for d-dimensional state $|\phi\rangle$ and $|\psi\rangle$. The measurement on the first qubit produces 0 with probability $P_0 = \frac{1}{2}(1 + |\langle\phi|\psi\rangle|^2)$. Thus, by replacing step 2-3 in Algortihm 6 with the Quantum Swap Test operation, we could build an algorithm to estimate the square of the state overlap.

Proposition 5. *There exists a quantum algorithm which could present ϵ-estimation to value $b_i^2 = |\langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle|^2$ with probability at least $1 - \delta$ in running time $\mathcal{O}(T_{Input}\mathrm{polylog}(d)\epsilon^{-2}\log(1/\delta))$, where T_{Input} is the time complexity to generate states $|\boldsymbol{u}_t\rangle$ and $|\boldsymbol{s}_i\rangle$.*

In order to estimate values $b_i = \langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle$ for $i \in [r]$, we need to discriminate the positive and negative of b_i. Note that for state $|\boldsymbol{u}_t\rangle$, the state $|-\boldsymbol{u}_t\rangle$ is also a target state which shares the same eigenvalue. So both states $|\boldsymbol{u}_t\rangle$ and $|-\boldsymbol{u}_t\rangle$ are legal outputs and indistinguishable for our algorithm in Section 3. Thus we analysis the value $b_i = sgn(u_t^{(k)})\langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle$ as the overlap between states $|\boldsymbol{u}_t\rangle$ and $|\boldsymbol{s}_i\rangle$, where $u_t^{(k)}$ is the k-th component of vector $\boldsymbol{u}_t$. Generally k could be any index such that the corresponding component is non-zero.

Note that $b_i = sgn(u_t^{(k)})\langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle = -sgn(\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle)\langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle$. Define two states $|\psi_+\rangle = \frac{1}{Z_+}(|\boldsymbol{h}_k\rangle + |\boldsymbol{h}_{g(i)}\rangle)$ and $|\psi_-\rangle = \frac{1}{Z_-}(|\boldsymbol{h}_k\rangle - |\boldsymbol{h}_{g(i)}\rangle)$, where $Z_\pm$ are normalized constants such that $Z_\pm^2 = 2 \pm 2\langle \boldsymbol{h}_k | \boldsymbol{h}_{g(i)} \rangle$. Then there is:

$$|\langle \boldsymbol{u}_t | \psi_+ \rangle|^2 = \frac{1}{Z_+^2}\Big[\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle^2 + \langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle^2 + 2\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle\langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle\Big], \tag{46}$$

$$|\langle \boldsymbol{u}_t | \psi_- \rangle|^2 = \frac{1}{Z_-^2}\Big[\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle^2 + \langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle^2 - 2\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle\langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle\Big]. \tag{47}$$

States $|\psi_+\rangle$ and $|\psi_-\rangle$ could be generated by step 2-4 in Algorithm 6. The overlap $\langle \boldsymbol{h}_k | \boldsymbol{h}_{g(i)} \rangle$ could be estimated by Algorithm 6. The square overlap $|\langle \boldsymbol{u}_t | \psi_+ \rangle|^2$ and $|\langle \boldsymbol{u}_t | \psi_- \rangle|^2$ could be estimated by Quantum Swap Test. Thus for $|\langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle| > \epsilon'$, one could discriminate the positive and negative of $\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle\langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle$ by calculate the value $Z_+^2|\langle \boldsymbol{u}_t | \psi_+ \rangle|^2 - Z_-^2|\langle \boldsymbol{u}_t | \psi_- \rangle|^2$. The estimation on the square overlap $|\langle \boldsymbol{u}_t | \psi_+ \rangle|^2$ and $|\langle \boldsymbol{u}_t | \psi_- \rangle|^2$ need to have the precision $\epsilon'|\langle \boldsymbol{u}_t | \boldsymbol{h}_k \rangle|/2$, which takes time $\mathcal{O}(T_{Input}\epsilon'^{-2})$. For $|\langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle| \leq \epsilon'$, 0 is an ϵ' estimation to $\langle \boldsymbol{u}_t | \boldsymbol{h}_{g(i)} \rangle$. Since an ϵ'-estimation to $|b_i|$ could be achieved by an ϵ'^2-estimation to b_i^2 which takes time $\mathcal{O}(T_{Input}\epsilon'^{-4})$, we could derive the time complexity of estimating b_i in Theorem 10.

Theorem 10. *There exists a quantum algorithm which could present ϵ-estimation to value $b_i = sgn(u_t^{(k)})\langle \boldsymbol{u}_t | \boldsymbol{s}_i \rangle$ with probability at least $1 - \delta$ in running time $\mathcal{O}(T_{Input}\mathrm{polylog}(d)\epsilon^{-4}\log(1/\delta))$, where T_{Input} is the time complexity to generate states $|\boldsymbol{u}_t\rangle$ and $|\boldsymbol{s}_i\rangle$.*