Quantum Divide-and-Conquer Anchoring for Separable Non-negative Matrix Factorization

Yuxuan Du[1 *], **Tongliang Liu**[1*], **Yinan Li**[2†], **Runyao Duan**[2,3†], **Dacheng Tao**[1*]
[1] UBTECH Sydney AI Centre, SIT, FEIT, University of Sydney, Australia
[2] Centre for Quantum Software and Information, University of Technology Sydney
[3] Institute for Quantum Computing, Baidu Inc., Beijing 100193, China

Abstract

It is NP-complete to find non-negative factors W and H with fixed rank r from a non-negative matrix X by minimizing $\|X - WH^\top\|_F^2$. Although the separability assumption (all data points are in the conical hull of the extreme rows) enables polynomial-time algorithms, the computational cost is not affordable for big data. This paper investigates how the power of quantum computation can be capitalized to solve the non-negative matrix factorization with the separability assumption (SNMF) by devising a quantum algorithm based on the divide-and-conquer anchoring (DCA) scheme [Zhou *et al.*, 2013]. The design of quantum DCA (QDCA) is challenging. In the divide step, the random projections in DCA is completed by a quantum algorithm for linear operations, which achieves the exponential speedup. We then devise a *heuristic post-selection* procedure which extracts the information of anchors stored in the quantum states efficiently. Under a plausible assumption, QDCA performs efficiently, achieves the quantum speedup, and is beneficial for high dimensional problems.

1 Introduction

Non-negative matrix factorization (NMF) [Lee and Seung, 1999; Pauca *et al.*, 2004] is popular in computer vision and machine learning, because the underlying non-negativity constraints on the two low-rank factors usually yield sparse representations of the given non-negative matrix. It has proven that NMF is NP-complete [Vavasis, 2009]. Thus, the separability assumption has been introduced [Donoho and Stodden, 2004] to NMF and induces SNMF. This assumption enables not only polynomial-time algorithms [Zhou *et al.*, 2013; Recht *et al.*, 2012; Van Buskirk *et al.*, 2017], but also a geometric interpretation [Donoho and Stodden, 2004].

However, the rapid progress of the Internet technology, and the computational power and storage, as well as the wide distribute of sensors, grows data exponentially, which challenges many polynomial machine learning algorithms [Lin, 2007; Liu *et al.*, 2017; You *et al.*, 2017; Wang *et al.*, 2017]. Thanks to quantum physics, quantum computing machinery and quantum machine learning are arising [Biamonte *et al.*, 2017]. Many encouraging results have been reported recently, such as quantum support vector machine [Rebentrost *et al.*, 2014] and quantum perceptron [Kapoor *et al.*, 2016], which dramatically reduce the runtime complexity and achieve a more efficient learning ability.

Through exploiting quantum advantages, a logarithmic runtime complexity of SNMF is desired and then many emergent applications can be beneficial from this acceleration. Thus, we need to consider restrictions in quantum computing and answer the following questions: (1) how to convert the classical SNMF problem to accord with a quantum framework; (2) how to exploit quantum advantages, preferring to achieve the exponential speedup; and (3) how to circumvent reading out bottleneck in measurements [Aaronson, 2015].

We select the divide-and-conquer anchoring (DCA) [Zhou *et al.*, 2013] scheme and devise quantum DCA (QDCA) for SNMF, because DCA only contains linear operations in the time-consuming divide step and this characteristic echoes with the nature of quantum computing. This answers the first question and confirms the second question by guaranteeing the exponential speedup for operations in a quantum machine.

Thanks to that indexes of anchors can be sampled with a high probability from a probability distribution in the resulting quantum states, we propose an efficient *heuristic post-selection* method instead of reading out all the quantum data directly (reading is an expensive operation, especially for high-dimensional vectors). This *heuristic post-selection* method answers the third question and ensures us to achieve the quantum speedup after measurements.

The exponential speedup for computations in quantum machine and that for transmitting indexes of anchors from quantum machine to classical computer after measurements together guarantee that the runtime complexity of QDCA achieves $O(poly \log(n + m))$, where $n \times m$ is the size of the input non-negative matrix.

* {yudu5543@uni.,tongliang.liu@,dacheng.tao@}sydney.edu.au
† {Yinan.Li@student.,Runyao.Duan@}uts.edu.au

In addition, we deliver an important message through this paper. QDCA is the first algorithm that seamlessly integrates quantum and classical computations with the exponential speedup. Such kind of integration forms a general strategy to develop algorithms with quantum advantages.

The rest of this paper is organized as follows: Section 2 reviews SNMF and DCA; Section 3 elaborates QDCA and analyzes the runtime complexity; and Section 4 concludes the paper and discusses the future works.

2 Background

NMF aims to approximate a non-negative matrix $X \in \mathbb{R}_+^{n\times m}$ by the product of two nonnegative low rank factors (a basis matrix $W \in \mathbb{R}_+^{n\times r}$ and an encoding matrix $H \in \mathbb{R}_+^{m\times r}$), i.e., $X \approx WH^\top$, where $r = O(\log(n+m)) \ll \min\{n, m\}$, via solving the following optimization problem

$$\begin{aligned} &\min_{W\in\mathbb{R}_+^{n\times r}, H\in\mathbb{R}_+^{m\times r}} \frac{1}{2}\|X - WH^\top\|_F^2 \\ =&\min_{W\in\mathbb{R}_+^{n\times r}, H\in\mathbb{R}_+^{m\times r}} \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}(X_{ij} - (WH^\top)_{ij})^2 \ . \end{aligned} \tag{1}$$

2.1 Separable Non-negative Matrix Factorization

Solving the original NMF problem (equation (1)) is in general NP-complete [Vavasis, 2009]. Thus it is computationally intractable to obtain the global optimum in polynomial time with respect to the input size. To circumvent this difficulty, [Donoho and Stodden, 2004] introduced the *separability assumption* to the non-negative matrix X, i.e., X can be decomposed into $X = FX(R,:)$, where the basis matrix $X(R,:)$ is composited by r rows from X and F is the non-negative encoding matrix.

We denote $R = \{k_1, k_2, ..., k_r\}$, where $k_i \in \{1, 2, ..., n\}$ and $|R| = r$. If a cone can be defined as $cone(X(R,:)) = \sum_{i=1}^{r}\alpha_i X(k_i,:),\ \alpha_i \in \mathbb{R}_+$, the $cone(X(R,:))$ is the conical hull of $X(R,:)$. We say X is separable if $\forall k_i \in \{1, ..., n\}$, we have

$$X(k_i,:) \in cone(X(R,:)),\ X(R,:) = \{X(k_i,:)\}_{k_i\in R}\ , \tag{2}$$

where all the data points in X are covered in a finitely generated and pointed $cone(X(R,:))$.

By adding an extra constraint $\sum_{i=1}^{r}\alpha_i = 1$, we say the simplex $\Delta(X(R,:))$ is the convex hull of $X(R,:)$. This is valuable in practice. The selected rows in $X(R,:)$ are called *anchors* (or extreme rows) of X. The non-anchor vectors in X, which are the rest $n - r$ points in $\mathbb{R}_+^m$, lie in the convex hull or conical hull, generated by the anchors and thus can be non-negatively and linearly expressed by the anchors. Figure 1 shows the geometrical interpretation of convex hull. Likewise, for the near-separable case, all data points are in or around the conical hull of the extreme rows. The concept of near-separable NMF can be straightforwardly defined by $X = FX(R,:) + N$, where N is a noise matrix for convenient reconstruction.

2.2 Divide-and-Conquer Anchoring

Based on a divide-and-conquer scheme that exploits the geometric information of convex hull or conical hull partially

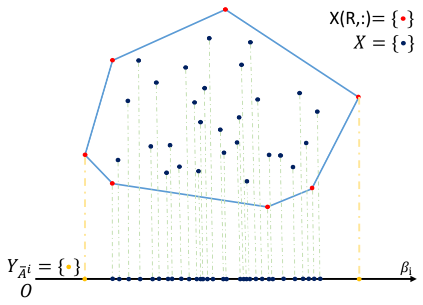

Figure 1: An illustration for SNMF. The red points stand for the anchors. All data of X, except for anchors, are denoted as blue points and are contained in the convex hull, generated by the anchors. After any random projection into 1-dimensional space, the geometric information is still partially preserved, where the anchors in the projected space are denoted as yellow points.

preserved in their projections, divide-and-conquer anchoring (DCA) [Zhou *et al.*, 2013] selects the indexes of anchors from a number of operations in a low-dimensional space, for example the 1-dimensional space used in the rest of the paper. DCA is comprised of two parts: (1) the divide step, targeting the collection of indexes of anchors in the 1-dimensional space, and (2) the conquer step, aiming to determine R via collecting all (for separable case) or high frequency (for near-separable case) indexes of anchors in the 1-dimensional space. To better understand the proposed QDCA, we detail the procedure of DCA.

Divide step. Given a set of unit vectors $B = \{\beta_i\}_{i=1}^{s} \in \mathbb{R}^{m\times s}$ randomly sampled from the unit hypersphere $\mathbb{S}^{m-1}$, where $s = O(r\log r)$, we project X onto β_i and obtain

$$Y_i = X\beta_i,\ Y_i \in \mathbb{R}^n\ . \tag{3}$$

Denote the indexes of the smallest and the largest entries of Y_i in the 1-dimensional space as $\bar{A}^i$, i.e.,

$$\bar{A}^i := \left\{\arg\max_{k_j} X(k_j,:)\beta_i,\ \arg\min_{k_j} X(k_j,:)\beta_i\right\}\ , \tag{4}$$

where $i = \{1, \ldots, s\}$. As illustrated in Figure 1, each $\bar{A}^i$ corresponds to a particular pair of anchors in the original high dimensional space.

Conquer step. Under the separability assumption, we find r distinct indexes to identify all the anchors for X. Under the near-separability assumption, the indexes of anchors in randomly projected spaces are collected by selecting the most r frequently appeared indexes from $\{\bar{A}^i\}_{i=1}^{s}$. The selection rule is defined as

$$R := \arg\max_{H\subseteq[n],|H|=r}\sum_{i\in H}\sum_{j=1}^{s} I(i\in\bar{A}^j)\ , \tag{5}$$

where $|H|$ is the size of the set H, and $I(i \in \bar{A}^j) : i \to \{0, 1\}$ is the indicator function for the event that an index i is within $\bar{A}^j$ of the j-th random projection operation.

3 Quantum Divide-and-Conquer Anchoring

Devising QDCA is challenging and so non-trivial. To realize quantum advantages for solving SNMF, we shall transform the DCA scheme. For simplicity, we decompose DCA into s sub-problems corresponding to s random projections, in which the i-th sub-problem is comprised of the i-th random projection and the subsequent procedure for determining $\bar{A}^i$. It is worth noting that, in QDCA, for preserving the exponential speedup, only the index of an anchor with the maximum absolute value is collected in $\bar{A}^i$, i.e., $\bar{A}^i = \arg\max_{k_j}\{|X(k_j,:)\beta_i|\}$. The random projection can be completed by an efficient quantum algorithm for linear operations [Lloyd *et al.*, 2014], achieving an exponential speedup with respect to the classical counterpart. After the random projections, the resulting vectors will be presented as quantum states (proportional to $X\beta_i$), which are infeasible to readout all its probability amplitudes in a logarithmic runtime. To overcome this barrier and target exponential speedup, we devise a *heuristic post-selection* method to obtain $\{\bar{A}^i\}_{i=1}^s$. In the conquer step, we employ equation (5) in classical computing. Figure 2 shows the diagram of QDCA.

Prior to detail the proposed QDCA, we introduce the notations, which are necessary to explain our results. The *Dirac notations* are used to follow the convention. Basically, we use $|\psi\rangle \in \mathcal{H}_d$, a normalized d-dimensional vector, to denote a d-dimensional pure quantum state in the underlying state (Hilbert) space $\mathcal{H}_d$. We use $\langle\psi|$ to denote the conjugate transpose of $|\psi\rangle$, i.e., $|\psi\rangle = \langle\psi^\dagger|$. The standard inner product of $|\psi\rangle$ and $|\phi\rangle$ is denoted as $\langle\psi|\phi\rangle$. For a quantum system with Hamiltonian H (a hermitian matrix, satisfying $H^\dagger = H$), the unitary time evolution is characterized by the *matrix exponential*, $e^{-iHt} := \sum_{k=0}^{\infty}\frac{1}{k!}(-iH)^k t^k$, where t is the simulation time. To simulate a quantum system, we are required to implement a *quantum circuit* which mimics the time evolution e^{-iHt} at any time t. The spectral decomposition of H is denoted by $\sum_j \lambda_j |u_j\rangle\langle u_j|$, where λ_j and u_j are the eigenvalue and the corresponding eigenvector of H, respectively. Specifically, the matrix exponential e^H admits the form $\sum_j e^{\lambda_j}|u_j\rangle\langle u_j|$. By default, we have $|0\rangle = [1\ 0]^\top$ and $|1\rangle = [0\ 1]^\top$. For n-qubits, let $|i\rangle$ be the computational basis, where $|i\rangle \in \{|0\rangle, |1\rangle\}^{\otimes n}$ and $\otimes$ stands for the operation of tensor product. Detailed basic notations and preliminaries for quantum computation are referring to [Nielsen and Chuang, 2002].

Algorithm 1 summarizes the proposed QDCA algorithm. Here, we discuss its runtime complexity. In this paper, we focus on the dependence of runtime complexity on the parameters m and n, which are dimensions of the input matrix. In total s sub-problems, for the random projection operations, there exists a quantum algorithm which runs in time $O(poly\ \log(m+n))$, achieving the exponential speedup with respect to classical counterparts (see Subsections 3.1 and 3.2). Indexes of anchors in projected spaces $\bar{A}^i$ for $i = \{1, ..., s\}$ will be obtained by a newly designed *heuristic post-selection* method.

Under a plausible assumption, we prove that $O(poly\ \log(m+n))$ runtime is sufficient to locate the s indexes corresponding to the largest absolute amplitude of resulting quantum states with a high probability, which constructs $\{\bar{A}^i\}_{i=1}^s$ and maintains the exponential speedup achieved in the previous steps. However, we have not provided a rigorous runtime analysis for the worst case without any assumption, which is an interesting open problem. In the conquer step (Subsection 3.4), a classical sorting algorithm is employed, which runs in time $O(s\log s)$, where $s = O(r\log r)$ and $r = O(\log(n+m))$. This implies that the conquer step can be completed with a runtime far less than $O(poly\ \log(m+n))$. Under the plausible assumption, the overall runtime of QDCA is $O(ploy\ \log(m+n))$.

Algorithm 1: QDCA

```
Input : $X \in \mathbb{R}_+^{n\times m}$ via oracle access (see Subsection 3.1);
        $s = O(\log(m+n)) \in \mathbb{R}_+$
        $k = O(poly\log(n+m)) \in \mathbb{R}_+$.
Output: The indexes of anchors $R$.
1  if $m \neq n$ or $X$ is not hermitian then
2      $X \leftarrow \begin{pmatrix} 0 & X \\ X^\dagger & 0 \end{pmatrix}$;
3  end
4  Generating random vectors $\{\beta_i\}_{i=1}^s$ and preparing
     corresponding quantum states $\{|\beta_i\rangle\}_{i=1}^s$ (see
     Subsection 3.1);
5  Preparing a quantum circuit to simulate the unitary time
     evolution of $e^{-iXt}$ (see Subsection 3.1);
6  for $i < s$ do
7      Applying the quantum algorithm for linear
         operations to produce $k$ copies of $|\psi_i\rangle$:
8              $|\psi_i\rangle \propto X\beta_i$ (see Subsection 3.2);
9      Measuring $k$ copies of $|\psi_i\rangle$ using the computational
         basis (see Subsection 3.3);
10     Setting $\bar{A}^i = \{l\}$, where $l$ is the outcome that
         appears most frequently in quantum measurements
11     (see Subsection 3.3);
12 end
13 Constructing $R$ by $\{\bar{A}^i\}_{i=1}^s$,
     $R \leftarrow \arg\max_{H\subseteq[n+m],|H|=r}\sum_{i\in H}\sum_{j=1}^s I(i\in\bar{A}^j)$
     (see Subsection 3.4);
```

In the following subsections, we detail the QDCA. Subsection 3.1 introduces how to read classical data into quantum computer. To complete the divide step under a logarithmic runtime, Subsections 3.2 and 3.3 sequentially demonstrate how to employ the quantum algorithm for linear operations to realize random projections and how to devise *heuristic post-selection* for transmitting. Finally, Subsection 3.4 shows that applying a classical sorting algorithm in the conquer step preserves the logarithmic runtime in QDCA.

3.1 Quantum Simulation

For the sake of exploiting quantum advantages, we first describe how to encode the given classical data matrix and random vectors into quantum states.

Without loss of generality, the input of the SNMF problem is a data matrix $\tilde{X} \in \mathbb{R}^{n\times m}$ with $rank(\tilde{X}) \ll \min\{n, m\}$. In the rest of the paper, we focus on the general case, where $m \neq n$. Referring to [Harrow *et al.*, 2009], for satisfying the quantum computing requirement, we should embed $\tilde{X}$ into a

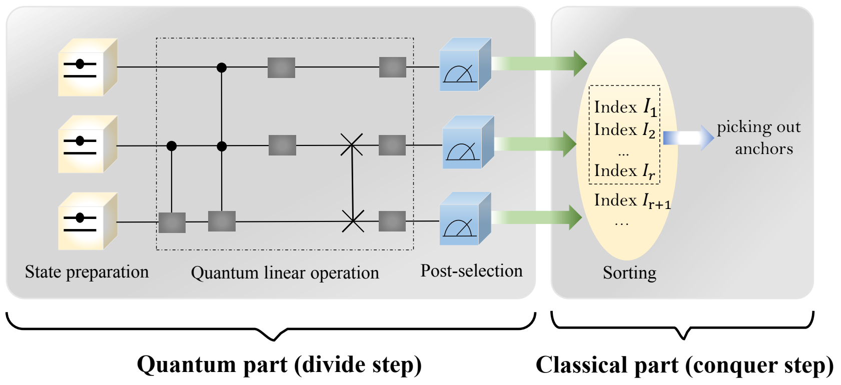

Figure 2: The circuit of QDCA. The circuit is composed of the quantum part and the classical part. In the quantum part, the classical data are encoded into the quantum form and the random projections are performed through the quantum circuit. Then, the *heuristic post-selection* collects indexes set $\{\bar{A}^i\}_{i=1}^s$ with the classical form. Finally, the classical part sorting the top r most frequently appearing indexes from $\{\bar{A}^i\}_{i=1}^s$ as the anchor indexes R.

high-dimensional matrix by employing the "extend matrix", i.e.,

$$X := \begin{pmatrix} 0 & \tilde{X} \\ \tilde{X}^\dagger & 0 \end{pmatrix} \in \mathbb{R}^{(n+m)\times(n+m)}, \quad (6)$$

which is square and hermitian. Naturally, if we can locate the anchors of X, we can then easily obtain the anchors of $\tilde{X}$.

Given a low-rank hermitian matrix $X \in \mathbb{R}^{(n+m)\times(n+m)}$, there exists an oracle, which can efficiently access the elements of X by either performing an efficient computation of the matrix elements or authorizing access to a storage medium for the elements such as quantum RAM [Giovannetti *et al.*, 2008]. With the oracle access to X, we could prepare a quantum circuit which simulates the unitary time evolution e^{-iXt} for a simulation time t, following the method introduced in [Berry *et al.*, 2007; Harrow *et al.*, 2009].

Note that, for an efficient quantum simulation, the input matrix is required to be sparse. Thus, to simulate a dense matrix X that may be encountered in SNMF, we refer to the method introduced in [Rebentrost *et al.*, 2016; Rebentrost *et al.*, 2018]. Specifically, X can be efficiently embedded into a large sparse matrix $S_X \in \mathbb{R}^{(n+m)^2\times(n+m)^2}$. Since S_X is sparse, we can use S_X to conduct an efficient quantum simulation. This "modified swap matrix" S_X is defined as

$$S_X := \sum_{i,j=1}^{n+m} X_{i,j} \left|i\right\rangle \left\langle j\right| \otimes \left|j\right\rangle \left\langle i\right| \in \mathbb{R}^{(n+m)^2\times(n+m)^2}, \quad (7)$$

where $X_{i,j}$ is the (i,j)-th entry of X. It is easy to see that S_X is one-sparse, at which the number of non-zero elements in each column and in each row is one.

We show that, applying the simulated unitary time evolution $e^{-iXt/(n+m)}$ to the target state σ, is equivalent to applying e^{-iS_Xt} on $\rho\otimes\sigma$, where the ancillary state $\rho = |\vec{1}\rangle\langle\vec{1}|$ and $|\vec{1}\rangle = \frac{1}{\sqrt{n+m}}\sum_i |i\rangle$. Since $1/(n+m)$ is only a scale factor that does not influence the final result, e^{-iXt} can therefore be efficiently simulated in quantum computer. We can trace out the ancillary state ρ from the quantum simulation system $e^{-iS_X\Delta t}\rho\otimes\sigma$ by

$$tr_\rho e^{iS_X\Delta t}\rho\otimes\sigma e^{-iS_X\Delta t} = e^{i\frac{X}{(n+m)}\Delta t}\sigma e^{-i\frac{X}{(n+m)}\Delta t} + O(\Delta t^2)\,. \quad (8)$$

This indicates that for small time Δt, evolving with the modified swap matrix S_X on the large system $\rho\otimes\sigma$ is equivalent to evolving with $X/(n+m)$ on the σ system with a negligible error. Then generalizing to any simulation t, we can slice t into multiple Δt. And in each Δt, a copy of ρ is required. With an efficient oracle access to the matrix elements, we can simulate sparse matrix S_X with a constant number of oracle calls and a negligible error [Berry *et al.*, 2007; Harrow *et al.*, 2009].

Random vectors used in DCA will be prepared as quantum states according to [Grover and Rudolph, 2002; Harrow *et al.*, 2009]. Specifically, for a normalized vector $\beta = [\alpha_0, \cdots, \alpha_{(n+m)-1}]^\top \in \mathbb{R}^{(n+m)}$, if all of the entries as well as $\sum_{i=i_1}^{i_2}|\alpha_i|^2$ are efficiently computable, where $i_2 > i_1$ are any numbers in $\{1,\cdots,n+m\}$, we can prepare the state $|\beta\rangle = \sum_{i=1}^{n+m}\alpha_i|i\rangle$ efficiently.

Concluding remark 1. In the state preparation step, given an oracle access to the elements of a low-rank and normalized rows hermitian matrix $X \in \mathbb{R}^{(n+m)\times(n+m)}$, there exists a quantum algorithm [Berry *et al.*, 2007; Harrow *et al.*, 2009] which simulates the unitary time evolution $e^{-iXt/(n+m)}$ in runtime $O(\text{poly}\ \log\,(n+m))$. Likewise, given an oracle access to the elements of classical random and normalized vectors $\{\beta_i\}_{i=1}^s$, we can efficiently prepare corresponding quantum states $\{|\beta_i\rangle\}_{i=1}^s$ under a logarithmic runtime.

3.2 Quantum Algorithm for Linear Operations

After classical data are read into quantum forms, we shall utilize quantum principal component analysis scheme (QPCA) [Lloyd *et al.*, 2014] and its subsequent phase estimation algorithm [Shor, 1999] to obtain a quantum state $|\psi_i\rangle$ which is proportional to random projections $X\beta_i$, for $i = \{1,\ldots,s\}$.

Let the eigen-decomposition of X be $\sum_j \lambda_j |u_j\rangle\langle u_j|$, where λ_j and $|u_j\rangle$ stand for eigenvalues and their corresponding eigenvectors. Specifically, for the dense matrix case, with the oracle $\Lambda_q(\cdot)$, the exponential matrix $e^{-iXt_0/(n+m)}$ with simulation time t_0 is applied to $|\beta_i\rangle$, resulting in

$$\Lambda_q(e^{-iXt_0/(n+m)})\,|k\rangle\,|\beta_i\rangle = \frac{1}{\sqrt{2^q}}\sum_k |k\rangle\, e^{-ikXt_0/(n+m)}\,|\beta_i\rangle\ ,$$

where q is a positive integer, $|k\rangle$ is composed with q qubits (i.e., $|k\rangle = |0\rangle^{\otimes q}$) and the oracle $\Lambda_q(\cdot)$ applies k times of e^{-iXt_0} onto $|\beta_i\rangle$.

Next, taking $|k\rangle$ as the eigenvalue register with quantum operations, we can obtain the quantum state

$$\frac{1}{\sqrt{\sum_j |\beta_j|^2}}\sum_{\frac{|\lambda_j|}{(n+m)}\geq\epsilon} \alpha_j\,|\frac{\lambda_j}{(n+m)}\rangle\,|u_j\rangle\ ,$$

where $\alpha_j = \langle u_j|\beta_i\rangle$ in time $O(1/\epsilon)$.

To obtain the analogous quantum form $X|\beta_i\rangle = \sum_j \lambda_j \langle u_j|\beta_i\rangle\,|u_j\rangle$, that corresponds to the result of the random projection $X\beta_i$, the eigenvalues will be extracted into probability amplitudes of the resulting quantum state. We then follow the procedure in [Harrow *et al.*, 2009]. Specifically, through introducing an ancilla qubit, applying rotating condition on $|\lambda_j\rangle$, and uncomputing the eigenvalue register, the resulting quantum state is proportional to

$$\sum_{\frac{|\lambda_j|}{(n+m)}\geq\epsilon} \alpha_j\,|u_j\rangle \left(\sqrt{1-\frac{\lambda_j^2}{C^2(n+m)^2}}\,|0\rangle + \frac{\lambda_j}{C(n+m)}\,|1\rangle\right)\ ,$$

where $C = O(1/\lambda_{max})$ and λ_{max} is the largest eigenvalue of X.

Measuring the last qubit, conditioned on seeing 1 [Rebentrost *et al.*, 2016], the final output state is proportional to $X\beta_i$, i.e.,

$$|\psi_i\rangle = \frac{1}{\sqrt{\sum_j \frac{|\beta_j\lambda_j|^2}{C^2(n+m)^2}}}\sum_j \alpha_j\frac{\lambda_j}{C(n+m)}\,|u_j\rangle\ . \qquad (9)$$

Concluding remark 2. Given the quantum circuit that simulates $e^{iS_X t}$ and a quantum state $|\beta_i\rangle$ encoding the vector β_i, there exists a quantum algorithm for linear operations that outputs a quantum state $|\psi_i\rangle$, c.f., equation (9), which is proportional to the vector $X\beta_i$, in time $\tilde{O}(1/\epsilon)$. Combined with the quantum circuit and the state preparation step in Subsection 3.1, the total runtime complexity of computing $X\beta_i$ for $i = \{1,\ldots,s\}$ is $O(poly\ \log(n+m)/\epsilon)$. Let the desired error be $1/\epsilon = O(poly\ \log(n+m))$, the runtime complexity is $O(poly\ \log(n+m))$.

3.3 Heuristic Post-Selection

The resulting quantum state $|\psi_i\rangle$, which is proportional to the vector $X\beta_i$, is used to determine $\bar{A}^i$. The method to extract the expected index is non-trivial. In DCA, as the resulting vector is given explicitly after a random projection, we can pick up the indexes with the largest and the smallest entries in time $O(n+m)$. In the quantum setting, the entries of $X\beta_i$ are encoded into the probability amplitudes of $|\psi_i\rangle$. Reading out all probability amplitudes is exponential expensive. Even employing efficient tomography methods, such as compressed sensing [Gross *et al.*, 2010] or sample optimal tomography [Haah *et al.*, 2017], the runtime complexity is $\tilde{O}(n+m)$. Such a large cost breaks the exponential speedup achieved in the random projection step.

For the purpose of preserving the quantum advantages, we devise an alternative heuristic method to obtain $\bar{A}^i$. The *heuristic post-selection* is to find the index of an anchor in the projected space (corresponding to the maximum probability amplitude of $|\psi_i\rangle$) under a logarithmic runtime by consuming N copies of $|\psi_i\rangle$. Given N copies of $|\psi_i\rangle$, the procedure to perform the *heuristic post-selection* is

1. measuring each copy $|\psi_i\rangle$ by the computational basis ;
2. recording the most appearing index among the N outputs as the index of an anchor in the projected space.

The quantum state $|\psi_i\rangle$ contains $(n+m)$ superposition states which correspond to $(n+m)$ indexes, i.e., the possible measurement outcomes are $\{1,\ldots,n+m\}$ and the probability of obtaining the index k ($k\in\{1,\ldots,n+m\}$) is given by $p_k = |\langle k|\psi_i\rangle|^2$. Among the N outputs, the most frequently appearing index corresponds to the index with the largest absolute amplitude with a high probability, which is also the index of an anchor in the projected space. Note that the work [Yu *et al.*, 2016] has proved that collecting largest absolute amplitude can also find all anchors.

The probability of finding the index of an anchor is proportional to the number of quantum state copies $|\psi_i\rangle$, where the index corresponds to the maximum absolute amplitude of the quantum state $|\psi_i\rangle$. Then, a natural question is how many copies are sufficient to determine the index of an anchor in the projected space with a high probability. The number of quantum state copies influences the runtime complexity of a quantum algorithm. Namely, using the computational basis to measure one copy of quantum state requires runtime complexity $O(1)$, and the measurement runtime by the computational basis is proportional to the number of copies. In the following, we show that, under a plausible assumption, only $N = O(poly\ \log(n+m))$ copies of $|\psi_i\rangle$ are sufficient to determine the index of an anchor in the projected space with a high probability.

Theorem 1. *Let D be a multinomial distribution. If $\mathbf{x}\sim D$, we assume $P(\mathbf{x}=i) = p_i, i\in\{1,\cdots,N\}$, and $\sum_{i=1}^N p_i = 1$. Let $x_1,\cdots,x_N$ be examples independently sampled from D and N_i be the number of examples taking value of i. Let $p_{max} = \max\{p_1,\cdots,p_N\}$ and $p_{secmax} = \max\{p_1,\cdots,p_N\}\setminus p_{max}$. If $p_{max} - p_{secmax} > 2\sqrt{2\log(4N/\delta)/N}$, then, for any $\delta > 0$, with a probability at least $1-\delta$, we have*

$$\arg\max_i\{N_i|1\leq i\leq N\} = \arg\max_i\{p_i|1\leq i\leq N\}\ . \quad (10)$$

In Theorem 1, we have a plausible assumption $p_{max} - p_{secmax} > 2\sqrt{2\log(4N/\delta)/N}$, which is easy to satisfy in practice. To achieve the exponential speedup, we could set $N = \log^2(n+m)$ and then we have $p_{max} - p_{secmax} > 2\sqrt{2\log(4\log^2(n+m)/\delta)/\log^2(n+m)}$, which will converge to zero as N goes to infinity. This implies that given the above plausible assumption, by using the proposed *heuristic post-selection* method, we could find the measured index corresponding to the maximum absolute amplitude of the resulting quantum state with a high probability. Here, the measured index also corresponds to the index of an anchor in the projected space. Recall that measuring $O(poly\log(n+m))$ quantum state copies by the computational basis implements *heuristic post-selection* method.

The proof of Theorem 1 is based upon the following Breteganolle-Huber-Carol inequality [Van Der Vaart and Wellner, 1996]:

Theorem 2 (Breteganolle-Huber-Carol inequality)**.** *Let D be a multinomial distribution with l events probabilities $p_1, \cdots, p_l$. Let N_i be the number of event i sampled from randomly sampled N events. Then, for any $\delta > 0$, the following inequality holds*

$$P\left(\sum_{i=1}^{l}\left|\frac{N_i}{N} - p_i\right| \geq \lambda\right) \leq 2^l \exp\left(\frac{-N\lambda^2}{2}\right). \quad (11)$$

Proof of Theorem 1. *By utilizing the Breteganolle-Huber-Carol inequality, for any $j \in \{1, \cdots, N\}$, and $\delta > 0$, we have*

$$P\left(\left|\frac{N_j}{N} - p_j\right| + \left|\sum_{i\neq j}\left(\frac{N_i}{N} - p_i\right)\right| \geq \lambda\right) \leq 4\exp\left(\frac{-N\lambda^2}{2}\right). \quad (12)$$

Let $\delta = 4\exp\left(\frac{-N\lambda^2}{2}\right)$. The above inequality implies that for any given $j \in \{1, \cdots, N\}$, with probability at least $1-\delta$, we have

$$\left|\tfrac{N_j}{N} - p_j\right| + \left|\textstyle\sum_{i\neq j}\left(\tfrac{N_i}{N} - p_i\right)\right| \leq \sqrt{\tfrac{2\log(4/\delta)}{N}}. \quad (13)$$

By using the union bound of probability, we have that for any $\delta > 0$ and any $j \in \{1, \cdots, N\}$, with probability at least $1-\delta$, for the following inequality holds

$$\left|\frac{N_j}{N} - p_j\right| \leq \sqrt{\frac{2\log(4N/\delta)}{N}}. \quad (14)$$

Since $p_{max} - p_{secmax} > 2\sqrt{2\log(4N/\delta)/N}$, it can be easily verified that, with a probability at least $1-\delta$, there is only one value N_j/N that is in the $\sqrt{2\log(4N/\delta)/N}$-neighborhood of p_j. We therefore conclude that $\arg\max_i\{N_i|1 \leq i \leq N\} = \arg\max_i\{p_i|1 \leq i \leq N\}$. ■

We also conduct experiments to test the case without the plausible assumption. We empirically find that, the $O(poly\ \log(n+m))$ measurements are sufficient to locate the index with the largest absolute amplitude at a very high probability. Given limited page length, we do not detail the procedure of the experiment. In a nutshell, we first generate the synthetic data in accordance with [Zhou *et al.*, 2013]. And then, we convert the result of each random projection into a probability distribution. Afterwards, the Monte Carlo simulation is introduced to sample examples from the distribution [Binder *et al.*, 1993]. Finally, the statistical results indicate that, with the sample size $O(poly\ \log(n+m))$, the index with the largest entry can be located with a high probability.

After measuring polynomial logarithmic N copies by the computational basis with runtime $O(poly\log(n+m))$, the most appearing index should be recorded among N outputs, which can be obtained by a classical searching algorithm. The $\bar{A}^i$ then be determined with the runtime $O(poly\log(n+m))$. After applying the *heuristic post-selection* onto s sub-problems, the $\{\bar{A}^i\}_{i=1}^s$ will be obtained. This achieves the divide step of QDCA. It is worth noting that, different to the classical DCA, QDCA only obtains the index with the largest absolute entry value. Therefore, the number of random vectors β_i should be doubled. Since $s \ll \min(n,m)$, we have $2s \ll n+m$ and s is still $O(r\log r)$.

Concluding remark 3. Supported by Theorem 1, when the data size is $(n+m)$, $O(poly\log(n+m))$ random samples measured by computational basis are sufficient to locate the index of an anchor in a projected space with a high probability. Adding the runtime to search the most frequently appeared index, the runtime of this step is $O(poly\log(n+m))$.

3.4 The Classical Conquer Step

Via the *heuristic post-selection* method, $\{\bar{A}^i\}_{i=1}^s$ are collected in the classical form. As analysis in Theorem 1, non-anchors may be collected with probability at most δ. Therefore, equation (5) is applied to determine the indexes of anchors. This part is completed by a classical sorting algorithm. Through employing the sorting algorithm [Knuth, 1998], s indexes of $\{\bar{A}^i\}_{i=1}^s$ can be sorted in time $O(s\log s)$. Since $s = O(r\log r)$, the runtime complexity is $O(poly\log(n+m))$. After sorting, top r indexes are selected as R, which are most frequently appeared indexes in all s sub-problems. With the selected R, the decomposed matrix $X(R,:)$ is obtained.

Concluding remark 4. Through employing the sorting algorithm on $\{\bar{A}^i\}_{i=1}^s$, the indexes of anchors are obtained with runtime complexity $O(poly\log(n+m))$.

4 Conclusion

This paper presents QDCA to dramatically reduce the runtime to achieve the exponential speedup for extracting part-based representations from a large-scale matrix. Analogous to DCA, QDCA can also solve near-separable non-negative matrix factorization problem with the exponential speedup. Moreover, the strategy of combining quantum and classical computations paves a new way to develop quantum machine learning algorithms. Through employing *heuristic post-selection* method, the quantum advantage is achieved after reading out quantum data into the classical form. In the future, we plan to apply this QDCA scheme to various machine learning algorithms to achieve the quantum speedup.

Acknowledgments

This research is supported by ARC projects FL-170100117 and DP-180103424.

References

[Aaronson, 2015] Scott Aaronson. Read the fine print. *Nature Physics*, 11(4):291–293, 2015.

[Berry *et al.*, 2007] Dominic W Berry, Graeme Ahokas, Richard Cleve, and Barry C Sanders. Efficient quantum algorithms for simulating sparse hamiltonians. 2007.

[Biamonte *et al.*, 2017] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195, 2017.

[Binder *et al.*, 1993] Kurt Binder, Dieter Heermann, Lyle Roelofs, A John Mallinckrodt, Susan McKay, et al. Monte carlo simulation in statistical physics. 1993.

[Donoho and Stodden, 2004] David Donoho and Victoria Stodden. When does non-negative matrix factorization give a correct decomposition into parts? 2004.

[Giovannetti *et al.*, 2008] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Physical review letters*, 2008.

[Gross *et al.*, 2010] David Gross, Yi-Kai Liu, Steven T Flammia, Stephen Becker, and Jens Eisert. Quantum state tomography via compressed sensing. *Physical review letters*, 2010.

[Grover and Rudolph, 2002] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions. *arXiv preprint quant-ph/0208112*, 2002.

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

[Harrow *et al.*, 2009] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[Kapoor *et al.*, 2016] Ashish Kapoor, Nathan Wiebe, and Krysta Svore. Quantum perceptron models. In *Advances in Neural Information Processing Systems*, pages 3999–4007, 2016.

[Knuth, 1998] Donald Knuth. Section 5.2. 4: Sorting by merging. 1998.

[Lee and Seung, 1999] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[Lin, 2007] Chih-Jen Lin. On the convergence of multiplicative update algorithms for nonnegative matrix factorization. *IEEE Transactions on Neural Networks*, 2007.

[Liu *et al.*, 2017] Tongliang Liu, Mingming Gong, and Dacheng Tao. Large-cone nonnegative matrix factorization. *IEEE transactions on neural networks and learning systems*, 2017.

[Lloyd *et al.*, 2014] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10(9):631–633, 2014.

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

[Pauca *et al.*, 2004] V Paul Pauca, Farial Shahnaz, Michael W Berry, and Robert J Plemmons. Text mining using non-negative matrix factorizations. 2004.

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

[Rebentrost *et al.*, 2016] Patrick Rebentrost, Maria Schuld, Francesco Petruccione, and Seth Lloyd. Quantum gradient descent and newton's method for constrained polynomial optimization. 2016.

[Rebentrost *et al.*, 2018] Patrick Rebentrost, Adrian Steffens, Iman Marvian, and Seth Lloyd. Quantum singular-value decomposition of nonsparse low-rank matrices. *Physical review A*, 2018.

[Recht *et al.*, 2012] Ben Recht, Christopher Re, Joel Tropp, and Victor Bittorf. Factoring nonnegative matrices with linear programs. 2012.

[Shor, 1999] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[Van Buskirk *et al.*, 2017] Greg Van Buskirk, Ben Raichel, and Nicholas Ruozzi. Sparse approximate conic hulls. In *Advances in Neural Information Processing Systems 30*. 2017.

[Van Der Vaart and Wellner, 1996] Aad W Van Der Vaart and Jon A Wellner. Weak convergence. In *Weak Convergence and Empirical Processes*, pages 16–28. Springer, 1996.

[Vavasis, 2009] Stephen A Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20(3):1364–1377, 2009.

[Wang *et al.*, 2017] Chaoyue Wang, Chang Xu, Chaohui Wang, and Dacheng Tao. Perceptual adversarial networks for image-to-image transformation. 2017.

[You *et al.*, 2017] Shan You, Chang Xu, Yunhe Wang, Chao Xu, and Dacheng Tao. Privileged multi-label learning. *arXiv preprint arXiv:1701.07194*, 2017.

[Yu *et al.*, 2016] Xiyu Yu, Wei Bian, and Dacheng Tao. Scalable completion of nonnegative matrices with the separable structure. 2016.

[Zhou *et al.*, 2013] Tianyi Zhou, Wei Bian, and Dacheng Tao. Divide-and-conquer anchoring for near-separable nonnegative matrix factorization and completion in high dimensions. 2013.