Quantum Algorithms with Tensor Approach

Lejia Gu

PhD

The Hong Kong Polytechnic University

2020

The Hong Kong Polytechnic University
Department of Applied Mathematics

Quantum Algorithms with Tensor Approach

Lejia Gu

A thesis submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

June 2020

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

GU Lejia (Name of student)

Abstract

Recently quantum computing becomes more and more popular and realizable, and tensor is an effective method in quantum computing. This thesis is devoted to studying structured tensors and providing several quantum algorithms. Three topics are included:

1. introducing a new subclass of Hankel tensors and verifying their properties;
2. designing two quantum algorithms for higher order singular value decomposition;
3. presenting two quantum algorithms for polynomial optimization.

For the first topic, a subclass of Hankel tensors called basic positive semi-definite (PSD) Hankel tensors is introduced, and the purpose is to find some low-rank basic PSD non-strong Hankel tensors. It is shown that rank-1 even order strong Hankel tensors are equivalent to rank-1 basic PSD Hankel tensors, and all even order strong Hankel tensors with rank larger than 1 can be expressed as the sum of rank-1 basic PSD Hankel tensors. Thus, the study of non-strong PSD Hankel tensors is reduced to the study of basic PSD Hankel tensors with rank larger than 1. It is proved that (i) there are no rank-2 basic PSD Hankel tensors, (ii) rank-3 basic PSD Hankel tensors with dimension no less than 3 do not exist. Furthermore, an example of basic PSD Hankel tensor whose rank equals 3 or 4 is provided.

For the second topic, higher order singular value decomposition (HOSVD) is

studied, as it is a vital method for analyzing big data in multilinear algebra and machine learning. We present two quantum algorithms for HOSVD. The methods allow one to decompose a tensor into a core tensor containing tensor singular values and some unitary matrices by quantum computers. Compared to the classical HOSVD algorithm, our quantum algorithms provide an exponential speedup. Furthermore, a hybrid quantum-classical algorithm of HOSVD model applied in recommendation systems is introduced.

For the last topic, the quantum version of Barzilai-Borwein (BB) gradient method is proposed and applied to polynomial optimization with a unit norm constraint. It is known that gradient methods are widely used in optimization and machine learning problems. However, standard gradient descent method usually converges very slowly while BB gradient method overcomes this obstacle with nearly no more cost. Our quantum algorithms scale polylogarithmically in the dimension of solution vector. Compared with the classical counterpart, our quantum methods provide an exponential speedup, and succeed to find the optimal value in fewer iterations than the existing quantum gradient methods.

Acknowledgements

First and foremost, I would like to express my sincerest gratitude to my supervisor Dr. Guofeng Zhang for his irradiative guidance, warm encouragement, and invaluable advice throughout my Ph.D. period. Dr. Zhang not only helps me with research, but also sets an example to me with his rigorous and assiduous attitude. Then, I would like to thank my co-supervisor Prof. Liqun Qi, also my supervisor during my master's period, who brought me to the world of scientific research. Prof. Qi's indefatigable spirit towards scientific exploration will encourage me all over my life. I am very honoured to have two great supervisors.

Next, I want to thank my junior and research partner, Ms. Xiaoqiang Wang. I really enjoy the time spent with Xiaoqiang, and learn a lot in every discussion. Then, I would like to express my deep gratitude to my seniors Dr. Zhiyuan Dong and Dr. Weiyang Ding for their careful guidance and kindly assistance. I also want to give my gratitude to all the professors, seniors, and supporting staffs who help me during my research period. The following list is by no means complete and in no particular order: Prof. Yimin Wei, Prof. Zhenghai Huang, Prof. Haidong Yuan, Prof. Yu Pan, Prof. Xiaoting Wang, Dr. Zaikun Zhang; Prof. Shenglong Hu, Dr. Yi Xu, Dr. Zhongming Chen, Dr. Yannan Chen, Dr. Haibin Chen, Dr. Qun Wang, Dr. Jingya Chang, Dr. Chen Ouyang, Dr. Jinjie Liu, Dr. Jinghao Li.

Last but not least, I want to thank my parents, Peixin Gu and Jie Jin, and my grandparents for their support, understanding and unconditional love throughout

my life.

Contents

List of Figures

List of Tables

List of Algorithms

List of Notations

$\mathbb{R}, \mathbb{R}^+$	the set of real numbers, the set of positive real numbers
$\mathbb{C}$	the set of complex numbers
$a, b, \ldots; A, B, \ldots$	scalar
$\mathbf{a}, \mathbf{b}, \ldots$	vector
$\mathbf{A}, \mathbf{B}, \ldots$	matrix or operator
$\mathcal{A}, \mathcal{B}, \ldots$	tensor
$T^{\mathbb{R}}_{m,n}/T^{\mathbb{C}}_{m,n}$	the set of all the real/complex mth-order n-dimensional tensors
$S_{m,n}$	the set of all the real symmetric mth-order n-dimensional tensors
$\lvert\cdot\rangle$	ket, dirac notation of a column vector in quantum mechanics
$\langle\cdot\rvert$	bra, dirac notation of a row vector in quantum mechanics
$[m]$	a set $\{1, 2, \cdots, m\}$
$O(f(n))$	big O notation, which indicates a running time with upper bound $cf(n)$ for a fixed $c \in \mathbb{R}^+$ and sufficiently large positive n
$\tilde{O}(f(n))$	$O(f(n))$ times a polylogarithmic factor

$\times_k$	k-mode tensor-matrix multiplication
$\cdot$ or $\langle \cdot, \cdot \rangle$	inner product
$\circ$	outer product
$\otimes$	Kronecker product

Chapter 1

Introduction

1.1 Background

Quantum computing has caught more people's eyes in the recent years, as classical computing is reaching its limit. Quantum computing uses quantum-mechanical phenomena for computation, such as superposition and entanglement. The device that performs such computation is called the quantum computer. Last year, Google invented a 53-qubit quantum computer and claimed that it outperforms the current best supercomputer when running random quantum circuits, which is named as "quantum supremacy" or "quantum advantage" [3]. This milestone will speed up the development of quantum computing.

As the quantum physicists are working on setting up the quantum computers, theoretical research on quantum algorithms and programming is also being conducted. The concept of quantum computing was proposed in the early 1980s. In 1982, the famous theoretical physicist, Richard Feynman, claimed that the real world is quantum-mechanical. He pointed out that a quantum computer may simulate things that a classical computer could not [26]. In 1994, Peter Shor designed a quantum algorithm for factoring integers, which is exponentially faster than the corresponding classical algorithm [70]. This algorithm is able to break the current cryptography system if the large-scale quantum computer is built. Shor's algorithm made scientists

convinced that quantum computing has a great potential. In 1996, Grover developed a quantum algorithm that searches an unstructured database for an entry quadratically faster than the classical counterparts [30]. In 2009, the HHL algorithm, named after Harrow, Hassidim and Lloyd, solves a linear system exponentially faster than the classical counterpart. During the last decade, many quantum machine learning [10] algorithms are proposed, such as quantum support vector machine [65], quantum Boltzmann machine [2] and etc.

Quantum algorithms are usually implemented by quantum circuits, which are based on quantum bits and logic gates. All the quantum bits and logic gates can be considered as tensors, thus composing a tensor network. Tensor network is a graphical representation of tensors, and it is an efficient method to simulate quantum circuits, which is applied by many researchers such as IBM and Google teams [57, 3]. Besides that, a multipartite quantum state corresponds to a tensor (or hypermatrix). Therefore, the entanglement of a multipartite quantum state is studied by the spectral properties of tensors [34, 35, 63].

In this thesis, I focus on the quantum algorithms related to tensors. Two quantum algorithms for higher order singular value decomposition are proposed, and two quantum algorithms for BB gradient method are designed for polynomial optimizations. Also, a subclass of Hankel tensors is introduced to study the property of structured tensors.

1.2 Organization of the Thesis

The rest of the thesis is organized as follows.

In Chapter 2, some concepts in quantum computing, the definitions and multiplications related to tensors are introduced. Moreover, some well-known tensor decompositions and tensor networks are illustrated.

In Chapter 3, we discuss the quantum operations on tensors. The data structure of tensors with quantum access is proposed. Then, the quantum matrix unfolding is designed.

In Chapter 4, Hankel tensors are first introduced. Then, the definition of basic positive semi-definite (PSD) Hankel tensor is proposed and the relationships between strong Hankel tensors and basic PSD Hankel tensors are also given. After that, it is proved that within mth-order n-dimensional PSD Hankel tensors, rank-2 basic PSD Hankel tensors do not exist. Moreover, the existence of rank-3 basic PSD Hankel tensors whose dimensions are not smaller than 3 is disproved. Finally, a 4th-order 2-dimensional basic PSD Hankel tensor whose rank is 3 or 4 is presented.

In Chapter 5, the definition and properties of higher order singular value decomposition (HOSVD) are introduced. Two quantum higher order singular value decomposition algorithms are presented, and the computational complexity is discussed then. After that, we give an application of HOSVD model on quantum recommendation systems. At last, we summarize the results and compare the quantum HOSVD algorithm with the classical counterpart.

In Chapter 6, the classical Barzilai-Borwein (BB) gradient method is introduced, and the detailed polynomial optimization problem is stated. After that, two kinds of quantum BB gradient methods are described. Then, the convergence and computational complexity are discussed, and the numerical results are listed and compared with the quantum standard gradient descent method. Eventually, the quantum BB algorithms are confronted with the classical counterparts.

In the last chapter, conclusions and some future work are discussed.

Chapter 2

Preliminaries

2.1 Quantum Computing

In this section, some concepts in quantum computing are introduced, which can be found in [51].

Definition 2.1. (Quantum state)

The state space of a quantum system is a Hilbert space. The system is completely described by its state vector, which is a unit vector in the Hilbert space.

Definition 2.2. (Quantum bit (Qubit))

The state space is $\mathbb{C}^2$ *with basis vectors*

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \equiv |0\rangle, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \equiv |1\rangle. \tag{2.1}$$

An arbitrary state vector is given by

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle, \quad |\alpha|^2 + |\beta|^2 = 1. \tag{2.2}$$

Quantum states are represented by Dirac notation (or called bra-ket notation). The conjugate transpose of ket $|\psi\rangle$, denoted by $\langle\psi|$, is a row vector.

The ket $|\psi\rangle$ is a pure state. A quantum system may also be in a mixed state,

usually characterized by the **density matrix**

$$\rho = \sum_{j=0}^{M-1} p_j \left|\psi_j\right\rangle\left\langle\psi_j\right|, \quad p_j > 0, \quad \sum_j p_j = 1. \tag{2.3}$$

The density matrix is positive semi-definite and its trace is 1.

Definition 2.3. (Composite system)

The state space of the composite system, composed of subsystems A and B, is the tensor product Hilbert space $H_A \otimes H_B$. If the state vectors of A and B are $|\psi_A\rangle$ and $|\psi_B\rangle$ respectively, then the state vector of the composite system is

$$\left|\psi_A\right\rangle \otimes \left|\psi_B\right\rangle \equiv \left|\psi_A\psi_B\right\rangle, \tag{2.4}$$

where tensor product $\otimes$ refers to Kronecker product.

An **entangled state** is a state which cannot be written as a **product state** $|\psi_A\rangle \otimes |\psi_B\rangle$.

A quantum circuit consists of qubits and logic gates. Quantum **logic gates** are local unitary operations that act on a small number of qubits. A unitary operator is a bounded linear operator $\mathbf{U} : H \to H$ on a Hilbert space H that satisfies $\mathbf{U}^\dagger\mathbf{U} = \mathbf{U}\mathbf{U}^\dagger = \mathbf{I}$, where $\mathbf{U}^\dagger$ is the conjugate transpose of $\mathbf{U}$.

Definition 2.4. (Quantum measurement)

Quantum measurements are described by a collection $\{\mathbf{M}_m\}$ of measurement operators. The index m refers to the measurement outcome that may occur in the experiment.

If the state of the quantum system is $|\psi\rangle$ before the measurement, then the probability that result m occurs is

$$p(m) = \left\langle\psi\right| \mathbf{M}_m^\dagger \mathbf{M}_m \left|\psi\right\rangle, \tag{2.5}$$

and the state of the system immediately after measurement is

$$\frac{\mathbf{M}_m |\psi\rangle}{\sqrt{p(m)}}. \tag{2.6}$$

The operators satisfy the completeness condition: $\sum_m \mathbf{M}_m^\dagger \mathbf{M}_m = \mathbf{I}$.

Definition 2.5. (Partial trace)

Suppose there is a bipartite system, whose state is described by a density operator ρ. The reduced density operator for the first subsystem is defined by

$$\rho_1 = \mathrm{tr}_2(\rho), \tag{2.7}$$

where tr_2 is a map of operators known as the partial trace over the second subsystem. The partial trace is defined as

$$\begin{aligned} \mathrm{tr}_2 \left(|a_1\rangle\langle a_2| \otimes |b_1\rangle\langle b_2| \right) &= |a_1\rangle\langle a_2| \, \mathrm{tr}(|b_1\rangle\langle b_2|) \\ &= \langle b_2|b_1\rangle \, |a_1\rangle\langle a_2| \, , \end{aligned} \tag{2.8}$$

where $|a_1\rangle$ and $|a_2\rangle$ are two states in the first subsystem, $|b_1\rangle$ and $|b_2\rangle$ are two states in the second subsystem.

Recall that discrete Fourier transform (DFT) takes as input a vector of complex numbers, $x_0, \ldots, x_{N-1}$ where the length N of the vector is a fixed parameter. It outputs a vector of complex numbers $y_0, \ldots, y_{N-1}$, defined by

$$y_k \equiv \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{2\pi \mathrm{i} jk/N} x_j. \tag{2.9}$$

The quantum Fourier transform is exactly the same transformation and presented in the following lemma.

Lemma 2.1. *The* **quantum Fourier transform (QFT)** *on an orthonormal basis $\{|0\rangle, |1\rangle, \cdots, |N-1\rangle\}$ is defined to be a linear operator with the following action on*

the basis states,

$$|j\rangle \to \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi \mathrm{i} jk/N} |k\rangle . \tag{2.10}$$

By the above two equations, we have

$$\sum_{j=0}^{N-1} x_j |j\rangle \to \sum_{k=0}^{N-1} y_k |k\rangle . \tag{2.11}$$

Denote $N = 2^n$. For n bits, the complexity of DFT is $O(n2^n)$, while that of QFT is only $O(n^2)$, so that QFT provides an exponential speedup.

2.2 Definitions and Multiplications of Tensors

A tensor (or hypermatrix) is a multi-array. The definition and the multiplications related to tensors are given in this section. More details can be found in [62, 61].

Definition 2.6. *An mth-order* **tensor** $\mathcal{A} = (a_{i_1 \cdots i_m}) \in \mathbb{F}^{I_1 \times I_2 \times \cdots \times I_m}$ *is a multi-array of $\Pi_{j=1}^m I_j$ entries, where $i_j \in [I_j]$ for $j \in [m]$ and $\mathbb{F}$ is a field. $(I_1, I_2, \ldots, I_m)$ is the dimension of $\mathcal{A}$.*

The entries can be also represented as $\mathcal{A}(i_1, i_2, \ldots, i_m)$. When $I_1 = I_2 = \cdots = I_m = n$, $\mathcal{A}$ is called an mth-order n-dimensional tensor. Usually, we consider real and complex tensors, i.e., $\mathbb{F} = \mathbb{R}$ or $\mathbb{C}$. Denote the set of all the real (complex) mth-order n-dimensional tensors $T_{m,n}^{\mathbb{R}}(T_{m,n}^{\mathbb{C}})$.

For any tensor $\mathcal{A} \in T_{m,n}^{\mathbb{R}}$, if its entries $a_{i_1 \cdots i_m}$'s are invariant under any permutation of its indices, then $\mathcal{A}$ is called a **symmetric** tensor. Let the set of all the real symmetric mth-order n-dimensional tensors be $S_{m,n}$.

For $\mathcal{A} \in S_{m,n}$ and $\mathbf{x} \in \mathbb{R}^n$, we have a homogeneous polynomial $f(\mathbf{x})$ of n variables and degree m,

$$f(\mathbf{x}) = \mathcal{A} \cdot \mathbf{x}^{\circ m} := \mathcal{A}\mathbf{x}^m = \sum_{i_1, \cdots, i_m \in [n]} a_{i_1 \cdots i_m} x_{i_1} \cdots x_{i_m}. \tag{2.12}$$

Note that there is a one-to-one correspondence between homogeneous polynomials and symmetric tensors.

Definition 2.7. *For even order tensor $\mathcal{A} \in S_{m,n}$, if $f(\mathbf{x}) \geqslant 0$ for all $\mathbf{x} \in \mathbb{R}^n$, then the homogeneous polynomial $f(\mathbf{x})$ and symmetric tensor $\mathcal{A}$ are called* **positive semi-definite** *(PSD). If $f(\mathbf{x}) > 0$ for all nonzero $\mathbf{x} \in \mathbb{R}^n$, then $f(\mathbf{x})$ and $\mathcal{A}$ are called* **positive definite** *(PD).*

Definition 2.8. *Given a tensor $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$ and a matrix $\mathbf{B} \in \mathbb{C}^{J_k \times I_k}$, their k-mode* **tensor-matrix multiplication**

$$(\mathcal{A} \times_k \mathbf{B})_{i_1 i_2 \cdots i_{k-1} j_k i_{k+1} \cdots i_m} = \sum_{i_k=1}^{I_k} a_{i_1 i_2 \cdots i_{k-1} i_k i_{k+1} \cdots i_m} b_{j_k i_k} \tag{2.13}$$

produces an $I_1 \times I_2 \times \cdots \times I_{k-1} \times J_k \times I_{k+1} \times \cdots \times I_m$ tensor.

Definition 2.9. *The* **outer product** *of two tensors $\mathcal{A} = (a_{i_1 \cdots i_m}) \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$ and $\mathcal{B} = (b_{i_{m+1} \cdots i_p}) \in \mathbb{C}^{I_{m+1} \times \cdots \times I_p}$, denoted as $\mathcal{A} \circ \mathcal{B}$, is $(a_{i_1 \cdots i_m} b_{i_{m+1} \cdots i_p}) \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_p}$.*

Definition 2.10. *The* **inner product** *of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$, denoted as $\mathcal{A} \cdot \mathcal{B}$ or $\langle \mathcal{A}, \mathcal{B} \rangle$, is defined as*

$$\mathcal{A} \cdot \mathcal{B} = \sum_{i_1=1}^{I_1} \cdots \sum_{i_m=1}^{I_m} a^*_{i_1 i_2 \cdots i_m} b_{i_1 i_2 \cdots i_m}, \tag{2.14}$$

where $*$ is the conjugate.

Definition 2.11. *The induced norm $\sqrt{\mathcal{A} \cdot \mathcal{A}}$ is called the* **Frobenius norm** *of $\mathcal{A}$, denoted as $\|\mathcal{A}\|_F$. The l_1-norm of the tensor $\mathcal{A}$ is defined as $\|\mathcal{A}\|_1 = \sum_{i_1=1}^{I_1} \cdots \sum_{i_m=1}^{I_m} |a_{i_1 i_2 \cdots i_m}|$.*

2.3 Tensor Decompositions and Tensor Networks

It is known that matrices can be expressed as a sum of rank-1 matrices by singular value decomposition (SVD). The singular values are the principal components of the matrix, and we can compress the matrix by eliminating several smallest singular values. Due to the complex structure of tensors, there do not exist such powerful decompositions as matrix SVD. In the following, some well-known tensor decompositions modified from SVD are introduced.

Definition 2.12. (CP Decomposition) [14, 32]
For any tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_m}$, it has a tensor rank decomposition or CANDECOMP/PARAFAC (CP) decomposition that it may be represented with a suitably large r as a linear combination of r rank-1 tensors:

$$\mathcal{A} = \sum_{k=1}^{r} \alpha_k \mathbf{v}_k^1 \circ \cdots \circ \mathbf{v}_k^m, \tag{2.15}$$

with each $\alpha_k \in \mathbb{R}$, $\mathbf{v}_k^i \in \mathbb{R}^n$, for $i = 1, ..., m$.

We usually use r to define the **rank** of tensor $\mathcal{A}$.

If $\mathcal{A}$ is a symmetric tensor, then we have the following symmetric CP decomposition

$$\mathcal{A} = \sum_{k=1}^{r} a_k \underbrace{\mathbf{v}_k \circ \cdots \circ \mathbf{v}_k}_{m} = \sum_{k=1}^{r} a_k \mathbf{v}_k^{\circ m}. \tag{2.16}$$

Accordingly, the minimum of r is the **symmetric rank** of $\mathcal{A}$.

Definition 2.13. (Tensor-train Decomposition) [55]
For tensor $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$, by tensor-train (TT-) decomposition the entries can be approximated by a series of matrices

$$\mathcal{A}(i_1, i_2, \ldots, i_m) = \mathbf{G}_1(i_1)\mathbf{G}_2(i_2) \cdots \mathbf{G}_m(i_m), \tag{2.17}$$

where $\mathbf{G}_k(i_k)$ *is an* $r_{k-1} \times r_k$ *matrix, and the boundaries* $r_0 = r_m = 1$.

Actually, the matrix $\mathbf{G}_k(i_k)$ is a third-order tensor $\mathcal{G}$, then the decomposition is rewritten as

$$\mathcal{A}(i_1, i_2, \ldots, i_m) = \sum_{\alpha_0,\ldots,\alpha_{m-1},\alpha_m} \mathcal{G}_1(\alpha_0, i_1, \alpha_1)\, \mathcal{G}_2(\alpha_1, i_2, \alpha_2) \cdots \mathcal{G}_m(\alpha_{m-1}, i_m, \alpha_m), \tag{2.18}$$

where $\mathcal{G}_k$ is an $r_{k-1} \times I_k \times r_k$ tensor. The values r_k, $k \in [m-1]$ are called the TT-rank or bond dimension.

Tensor-train decomposition is a powerful tool for quantum mechanics, since its structure resembles a pure quantum state of several particles. It is also named as matrix product state (MPS), and a multipartite quantum state can be written in the following form:

$$|\Psi\rangle = \sum_{i_1,i_2,\ldots,i_m} \mathrm{tr}\left[\mathbf{G}_1(i_1)\mathbf{G}_2(i_2)\cdots\mathbf{G}_m(i_m)\right] |i_1 i_2 \cdots i_m\rangle. \tag{2.19}$$

Definition 2.14. (Tucker Decomposition) [78]

For tensor $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$, *if there exist matrices* $\mathbf{X}^{(k)} = [\mathbf{x}_1^{(k)} \mathbf{x}_2^{(k)} \cdots \mathbf{x}_{I_k}^{(k)}] \in \mathbb{C}^{I_k \times I_k}$ *with* $\|\mathbf{x}_{i_k}^{(k)}\| = 1$ *for* $k \in [m]$ *and* $i_k \in [I_k]$ *such that*

$$\mathcal{A} = \mathcal{S} \times_1 \mathbf{X}^{(1)} \times_2 \mathbf{X}^{(2)} \times_3 \cdots \times_m \mathbf{X}^{(m)}, \tag{2.20}$$

then (2.20) is said to be a Tucker decomposition of $\mathcal{A}$, *and* $\mathcal{S} = (s_{i_1 i_2 \cdots i_m})$ *is called the core tensor of* $\mathcal{A}$.

If we remain the first r_k columns of $\mathbf{X}^{(k)}$, i.e., $\mathbf{X}^{(k)} = [\mathbf{x}_1^{(k)} \mathbf{x}_2^{(k)} \cdots \mathbf{x}_{r_k}^{(k)}] \in \mathbb{C}^{I_k \times r_k}$, then core tensor $\mathcal{S} \in \mathbb{C}^{r_1 \times r_2 \times \cdots \times r_m}$. This is called the truncated Tucker Decomposition [43].

Definition 2.15. (Hierarchical Tucker Decomposition) [31]

The tensor $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$ *is decomposed into a multi-layer of tensors through*

Hierarchical Tucker (HT) decomposition by

$$
\begin{aligned}
\phi^{1,j,\gamma} &= \sum_{\alpha=1}^{r_0} a_{\alpha}^{1,j,\gamma} \mathbf{x}^{0,2j-1,\alpha} \circ \mathbf{x}^{0,2j,\alpha} \\
&\vdots \\
\phi^{l,j,\gamma} &= \sum_{\alpha=1}^{r_{l-1}} a_{\alpha}^{l,j,\gamma} \underbrace{\phi^{l-1,2j-1,\alpha}}_{\text{order } 2^{l-1}} \circ \underbrace{\phi^{l-1,2j,\alpha}}_{\text{order } 2^{l-1}} \\
&\vdots \\
\phi^{L-1,j,\gamma} &= \sum_{\alpha=1}^{r_{L-2}} a_{\alpha}^{L-1,j,\gamma} \underbrace{\phi^{L-2,2j-1,\alpha}}_{\text{order } \frac{m}{4}} \circ \underbrace{\phi^{L-2,2j,\alpha}}_{\text{order } \frac{m}{4}} \\
\mathcal{A} &= \sum_{\alpha=1}^{r_{L-1}} a_{\alpha}^{L} \underbrace{\phi^{L-1,1,\alpha}}_{\text{order } \frac{m}{2}} \circ \underbrace{\phi^{L-1,2,\alpha}}_{\text{order } \frac{m}{2}},
\end{aligned}
\tag{2.21}
$$

where $\{\mathbf{x}^{0,j,\gamma}\}_{j\in[m],\gamma\in[r_0]}$ *are the assembling vectors in the first layer, the intermediate weights* $\left\{\mathbf{a}^{l,j,\gamma} \in \mathbb{R}^{r_{l-1}}\right\}_{l\in[L-1],j\in[N/2^l],\gamma\in[r_l]}$*, the weights in the final layer* $\{\mathbf{a}^{L} \in \mathbb{R}^{r_{L-1}}\}$ *and* $\left\{\phi^{l,j,\gamma}\right\}_{l\in[L-1],j\in[m/2^l],\gamma\in[r_l]}$ *are* 2^l*th-order tensors.*

It is proved in [19] that a deep neural network corresponds to a HT decomposition, and a HT decomposition of polynomial size is able to express a CP decomposition with exponential size.

For $\mathcal{A} \in T_{m,n}^{\mathbb{R}}$, assume $r = \max_k r_k$, then the number of parameters we use to approximate the original tensor by the above tensor decompositions is given in Table 2.1.

Tensor networks (TNs) represent a high order tensor as interconnected lower order tensors. It is a direct way to realize tensor decompositions in the form of tensor networks. The basic TN notations and some famous tensor networks are given in Fig. 2.1 and Fig. 2.2 [18].

Tensor decomposition	The number of parameters
CP	$O(mnr)$
truncated Tucker	$O(r^m + mnr)$
TT	$O(mnr^2)$
HT	$O(mnr + mr^2)$

Table 2.1: The space complexity of tensor decompositions.

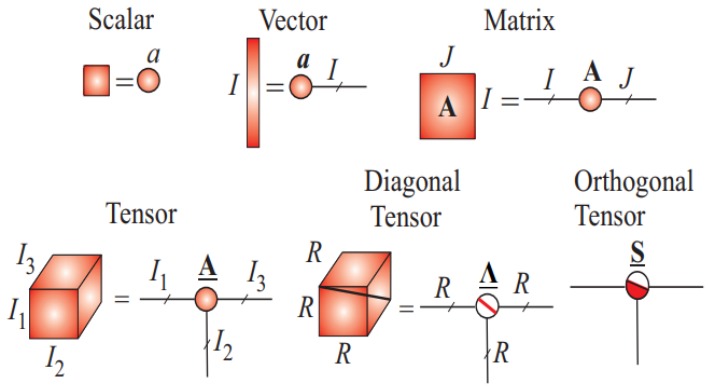


Figure 2.1: Basic tensor network notations.

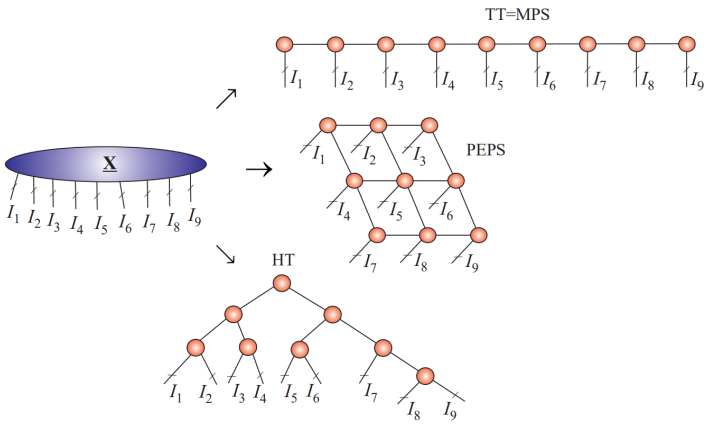


Figure 2.2: Some examples of tensor networks.

Chapter 3

Quantum Tensor Operations

In this chapter, some quantum operations on tensors are given, and these operations will be applied in the quantum algorithms in Chapters 5 & 6.

3.1 Quantum Data Structure of Tensors

A normalized vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{R}^n$ can be loaded into a quantum register by an oracle named quantum random access memory (qRAM) [29]:

$$\mathbf{x} \mapsto |x\rangle = \sum_{i=0}^{n-1} x_{i+1} |i\rangle \tag{3.1}$$

with preparation time $O(\log n)$. Note that in quantum computing the indices usually count from 0. The operation of qRAM is non-trivial to implement. A necessary condition to realize a quantum exponential speedup is the operation running in time at most polylogarithmic in n.

Similarly, a tensor $\mathcal{A} \in T^{\mathbb{R}}_{m,n}$ can be accessed by the following multipartite state

$$|\Psi\rangle = \sum_{i_1, i_2, \ldots, i_m = 0}^{n-1} a_{i_1 i_2 \cdots i_m} |i_1 i_2 \cdots i_m\rangle, \tag{3.2}$$

where $i_k = 0, \ldots, n-1$ for $k \in [m]$. This procedure can be achieved in time $O(m \log n)$.

A possible realization of qRAM is the data structure designed by Kerenidis and Prakash in [58, 39], which is a classical data structure with quantum access. The information is stored classically, but it can be accessed in quantum superposition. In the following, we first introduce the tree data structure for matrices and then extend this data structure to both real and complex mth-order tensors.

Lemma 3.1. (Tree data structure for matrices) [39]

Consider a matrix $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{I_1 \times I_2}$ *with* ω *nonzero entries. Let* $\mathbf{A}_i$ *be its* i*-th row of* $\mathbf{A}$*, and* $\hat{\mathbf{A}} = \frac{1}{\|\mathbf{A}\|_F} \left[\|\mathbf{A}_0\|_2, \|\mathbf{A}_1\|_2, \cdots, \|\mathbf{A}_{I_1-1}\|_2\right]^\top$*. There exists a data structure storing the matrix* $\mathbf{A}$ *in* $O(\omega \log^2(I_1 I_2))$ *space such that a quantum algorithm having access to the data structure can perform the mapping*

$$\mathbf{U}_P : |i\rangle |0\rangle \to |i\rangle |\mathbf{A}_i\rangle = \frac{1}{\|\mathbf{A}_i\|_2} \sum_{j=0}^{I_2-1} a_{ij} |i\rangle |j\rangle$$

$$\text{for} \quad i = 0, \cdots, I_1 - 1;$$

$$\mathbf{U}_Q : |0\rangle |j\rangle \to |\hat{\mathbf{A}}\rangle |j\rangle = \frac{1}{\|\mathbf{A}\|_F} \sum_{i=0}^{I_1-1} \|\mathbf{A}_i\|_2 |i\rangle |j\rangle$$

$$\text{for} \quad j = 0, \cdots, I_2 - 1 \tag{3.3}$$

in time $O(\text{polylog}(I_1 I_2))$.

In simple terms, there exists a unitary operator $\mathbf{U}_A$ that prepares $\mathbf{A}$ by

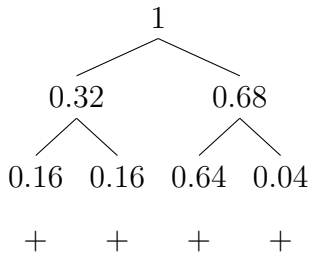
$$\mathbf{U}_A(|0\rangle^{\log I_1} |0\rangle^{\log I_2}) = \frac{1}{\|\mathbf{A}\|_F} \sum_{i,j} a_{ij} |i\rangle |j\rangle \tag{3.4}$$

in $O(\text{polylog}(I_1 I_2))$ time.

Denote $\mathbf{Y} \equiv \begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix}$ and $\mathbf{R}_y(\theta) \equiv e^{-\mathrm{i}\theta\mathbf{Y}/2} = \cos\frac{\theta}{2}\mathbf{I} - \mathrm{i}\sin\frac{\theta}{2}\mathbf{Y} = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$.

For example, for a 4-dimensional vector in the data structure as follows to be prepared as a 2-qubit state

$$|\varphi\rangle = 0.4\,|00\rangle + 0.4\,|01\rangle + 0.8\,|10\rangle + 0.2\,|11\rangle,$$



two rotations are necessary to be performed on the initial state $|00\rangle$, the first rotation is

$$\left(\mathbf{R}_y(2\arccos\sqrt{0.32})\otimes\mathbf{I}\right)|0\rangle|0\rangle = (\sqrt{0.32}\,|0\rangle + \sqrt{0.68}\,|1\rangle)\,|0\rangle\,. \tag{3.5}$$

Then, the second rotation is

$$\left(|0\rangle\langle 0|\otimes\mathbf{R}_y\left(\frac{\pi}{2}\right) + |1\rangle\langle 1|\otimes\mathbf{R}_y\left(2\arccos\sqrt{\frac{0.64}{0.68}}\right)\right)(\sqrt{0.32}\,|0\rangle + \sqrt{0.68}\,|1\rangle)\,|0\rangle$$

$$=0.4\,|00\rangle + 0.4\,|01\rangle + 0.8\,|10\rangle + 0.2\,|11\rangle\,. \tag{3.6}$$

Theorem 3.1. (Tree data structure for tensors)

For a tensor $\mathcal{A}\in\mathbb{R}(\mathbb{C})^{I_1\times I_2\times\cdots\times I_m}$, there exists a data structure for storing $\mathcal{A}$ with quantum access in time $O(\mathrm{polylog}(I_1I_2\cdots I_m))$.

Proof. We prepare a series of unitary operators and append an ancilla $|0\rangle$ at the

front when applying every operator

$$\mathbf{U}_m : |0\rangle \to |\phi_m\rangle = \frac{1}{\|\mathcal{A}\|_F} \sum_{i_m=0}^{I_m-1} \|\mathcal{A}(:,\ldots,:,i_m)\|_F \, |i_m\rangle$$

$$\begin{aligned}
\mathbf{U}_{m-1} : |0\rangle |\phi_m\rangle &\to |\phi_{m-1}\rangle \\
&= \frac{1}{\|\mathcal{A}\|_F} \sum_{i_{m-1}=0}^{I_{m-1}-1} \sum_{i_m=0}^{I_m-1} \frac{\|\mathcal{A}(:,\ldots,:,i_{m-1},i_m)\|_F}{\|\mathcal{A}(:,\ldots,:,i_m)\|_F} \|\mathcal{A}(:,\ldots,:,i_m)\|_F \, |i_{m-1}\rangle |i_m\rangle \\
&= \frac{1}{\|\mathcal{A}\|_F} \sum_{i_{m-1}=0}^{I_{m-1}-1} \sum_{i_m=0}^{I_m-1} \|\mathcal{A}(:,\ldots,:,i_{m-1},i_m)\|_F \, |i_{m-1}\rangle |i_m\rangle \\
&\vdots \\
\mathbf{U}_1 : |0\rangle |\phi_2\rangle &\to |\phi_1\rangle = \frac{1}{\|\mathcal{A}\|_F} \sum_{i_1=0}^{I_1-1} \cdots \sum_{i_m=0}^{I_m-1} a_{i_1\cdots i_m} \, |i_1\rangle \cdots |i_m\rangle .
\end{aligned} \tag{3.7}$$

Basically, the data structure consists of several binary trees named as $B_i^{(k)}$, $i = 0,\ldots,I_k-1, k \in [m]$. The top root stores the Frobenius norm of $\mathcal{A}$. The root of each $B_i^{(k)}$ stores the value $\|\mathcal{A}(:,\ldots,:,i_{k+1},\ldots,i_m)\|_F^2$ for $k \in [m-1]$ and the weight of an interior node is just the sum of the weights of its children. The leaf nodes at the bottom store the weights $a_{i_1 i_2\cdots i_m}^2$ and its sign $\mathrm{sgn}(a_{i_1 i_2\cdots i_m})$ for real tensors. For complex tensors, there is one more layer of binary trees for storing the real and imaginary parts of a complex entry $a_{i_1 i_2\cdots i_m}$ and their signs. Therefore, one more qubit is required for storing a complex tensor, and the extra time is $O(1)$ so we can omit it. If a quantum algorithm has access to this data structure, a series of controlled rotations are applied on the initial state in time $O(\lceil \log I_1 \rceil \cdots \lceil \log I_m \rceil)$. We consider it as $O(\mathrm{polylog}(I_1 I_2 \cdots I_m))$. □

For demonstration, the graph illustrations of the data structure for a real and complex $2 \times 2 \times 2$ tensor are given in Fig. 3.1 and 3.2 respectively.

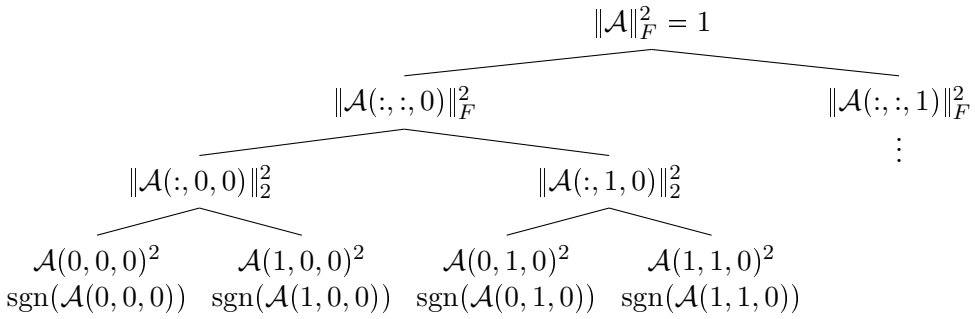


Figure 3.1: The data structure for a real $2 \times 2 \times 2$ tensor.

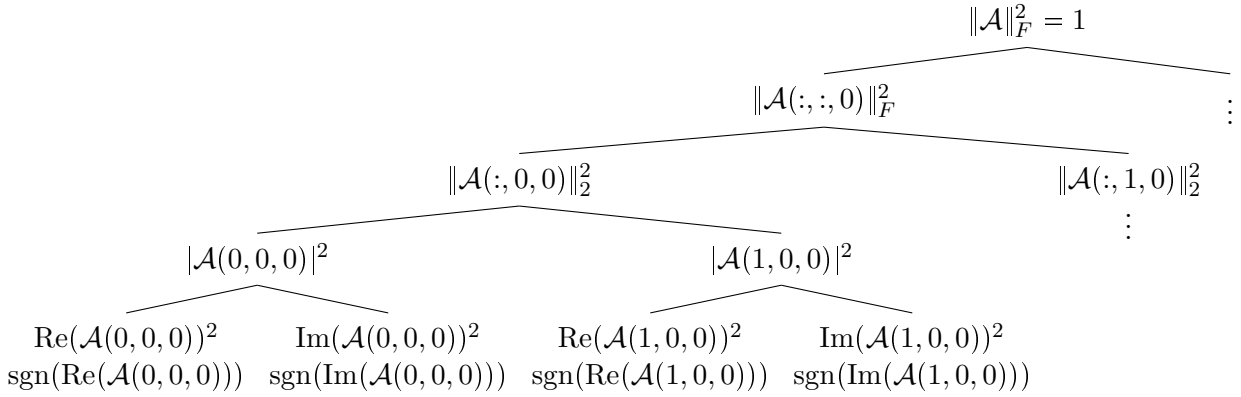


Figure 3.2: The data structure for a complex $2 \times 2 \times 2$ tensor.

3.2 Quantum Matrix Unfolding

In the following, the matrix unfolding which transforms a tensor to a matrix is introduced, and the quantum operations to perform matrix unfolding are described.

Definition 3.1. *For an mth-order tensor* $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$*, the* **mode-$k$ matrix unfolding** $\mathbf{A}^{(k)} \in \mathbb{C}^{I_k \times (\Pi_{j \neq k} I_j)}$ *contains the element* $a_{i_1 \cdots i_m}$ *at the position with row number* i_k *and column number*

$$(i_{k+1}-1)I_{k+2}I_{k+3}\cdots I_m I_1 I_2 \cdots I_{k-1} + (i_{k+2}-1)I_{k+3}I_{k+4}\cdots I_m I_1 I_2 \cdots I_{k-1} + \cdots$$

$$+(i_m - 1)I_1 I_2 \cdots I_{k-1} + (i_1 - 1)I_2 I_3 \cdots I_{k-1} + (i_2 - 1)I_3 I_4 \cdots I_{k-1} + \cdots + i_{k-1}.$$

By the above construction, the rank of $\mathbf{A}^{(k)}$ is at most I_k. Clearly, the elements of tensor $\mathcal{A}$ and unfolding matrix $\mathbf{A}^{(k)}$ have a one-to-one correspondence to each other.

The quantum unfolding matrix $\mathbf{A}^{(k)} = \left(a'^{(k)}_{i_k j_k}\right)$ can be processed by a SWAP operator $\mathbf{U}^{(k)}_{\mathrm{SP}}$:

$$\begin{aligned}
&\sum_{i_1,i_2,\ldots,i_m=0}^{n-1} a_{i_1 i_2 \cdots i_m} \left|i_1 i_2 \cdots i_m\right\rangle \\
\xrightarrow{\mathbf{U}^{(k)}_{\mathrm{SP}}} &\sum_{i_1,i_2,\ldots,i_m=0}^{n-1} a_{i_1 i_2 \cdots i_m} \left|i_k i_{k+1} \cdots i_m i_1 \cdots i_{k-1}\right\rangle \\
= &\sum_{i_k=0}^{n-1} \sum_{j_k=0}^{n^{m-1}-1} a'^{(k)}_{i_k j_k} \left|i_k j_k\right\rangle, \qquad (3.8)
\end{aligned}$$

where $|j_k\rangle = |i_{k+1} \cdots i_m i_1 \cdots i_{k-1}\rangle$. For example, for a $2 \times 2 \times 2$ tensor $\mathcal{A}$, the entries correspond to those of mode-3 unfolding matrix $\mathbf{A}^{(3)}$ by

$$\begin{aligned}
a_{000} \left|000\right\rangle &\to a'^{(3)}_{00} \left|00\right\rangle \\
a_{010} \left|010\right\rangle &\to a'^{(3)}_{01} \left|01\right\rangle \\
&\vdots \qquad (3.9) \\
a_{101} \left|101\right\rangle &\to a'^{(3)}_{12} \left|12\right\rangle \\
a_{111} \left|111\right\rangle &\to a'^{(3)}_{13} \left|13\right\rangle.
\end{aligned}$$

The corresponding SWAP operator is $\mathbf{U}^{(3)}_{\mathrm{SP}} = (\mathrm{SWAP} \otimes \mathbf{I})(\mathbf{I} \otimes \mathrm{SWAP})$, where $\mathbf{I}$ is a 2×2 identity matrix, and SWAP is the well-known 4×4 SWAP operator $\mathrm{SWAP} = \sum_{j,\ell=0}^{1} |\ell\rangle\langle j| \otimes |j\rangle\langle \ell|$. The circuit of the above operations and tensor input is given in Fig. 3.3. Except for mode-1 unfolding which does not require SWAP operations, other mode-k unfoldings require $m-1$ SWAP operations. Combined with the complexity of input, the total complexity is still $O(m \log n)$.

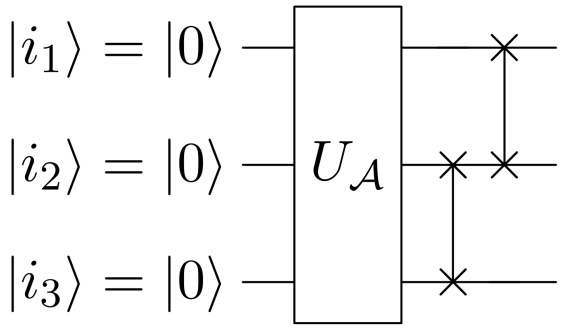


Figure 3.3: The quantum circuit to perform mode-3 matrix unfolding of a $2 \times 2 \times 2$ tensor. The unitary operator $\mathbf{U}_{\mathcal{A}}$ is the one for preparing tensor $\mathcal{A}$.

Chapter 4

Basic PSD Hankel Tensors

4.1 Introduction

The concept of Hankel tensors was first introduced by Luque and Thibon [48] to our best knowledge. Papy, De Lathauwer, and Van Huffel [56] initially employed Hankel tensors in the harmonic retrieval problem, which is at the heart of many signal processing problems. Hankel tensors have been widely applied in signal processing [4, 12, 15, 24], automatic control [71], and geophysics [54, 77]. Particularly, the positive semi-definiteness of Hankel tensors can be a criterion for the solvability of multidimensional moment problems [8, 44, 60].

It was proved by D. Hilbert [33] that for homogeneous polynomials, only in the following three cases, a positive semi-definite (PSD) polynomial is definitely a sum-of-squares (SOS) polynomial: 1) $n = 2$; 2) $m = 2$; 3) $m = 4$ and $n = 3$, where m is the degree of the polynomial and n is the number of variables. Hilbert proved that in all the other possible combinations of n and even m, there are PSD non-SOS (short for PNS as in [17]) homogeneous polynomials. The most well-known PNS homogeneous polynomial is the Motzkin function [49] with $m = 6$ and $n = 3$. A homogeneous polynomial is uniquely corresponding to a symmetric tensor [62], and a Hankel tensor is clearly a symmetric tensor.

In [59], it was showed that an mth-order n-dimensional tensor is a Hankel tensor

if and only if it has a Vandermonde decomposition. Two classes of PSD Hankel tensors were identified: even order strong Hankel tensors and even order complete Hankel tensors. It was proved in [46] that complete Hankel tensors are strong Hankel tensors, and even order strong Hankel tensors are SOS tensors. There were also some examples of SOS Hankel tensors and PSD Hankel tensors which are not strong Hankel tensors. Thus, a question was raised in [46]: Are all PSD Hankel tensors SOS tensors? If there are no PNS Hankel tensors, the problem for determining a given even order Hankel tensor is PSD or not can be answered by solving a semi-definite linear programming problem. The problem raised by the above question is called the Hilbert-Hankel problem, which is the first one of three open problems on Hankel tensors [73].

Generalized anti-circulant tensors [45] were studied, which are one special class of Hankel tensors. The necessary and sufficient conditions for positive semi-definiteness of even order generalized anti-circulant tensors in some cases were given, and the tensors are strong Hankel tensors and SOS tensors in these cases. An inheritance property was established in [59] for strong Hankel tensors, and this property was then extended to general Hankel tensors in [25], which means that if a lower-order Hankel tensor is positive semi-definite (or positive definite, or negative semi-definite, or negative definite, or SOS), then its associated higher-order Hankel tensor with the same generating vector, where the higher order is a multiple of the lower order, is also positive semi-definite (or positive definite, or negative semi-definite, or negative definite, or SOS, respectively). In addition, the SOS decomposition of strong Hankel tensors was also given in [25]. Other discussions about PSD Hankel tensors, SOS Hankel tensors and PNS Hankel tensors and some regions where PNS Hankel tensors do not exist were given in [16]. More properties of the above tensors are introduced in Chapter 5 of [62]. An algorithm for computing Vandermonde rank decompositions for all Hankel tensors was given in [50] and it was also proved that for a generic

Hankel tensor of order even or three, the CP rank, symmetric rank, border rank, symmetric border rank and Vandermonde rank are all the same.

Ding, Qi, and Wei [25] proved that a Hankel tensor is a strong Hankel tensor if and only if it admits a Vandermonde decomposition with positive coefficients or an augmented Vandermonde decomposition with positive coefficients. Thus, the decomposition of strong Hankel tensors has been settled. However, still little is known for non-strong Hankel tensors. Some non-strong PSD Hankel tensors were characterized in [80], yet a systematic investigation on non-strong Hankel tensors needs to be conducted.

Here we continue to study positive semi-definite Hankel tensors that are not strong. A new subclass of Hankel tensors called basic PSD Hankel tensors is introduced. We show that a rank-1 even order Hankel tensor is a strong Hankel tensor if and only if it is a basic PSD Hankel tensor, and even order strong Hankel tensors with rank higher than 1 can be represented as the sum of rank-1 basic PSD Hankel tensors. Therefore, the study of non-strong PSD Hankel tensors is converted to the study of basic PSD Hankel tensors with rank > 1. The properties of basic PSD Hankel tensors and decomposition of non-basic PSD Hankel tensors will help us find solutions to the three open problems on Hankel tensors in further research, since all of the open problems are in the context of PSD Hankel tensors.

4.2 Preliminaries

Definition 4.1. (Hankel tensor) [46, 59]

Let $\mathbf{v} = (v_1, \cdots, v_{(n-1)m+1})^\top$. *Define* $\mathcal{A} = (a_{i_1 \cdots i_m}) \in S_{m,n}$ *by*

$$a_{i_1 \cdots i_m} = v_{i_1 + \cdots + i_m - m + 1}, \tag{4.1}$$

for $i_1, \cdots, i_m \in [n]$. *Then* $\mathcal{A}$ *is called a Hankel tensor and* $\mathbf{v}$ *is called the generating vector of* $\mathcal{A}$.

If $\mathcal{A}$ is a Hankel tensor, then homogeneous polynomial $f(\mathbf{x}) = \mathcal{A}\mathbf{x}^m$ is called a Hankel polynomial. Let $\mathbf{A} = (a_{ij})$ be an $\left\lceil \frac{(n-1)m+2}{2} \right\rceil \times \left\lceil \frac{(n-1)m+2}{2} \right\rceil$ matrix with $a_{ij} \equiv v_{i+j-1}$, where $v_{2\left\lceil \frac{(n-1)m}{2} \right\rceil}$ is an additional number which can be arbitrarily selected when $(n-1)m$ is odd. Such $\mathbf{A}$ is called a **Hankel matrix**, associated with the Hankel tensor $\mathcal{A}$. When $(n-1)m$ is even, the associated Hankel matrix is unique. Recall from [59] that $\mathcal{A}$ is called a **strong Hankel tensor** if there exists an associated Hankel matrix $\mathbf{A}$ which is positive semi-definite. Let $g(\mathbf{y}) = \mathbf{y}^\top \mathbf{A}\mathbf{y}$, where $\mathbf{y} = \left(y_1, \cdots, y_{\frac{(n-1)m+2}{2}}\right)^\top$ and $\mathbf{A}$ is an associated Hankel matrix of $\mathcal{A}$. Then, $\mathcal{A}$ is a strong Hankel tensor if and only if g is PSD for at least one associated Hankel matrix $\mathbf{A}$ of $\mathcal{A}$.

In [25], it is proved that if $\mathbf{v}^{\circ m}$ is a rank-1 Hankel tensor, then $\mathbf{v} = \alpha(1, \xi, \cdots, \xi^{n-1})^\top$ or $\alpha\mathbf{e}_n = \alpha(0, 0, \cdots, 0, 1)^\top$, here vectors $(1, \xi, \cdots, \xi^{n-1})^\top$ and $\mathbf{e}_n$ are called **Vandermonde vectors**.

Lemma 4.1. *Let $\mathcal{A}$ be an mth-order n-dimensional Hankel tensor and the rank of its associated Hankel matrix be r. $\mathcal{A}$ is a strong Hankel tensor if and only if it admits a* **Vandermonde decomposition** *with positive coefficients:*

$$\mathcal{A} = \sum_{k=1}^{r} \alpha_k \mathbf{v}_k^{\circ m}, \tag{4.2}$$

$\alpha_k > 0$, $\mathbf{v}_k$ are Vandermonde vectors.

Also, there are many PSD Hankel tensors that are not strong Hankel tensors. For instance, consider the Hankel tensors $\mathcal{A}$ generated by $\mathbf{v} = \left(v_0, 0, \cdots, 0, v_{\frac{(n-1)m}{2}}, 0, \cdots, 0, v_{(n-1)m}\right)^\top$ where n is odd. Such Hankel tensors are called **truncated Hankel tensors** in [80]. If $\mathbf{v} = \left(v_0, 0, \cdots, 0, v_{\frac{(n-1)m}{2}}, 0, \cdots, 0, v_{(n-1)m}\right)^\top$ where n is odd, then $f(\mathbf{x})$ and $g(\mathbf{y})$

have a simple form

$$\begin{aligned} f(\mathbf{x}) &= v_0 x_1^m + v_{(n-1)m} x_n^m \\ &\quad + v_{\frac{(n-1)m}{2}} \sum \left\{ \binom{m}{t_1} \binom{m-t_1}{t_2} \cdots \binom{m-t_1-t_2-\cdots-t_{n-2}}{t_{n-1}} x_1^{t_1} x_2^{t_2} \cdots x_n^{m-t_1-t_2-\cdots-t_{n-1}} \right. \\ &\qquad \left. : (n-1)t_1 + (n-2)t_2 + \cdots + t_{n-1} = \frac{(n-1)m}{2} \right\}, \end{aligned} \tag{4.3}$$

and

$$g(\mathbf{y}) = v_0 y_1^2 + v_{(n-1)m} y_{\frac{(n-1)m+2}{2}}^2 + v_{\frac{(n-1)m}{2}} \left(y_{\frac{(n-1)m}{4}+1}^2 + \sum_{i \neq j} \left\{ y_i y_j : i + j = \frac{(n-1)m}{2} + 2 \right\} \right). \tag{4.4}$$

Since we are only concerned about PSD Hankel tensors, we may assume that $v_0, v_{\frac{(n-1)m}{2}}$, and $v_{(n-1)m}$ are all nonnegative. If $v_{\frac{(n-1)m}{2}} = 0$, then the truncated Hankel tensor $\mathcal{A}$ is a strong Hankel tensor, and furthermore an SOS Hankel tensor if m is even. If $v_{\frac{(n-1)m}{2}} > 0$, then $\mathcal{A}$ is not a strong Hankel tensor [80].

4.3 Basic PSD Hankel Tensors

Definition 4.2. (Basic PSD Hankel tensor)
Let $\mathcal{A}$ be an mth-order n-dimensional PSD Hankel tensor and its rank be r. Then $\mathcal{A}$ is called a basic PSD Hankel tensor, if there is no nonzero PSD Hankel tensor $\mathcal{B}$ with $\text{rank}(\mathcal{B}) < r$ *such that $\mathcal{A} - \mathcal{B}$ is PSD.*

From the definition, we can derive the following lemma straightforwardly.

Lemma 4.2. *Given that $\mathcal{A}$ is a rank-1 even order Hankel tensor, $\mathcal{A}$ is a strong Hankel tensor if and only if $\mathcal{A}$ is a basic PSD Hankel tensor.*

Proof. This lemma can be easily derived from the Vandermonde decomposition of strong Hankel tensors. □

Theorem 4.1. *All even order strong Hankel tensors with rank larger than 1 are not basic PSD Hankel tensors.*

Proof. Assume that an even order strong Hankel tensor $\mathcal{A}$ with rank $r \geqslant 2$ is basic, then from (4.2), $\mathcal{A} = \sum_{k=1}^{r} \alpha_k \mathbf{v}_k^{\circ m}$, $\alpha_k > 0$, $\mathbf{v}_k$ are Vandermonde vectors. Let $\mathcal{B} = \alpha_1 \mathbf{v}_1^{\circ m}$, then $\mathcal{B}$ is a positive semi-definite Hankel tensor, while $\mathcal{A} - \mathcal{B}$ is still positive semi-definite, which is a contradiction. □

Corollary 4.1. *All even order strong Hankel tensors with rank larger than 1 can be expressed as the sum of rank-1 basic PSD Hankel tensors.*

Clearly, PSD non-strong Hankel tensors with rank larger than 1 do exist. For example [80], for PSD truncated Hankel tensor $\mathcal{A}$ when $v_{\frac{(n-1)m}{2}} > 0$, consider vector $\bar{\mathbf{y}} = \mathbf{e}_i - \mathbf{e}_j$ where $i + j = \frac{(n-1)m}{2} + 2$, $i \neq j$ and $i \neq 1$ or $\frac{(n-1)m+2}{2}$. We see that $g(\bar{\mathbf{y}}) = -2v_{\frac{(n-1)m}{2}} < 0$, hence $\mathcal{A}$ is not a strong Hankel tensor. Therefore basic PSD Hankel tensors with rank larger than 1 also exist. What we concern is the smallest symmetric rank of non-strong basic PSD Hankel tensors.

Lemma 4.3. *If PSD Hankel tensor $\mathcal{A}$ has the following Vandermonde decomposition*

$$\mathcal{A} = \sum_{k=1}^{r} \alpha_k \mathbf{v}_k^{\circ m},$$

where $\mathbf{v}_k \in \mathbb{R}^n$ are mutually distinct Vandermonde vectors, $r \leqslant n$, then $\alpha_k > 0$, $k = 1, 2, \cdots, r$. Thus, $\mathcal{A}$ is a strong Hankel tensor.

Proof. $\mathcal{A}$ can be also expressed as

$$\mathcal{A} = \mathbf{D} \times_1 \mathbf{V}^\top \times_2 \mathbf{V}^\top \cdots \times_m \mathbf{V}^\top,$$

where $\mathbf{D}$ is a diagonal tensor with diagonal entries $\alpha_1, \alpha_2, \cdots, \alpha_r$ and matrix $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_r)$ is of full column rank. Without loss of generality, assume that $\alpha_1 < 0$, then there exists a unique $\mathbf{x}$ satisfying $\mathbf{V}^\top \mathbf{x} = \mathbf{e}_1$ such that $f(\mathbf{x}) = \mathcal{A}\mathbf{x}^m = \mathbf{D}\mathbf{e}_1^m = \alpha_1 < 0$, which is a contradiction to the positive semi-definiteness of $\mathcal{A}$. Therefore we have all $\alpha_k \geqslant 0$ for $k = 1, 2, \cdots, r$, thus $\mathcal{A}$ is a strong Hankel tensor. □

4.4 Rank-2 Basic PSD Hankel Tensors

We begin with the rank-2 case and we shall shortly see that there are no rank-2 basic PSD Hankel tensors. For mth-order n-dimensional PSD Hankel tensor $\mathcal{A}$, $m \geqslant 4$, if $\mathcal{A}$ is basic and its rank is 2, then it has the following form

$$\mathcal{A} = \alpha \mathbf{x}^{\circ m} + \beta \mathbf{y}^{\circ m}, \tag{4.5}$$

where $\alpha, \beta \neq 0$, $\mathbf{x} = (x_1, x_2, ..., x_n)^\top \in \mathbb{R}^n$, $\mathbf{y} = (y_1, y_2, ..., y_n)^\top \in \mathbb{R}^n$, $\mathbf{x} \neq \mathbf{y}$. As $\mathcal{A}$ is positive semi-definite, at least one of α and β is positive. Without loss of generality, assume $\beta = 1$, i.e.,

$$\mathcal{A} = \alpha \mathbf{x}^{\circ m} + \mathbf{y}^{\circ m}.$$

Theorem 4.2. *Rank-2 basic PSD Hankel tensors do not exist.*

Proof. If $n = 2$, from Lemma 2, $\mathcal{A}$ is not a basic PSD Hankel tensor. If $n \geqslant 3$, we classify the decomposition into the next four cases.

Case 1. $\mathbf{x} = (1, x_2, x_3, \cdots, x_n)^\top, \mathbf{y} = (1, y_2, y_3, \cdots, y_n)^\top$.

From the definition of Hankel tensors, $a_{11\ldots122} = a_{11\ldots13}$, then

$$\alpha(x_2^2 - x_3) = y_3 - y_2^2. \tag{4.6}$$

(a) If $x_2^2 \neq x_3$ and $y_2^2 \neq y_3$, then similarly we have $\alpha x_i(x_2^2 - x_3) = y_i(y_3 - y_2^2)$ for $2 \leqslant i \leqslant n$. By dividing this equation by (4.6) on both sides, we get $x_i = y_i$ for $2 \leqslant i \leqslant n$, hence $\mathbf{x} = \mathbf{y}$, the rank of $\mathcal{A}$ is actually 1, which is a contradiction.

(b) If $x_2^2 = x_3$, then $y_2^2 = y_3$. According to the definition of Hankel tensors, we have

$$\begin{cases} \alpha x_1^{m-3}x_2^3 + y_1^{m-3}y_2^3 = \alpha x_1^{m-2}x_4 + y_1^{m-2}y_4, \\ \alpha x_1^{m-1}x_5 + y_1^{m-1}y_5 = \alpha x_1^{m-2}x_2x_4 + y_1^{m-2}y_2y_4, \\ \alpha x_1^{m-2}x_3^2 + y_1^{m-2}y_3^2 = \alpha x_1^{m-1}x_5 + y_1^{m-1}y_5. \end{cases}$$

Processing these equations, we have $\alpha(x_2^4 - x_5) = x_2y_4 - x_2y_2^3 + y_5 - y_2y_4 = y_5 - y_2^4$. Thus $x_2 = y_2$ or $y_4 = y_2^3$. If $x_2 = y_2$, then $x_3 = y_3$ and $\alpha = 1$, hence $x_4 = y_4, ..., x_n = y_n$, $\mathbf{x} = \mathbf{y}$, the rank of $\mathcal{A}$ is 1, which is a contradiction. If $y_4 = y_2^3$, then $y_5 = y_2^4$, ..., $y_n = y_2^{n-1}$, and $x_4 = y_2^3$, ..., $x_n = x_2^{n-1}$, $\mathbf{x}$ and $\mathbf{y}$ are both Vandermonde vectors. By Lemma 2, it is a contradiction.

Case 2. $\mathbf{x} = (1, x_2, x_3, \cdots, x_n)^\top, \mathbf{y} = (0, \cdots, 0, 1, y_{k+1}, \cdots, y_n)^\top$, $2 \leqslant k \leqslant n$.

If $k = n$, i.e., $\mathbf{y} = \mathbf{e}_n$, $x_j = x_2^{j-1}$ for $2 \leqslant j \leqslant n$, it is a Vandermonde decomposition, which is a contradiction. If $2 \leqslant k \leqslant n-1$, obviously $x_j = x_2^{j-1}$ for $2 \leqslant j \leqslant n$, then (i) $\alpha x_k^m + y_k^m = \alpha x_{k-1}^{\frac{m}{2}} x_{k+1}^{\frac{m}{2}} + y_{k-1}^{\frac{m}{2}} y_{k+1}^{\frac{m}{2}}$ for m is even, (ii) $\alpha x_k^m + y_k^m = \alpha x_{k-1}^{\frac{m-1}{2}} x_k x_{k+1}^{\frac{m-1}{2}} + y_{k-1}^{\frac{m-1}{2}} y_k y_{k+1}^{\frac{m-1}{2}}$ for m is odd, i.e., $\alpha x_k^m + 1 = \alpha x_k^m$, which is also a contradiction.

Case 3. $\mathbf{x} = (0, \cdots, 0, 1, x_{k+1}, \cdots, x_n)^\top, \mathbf{y} = (0, \cdots, 0, 1, y_{l+1}, \cdots, y_n)^\top$, $2 \leqslant k \leqslant n, 2 \leqslant l \leqslant n-1, k \neq l$.

Without loss of generality, assume $k > l$. We have (i) $\alpha x_k^m + y_k^m = \alpha x_{k-1}^{\frac{m}{2}} x_{k+1}^{\frac{m}{2}} + y_{k-1}^{\frac{m}{2}} y_{k+1}^{\frac{m}{2}}$ for m is even, (ii) $\alpha x_k^m + y_k^m = \alpha x_{k-1}^{\frac{m-1}{2}} x_k x_{k+1}^{\frac{m-1}{2}} + y_{k-1}^{\frac{m-1}{2}} y_k y_{k+1}^{\frac{m-1}{2}}$ for m is odd, i.e., $\alpha = 0$, which is a contradiction.

Case 4. $\mathbf{x} = (0, \cdots, 0, 1, x_{k+1}, \cdots, x_n)^\top, \mathbf{y} = (0, \cdots, 0, 1, y_{k+1}, \cdots, y_n)^\top$, $2 \leqslant k \leqslant n$.

If $k = n$, the rank of $\mathcal{A}$ is 1, which is a contradiction. If $2 \leqslant k \leqslant n-1$,

we have (i) $\alpha x_k^m + y_k^m = \alpha x_{k-1}^{\frac{m}{2}} x_{k+1}^{\frac{m}{2}} + y_{k-1}^{\frac{m}{2}} y_{k+1}^{\frac{m}{2}}$ for m is even, (ii) $\alpha x_k^m + y_k^m = \alpha x_{k-1}^{\frac{m-1}{2}} x_k x_{k+1}^{\frac{m-1}{2}} + y_{k-1}^{\frac{m-1}{2}} y_k y_{k+1}^{\frac{m-1}{2}}$ for m is odd. From both situations we get $\alpha = -1$, then (i) $\alpha x_k^{m-1} x_{k+1} + y_k^{m-1} y_{k+1} = \alpha x_{k-1}^{\frac{m}{2}-1} x_k x_{k+1}^{\frac{m}{2}} + y_{k-1}^{\frac{m}{2}-1} y_k y_{k+1}^{\frac{m}{2}}$ for m is even, (ii) $\alpha x_k^{m-1} x_{k+1} + y_k^{m-1} y_{k+1} = \alpha x_{k-1}^{\frac{m-1}{2}} x_{k+1}^{\frac{m+1}{2}} + y_{k-1}^{\frac{m-1}{2}} y_{k+1}^{\frac{m+1}{2}}$ for m is odd. Thus $x_{k+1} = y_{k+1}$, ..., $x_n = y_n$, i.e., $\mathcal{A} = \mathbf{0}$, which is a contradiction. □

4.5 Rank-3 Basic PSD Hankel Tensors

For an mth-order n-dimensional PSD Hankel tensor $\mathcal{A}$, $m \geqslant 4$, $n \geqslant 3$, if $\mathcal{A}$ is a basic PSD Hankel tensor and its rank is 3, then it can be expressed as

$$\mathcal{A} = \alpha \mathbf{x}^{\circ m} + \beta \mathbf{y}^{\circ m} + \gamma \mathbf{z}^{\circ m}, \tag{4.7}$$

where $\alpha, \beta, \gamma \neq 0$, $\mathbf{x} = (x_1, x_2, ..., x_n)^\top \in \mathbb{R}^n$, $\mathbf{y} = (y_1, y_2, ..., y_n)^\top \in \mathbb{R}^n$, $\mathbf{z} = (z_1, z_2, ..., z_n)^\top \in \mathbb{R}^n$, $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are mutually distinct. Similar to the previous chapter, at least one of α, β and γ is positive. Without loss of generality, let $\gamma = 1$, then

$$\mathcal{A} = \alpha \mathbf{x}^{\circ m} + \beta \mathbf{y}^{\circ m} + \mathbf{z}^{\circ m}.$$

Theorem 4.3. *Rank-3 basic PSD Hankel tensors with dimension no less than 3 do not exist.*

Proof. The decomposition can be classified into the next four cases.

Case 1. $\mathbf{x} = (1, x_2, x_3, \cdots, x_n)^\top, \mathbf{y} = (1, y_2, y_3, \cdots, y_n)^\top, \mathbf{z} = (1, z_2, z_3, \cdots, z_n)^\top$.

Similar to the rank-2 situation, from the definition of Hankel tensors, $\alpha x_1^{m-2} x_2^2 + \beta y_1^{m-2} y_2^2 + z_1^{m-2} z_2^2 = \alpha x_1^{m-1} x_3 + \beta y_1^{m-1} y_3 + z_1^{m-1} z_3$, so

$$z_3 - z_2^2 = \alpha(x_2^2 - x_3) + \beta(y_2^2 - y_3). \tag{4.8}$$

Similarly, we have

$$\begin{cases} \alpha x_2^3 + \beta y_2^3 + z_2^3 = \alpha x_2 x_3 + \beta y_2 y_3 + z_2 z_3, \\ \alpha x_2^4 + \beta y_2^4 + z_2^4 = \alpha x_2^2 x_3 + \beta y_2^2 y_3 + z_2^2 z_3. \end{cases}$$

By substituting (4.8) into the above equations, we obtain

$$\alpha(x_2 - z_2)(x_2^2 - x_3) = -\beta(y_2 - z_2)(y_2^2 - y_3), \tag{4.9}$$

$$\alpha(x_2^2 - z_2^2)(x_2^2 - x_3) = -\beta(y_2^2 - z_2^2)(y_2^2 - y_3). \tag{4.10}$$

Next, we discuss whether the factors in equation (4.9) are zero or not, and classify it into the next four situations (a) $\sim$ (d).

(a) $x_2 \neq z_2$, $y_2 \neq z_2$, $x_2^2 \neq x_3$, $y_2^2 \neq y_3$.

Divide (4.10) by (4.9) on both sides, we get $x_2 = y_2$. Then, similarly $\alpha(x_k - z_k)(x_2^2 - x_3) = -\beta(y_k - z_k)(y_2^2 - y_3)$ for $3 \leqslant k \leqslant n$, hence $\mathbf{x} = \mathbf{y}$, which is a contradiction.

(b) $x_2 = z_2$, $y_2 = z_2$.

For x_3, y_3, z_3, we have

$$\alpha(z_3 - x_3)(x_2^2 - x_3) = -\beta(y_3 - z_3)(y_2^2 - y_3), \tag{4.11}$$

$$\alpha(z_3^2 - x_3^2)(x_2^2 - x_3) = -\beta(y_3^2 - z_3^2)(y_2^2 - y_3). \tag{4.12}$$

We get two similar equations about x_3, y_3, z_3, and also discuss the factors in situations (i) $\sim$ (iv). (i) If $x_3 \neq z_3$, $y_3 \neq z_3$, $x_2^2 \neq x_3$, $y_2^2 \neq y_3$, let (4.12) divided by (4.11), we get $x_3 = y_3$. (ii) If $x_3 = z_3$, $y_3 = z_3$, obviously $x_3 = y_3$. (iii) If $x_3 = x_2^2$, $y_3 = y_2^2$, we obtain $x_3 = y_3$, because $x_2 = y_2$. (iv) If $x_3 = z_3$, $y_3 = y_2^2$ (or $y_3 = z_3$, $x_3 = x_2^2$), substitute the two equations into (4.8), we have $x_3 = x_2^2 = y_2^2 = y_3$. Then $x_3 = y_3 = z_3$ for (i) $\sim$ (iv). Similarly, we can prove $x_k = y_k$ for $4 \leqslant k \leqslant n$, hence $\mathbf{x} = \mathbf{y}$, which is a contradiction.

(c) $x_2^2 = x_3$, $y_2 = z_2$.

(i) If $y_2^2 \neq y_3$, we have $\alpha x_2^2 + \beta y_2^2 + y_2^2 = \alpha x_3 + y_3 + z_3$, hence $\beta(y_2^2 - y_3) = z_3 - y_2^2$. Also, for $3 \leqslant i \leqslant n$, $\beta y_i(y_2^2 - y_3) = z_i(z_3 - y_2^2)$, hence we get $y_3 = z_3$, ..., $y_n = z_n$, i.e., $\mathbf{y} = \mathbf{z}$. Thus, the rank is 2, which is a contradiction. (ii) If $y_2^2 = y_3$, we will find $\mathbf{x}$ is a Vandermonde vector, then by (b) of Case 1 in the previous chapter, $\mathbf{y}$ and $\mathbf{z}$ are also Vandermonde vectors. The situation is similar for $y_2^2 = y_3$, $x_2 = z_2$.

(d) $x_2^2 = x_3$, $y_2^2 = y_3$.

It is included in (b) and (c) if at least one of $x_2 = z_2$ and $y_2 = z_2$ is satisfied. So we discuss $x_2 \neq z_2$ and $y_2 \neq z_2$ here. Obviously $z_3 = z_2^2$, if $n = 3$, $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are all Vandermonde vectors. If $n \geqslant 3$, then from the definition of Hankel tensors, $a_{11\ldots14} = a_{11\ldots1222}$, which implies

$$\alpha x_4 + \beta y_4 + z_4 = \alpha x_2^3 + \beta y_2^3 + z_2^3. \tag{4.13}$$

From $a_{11\ldots124} = a_{11\ldots1223}$ and $a_{11\ldots134} = a_{11\ldots1233}$, we have the following two equations

$$\begin{cases} \alpha x_2 x_4 + \beta y_2 y_4 + z_2 z_4 = \alpha x_2^2 x_3 + \beta y_2^2 y_3 + z_2^2 z_3, \\ \alpha x_3 x_4 + \beta y_3 y_4 + z_3 z_4 = \alpha x_2 x_3^2 + \beta y_2 y_3^2 + z_2 z_3^2. \end{cases}$$

Substitute (4.13) into the above equations and we have

$$\alpha(x_2 - z_2)(x_2^3 - x_4) + \beta(y_2 - z_2)(y_2^3 - y_4) = 0, \tag{4.14}$$

$$\alpha(x_2^2 - z_2^2)(x_2^3 - x_4) + \beta(y_2^2 - z_2^2)(y_2^3 - y_4) = 0. \tag{4.15}$$

(i) If $x_2^3 \neq x_4$, $y_2^3 \neq y_4$, then divide (4.15) by (4.14), we have $x_2 = y_2$, hence $x_3 = y_3$. Similarly $\alpha(x_k - z_k)(x_2^3 - x_4) + \beta(y_k - z_k)(y_2^3 - y_4) = 0$ and $\alpha(x_k^2 - z_k^2)(x_2^3 - x_4) + \beta(y_k^2 - z_k^2)(y_2^3 - y_4) = 0$ for $4 \leqslant k \leqslant n$. We get $x_k = y_k = z_k$ or $x_k = y_k$, thus $\mathbf{x} = \mathbf{y}$, which is a contradiction. (ii) If $x_2^3 = x_4$, $y_2^3 = y_4$, obviously $z_4 = z_2^3$, then repeat (d) for $x_2^4 = x_5$, $y_2^4 = y_5$, ..., $x_2^{n-1} = x_n$, $y_2^{n-1} = y_n$, and we find $\mathbf{x} = \mathbf{y}$, which is a contradiction, or $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are Vandermonde vectors.

Case 2. $\mathbf{x} = (1, x_2, x_3, \cdots, x_n)^\top, \mathbf{y} = (1, y_2, y_3, \cdots, y_n)^\top, \mathbf{z} = (0, \cdots, 0, 1, z_{k+1}, \cdots, z_n)^\top$, $2 \leqslant k \leqslant n$.

If $k = n$, i.e., $\mathbf{z} = \mathbf{e}_n$, then similar to the first case of rank 2, $\mathbf{x}$ and $\mathbf{y}$ are Vandermonde vectors, and $\mathbf{z}$ is also a Vandermonde vector. If $2 \leqslant k \leqslant n-1$, $\mathbf{x}$ and $\mathbf{y}$ are Vandermonde vectors, then $\alpha x_k^m + \beta y_k^m + 1 = \alpha x_k^m + \beta y_k^m$, which is a contradiction.

Case 3. $\mathbf{x} = (1, x_2, x_3, \cdots, x_n)^\top, \mathbf{y} = (0, \cdots, 0, 1, y_{k+1}, \cdots, y_n)^\top,$
$\mathbf{z} = (0, \cdots, 0, 1, z_{l+1}, \cdots, z_n)^\top$, $2 \leqslant k \leqslant n-1, 2 \leqslant l \leqslant n, k \neq l$.

From the definition of Hankel tensors, $\alpha x_1^{m-1} x_{i+1} + \beta y_1^{m-1} y_{i+1} + z_1^{m-1} z_{i+1} = \alpha x_1^{m-2} x_2 x_i + \beta y_1^{m-1} y_i + z_1^{m-1} z_i$ for $i = 2, 3, ..., n-1$. Since $y_1 = z_1 = 0$, we have $x_{i+1} = x_2 x_i$, hence $\mathbf{x}$ is a Vandermonde vector. Without loss of generality, assume $k < l$, we have (i) $\alpha x_k^m + \beta = \alpha x_{k-1}^{\frac{m}{2}} x_{k+1}^{\frac{m}{2}}$ for m is even, (ii) $\alpha x_k^m + \beta = \alpha x_{k-1}^{\frac{m-1}{2}} x_k x_{k+1}^{\frac{m-1}{2}}$ for m is odd, i.e., $\beta = 0$, which is a contradiction.

Case 4. $\mathbf{x} = (0, \cdots, 0, 1, x_{j+1}, \cdots, x_n)^\top, \mathbf{y} = (0, \cdots, 0, 1, y_{k+1}, \cdots, y_n)^\top,$
$\mathbf{z} = (0, \cdots, 0, 1, z_{l+1}, \cdots, z_n)^\top$, $2 \leqslant j, k, l \leqslant n, l \leqslant k \leqslant j$.

If $j < k \leqslant l$, then $\alpha = 0$, which is a contradiction.

If $j = k < l$, the situation is the same as Case 4 of rank 2, which is a contradiction.

If $j = k = l$, obviously j, k, l cannot be n, $2 \leqslant j = k = l \leqslant n-1$. As (i) $\alpha x_k^m + \beta y_k^m + z_k^m = \alpha x_{k-1}^{\frac{m}{2}} x_{k+1}^{\frac{m}{2}} + \beta y_{k-1}^{\frac{m}{2}} y_{k+1}^{\frac{m}{2}} + z_{k-1}^{\frac{m}{2}} z_{k+1}^{\frac{m}{2}}$ for m is even, (ii) $\alpha x_k^m + \beta y_k^m + z_k^m = \alpha x_{k-1}^{\frac{m-1}{2}} x_k x_{k+1}^{\frac{m-1}{2}} + \beta y_{k-1}^{\frac{m-1}{2}} y_k y_{k+1}^{\frac{m-1}{2}} + z_{k-1}^{\frac{m-1}{2}} z_k z_{k+1}^{\frac{m-1}{2}}$ for m is odd, we have

$$\begin{cases} \alpha + \beta + 1 = 0, \\ \alpha x_k + \beta y_k + z_k = 0, \\ \alpha x_k^2 + \beta y_k^2 + z_k^2 = 0, \\ \alpha x_k^3 + \beta y_k^3 + z_k^3 = 0. \end{cases}$$

Then

$$\begin{cases} \alpha x_k(x_k - z_k) + \beta y_k(y_k - z_k) = 0, \\ \alpha x_k(x_k^2 - z_k^2) + \beta y_k(y_k^2 - z_k^2) = 0. \end{cases}$$

(i) If $x_k = 0, y_k = z_k$, we get $\beta = -1$, $\alpha = 0$, which is a contradiction. (ii) If $y_k = 0, x_k = z_k$, then similar to (i), $\beta = 0$, which is also a contradiction. (iii) If $x_k, y_k \neq 0, x_k \neq z_k, y_k \neq z_k$, from $x_k + z_k = y_k + z_k$, we get $x_k = y_k$. (iv) Assume $x_k = y_k = 0$. (v) Assume $x_k = y_k = z_k$. From (iii) to (v), we all get $x_k = y_k$. If we continue checking $x_{k+1}, ..., x_n$, we will find $\mathbf{x} = \mathbf{y}$, or $\alpha = 0$, or $\beta = 0$, all situations cause contradictions.

Up till now, we have proved if there exists a rank-3 basic PSD Hankel tensor $\mathcal{A}$ with dimension no less than 3, $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are mutually distinct Vandermonde vectors. However, by Lemma 2, $r = 3 \leqslant n$, $\mathcal{A}$ is a strong Hankel tensor, which is a contradiction. □

Therefore, we put forward the following theorem.

Theorem 4.4. *For any non-basic PSD Hankel tensor $\mathcal{A}$ with* $\text{rank}(\mathcal{A}) \geqslant 2$*, $n \geqslant 3$, $\mathcal{A}$ can be expressed as*

$$\mathcal{A} = \sum_{k=1}^{r} \alpha_k \mathcal{B}_k, \tag{4.16}$$

where $r \in \mathbb{N}$, $\mathcal{B}_k$ are basic PSD Hankel tensors with $\text{rank}(\mathcal{B}_k) \geqslant 4$ *or* $\text{rank}(\mathcal{B}_k) = 1$.

This theorem can be proved by Theorem 2 and 3 straightforwardly.

4.6 An Example of Basic PSD Hankel Tensors with Rank Higher than 2

We shall present an example of basic PSD Hankel tensors in this section with rank > 2. Consider the following example. Let $\mathcal{A}$ be a 4th-order 2-dimensional

Hankel tensor generated by $\mathbf{v} = \left(1, 0, -\frac{1}{3}, 0, 1\right)^\top$. For $\mathbf{x} = (x_1, x_2)^\top \in \mathbb{R}^2$, the Hankel polynomial $f_{\mathcal{A}}(\mathbf{x}) = \mathcal{A}\mathbf{x}^4 = x_1^4 - 2x_1^2x_2^2 + x_2^4 = (x_1^2 - x_2^2)^2 \geqslant 0$. However, the associated Hankel matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & -\frac{1}{3} \\ 0 & -\frac{1}{3} & 0 \\ -\frac{1}{3} & 0 & 1 \end{pmatrix}$$

is apparently not positive semi-definite since there is a negative entry on its diagonal. Therefore, $\mathcal{A}$ is a PSD Hankel tensor, but not a strong Hankel tensor. Furthermore, $\mathcal{A}$ has a following decomposition that

$$\mathcal{A} = \frac{4}{3}\begin{pmatrix} 1 \\ 0 \end{pmatrix}^{\circ 4} - \frac{1}{6}\begin{pmatrix} 1 \\ 1 \end{pmatrix}^{\circ 4} - \frac{1}{6}\begin{pmatrix} 1 \\ -1 \end{pmatrix}^{\circ 4} + \frac{4}{3}\begin{pmatrix} 0 \\ 1 \end{pmatrix}^{\circ 4},$$

thus $\operatorname{rank}(\mathcal{A}) \leqslant 4$.

Next we prove that $\mathcal{A}$ is a basic PSD Hankel tensor. Assume $\mathcal{A}$ is not basic, then there exist two PSD Hankel tensors $\mathcal{B}$ and $\mathcal{C}$ such that $\mathcal{A} = \mathcal{B} + \mathcal{C}$ and for any vector $\mathbf{x} \in \mathbb{R}^2$, the Hankel polynomial $f_{\mathcal{A}}(\mathbf{x}) = f_{\mathcal{B}}(\mathbf{x}) + f_{\mathcal{C}}(\mathbf{x}) = (x_1^2 - x_2^2)^2$, and $f_{\mathcal{B}}(\mathbf{x}), f_{\mathcal{C}}(\mathbf{x}) \geqslant 0$. If $f_{\mathcal{B}}(\mathbf{x})$ does not have the factors $x_1 + x_2$ or $x_1 - x_2$, then take $x_1 = \pm x_2$, and we have $f_{\mathcal{B}}(\mathbf{x}) > 0$, $f_{\mathcal{C}}(\mathbf{x}) = 0 - f_{\mathcal{B}}(\mathbf{x}) < 0$, which is a contradiction. If both $x_1 + x_2$ and $x_1 - x_2$ are the factors of $f_{\mathcal{B}}(\mathbf{x})$, then by D. Hilbert [33], for 2-dimensional homogeneous polynomials, a PSD polynomial is definitely an SOS polynomial, hence $f_{\mathcal{B}}(\mathbf{x}) = \alpha(x_1^2 - x_2^2)^2$, $\alpha > 0$, tensor $\mathcal{B}$ is proportional to $\mathcal{A}$, which is a contradiction.

Therefore, we have found a basic PSD Hankel tensor whose rank equals 3, which implies that basic PSD Hankel tensor with rank > 2 does exist. To verify whether this tensor is exactly rank 4, we use Tensorlab toolbox in Matlab software to decompose this symmetric tensor. After running 1000 times, the minimum error (calculated by the Frobenius norm of tensor $\mathcal{A}$ minus the recombinated tensor) of finding the rank-3 decomposition is about 10^{-3} while that of find the rank-4 decomposition is around

10^{-17}. Therefore, there is a high probability that the rank of this tensor is exactly 4.

4.7 Conclusions and Conjectures

We have introduced a new subclass of Hankel tensors called basic PSD Hankel tensors. It is proved that for mth-order n-dimensional positive semi-definite Hankel tensors, there are no rank-2 basic PSD Hankel tensors. Moreover, rank-3 basic PSD Hankel tensors with dimension no less than 3 do not exist, either.

In the previous section, an example is given to show the existence of a basic PSD Hankel tensor with rank $\geqslant 3$. It is thus reasonable to conject the existence of other basic PSD Hankel tensors. The critical truncated Hankel tensor $\mathcal{A}$ in [80] may also be a basic PSD Hankel tensor. $\mathcal{A}$ is a sixth-order three-dimensional PSD truncated Hankel tensor, and the elements of its generating vector $(v_0, 0, \cdots, 0, v_6, 0, \cdots, 0, v_{12})^\top$ satisfy $\sqrt{v_0 v_{12}} = (560 + 70\sqrt{70})v_6$.

For a rank-3 PSD Hankel tensor with $n = 2$, we cannot prove or disprove it is a basic PSD Hankel tensor. Therefore, we put forward a conjecture that 2-dimensional rank-3 basic PSD Hankel tensors do not exist and the rank of tensor $\mathcal{A}$ in the given example in Section 6 is 4.

Chapter 5

Quantum Higher Order Singular Value Decomposition

5.1 Introduction

Higher order singular value decomposition is a specific orthogonal Tucker decomposition, and can be considered as an extension of SVD from matrices to tensors. Classical HOSVD has been well studied, see, e.g., De Lathauwer, De Moor, and Vandewalle in 2000 [23], and it has been successfully applied to signal processing [53] and pattern recognition [79] problems. Furthermore, HOSVD has shown its strong power in quantum chemistry, especially in the second order Møller Plesset perturbation theory calculations [6]. In addition, HOSVD is used in [84] to derive the output m photon state of a quantum linear passive system which is driven by an m photon input state; more specifically, the wave function of the output is expressed in terms of the HOSVD of the input wave function.

Since HOSVD deals with high dimensional data, it has been put into practice in some machine learning methods. For example, it has been successfully applied in recommendation systems [38, 74]. In [68], HOSVD representation for neural networks is proposed. By applying HOSVD the parameter-varying system can be expressed in a tensor product form by locally tuned neural network models. Additionally in

[41], HOSVD is applied for compressing convolutional neural networks (CNN).

In this chapter, we propose the quantum higher order singular value decomposition (Q-HOSVD) algorithm. This is the first quantum algorithm concerning tensor decompositions. Our Q-HOSVD algorithms are based upon the quantum matrix singular value decomposition algorithm [67], quantum singular value estimation [39], and several other quantum computing techniques. The input can be a tensor of any order and dimension. By our Q-HOSVD algorithms, it is possible to perform singular value decomposition on tensors exponentially faster than classical algorithms. It can be directly applied to quantum machine learning algorithms, and may help solve computationally challenging problems arising in quantum mechanics and chemistry.

5.2 Definition and Properties

Definition 5.1. [23] *For* $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$*, the* **higher order singular value decomposition (HOSVD)** *is defined as*

$$\mathcal{A} = \mathcal{S} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \cdots \times_m \mathbf{U}^{(m)}, \tag{5.1}$$

where the k-mode singular matrix $\mathbf{U}^{(k)} = \left[\mathbf{u}_1^{(k)} \mathbf{u}_2^{(k)} \cdots \mathbf{u}_{I_k}^{(k)}\right]$ *is a complex unitary* $I_k \times I_k$ *matrix, the core tensor* $\mathcal{S} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_m}$.

The core tensor $\mathcal{S}$ and its subtensors $\mathcal{S}_{i_k=\alpha}$, of which the kth index is fixed to $\alpha \in [I_k]$, have the following properties.

(i) all-orthogonality:

Two subtensors $\mathcal{S}_{i_k=\alpha}$ and $\mathcal{S}_{i_k=\beta}$ are orthogonal for $k = 1, 2, \cdots, m$:

$$\mathcal{S}_{i_k=\alpha} \cdot \mathcal{S}_{i_k=\beta} = 0 \qquad \text{when} \quad \alpha \neq \beta, \tag{5.2}$$

(ii) ordering:

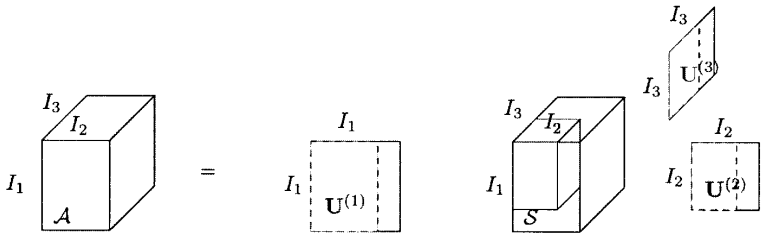


Figure 5.1: Block diagram of the HOSVD for a third-order tensor. The full lines indicate the full HOSVD in (5.1). The dashed lines and the small block in $\mathcal{S}$ indicate the truncated HOSVD.

Similar to the matrix case, the **tensor singular values** are defined as the Frobenius norms of the $(m-1)$th-order subtensors of the core tensor $\mathcal{S}$:

$$\sigma_\alpha^{(k)} = \|\mathcal{S}_{i_k=\alpha}\|_F, \tag{5.3}$$

for $k \in [m]$ and $\alpha \in [I_k]$. Furthermore, these tensor singular values have the following ordering property

$$\sigma_1^{(k)} \geqslant \sigma_2^{(k)} \geqslant \cdots \geqslant \sigma_{I_k}^{(k)} \geqslant 0 \tag{5.4}$$

for $k \in [m]$. The block diagram of the HOSVD for a third-order tensor $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times I_3}$ is described in Fig. 5.1. When $m = 2$, i.e., $\mathcal{A}$ is a matrix, the HOSVD is degenerated to the well-known matrix SVD.

$\mathbf{U}^{(k)}$ is calculated through the SVD of unfolding matrix $\mathbf{A}^{(k)}$, where matrix unfolding is defined in Chapter 3. If $\mathcal{A}$ is a symmetric tensor, i.e., $\mathcal{A} \in S_{m,n}$, then all $\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(m)}$ are the same unitary matrices. Thus, we calculate the SVD of unfolding matrices only once instead of m times, and the decomposition is converted to

$$\mathcal{A} = \mathcal{S} \times_1 \mathbf{U} \times_2 \mathbf{U} \times_3 \cdots \times_m \mathbf{U}. \tag{5.5}$$

HOSVD performs orthogonal coordinate transformations for a higher-order tensor. Here, the unitary matrix $\mathbf{U}^{(k)}$ is also called the k-mode factor matrix and

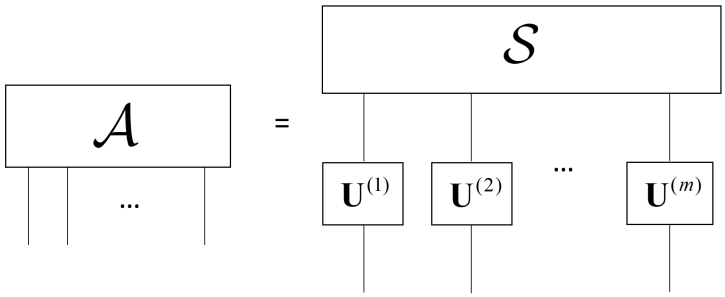


Figure 5.2: The tensor network notation of HOSVD.

considered as the principal components in kth mode. Moreover, the entries of the core tensor $\mathcal{S}$ show the level of interaction among different components.

The tensor network notation of HOSVD is depicted in Fig. 5.2. It is known that a tensor corresponds to a multipartite quantum state. Any local unitary transformation on the original tensor can be considered as the local unitary transformation on the corresponding singular matrices, which is vital and useful in quantum computation. The core tensor and singular matrices can also be considered as the two layers in the neural network, with local operations in the first layer and global operations in the second layer.

In HOSVD, the columns of $\mathbf{U}^{(k)}$ have been sorted such that the jth column $\mathbf{u}_j^{(k)}$ corresponds to the jth largest nonzero singular value of $\mathbf{A}^{(k)}$. Then, we can similarly define the **truncated (or compact) HOSVD** [75]. For $k \in [m]$, we remain the first r_k columns of $\mathbf{U}^{(k)}$, then $\mathbf{U}^{(k)} \in \mathbb{C}^{I_k \times r_k}$. Finally, the core tensor $\mathcal{S}$ is of size $r_1 \times r_2 \times \cdots \times r_m$, and the tuple of numbers $(r_1, r_2, \cdots, r_m)$ is called a multilinear rank. The block diagram of the truncated HOSVD for a third-order tensor is depicted in Fig. 5.1. This truncation is widely used in big data problems. Since the data may be sparse or low-rank, we can take the value of r_k such that $r_k \ll I_k$. Denote $r = \max_{k\in[m]} r_k$, and $I = \max_{k\in[m]} I_k$. The total number of entries reduces from I^m to $r^m + mIr$.

5.3 Q-HOSVD Algorithm 1

In this section, we present our first Q-HOSVD algorithm.

Algorithm 1 Quantum Higher Order Singular Value Decomposition (Q-HOSVD)

Input: $\mathcal{A} \in \mathbb{C}^{I_1 \times \cdots \times I_m}, \epsilon, |b\rangle$
Output: $\mathcal{S}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(m)}$

1. Load $\mathcal{A}$ into qRAM, and initialize $|0\rangle |\vec{1}\rangle |b\rangle$.
for $k = 1, \ldots, m$ **do**
2. Implement qPCA by the SWAP operator $\mathbf{S}_{\tilde{A}}^{(k)}$.
3. Apply phase estimation to obtain

$$|\psi\rangle = \sum_{j=0}^{r-1} \beta_j |\tilde{\lambda}_j / N\rangle |\tilde{u}_j\rangle . \tag{5.6}$$

4. Perform measurement on $|\tilde{\lambda}_j / N\rangle$ and extract $|\tilde{u}_j\rangle$ to compose $\mathbf{U}^{(k)}$.
end for
5. $\mathcal{S} \leftarrow \mathcal{A} \times_1 \mathbf{U}^{(1)^\dagger} \times_2 \mathbf{U}^{(2)^\dagger} \times_3 \cdots \times_m \mathbf{U}^{(m)^\dagger}$.

Several techniques and subroutines are applied in Algorithm 1. First, tensor $\mathcal{A}$ to be decomposed is loaded into the quantum register by qRAM. For a fixed k, we design a SWAP operator $\mathbf{S}_{\tilde{A}}^{(k)}$ based on matrix unfolding and Hermitian extension, and harness it to apply qPCA. After that, we initialize the state $|0\rangle |\vec{1}\rangle |b\rangle$, where $|\vec{1}\rangle = \frac{1}{\sqrt{N}} \sum_{\ell=0}^{N-1} |\ell\rangle$, and $|b\rangle$ could be any state and considered as a superposition of eigenstates of $\tilde{\mathbf{A}}^{(k)}$, then apply phase estimation on it to obtain the state $|\psi\rangle$ which is a superposition state composed of eigenvalues and eigenstates. Then, if we hope to obtain the core tensor $\mathcal{S}$, quantum measurement is performed to reconstruct the singular matrices. Finally, $\mathcal{S}$ is calculated by the quantum tensor-matrix multiplication among tensor $\mathcal{A}$ and the singular matrices.

In the following sections, we explain the implementation of Algorithm 1 step by step. Without loss of generality, we assume $\|\mathcal{A}\|_F = 1$ and $I_1 = I_2 = \cdots = I_m = n$.

For **Step 1**, tensor $\mathcal{A}$ can be accessed by qRAM or the tree structure in Chapter

3.

For **Step 2**, denote $N = n + n^{m-1}$. Since $\mathbf{A}^{(k)} = \left(a'^{(k)}_{ij}\right) \in \mathbb{C}^{n\times n^{m-1}}$ is not a Hermitian matrix, we consider the following extended matrix

$$\begin{aligned}
\tilde{\mathbf{A}}^{(k)} &:= \begin{bmatrix} 0 & \mathbf{A}^{(k)} \\ \mathbf{A}^{(k)\dagger} & 0 \end{bmatrix} \\
&= \sum_{i_k=0}^{n-1} \sum_{j_k=0}^{n^{m-1}-1} a'^{(k)}_{i_k j_k} \left|i_k\right\rangle\left\langle j_k + n\right| + \overline{a'^{(k)}_{i_k j_k}} \left|j_k + n\right\rangle\left\langle i_k\right| \\
&= \sum_{i_1,i_2,\ldots,i_m=0}^{n-1} a_{i_1 i_2 \cdots i_m} \left|i_k\right\rangle\left\langle i_{k+1}\cdots i_m i_1 \cdots i_{k-1} + n\right| \\
&\quad + \overline{a_{i_1 i_2 \ldots i_m}} \left|i_{k+1}\cdots i_m i_1 \cdots i_{k-1} + n\right\rangle\left\langle i_k\right|, \qquad (5.7)
\end{aligned}$$

where $|i_k\rangle \in \mathbb{C}^N$ is the i_k-th computational basis. Note that i_k runs from 0 to $n-1$. Then $\tilde{\mathbf{A}}^{(k)}$ is an $N \times N$ Hermitian matrix. For Hermitian matrices, the singular values are the absolute value of eigenvalues, so phase estimation [51] can be used to apply the singular value decomposition. Let $r = \mathrm{rank}(\tilde{\mathbf{A}}^{(k)})$. Since $\mathrm{rank}(\mathbf{A}^{(k)}) \leqslant n$, $r \leqslant 2n$.

For the Hermitian matrix $\tilde{\mathbf{A}}^{(k)}$, we define a SWAP-like operator $\mathbf{S}^{(k)}_{\tilde{A}} \in \mathbb{C}^{N^2\times N^2}$ based on the entries of $\tilde{\mathbf{A}}^{(k)}$:

$$\begin{aligned}
\mathbf{S}^{(k)}_{\tilde{A}} &:= \sum_{\ell,j=0}^{N-1} \tilde{\mathbf{A}}^{(k)}_{\ell j} \left|j\right\rangle\left\langle \ell\right| \otimes \left|\ell\right\rangle\left\langle j\right| \\
&= \sum_{i_k=0}^{n-1} \sum_{j_k=0}^{n^{m-1}-1} a'^{(k)}_{i_k j_k} \left|j_k + n\right\rangle\left\langle i_k\right| \otimes \left|i_k\right\rangle\left\langle j_k + n\right| + \overline{a'^{(k)}_{i_k j_k}} \left|j_k\right\rangle\left\langle j_k + n\right| \otimes \left|j_k + n\right\rangle\left\langle i_k\right| \\
&= \sum_{i_1,i_2,\ldots,i_m=0}^{n-1} a_{i_1 i_2 \ldots i_m} \left|i_{k+1}\cdots i_m i_1 \cdots i_{k-1} + n\right\rangle\left\langle i_k\right| \otimes \left|i_k\right\rangle\left\langle i_{k+1}\cdots i_m i_1 \cdots i_{k-1} + n\right| \\
&\quad + \overline{a_{i_1 i_2 \ldots i_m}} \left|i_k\right\rangle\left\langle i_{k+1}\cdots i_m i_1 \cdots i_{k-1} + n\right| \otimes \left|i_{k+1}\cdots i_m i_1 \cdots i_{k-1} + n\right\rangle\left\langle i_k\right|. \qquad (5.8)
\end{aligned}$$

This operator is one-sparse in a quadratically bigger space, i.e., there is no more than

one non-zero entry in every row and column, and its entries are efficiently computable. Therefore, the matrix exponentiation $e^{-\mathrm{i}\mathbf{S}_{\tilde{A}}^{(k)}\Delta t}$ is efficiently implemented [9].

The SWAP-like operator (5.8) combines the mode-k unfolding (3.8) and Hermitian extension (5.7), since they are all related to SWAP operations. It only requires access to the entries of original tensor $\mathcal{A}$.

For simplicity, in the following we use $\mathbf{A}$ to represent $\mathbf{A}^{(k)}$ when k is fixed. Let ρ_1 and ρ_2 be two distinct density matrices, where $\rho_1 = |\vec{1}\rangle\langle\vec{1}|$.

Lemma 5.1. [47] *By quantum principal component analysis (qPCA), the unitary* $e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t}$ *is simulated using* $\mathbf{S}_{\tilde{A}}$ *through*

$$\mathrm{tr}_1\{e^{-\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\rho_1 \otimes \rho_2 e^{\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\} \approx e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t}\rho_2 e^{\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t}. \tag{5.9}$$

Let ϵ_0 be the trace norm of the error term $O(\Delta t^2)$ in (5.9). For s steps, the resulting error is $\epsilon_1 = s\epsilon_0 \leqslant 2s\|\mathcal{A}\|_{\max}^2\Delta t^2$, where $\|\mathcal{A}\|_{\max} = \max_{i_1,\ldots,i_m}|a_{i_1\cdots i_m}|$. The simulated time is $t = s\Delta t$. Then,

$$\frac{\epsilon_1}{s} \leqslant 2\|\mathcal{A}\|_{\max}^2\left(\frac{t}{s}\right)^2. \tag{5.10}$$

Thus,

$$s = O\left(\frac{t^2}{\epsilon_1}\|\mathcal{A}\|_{\max}^2\right) \tag{5.11}$$

steps are required to simulate $e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t}$ if ϵ_1 and t are fixed.

Since we have assumed $\|\mathcal{A}\|_F = 1$, then $\|\mathcal{A}\|_{\max} = O(1)$. Hence, $s = O(t^2/\epsilon_1)$.

Applying the output in equation (5.9) again in the second register, we obtain

$$\begin{aligned}
&\mathrm{tr}_1\left\{e^{-\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\rho_1\otimes\left(\rho_2-\mathrm{i}\frac{\Delta t}{N}[\tilde{\mathbf{A}},\rho_2]+O(\Delta t^2)\right)e^{\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\right\}\\
=&\mathrm{tr}_1\{e^{-\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\rho_1\otimes\rho_2 e^{\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\}-\mathrm{i}\frac{\Delta t}{N}\mathrm{tr}_1\{e^{-\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}(\rho_1\otimes[\tilde{\mathbf{A}},\rho_2])e^{\mathrm{i}\mathbf{S}_{\tilde{A}}\Delta t}\}+O(\Delta t^2)\\
=&\rho_2-\mathrm{i}\frac{\Delta t}{N}[\tilde{\mathbf{A}},\rho_2]-\mathrm{i}\frac{\Delta t}{N}\mathrm{tr}_1\{\rho_1\otimes[\tilde{\mathbf{A}},\rho_2]\}+O(\Delta t^2)\\
=&\rho_2-\mathrm{i}\frac{2\Delta t}{N}[\tilde{\mathbf{A}},\rho_2]+O(\Delta t^2).
\end{aligned}\tag{5.12}$$

Thus, by continuously using k copies of ρ_1 we can simulate $e^{-\mathrm{i}(\tilde{\mathbf{A}}/N)k\Delta t}$.

For **Step 3**, we use the quantum phase estimation algorithm to estimate the eigenvalues of $e^{-\mathrm{i}(\tilde{\mathbf{A}}/N)\Delta t}$. First, we give a lemma explaining phase estimation.

Lemma 5.2. [42] *Let* $\mathbf{U}$ *be an* $n\times n$ *unitary operator, with eigenvectors* $|v_j\rangle$ *and eigenvalues* $e^{2\pi\mathrm{i}\theta_j}$ *for* $\theta_j\in[0,1)$, *i.e. we have* $\mathbf{U}|v_j\rangle=e^{2\pi\mathrm{i}\theta_j}|v_j\rangle$ *for* $j=0,1,\ldots,n-1$. *For a precision parameter* $\epsilon>0$, *there exists a quantum phase estimation algorithm that runs in time* $O(T(\mathbf{U})\log n/\epsilon)$ *and with probability* $1-1/\mathrm{poly}(n)$ *maps a state* $|b\rangle=\sum_{j=0}^{n-1}\alpha_j|v_j\rangle$ *to the state* $\sum_{j=0}^{n-1}\alpha_j|v_j\rangle|\bar{\theta}_j\rangle$ *such that* $\bar{\theta}_j\in\theta_j\pm\epsilon$ *for all* $j=0,1,\ldots,n-1$.

Theorem 5.1. *For the input* $|0\rangle^{\otimes d}|\vec{1}\rangle|b\rangle$, *by applying qPCA in Lemma 5.1 and phase estimation in Lemma 5.2, the superposition state (5.6)* $|\psi\rangle=\sum_{j=0}^{r-1}\beta_j|\tilde{\lambda}_j/N\rangle|\tilde{u}_j\rangle$ *is obtained, where* $|\tilde{\lambda}_j/N\rangle$ *is the estimated eigenvalue of* $\tilde{\mathbf{A}}/N$ *encoded in basis qubits.*

Proof. Given an initial quantum state

$$|0\rangle^{\otimes d}|\vec{1}\rangle|b\rangle=|\underbrace{00\cdots0}_{d}\rangle|\vec{1}\rangle|b\rangle\tag{5.13}$$

with $d=O(\lceil\log(1/\epsilon_2)\rceil)$ control qubits, where $|b\rangle$ is the superposition of eigenvectors

$|\tilde{u}_j\rangle$ corresponding to $\tilde{\lambda}_j$:

$$|b\rangle = \sum_{j=0}^{N-1} \beta_j |\tilde{u}_j\rangle, \qquad \sum_{j=0}^{N-1} |\beta_j|^2 = 1, \tag{5.14}$$

ϵ_2 is the accuracy for approximating the eigenvalues. Let $\rho_2 = |b\rangle\langle b|$. We first apply Hadamard operators to the first register, then the state (5.13) becomes

$$\frac{1}{\sqrt{2^d}} \sum_{\ell=0}^{2^d-1} |\ell\rangle |\vec{1}\rangle |b\rangle, \tag{5.15}$$

whose density matrix has the following form

$$\frac{1}{2^d} \sum_{\ell=0}^{2^d-1} |\ell\rangle\langle\ell| \otimes \rho_1 \otimes \rho_2. \tag{5.16}$$

Then we multiply $\sum_{\ell=0}^{2^d-1} |\ell\rangle\langle\ell| \otimes (e^{-\mathrm{i}\mathbf{S}_A\Delta t})^\ell$ and $\sum_{\ell=0}^{2^d-1} |\ell\rangle\langle\ell| \otimes (e^{\mathrm{i}\mathbf{S}_A\Delta t})^\ell$ to both sides of (5.16) to obtain

$$\sum_{\ell=0}^{2^d-1} |\ell\rangle\langle\ell| \otimes \left((e^{-\mathrm{i}\mathbf{S}_A\Delta t})^\ell \rho_1 \otimes \rho_2 (e^{\mathrm{i}\mathbf{S}_A\Delta t})^\ell\right). \tag{5.17}$$

Note that $d = O(\lceil \log(1/\epsilon_2) \rceil)$, after applying the operator 2^d times, the accumulation error is $O(1/\epsilon_2)$. Next, we perform a partial trace to the second register using (5.9) resulting in

$$\sum_{\ell=0}^{2^d-1} |\ell\rangle\langle\ell| \otimes \left(\left(e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t}\right)^\ell \rho_2 \left(e^{\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t}\right)^\ell\right). \tag{5.18}$$

After that, we apply the phase estimation algorithm in Lemma 5.2 to obtain the estimated eigenvalues of $\tilde{\mathbf{A}}/N$, since

$$\begin{aligned} e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t} |b\rangle &= \sum_{j=0}^{N-1} \beta_j e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}\Delta t} |\tilde{u}_j\rangle \\ &= \sum_{j=0}^{N-1} \beta_j e^{-\mathrm{i}\lambda_j(\frac{\tilde{\mathbf{A}}}{N})\Delta t} |\tilde{u}_j\rangle. \end{aligned} \tag{5.19}$$

At last, we implement the inverse quantum Fourier transform [51] and remove the first register, the final state (5.6)

$$|\psi\rangle = \sum_{j=0}^{r-1} \beta_j \, |\tilde{\lambda}_j/N\rangle \, |\tilde{u}_j\rangle$$

is obtained, where $|\tilde{\lambda}_j/N\rangle$ is the estimated eigenvalue of $\tilde{\mathbf{A}}/N$ encoded in basis qubits. The corresponding eigenvector $|\tilde{u}_j\rangle$ is proportional to $(\mathbf{u}_j; \pm\mathbf{v}_j) \in \mathbb{C}^N$, where $\mathbf{u}_j$ and $\mathbf{v}_j$ are the left and right singular vectors of $\tilde{\mathbf{A}}$, and the norm of each subvector $\mathbf{u}_j$ and $\mathbf{v}_j$ is $1/\sqrt{2}$, independent of their respective lengths n and n^{m-1}. □

For **Step 4**, since $\mathbf{A}$ is of size $n \times n^{m-1}$, $\mathbf{A}$ has at most n singular values $\{\sigma_j\}$. As a result, $\tilde{\mathbf{A}}$ has at most $2n$ nonzero eigenvalues $\tilde{\lambda}_j \in \{\pm\sigma_j\}$. Next, we measure the first register of state (5.6) in the computational basis $\{|0\rangle, \cdots, |2^d - 1\rangle\}$, all eigenpairs $|\tilde{\lambda}_j/N\rangle \, |\tilde{u}_j\rangle$ are obtained with probability $|\beta_j|^2$. Discarding the first register and projecting $|\tilde{u}_j\rangle$ onto the $\mathbf{u}_j$ part by using projection operators $\mathbf{P}_u = \sum_{i=0}^{n-1} |i\rangle\langle i|$ and $\mathbf{P}_v = \sum_{i=n}^{n^{m-1}+n-1} |i\rangle\langle i|$ result in $|\mathbf{u}_j\rangle$ with probability $\langle \tilde{u}_j | \mathbf{u}_j, 0\rangle = \frac{1}{2}$. Then, the singular matrix $\mathbf{U}$ is calculated by

$$\mathbf{U} = \sum_{j=1}^{n} |\mathbf{u}_j\rangle\langle j| \, . \tag{5.20}$$

Repeating measurements with the initial state $|b\rangle = |0\rangle, |1\rangle, \cdots, |n-1\rangle$ and applying amplitude amplification [1], we can obtain all the singular vectors in $T_U = O(n^{3/2})$ times with probability close to 1. Thus, the singular matrix $\mathbf{U}^{(k)}$ is reconstructed.

For **Step 5**, after we get all $\mathbf{U}^{(k)}$ for $k = 1, 2, \ldots, m$, in this step we calculate the core tensor $\mathcal{S}$:

$$\mathcal{S} = \mathcal{A} \times_1 \mathbf{U}^{(1)^\dagger} \times_2 \mathbf{U}^{(2)^\dagger} \times_3 \cdots \times_m \mathbf{U}^{(m)^\dagger}. \tag{5.21}$$

Here, the calculation is accelerated by the quantum tensor-matrix multiplication, which is similar to the quantum matrix multiplication algorithm by swap test [69].

We may calculate $\mathcal{A} \times_k \mathbf{U}^{(k)^\dagger}$ through the following state

$$\frac{1}{\|\mathcal{A}\|_F \|\mathbf{U}^{(k)}\|_F} \sum_{i_1,\ldots,i_{k-1},j_k,i_{k+1},\ldots,i_m=0}^{n-1} \|\mathbf{U}^{(k)}_{\bullet j_k}\|_2 \|\mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m}\|_2 \langle \mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m} | \mathbf{U}^{(k)}_{\bullet j_k} \rangle$$
$$|i_1,\cdots,i_{k-1},j_k,i_{k+1},\cdots,i_m\rangle |0\rangle + |0\rangle^\perp, \tag{5.22}$$

where $|\mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m}\rangle$ is an n-level quantum state (i.e., n-dimensional vector) if $i_1,\ldots,i_{k-1},i_{k+1},\ldots,i_m$ are all fixed.

By (5.22), the success probability is

$$\sum_{i_1,\ldots,i_{k-1},j_k,i_{k+1},\ldots,i_m=0}^{n-1} \|\mathbf{U}^{(k)}_{\bullet j_k}\|_2^2 \|\mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m}\|_2^2 \langle \mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m} | \mathbf{U}^{(k)}_{\bullet j_k} \rangle^2 \Big/ \left(\|\mathcal{A}\|_F^2 \|\mathbf{U}^{(k)}\|_F^2\right)$$
$$= \frac{\|\mathcal{A} \times_k \mathbf{U}^{(k)^\dagger}\|_F^2}{\|\mathcal{A}\|_F^2 \|\mathbf{U}^{(k)}\|_F^2}. \tag{5.23}$$

Note that unitary matrices preserve norms, and we have assumed that $\|\mathcal{A}\|_F = 1$, therefore

$$\|\mathcal{A} \times_k \mathbf{U}^{(k)^\dagger}\|_F = \|\mathcal{A}\|_F = 1. \tag{5.24}$$

Thus, after post-selecting $|0\rangle$, the state (5.22) becomes

$$|\Psi^{(k)}\rangle := \sum_{i_1,\ldots,i_{k-1},j_k,i_{k+1},\ldots,i_m=0}^{n-1} \|\mathbf{U}^{(k)}_{\bullet j_k}\|_2 \|\mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m}\|_2 \langle \mathcal{A}_{i_1\cdots i_{k-1}\bullet i_{k+1}\cdots i_m} | \mathbf{U}^{(k)}_{\bullet j_k} \rangle$$
$$|i_1,\cdots,i_{k-1},j_k,i_{k+1},\cdots,i_m\rangle. \tag{5.25}$$

$|\Psi^{(k)}\rangle$ corresponds to the tensor $\mathcal{A} \times_k \mathbf{U}^{(k)^\dagger}$, whose amplitudes are exactly the entries of tensor $\mathcal{A} \times_k \mathbf{U}^{(k)^\dagger}$.

After applying amplitude amplification [1], the computational complexity is

$$T_M = \tilde{O}\left(\frac{\|\mathcal{A}\|_F \|\mathbf{U}^{(k)}\|_F}{\epsilon_3 \|\mathcal{A} \times_k \mathbf{U}^{(k)^\dagger}\|_F}\right)$$
$$= \tilde{O}\left(\frac{\|\mathbf{U}^{(k)}\|_F}{\epsilon_3}\right) = \tilde{O}\left(\frac{\sqrt{n}}{\epsilon_3}\right) \tag{5.26}$$

to accuracy ϵ_3.

Repeating the multiplication between $\mathcal{A}$ and $\mathbf{U}^{(k)^\dagger}$ for $k = 1, 2, \ldots, m$, we obtain the final state

$$|\Psi_f\rangle = \sum_{j_1,\cdots,j_m=0}^{n-1} s_{j_1\cdots j_m} |j_1, \cdots, j_m\rangle, \tag{5.27}$$

corresponding to the core tensor $\mathcal{S}$. The total complexity is $\tilde{O}\left(\frac{m\sqrt{n}}{\epsilon_3}\right)$.

Without loss of generality, we can regard the accuracy of matrix exponentiation, phase estimation and tensor-matrix multiplication as the same value, i.e. $\epsilon_1 = \epsilon_2 = \epsilon_3 =: \epsilon$.

5.4 Q-HOSVD Algorithm 2

For Algorithm 1 to be efficiently implemented, the unfolding matrices are required to be low-rank. This result is analyzed in the next section for complexity analysis. However, in some cases the input may not have such good structure. In this section we propose an alternative quantum HOSVD algorithm which is based on quantum singular value estimation (QSVE) [39]. In this method, the input is a general matrix, not required to be sparse, low-rank or well-conditioned.

Recall that in Chapter 3, a tree data structure with quantum access is introduced in Lemma 3.1, where the quantum states are efficiently prepared corresponding to the rows and columns of matrices. Based on this data structure, a fast quantum algorithm to perform singular value estimation stated in Lemma 5.3 is designed.

For a matrix $\mathbf{A} \in \mathbb{C}^{N_1 \times N_2}$, in the data structure,

$$\mathbf{U}_P = \sum_{i=0}^{N_1-1} |i\rangle |\mathbf{A}_i\rangle\langle i| \langle 0|, \quad \mathbf{U}_Q = \sum_{j=0}^{N_2-1} |\hat{\mathbf{A}}\rangle |j\rangle\langle 0| \langle j|. \tag{5.28}$$

It has been shown in Section 3.1 that $\mathbf{U}_P$ and $\mathbf{U}_Q$ are implemented through $\mathbf{R}_y$

rotation operators. Then they can be expressed as

$$\mathbf{U}_P = \sum_{i=0}^{N_1-1} |i\rangle\langle i| \otimes \mathbf{U}_i,$$

$$\mathbf{U}_Q = \sum_{j=0}^{N_2-1} \mathbf{U}_q \otimes |j\rangle\langle j| = \mathbf{U}_q \otimes \mathbf{I}_{N_2}, \tag{5.29}$$

where $\mathbf{U}_i |0\rangle = |\mathbf{A}_i\rangle$ for $i = 0, \ldots, N_1 - 1$, and $\mathbf{U}_q |0\rangle = |\tilde{\mathbf{A}}\rangle$, $\mathbf{U}_i$ and $\mathbf{U}_q$ are composed of rotation operators.

Define two isometries $\mathbf{P} \in \mathbb{C}^{N_1 N_2 \times N_1}$ and $\mathbf{Q} \in \mathbb{C}^{N_1 N_2 \times N_2}$ related to $\mathbf{U}_P$ and $\mathbf{U}_Q$:

$$\mathbf{P} = \sum_{i=0}^{N_1-1} |i\rangle |\mathbf{A}_i\rangle\langle i|, \quad \mathbf{Q} = \sum_{j=0}^{N_2-1} |\hat{\mathbf{A}}\rangle |j\rangle\langle j|. \tag{5.30}$$

It can be proved that $\mathbf{P}^\dagger\mathbf{P} = \mathbf{I}_{N_1}, \mathbf{Q}^\dagger\mathbf{Q} = \mathbf{I}_{N_2}$, $2\mathbf{P}\mathbf{P}^\dagger - \mathbf{I}_{N_1 N_2}$ is unitary and it can be efficiently implemented in time $O(\text{polylog}(N_1 N_2))$ in the form of $\mathbf{U}_P$ and $\mathbf{U}_Q$. Actually,

$$\begin{aligned} 2\mathbf{P}\mathbf{P}^\dagger - I_{N_1 N_2} &= 2\sum_i |i\rangle |\mathbf{A}_i\rangle\langle i| \langle \mathbf{A}_i| - \mathbf{I}_{N_1 N_2} \\ &= \mathbf{U}_P \mathbf{G}_P \mathbf{U}_P^\dagger, \end{aligned} \tag{5.31}$$

where $\mathbf{G_P} := 2\sum_i |i\rangle |0\rangle\langle i| \langle 0| - \mathbf{I}_{N_1 N_2}$ is a reflection. Similarly, $2\mathbf{Q}\mathbf{Q}^\dagger - \mathbf{I}_{N_1 N_2} = \mathbf{U}_Q \mathbf{G}_Q \mathbf{U}_Q^\dagger$, where $\mathbf{G}_Q := 2\sum_j |0\rangle |j\rangle\langle 0| \langle j| - \mathbf{I}_{N_1 N_2}$.

Now denote

$$\begin{aligned} \mathbf{W} &= (2\mathbf{P}\mathbf{P}^\dagger - \mathbf{I}_{N_1 N_2})(2\mathbf{Q}\mathbf{Q}^\dagger - \mathbf{I}_{N_1 N_2}) \\ &= \mathbf{U}_P \mathbf{G}_P \mathbf{U}_P^\dagger \mathbf{U}_Q \mathbf{G}_Q \mathbf{U}_Q^\dagger. \end{aligned} \tag{5.32}$$

All the factors are unitary operators. The eigenvalue of $\mathbf{W}$ is $e^{\mathrm{i}\theta_i}$ such that

$$\cos\left(\frac{\theta_i}{2}\right) = \frac{\sigma_i}{\|\mathbf{A}\|_F}, \tag{5.33}$$

where σ_i is the singular value of $\mathbf{A}$. Then, $\mathbf{W}$ is used for phase estimation to obtain the singular values. The result of QSVE is summarized in the following lemma:

Lemma 5.3. (Quantum singular value estimation) [39]

Let $\mathbf{A} \in \mathbb{R}^{N_1 \times N_2}$ *stored in the data structure stated in Lemma 3.1, and the singular value decomposition of* $\mathbf{A}$ *can be written as* $\mathbf{A} = \sum_{\ell=0}^{r-1} \sigma_\ell \left|u_\ell\right\rangle\left\langle v_\ell\right|$, *where* $r = \min(N_1, N_2)$. *Let* $\epsilon > 0$ *be the precision parameter. Then the quantum singular value estimation runs in* $O(\mathrm{polylog}(N_1 N_2)/\epsilon)$ *and achieves* $\sum_i \beta_i \left|v_i\right\rangle\left|0\right\rangle \mapsto \sum_i \beta_i \left|v_i\right\rangle\left|\overline{\sigma}_i\right\rangle$, *and* $|\overline{\sigma}_i - \sigma_i| \leqslant \epsilon \|\mathbf{A}\|_F$ *for all* i *with probability at least* $1 - 1/\mathrm{poly}(N_2)$.

The circuit of QSVE on a single matrix is shown in Fig. 5.3. In Fig. 5.3, $\mathbf{U}_Q \left|b\right\rangle$ is an superposition of eigenstates of $\mathbf{W}$. The second register has t qubits, indicating that the accuracy is 2^{-t}, and j takes the value of $2^0, 2^1, \ldots, 2^{t-1}$ respectively. $\mathbf{U}_f$ maps the eigenvalues of $\mathbf{W}$ to the singular values of $\mathbf{A}$ through equation (5.33). The output is the superposition state of estimated singular values and the corresponding right singular vectors. Since in HOSVD we aim to obtain the left singular vectors, we can obtain the left singular vectors $\left|u_i^{(k)}\right\rangle$ of $\mathbf{A}^{(k)}$ from the right singular vectors of its conjugate transpose $\mathbf{A}^{(k)^\dagger}$. Thus, we perform the QSVE on the unfolding matrix $\mathbf{A}^{(k)^\dagger}$. The Q-HOSVD algorithm based on QSVE is given in Algorithm 2.

Algorithm 2 Q-HOSVD by QSVE

Input: $\mathcal{A} \in \mathbb{C}^{I_1 \times \cdots \times I_m}, \epsilon, \left|b\right\rangle$
Output: $\mathcal{S}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(m)}$

1. Prepare the initial state $\dfrac{1}{\sqrt{m}} \sum_{k=0}^{m-1} \left|b\right\rangle\left|0\right\rangle\left|k\right\rangle$.
2. Implement the controlled-k QSVE by the last register to obtain the superposition state
$$\left|\psi\right\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} \sum_{i=0}^{r-1} \beta_i^{(k)} \left|u_i^{(k)}\right\rangle\left|\overline{\sigma}_i^{(k)}\right\rangle\left|k\right\rangle. \tag{5.34}$$
3. Post-select k and perform measurement on $\left|\overline{\sigma}_i^{(k)}\right\rangle$ and extract $\left|u_i^{(k)}\right\rangle$ to compose $\mathbf{U}^{(k)}$.
4. $\mathcal{S} \leftarrow \mathcal{A} \times_1 \mathbf{U}^{(1)^\dagger} \times_2 \mathbf{U}^{(2)^\dagger} \times_3 \cdots \times_m \mathbf{U}^{(m)^\dagger}$.

Algorithm 2 is similar to Algorithm 1, but we do not need to apply phase estimation on the extended Hermitian matrices.

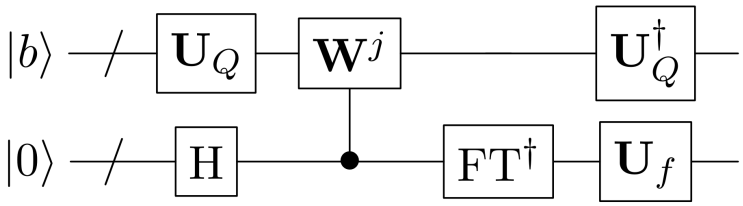


Figure 5.3: Circuit of QSVE on a matrix $\mathbf{A}$.

For **Step 1**, we prepare the initial state

$$|\psi_0\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |b\rangle |0\rangle |k\rangle, \tag{5.35}$$

where the first register could be any state and always expressed as a superposition of singular vectors, the second register stores the estimated singular values after phase estimation, and the last register is the index for mode-k unfolding.

For **Step 2**, assume that the tensor $\mathcal{A}$ is mth-order n-dimensional for simplicity. Recall that for fixed k, $|j_k\rangle = |i_{k+1} \cdots i_m i_1 \cdots i_{k-1}\rangle$ as the same in (3.8), where $j_k = 0, \ldots, n^{m-1} - 1$. Denote $|\mathbf{A}_{j_k}\rangle$ be the tube $|\mathcal{A}_{i_1 \cdots i_{k-1} \bullet i_{k+1} \cdots i_m}\rangle$. Different from Algorithm 1, we directly prepare the mode-k unfolding matrix through the unitary operators $\mathbf{U}_P^{(k)}$ and $\mathbf{U}_Q^{(k)}$ as in Lemma 3.1 according to the mode of unfolding:

$$\begin{aligned}
&\mathbf{U}_P^{(k)} : |j_k\rangle |0\rangle \to |j_k\rangle |\overline{\mathbf{A}}_{j_k}\rangle = \frac{1}{\|\mathbf{A}_{j_k}\|_2} \sum_{i_k=0}^{n-1} \overline{a}_{i_1 i_2 \cdots i_m} |j_k\rangle |i_k\rangle \\
&\qquad \text{for} \quad j_k = 0, \cdots, n^{m-1} - 1; \\
&\mathbf{U}_Q^{(k)} : |0\rangle |i_k\rangle \to |\hat{\mathbf{A}}_k\rangle |i_k\rangle = \frac{1}{\|\mathcal{A}\|_F} \sum_{j_k=0}^{n^{m-1}-1} \|\mathbf{A}_{j_k}\|_2 |j_k\rangle |i_k\rangle \\
&\qquad \text{for} \quad i_k = 0, \cdots, n - 1.
\end{aligned} \tag{5.36}$$

By this way, the above two operators are prepared in time $O(m \,\mathrm{polylog} n)$, corresponding to $\mathbf{A}^{(k)^\dagger}$. Then we implement the controlled-k singular value estimation in

parallel by applying the operator $\sum_{k=0}^{m-1} \mathbf{W}^{(k)} \otimes |k\rangle\langle k|$ on the initial state $|\psi_0\rangle$. Finally, we undo the phase estimation and apply the inverse of operator $\mathbf{U}_P^{(k)}$, obtaining the state (5.34)

$$|\psi\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} \sum_{i=0}^{r-1} \beta_i^{(k)} \, |u_i^{(k)}\rangle \, |\overline{\sigma}_i^{(k)}\rangle \, |k\rangle \, .$$

For the last two steps, similar to the Steps 4 and 5 in Algorithm 1, we can make measurements and obtain the singular matrices and core tensor.

5.5 Complexity Analysis

For simplicity, we assume the input is an mth-order n-dimensional tensor.

For the first Q-HOSVD algorithm, the computational complexity mainly comes from data access, matrix exponential simulation and phase estimation. The data input time is $O(m\,\mathrm{polylog}n)$. At a simulation time t, only the eigenvalues of $\tilde{\mathbf{A}}^{(k)}/N$ with $|\tilde{\lambda}_j|/N = \Omega(1/t)$ matter [67], and the eigenvalues smaller than ϵ are omitted. Note that $\mathbf{A}^{(k)}$ is an $n \times n^{m-1}$ matrix. For a fixed k, let the number of these eigenvalues be $r \leqslant 2n$, then by

$$\begin{cases} \mathrm{tr}\{\tilde{\mathbf{A}}_r^2/N^2\} = \sum_{j=0}^{r-1} \tilde{\lambda}_j^2/N^2 = \Omega(r/t^2), \\ \mathrm{tr}\{\tilde{\mathbf{A}}_r^2/N^2\} \leqslant \mathrm{tr}\{\tilde{\mathbf{A}}^2/N^2\} = \|\tilde{\mathbf{A}}\|_F^2/N^2 \leqslant \|\mathcal{A}\|_{\max}^2, \end{cases} \tag{5.37}$$

we find that the rank of the effectively simulated matrix is $r = O(\|\mathcal{A}\|_{\max}^2 t^2)$. By (5.11), there are $O\left(t^2\|\mathcal{A}\|_{\max}^2/\epsilon\right)$ steps required to simulate $e^{-\mathrm{i}\frac{\tilde{\mathbf{A}}}{N}t}$, where $\|\mathcal{A}\|_{\max} = O(1)$, and $1/\epsilon$ can be chosen as $O(\mathrm{polylog}n)$. To make this algorithm efficient, $t = O(\mathrm{polylog}n)$, then the rank $r = O(\mathrm{polylog}n)$, i.e. the matrices have to be low-rank. Thus, the time to implement phase estimation is $O(\mathrm{polylog}n \cdot \log(n)/\epsilon)$. Therefore, the total computational complexity of obtaining (5.6) is $O(m^2\mathrm{polylog}n)$.

For the second Q-HOSVD algorithm, for each unfolding the time to access the data structure is $O(m\,\mathrm{polylog}n)$, and the time to implement QSVE is $O(m\,\mathrm{polylog}n/\epsilon)$,

so the total complexity of obtaining (5.34) is $O(m^3 \text{polylog} n)$. Furthermore, there are no requirement for the structure of the input tensor.

Usually, in practical problems, $m \ll n$. Thus, we can omit the order m, and the algorithm runs polylogarithmically in the dimensions.

If we want to obtain the singular matrices and core tensor explicitly in the quantum register, we need to make measurements on the states (5.6) and (5.34), and reconstruct singular matrices and finally calculate the core tensor by quantum tensor matrix multiplication, the complexities are $O(m^3 n^2 \text{polylog} n)$ and $O(m^4 n^2 \text{polylog} n)$ respectively.

5.6 Application on Recommendation Systems

In [74], the authors make use of HOSVD for tag recommendations. Given an initial third-order tensor with usage data triplets (user, item, tag), they implement HOSVD and do truncations to obtain the core tensor and reconstructed tensor with smaller dimensions. Then, based on the entries of the reconstructed tensor, the tags are recommended to users. We have carried out the similar SVD and truncation operations in [81] by another tensor decomposition called t-svd.

In this section, we introduce a *hybrid* quantum-classical recommendation method for context-aware collaborative filtering (CF) based on tensor factorization (TF), named as multiverse recommendation [38]. TF is an extension of matrix factorization (MF) to multiple dimensions. HOSVD is chosen as our TF approach to analyze the recommendation systems, due to its relevance among the different categories. Given the known preference tensor, we use HOSVD model to find out the missing information. This problem is well-known as the completion problem in recommendation systems [72]. Our contribution is designing a *hybrid* quantum-classical recommendation algorithm to accelerate this process.

Context has been universally acknowledged as an important factor for analyzing recommendation systems. A pair (user, item) is extended to a triplet (user, item, context) or even larger multiplets, where context denotes the factor that may influence a user's preference on a specified item, e.g. time, location, and we consider the interactions among them. Generally, any number of contexts can be added to this recommendation system, and the correlation is described by a relevance score function f_{rs} as follows:

$$\begin{aligned} f_{rs} : &\text{User} \times \text{Item} \times \text{Context}_1 \times \cdots \times \text{Context}_\ell \\ &\longrightarrow \text{Relevance Score} \end{aligned} \tag{5.38}$$

with ℓ different contexts, so that the total number of dimensions is $\ell + 2$. Thus, we can use a tensor with order $m = \ell + 2$ to express the set of relevance scores, and utilize HOSVD model to describe it. Such methods are widely applied in recommendation systems like Netflix prize problems [7] and so on.

Denote the given preference tensor $\mathcal{Y} \in \{0, \cdots, 5\}^{I_1 \times I_2 \times \cdots \times I_m}$ containing the observed ratings ranging from 1 to 5, and value 0 indicates that the item has not been rated yet. The aim is to find out such missing values and give a good recommendation to users. Denote the factor matrices $\mathbf{U} \in \mathbb{R}^{I_1 \times d_1}$, $\mathbf{M} \in \mathbb{R}^{I_2 \times d_2}$, $\mathcal{C}^{(k)} \in \mathbb{R}^{I_{k+2} \times d_{k+2}}$, for $k = 1, \ldots, \ell$. Then, $\mathcal{S} \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_m}$. Let $\mathbf{d} = [d_1, d_2, \cdots, d_m]^\top$, and $d = \max_{j \in [m]} d_j$.

To obtain the recommendations based on HOSVD, we design a loss function and optimize over it. The loss function is characterized as $L(\mathcal{T}(\theta), \mathcal{Y})$, where θ is the model parameter, i.e., $\theta := (\mathcal{S}, \mathbf{U}, \mathbf{M}, \mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \cdots, \mathcal{C}^{(\ell)})$. Denote a set $D := \{(i_1, i_2, \cdots, i_m) \mid y_{i_1 i_2 \cdots i_m} > 0\}$ an observation history, and $K := |D|$ the number of observed ratings. The total loss function is defined as

$$L(\mathcal{T}(\theta), \mathcal{Y}) := \frac{1}{\|\mathcal{S}\|_1} \sum_{(i_1, i_2, \ldots, i_m) \in D} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right), \tag{5.39}$$

which only applies on the observed values in $\mathcal{Y}$. Function $l(t, y)$ is a pointwise loss function, that can be based on l_2 norm, e.g., $l(t, y) = (t - y)^2/2$, or other types of distance measure. By adding a regularization term to avoid overfitting, we establish the objective function

$$J(\theta) := L(\mathcal{T}(\theta), \mathcal{Y}) + \Omega(\theta) \tag{5.40}$$

with trivial regularizers

$$\Omega(\theta) = \lambda_1 \|\mathbf{U}\|_F^2 + \lambda_2 \|\mathbf{M}\|_F^2 + \sum_{k=1}^{\ell} \lambda_{k+2} \|\mathcal{C}^{(k)}\|_F^2 + \lambda_{\mathcal{S}} \|\mathcal{S}\|_F^2. \tag{5.41}$$

Usually, the parameters of matrices can be chosen as the same value, i.e., $\lambda_1 = \lambda_2 = \cdots = \lambda_m =: \lambda$.

We optimize these matrices and the core tensor by stochastic gradient descent (SGD) method [11]. SGD randomly picks a sample at one time and perform gradient descent. It usually compares with batch gradient descent (BGD) which runs over all the samples each iteration. BGD converges globally in every step but it is computationally prohibitive for our problem. The cost of SGD is low, but it usually converges in a local minimum. For big data problems, SGD often converges without running over all the samples. The whole tensor completion algorithm based on the HOSVD model is given in Algorithm 3.

Algorithm 3 Tensor Completion by HOSVD

Input: $\mathcal{Y}, \mathbf{d}, \eta, \lambda, \lambda_{\mathcal{S}}$

Output: $\mathcal{S}, \mathbf{U}, \mathbf{M}, \mathcal{C}^{(1)}, \cdots, \mathcal{C}^{(\ell)}$

Initialize $\mathbf{U}, \mathbf{M}, \mathcal{C}^{(1)}, \cdots, \mathcal{C}^{(\ell)}, \mathcal{S}$ with small values, $\mathcal{T}$ with all zeros.

while $(i_1, i_2, \ldots, i_m) \in D$ **do**

$t_{i_1 i_2 \cdots i_m} = \mathcal{S} \times_1 \mathbf{U}_{i_1 \bullet} \times_2 \mathbf{M}_{i_2 \bullet} \times_3 \mathcal{C}^{(1)}_{i_3 \bullet} \times_4 \cdots \times_m \mathcal{C}^{(\ell)}_{i_m \bullet}$

$\mathbf{U}_{i_1 \bullet} \longleftarrow \mathbf{U}_{i_1 \bullet} - \eta\lambda \mathbf{U}_{i_1 \bullet} - \eta \partial_{\mathbf{U}_{i_1 \bullet}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right)$

$\mathbf{M}_{i_2 \bullet} \longleftarrow \mathbf{M}_{i_2 \bullet} - \eta\lambda \mathbf{M}_{i_2 \bullet} - \eta \partial_{\mathbf{M}_{i_2 \bullet}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right)$

$\mathcal{C}^{(1)}_{i_3 \bullet} \longleftarrow \mathcal{C}^{(1)}_{i_3 \bullet} - \eta\lambda \mathcal{C}^{(1)}_{i_3 \bullet} - \eta \partial_{\mathcal{C}^{(1)}_{i_3 \bullet}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right)$

$\vdots$

$\mathcal{C}^{(\ell)}_{i_m \bullet} \longleftarrow \mathcal{C}^{(\ell)}_{i_m \bullet} - \eta\lambda \mathcal{C}^{(\ell)}_{i_m \bullet} - \eta \partial_{\mathcal{C}^{(\ell)}_{i_m \bullet}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right)$

$\mathcal{S} \longleftarrow \mathcal{S} - \eta\lambda_{\mathcal{S}} \mathcal{S} - \eta \partial_{\mathcal{S}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right)$

end while

Algorithm 3 can be considered as a training method by SGD. After we obtain the factor matrices and core tensor, $\mathcal{T}$ is computed explicitly by

$$\mathcal{T} = \mathcal{S} \times_1 \mathbf{U} \times_2 \mathbf{M} \times_3 \mathcal{C}^{(1)} \times_4 \cdots \times_m \mathcal{C}^{(\ell)} \tag{5.42}$$

as an approximation of the preference tensor $\mathcal{Y}$ and we give recommendations to users according to the entries of $\mathcal{T}$.

This algorithm is a *hybrid* quantum-classical algorithm. The computation of gradients is accelerated by some quantum subroutines, and the rest procedures are performed by classical computers. The gradients are, e.g.,

$$\partial_{\mathbf{U}_{i_1 \bullet}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right) = \partial_{t_{i_1 i_2 \cdots i_m}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right) \mathcal{S} \times_2 \mathbf{M}_{i_2 \bullet} \times_3 \cdots \times_m \mathcal{C}^{(\ell)}_{i_m \bullet}, \tag{5.43}$$

$$\partial_{\mathcal{S}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right) = \partial_{t_{i_1 i_2 \cdots i_m}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right) \mathbf{U}_{i_1 \bullet} \circ \mathbf{M}_{i_2 \bullet} \circ \cdots \circ \mathcal{C}^{(\ell)}_{i_m \bullet}. \tag{5.44}$$

For gradient (5.43), $\partial_{t_{i_1 i_2 \cdots i_m}} l\left(t_{i_1 i_2 \cdots i_m}, y_{i_1 i_2 \cdots i_m}\right)$ is a simple function, if the loss function takes l_2 norm, then $\partial_t l(t, y) = t - y$. Define $\mathbf{g}(\mathbf{U}_{i_1 \bullet}) = \mathcal{S} \times_2 \mathbf{M}_{i_2 \bullet} \times_3 \cdots \times_m \mathcal{C}^{(\ell)}_{i_m \bullet}$. The entry $\mathbf{g}_j(\mathbf{U}_{i_1 \bullet})$ is equivalent to

$$\mathcal{S}_{i_1 = j} \cdot (\mathbf{M}_{i_2 \bullet} \circ \cdots \circ \mathcal{C}^{(\ell)}_{i_m \bullet}) =: \mathcal{S}_{i_1 = j} \cdot \mathcal{Z}, \tag{5.45}$$

the inner product of two $(m-1)$th order tensors. By quantum matrix multiplication algorithm [69], the outer product of two vectors $|a\rangle$ and $|b\rangle$ can be performed in time $O(\|a\|_F^3\|b\|_F^3\text{polylog}n/\epsilon_4\|a\circ b\|_F^3) = O(\text{polylog}n/\epsilon_4)$ to accuracy ϵ_4, which holds for classical vectors. Thus, we can perform the quantum outer product $\mathbf{M}_{i_2\bullet}\circ\cdots\circ\mathcal{C}^{(\ell)}_{i_m\bullet}$ in time $O(\text{polylog}n/\epsilon_4)$. The resulting state is

$$|z\rangle = \frac{1}{\|\mathcal{Z}\|_F}\sum_{i_2,i_3,\ldots,i_m=0}^{n-1} z_{i_2i_3\cdots i_m}|i_2\rangle|i_3\rangle\cdots|i_m\rangle. \tag{5.46}$$

Next, we load the subtensor $\mathcal{S}_{i_1=j}$ into the quantum register as

$$|s\rangle = \frac{1}{\|\mathcal{S}_{i_1=j}\|_F}\sum_{i_2,i_3,\ldots,i_m=0}^{n-1} s_{ji_2i_3\cdots i_m}|i_2\rangle|i_3\rangle\cdots|i_m\rangle. \tag{5.47}$$

Then, we can construct the following superposition state:

$$\begin{aligned}|\phi_1\rangle &= \frac{1}{\sqrt{2}}(|+\rangle|s\rangle + |-\rangle|z\rangle)\\ &= \sin\theta|0\rangle|u\rangle + \cos\theta|1\rangle|v\rangle\end{aligned} \tag{5.48}$$

with $\cos\theta = \sqrt{(1-\langle s|z\rangle)/2}$. Then, by applying quantum amplitude estimation algorithm [13], we can obtain h such that

$$\left|\frac{1-\langle s|z\rangle}{2} - h\right| \leqslant \epsilon_5 \tag{5.49}$$

in time $O(m\log d/\epsilon_5)$. Therefore, $\|\mathcal{S}_{i_1=j}\|_F\|\mathcal{Z}\|_F(1-2h)$ gives a $2\epsilon_5\|\mathcal{S}_{i_1=j}\|_F\|\mathcal{Z}\|_F$-approximate of $\mathbf{g}(\mathbf{U}_{i_1\bullet})$. Then, if we take $\epsilon = 2\epsilon_5\|\mathcal{S}_{i_1=j}\|_F\|\mathcal{Z}\|_F$, we can obtain the value of $\mathbf{g}_j(\mathbf{U}_{i_1\bullet})$ in time $O(m\log d\|\mathcal{S}_{i_1=j}\|_F\|\mathcal{Z}\|_F/\epsilon)$ to accuracy ϵ. Since the gradient $\mathbf{g}$ has d entries, and we have to repeat the above procedure for all the singular matrices, then the total complexity of matrix optimization is $O(Km^2d\text{polylog}d)$. Compared to the corresponding classical algorithm which takes $O(Kmd^m)$ classical calculations, our quantum algorithm is exponentially faster. To calculate the core tensor $\mathcal{S}$, we can directly use the classical computation, the complexity is $O(Kd^m)$.

Chapter 6

Quantum Barzilai-Borwein Gradient Method

6.1 Introduction

Optimization is an important tool in analyzing data. Gradient descent method and Newton's method are the two most commonly used optimization algorithms. Standard gradient descent method is a first-order iterative algorithm for finding a local minimum (or maximum) of a function. This method is simple, but usually converges slowly. Furthermore, for ill-conditioned case in the sense that the condition number of the Hessian matrix is very large, the progress is extremely slow. Newton's method is a second-order algorithm, which converges quadratically with fewer steps than gradient descent method. However, it is very time-consuming in calculating the inverse of the Hessian matrix. Research has been done for long time on how to balance the computational cost and convergence speed. In 1988, Barzilai and Borwein found a linear method that significantly outperforms the standard gradient descent method with nearly no extra cost. The only difference between their method and the standard one is the choice of step sizes. They designed two specific step sizes and the algorithm is proved to converge Q-superlinearly for convex quadratic functions [64]. This method is named as two-point step size gradient method [5], and also known as

the BB gradient method. After the BB gradient method is proposed, many follow up works spread this method to a broader area. Now it has been successfully applied in several areas, such as compressed sensing [27], support vector machine in machine learning [20], image processing [82], stochastic linear complementarity problems [36] and etc.

Quantum optimization is an indispensable part of quantum machine learning theory. However, so far there are only a few quantum optimization algorithms. In 2017, quantum gradient descent and Newton's method for constrained polynomial optimization [66] are raised based on the HHL algorithm. The objective function is an even order polynomial. The authors consider the coefficients as a huge matrix and then decompose it into a sum of tensor products among smaller matrices. The iteration functions of the gradient descent method and Newton's method are obtained by using several quantum techniques. In contrast to the classical counterpart, for each iteration of the quantum methods, multiple copies of the current step are consumed to proceed to the next step. Thus, the above quantum algorithms scale exponentially in the number of iterations. The authors think that it is acceptable in cases when the optimal solution is reached in only a few steps. If this condition is satisfied, the runtimes of the two quantum optimization algorithms are $O(\text{polylog}N)$, where N is the number of variables. They offer an exponential speedup over their classical counterparts. In [40], the authors introduce a quantum gradient descent method requiring time linear to the number of iterations when the gradient is an affine function.

Inspired by [66], we propose the quantum Barzilai-Borwein gradient methods, and apply them to polynomial optimization with a unit norm constraint. We reformulate the coefficients of the objective function as a tensor, and use tensor decompositions to simplify the calculations. Our quantum methods run polylogarithmically in the dimension of the solution state, and provide exponential improvements over the

classical counterpart.

6.2 Problem Statement

The objective function we want to minimize is a polynomial of order m defined over $\mathbf{x} \in \mathbb{R}^N$,

$$f(\mathbf{x}) = \sum_{i_1,\ldots,i_m=1}^{N} a_{i_1\cdots i_m} x_{i_1} \cdots x_{i_m}, \tag{6.1}$$

with N^m coefficients $a_{i_1\cdots i_m} \in \mathbb{R}$ and $\mathbf{x} = (x_1, ..., x_N)^\top$. The coefficients $a_{i_1\cdots i_m}$ compose an mth-order N-dimensional tensor. From equation (2.12), we know that the objective function can be rewritten as the inner product between $\mathcal{A}$ and $\mathbf{x}^m$:

$$f(\mathbf{x}) = \mathcal{A}\mathbf{x}^m. \tag{6.2}$$

Without loss of generality, we assume $\|\mathcal{A}\|_F = 1$.

If $\mathcal{A}$ is not symmetric, we may symmetrize $\mathcal{A}$ with respect to all modes so that $\bar{\mathcal{A}}$ is symmetric with respect to any permutation of indices. By this symmetrization, we have

$$f(\mathbf{x}) = \mathcal{A}\mathbf{x}^m = \bar{\mathcal{A}}\mathbf{x}^m. \tag{6.3}$$

Since $\mathcal{A}$ can always be symmetrized in this optimization problem, we assume in the following $\mathcal{A}$ is a symmetric tensor.

Recall that the symmetric tensor can be decomposed by symmetric CP decomposition and symmetric HOSVD through equations (2.16) and (5.5). Therefore, for CP decomposition, the gradient of the objective function (6.1) at point $\mathbf{x}$ can be written as

$$\nabla f(\mathbf{x}) = \sum_{j=1}^{r} \alpha_j m(\mathbf{v}_j^\top \mathbf{x})^{m-1}\mathbf{v}_j = m\sum_{j=1}^{r} \alpha_j (\mathbf{v}_j^\top \mathbf{x})^{m-2}\mathbf{v}_j\mathbf{v}_j^\top \mathbf{x} =: m\mathbf{D}\mathbf{x}, \tag{6.4}$$

where $\mathbf{D}$ is a function of $\mathbf{x}$, while for HOSVD,

$$\mathbf{D} = m\mathbf{U}^\top \langle \mathcal{S}, (\mathbf{U}\mathbf{x})^{m-1}\rangle \mathbf{x}^\top, \tag{6.5}$$

where $\mathcal{S}$ and $\mathbf{U}$ are the core tensor and the singular matrix for symmetric HOSVD respectively. Note that $\langle \mathcal{S}, (\mathbf{U}\mathbf{x})^{m-1} \rangle$ is a column vector.

Since later we propose to represent vectors $\mathbf{x}$ as quantum states, the quantum algorithm naturally produces normalized vectors with $\mathbf{x}^\top \mathbf{x} = 1$. The problem we solve is thus,

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = \mathcal{A}\mathbf{x}^m \\ \text{subject to} \quad & \mathbf{x}^\top \mathbf{x} = 1, \end{aligned} \tag{6.6}$$

or in quantum form

$$\underset{|\mathbf{x}\rangle}{\text{minimize}} \quad f(|\mathbf{x}\rangle). \tag{6.7}$$

This is a polynomial optimization problem constrained under a unit sphere.

For the quadratic case, i.e. $m = 2$, the objective function is simplified as

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top \mathbf{A}\mathbf{x}, \tag{6.8}$$

where $\mathbf{A}$ is symmetric. The gradient $\nabla f(\mathbf{x}) = \mathbf{A}\mathbf{x}$.

6.3 Quantum BB Gradient Algorithms

Gradient descent is a first-order iterative optimization algorithm for finding a minimum of a function. For classical gradient descent method, let $f : \mathbb{R}^N \to \mathbb{R}$ be the objective function. Given the initial point $\mathbf{x}_0$, one finds a minimum by searching along the negative of the gradient iteratively at the current point

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \tau \nabla f(\mathbf{x}_k), \tag{6.9}$$

where $\tau > 0$ is the step size or learning rate in machine learning. For the standard gradient descent method, the step sizes can be found via line search satisfying Wolfe conditions [83]. However, the standard gradient descent method does not perform

efficiently when the condition number is large. Thus, the BB gradient descent method is designed in [5], where the step size of every iteration depends on the gradient and point of current step and last step.

For the quantum BB method, the coefficient tensor, points, and gradients are stored in the quantum register, and the iteration function (6.9) becomes

$$|\mathbf{x}_{k+1}\rangle \propto \Big(|\mathbf{x}_k\rangle - \tau_k |\nabla f(\mathbf{x}_k)\rangle \Big) \tag{6.10}$$

with a normalization factor. Here, the points and gradients are expressed as quantum states. In this section, we show how this quantum iteration function is performed. First, we introduce the data input oracle in the following.

6.3.1 Data Input

The tensor $\mathcal{A}$ can be accessed by qRAM, or specifically, the tree data structure, which is introduced in Chapter 3. The initial guess $|\mathbf{x}_0\rangle$ can be an arbitrary quantum state, e.g., we can set $|\mathbf{x}_0\rangle$ as $|0\rangle$ in the computational basis. Since the choice of initial states greatly affects the convergence point and speed, we may choose a set of initial states and apply the BB gradient descent method simultaneously, then there is a high probability that the global minima is reached. For example, the set of initial states is defined as

$$\frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |(\mathbf{x}_0)_i\rangle |i\rangle := \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle |i\rangle, \tag{6.11}$$

we start the initial guesses from different modes $|0\rangle, |1\rangle, \ldots, |N-1\rangle$ stored in the first register, and the second register stores the index i which refers to the i-th initial state.

6.3.2 BB Step Sizes

Now assume we are at the kth step and have prepared multiple copies of the current solution $|\mathbf{x}_k\rangle$ to an accuracy $\epsilon_k > 0$. We would like to propagate a single copy of $|\mathbf{x}_k\rangle$ to an improved $|\mathbf{x}_{k+1}\rangle$. The improved copy will be prepared to accuracy $\epsilon_{k+1} = O(\epsilon_k + \tau_k \epsilon_{D_k})$, where ϵ_{D_k} is the accuracy of $\mathbf{D}$ in kth step.

Denote $\nabla f(\mathbf{x}_k) = \mathbf{D}\,\mathbf{x}_k$, $\mathbf{s}_{k-1} = \mathbf{x}_k - \mathbf{x}_{k-1}$ and $\mathbf{y}_{k-1} = \nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1})$. Hessian matrix $\mathbf{H}$ satisfies $\mathbf{H}\,\mathbf{s}_{k-1} = \mathbf{y}_{k-1}$. The goal of BB gradient method is to approximate $\mathbf{H}$ by using $\tau_k^{-1}\mathbf{I}$, so that $(\tau_k^{-1}\mathbf{I})\mathbf{s}_{k-1} \approx \mathbf{y}_{k-1}$. Then, the problem is transferred to the following two least-square problem

$$\tau_k = \arg\min_{\beta} \frac{1}{2}\|\beta \mathbf{s}_{k-1} - \mathbf{y}_{k-1}\|^2, \tag{6.12}$$

and

$$\tau_k = \arg\min_{\beta} \frac{1}{2}\|\mathbf{s}_{k-1} - \beta^{-1}\mathbf{y}_{k-1}\|^2. \tag{6.13}$$

Solving these two problems, we obtain two BB step sizes

$$\tau_k^{\mathrm{LBB}} = \frac{\|\mathbf{s}_{k-1}\|_2^2}{\left|\mathbf{s}_{k-1}^\top \mathbf{y}_{k-1}\right|}, \tag{6.14}$$

$$\tau_k^{\mathrm{SBB}} = \frac{\left|\mathbf{s}_{k-1}^\top \mathbf{y}_{k-1}\right|}{\|\mathbf{y}_{k-1}\|_2^2}. \tag{6.15}$$

Note that $\tau_k^{\mathrm{LBB}} \geqslant \tau_k^{\mathrm{SBB}}$ due to Cauchy-Schwarz Inequality, thus 'L' and 'S' refer to 'Large' and 'Small' respectively.

Combining the above two BB step sizes, the following alternative BB (ABB) step size and adaptive BB (ADBB) step size are designed.

$$\tau_k^{\mathrm{ABB}} = \begin{cases} \tau_k^{\mathrm{LBB}}, & k \text{ is even}, \\ \tau_k^{\mathrm{SBB}}, & k \text{ is odd}, \end{cases} \tag{6.16}$$

$$\tau_k^{\mathrm{ADBB}} = \begin{cases} \tau_k^{\mathrm{SBB}}, & \text{if } \frac{\tau_k^{\mathrm{SBB}}}{\tau_k^{\mathrm{LBB}}} < \kappa, \quad \kappa \in (0,1), \\ \tau_k^{\mathrm{LBB}}, & \text{otherwise}. \end{cases} \tag{6.17}$$

Writing the solution vectors as quantum states, we have

$$\tau_k^{\mathrm{LBB}} = \frac{2 - 2\langle \mathbf{x}_k | \mathbf{x}_{k-1} \rangle}{\left| \left(\langle \mathbf{x}_k | - \langle \mathbf{x}_{k-1} | \right) \left(\mathbf{D} | \mathbf{x}_k \rangle - \mathbf{D} | \mathbf{x}_{k-1} \rangle \right) \right|}, \tag{6.18}$$

and

$$\tau_k^{\mathrm{SBB}} = \frac{\left| \left(\langle \mathbf{x}_k | - \langle \mathbf{x}_{k-1} | \right) \left(\mathbf{D} | \mathbf{x}_k \rangle - \mathbf{D} | \mathbf{x}_{k-1} \rangle \right) \right|}{\| \mathbf{D} | \mathbf{x}_k \rangle - \mathbf{D} | \mathbf{x}_{k-1} \rangle \|^2}. \tag{6.19}$$

In particular, for the quadratic objective function (6.8),

$$\tau_k^{\mathrm{LBB}} = \frac{2 - 2\langle \mathbf{x}_k | \mathbf{x}_{k-1} \rangle}{| \langle \mathbf{x}_k | \mathbf{A} | \mathbf{x}_k \rangle - \langle \mathbf{x}_k | \mathbf{A} | \mathbf{x}_{k-1} \rangle - \langle \mathbf{x}_{k-1} | \mathbf{A} | \mathbf{x}_k \rangle + \langle \mathbf{x}_{k-1} | \mathbf{A} | \mathbf{x}_{k-1} \rangle |}, \tag{6.20}$$

$$\tau_k^{\mathrm{SBB}} = \frac{| \langle \mathbf{x}_k | \mathbf{A} | \mathbf{x}_k \rangle - \langle \mathbf{x}_k | \mathbf{A} | \mathbf{x}_{k-1} \rangle - \langle \mathbf{x}_{k-1} | \mathbf{A} | \mathbf{x}_k \rangle + \langle \mathbf{x}_{k-1} | \mathbf{A} | \mathbf{x}_{k-1} \rangle |}{\langle \mathbf{x}_k | \mathbf{A}^2 | \mathbf{x}_k \rangle - \langle \mathbf{x}_k | \mathbf{A}^2 | \mathbf{x}_{k-1} \rangle - \langle \mathbf{x}_{k-1} | \mathbf{A}^2 | \mathbf{x}_k \rangle + \langle \mathbf{x}_{k-1} | \mathbf{A}^2 | \mathbf{x}_{k-1} \rangle}. \tag{6.21}$$

In this case, BB step sizes only depend on the current and last states. Note that $\langle \mathbf{x} | \mathbf{A} | \mathbf{x} \rangle = \mathrm{tr}\{\mathbf{A}\rho\} = \mathrm{tr}\{\rho\mathbf{A}\}$, where $\rho = |\mathbf{x}\rangle\langle\mathbf{x}|$.

6.3.3 Quantum BB Gradient Method

Let $m = 2p$. The full operator $\mathbf{D} = \sum_{j=1}^{r} \alpha_k (\mathbf{v}_j^\top \mathbf{x})^{m-2} \mathbf{v}_j \mathbf{v}_j^\top$ in the form of CP decomposition (6.4) can be reproduced by a quantum operator that is equivalent to

$$\mathbf{D} = \mathrm{tr}_{1\ldots m-2}\{\rho^{\circ(p-1)} \mathcal{A}\}, \tag{6.22}$$

where ρ is the corresponding density matrix.

To implement the multiplication with operator $\mathbf{D} |\mathbf{x}\rangle = |\nabla f(\mathbf{x})\rangle$ used for the quantum gradient descent step, matrix exponentiation $e^{\mathrm{i}\mathbf{D}\Delta t} |\mathbf{x}\rangle$ adapting the quantum principal component analysis (qPCA) procedure [47] and subsequent phase estimation [51] are adopted. qPCA enables us to use multiple copies of a quantum system with density matrix ρ to construct the unitary transformation $e^{-\mathrm{i}\rho t}$. Since $\mathbf{D}$ depends on the current state, we cannot directly exponentiate it. Instead we

exponentiate the tensor $\mathcal{A}$ by using multiple copies of $\rho = |\mathbf{x}\rangle\langle\mathbf{x}|$ for a short time Δt.

Theorem 6.1. *By qPCA, the following operation is performed*

$$\mathrm{tr}_{1\ldots m-2}\{e^{-\mathrm{i}\mathcal{A}\Delta t}\,\rho^{\circ p}\,e^{\mathrm{i}\mathcal{A}\Delta t}\} = e^{-\mathrm{i}\mathbf{D}\Delta t}\,\rho\,e^{\mathrm{i}\mathbf{D}\Delta t} + O(\Delta t^2). \tag{6.23}$$

Proof.

$$\begin{aligned}
&\mathrm{tr}_{1\ldots m-2}\{e^{-\mathrm{i}\mathcal{A}\Delta t}\,\rho^{\circ p}\,e^{\mathrm{i}\mathcal{A}\Delta t}\}\\
=&\,\mathrm{tr}_{1\ldots m-2}\{(\mathbf{I}-\mathrm{i}\mathcal{A}\Delta t)\rho^{\circ p}(\mathbf{I}+\mathrm{i}\mathcal{A}\Delta t)\} + O(\Delta t^2)\\
=&\,\mathrm{tr}_{1\ldots m-2}\{\rho^{\circ p} - \mathrm{i}\mathcal{A}\rho^{\circ p}\Delta t + \mathrm{i}\rho^{\circ p}\mathcal{A}\Delta t\} + O(\Delta t^2)\\
=&\,\rho + \mathrm{i}\,\mathrm{tr}_{1\ldots m-2}\{-\mathcal{A}\rho^{\circ p} + \rho^{\circ p}\mathcal{A}\}\Delta t + O(\Delta t^2),
\end{aligned}$$

where

$$\mathrm{tr}_{1\ldots m-2}\{\mathcal{A}\rho^{\circ p}\} = \sum_{j=1}^{r}\alpha_j(\mathbf{v}_j^\top\mathbf{x})^{m-1}\mathbf{v}_j\mathbf{x}^\top, \tag{6.24}$$

and

$$\mathrm{tr}_{1\ldots m-2}\{\rho^{\circ p}\mathcal{A}\} = \sum_{j=1}^{K}\alpha_j(\mathbf{x}^\top\mathbf{v}_j)^{m-1}\mathbf{x}\mathbf{v}_j^\top. \tag{6.25}$$

Thus,

$$\mathrm{tr}_{1\ldots m-2}\{e^{-\mathrm{i}\mathcal{A}\Delta t}\,\rho^{\circ p}\,e^{\mathrm{i}\mathcal{A}\Delta t}\} = \rho + \mathrm{i}\sum_{j=1}^{r}\alpha_j\left(-(\mathbf{v}_j^\top\mathbf{x})^{m-1}\mathbf{v}_j\mathbf{x}^\top + (\mathbf{x}^\top\mathbf{v}_j)^{m-1}\mathbf{x}\mathbf{v}_j^\top\right)\Delta t + O(\Delta t^2). \tag{6.26}$$

Also,

$$\begin{aligned}
&e^{-\mathrm{i}\mathbf{D}\Delta t}\,\rho\,e^{\mathrm{i}\mathbf{D}\Delta t}\\
=&\,(\mathbf{I}-\mathrm{i}\mathbf{D}\Delta t)\rho(\mathbf{I}+\mathrm{i}\mathbf{D}\Delta t) + O(\Delta t^2)\\
=&\,\rho - \mathrm{i}\mathbf{D}\rho\Delta t + \mathrm{i}\rho\mathbf{D}\Delta t + O(\Delta t^2)\\
=&\,\rho + \mathrm{i}\sum_{j=1}^{r}\alpha_j\left(-(\mathbf{v}_j^\top\mathbf{x})^{m-1}\mathbf{v}_j\mathbf{x}^\top + (\mathbf{x}^\top\mathbf{v}_j)^{m-1}\mathbf{x}\mathbf{v}_j^\top\right)\Delta t + O(\Delta t^2).
\end{aligned} \tag{6.27}$$

□

Next, we implement the multiplication $\mathbf{D}\,|\mathbf{x}\rangle = |\nabla f(\mathbf{x})\rangle$ via phase estimation. The phase estimation algorithm is performed to estimate the unknown value θ to a finite level of precision, where $\mathbf{U}\,|\varphi\rangle = e^{2\pi \mathrm{i}\theta}\,|\varphi\rangle$, $0 \leqslant \theta < 1$, which means $e^{2\pi \mathrm{i}\theta}$ and $|\varphi\rangle$ are an eigenpair of a unitary operator $\mathbf{U}$. The quantum phase estimation procedure uses two registers. The first register contains d qubits initially in the state $|0\rangle$, where d decides the number of digits we wish to estimate for θ. The second register begins in the state $|\varphi\rangle$, and contains as many qubits as is necessary to store $|\varphi\rangle$.

In phase estimation, a multi-qubit register with $d = O(\lceil \log(1/\epsilon_2) \rceil)$ control qubits is used for forming an eigenvalue register. In this manner, for $\ell = 0, 1, \ldots, 2^d - 1$, we can prepare $(e^{-\mathrm{i}\mathbf{D}\Delta t})^\ell\,\rho\,(e^{\mathrm{i}\mathbf{D}\Delta t})^\ell$ by (6.23). Finally, we implement the inverse quantum Fourier transform, the result of the phase estimation algorithm is a quantum state proportional to

$$\sum_j \beta_j\,|\lambda_j(\mathbf{D})\rangle\,|u_j(\mathbf{D})\rangle\,, \tag{6.28}$$

where $|\mathbf{x}\rangle = \sum_j \beta_j\,|u_j(\mathbf{D})\rangle$ is the original state $|\mathbf{x}\rangle$ written in the eigenbasis $\{|u_j(\mathbf{D})\rangle\}$ of $\mathbf{D}$, and $|\lambda_j(\mathbf{D})\rangle$ is an additional register representing the corresponding eigenvalue in basis states $|0\rangle$ or $|1\rangle$.

Assume the gradient method is at the kth step, then we start from an initial state $|\psi_0\rangle$ which is prepared easily, after phase estimation, conditional rotation, several measurements and etc., the iteration function (6.29) is derived. The details are given in the following theorem and proof.

Theorem 6.2. *Given access to tensor $\mathcal{A}$ and the state of the kth step $|\mathbf{x}_k\rangle$ to accuracy $0 < \epsilon_k < 1$, we can obtain the next state $|\mathbf{x}_{k+1}\rangle$ by the iteration function*

$$|\mathbf{x}_{k+1}\rangle = \frac{1}{C_{k+1}}\Big(\,|\mathbf{x}_k\rangle - \tau_k\,|\nabla f(\mathbf{x}_k)\rangle\,\Big) \tag{6.29}$$

to accuracy $\epsilon_{k+1} = O(\tau_k \epsilon_{D_k} + \epsilon_k)$*, where* C_{k+1} *is the normalization factor with value* $C_{k+1}^2 = 1 - 2\tau_k \langle \mathbf{x}_k | \mathbf{D} | \mathbf{x}_k \rangle + \tau_k^2 \langle \mathbf{x}_k | \mathbf{D}^2 | \mathbf{x}_k \rangle$.

Proof. We prepare the state

$$|\psi_0\rangle = (\cos\theta |0\rangle - \mathrm{i} \sin\theta |1\rangle) |\mathbf{x}_k\rangle, \tag{6.30}$$

where θ is an external parameter. The eigenstates of $\mathbf{D}$ are given by $|u_j(\mathbf{D})\rangle$ and the eigenvalues by $\lambda_j(\mathbf{D})$. After the conditional phase estimation which is specified in Chapter 5 on $|1\rangle$ in the first register, we obtain

$$|\psi_1\rangle = \cos\theta |0\rangle |\mathbf{x}_k\rangle |0\rangle - \mathrm{i} \sin\theta |1\rangle \sum_j \beta_j |u_j(\mathbf{D})\rangle |\lambda_j(\mathbf{D})\rangle, \tag{6.31}$$

where $\beta_j = \langle u_j(\mathbf{D}) | \mathbf{x}_k \rangle$. Now perform a rotation on $|\lambda_j(\mathbf{D})\rangle$, uncompute the eigenvalue, and apply a σ_X operator on the third register to arrive at the state

$$|\psi_2\rangle = \cos\theta |0\rangle |\mathbf{x}_k\rangle |1\rangle - \mathrm{i} \sin\theta |1\rangle \sum_j \beta_j |u_j(\mathbf{D})\rangle \Big(\sqrt{1 - (C_D \lambda_j(\mathbf{D}))^2} |0\rangle + C_D \lambda_j(\mathbf{D}) |1\rangle \Big). \tag{6.32}$$

We choose a constant $C_D = O(1/\kappa_D)$, where κ_D is the condition number of $\mathbf{D}$. A measurement of the ancilla in $|1\rangle$ arrives at

$$|\psi_3\rangle = \frac{1}{\sqrt{P_D}} \Big(\cos\theta |0\rangle |\mathbf{x}_k\rangle - \mathrm{i} \sin\theta |1\rangle \sum_j C_D \lambda_j(\mathbf{D}) \beta_j |u_j(\mathbf{D})\rangle \Big), \tag{6.33}$$

which is the desired state

$$|\psi_4\rangle = \frac{1}{\sqrt{P_D}} \Big(\cos\theta |0\rangle |\mathbf{x}_k\rangle - \mathrm{i} C_D \sin\theta |1\rangle \mathbf{D} |\mathbf{x}_k\rangle \Big). \tag{6.34}$$

Therefore, we can multiply the operator $\mathbf{D}$ to $|\mathbf{x}_k\rangle$ conditioned on the ancilla being in state $|1\rangle$ to obtain $|\psi_4\rangle$. The success probability is given by

$$P_D = \cos^2\theta + C_D^2 \sin^2\theta \langle \mathbf{x}_k | \mathbf{D}^2 | \mathbf{x}_k \rangle. \tag{6.35}$$

We measure the state (6.34) in the basis $|\text{yes}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \text{i}\,|1\rangle)$ and $|\text{no}\rangle = \frac{1}{\sqrt{2}}(\text{i}\,|0\rangle + |1\rangle)$. Measuring the 'yes' basis state results in the quantum system being in a state

$$|\mathbf{x}_{k+1}\rangle = \frac{1}{\sqrt{2P_D P_{\text{yes}}^{\text{grad}}}}\Big(\cos\theta\,|\mathbf{x}_k\rangle - C_D \sin\theta\,\mathbf{D}\,|\mathbf{x}_k\rangle\Big). \tag{6.36}$$

Since the BB steps are calculated by (6.18) or (6.19), we choose θ such that

$$\cos\theta = \frac{1}{\sqrt{1+(\tau_k)^2/C_D^2}},\ \ \sin\theta = \frac{\tau_k}{C_D\sqrt{1+(\tau_k)^2/C_D^2}}, \tag{6.37}$$

which leads to the following iteration function (6.29) with $C_{k+1}^2 = 1 - 2\tau_k \langle \mathbf{x}_k|\,\mathbf{D}\,|\mathbf{x}_k\rangle + \tau_k^2 \langle \mathbf{x}_k|\,\mathbf{D}^2\,|\mathbf{x}_k\rangle$. The total probability of obtaining this state successfully is given by

$$P_D P_{\text{yes}}^{\text{grad}} = \frac{1}{2}(\cos^2\theta + C_D^2 \sin^2\theta\,\langle \mathbf{x}_k|\,\mathbf{D}^2\,|\mathbf{x}_k\rangle). \tag{6.38}$$

To successfully obtain the state $|\mathbf{x}_{k+1}\rangle$, we need to repeat $O(\sqrt{1/P_D P_{\text{yes}}^{\text{grad}}})$ times by using amplitude amplification [1]. □

Now that we have the iteration function (6.29), then we repeat the iterations starting from $k = 0, 1, ...$ until the stopping criteria is satisfied. Finally, we obtain the desired state, denoted as $|\mathbf{x}_T\rangle$.

If we start from different initial states simultaneously as (6.11), we apply the iteration function on (6.11) in parallel, then the final state is $\frac{1}{\sqrt{N}}\sum_{i=0}^{N-1} |(\mathbf{x}_k)^i\rangle\,|i\rangle$. Post-selecting i in the second register, we get the i-th final state. Furthermore, it is highly possible that there exists a global minimum in these final states.

In conclusion, we put forward the quantum Barzilai-Borwein gradient algorithm:

Algorithm 4 Quantum Barzilai-Borwein (QBB) Gradient Method

Input: $\mathcal{A}, |\mathbf{x}_0\rangle, \tau_0, 0 < \epsilon \ll 1, k = 1$
Output: $|\mathbf{x}_T\rangle$

1. $|\mathbf{x}_1\rangle$ is calculated by standard gradient descent method with τ_0.
2. If $\||\mathbf{x}_k\rangle - |\mathbf{x}_{k-1}\rangle\| \leqslant \epsilon$, stop. Set $T = k$.
3. For $k \geqslant 1$, τ_k is computed by formulae (6.18) or (6.19).
4. Calculate $|\mathbf{x}_{k+1}\rangle = \frac{1}{C_{k+1}} \left(|\mathbf{x}_k\rangle - \tau_k |\nabla f(\mathbf{x}_k)\rangle\right)$ by Theorem 6.2.
5. $k = k + 1$, return to Step 2.

To reach the optimal state, we need to perform the iteration several times. For the kth step, the state $|\mathbf{x}_k\rangle$ proceeds to $|\mathbf{x}_{k+1}\rangle$ probabilistically. Due to the no-cloning property, if the iteration fails at a certain stage, the algorithm has to restart at the initial state. Thus, the whole algorithm scales exponentially in the number of iterations.

6.3.4 Quantum Feasible BB-like Method

In [37], a feasible BB-like method for solving problem (6.6) under the quadratic case, i.e., (6.8) is proposed, where $\mathbf{A} \in \mathbb{R}^{N\times N}$ is a symmetric matrix. The feasible set $\Omega = \left\{\mathbf{x} : \mathbf{x}^\top\mathbf{x} = 1, \mathbf{x} \in \mathbb{R}^N\right\}$ is a special case of Stiefel manifold. In quantum computing, $|\mathbf{x}\rangle$ is an n-qubit quantum state, where $N = 2^n$. Thus, the constraint of problem (6.6) is naturally satisfied.

Let the Lagrange function

$$L(\mathbf{x}, \lambda) = \frac{1}{2}\mathbf{x}^\dagger\mathbf{A}\mathbf{x} - \lambda\mathbf{x}^\dagger\mathbf{x}. \tag{6.39}$$

Suppose $\mathbf{x}$ is a local minimizer of problem (6.8). Then $\mathbf{x}$ satisfies the first-order optimality condition $\partial_{\mathbf{x}} L(\mathbf{x}, \lambda) = \mathbf{A}\mathbf{x} - \lambda\mathbf{x}$. Set the lagrange multiplier $\lambda = \mathbf{x}_k^\dagger\mathbf{A}\mathbf{x}_k$, and denote $\mathbf{g}_k^L(\mathbf{x}) = \mathbf{A}\mathbf{x}_k - (\mathbf{x}_k^\dagger\mathbf{A}\mathbf{x}_k)\mathbf{x}_k$.

Given $|\mathbf{x}_k\rangle$,

$$\begin{aligned}\mathbf{g}_k^L = \|\mathbf{g}_k^L\| \, |\mathbf{g}_k^L\rangle &= \mathbf{A} \, |\mathbf{x}_k\rangle - \langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_k\rangle \, |\mathbf{x}_k\rangle \\ &= (\mathbf{A} - \langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_k\rangle \mathbf{I}) \, |\mathbf{x}_k\rangle \\ &= (\mathbf{A} - \mathrm{tr}(\mathbf{A}\rho_k)\mathbf{I}) \, |\mathbf{x}_k\rangle \\ &=: \mathbf{D} \, |\mathbf{x}_k\rangle . \end{aligned} \tag{6.40}$$

Note that $\mathbf{D}$ is not unitary and depends on the current state $|\mathbf{x}_k\rangle$.

Let $Y(\tau, \mathbf{x})$ be the general curve defined in [37] and given as follows

$$Y(\tau, |\mathbf{x}\rangle) = \alpha(\tau, |\mathbf{x}\rangle) \, |\mathbf{x}\rangle - \beta(\tau, |\mathbf{x}\rangle)\mathbf{g}^L, \tag{6.41}$$

where

$$\alpha(\tau, |\mathbf{x}\rangle) = \frac{1 - a^2\tau^2 \left\|\mathbf{g}^L\right\|^2}{1 + a^2\tau^2 \left\|\mathbf{g}^L\right\|^2}, \quad \beta(\tau, |\mathbf{x}\rangle) = \frac{2a\tau}{1 + a^2\tau^2 \left\|\mathbf{g}^L\right\|^2}, \tag{6.42}$$

and a is a predetermined constant in $(0, 1)$.

Then, the iteration function (6.41) becomes

$$\begin{aligned} Y(\tau_k, \mathbf{x}_k) &= \frac{(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2) \, |\mathbf{x}_k\rangle - 2a\tau_k \|\mathbf{g}_k^L\| \, |\mathbf{g}_k^L\rangle}{1 + a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2} \\ &= \frac{(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2) \, |\mathbf{x}_k\rangle - 2a\tau_k \mathbf{D} \, |\mathbf{x}_k\rangle}{1 + a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2}, \end{aligned} \tag{6.43}$$

where

$$\tau_k^{\mathrm{LBB}} = \frac{2(1 - \langle \mathbf{x}_{k-1}|\mathbf{x}_k\rangle)}{|\langle \mathbf{x}_{k-1}|\mathbf{x}_k\rangle\langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_k\rangle - \langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_{k-1}\rangle - \langle \mathbf{x}_{k-1}|\mathbf{A}|\mathbf{x}_k\rangle + \langle \mathbf{x}_k|\mathbf{x}_{k-1}\rangle\langle \mathbf{x}_{k-1}|\mathbf{A}|\mathbf{x}_{k-1}\rangle|}, \tag{6.44}$$

or

$$\tau_k^{\mathrm{SBB}} = \frac{|\langle \mathbf{x}_{k-1}|\mathbf{x}_k\rangle\langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_k\rangle - \langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_{k-1}\rangle - \langle \mathbf{x}_{k-1}|\mathbf{A}|\mathbf{x}_k\rangle + \langle \mathbf{x}_k|\mathbf{x}_{k-1}\rangle\langle \mathbf{x}_{k-1}|\mathbf{A}|\mathbf{x}_{k-1}\rangle|}{\|(\mathbf{A} - \langle \mathbf{x}_k|\mathbf{A}|\mathbf{x}_k\rangle) \, |\mathbf{x}_k\rangle - (\mathbf{A} - \langle \mathbf{x}_{k-1}|\mathbf{A}|\mathbf{x}_{k-1}\rangle) \, |\mathbf{x}_{k-1}\rangle\|^2}, \tag{6.45}$$

which can be derived by equations (6.18)-(6.19) and (6.43).

By (6.31), after conditional phase estimation, we obtain

$$|\psi\rangle = \cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle\,|0\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j \beta_j\,|u_j(\mathbf{A})\rangle\,|\lambda_j(\mathbf{A})\rangle\,, \tag{6.46}$$

the next step is to extract the $\lambda_j(\mathbf{A})$. We know that $\mathbf{A} = \mathbf{U}\Lambda\mathbf{U}^\dagger$, then

$$\mathbf{D} = \mathbf{A} - \mathrm{tr}(\mathbf{A}\rho_k)\mathbf{I} = \mathbf{U}\left(\Lambda - \mathrm{tr}(\mathbf{A}\rho_k)\mathbf{I}\right)\mathbf{U}^\dagger, \tag{6.47}$$

i.e.

$$\lambda(\mathbf{D}) = \lambda(\mathbf{A}) - \mathrm{tr}(\mathbf{A}\rho_k). \tag{6.48}$$

After calculating $\mathrm{tr}(\mathbf{A}\rho_k)$, then by conditional rotation, we have

$$\begin{aligned}|\psi\rangle &= \cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j \beta_j(\lambda(\mathbf{A}) - \mathrm{tr}(\mathbf{A}\rho_k))\,|u_j(\mathbf{A})\rangle \\ &= \cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j \beta_j\lambda(\mathbf{D})\,|u_j(\mathbf{D})\rangle\,.\end{aligned} \tag{6.49}$$

Then, we show how we obtain the iteration function of quantum feasible BB-like method. We prepare the state

$$|\psi_0\rangle = (\cos\theta\,|0\rangle - \mathrm{i}\sin\theta\,|1\rangle)\,|\mathbf{x}_k\rangle\,, \tag{6.50}$$

where θ is an external parameter. It is inferred above that the eigenpairs of $\mathbf{D}$ have such relations with those of $\mathbf{A}$: $\lambda(\mathbf{D}) = \lambda(\mathbf{A}) - \mathrm{tr}(\mathbf{A}\rho_k)$ and $|u_j(\mathbf{D})\rangle = |u_j(\mathbf{A})\rangle$. After the conditional phase estimation by QSVE, we obtain:

$$|\psi_1\rangle = \cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle\,|0\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j \beta_j\,|u_j(\mathbf{A})\rangle\,|\lambda_j(\mathbf{A})\rangle\,, \tag{6.51}$$

where $\beta_j = \langle u_j(\mathbf{A})|\mathbf{x}_k\rangle$. Now we first perform a conditional rotation of another ancilla, then apply another rotation according to the value of $\mathrm{tr}(\mathbf{A}\rho_k)$, finally un-

compute the eigenvalue register to arrive at the state

$$|\psi_2\rangle = \cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle\,|1\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j \beta_j\,|u_j(\mathbf{A})\rangle$$

$$\left(\sqrt{1-(C_A(\lambda_j(\mathbf{A})-\mathrm{tr}(\mathbf{A}\rho_k))^2}\,|0\rangle + C_A(\lambda_j(\mathbf{A})-\mathrm{tr}(\mathbf{A}\rho_k))\,|1\rangle\right)$$

$$= \cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle\,|1\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j \beta_j\,|u_j(\mathbf{D})\rangle\left(\sqrt{1-(C_A\lambda_j(\mathbf{D}))^2}\,|0\rangle + C_A\lambda_j(\mathbf{D})\,|1\rangle\right). \tag{6.52}$$

We choose a constant $C_A = O(1/\kappa_A)$, where κ_A is the condition number of $\mathbf{A}$. A measurement of the ancilla in $|1\rangle$ arrives at

$$|\psi_3\rangle = \frac{1}{\sqrt{P_D}}\Big(\cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle - \mathrm{i}\sin\theta\,|1\rangle\sum_j C_A\beta_j\lambda_j(\mathbf{D})\,|u_j(\mathbf{D})\rangle\Big), \tag{6.53}$$

which is the desired state

$$|\psi_4\rangle = \frac{1}{\sqrt{P_D}}\Big(\cos\theta\,|0\rangle\,|\mathbf{x}_k\rangle - \mathrm{i}C_A\sin\theta\,|1\rangle\,\mathbf{D}\,|\mathbf{x}_k\rangle\Big). \tag{6.54}$$

Therefore, we can multiply the operator $\mathbf{D}$ to $|\mathbf{x}_k\rangle$ conditioned on the ancilla being in state $|1\rangle$ to obtain $|\psi_4\rangle$. The success probability is given by

$$P_D = \cos^2\theta + C_A^2\sin^2\theta\,\langle\mathbf{x}_k|\,\mathbf{D}^2\,|\mathbf{x}_k\rangle = \cos^2\theta + C_A^2\sin^2\theta\|\mathbf{g}_k^L\|^2. \tag{6.55}$$

We measure the state (6.54) in the basis $|\mathrm{yes}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \mathrm{i}\,|1\rangle)$ and $|\mathrm{no}\rangle = \frac{1}{\sqrt{2}}(\mathrm{i}\,|0\rangle + |1\rangle)$. Measuring the 'yes' basis state results in the quantum system being in a state

$$|\mathbf{x}_{k+1}\rangle = \frac{1}{\sqrt{2P_D P_{\mathrm{yes}}^{\mathrm{grad}}}}\Big(\cos\theta\,|\mathbf{x}_k\rangle - C_A\sin\theta\,\mathbf{D}\,|\mathbf{x}_k\rangle\Big). \tag{6.56}$$

Since the BB steps are calculated by (6.44) or (6.45), we choose θ such that

$$\cos\theta = \frac{1-a^2\tau_k^2\left\|\mathbf{g}_k^L\right\|^2}{\sqrt{\left(1-a^2\tau_k^2\left\|\mathbf{g}_k^L\right\|^2\right)^2 + (2a\tau_k)^2/C_A^2}},\ \sin\theta = \frac{2a\tau_k/C_A}{\sqrt{\left(1-a^2\tau_k^2\left\|\mathbf{g}_k^L\right\|^2\right)^2 + (2a\tau_k)^2/C_A^2}}, \tag{6.57}$$

which leads to the following iteration function

$$|\mathbf{x}_{k+1}\rangle = \frac{1}{C_{\text{grad}}^{(k+1)}}\Big((1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2)\,|\mathbf{x}_k\rangle - 2a\tau_k \mathbf{D}\,|\mathbf{x}_k\rangle\Big). \tag{6.58}$$

with $C_{\text{grad}}^{(k+1)} = 1 + a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2$. The probability of obtaining this state through a successful 'yes' measurement is given by

$$P_{\text{yes}}^{\text{grad}} = \frac{(1 + a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2)^2}{2(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2)^2 + 8a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2} = \frac{1}{2}. \tag{6.59}$$

The total probability of measuring $|1\rangle$ and $|\text{yes}\rangle$ is

$$P_D P_{\text{yes}}^{\text{grad}} = \frac{1}{2} \cdot \frac{(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2)^2 + (2a\tau_k \left\|\mathbf{g}_k^L\right\|)^2}{(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2)^2 + (2a\tau_k/C_A)^2}. \tag{6.60}$$

Assume $\|\mathbf{A}\|_F = 1$, since $\mathbf{A}$ is a symmetric matrix,

$$\sum_{i=1}^{N} \lambda_i^2(\mathbf{A}) = \sum_{i=1}^{N} \sigma_i^2(\mathbf{A}) = \|\mathbf{A}\|_F^2 = 1. \tag{6.61}$$

Therefore, $|\lambda_{\max}(\mathbf{A})| \leqslant 1$, we can take C_A as 1. The probability becomes

$$P_D P_{\text{yes}}^{\text{grad}} = \frac{1}{2} \cdot \frac{\left(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2\right)^2 + \left(2a\tau_k \left\|\mathbf{g}_k^L\right\|\right)^2}{\left(1 - a^2\tau_k^2 \left\|\mathbf{g}_k^L\right\|^2\right)^2 + (2a\tau_k)^2}. \tag{6.62}$$

Thus, we propose the following quantum feasible BB-like method

Algorithm 5 Quantum Feasible BB-like (QFBB) Method

Input: $\mathbf{A}, \mathbf{x}_0, k = 0, \tau_0 > 0, 0 < \epsilon \ll 1, 0 < \sigma < 1, 0 < \delta < 1, 0 < a < 1$

Output: $|\mathbf{x}_T\rangle$

1. $|\mathbf{x}_1\rangle$ is calculated by standard gradient descent method with τ_0.
2. If $\||\mathbf{x}_k\rangle - |\mathbf{x}_{k-1}\rangle\| \leqslant \epsilon$, stop.
3. For $k \geqslant 1$, τ_k is computed by formulae (6.44) or (6.45).
4. Calculate $|\mathbf{x}_{k+1}\rangle = Y(\tau_k, |\mathbf{x}_k\rangle)$ by formulae (6.41) and (6.42).
5. $k = k + 1$. Go to Step 1.

6.4 Analysis

6.4.1 Two Classes of Convergence Rates

In this section, we introduce some concepts for our convergence analysis in the next section. The definitions below can be found in [52].

Definition 6.1. Quotient-Linear (Q-linear)

Let $\{x_k\}$ be a sequence in $\mathbb{R}$ that converges to x^*. Then the convergence is called Q-linear if there exists $r \in (0,1)$ such that

$$\frac{|x_{k+1} - x^*|}{|x_k - x^*|} \leqslant r \tag{6.63}$$

for all k sufficiently large.

Definition 6.2. Quotient-Superlinear (Q-superlinear)

Let $\{x_k\}$ be a sequence in $\mathbb{R}$ that converges to x^*. Then the convergence is called Q-superlinear if

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|} = 0. \tag{6.64}$$

Definition 6.3. Root-Linear (R-linear)

Let $\{x_k\}$ be a sequence in $\mathbb{R}$ that converges to x^*. Then the convergence is called R-linear if there exists a sequence $\{v_k\}$ with $v_k \geqslant 0$ for all k, $\{v_k\}$ converges Q-linearly to 0, and

$$|x_k - x^*| \leqslant v_k \tag{6.65}$$

for all k.

Definition 6.4. Root-Superlinear (R-superlinear)

Convergence is R-superlinear if $\{v_k\}$ converges Q-superlinearly to 0.

Note that root convergence is concerned only with the overall rate of decrease of the error while quotient convergence requires the error to decrease at each iteration of the algorithm. Thus, Q-convergence is a stronger form of convergence than R-convergence, and R-convergence implies Q-convergence.

6.4.2 Convergence Analysis

For any-dimensional strictly convex quadratic function, it is proved that either $\mathbf{g}_k = 0$ for some finite k, or the sequence $\{\|\mathbf{g}_k\|\}$ converges to zero R-linearly [22].

In particular, if $f(\mathbf{x})$ is a strictly quadratic convex function with 2 variables, i.e., $m = 2$, $n = 2$, the gradient method with BB step size (6.18) almost always converges R-superlinearly that

$$\|\mathbf{g}_k\| \leqslant C\lambda^{-(\sqrt{2})^k} \tag{6.66}$$

holds asymptotically, where $\lambda = \sigma_1(\mathbf{H})/\sigma_2(\mathbf{H})$, $\mathbf{H}$ is a symmetric positive definite matrix, C is a constant independent of k.

Furthermore, it is proved in [21] that the BB gradients satisfy

$$\lim_{k\to\infty} \min\left\{ \frac{\|\mathbf{g}_{k+1}\|}{\|\mathbf{g}_k\|}, \frac{\|\mathbf{g}_{k+2}\|}{\|\mathbf{g}_{k+1}\|}, \frac{\|\mathbf{g}_{k+3}\|}{\|\mathbf{g}_{k+2}\|} \right\} = 0, \tag{6.67}$$

which means that the BB method has a Q-superlinear convergence step in at most three consecutive steps.

It is proved by Raydan [64] that the BB method is globally convergent for any n if the objective function is a convex quadratic. However, for $m > 2$, no superlinear convergence results have been established for the BB method, though numerical results indicate quite offen that the BB method converges superlinearly.

For the problem with normalization constraint, the local superlinear convergence is ensured for feasible BB-like methods for the two-dimensional case. There is also a counter-example showing that the algorithms may cycle or stop at a non-stationary

point. To ensure the global convergence, an adaptive non-monotone line search with an improved line search is adopted in [37].

6.4.3 Computational Complexity

For the quantum BB algorithm, the computational complexity mainly comes from data input and phase estimation which are both polylogarithmic in N. Thus, the overall computational complexity is $O(\text{polylog}N)$, as long as the number of iterations T is small. Actually, the algorithm runs exponentially in T, because in every iteration, there is a probability for the algorithm to proceed to the next step. Thus, if the BB algorithm fails in any step, we have to reperform the iterations from the initial state.

6.5 Numerical Experiments

In this part, we compare four quantum optimization methods under the quadratic case: (i) quantum gradient descent (QGD) method [66], (ii) quantum Newton's (QNT) method [66], (iii) quantum BB gradient descent (QBB) method Algorithm 4, (iv) quantum feasible BB-like gradient (QFBB) method Algorithm 5.

For a random matrix $\mathbf{A} \in \mathbb{R}^{100\times 100}$, all the methods succeed to find the optimal solution. QGD and QNT need to run around 1000 times to get the optimal solution, and the success probability in every step is around 0.25-0.3. QBB runs about 200 times, and the success probability is roughly ascending, and stable at around 0.2. QFBB uses only 55 times, but the success probability is quite small, around 0.05-0.2, and even some could be very small. As a whole, the total success probability (calculated by the product of the success probability in every step) of QFBB is higher than the other three methods.

For a matrix $\mathbf{A} \in \mathbb{R}^{5\times 5}$ with a large condition number 200, QGD and QWT run 2000 times, while QBB and QFBB consume only 30 times. The total success

probability of QBB and QFBB is higher than QGD and QNT.

6.6 Summary

We have introduced the quantum version of the Barzilai-Borwein gradient method. Contrast to the standard gradient method whose step sizes are predetermined, the step sizes of BB methods depend on the current and last points and gradients. Our quantum methods consider an iterative constrained polynomial optimization in the quantum computing framework, where the solution vectors in iterations are quantum states. The first quantum BB method directly replaces the predetermined step sizes with BB step sizes, and projects the solution state onto the unit sphere in every iteration. We use qPCA to implement the gradient matrix. The second method is a quantum feasible BB-like method for the quadratic case, which searches the optimal solution along the unit sphere.

For the computational complexity, it is known that the running time of classical BB gradient method is $O(N)$, where N is the number of variables of the objective function, while for the quantum version, if the quantum BB gradient method is able to find good solutions in a few iterations, it runs in $O(\text{polylog}N)$. Thus, the quantum BB gradient method provides exponential speedups over the classical counterpart. By the numerical experiments, contrast to the quantum gradient descent method in [66], our quantum BB gradient methods find the optimal value in fewer iterations, and the total success probability is higher.

Chapter 7

Conclusions and Suggestions for Future Research

This thesis introduces some quantum operations on tensors, and provides a subclass of Hankel tensors, two quantum algorithms for higher order singular value decomposition, and quantum Barzilai-Borwein gradient methods. The quantum algorithms related to tensors may be used for research on the multipartite quantum system. Also, they can be applied as subroutines in quantum machine learning methods. Once quantum computers of a certain scale are constructed, our quantum algorithms will come into play.

For the future work, I hope to continue working on algorithms related to quantum and tensors. Tensor network is an effective way to run the quantum circuits as in [57, 3]. Maybe quantum algorithms combined with neural networks will be a new research direction. Also, recently quantum-inspired classical algorithms become very popular, since such algorithms run in the same level of complexity related to the dimensions compared to the pure quantum counterparts [76, 28]. Whether quantum-inspired algorithms could perform as fast as pure quantum algorithms remains to be explored.

Bibliography

[1] A. Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *STACS'12 (29th Symposium on Theoretical Aspects of Computer Science)*, volume 14, pages 636–647. LIPIcs, 2012.

[2] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko. Quantum Boltzmann machine. *Physical Review X*, 8(2):021050, 2018.

[3] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[4] R. Badeau and R. Boyer. Fast multilinear singular value decomposition for structured tensors. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1008–1021, 2008.

[5] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA journal of numerical analysis*, 8(1):141–148, 1988.

[6] F. Bell, D. S. Lambrecht, and M. Head-Gordon. Higher order singular value decomposition in quantum chemistry. *Molecular Physics*, 108(19-20):2759–2773, 2010.

[7] J. Bennett and S. Lanning. The Netflix Prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA., 2007.

[8] C. Berg. The multidimensional moment problem and semigroups. In *Proc. Symp. Appl. Math*, volume 37, pages 110–124, 1987.

[9] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders. Efficient quantum algorithms for simulating sparse Hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371, 2007.

[10] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

[11] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[12] R. Boyer, L. De Lathauwer, and K. Abed-Meraim. Higher order tensor-based method for delayed exponential fitting. *IEEE Transactions on Signal Processing*, 55(6):2795–2809, 2007.

[13] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.

[14] J. D. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[15] Y. Chen, L. Qi, and Q. Wang. Computing extreme eigenvalues of large scale Hankel tensors. *Journal of scientific computing*, 68(2):716–738, 2016.

[16] Y. Chen, L. Qi, and Q. Wang. Positive semi-definiteness and sum-of-squares property of fourth order four dimensional Hankel tensors. *Journal of Computational and Applied Mathematics*, 302:356–368, 2016.

[17] G. Chesi. On the gap between positive polynomials and SOS of polynomials. *IEEE Transactions on Automatic Control*, 52(6):1066–1072, 2007.

[18] A. Cichocki. Era of big data processing: A new approach via tensor networks and tensor decompositions. *arXiv preprint arXiv:1403.2048*, 2014.

[19] N. Cohen, O. Sharir, and A. Shashua. On the expressive power of deep learning: A tensor analysis. In *Conference on Learning Theory*, pages 698–728, 2016.

[20] D. Cores, R. Escalante, M. González-Lima, and O. Jimenez. On the use of the spectral projected gradient method for support vector machines. *Computational & Applied Mathematics*, 28(3):327–364, 2009.

[21] Y.-H. Dai. A new analysis on the Barzilai-Borwein gradient method. *Journal of the operations Research Society of China*, 1(2):187–198, 2013.

[22] Y.-H. Dai and L.-Z. Liao. R-linear convergence of the Barzilai and Borwein gradient method. *IMA Journal of Numerical Analysis*, 22(1):1–10, 2002.

[23] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[24] W. Ding, L. Qi, and Y. Wei. Fast Hankel tensor–vector product and its application to exponential data fitting. *Numerical Linear Algebra with Applications*, 22(5):814–832, 2015.

[25] W. Ding, L. Qi, and Y. Wei. Inheritance properties and sum-of-squares decomposition of Hankel tensors: theory and algorithms. *BIT Numerical Mathematics*, 57(1):169–190, 2017.

[26] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7), 1982.

[27] M. A. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of selected topics in signal processing*, 1(4):586–597, 2007.

[28] A. Gilyén, S. Lloyd, and E. Tang. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension. *arXiv preprint arXiv:1811.04909*, 2018.

[29] V. Giovannetti, S. Lloyd, and L. Maccone. Quantum random access memory. *Physical review letters*, 100(16):160501, 2008.

[30] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, pages 212–219. New York, NY, USA., 1996.

[31] W. Hackbusch and S. Kühn. A new scheme for the tensor representation. *Journal of Fourier analysis and applications*, 15(5):706–722, 2009.

[32] R. A. Harshman et al. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. 1970.

[33] D. Hilbert. Über die darstellung definiter formen als summe von formenquadraten. *Mathematische Annalen*, 32(3):342–350, 1888.

[34] S. Hu, L. Qi, Y. Song, and G. Zhang. Geometric measure of quantum entanglement for multipartite mixed states. *International Journal of Software & Informatics*, 8:317–326, 2014.

[35] S. Hu, L. Qi, and G. Zhang. Computing the geometric measure of entanglement of multipartite pure states by means of non-negative tensors. *Physical Review A*, 93(1):012304, 2016.

[36] Y. Huang, H. Liu, and T. Yu. Smoothing projected cyclic Barzilai–Borwein method for stochastic linear complementarity problems. *International Journal of Computer Mathematics*, 93(7):1188–1199, 2016.

[37] B. Jiang and Y.-H. Dai. Feasible Barzilai–Borwein-like methods for extreme symmetric eigenvalue problems. *Optimization Methods and Software*, 28(4):756–784, 2013.

[38] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86. ACM, 2010.

[39] I. Kerenidis and A. Prakash. Quantum recommendation systems. In C. H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 49:1–49:21, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[40] I. Kerenidis and A. Prakash. Quantum gradient descent for linear systems and least squares. *Physical Review A*, 101(2):022316, 2020.

[41] M. Kholiavchenko. Iterative low-rank approximation for CNN compression. *arXiv preprint arXiv:1803.08995*, 2018.

[42] A. Y. Kitaev. Quantum measurements and the Abelian stabilizer problem. *arXiv preprint quant-ph/9511026*, 1995.

[43] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[44] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on optimization*, 11(3):796–817, 2001.

[45] G. Li, L. Qi, and Q. Wang. Positive semi-definiteness of generalized anti-circulant tensors. *Communications in Mathematical Sciences*, 14(4):941–952, 2016.

[46] G. Li, L. Qi, and Y. Xu. SOS-Hankel tensors: theory and application. *arXiv preprint arXiv:1410.6989*, 2014.

[47] S. Lloyd, M. Mohseni, and P. Rebentrost. Quantum principal component analysis. *Nature Physics*, 10(9):631, 2014.

[48] J.-G. Luque and J.-Y. Thibon. Hankel hyperdeterminants and Selberg integrals. *Journal of Physics A: mathematical and general*, 36(19):5267, 2003.

[49] T. S. Motzkin. The arithmetic-geometric inequality. *Inequalities (Proc. Sympos. Wright-Patterson Air Force Base, Ohio, 1965)*, pages 205–224, 1967.

[50] J. Nie and K. Ye. Hankel tensor decompositions and ranks. *SIAM Journal on Matrix Analysis and Applications*, 40(2):486–516, 2019.

[51] M. A. Nielsen and I. Chuang. Quantum computation and quantum information, 2002.

[52] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[53] L. Omberg, G. H. Golub, and O. Alter. A tensor higher-order singular value decomposition for integrative analysis of DNA microarray data from different studies. *Proceedings of the National Academy of Sciences*, 104(47):18371–18376, 2007.

[54] V. Oropeza and M. Sacchi. Simultaneous seismic data denoising and reconstruction via multichannel singular spectrum analysis. *Geophysics*, 76(3):V25–V32, 2011.

[55] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[56] J.-M. Papy, L. De Lathauwer, and S. Van Huffel. Exponential data fitting using multilinear algebra: the single-channel and multi-channel case. *Numerical linear algebra with applications*, 12(8):809–826, 2005.

[57] E. Pednault, J. A. Gunnels, G. Nannicini, L. Horesh, and R. Wisnieff. Leveraging secondary storage to simulate deep 54-qubit Sycamore circuits. *arXiv preprint arXiv:1910.09534*, 2019.

[58] A. Prakash. *Quantum algorithms for linear algebra and machine learning*. PhD thesis, UC Berkeley, 2014.

[59] L. Qi. Hankel tensors: Associated Hankel matrices and Vandermonde decomposition. *Communications in Mathematical Sciences*, 13(1):113–125, 2015.

[60] L. Qi. A note on the multidimensional moment problem. In *Contemporary Computational Mathematics-A Celebration of the 80th Birthday of Ian Sloan*, pages 1075–1079. Springer, 2018.

[61] L. Qi, H. Chen, and Y. Chen. *Tensor eigenvalues and their applications*, volume 39. Springer, 2018.

[62] L. Qi and Z. Luo. *Tensor analysis: spectral theory and special tensors*, volume 151. SIAM, 2017.

[63] L. Qi, G. Zhang, and G. Ni. How entangled can a multi-party system possibly be? *Physics Letters A*, 382(22):1465–1471, 2018.

[64] M. Raydan. On the Barzilai and Borwein choice of steplength for the gradient method. *IMA Journal of Numerical Analysis*, 13(3):321–326, 1993.

[65] P. Rebentrost, M. Mohseni, and S. Lloyd. Quantum support vector machine for big data classification. *Physical review letters*, 113(13):130503, 2014.

[66] P. Rebentrost, M. Schuld, L. Wossnig, F. Petruccione, and S. Lloyd. Quantum gradient descent and Newton's method for constrained polynomial optimization. *New Journal of Physics*, 21(7):073023, 2019.

[67] P. Rebentrost, A. Steffens, I. Marvian, and S. Lloyd. Quantum singular-value decomposition of nonsparse low-rank matrices. *Physical review A*, 97(1):012327, 2018.

[68] A. Rövid, L. Szeidl, and P. Várlaki. On tensor-product model based representation of neural networks. In *2011 15th IEEE International Conference on Intelligent Engineering Systems*, pages 69–72. IEEE, 2011.

[69] C. Shao. Quantum algorithms to matrix multiplication. *arXiv preprint arXiv:1803.01601*, 2018.

[70] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. IEEE, 1994.

[71] R. S. Smith. Frequency domain subspace identification using nuclear norm minimization and Hankel matrix realizations. *IEEE Transactions on Automatic Control*, 59(11):2886–2896, 2014.

[72] Q. Song, H. Ge, J. Caverlee, and X. Hu. Tensor completion algorithms in big data analytics. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(1):1–48, 2019.

[73] B. Sturmfels. Three open problems on Hankel tensors. *preprint*, 2017.

[74] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 43–50. ACM, 2008.

[75] L. Szeidl, P. Baranyi, Z. Petres, and P. Varlaki. Numerical reconstruction of the HOSVD based canonical form of polytopic dynamic models. In *2007 International Symposium on Computational Intelligence and Intelligent Informatics*, pages 111–116. IEEE, 2007.

[76] E. 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, 2019.

[77] S. Trickett, L. Burroughs, and A. Milton. Interpolation using Hankel tensor completion. In *SEG Technical Program Expanded Abstracts 2013*, pages 3634–3638. Society of Exploration Geophysicists, 2013.

[78] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[79] M. A. O. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *European Conference on Computer Vision*, pages 447–460. Springer, 2002.

[80] Q. Wang, G. Li, L. Qi, and Y. Xu. New classes of positive semi-definite Hankel tensors. *Minimax theory and its applications*, 2:231–248, 2017.

[81] X. Wang, L. Gu, H.-w. J. Lee, and G. Zhang. Quantum tensor singular value decomposition with applications to recommendation systems. *arXiv preprint arXiv:1910.01262*, 2019.

[82] Y. Wang and S. Ma. Projected Barzilai–Borwein method for large-scale nonnegative image restoration. *Inverse Problems in Science and Engineering*, 15(6):559–583, 2007.

[83] P. Wolfe. Convergence conditions for ascent methods. *SIAM review*, 11(2):226–235, 1969.

[84] G. Zhang. Dynamical analysis of quantum linear systems driven by multi-channel multi-photon states. *Automatica*, 83:186–198, 2017.