Quantum Algorithm for Finding the Negative Curvature Direction

KAINING ZHANG

Supervisor: Prof. Dacheng Tao

A thesis submitted in fulfilment of the requirements for the degree of Master of Philosophy

Faculty of Engineering and Information Technologies
University of Sydney

2019

Statement of Originality

This is to certify that to the best of my knowledge, the work contained in this thesis has not been submitted for any degree or other purposes. I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Kaining Zhang

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Professor Dacheng Tao. I learned a lot about doing research when working with him.

Secondly, I would like to thank Professor Min-Hsiu Hsieh, Dr. Liu Liu and Mr. Yuxuan Du. They all gave me great guidance on the way of studying optimization and quantum computing.

I also thank my friends in the research group of UBTECH Sydney Artificial Intelligence Centre for helpful discussion.

Finally, I thank my parents for their encouragement and support in my whole life.

Abstract

Non-convex optimization is an essential problem in the field of machine learning. 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 are applied to find the effective direction to escape the saddle point. Specifically, finding the Negative Curvature is considered as the subroutine to analyze the characteristic of the saddle point. However, the calculation of the Negative Curvature is expensive, which prevents the practical usage of second-order algorithms.

In this thesis, 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 $\tilde{O}(\mathrm{polylog}(d)\epsilon^{-1})$, 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 $O(d\epsilon^{-1/2})$. Moreover, we propose an efficient quantum 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.

CONTENTS

CHAPTER 1

Introduction

Algorithms for finding the minimum 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 minimum. 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 [1, 2, 3] based on the second-order Newton-type and the first-order methodology have been analyzed from such a perspective. However, those methods normally deal with the situations in which the iterations may be trapped in the saddle points, such as deep neural networks [4, 5]. The existence of many saddle points is the main bottleneck.

Many algorithms have been proposed to escape the saddle points in general non-convex optimizations. These algorithms can be divided into the following two categories: first-order gradient-based algorithms and second-order Hessian-based algorithms. Generally, second-order algorithms have better iteration complexity compared with first-order algorithms (cf. [6]). However, each iteration in the second-order method involves the computation of the **negative curvature direction**, namely, the eigenvector of a Hessian matrix $H = \nabla^2 f(\boldsymbol{x})$ with negative eigenvalue. This computation could take time $O(d^2)$ when the Hessian matrix is given, or $O(d/\sqrt{\epsilon})$ when the 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 [7], support-vector machine [8], singular value decomposition [9]. These works encourage us to develop an efficient quantum algorithm for finding 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, along with parameters $\alpha \in (0, L)$ and $\epsilon \in (0, \alpha)$, we aim to build a quantum algorithm that could efficiently provide the unit vector $\boldsymbol{u}$ with the condition:*

$$\boldsymbol{u}^T H \boldsymbol{u} \leq -\alpha + \epsilon, \tag{1.1}$$

or make the non-vector statement that all unit vector $\boldsymbol{u}$ satisfying the following condition with high probability:

$$\boldsymbol{u}^T H \boldsymbol{u} \geq -\alpha. \tag{1.2}$$

We present definitions of smoothness and γ-separation here.

DEFINITION 1.1. (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 1.2. (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 matrix H in this article has two properties:

(1) $H \in \mathbb{R}^{d \times d}$ is a r-rank Hessian matrix which is derived from the d-dimensional optimization problem $\min_{\boldsymbol{x} \in \mathbb{R}^d} f(\boldsymbol{x})$ in which the objective function f has L-Lipschitz continuous gradient;

(2) The absolute value of H's non-zero eigenvalue is ϵ-separated.

The first property is directly derived from the assumption of the previous classical non-convex optimization method [10], and the low-rank Hessian case has been observed in neural networks [11]. The second property is assumed such that we could distinguish different eigenvalues by their absolute value. We further assume that the Hessian matrix H has the eigen-decomposition $H = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^T$, for the convenience of the following discussion.

1.1 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 rows of Hessian H with rank r.

Negative Curvature Finding: We develop an efficient quantum algorithm to produce the target state $|\boldsymbol{u}_t\rangle$ (for case (1.1)) or make the non-vector statement (for case (1.2)). We provide Proposition 1.1 as the main result of this part, which guarantees the time complexity of our NCF algorithm:

PROPOSITION 1.1. *There exists a quantum algorithm that could solve the Negative Curvature Finding problem in time $\tilde{O}(\mathrm{poly}(r, \log d)\epsilon^{-1})$, by providing the target state $|\boldsymbol{u}_t\rangle$ (for case (1.1)), or making the non-vector statement (for case (1.2)).*

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 $O(d\epsilon^{-2})$ [12], 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 row 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.

Our state read-out algorithm contains two subroutines named as the complete basis selection and the coordinate estimation algorithm, with main results summarized (informally) as following theorems:

THEOREM 1.1. *There exists a quantum algorithm that finds an index set $\{g(i)\}_{i=1}^{r}$ in time $\tilde{O}(\mathrm{poly}(\lambda_{\min}^{-1}(H), r))$, where r is the rank of H and $\{g(i)\}_{i=1}^{r}$ forms a complete basis $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$ with probability at least 3/4.*

THEOREM 1.2. *The classical description $\boldsymbol{u}_t = \sum_{i=1}^{r} x_i \boldsymbol{s}_i / \|\boldsymbol{s}_i\|$ for the target state could be presented in time $\tilde{O}(\mathrm{poly}(r, \log d)\epsilon^{-3})$ with error bounds in ϵ, when the basis set $\{\boldsymbol{s}_j\}_{j=1}^{r}$ is given.*

The rest of this thesis is organized as follows. Some literature about non-convex optimization and quantum computing are introduced in Chapter 2. In Chapter 3, we develop a quantum algorithm to solve the **NCF problem**. In Chapter 4, we develop a quantum algorithm aiming to read out the target state. We summarize our results and contributions in Chapter 5.

CHAPTER 2

Literature Review

2.1 Non-convex Optimization

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 [10, 3] are exploited to find the effective direction to escape the saddle point. Specifically, finding the Negative Curvature is considered to be 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 minimum, local minimum, 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. Recent works [13, 6, 14] showed that by adding noise at each step, gradient descent could escape all saddle points in a polynomial number of iterations. Ref.[15] 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 $O(d)$ operations, and their iteration complexity is higher than second-order algorithms [6].

Second-order algorithms: Traditionally, second-order Newton-based methods can converge to local minima, which uses 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 [2] and cubic regularization [1] 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 $O(d^2)$. 2) Hessian-free: The Hessian-free methods use the Lanczos method to calculate the negative curvature direction and use the gradient to approximate the Hessian-vector product [3, 10, 16]. The Hessian-free method involves $O(d\epsilon^{-1/2})$ 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.

2.2 Quantum Computing

2.2.1 Basic Knowledge

In this section, we present some basic quantum knowledge. Here we introduce the Dirac notation, which is often used in quantum computing. The form $|\boldsymbol{x}\rangle$ denotes the state, which corresponds to the vector $\boldsymbol{x}$, and the form $\langle \boldsymbol{y}|$ denotes 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 $\boldsymbol{x} \in \mathbb{C}^d$, 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 $x_j^2/\|\boldsymbol{x}\|^2$.

2.2.2 Hamiltonian Simulation

Hamiltonian simulation is a task in quantum information field [17] aiming to simulate the evolution of a quantum system. The time evolution with given Hamiltonian H can be described as $|\phi(t)\rangle = e^{-iHt}|\phi(0)\rangle$ in the Schrödinger picture with initial state $|\phi(0)\rangle$, or $\sigma(t) = e^{-iHt}\sigma(0)e^{iHt}$ in the Heisenberg picture with initial state (in the form of density matrix) $\sigma(0)$. Given a Hamiltonian ($2^n \times 2^n$ dimensional hermitian matrix on n qubits system), an evolution time t, and an error bound ϵ, the goal is to implement the unitary operation U, such that:

$$\|U - e^{-iHt}\| \leq \epsilon,$$

where $\|\cdot\|$ is the spectral norm of a matrix.

There are many proposed frameworks for the Hamiltonian simulation task, such as the technique based on Trotter-Suzuki decompositions [18] that suits the sparse Hamiltonian case. Some other Hamiltonian simulation algorithms are developed by using techniques like the Taylor series expansion [19], quantum walk [20], or quantum signal processing [21].

Here we explain a Hamiltonian simulation framework developed in the quantum principal component analysis (PCA) [7], which is employed in our quantum algorithms developed in this thesis. The Hamiltonian simulation framework in quantum PCA aims to simulate the evolution $e^{-i\rho t}$ by using multiple copies of ρ, where ρ is the density matrix of some unknown quantum state. Specifically, consider the evolution $e^{-\rho t}\sigma e^{i\rho t}$ on state σ. One could simply prepare the initial state $\rho \otimes \sigma$ and perform operation:

$$\begin{aligned}
\mathrm{Tr}_1 e^{-iS\Delta t}(\rho \otimes \sigma)e^{iS\Delta t} &= (\cos^2 \Delta t)\sigma + (\sin^2 \Delta t)\rho - i \sin \Delta t[\rho, \sigma] \\
&= \sigma - i\Delta t[\rho, \sigma] + O(\Delta t^2) \\
&= e^{-i\rho\Delta t}\sigma e^{i\rho\Delta t} + O(\Delta t^2),
\end{aligned}$$

where Tr_1 denotes the partial trace over the first variable and $S : |i\rangle \otimes |j\rangle \rightarrow |j\rangle \otimes |i\rangle$ is the SWAP operator. Remark that S is a sparse matrix so that $e^{-S\Delta t}$ could be implemented efficiently [22]. Repeat the above operation for n times could yield $e^{-i\rho n\Delta t}\sigma e^{i\rho n\Delta t} + O(n\Delta t^2)$. Let $n\Delta t = t$, then operation $e^{-i\rho t}$ could be implemented with error $O(t^2/n)$, so $n = t^2/\epsilon$ copies of state ρ is required to achieve a ϵ error bounded Hamiltonian simulation with time t.

Note that in many cases, the operator e^{-iHt} is used in the controlled form, such as the phase estimation algorithm [23]. The controlled $e^{-i\rho t}$ operation here could be similarly constructed by using a controlled version of SWAP operator.

2.2.3 Quantum Singular Value Estimation (SVE) Algorithm

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 $H \in \mathbb{R}^{d\times d}$, we assume that H is stored in a data structure named as the quantum random access memory (QRAM), such that the following quantum oracles could be implemented:

$$U_H : |i\rangle|0\rangle \rightarrow |i\rangle|\boldsymbol{h}_i\rangle = \frac{1}{\|\boldsymbol{h}_i\|}\sum_{j=1}^{d} h_{ij}|i\rangle|j\rangle, \forall i \in [d], \tag{2.1}$$

$$V_H : |0\rangle|j\rangle \rightarrow |\tilde{\boldsymbol{h}}\rangle|j\rangle = \frac{1}{\|H\|_F}\sum_{i=1}^{d} \|\boldsymbol{h}_i\||i\rangle|j\rangle, \forall j \in [d], \tag{2.2}$$

where $\boldsymbol{h}_j \in \mathbb{R}^{d\times 1}$ denotes the transpose of j-th row vector of matrix H, $\|H\|_F = \sqrt{\sum_{i,j=1}^{d} h_{ij}^2}$ is the Frobenius norm of matrix H, and $\tilde{\boldsymbol{h}}$ stands for the d-dimensional vector whose i-th component is $\|\boldsymbol{h}_i\|/\|H\|_F$. 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. The detailed description about this data structure can be referred to [24]. We denote T_H as the time complexity of these oracles.

Suppose the matrix $H \in \mathbb{R}^{d\times d}$, which has the eigenvalue decomposition $H = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j$, is stored in the data structure mentioned before. Previous work by I. Kerenidis and A. Prakash [24] provides an efficient 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 2.1. *[24] Suppose quantum accesses to oracles (2.1) and (2.2) exist. There is a quantum algorithm which could perform the mapping* $\sum_j \beta_j |\boldsymbol{u}_j\rangle \to \sum_j \beta_j |\boldsymbol{u}_j\rangle |\hat{\lambda}_j|\rangle$ *with time complexity* $\mathcal{O}(T_H \mathrm{polylog}(d)\epsilon^{-1})$*, where* $\hat{\lambda}_j \in [\lambda_j - \epsilon\|H\|_F, \lambda_j + \epsilon\|H\|_F]$ *with probability at least* $1 - 1/\mathrm{poly}(d)$.

2.2.4 Quantum SWAP Test

Here we introduce a quantum algorithm named the Quantum SWAP test [25]. The goal of quantum SWAP test is to estimate the projection $|\langle\phi|\psi\rangle|^2$ between two states $|\phi\rangle$ and $|\psi\rangle$. We present a quantum circuit for the SWAP test in Figure 2.1.

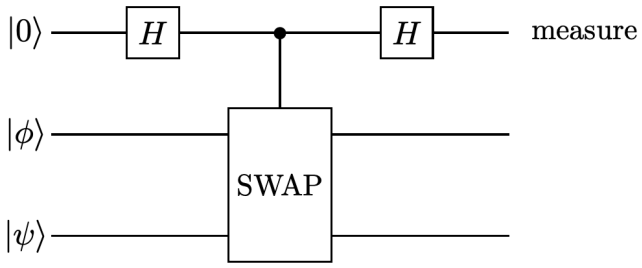

FIGURE 2.1: Circuit of the Quantum SWAP Test

As shown in Figure 2.1, Quantum SWAP test performs the operation:

$$|0\rangle|\phi\rangle|\psi\rangle \to (H \otimes I)(|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes S)(H \otimes I)|0\rangle|\phi\rangle|\psi\rangle. \tag{2.3}$$

The final state could be written as:

$$\frac{1}{2}|0\rangle(|\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle) + \frac{1}{2}|1\rangle(|\phi\rangle|\psi\rangle - |\psi\rangle|\phi\rangle). \tag{2.4}$$

The S gate performs the SWAP operation $|\phi\rangle|\psi\rangle \rightarrow |\psi\rangle|\phi\rangle$ for state $|\phi\rangle$ and $|\psi\rangle$. The measurement on the first qubit of the final state produces 0 with probability:

$$P_0 = \frac{1}{4}\||\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle\|^2 = \frac{1}{2}(1 + |\langle\phi|\psi\rangle|^2).$$

By the Hoeffding's inequality (Lemma 6.1 in Appendix), one could perform the measurement for $\mathcal{O}(\epsilon'^{-2}\log(1/\delta))$ times to obtain an estimation on the square of the overlap $|\langle\phi|\psi\rangle|^2$ with error bounded in ϵ' with probability at least $1-\delta$.

CHAPTER 3

QNCF Algorithm

Our main contribution in this chapter is the quantum Negative Curvature Finding (NCF) algorithm presented as Algorithm 1. The quantum NCF algorithm solves the NCF problem by providing the target state $|\boldsymbol{u}_t\rangle$ for case (1.1) or making the non-vector statement for case (1.2). The target state $|\boldsymbol{u}_t\rangle$ here corresponds to the eigenvector $\boldsymbol{u}_t$, which satisfies the condition $\boldsymbol{u}_t^T H \boldsymbol{u}_t \leq -\alpha + \epsilon/2$. We present a tighter restrict on the target state $|\boldsymbol{u}_t\rangle$ than the condition in case (1.1) to keep a $\epsilon/2$ redundancy for the classical read-out of the quantum state. Quantum NCF Algorithm uses the Proper Eigenvalue Labelling (Algorithm 3) and the Target State Generating (Algorithm 4) as subroutines proposed in Section 3.1 and Section 3.3, respectively. We summarize our conclusion on the complexity

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: A target state $|\boldsymbol{u}_t\rangle$ such that the corresponding vector satisfies $\boldsymbol{u}_t^T H \boldsymbol{u}_t \leq -\alpha + \epsilon/2$; or a statement that with high probability there is no unit vector $\boldsymbol{u}$ satisfies the condition $\boldsymbol{u}^T H \boldsymbol{u} < -\alpha$.

1: Label the **proper** (less than $-\alpha + \epsilon/2$) eigenvalue of H (Proper Eigenvalue Labelling).
2: **if** the least eigenvalue of H is labeled to be less than $-\alpha + \epsilon/2$, **then**
3: generate the target state (Target State Generating) and output the state;
4: **else**,
5: claim that there is no unit vector $\boldsymbol{u}$ satisfies the condition $\boldsymbol{u}^T H \boldsymbol{u} < -\alpha$.
6: **end if**

of Algorithm 1 in Theorem 3.1.

THEOREM 3.1. *Algorithm 1 takes time $O(T_H r^{5/2} L^5 \alpha^{-4} \mathrm{polylog}(d)\epsilon^{-1})$ to solve the NCF problem by providing the target state $|\boldsymbol{u}_t\rangle$ or making the statement that there is no unit vector satisfies the condition $\boldsymbol{u}^T H \boldsymbol{u} < -\alpha$.*

PROOF. The complexity of Algorithm 1 could be directly obtained by the complexity of Algorithm 3 and Algorithm 4, whose complexity analyses are presented in Theorem 3.2 and Theorem 3.3, respectively. □

The core technical component of our quantum NCF algorithm is the quantum SVE algorithm. However, there are three major challenges that we have to overcome. Firstly, the positive-negative eigenvalue problem. In the NCF problem, we are interested in eigenvectors with negative eigenvalues, but the quantum SVE algorithm only gives estimations on singular values $|\lambda_j|$. To overcome this issue, we develop Algorithm 2 to label negative eigenvalues.

Secondly, since the quantum SVE algorithm presents ϵ-estimations on singular values with complexity $O(T_H \|H\|_F \mathrm{polylog}(d)\epsilon^{-1})$ (Theorem 2.1), we need to provide a tight upper bound for the Frobenius norm $\|H\|_F$, which is shown in Lemma 3.1 (proved in Appendix):

LEMMA 3.1. *Suppose $H \in \mathbb{R}^{d\times d}$ is the Hessian matrix derived from the function $f : \mathbb{R}^d \to \mathbb{R}$ with L-Lipschitz continuous gradient. Thus the Frobenius norm of H is upper bounded by $\sqrt{r}L$, where r is the rank of H.*

Finally, the input-state problem. For a general input state $\sum_j \beta_j |\boldsymbol{u}_j\rangle$, the output of the quantum SVE algorithm has the form $\sum_j \beta_j |\boldsymbol{u}_j\rangle||\hat{\lambda}_j|\rangle$. We could generate different eigenstates $|\boldsymbol{u}_j\rangle$ with probability $|\beta_j|^2$ by the measurement on the eigenvalue register. Thus to guarantee a small complexity, we need to prepare a specific input such that the projection between the input and the target state is relatively large.

3.1 The Sign of Eigenvalue

In this section, we propose a quantum algorithm aiming to label the eigenvalue that is less than $-\alpha + \epsilon/2$. This algorithm helps verify the existence of the solution to the NCF problem and generating the target state. Since the quantum SVE algorithm only provides estimations on singular values $|\lambda_j|$, we need to develop Algorithm 2 first for obtaining the sign of an eigenvalue.

Algorithm 2 sign(λ) Algorithm

Input: Quantum access to oracles U_H and V_H. An eigenstate $|\boldsymbol{u}\rangle$ whose corresponding eigenvalue is λ.

Output: A random variable which has different values 0 and 1 with probability $P(0) = \frac{1+\lambda/\|H\|_F}{2}$ and $P(1) = \frac{1-\lambda/\|H\|_F}{2}$.

1: Create state $|\boldsymbol{u}\rangle|0\rangle|0\rangle$. The second register has the same qubit length with state $|\boldsymbol{u}\rangle$ and the third register has one qubit length.
2: Apply the Hadamard gate to obtain the state $\frac{1}{\sqrt{2}}(|\boldsymbol{u}\rangle|0\rangle|0\rangle + |\boldsymbol{u}\rangle|0\rangle|1\rangle)$.
3: Apply the controlled SWAP operation to obtain the state $\frac{1}{\sqrt{2}}(|\boldsymbol{u}\rangle|0\rangle|0\rangle + |0\rangle|\boldsymbol{u}\rangle|1\rangle)$.
4: Apply operation $U_H \otimes |0\rangle\langle 0| + V_H \otimes |1\rangle\langle 1|$ to obtain state $\frac{1}{\sqrt{2}}(|P\boldsymbol{u}\rangle|0\rangle + |Q\boldsymbol{u}\rangle|1\rangle)$.
5: Apply the Hadamard gate to obtain the state $\frac{|P\boldsymbol{u}\rangle+|Q\boldsymbol{u}\rangle}{2}|0\rangle + \frac{|P\boldsymbol{u}\rangle-|Q\boldsymbol{u}\rangle}{2}|1\rangle$.
6: Measure the single qubit register and output the result.

In Algorithm 2, $P \in \mathbb{R}^{d^2\times d}$ is the matrix with column vectors $\boldsymbol{p}_i = \boldsymbol{e}_i \otimes \frac{\boldsymbol{h}_i}{\|\boldsymbol{h}_i\|}$ for $i \in [d]$, and $Q \in \mathbb{R}^{d^2\times d}$ is the matrix with column vectors $\boldsymbol{q}_j = \frac{\tilde{\boldsymbol{h}}}{\|H\|_F} \otimes \boldsymbol{e}_j$ for $j \in [d]$. $\boldsymbol{h}_j \in \mathbb{R}^{d\times 1}$ is the transpose of j-th row vector of matrix H, and $\tilde{\boldsymbol{h}}$ is the d-dimensional vector whose i-th component is $\|\boldsymbol{h}_i\|/\|H\|_F$. By direct calculation, there is:

$$P^TQ = \frac{H}{\|H\|_F}, \;\; P^TP = Q^TQ = I.$$

Mappings $|\boldsymbol{u}\rangle|0\rangle \to |P\boldsymbol{u}\rangle$ and $|0\rangle|\boldsymbol{u}\rangle \to |Q\boldsymbol{u}\rangle$ can be performed by quantum oracles U_H and V_H, respectively.

LEMMA 3.2. *One could obtain the correct $sign(\lambda)$ with probability $1-\delta$ by using $n = 2\frac{\|H\|_F^2}{\lambda^2}\log\frac{1}{\delta}$ copies of eigenstate $|\boldsymbol{u}\rangle$ for Algorithm 2.*

PROOF. The measurement in step 6 of Algorithm 2 outputs 1 with probability:

$$P(1) = \|\frac{|P\boldsymbol{u}\rangle - |Q\boldsymbol{u}\rangle}{2}\|^2 = \frac{1}{4}(\langle P\boldsymbol{u}| - \langle Q\boldsymbol{u}|)(|P\boldsymbol{u}\rangle - |Q\boldsymbol{u}\rangle).$$

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

Suppose that we need $n = 2x + 1$ times of measurement to give a $1 - \delta$ correct statement about whether $\lambda > 0$ or $\lambda < 0$. The problem can be viewed as the biased coin task. Define random variables X_i such that $P(X_i = 1) = p$ and $P(X_i = 0) = 1 - p$, define $S_n = \sum_{i=1}^n X_i/n$. Then by the Hoeffding's inequality (Lemma 6.1), there is:

$$P(S_n/n - p \leq -\epsilon) \leq e^{-2n\epsilon^2}\ ,\ P(S_n/n - p \geq \epsilon) \leq e^{-2n\epsilon^2}.$$

Back to the problem, the probability $p = \frac{1-\lambda/\|H\|_F}{2}$. Suppose $\lambda < 0$, and we consider:

$$\text{sign}(\lambda) = -\text{sign}(S_n - x).$$

The probability of obtaining the wrong sign is:

$$\begin{aligned} P(S_n \leq x) &= P(S_n/n - p \leq x/n - p) \\ &\leq \exp\left(-2n(x/n - p)^2\right) \\ &= \exp\left(-2n\left(\frac{1}{2n} - \frac{\lambda}{2\|H\|_F}\right)^2\right) \\ &\leq \exp\left(-\frac{n\lambda^2}{2\|H\|_F^2}\right). \end{aligned}$$

Similarly for $\lambda > 0$, there is $P(S_n \geq x) \leq \exp\left(-\frac{n\lambda^2}{2\|H\|_F^2}\right)$.

Let $\exp\left(-\frac{n\lambda^2}{2\|H\|_F^2}\right) \leq \delta$, we have $n \geq 2\frac{\|H\|_F^2}{\lambda^2}\log\frac{1}{\delta}$. □

3.2 Proper Eigenvalue Labelling

Now we build the Algorithm 3 to label the **proper** eigenvalue, which would also benefit the target state generation task in the following section. The **proper** here means the eigenvalue is less than $-\alpha + \epsilon/2$. We view the corresponding eigenvector as our target vector.

Algorithm 3 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 singular value $|\lambda_j|$ such that $\lambda_j \leq -\alpha + \epsilon/2$ with probability $1 - \delta$, or a non-vector statement that there is no unit vector $\boldsymbol{u}$ satisfies $\boldsymbol{u}^T H \boldsymbol{u} < -\alpha$.

1: **for** $k = 1$ to $32\frac{\|H\|_F^4}{\alpha^4} \log \frac{1}{\delta}$ **do**
2: Create the state $|\psi_{\text{eigen}}\rangle$ in Equation 3.1.
3: Measure the singular value register and mark the result.
4: Use the rest state in the first register as the input for Algorithm 2.
5: **end for**
6: Count the result in step 3 and step 4 for the sequence $\{(|\tilde{\lambda}_j|, n_j, m_j)\}_{j=1}^r$, where n_j is the number of resulting $|\tilde{\lambda}_j|$ in step 3, and m_j is the number of resulting 1 in step 4 for different j.
7: **if** $\frac{m_j}{n_j} < \frac{1}{2}$ for all $j \in [r]$, **then** make the non-vector statement;
8: **else**, choose the largest $|\tilde{\lambda}_j|$ that satisfies the condition $\frac{m_j}{n_j} > \frac{1}{2}$.
9: **if** $|\tilde{\lambda}_j| < \alpha - \epsilon/4$, **then** make the non-vector statement;
10: **else**, label eigenvalue λ_j as the **proper** eigenvalue.
11: **end if**
12: **end if**

The mean idea of Algorithm 3 is to use the input state:

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

for the quantum SVE model and obtain the state:

$$|\psi_{\text{eigen}}\rangle = \frac{1}{\|H\|_F} \sum_{j=1}^{r} \lambda_j |\boldsymbol{u}_j\rangle |\boldsymbol{u}_j\rangle |\tilde{\lambda}_j|\rangle, \tag{3.1}$$

where $|\tilde{\lambda}_j| \in |\lambda_j| \pm \epsilon/4$ with probability $1 - 1/\text{poly}(d)$. The measurement on the singular value register would let this entangled state collapse to different states

$|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle$ for $j \in [r]$. Since $|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle$ is a pure state, we could obtain the state $|\boldsymbol{u}_j\rangle$ by neglecting the state in any other register. Using state $|\boldsymbol{u}_j\rangle$ as the input for Algorithm 2 could provide discrimination on the value of $\text{sign}(\lambda_j)$. 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 eigenvalues are greater than $-\alpha$. We analyze the complexity of Algorithm 3 in Theorem 3.2.

THEOREM 3.2. *Algorithm 3 labels the **proper** eigenvalue of H, or claim with high probability that there is no unit vector $\boldsymbol{u}$ satisfies $\boldsymbol{u}^T H \boldsymbol{u} < -\alpha$, with time complexity $O(T_H \|H\|_F^5 \alpha^{-4} \text{polylog}(d)\epsilon^{-1})$, where T_H is the time complexity for quantum oracles U_H and V_H.*

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

$$|0\rangle|0\rangle \xrightarrow{V_H} \frac{1}{\|H\|_F}\sum_{i=1}^{d}\|\boldsymbol{h}_i\||i\rangle|0\rangle \xrightarrow{U_H} \frac{1}{\|H\|_F}\sum_{i=1}^{d}\sum_{j=1}^{d}h_{ij}|i\rangle|j\rangle. \tag{3.2}$$

Since H has the eigen-decomposition $H = \sum_{k=1}^{r}\lambda_k \boldsymbol{u}_k \boldsymbol{u}_k^T$, we could rewrite each entry of H 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 $\boldsymbol{u}_k$. 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:

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

Then we apply the quantum SVE model on this state. To provide $\epsilon/4$-estimation on the singular value, the time complexity to run the quantum SVE algorithm needs to be $O(T_H\|H\|_F\text{polylog}(d)\epsilon^{-1})$ by Theorem 2.1.

Suppose there exist some eigenvalues $\lambda_j \leq -\alpha + \epsilon/2$. We denote the least one of them as λ_t and label it as the **proper** eigenvalue. By Theorem 3.2, we need $n_t = 2\frac{\|H\|_F^2}{\lambda_t^2}\log\frac{1}{\delta}$ numbers of state $|\boldsymbol{u}_t\rangle$ 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 2-3 in Algorithm 3 is $P_t = \frac{\lambda_t^2}{\|H\|_F^2}$. So we need to perform step 2-3 in Algorithm 3 for $n = 2\frac{\|H\|_F^4}{\lambda_t^4}\log\frac{1}{\delta}$ times. The number n can be roughly upper bounded by $n = 32\frac{\|H\|_F^4}{\alpha^4}\log\frac{1}{\delta}$, 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 and setting the error bound $\delta = 1/\mathrm{poly}(d)$, we could obtain the time complexity of Algorithm 3, that is $O(T_H\|H\|_F^5\alpha^{-4}\mathrm{polylog}(d)\epsilon^{-1})$. □

3.3 Target State Generating

Suppose the result of Algorithm 3 implies the existence of the target eigenvector $\boldsymbol{u}_t$ that satisfies $\boldsymbol{u}_t^T H\boldsymbol{u}_t \leq -\alpha + \epsilon/2$. To give a solution to the NCF problem, we need to obtain the vector $\boldsymbol{u}_t$ efficiently. Thus we develop Algorithm 4 in Section 3.3, which generates the quantum state $|\boldsymbol{u}_t\rangle$ in time $\tilde{O}(\mathrm{polylog}(d))$. The classical read-out of $|\boldsymbol{u}_t\rangle$, which means to obtain the classical form $\boldsymbol{u}_t$ from quantum state $|\boldsymbol{u}_t\rangle$, will be discussed in Chapter 4.

Algorithm 4 Target State Generating

Input: Quantum access to oracles U_H and V_H. The number α and the error bound ϵ in the NCF problem. The probability error bound δ.

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 $4\frac{\|H\|_F^2}{\alpha^2}\log\frac{1}{\delta}$ **do**
2: Create the state $|\psi_{\text{eigen}}\rangle$ (Equation 3.1).
3: Measure the eigenvalue register and mark the result.
4: **if** the eigenvalue is labelled to be **proper** in Algorithm 3, **then**
5: output the state in the first register as the target state.
6: **end if**
7: **end for**

The main idea of Algorithm 4 is similar to Algorithm 3. We still use the state $\frac{1}{\|H\|_F}\sum_{j=1}^r \lambda_j|\boldsymbol{u}_j\rangle|\boldsymbol{u}_j\rangle$ as the input of the quantum SVE algorithm to obtain

state:

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

Denote the target eigenvalue as $\lambda_t \leq -\alpha+\epsilon/2$ and the target eigenstate as $|\boldsymbol{u}_t\rangle$. The probability of generating $|\boldsymbol{u}_t\rangle$ in each iteration of step 2-6 in Algorithm 4 is $P_t = \frac{\lambda_t^2}{\|H\|_F^2} \geq \frac{\alpha^2}{4\|H\|_F^2}$. Thus the probability of generating at least one state $|\boldsymbol{u}_t\rangle$ in $N = 4\frac{\|H\|_F^2}{\alpha^2}\log\frac{1}{\delta}$ times of step 2-6 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 Algorithm 4 could generate at least one state $|\boldsymbol{u}_t\rangle$ with probability at least $1-\delta$. By considering the time complexity to run the quantum SVE algorithm ($O(T_H\|H\|_F\text{polylog}(d)\epsilon^{-1})$) and setting the probability error bound $\delta = 1/\text{poly}(d)$, we could derive the complexity of Algorithm 4 in Theorem 3.3:

THEOREM 3.3. *Suppose the least eigenvalue of H is less than $-\alpha+\epsilon/2$. Denote $|\boldsymbol{u}_t\rangle$ as the corresponding eigenstate. Then Algorithm 4 generates state $|\boldsymbol{u}_t\rangle$ in time $O(T_H\|H\|_F^3\alpha^{-2}\text{polylog}(d)\epsilon^{-1})$ with probability at least $1-1/\text{poly}(d)$, , where T_H is the time complexity for quantum oracles U_H and V_H.*

By using Lemma 3.1, Theorem 3.2 and Theorem 3.3, we could provide the main result of this chapter in Theorem 3.1.

CHAPTER 4

State read-out

Recovering the unknown quantum state from measurements (state read-out) is also known as the quantum state tomography (QST), which is one of the fundamental problems in quantum information science. QST has attracted significant interest from both theoretical [26, 27, 28, 29, 30, 31] and experimental [32, 33, 34] perspectives in recent years. Specifically, reconstructing a $d \times d$ density matrix ρ requires at least $n = O(d^2/\epsilon^2)$ copies for general mixed state case or $n = O(d/\epsilon^2)$ copies for pure state case $\rho = |\boldsymbol{v}\rangle\langle\boldsymbol{v}|$[29]. Directly using state tomography methods for state read-out is computationally expensive and would offset the gained quantum speedup [12]. Since the required number n is proven optimal for both cases [29], any further improvement on n could be achieved only by assuming specific prior knowledge on state ρ. For example, QST via local measurements provides efficient estimation for states which can be determined by locally reduced density matrices [31] or states with a low-rank tensor decomposition [30]. However, the output states generated by most quantum machine learning (QML) algorithms do not have these structures.

Instead, many QML algorithms, which involve an input matrix, have the solution state lies in the row or column matrix space. For example, the quantum SVD algorithm focuses on providing singular value σ_i and corresponding singular vector states $|\boldsymbol{u}_i\rangle$ and $|\boldsymbol{v}_i\rangle$ for matrix $A = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$. States $|\boldsymbol{u}_i\rangle$ and $|\boldsymbol{v}_i\rangle$ lies in the column and row space of A, respectively. Another example is the quantum linear system solver for linear system $A\boldsymbol{x} = \boldsymbol{b}$. Solution state $|\boldsymbol{x}\rangle \propto A^{-1}\boldsymbol{b}$ here lies in the row space of A. Since these solution vectors play

crucial roles in modern machine learning [10], a fast read-out protocol could enhance the capability of existing QML algorithms by providing efficient end-to-end versions.

In this chapter, we design an efficient state read-out protocol that works for QML algorithms which involve an r-rank input matrix stored in quantum random access memory (QRAM), and the output state $|\boldsymbol{v}\rangle$ lies in the row space of the matrix. Our algorithm takes $\tilde{O}(\mathrm{poly}(r))$ copies of output state for the tomography. The main idea is to obtain the description $|\boldsymbol{v}\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle$, or to say $\boldsymbol{v} = \sum_{i=1}^{r} x_i \boldsymbol{s}_i / \|\boldsymbol{s}_i\|$, where $\{\boldsymbol{s}_i\}_{i=1}^{r}$ is a basis selected from rows of the input matrix.

Now we back to the read-out problem of the target state. Recall that our Hessian matrix has the eigen decomposition $H = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^T$. Then, any eigenvector $\boldsymbol{u}_j$ lies in the row space $\mathrm{span}\{\boldsymbol{h}_j\}_{j=1}^{d}$, because $\boldsymbol{u}_j^T H = \lambda_j \boldsymbol{u}_j^T$ can be rewritten as $\sum_{i=1}^{d} \boldsymbol{h}_i^T u_j^{(i)} = \lambda_j \boldsymbol{u}_j^T$. Since H has the rank of r, there exists an r-elements subset of all rows that is complete for the row space:

$$\mathrm{span}\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r} = \mathrm{span}\{\boldsymbol{h}_j\}_{j=1}^{d}.^{1}$$

Thus, eigenvector $\boldsymbol{u}_j$ could also be represented as the linear combination of vectors in $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$. We denote $\boldsymbol{h}_{g(i)}$ as $\boldsymbol{s}_i$ for simplicity. The read-out problem can be viewed as solving the equation:

$$|\boldsymbol{u}_t\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle,$$

where $x_i \in \mathbb{R}, \forall i \in [r]$ are unknown variables. Thus, instead of simply reading out components of vector $\boldsymbol{u}_t$, we could get the classical description $\boldsymbol{u}_t$ by $|\boldsymbol{u}_t\rangle = \sum_{i=1}^{r} x_i |\boldsymbol{s}_i\rangle$. Note that the complete basis $\{\boldsymbol{s}_i\}_{i=1}^{r}$ is not unique, and we only need to identify one of them. The main result is informally stated in Theorem 4.1.

[1] g is a map from $[r]$ to $[d]$, such that $g(i)$ is the index of the i-th row basis vector.

THEOREM 4.1. *The classical description* $\boldsymbol{u}_t = \sum_{i=1}^{r} x_i \boldsymbol{s}_i / \|\boldsymbol{s}_i\|$ *for the negative curvature direction could be presented in time* $\tilde{O}(\mathrm{poly}(r, \log d)\epsilon^{-3})$ *with* l_2 *norm error bounded in* ϵ*, when the complete basis set* $\{\boldsymbol{s}_i\}_{i=1}^{r}$ *is given.*

We also notice some recent breakthroughs about quantum-inspired algorithms [35, 36] based on sampling techniques and the FKV algorithm [37]. These quantum-inspired algorithms perform the approximate SVD and output eigenvector as the linear-sum on a group of row vectors. However, to cover the whole column space, the quantum-inspired algorithm need to sample at least $O(\frac{r^2}{\epsilon^2})$ numbers of rows and columns to form the basis, while our method exactly generates the linear-sum form on r rows. A more detailed comparison between the quantum-inspired sampling algorithm and our read-out protocol is discussed in Section 4.3.

4.1 Complete Basis Selection

In this section, we develop a quantum algorithm to select an index set $S_I = \{g(i)\}_{i=1}^{r}$ from $[d]$, which corresponds to the complete row basis $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$. The complete basis selection (CBS) algorithm is summarized in Algorithm 5 along with circuit implementation illustrated in Figure 4.1. The CBS algorithm can be viewed as a quantum generalization of the Gram-Schmidt orthogonalization.

We remark that there are some related literatures for constructing orthogonal states [38, 39, 40, 41]. However, Ref. [38] focuses on the single-qubit system. Ref. [39, 40] focus on generating the state orthogonal to the input state, which implies a $O(d)$ time complexity for all rows. Ref. [41] constructs orthogonal states from original states by simply lifting the dimension of the Hilbert space, which, although named as the "generalized" Gram-Schmidt process, cannot select the needed complete basis as standard Gram-Schmidt process. Hence, our CBS algorithm is the first efficient quantum version for the Gram-Schmidt process that can be of independent interest.

4.1.1 The CBS Algorithm

Now we describe the CBS algorithm in detail. In the first iteration of the CBS algorithm, we choose $|\boldsymbol{h}_{g(1)}\rangle$ from $\{|\boldsymbol{h}_j\rangle\}_{j\in[d]}$ with probability proportional to $\|\boldsymbol{h}_{g(1)}\|^2$, and define $|\boldsymbol{t}_1\rangle := |\boldsymbol{h}_{g(1)}\rangle$. We initialize the index set $S_I = \{g(1)\}$. In the l-th iteration, a group of orthogonal states $\{|\boldsymbol{t}_m\rangle\}_{m=1}^{l-1}$ and an index set $S_I = \{g(i)\}_{i=1}^{l-1}$ for a group of linearly independent rows are given. We perform the quantum circuit illustrated in Fig 4.1, where the unitary $R_m = I - 2|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|$. We choose the new index $g(l)$ from the set:

$$\left\{ j : \left\| |\boldsymbol{h}_j\rangle - \sum_{m=1}^{l-1} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle \right\| \neq 0 \right\}$$

with probability proportional to $\| |\boldsymbol{h}_j\rangle - \sum_{m=1}^{l-1} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle \|^2$, and obtain the new orthonormal state:

$$|\boldsymbol{t}_l\rangle \propto |\boldsymbol{h}_{g(l)}\rangle - \sum_{m=1}^{l-1} |\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_{g(l)}\rangle.$$

The index set is updated as $S_I = S_I \cup \{g(l)\}$. Finally, after r iterations, we obtain the index set $S_I = \{g(i)\}_{i=1}^{r}$ such that $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$ forms a linearly independent basis, which is complete for the r-rank row space.

Algorithm 5 Complete Basis Selection (CBS)

Input: Quantum access to oracles 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** $l = 1$ to r **do**
3: Run the quantum circuit in Fig 4.1. Measure the third register and post-select on result 0. Measure the first register to obtain an index $g(l)$. Let $S_I = S_I \cup \{g(l)\}$.
4: **end for**

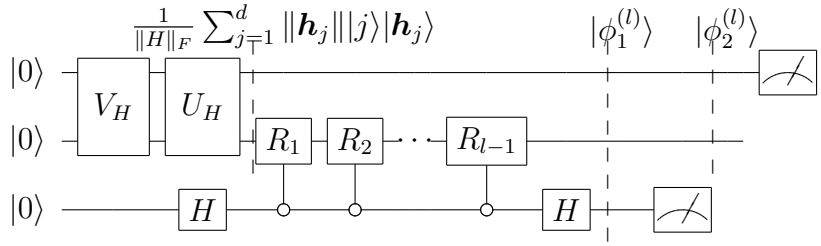

FIGURE 4.1: Quantum circuit for the l-th iteration of the CBS Algorithm.

In the l-th iteration, the quantum circuit first creates the state:

$$\frac{1}{\|H\|_F}\sum_{j=1}^{d}\|\boldsymbol{h}_j\||j\rangle|\boldsymbol{h}_j\rangle|0\rangle,$$

by using oracles U_H and V_H. Then a Hadamard gate is applied to the third register, followed by a list of controlled R_m gate:

$$C(R_m) = R_m \otimes |0\rangle\langle 0| + I \otimes |1\rangle\langle 1|,$$

where the unitary $R_m = I - 2|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|, \forall m \in [l-1]$. The final unitary operation is again a Hadamard gate to the third register. The state before the measurement is:

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

Then we measure the third register and post-select on result 0. The probability of outcome 0 is

$$P_l = \frac{1}{\|A\|_F^2}\sum_{j=1}^{d}\|\boldsymbol{h}_j\|^2\left\||\boldsymbol{h}_j\rangle - \sum_{m=1}^{l-1}|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right\|^2, \tag{4.1}$$

and the post-selected state is

$$|\phi_2^{(l)}\rangle = \frac{1}{\sqrt{P_l}\|H\|_F}\sum_{j=1}^{d}\|\boldsymbol{h}_j\||j\rangle\left[|\boldsymbol{h}_j\rangle - \sum_{m=1}^{l-1}|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_j\rangle\right].$$

Note that we need $O(1/P_l)$ copies of $|\phi_1^{(l)}\rangle$ to generate the state $|\phi_2^{(l)}\rangle$. Finally, we measure the first register for a new basis index $g(l)$ and a new orthogonal

state:

$$|\boldsymbol{t}_l\rangle = \frac{1}{Z_l}\left[|\boldsymbol{h}_{g(l)}\rangle - \sum_{m=1}^{l-1}|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_{g(l)}\rangle\right], \tag{4.2}$$

where $Z_l = \||\boldsymbol{h}_{g(l)}\rangle - \sum_{m=1}^{l-1}|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|\boldsymbol{h}_{g(l)}\rangle\|$ is the normalizing constant.

The difficulty of constructing the circuit in Fig 4.1 is to efficiently implement the controlled version of unitary $R_m = I - 2|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|$. Since each $|\boldsymbol{t}_m\rangle$ is generated during the algorithm, we do not have oracle O_t to prepare $|0\rangle \to |\boldsymbol{t}_m\rangle$, so we cannot implement R_m by $O_t(I - 2|0\rangle\langle 0|)O_t^\dagger$. To overcome the aforementioned difficulty, we note that the state $|\boldsymbol{t}_m\rangle$ lies in $\text{span}\{|\boldsymbol{h}_{g(i)}\rangle\}_{i=1}^m$, so there is $|\boldsymbol{t}_m\rangle = \sum_{i=1}^m z_i|\boldsymbol{h}_{g(i)}\rangle$, where parameters $\{z_i\}_{i=1}^m$ could be calculated. Then we could generate $|\boldsymbol{t}_m\rangle$ with post-selections by the linear combination of unitary (LCU) method [42]. Given copies of $|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|$, we can implement $R_m = I - 2|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m| = e^{-i\pi|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|}$ with the help of the Hamiltonian Simulation technique developed in Quantum PCA [7]. By considering the error of implementing each R_m, we prove that the CBS algorithm could select a linearly independent basis in time $\tilde{O}(\text{poly}(r))$ with probability at least $3/4$. We detail the error and time complexity analysis in Section 4.1.2 and 4.1.3 with the main result provided as Theorem 4.2.

THEOREM 4.2. *By using $O(r^3\|H\|_F^2\lambda_{\min}^{-2}(H)c^{-1/2}(\|H\|_F^2\lambda_{\min}^{-2}(H) + r^{2.5}c^{-1}))$ queries to input oracles, the CBS algorithm could 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, where c is a parameter [2] between $(0, 1)$.*

4.1.2 Implementation of $C(R_l)$

The crucial part in Algorithm 5 is to implement the controlled operation:

$$C(R_l) = R_l \otimes |0\rangle\langle 0| + I \otimes |1\rangle\langle 1|, \forall l \in [r-1],$$

[2] The detail about c will be discussed in Section 4.1 with numerical analysis in Section 4.3.

where $R_l = I - 2|\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l|$. In the following, we denote $\boldsymbol{s}_i := \boldsymbol{h}_{g(i)}$ for the simplicity of notation. By the definition of states $\{|\boldsymbol{t}_l\rangle\}_{l=1}^r$:

$$|\boldsymbol{t}_l\rangle = \frac{1}{Z_l}(|\boldsymbol{s}_l\rangle - \sum_{i=1}^{l-1} |\boldsymbol{t}_i\rangle\langle\boldsymbol{t}_i|\boldsymbol{s}_l\rangle), \tag{4.3}$$

each state $|\boldsymbol{t}_l\rangle$ can be written as the linear combination of states $\{|\boldsymbol{s}_i\rangle\}_{i=1}^l$, namely,

$$|\boldsymbol{t}_l\rangle = \sum_{j=1}^{l} z_j|\boldsymbol{s}_j\rangle,$$

for some coefficients $\{z_j\}_{j=1}^l$. By using the linear combination of unitaries (LCU) and the Hamiltonian simulation methods, we could implement operation $C(R_l)$ with given coefficients $\{z_j\}_{j=1}^l$. See Lemma 4.3 for detail. Lemma 4.1 provides a calculation method of coefficients $\{z_j\}_{j=1}^l$, while the error analysis is in Lemma 4.2. The main result about implementing operation $C(R_l)$ is provided in Proposition 4.1.

First we focus on calculating coefficients $\{z_j\}_{j=1}^l$. Define:

$$\boldsymbol{b} = \sum_{j=1}^{l-1}\langle\boldsymbol{s}_j|\boldsymbol{s}_l\rangle\boldsymbol{e}_j;\ C_m = [c_{ij} \equiv \langle\boldsymbol{s}_i|\boldsymbol{s}_j\rangle]_{i,j}^{m,m}, \forall m \in [r].$$

We provide the equation of $\{z_j\}_{j=1}^l$ in Lemma 4.1. Notation $|\cdot|$ here denotes the determinant of a matrix.

LEMMA 4.1. *All components in vector* $\boldsymbol{z}^T = \sqrt{\frac{|C_{l-1}|}{|C_l|}}(-\boldsymbol{b}^T C_{l-1}^{-1}, 1)$ *could form coefficients* $\{z_i\}_{i=1}^l$, *such that* $|\boldsymbol{t}_l\rangle = \sum_{j=1}^l z_j|\boldsymbol{s}_j\rangle$.

PROOF. The restriction that $|\boldsymbol{t}_l\rangle$ is normalized and is orthogonal to states $|\boldsymbol{s}_1\rangle, |\boldsymbol{s}_2\rangle, \cdots |\boldsymbol{s}_{l-1}\rangle$ could yield:

$$\sum_{i=1}^{l} z_i\langle\boldsymbol{s}_j|\boldsymbol{s}_i\rangle = 0,\ \forall j \in [l-1], \tag{4.4}$$

$$\sum_{j=1}^{l}\sum_{i=1}^{l} z_j z_i\langle\boldsymbol{s}_j|\boldsymbol{s}_i\rangle = 1. \tag{4.5}$$

Note that $z_l = 1/Z_l$ by (4.3). Define the $l-1$ dimensional vector

$$\boldsymbol{y} = -Z_l \sum_{i=1}^{l-1} z_i \boldsymbol{e}_i,$$

and rewrite (4.4) in the vector form:

$$C_{l-1}\boldsymbol{y} = \boldsymbol{b}. \tag{4.6}$$

Equation (4.5) could be written as:

$$\begin{aligned} 1 &= \sum_{j=1}^{l}\sum_{i=1}^{l} z_i z_j \langle \boldsymbol{s}_j | \boldsymbol{s}_i \rangle \\ &= \sum_{i=1}^{l-1}\sum_{j=1}^{l-1} z_i z_j c_{ij} + 2z_l \sum_{i=1}^{l-1} z_i b_i + z_l^2 \\ &= \frac{\boldsymbol{y}^T C_{l-1}\boldsymbol{y} - 2\boldsymbol{y}^T\boldsymbol{b} + 1}{Z_l^2} \\ &= \frac{1 - \boldsymbol{y}^T\boldsymbol{b}}{Z_l^2}, \end{aligned}$$

which yields

$$\boldsymbol{y}^T\boldsymbol{b} = 1 - Z_l^2. \tag{4.7}$$

Solving (4.6) is trivial:

$$\boldsymbol{y} = C_{l-1}^{-1}\boldsymbol{b}, \text{ or, } y_i = \frac{|C_{l-1}^{(i)}|}{|C_{l-1}|}, \ \forall i \in [l-1], \tag{4.8}$$

where $C_{l-1}^{(i)}$ denotes the matrix generated from C_{l-1} by replacing the i-th column with $\boldsymbol{b}$. Replacing y_i in (4.7) by (4.8), there is:

$$\begin{aligned} Z_l^2 &= 1 - \boldsymbol{y}^T\boldsymbol{b} \\ &= 1 - \frac{1}{|C_{l-1}|}\sum_{i=1}^{l-1} |C_{l-1}^{(i)}| b_i \\ &= \frac{|C_{l-1}| - \sum_{i=1}^{l-1} |C_{l-1}^{(i)}| c_{il}}{|C_{l-1}|} \end{aligned}$$

$$= \frac{1}{|C_{l-1}|}\Big[(-1)^{2l}|M_{ll}| + \sum_{i=1}^{l-1}(-1)^{l+i}|M_{il}|c_{il}\Big]$$
$$= \frac{|C_l|}{|C_{l-1}|}.$$

Matrix M_{ij} denotes the minor of $l \times l$ matrix C_l by removing the i-th row and j-th column. The fourth equation holds by noticing:

$$|C_{l-1}^{(i)}| = (-1)^{l-1-i}|M_{il}| = (-1)^{l-1+i}|M_{il}|,$$

since we could obtain the transpose of the matrix $C_{l-1}^{(i)}$ by exchanging the last row of M_{il} with its previous row for $(l-1-i)$ times. Therefore, we have:

$$\boldsymbol{z} = \sqrt{\frac{|C_{l-1}|}{|C_l|}}(-\boldsymbol{b}^T C_{l-1}^{-1}, 1)^T. \tag{4.9}$$

There is another equivalent formulation of coefficients $\boldsymbol{z}$. Consider state:

$$|\boldsymbol{t}_l\rangle = \frac{1}{Z_l}(|\boldsymbol{s}_l\rangle - \sum_{i=1}^{l-1}|\boldsymbol{t}_i\rangle\langle\boldsymbol{t}_i|\boldsymbol{s}_l\rangle),$$

multiply the state $\langle\boldsymbol{t}_l|$ on both sides could yield:

$$\langle\boldsymbol{s}_l|\boldsymbol{t}_l\rangle = \sum_{i=1}^{l} z_i\langle\boldsymbol{s}_l|\boldsymbol{s}_i\rangle = Z_l. \tag{4.10}$$

Define the l-dimensional vector $\boldsymbol{e}_l = (0, 0, \cdots, 0, 1)^T$. Rewrite Equation (4.10), (4.4) and (4.5) in the vector form

$$C_l\boldsymbol{z} = Z_l\boldsymbol{e}_l, \tag{4.11}$$

$$\boldsymbol{z}^T C_l\boldsymbol{z} = 1. \tag{4.12}$$

We could also calculate the coefficients as $\boldsymbol{z} = Z_l C_l^{-1}\boldsymbol{e}_l$. □

By Lemma 4.1, coefficients $\{z_j\}_{j=1}^{l}$ is calculated as:

$$Z_l = \sqrt{|C_l|/|C_{l-1}|},$$

$$\boldsymbol{y} = C_{l-1}^{-1}\boldsymbol{b},$$
$$\boldsymbol{z} = Z_l^{-1}(-\boldsymbol{y}^T, 1)^T.$$

Remark that each element in vector $\boldsymbol{b}$ or matrix C_{l-1} is the inner product between states. We use the technique developed in [43] to estimate each inner product $\langle \boldsymbol{s}_i|\boldsymbol{s}_j\rangle$, which provides ϵ error estimation by using $O(1/\epsilon)$ queries to input oracles. Thus, the error on $\boldsymbol{b}$ and C_{l-1} could influence the accuracy of coefficients $\{z_j\}$. We provide Lemma 4.2 to verify the influence of error to the description $|\boldsymbol{t}_l\rangle = \sum_{j=1}^{l} z_j|\boldsymbol{s}_j\rangle$.

LEMMA 4.2. *The approximate coefficients $\{\tilde{z}_j\}_{j=1}^{l}$ could be obtained by using $O(l^{7/2}\lambda_{\min}^{-3/2}(C_l)\epsilon^{-1})$ queries to U_H, which introduces a l_2 norm error on $|\boldsymbol{t}_l\rangle$ bounded as $\||\boldsymbol{t}_l\rangle - \frac{|\tilde{\boldsymbol{t}}_l\rangle}{\||\tilde{\boldsymbol{t}}_l\rangle\|}\| \leq \epsilon$.*

PROOF. Define $\tilde{C}_{l-1}$ and $\tilde{\boldsymbol{b}}$ as estimations to C_{l-1} and $\boldsymbol{b}$, respectively, where each element in $\tilde{C}_{l-1}$ or $\tilde{\boldsymbol{b}}$ has error bounded by ϵ_1. Denote $\Delta C_{l-1} = \tilde{C}_{l-1} - C_{l-1}$, $\Delta\boldsymbol{b} = \tilde{\boldsymbol{b}} - \boldsymbol{b}$ and $\tilde{\boldsymbol{y}} = \tilde{C}_{l-1}^{-1}\tilde{\boldsymbol{b}}$. Then the norm of $\Delta\boldsymbol{y} = \tilde{\boldsymbol{y}} - \boldsymbol{y}$ could be bounded as

$$\begin{aligned}
\|\Delta\boldsymbol{y}\| &= \|\tilde{C}_{l-1}^{-1}\tilde{\boldsymbol{b}} - C_{l-1}^{-1}\boldsymbol{b}\| \\
&= \|(C_{l-1} + \Delta C_{l-1})^{-1}(\boldsymbol{b} + \Delta\boldsymbol{b}) - C_{l-1}^{-1}\boldsymbol{b}\| \\
&= \|(C_{l-1} + \Delta C_{l-1})^{-1}(\Delta\boldsymbol{b} - \Delta C_{l-1}\cdot C_{l-1}^{-1}\boldsymbol{b})\| \\
&\leq \|C_{l-1}^{-1}\|\cdot\|(I + C_{l-1}^{-1}\Delta C_{l-1})^{-1}\|\cdot(\|\Delta\boldsymbol{b}\| + \|\Delta C_{l-1}\cdot C_{l-1}^{-1}\boldsymbol{b}\|) \\
&\leq \|C_{l-1}^{-1}\|\cdot\frac{\|\Delta\boldsymbol{b}\| + \|\Delta C_{l-1}\|\|\boldsymbol{y}\|}{1 - \|C_{l-1}^{-1}\Delta C_{l-1}\|},
\end{aligned}$$

where the norm $\|\cdot\|$ denotes the largest singular value of the matrix. Based on (4.12), the l_2 norm of vector $\boldsymbol{z}$ is bounded as

$$\|\boldsymbol{z}\|^2 \leq \frac{1}{\lambda_{\min}(C_l)},$$

so there is

$$\|\boldsymbol{y}\| = \sqrt{Z_l^2\|\boldsymbol{z}\|^2 - 1} < Z_l\|\boldsymbol{z}\| \leq Z_l\lambda_{\min}^{-1/2}(C_l).$$

Based on (4.11), there is

$$Z_l = \|C_l\boldsymbol{z}\| \geq \lambda_{\min}(C_l)\|\boldsymbol{z}\| \geq \lambda_{\min}(C_l)|z_l| = \frac{\lambda_{\min}(C_l)}{Z_l},$$

so

$$Z_l \geq \sqrt{\lambda_{\min}(C_l)}. \tag{4.13}$$

There is:

$$\begin{aligned}
\|\Delta\boldsymbol{y}\| &\leq \|C_{l-1}^{-1}\| \cdot \frac{\|\Delta\boldsymbol{b}\| + \|\Delta C_{l-1}\|\|\boldsymbol{y}\|}{1 - \|C_{l-1}^{-1}\Delta C_{l-1}\|} \\
&\leq \frac{\|C_{l-1}^{-1}\|(\sqrt{l-1}\epsilon_1 + (l-1)\epsilon_1 Z_l\lambda_{\min}^{-1/2}(C_l))}{1 - \|C_{l-1}^{-1}\|(l-1)\epsilon_1} \\
&\leq \frac{2(l-1)Z_l\lambda_{\min}^{-3/2}(C_l)\epsilon_1}{1 - (l-1)\lambda_{\min}^{-1}(C_l)\epsilon_1} \leq \epsilon_2,
\end{aligned}$$

where $\epsilon_2 = 3Z_l(l-1)\lambda_{\min}^{-3/2}(C_l)\epsilon_1$. The second equation is derived by noticing $\|\Delta\boldsymbol{b}\| \leq \sqrt{l-1}\epsilon_1$, $\|\boldsymbol{b}\| \leq \sqrt{l-1}$, $\|\Delta C_{l-1}\| \leq (l-1)\epsilon_1$ and $Z_l \geq \lambda_{\min}^{1/2}(C_l)$. The third equation is derived by noticing $\|C_{l-1}^{-1}\| = \lambda_{\min}(C_{l-1}) \leq \lambda_{\min}(C_l)$.

Now we analyse how the error $\Delta\boldsymbol{y}$ could influence the infidelity of the coefficients $\{z_j\}_{j=1}^l$ for state $|\boldsymbol{t}_l\rangle$. Define the unnormalized state as $|\tilde{\boldsymbol{t}}_l\rangle = \sum_{i=1}^l \tilde{z}_i|\boldsymbol{s}_i\rangle$, there is:

$$\begin{aligned}
\langle\boldsymbol{t}_l|\tilde{\boldsymbol{t}}_l\rangle &= \boldsymbol{z}^T C_l\tilde{\boldsymbol{z}} \\
&= \frac{1}{Z_l\tilde{Z}_l}[\boldsymbol{y}^T C_{l-1}\tilde{\boldsymbol{y}} - \boldsymbol{b}^T\boldsymbol{y} - \boldsymbol{b}^T\tilde{\boldsymbol{y}} + 1] \\
&= \frac{1}{Z_l\tilde{Z}_l}[1 - \boldsymbol{b}^T\boldsymbol{y}] = \frac{Z_l}{\tilde{Z}_l}.
\end{aligned}$$

The form $|\tilde{\boldsymbol{t}}_l\rangle = \sum_{i=1}^l \tilde{z}_i|\boldsymbol{s}_i\rangle$ is not normalized:

$\||\tilde{\boldsymbol{t}}_l\rangle\|^2 = \tilde{\boldsymbol{z}}^T C_l\tilde{\boldsymbol{z}}$

$$
\begin{aligned}
&= \frac{1}{\tilde{Z}_l^2}[\tilde{\boldsymbol{y}}^T C_{l-1}\tilde{\boldsymbol{y}} + 1 - 2\tilde{\boldsymbol{y}}^T \boldsymbol{b}] \\
&= \frac{1}{\tilde{Z}_l^2}\Big[\Delta\boldsymbol{y}^T C_{l-1}\Delta\boldsymbol{y} + 2\Delta\boldsymbol{y}^T C_{l-1}\boldsymbol{y} + \boldsymbol{y}^T C_{l-1}\boldsymbol{y} + 1 - 2\Delta\boldsymbol{y}^T\boldsymbol{b} - 2\boldsymbol{y}^T\boldsymbol{b}\Big] \\
&= \frac{1}{\tilde{Z}_l^2}[Z_l^2 + \Delta\boldsymbol{y}^T C_{l-1}\Delta\boldsymbol{y}],
\end{aligned}
$$

where the final equation is derived by noticing $C_{l-1}\boldsymbol{y} = \boldsymbol{b}$ and $Z_l^2 = 1 - \boldsymbol{y}^T\boldsymbol{b}$. So the l_2 norm distance between the state $|\boldsymbol{t}_l\rangle$ and $|\tilde{\boldsymbol{t}}_l\rangle$ is bounded as:

$$
\begin{aligned}
\left\| |\boldsymbol{t}_l\rangle - \frac{|\tilde{\boldsymbol{t}}_l\rangle}{\||\tilde{\boldsymbol{t}}_l\rangle\|} \right\| &= \sqrt{2 - 2\frac{\langle \boldsymbol{t}_l|\tilde{\boldsymbol{t}}_l\rangle}{\||\tilde{\boldsymbol{t}}_l\rangle\|}} \\
&= \sqrt{2 - 2\frac{Z_l}{\sqrt{Z_l^2 + \Delta\boldsymbol{y}^T C_{l-1}\Delta\boldsymbol{y}}}} \\
&\leq \sqrt{2 - 2\frac{Z_l}{\sqrt{Z_l^2 + \|\Delta\boldsymbol{y}\|^2\|C_{l-1}\|}}} \\
&\leq \sqrt{2 - 2\frac{Z_l}{Z_l + \frac{\|C_{l-1}\|\|\Delta\boldsymbol{y}\|^2}{2Z_l}}} \\
&\leq \frac{\|C_{l-1}\|^{1/2}\|\Delta\boldsymbol{y}\|}{Z_l} \\
&\leq \frac{(l-1)^{1/2}\epsilon_2}{Z_l}.
\end{aligned}
$$

The fourth equation follows from

$$
(Z_l^2 + \|\Delta\boldsymbol{y}\|^2\|C_{l-1}\|) - \left(Z_l + \frac{\|C_{l-1}\|\|\Delta\boldsymbol{y}\|^2}{2Z_l}\right)^2 = -\frac{\|C_{l-1}\|^2\|\Delta\boldsymbol{y}\|^4}{4Z_l^2} \leq 0.
$$

Denote

$$
\epsilon = \frac{(l-1)^{\frac{1}{2}}}{Z_l}\epsilon_2 = 3(l-1)^{3/2}\lambda_{\min}^{-3/2}(C_l)\epsilon_1.
$$

To guarantee an ϵ error bound on the accuracy of state $|\boldsymbol{t}_l\rangle$, each inner product $\langle \boldsymbol{s}_i|\boldsymbol{s}_j\rangle$ needs an ϵ_1 error bounded estimation, which takes

$$
O(1/\epsilon_1) = O(l^{3/2}\lambda_{\min}^{-3/2}(C_l)\epsilon^{-1})
$$

queries to input oracles. Considering $\frac{l(l-1)}{2}$ numbers of different inner products, the total query complexity is bounded by $O(l^{7/2}\lambda_{\min}^{-3/2}(C_l)\epsilon^{-1})$. □

Given coefficients $\{z_j\}_{j=1}^l$, we provide a framework to implement operation $C(R_l) = R_l \otimes |0\rangle\langle 0| + I \otimes |1\rangle\langle 1|$ in Lemma 4.3.

LEMMA 4.3. *Given coefficients $\{z_j\}$ such that $|\boldsymbol{t}_l\rangle = \sum_{j=1}^l z_j|\boldsymbol{s}_j\rangle$, the operation $C(R_l)$ can be implemented with error ϵ by using $O(l\lambda_{\min}^{-1/2}(C_l)\epsilon^{-1})$ queries to the oracle U_A.*

PROOF. Assume coefficients $\{z_j\}_{j=1}^l$ are given such that $|\boldsymbol{t}_l\rangle = \sum_{j=1}^l z_j|\boldsymbol{s}_j\rangle$. Define the index oracle: $U_{\text{index}} : |0\rangle^{\otimes \log d} \to \frac{1}{\sqrt{l}}\sum_{j=1}^l |g(j)\rangle$. We could prepare the pure state $\rho_l = |\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l|$ by the linear combination of unitaries method as follows. Firstly, initialize the state $|0\rangle^{\otimes \log d}|0\rangle^{\otimes \log d}|0\rangle$. Then, apply oracle U_{index} on the first register, followed by the oracle U_H, to yield:

$$\frac{1}{\sqrt{l}}\sum_{j=1}^{l}|g(j)\rangle|\boldsymbol{h}_{g(j)}\rangle|0\rangle = \frac{1}{\sqrt{l}}\sum_{j=1}^{l}|g(j)\rangle|\boldsymbol{s}_j\rangle|0\rangle.$$

Denote $z \equiv \max_j |z_j|$. Then we perform the controlled rotation $e^{-i\sigma_y \arccos(z_j/z)}$ on the third register, conditioned on the first register $|g(j)\rangle$, to obtain

$$\frac{1}{\sqrt{l}}\sum_{j=1}^{l}|g(j)\rangle|\boldsymbol{s}_j\rangle\left(\frac{z_j}{z}|0\rangle + \sqrt{1-\frac{z_j^2}{z^2}}|1\rangle\right).$$

Finally, apply $U_{\text{index}}^\dagger$ to the first register and obtain the state

$$\begin{aligned}&\sum_{j=1}^{l}|0\rangle\frac{z_j}{zl}|\boldsymbol{s}_j\rangle|0\rangle + \ orthogonal\ garbage\ state\\ =&\frac{1}{zl}|0\rangle|\boldsymbol{t}_l\rangle|0\rangle + \ orthogonal\ garbage\ state.\end{aligned}$$

The success probability of obtaining state $|\boldsymbol{t}_l\rangle$ is $1/z^2l^2$, so we could prepare the state $|\boldsymbol{t}_l\rangle$ with $O(zl)$ queries to U_H by using the amplitude amplification method [44].

Note that $R_l = I - 2|\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l|$ can be viewed as the unitary with Hamiltonian $\rho_l = |\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l|$:

$$R_l = I - 2|\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l| = e^{-i\pi\rho_l}.$$

Therefore, by using the Hamiltonian simulation method developed in Quantum PCA [7], the operation $C(R_l)$ could be performed with error ϵ consuming $O(\pi^2/\epsilon) = O(1/\epsilon)$ copies of ρ_l. Taking the complexity of generating state $|\boldsymbol{t}_l\rangle$ into account, we could implement operation $C(R_l)$ with error bounded as ϵ by using $O(l \max_i |z_j|/\epsilon)$ queries to U_H when coefficients $\{z_j\}_{j=1}^l$ are given. Based on (4.12), the l_2 norm of vector $\boldsymbol{z}$ is bounded as

$$\|\boldsymbol{z}\|^2 \leq \frac{1}{\lambda_{\min}(C_l)},$$

which yields:

$$\max_j |z_j| \leq \|\boldsymbol{z}\| \leq \lambda_{\min}^{-1/2}(C_l). \tag{4.14}$$

So the query complexity for implementing $C(R_l)$ with given coefficients could be bounded as $O(l\lambda_{\min}^{-1/2}(C_l)\epsilon^{-1})$. □

With the help of Lemma 4.1, Lemma 4.2 and Lemma 4.3, we could derive the main result about implementing operation $C(R_l)$ in Proposition 4.1.

PROPOSITION 4.1. *Operation $C(R_l)$ could be implemented with error bounded in ϵ by using $O(l\lambda_{\min}^{-1/2}(C_l)\epsilon^{-1})$ queries to input oracles. The construction of $C(R_l)$ needs a group of coefficients $\{z_i\}_{i=1}^l$, which could be obtained by using $O(l^{7/2}\lambda_{\min}^{-3/2}(C_l)\epsilon^{-1})$ queries to oracle U_H. C_l is the gram matrix of states $\{|\boldsymbol{s}_i\rangle\}_{i=1}^l$.*

PROOF. By Lemma 4.1 and Lemma 4.2, one could obtain coefficients $\{\tilde{z}_j\}_{j=1}^l$ by using $O(l^{7/2}\lambda_{\min}^{-3/2}(C_l)\epsilon^{-1})$ queries to oracle U_H, such that the corresponding state $|\tilde{\boldsymbol{t}}_l\rangle$ satisfies:

$$\left\| |\boldsymbol{t}_l\rangle - \frac{|\tilde{\boldsymbol{t}}_l\rangle}{\||\tilde{\boldsymbol{t}}_l\rangle\|} \right\| \leq \frac{\epsilon}{5}.$$

By Lemma 4.3, given known coefficients, constructing operation $C(R_l)$ with error bounded by $\frac{\epsilon}{5}$ needs $O(l\lambda_{\min}^{-1/2}(C_l)\epsilon^{-1})$ queries to input oracle. Then the total error for implementing $C(R_l)$ is:

$$\begin{aligned} error(R_l) &\leq \frac{\epsilon}{5} + \|R_l - \tilde{R}_l\| \\ &= \frac{\epsilon}{5} + 2\||\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l| - |\tilde{\boldsymbol{t}}_l\rangle\langle\tilde{\boldsymbol{t}}_l|\| \\ &\leq \frac{\epsilon}{5} + 4 \cdot error(|\boldsymbol{t}_l\rangle) \\ &\leq \frac{\epsilon}{5} + 4\frac{\epsilon}{5} = \epsilon. \end{aligned}$$

□

4.1.3 Error and Runtime Analysis

In this section, we analyse the error and runtime of the Complete Basis Selection Algorithm (Algorithm 5). The main result is provided in Theorem 4.3.

THEOREM 4.3. *By using* $O(r^3\|H\|_F^2\lambda_{\min}^{-2}(H)c^{-1/2}(\|H\|_F^2\lambda_{\min}^{-2}(H) + r^{2.5}c^{-1}))$ *queries to input oracles, Algorithm 5 could find an index set* $\{g(i)\}_{i=1}^r$*, which forms a complete basis* $\{\boldsymbol{h}_{g(i)}\}_{i=1}^r$ *with probability at least 3/4. Parameter* $c = \lambda_{\min}(C_{r-1})$ *is a positive number between* $(0, 1)$.

PROOF. First, we discuss the success probability of Algorithm 5, which could be influenced by the error on operation $R_l = I - 2|\boldsymbol{t}_l\rangle\langle\boldsymbol{t}_l|$. Suppose each R_l is implemented with error $\|\tilde{R}_l - R_l\| \leq \epsilon$. Denote $\Pi_l = \prod_{i=1}^l R_i$ and $\tilde{\Pi}_l = \prod_{i=1}^l \tilde{R}_i$, then there is $\|\tilde{\Pi}_l - \Pi_l\| \leq l\epsilon$ by [23]. State $|\phi_2^{(l)}\rangle$ in Algorithm 5 can be rewritten as:

$$|\phi_2^{(l)}\rangle = \frac{1}{\sqrt{P_l}\|H\|_F}\sum_{j=1}^{d}\|\boldsymbol{h}_j\||j\rangle\frac{\Pi_{l-1} + I}{2}|\boldsymbol{h}_j\rangle,$$

and P_l could also be rewritten as:

$$P_l = \frac{1}{\|H\|_F^2} \sum_{j=1}^{d} \|\boldsymbol{h}_j\|^2 \|\frac{\Pi_{l-1} + I}{2}|\boldsymbol{h}_j\rangle\|^2. \tag{4.15}$$

Similarly, we denote:

$$|\tilde{\phi}_2^{(l)}\rangle = \frac{1}{\sqrt{\tilde{P}_l}\|H\|_F} \sum_{j=1}^{d} \|\boldsymbol{h}_j\||j\rangle \frac{\tilde{\Pi}_{l-1} + I}{2}|\boldsymbol{h}_j\rangle, \tag{4.16}$$

$$\tilde{P}_l = \frac{1}{\|H\|_F^2} \sum_{j=1}^{d} \|\boldsymbol{h}_j\|^2 \|\frac{\tilde{\Pi}_{l-1} + I}{2}|\boldsymbol{h}_j\rangle\|^2. \tag{4.17}$$

Remark the objective of Algorithm 5 is to obtain the index set $\{g(l)\}$ such that corresponding rows are linearly independent. We denote P_l^{false} as the probability of selecting out the state $|\boldsymbol{s}_l\rangle \in \text{span}\{|\boldsymbol{s}_i\rangle\}_{i=1}^{l-1}$, which implies the failure of the Algorithm 5 at the l-th iteration. Denote $S_l = \text{span}\{|\boldsymbol{s}_i\rangle\}_{i=1}^{l-1}$, there is:

$$P_l^{false} \tag{4.18}$$

$$= \sum_{j:|\boldsymbol{h}_j\rangle \in S_l} P\left(\text{resulting } j \text{ when measure } |\tilde{\phi}_2^{(l)}\rangle\right) \tag{4.19}$$

$$= \frac{1}{\tilde{P}_l\|H\|_F^2} \sum_{j:|\boldsymbol{h}_j\rangle \in S_l} \|\boldsymbol{h}_j\|^2 \left\| \frac{\tilde{\Pi}_{l-1} + I}{2}|\boldsymbol{h}_j\rangle \right\|^2 \tag{4.20}$$

$$= \frac{\sum_{j:|\boldsymbol{h}_j\rangle \in S_l} \|\boldsymbol{h}_j\|^2 \left\|(\tilde{\Pi}_{l-1} + I)|\boldsymbol{h}_j\rangle\right\|^2}{\sum_{j=1}^{d} \|\boldsymbol{h}_j\|^2 \left\|(\tilde{\Pi}_{l-1} + I)|\boldsymbol{h}_j\rangle\right\|^2} \tag{4.21}$$

$$= \frac{\sum_{j:|\boldsymbol{h}_j\rangle \in S_l} \|\boldsymbol{h}_j\|^2 \left\|(\tilde{\Pi}_{l-1} - \Pi_{l-1})|\boldsymbol{h}_j\rangle\right\|^2}{\sum_{j=1}^{d} \|\boldsymbol{h}_j\|^2 \left[2 + 2\langle \boldsymbol{h}_j| \left(\tilde{\Pi}_{l-1} - \Pi_{l-1} + \Pi_{l-1}\right)|\boldsymbol{h}_j\rangle\right]} \tag{4.22}$$

$$\leq \frac{\sum_{j:|\boldsymbol{h}_j\rangle \in S_l} \|\boldsymbol{h}_j\|^2 (l-1)^2\epsilon^2}{\sum_{j=1}^{d} \|\boldsymbol{h}_j\|^2 \left[\|(\Pi_{l-1} + I)|\boldsymbol{h}_j\rangle\|^2 - 2\|\tilde{\Pi}_{l-1} - \Pi_{l-1}\|\right]} \tag{4.23}$$

$$\leq \frac{\|H\|_F^2 (l-1)^2\epsilon^2}{4\|H\|_F^2 P_l - 2\|H\|_F^2 (l-1)\epsilon}. \tag{4.24}$$

Equation (4.21) is derived by using (4.17). Equation (4.22) is derived by noticing that for state $|\boldsymbol{h}_j\rangle \in S_l$, $(\Pi_{l-1}+I)|\boldsymbol{h}_j\rangle = 0$. Inequality (4.23) is derived by using $\|\tilde{\Pi}_{l-1} - \Pi_{l-1}\| \leq (l-1)\epsilon$ and $\|(\Pi_{l-1}+I)|\boldsymbol{h}_j\rangle\|^2 = 2 + 2\langle \boldsymbol{h}_j|\Pi_{l-1}|\boldsymbol{h}_j\rangle$. Inequality (4.24) comes from (4.15). We provide Lemma 4.4 for a lower bound on P_l.

LEMMA 4.4. *The probability P_l is lower bounded by* $\frac{(r+1-l)\lambda^2_{\min}(H)}{\|H\|_F^2}$.

PROOF. Denote $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_r|$, where $\{\lambda_j\}_{j=1}^r$ are eigenvalues of the matrix $H = \sum_{i=1}^r \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T$. Since state $|\boldsymbol{t}_m\rangle$ is the linear sum of rows $\{|\boldsymbol{h}_j\rangle\}_{j=1}^d$, while each row is the linear sum $\boldsymbol{h}_j = \sum_{i=1}^r \lambda_i u_i^{(j)} \boldsymbol{u}_i$, we can further write $|\boldsymbol{t}_m\rangle = \sum_{i=1}^r w_{mi}|\boldsymbol{u}_i\rangle$. There is:

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

where $c_i = \sum_{m=1}^{l-1} w_{mi}^2$. The third equation is derived by using $\boldsymbol{h}_j = \sum_{i=1}^r \lambda_i u_i^{(j)}\boldsymbol{u}_i$ and $|\boldsymbol{t}_m\rangle = \sum_{i=1}^r w_{mi}|\boldsymbol{u}_i\rangle$, the fifth equation is derived by using $\boldsymbol{u}_i^T\boldsymbol{u}_i = \sum_{j=1}^N (u_i^{(j)})^2 = 1$ and $\sum_{i=1}^r w_{mi}w_{ni} = \langle\boldsymbol{t}_m|\boldsymbol{t}_n\rangle = \delta_{mn}$.

Define the r-dimensional vector $\boldsymbol{w}_m = \sum_{i=1}^r w_{mi}\boldsymbol{e}_i$. Vectors in set $\{\boldsymbol{w}_m\}_{m=1}^{l-1}$ are orthogonal with each other. We can add $\boldsymbol{w}_l, \cdots \boldsymbol{w}_r$ such that $\{\boldsymbol{w}_m\}_{m=1}^r$

forms an orthonormal basis in the r-dimensional space. Denote matrix $W = (\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_r)$. Since $W^T W = I$, we have:

$$0 \leq c_i = \sum_{m=1}^{l-1} w_{mi}^2 \leq \sum_{m=1}^{r} w_{mi}^2 = [WW^T]_{ii} = 1, \forall i \in [r].$$

Note that

$$\sum_{i=1}^{r} c_i = \sum_{i=1}^{r} \sum_{m=1}^{l-1} w_{mi}^2 = \sum_{m=1}^{l-1} \sum_{i=1}^{r} w_{mi}^2 = l - 1,$$

so there is:

$$P_l \geq 1 - \frac{1}{\|H\|_F^2} \sum_{i=1}^{l-1} \lambda_i^2 = \frac{\sum_{i=l}^{r} \lambda_i^2}{\|H\|_F^2} \geq \frac{(r+1-l)\lambda_{\min}^2(H)}{\|H\|_F^2}. \tag{4.25}$$

□

Let $\epsilon = \frac{\lambda_{\min}^2(H)}{r\|H\|_F^2}$ and insert (4.25) to (4.24), there is:

$$\begin{aligned} P^{false} &\leq \sum_{l=1}^{r} P_l^{false} \leq \sum_{l=0}^{r-1} \frac{l^2\epsilon^2}{4r(r-l)\epsilon - 2l\epsilon} < \sum_{l=0}^{r-1} \frac{l}{2}\epsilon \\ &= \frac{r(r-1)}{4} \frac{\lambda_{\min}^2(H)}{r\|H\|_F^2} = \frac{(r-1)\lambda_{\min}^2(H)}{4\|H\|_F^2} < \frac{1}{4}. \end{aligned}$$

Thus, when operations $R_l, \forall l \in [r-1]$ are implemented with error bounded by $\epsilon = \frac{\lambda_{\min}^2(H)}{r\|H\|_F^2}$, Algorithm 5 could select out a complete basis $\{|\boldsymbol{s}_i\rangle\}_{i=1}^r$ with probability at least $\frac{3}{4}$.

Now we analyse the time complexity of the Complete Basis Selection Algorithm. Denote T_{basis} as the required time to implement Algorithm 5 when each R_i could have an error bounded by ϵ. Denote T_{R_i} as the required time to implement operation R_i. Recall that in each iteration of $l \in [r]$ in Algorithm 5, we perform operation $U_H, V_H, R_1, R_2, \cdots, R_{l-1}$ for $1/P_l$ times. Denote T_H as the time complexity of oracles U_H, V_H, there is:

$$T_{\text{basis}} = \sum_{l=1}^{r} \frac{1}{P_l} \left(2T_H + \sum_{m=1}^{l-1} T_{R_m} \right)$$

$$\leq \sum_{l=1}^{r} \frac{\|H\|_F^2}{(r+1-l)\lambda_{\min}^2(H)} \sum_{m=1}^{l-1} T_H O(m\lambda_{\min}^{-1/2}(C_m)\epsilon^{-1})$$
$$\leq T_H O(c^{-1/2}\lambda_{\min}^{-2}(H)\|H\|_F^2\epsilon^{-1}) \sum_{l=1}^{r} \frac{l(l-1)}{r+1-l}$$
$$= T_H O(r^3\lambda_{\min}^{-4}(H)c^{-1/2}\|H\|_F^4),$$

where c is a positive number between $(\lambda_{\min}(C_{r-1}), 1)$. The second equation follows from (4.25) and Proposition 4.1, and the fourth equation follows from $\epsilon = \frac{\lambda_{\min}^2(H)}{r\|H\|_F^2}$. Calculation of coefficients $\{z_j\}_{j=1}^{l}$ for all $R_l, l \in [r-1]$ takes time:

$$\sum_{l=1}^{r-1} T_H O(l^{7/2}\epsilon^{-1}\lambda_{\min}^{-3/2}(C_l))$$
$$\leq T_H O(r\|H\|_F^2\lambda_{\min}^{-2}(H)c^{-3/2}) \sum_{l=1}^{r-1} l^{7/2}$$
$$= T_H O(r^{5.5}\|H\|_F^2\lambda_{\min}^{-2}(H)c^{-3/2}),$$

where c is a positive number between $(\lambda_{\min}(C_{r-1}), 1)$. By considering both the required time for calculating coefficients $\{z_j\}_{j=1}^{l}$ for $l \in [r-1]$ and the time T_{basis} to implement Algorithm 5, we provide Theorem 4.3. □

We also provide Lemma 4.5, which gives the time complexity of confirming whether vectors in the given set $\{\boldsymbol{s}_i\}_{i=1}^{r}$ are linearly independent. The proof of Lemma 4.5 is in Appendix.

LEMMA 4.5. *It takes $O(r^3)$ time to check whether the vector set $\{\boldsymbol{s}_i\}_{i=1}^{r}$ is linearly independent, when the classical access to Hessian H is given.*

4.2 Coordinate Estimation

In this section, we consider the read-out of a quantum state $|\boldsymbol{v}\rangle$ by providing the corresponding linear-sum description on the row basis, where $|\boldsymbol{v}\rangle$ can be

any state lies in the row space of matrix H. Given the selected complete basis $\{|\boldsymbol{s}_1\rangle,\ |\boldsymbol{s}_2\rangle,\ \cdots, |\boldsymbol{s}_r\rangle\}$, the read-out problem could be viewed as solving the equation $|\boldsymbol{v}\rangle = \sum_{i=1}^{r} x_i|\boldsymbol{s}_i\rangle$, where $x_i \in \mathbb{R}$ are unknown variables. We provide the read-out framework in Algorithm 6.

Algorithm 6 Coordinate Estimation

Input: QRAM oracle for basis rows $\{\boldsymbol{s}_i\}_{i=1}^{r}$. Copies of state $\rho = |\boldsymbol{v}\rangle\langle\boldsymbol{v}|$.

Output: Coordinates $\{x_i\}_{i=1}^{r}$ in the linear combination $|\boldsymbol{v}\rangle = \sum_{i=1}^{r} x_i|\boldsymbol{s}_i\rangle$.

1: Estimate the overlap $c_{ij} = \langle \boldsymbol{s}_i|\boldsymbol{s}_j\rangle$ for $i, j \in [r]$.

2: Estimate the overlap $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$ for $i \in [r]$.

3: Output the solution $\boldsymbol{x} = C^{-1}\boldsymbol{a}$, where $C = [c_{ij}]_{i,j=1}^{r\times r}$ and $\boldsymbol{a} = \sum_{i=1}^{r} a_i\boldsymbol{e}_i$.

Note that there is:

$$\langle \boldsymbol{s}_j|\boldsymbol{v}\rangle = \sum_{i=1}^{r} x_i\langle \boldsymbol{s}_j|\boldsymbol{s}_i\rangle,\ \forall j \in [r],$$

so coordinates $\{x_i\}_{i=1}^{r}$ could be obtained by solving the r-dimensional linear system $C\boldsymbol{x} = \boldsymbol{a}$, where $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$ and $c_{ij} = \langle \boldsymbol{s}_i|\boldsymbol{s}_j\rangle$ for $i, j \in [r]$. Here C_r is denoted by C for simplicity.

4.2.1 Overlap Estimation

The crucial part of Algorithm 6 is to obtain values c_{ij} and $a_i,\ \forall i, j \in [r]$. The overlap $c_{ij} = \langle \boldsymbol{s}_i|\boldsymbol{s}_j\rangle$ can be estimated by simply employing the technique developed in [43] with error bounded by ϵ using $O(\epsilon^{-1})$ queries to input oracles. The estimation to $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$ is more complicate. We could use the quantum SWAP test to estimate the projection $|\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle|^2$ firstly, while $\text{sign}(a_i)$ remains unknown. To overcome this difficulty, we could assume that the state $|\boldsymbol{v}\rangle$ and $|\boldsymbol{s}_k\rangle$ has the positive overlap, and analysis the value:

$$a_i = \text{sign}(\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle)|\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle| = \frac{\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle}{|\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|}$$

as the state overlap, where $k = \text{argmax}_{i\in[r]}|\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle|$ can be chosen with the help of the SWAP test. This assumption is equivalent to adding a global phase 0 or $e^{i\pi}$ on $|\boldsymbol{v}\rangle$. Note that for a solution state $|\boldsymbol{v}\rangle$ to some quantum algorithm, the state $-|\boldsymbol{v}\rangle$ is the same state which involves a π-global phase, and is also the solution in many scenarios. For example, in the negative curvature finding problem where the target $|\boldsymbol{v}\rangle$ is $|\boldsymbol{u}_t\rangle$, both $|\boldsymbol{u}_t\rangle$ and $-|\boldsymbol{u}_t\rangle$ are legal eigenstate.

We construct the quantum circuit illustrated in Fig 4.2 for estimating $\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$. Unitary U_i here is the input oracle that performs operation $|0\rangle \to |\boldsymbol{s}_i\rangle$ and can be implemented efficiently by the QRAM of H. We present the detail of estimating $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$ in Algorithm 7.

Algorithm 7 overlap estimation

Input: Quantum access to input oracles U_i for rows $\boldsymbol{s}_i$, $i \in [r]$. Copies of state $\rho = |\boldsymbol{v}\rangle\langle \boldsymbol{v}|$. The precision parameter ϵ.

Output: An ϵ-estimation $\tilde{a}_i$ to the value $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle, \forall i \in [r]$.

1: Calculate the $\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}$-estimation on value $a_i^2 = Tr(\rho|\boldsymbol{s}_i\rangle\langle \boldsymbol{s}_i|)$, for $i \in [r]$ by SWAP Test. Mark $k \equiv \text{argmax}_{i\in[r]}a_i^2$.
2: **for** $i \in [r]$ and $i \neq k$ **do**
3: Obtain $\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}$-estimation on value $\langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle$ by using the quantum circuit in Fig 4.2.
4: Calculate the value $\tilde{a}_i = \frac{\langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle}{|\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|}$.
5: **end for**

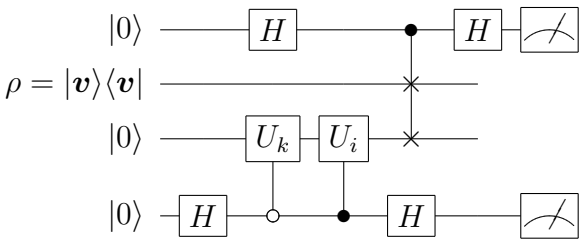

FIGURE 4.2: Quantum circuit for estimating $\langle \boldsymbol{s}_k|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$.

Apart from the relatively trivial part of estimating a_i^2 by SWAP Test, Algorithm 7 runs quantum circuit in Fig 4.2 to estimate $\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$. The whole state in Fig 4.2 before measurements is:

$$\begin{aligned}
&\frac{1}{4}|0\rangle\Big[|\boldsymbol{v}\rangle|\boldsymbol{s}_k\rangle + |\boldsymbol{v}\rangle|\boldsymbol{s}_i\rangle + |\boldsymbol{s}_k\rangle|\boldsymbol{v}\rangle + |\boldsymbol{s}_i\rangle|\boldsymbol{v}\rangle\Big]|0\rangle \\
+&\frac{1}{4}|0\rangle\Big[|\boldsymbol{v}\rangle|\boldsymbol{s}_k\rangle - |\boldsymbol{v}\rangle|\boldsymbol{s}_i\rangle + |\boldsymbol{s}_k\rangle|\boldsymbol{v}\rangle - |\boldsymbol{s}_i\rangle|\boldsymbol{v}\rangle\Big]|1\rangle \\
+&\frac{1}{4}|1\rangle\Big[|\boldsymbol{v}\rangle|\boldsymbol{s}_k\rangle + |\boldsymbol{v}\rangle|\boldsymbol{s}_i\rangle - |\boldsymbol{s}_k\rangle|\boldsymbol{v}\rangle - |\boldsymbol{s}_i\rangle|\boldsymbol{v}\rangle\Big]|0\rangle \\
+&\frac{1}{4}|1\rangle\Big[|\boldsymbol{v}\rangle|\boldsymbol{s}_k\rangle - |\boldsymbol{v}\rangle|\boldsymbol{s}_i\rangle - |\boldsymbol{s}_k\rangle|\boldsymbol{v}\rangle + |\boldsymbol{s}_i\rangle|\boldsymbol{v}\rangle\Big]|1\rangle.
\end{aligned}$$

Then we measure the first and the last register, which outcome results 00 or 11 with probability:

$$\begin{aligned}
P_{00} &= \frac{2 + 2\langle \boldsymbol{s}_i|\boldsymbol{s}_k\rangle + |\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|^2 + |\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle|^2 + 2\langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle}{8}, \\
P_{11} &= \frac{2 - 2\langle \boldsymbol{s}_i|\boldsymbol{s}_k\rangle - |\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|^2 - |\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle|^2 + 2\langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle}{8}.
\end{aligned}$$

Two measurement results are the same with probability:

$$P_{same} = P_{00} + P_{11} = \frac{1 + \langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle}{2}.$$

So similar to the SWAP Test, the proposed quantum circuit provides a ϵ error estimation to the value $\langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle$ with $O(1/\epsilon^2)$ measurements. By using this circuit along with the SWAP Test, we could estimate the value $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$. The error and time complexity about estimating a_i by Algorithm 7 is provided in Proposition 4.2.

PROPOSITION 4.2. *Algorithm 7 could present ϵ-estimation to the value $a_i = \langle \boldsymbol{v}|\boldsymbol{s}_i\rangle$ for all $i \in [r]$ by using $O(\lambda_{\min}^{-1}(C)r^2/\epsilon^2)$ copies of state $\rho = |\boldsymbol{v}\rangle\langle \boldsymbol{v}|$ and $O(\lambda_{\min}^{-1}(C)r^2/\epsilon^2)$ queries to input oracles.*

PROOF. Since there is $C\boldsymbol{x} = \boldsymbol{a}$ and $\langle \boldsymbol{v}|\boldsymbol{v}\rangle = \boldsymbol{x}^T C\boldsymbol{x} = 1$, we have:

$$\begin{aligned}|\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle| = \max_{i\in[r]} |\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle| &\geq \sqrt{\frac{1}{r}\sum_{i=1}^{r} |\langle \boldsymbol{v}|\boldsymbol{s}_i\rangle|^2} \\ &= \sqrt{\frac{1}{r}\sum_{i=1}^{r} a_i^2} = \sqrt{\frac{1}{r}\boldsymbol{a}^T\boldsymbol{a}} = \sqrt{\frac{1}{r}\boldsymbol{x}^T C^2\boldsymbol{x}} \\ &\geq \sqrt{\frac{1}{r}\lambda_{\min}(C)\boldsymbol{x}^T C\boldsymbol{x}} = \sqrt{\frac{\lambda_{\min}(C)}{r}}.\end{aligned}$$

For $\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}$-error estimation $\tilde{a}_k^2$ on value $a_k^2 = |\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|^2$, there is:

$$\epsilon(|a_k|) = |(|\tilde{a}_k| - |a_k|)| = \left|\frac{\tilde{a}_k^2 - a_k^2}{|\tilde{a}_k| + |a_k|}\right| \leq \frac{\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}}{\sqrt{\frac{\lambda_{\min}(C)}{r}}} = \frac{\epsilon}{2}.$$

So a $\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}$-error estimation on value $a_k^2 = |\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|^2$ could ensure a $\epsilon/2$-error estimation on value $|a_k| = |\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle|$. Denote $a_{ik} = \langle \boldsymbol{s}_i|\boldsymbol{v}\rangle\langle \boldsymbol{v}|\boldsymbol{s}_k\rangle$ for simplicity. Then Algorithm 7 calculate the overlap by $a_i = a_{ik}/|a_k|$ for all $i \in [r]$. Since the value a_{ik} is estimated with error bounded by $\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}$, and a_k has the error bounded by $\frac{\epsilon}{2}$, there is:

$$\begin{aligned}\epsilon(a_i) = |\tilde{a}_i - a_i| &= \left|\frac{\tilde{a}_{ik}}{|\tilde{a}_k|} - \frac{a_{ik}}{|a_k|}\right| \\ &\leq \left|\frac{\tilde{a}_{ik}}{|\tilde{a}_k|} - \frac{a_{ik}}{|\tilde{a}_k|}\right| + \left|\frac{a_{ik}}{|\tilde{a}_k|} - \frac{a_{ik}}{|a_k|}\right| \\ &= \frac{\epsilon(a_{ik})}{|\tilde{a}_k|} + |a_{ik}|\frac{\epsilon(a_k)}{|\tilde{a}_k a_k|} \leq \frac{\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}}}{\sqrt{\frac{\lambda_{\min}(C)}{r}}} + \frac{\epsilon}{2} = \epsilon.\end{aligned}$$

The required copies of $\rho = |\boldsymbol{v}\rangle\langle \boldsymbol{v}|$ is:

$$2r \cdot O((\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}})^{-2}) = O(\lambda_{\min}^{-1}(C)r^2/\epsilon^2).$$

The required queries to input oracles is:

$$3r \cdot O((\frac{\epsilon}{2}\sqrt{\frac{\lambda_{\min}(C)}{r}})^{-2}) = O(\lambda_{\min}^{-1}(C)r^2/\epsilon^2).$$

□

4.2.2 Error and Runtime Analysis

In this section, we discuss the error of Algorithm 6 along with the time complexity analysis. Remark that each element a_i or c_{ij} is estimated with error, which could introduce an error on the final classical description of state $|\boldsymbol{v}\rangle$. We provide Lemma 4.6 to verify the influence of error on $\boldsymbol{a}$ and C to the accuracy of the description $\boldsymbol{v} = \sum_{i=1}^{r} x_i \boldsymbol{s}_i / \|\boldsymbol{s}_i\|$.

LEMMA 4.6. *Suppose $\tilde{c}_{jk}$ is a ϵ_1-approximation to $c_{ij} = \langle \boldsymbol{s}_i | \boldsymbol{s}_j \rangle$ and $\tilde{a}_j$ is a ϵ_2-approximation to $a_j = \langle \boldsymbol{v} | \boldsymbol{s}_j \rangle$, $\forall j, k \in [r]$, where $\epsilon_1 = \frac{\lambda_{\min}^{3/2}(C)\epsilon}{3r^{3/2}}$ and $\epsilon_2 = \frac{\lambda_{\min}(C)\epsilon}{3r}$. Then the coordinate $\tilde{\boldsymbol{x}} = \tilde{C}^{-1}\tilde{\boldsymbol{a}}$ leads to an approximate vector $\tilde{\boldsymbol{v}} = \sum_{i=1}^{r} \tilde{x}_i \boldsymbol{s}_i$, such that $\|\tilde{\boldsymbol{v}} - \boldsymbol{v}\| \leq \epsilon$.*

PROOF. Denote $\Delta c_{ij} = \tilde{c}_{ij} - c_{ij}$ and $\Delta a_j = \tilde{a}_j - a_j$, $\forall i, j \in [r]$. Since the error $|\Delta c_{ij}| \leq \epsilon_1$ and $|\Delta a_j| \leq \epsilon_2$, there is:

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

The matrix norm $\|\cdot\|$ here is the spectrum norm. Note that both the value $c_{ij} = \langle \boldsymbol{s}_i | \boldsymbol{s}_j \rangle$ and $a_j = \langle \boldsymbol{v} | \boldsymbol{s}_j \rangle$ are bounded in $[-1, 1]$, so:

$$\|C\| \leq \|C\|_F \leq r \text{ and } \|\boldsymbol{a}\| \leq \sqrt{r}.$$

The norm of $\Delta \boldsymbol{x} = \tilde{\boldsymbol{x}} - \boldsymbol{x}$ is bounded by:

$$\begin{aligned}
\|\Delta \boldsymbol{x}\| &= \|\tilde{C}^{-1}\tilde{\boldsymbol{a}} - C^{-1}\boldsymbol{a}\| \\
&= \|(C + \Delta C)^{-1}(\boldsymbol{a} + \Delta \boldsymbol{a}) - (C + \Delta C)^{-1}(C + \Delta C)C^{-1}\boldsymbol{a}\| \\
&= \|(C + \Delta C)^{-1}(\Delta \boldsymbol{a} - \Delta C \cdot C^{-1}\boldsymbol{a})\| \\
&\leq \|C^{-1}\| \cdot \|(I + C^{-1}\Delta C)^{-1}\| \cdot (\|\Delta \boldsymbol{a}\| + \|\Delta C \cdot \boldsymbol{x}\|) \\
&\leq \|C^{-1}\| \cdot \frac{1}{1 - \|C^{-1}\Delta C\|} \cdot (\|\Delta \boldsymbol{a}\| + \|\Delta C\| \cdot \|\boldsymbol{x}\|)
\end{aligned}$$

$$\leq \frac{\|C^{-1}\|}{1-\|C^{-1}\|r\epsilon_1} \cdot (\sqrt{r}\epsilon_2 + r\epsilon_1 \cdot \lambda_{\min}^{-1/2}(C)) \leq \frac{\epsilon}{\sqrt{r}}.$$

The sixth equation is derived by noticing:

$$\lambda_{\min}(C)\|\boldsymbol{x}\|^2 \leq \boldsymbol{x}^T C \boldsymbol{x} = \langle \boldsymbol{v}|\boldsymbol{v}\rangle = 1.$$

The seventh equation is derived by using $\epsilon_1 = \frac{\lambda_{\min}^{3/2}(C)\epsilon}{3r^{3/2}}$, $\epsilon_2 = \frac{\lambda_{\min}(C)\epsilon}{3r}$ and $\|C^{-1}\| = \lambda_{\min}^{-1}(C)$. Thus, for $\boldsymbol{v} = \sum_{j=1}^{r} x_j \boldsymbol{s}_j/\|\boldsymbol{s}_j\|$ and $\tilde{\boldsymbol{v}} = \sum_{j=1}^{r} \tilde{x}_j \boldsymbol{s}_j/\|\boldsymbol{s}_j\|$, there is:

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

□

By using Proposition 4.2 and Lemma 4.6, we provide the required quantum oracle and state resources in Theorem 4.4 for a ϵ error bounded classical description.

THEOREM 4.4. *Algorithm 6 provides a classical description $\boldsymbol{v} = \sum_{i=1}^{r} x_i \boldsymbol{s}_i/\|\boldsymbol{s}_i\|$ with l_2 norm error bounded in ϵ by using $O(r^4\lambda_{\min}^{-3}(C)\epsilon^{-2})$ copies of $\rho = |\boldsymbol{v}\rangle\langle\boldsymbol{v}|$ and $O(r^4\lambda_{\min}^{-3}(C)\epsilon^{-2})$ queries to input oracles.*

PROOF. Theorem 4.4 is derived by inserting Proposition 4.2 into Lemma 4.6 and considering the query complexity $r^2 O(\epsilon_1^{-1}) = O(r^{7/2}\lambda_{\min}^{-3/2}(C)\epsilon^{-1})$ for estimating the matrix C_r, such that each element c_{ij} has the error bound ϵ_1. □

For the negative curvature finding problem where the target state $|\boldsymbol{v}\rangle = |\boldsymbol{u}_t\rangle$, the time complexity to generate $|\boldsymbol{u}_t\rangle$ is $O(T_H\|H\|_F^3\alpha^{-2}\mathrm{polylog}(d)\epsilon^{-1})$ as proposed in Theorem 3.3. We could derive Corollary 4.1 from Theorem 4.4.

COROLLARY 4.1. *The classical description $\boldsymbol{u}_t = \sum_{i=1}^{r} x_i \boldsymbol{s}_i/\|\boldsymbol{s}_i\|$ for the NCF problem could be presented in time $O(T_H\mathrm{polylog}(d)r^4\|H\|_F^3\alpha^{-2}\lambda_{\min}^{-3}(C_r)\epsilon^{-3})$ with l_2 norm error bounds in ϵ, when the complete basis set $\{\boldsymbol{s}_j\}_{j=1}^{r}$ is given.*

By Theorem 3.2, the time complexity to label the proper eigenvalue is:

$$T_1 = T_H O(\|H\|_F^5 \alpha^{-4} \text{polylog}(d)\epsilon^{-1}).$$

By Theorem 4.3, the time complexity to generate the complete basis set is:

$$T_2 = T_H O(r^3 \|H\|_F^2 \lambda_{\min}^{-2}(H)\lambda_{\min}^{-1/2}(C_{r-1})(\|H\|_F^2 \lambda_{\min}^{-2}(H) + r^{2.5}\lambda_{\min}^{-1}(C_{r-1}))).$$

By considering the upper bound $\|H\|_F \leq \sqrt{r}L$ (Lemma 3.1) and the time complexity for reading-out the state $|\boldsymbol{u}_t\rangle$ (Corollary 4.1):

$$T_3 = T_H O(\text{polylog}(d) r^4 \|H\|_F^3 \alpha^{-2} \lambda_{\min}^{-3}(C_r)\epsilon^{-3}),$$

we could present the time complexity of solving the NCF problem:

$$\begin{aligned} T &= T_1 + T_2 + T_3 \\ &\leq T_H O(r^{6.5} L^5 \text{polylog}(d)(\lambda_{\min}^{-4}(H)\lambda_{\min}^{-3/2}(C_{r-1}) + \alpha^{-4}\lambda_{\min}^{-3}(C_r)\epsilon^{-3})), \end{aligned}$$

by providing the target vector in the form $\boldsymbol{u}_t = \sum_{i=1}^r x_i \boldsymbol{h}_{g(i)}/\|\boldsymbol{h}_{g(i)}\|$ with error bounded in ϵ or making the non-vector statement.

4.3 Numerical Simulation

Our numerical simulation contains two parts. First, we do simulation for the complete basis selection algorithm (Algorithm 5), to check the behavior of the eigenvalues from the gram matrix of the basis. Then we check the read-out behavior of Algorithm 6 with the basis selected by Algorithm 5, with a comparison to sampling-based quantum-inspired algorithms [35, 36].

4.3.1 CBS Basis

Our aim in this section is to check the practical performance of the basis selected by the CBS Algorithm. The Hessian is initialized as a 20000×20000 matrix $H = \sum_{i=1}^r \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T$ for different rank $r \in \{5, 10, 20, 30, 40\}$, and $\lambda_i =$

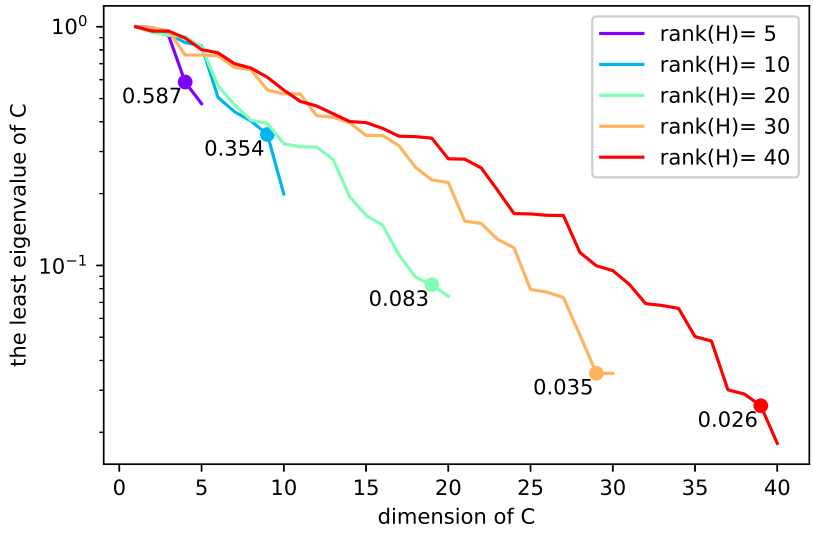

FIGURE 4.3: $\lambda_{\min}(C_l)$ for different $l \in [r]$ and different $r \in \{5, 10, 20, 30, 40\}$.

$(-1)^{i-1}(19+i), \forall i \in [r]$ for a fixed rank. For the convenience of the simulation, each operation $R_m = I - 2|\boldsymbol{t}_m\rangle\langle\boldsymbol{t}_m|$ is performed with random error 0.01. By Theorem 4.3, a basis with larger $\lambda_{\min}(C_{r-1})$ implies a smaller time complexity for Algorithm 5, where C_l is the gram matrix of $\{|\boldsymbol{s}_j\rangle\}_{j=1}^{l}, \forall l \in [r]$. We perform the i-th iteration in Algorithm 5 for 10 times, $\forall i \in [r]$, and choose the new basis row with the largest $\lambda_{\min}(C_i)$. We illustrate the value $\lambda_{\min}(C_l)$ for different $l = 1, 2, \cdots, r$ in Fig 4.3, where the rank is chosen from $\{5, 10, 20, 30, 40\}$. The value $\lambda_{\min}(C_{r-1})$ is shown to be roughly lower bounded by $1/r$, which implies the efficiency of the complete basis selection algorithm.

4.3.2 Read-out Error

In this section, we check the read-out behavior of Algorithm 6 and simulate the sampling-based quantum-inspired algorithms [35, 36] for comparison. Hessian H is initialized same with Section4.3.1 for rank $r \in \{5, 10, 20, 40\}$. The test state here for read-out corresponds to the eigenvector with the smallest $|\lambda_i|$. We generate the CBS basis with 10 counterparts, and choose the basis with the

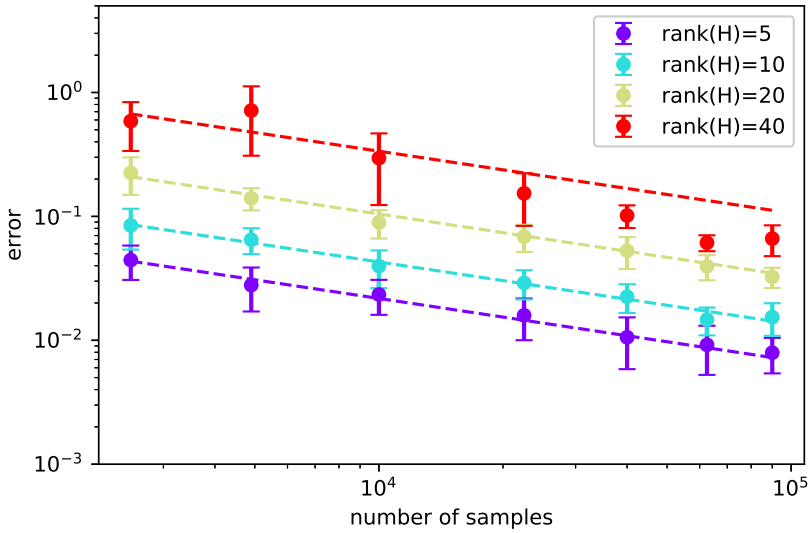

FIGURE 4.4: Read-out error for different sample numbers and rank.

largest $\lambda_{\min}(C_r)$. For fixed rank, each parameter $c_{ij} = \langle \boldsymbol{s}_i | \boldsymbol{s}_j \rangle$ or $a_i = \langle \boldsymbol{v} | \boldsymbol{s}_i \rangle$ is estimated with sampling number (n_1, n_2), where $n_1 \in \{50, 70, 100, 150, 200, 250, 300\}$ and $n_2 = n_1^2$. The required copies of the unknown quantum state is $n_\rho = O(rn_2)$,and the required queries to input oracles for Algorithm 6 is $n_{\text{oracle}} = O(r^2 n_1 + rn_2)$. Read-out error is defined as the l_2-norm distance between the exact state (vector) and the approximate state (vector). We illustrate the read-out error for different sample numbers n_2 and different rank in Fig 4.4. Dashed lines here denote $f(x) = Cx^{-1/2}$ curve fitting functions in the standard coordinate system, and error bars denote the standard deviation for 20 repetitions of the readout protocol. We draw all data in the log-log coordinate figure to better show the relationship $\epsilon \propto n_2^{-1/2}$, which coincides with Theorem 4.4. For a fixed sampling number, the read-out error increases with a larger r, which also coincides with Theorem 4.4.

Now we briefly introduce sampling-based quantum-inspired algorithms. The first sampling-based quantum-inspired algorithm was developed by E.Tang [35] for a quantum-inspired recommendation system. Several other quantum-inspired

algorithms were proposed after that, see [45, 46, 47, 48, 49, 50] for detail. All of these quantum-inspired algorithms use the approximate SVD designed by the FKV Algorithm [37] to achieve the claimed exponential speed-up on the dimension of input matrices.

Here we sketch the FKV Algorithm. For an input matrix $A \in \mathbb{R}^{m\times n}$ with unknown singular decomposition $A = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$, denote the i-th row vector as A_i and the j-th column vector as $A_{\cdot,j}$. Then the FKV Algorithm constructs a classical sampling oracle with l_2 norm distribution over rows: $p(i) = \frac{\|A_i\|^2}{\|A\|_F^2}$, and similar sampling oracles for every row A_i, with l_2 norm distribution $p_i(j) = \frac{A_{ij}^2}{\|A_i\|^2}$. These oracles could be constructed by the QRAM of H. The FKV Algorithm proceeds as follows:

(1) Sample t rows from the distribution $p(i)$. Denote sampled indices as $i_1, \cdots, i_t$.
(2) Renormalize the row as $R_s = \frac{\|A\|_F}{\sqrt{t}\|A_{i_s}\|} A_{t_s}, \forall s \in [t]$.
(3) Construct a new $t \times n$ matrix R from R_s.
(4) Select an index s from $[t]$ with uniform distribution.
(5) Sample a column index j with distribution $p_{i_s}(j)$.
(6) Repeat the procedure (4)-(5) for c times. Denote the selected column index as $j_1, \cdots, j_c$.
(7) Renormalize the column $B_{\cdot,q} = \frac{\|A\|_F}{\sqrt{c}\|R_{\cdot,j_q}\|}, \forall q \in [c]$.
(8) Construct a new $t \times c$ matrix B from $B_{\cdot,q}$.
(9) Do SVD on matrix B. Denote $\{\tilde{\sigma}_i\}$ as singular values of B. Denote $\{\boldsymbol{w}_i\}$ as left singular vectors of B.
(10) The right singular vector of A could be approximated as $\tilde{\boldsymbol{v}}_i = \frac{1}{\tilde{\sigma}_i} R^T \boldsymbol{w}_i$. The left singular vector of A could be approximated as $\tilde{\boldsymbol{u}}_i = \frac{1}{\tilde{\sigma}_i^2} A R^T \boldsymbol{w}_i$.

The accuracy of approximate singular values has been shown in previous work, see [36] for detail. However, the accuracy of approximate singular vectors has not been discussed yet. Remark that by the step (10) of FKV Algorithm, the

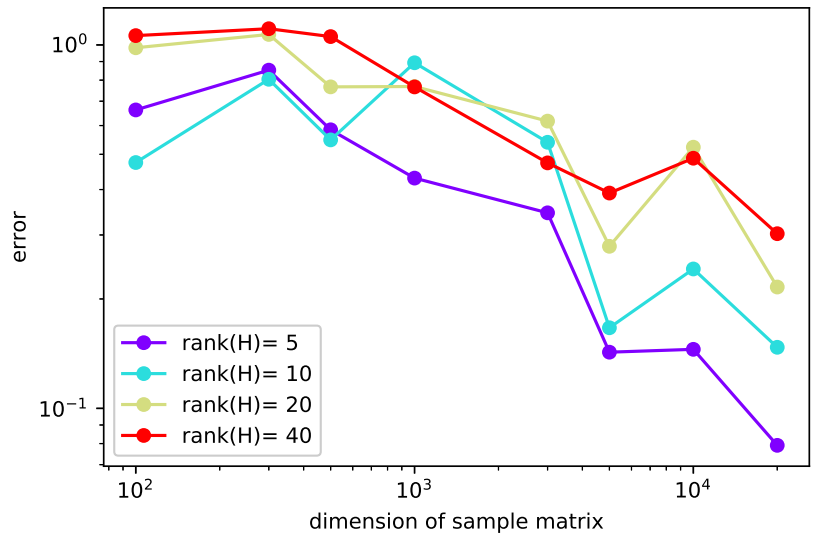

FIGURE 4.5: The error of the FKV Algorithm.

(approximate) right singular vector is written as the linear sum of t rows of A, which shows some point of similarity to our state read-out protocol, so we perform the FKV Algorithm with matrix H initialized the same as in previous experiments. We illustrate the result in Fig 4.5. The error here denotes the l_2 norm distance between the exact and the approximate singular vector. Different choice of $t = c \in \{100, 300, 500, 1000, 3000, 5000, 10000, 20000\}$ and different rank $r \in \{5, 10, 20, 40\}$ are considered. Based on the experiment result, the error decreases roughly when the dimension of the sample matrix increase. However, to achieve a relatively small error, say, less than 0.1, the FKV Algorithm needs to sample rows and columns with numbers the same as the dimension of H, and then a $O(d^3)$ time for SVD is required. Thus the speed-up could hardly maintain. Besides, the result shows the terrible behavior of the error when the rank of H slightly increases. Compare to the FKV Algorithm, our read-out protocol (Fig 4.3.2) shows better performance on the read-out accuracy and the required time complexity.

CHAPTER 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 $\tilde{O}(\mathrm{poly}(r, \log d)\epsilon^{-1})$ with probability $1 - 1/\mathrm{poly}(d)$, 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 $\tilde{O}(\mathrm{poly}(r, \log d)\epsilon^{-3})$, which is exponentially faster on the degree of d than existing general quantum state read-out methods.

CHAPTER 6

Appendix

The proof of Lemma 3.1:

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

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

By the definition of the Hessian matrix, for unit vector $\boldsymbol{v}$, we have:

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

From above equation, we can obtain:

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

and:

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

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

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

□

LEMMA 6.1. **Hoeffding's inequality[51]**

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:*

$$P(\overline{X} - E[\overline{X}] \geq \epsilon) \leq \exp\left(-\frac{2n^2\epsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right), \tag{6.1}$$

and

$$P(\overline{X} - E[\overline{X}] \leq -\epsilon) \leq \exp\left(-\frac{2n^2\epsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right). \tag{6.2}$$

The proof of Lemma 4.5:

PROOF. Define the index function $g : [r] \to [d]$ such that $\boldsymbol{s}_i = \boldsymbol{h}_{g(i)}, \forall i \in [r]$. Consider the eigen-decomposition of matrix H:

$$H = \sum_{j=1}^{r} \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^T. \tag{6.3}$$

It is natural to generate the decomposition:

$$\boldsymbol{h}_j = \sum_{i=1}^{r} \lambda_i \boldsymbol{u}_i u_i^{(j)}, \tag{6.4}$$

$$h_{jk} = \sum_{i=1}^{r} \lambda_i u_i^{(j)} u_i^{(k)}. \tag{6.5}$$

Define the $r \times r$ dimensional matrix:

$$C = (\boldsymbol{h}_{g(1)}^T, \boldsymbol{h}_{g(2)}^T, \cdots, \boldsymbol{h}_{g(r)}^T)^T (\boldsymbol{h}_{g(1)}, \boldsymbol{h}_{g(2)}, \cdots, \boldsymbol{h}_{g(r)}). \tag{6.6}$$

There is:

$$\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r} \; is \; linear \; independent \Leftrightarrow det(C) \neq 0. \tag{6.7}$$

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

$$det(C) = \begin{vmatrix} \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(1))} u_i^{(g(1))} & \cdots & \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(1))} u_i^{(g(r))} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(r))} u_i^{(g(1))} & \cdots & \sum_{i=1}^{r} \lambda_i^2 u_i^{(g(r))} u_i^{(g(r))} \end{vmatrix} \tag{6.8}$$

$$= \sum_{i_1=1}^{r}\sum_{i_2=1}^{r}\cdots\sum_{i_r=1}^{r}\begin{vmatrix} \lambda_{i_1}^2 u_{i_1}^{(g(1))}u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r}^2 u_{i_r}^{(g(1))}u_{i_r}^{(g(r))} \\ \vdots & \ddots & \vdots \\ \lambda_{i_1}^2 u_{i_1}^{(g(r))}u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r}^2 u_{i_r}^{(g(r))}u_{i_r}^{(g(r))} \end{vmatrix} \tag{6.9}$$

$$= \sum_{i_1=1}^{r}\sum_{i_2=1}^{r}\cdots\sum_{i_r=1}^{r}(\prod_{j=1}^{r}\lambda_{i_j}^2)(\prod_{j=1}^{r}u_{i_j}^{(g(j))})\begin{vmatrix} u_{i_1}^{(g(1))} & \cdots & u_{i_r}^{(g(1))} \\ \vdots & \ddots & \vdots \\ u_{i_1}^{(g(r))} & \cdots & u_{i_r}^{(g(r))} \end{vmatrix}. \tag{6.10}$$

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

$$det(H') = \begin{vmatrix} \sum_{i=1}^{r}\lambda_i u_i^{(g(1))}u_i^{(g(1))} & \cdots & \sum_{i=1}^{r}\lambda_i u_i^{(g(1))}u_i^{(g(r))} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{r}\lambda_i u_i^{(g(r))}u_i^{(g(1))} & \cdots & \sum_{i=1}^{r}\lambda_i u_i^{(g(r))}u_i^{(g(r))} \end{vmatrix} \tag{6.11}$$

$$= \sum_{i_1=1}^{r}\sum_{i_2=1}^{r}\cdots\sum_{i_r=1}^{r}\begin{vmatrix} \lambda_{i_1} u_{i_1}^{(g(1))}u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r} u_{i_r}^{(g(1))}u_{i_r}^{(g(r))} \\ \vdots & \ddots & \vdots \\ \lambda_{i_1} u_{i_1}^{(g(r))}u_{i_1}^{(g(1))} & \cdots & \lambda_{i_r} u_{i_r}^{(g(r))}u_{i_r}^{(g(r))} \end{vmatrix} \tag{6.12}$$

$$= \sum_{i_1=1}^{r}\sum_{i_2=1}^{r}\cdots\sum_{i_r=1}^{r}(\prod_{j=1}^{r}\lambda_{i_j})(\prod_{j=1}^{r}u_{i_j}^{(g(j))})\begin{vmatrix} u_{i_1}^{(g(1))} & \cdots & u_{i_r}^{(g(1))} \\ \vdots & \ddots & \vdots \\ u_{i_1}^{(g(r))} & \cdots & u_{i_r}^{(g(r))} \end{vmatrix}. \tag{6.13}$$

Note that the determinant in eq(6.10) and eq(6.13) 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:

$$det(C)/\prod_{i=1}^{r}\lambda_i^2 = det(H')/\prod_{i=1}^{r}\lambda_i \tag{6.14}$$

Thus the problem about whether group $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$ is linear 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 $O(r^3)$ time[52]. We could claim

that the group $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$ is linear independent if $det(H') \neq 0$, or $\{\boldsymbol{h}_{g(i)}\}_{i=1}^{r}$ is linear dependent if $det(H') = 0$. □

Bibliography

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

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

[3] 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*, pages 1195–1199. ACM, 2017.

[4] 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*, pages 2933–2941, 2014.

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

[6] 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*, pages 1724–1732. JMLR. org, 2017.

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

[8] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical review letters*,

113(13):130503, 2014.

[9] 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.

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

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

[12] Scott Aaronson. Read the fine print. *Nature Physics*, 11(4):291, 2015.

[13] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points − online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

[14] 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.

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

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

[17] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, Jun 1982.

[18] Thomas Barthel and Yikang Zhang. Optimized lie-trotter-suzuki decompositions for two and three non-commuting terms, 2019.

[19] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. Simulating hamiltonian dynamics with a truncated taylor series. *Phys. Rev. Lett.*, 114:090502, Mar 2015.

[20] Andrew M Childs, Richard Cleve, Enrico Deotto, Edward Farhi, Sam Gutmann, and Daniel A Spielman. Exponential algorithmic speedup by a quantum walk. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 59–68. ACM, 2003.

[21] Guang Hao Low and Isaac L Chuang. Optimal hamiltonian simulation by quantum signal processing. *Physical review letters*, 118(1):010501, 2017.

[22] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[23] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

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

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

[26] David Gross, Yi-Kai Liu, Steven T. Flammia, Stephen Becker, and Jens Eisert. Quantum state tomography via compressed sensing. *Phys. Rev. Lett.*, 105:150401, Oct 2010.

[27] Anastasios Kyrillidis, Amir Kalev, Dohyung Park, Srinadh Bhojanapalli, Constantine Caramanis, and Sujay Sanghavi. Provable compressed sensing quantum state tomography via non-convex methods. *npj Quantum Information*, 4(1):36, 2018.

[28] Jeongwan Haah, Aram W Harrow, Zhengfeng Ji, Xiaodi Wu, and Nengkun Yu. Sample-optimal tomography of quantum states. *IEEE Transactions on Information Theory*, 63(9):5628–5641, 2017.

[29] Ryan O'Donnell and John Wright. Efficient quantum tomography. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 899–912. ACM, 2016.

[30] Marcus Cramer, Martin B Plenio, Steven T Flammia, Rolando Somma, David Gross, Stephen D Bartlett, Olivier Landon-Cardinal, David Poulin,

and Yi-Kai Liu. Efficient quantum state tomography. *Nature communications*, 1:149, 2010.

[31] Tao Xin, Dawei Lu, Joel Klassen, Nengkun Yu, Zhengfeng Ji, Jianxin Chen, Xian Ma, Guilu Long, Bei Zeng, and Raymond Laflamme. Quantum state tomography via reduced density matrices. *Physical review letters*, 118(2):020401, 2017.

[32] Hartmut Häffner, Wolfgang Hänsel, CF Roos, Jan Benhelm, Michael Chwalla, Timo Körber, UD Rapol, Mark Riebe, PO Schmidt, Christoph Becher, et al. Scalable multiparticle entanglement of trapped ions. *Nature*, 438(7068):643, 2005.

[33] M Riebe, K Kim, P Schindler, T Monz, PO Schmidt, TK Körber, W Hänsel, H Häffner, CF Roos, and R Blatt. Process tomography of ion trap quantum gates. *Physical review letters*, 97(22):220407, 2006.

[34] Alexander I Lvovsky and Michael G Raymer. Continuous-variable optical quantum-state tomography. *Reviews of Modern Physics*, 81(1):299, 2009.

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

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

[37] Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *J. ACM*, 51(6):1025–1041, November 2004.

[38] MR Vanner, M Aspelmeyer, and MS Kim. Quantum state orthogonalization and a toolset for quantum optomechanical phonon control. *Physical review letters*, 110(1):010504, 2013.

[39] M Ježek, M Mičuda, I Straka, M Mikova, M Dušek, and J Fiurášek. Orthogonalization of partly unknown quantum states. *Physical Review A*, 89(4):042316, 2014.

[40] Antonio S. Coelho, Luca S. Costanzo, Alessandro Zavatta, Catherine Hughes, M.S. Kim, and Marco Bellini. Universal continuous-variable state orthogonalizer and qubit generator. *Physical Review Letters*, 116(11), Mar 2016.

[41] Hans Havlicek and Karl Svozil. Dimensional lifting through the generalized gramschmidt process. *Entropy*, 20(4):284, Apr 2018.

[42] Andrew M Childs and Nathan Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *arXiv preprint arXiv:1202.5822*, 2012.

[43] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum algorithms for supervised and unsupervised machine learning. *arXiv preprint arXiv:1307.0411*, 2013.

[44] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Quantum Computation and Information*, page 5374, 2002.

[45] András Gilyén, Seth Lloyd, and Ewin Tang. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension, 2018.

[46] Nai-Hui Chia, Han-Hsuan Lin, and Chunhao Wang. Quantum-inspired sublinear classical algorithms for solving low-rank linear systems, 2018.

[47] Dhawal Jethwani, François Le Gall, and Sanjay K. Singh. Quantum-inspired classical algorithms for singular value transformation, 2019.

[48] Zhihuai Chen, Yinan Li, Xiaoming Sun, Pei Yuan, and Jialin Zhang. A quantum-inspired classical algorithm for separable non-negative matrix factorization. *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, Aug 2019.

[49] Fabio L. Traversa. Aircraft loading optimization: Memcomputing the 5th airbus problem, 2019.

[50] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. A quantum-inspired algorithm for general minimum conical hull problems, 2019.

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

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