Variational Quantum Singular Value Decomposition

Xin Wang,[1] Zhixin Song,[1] and Youle Wang[1, 2]

[1]*Institute for Quantum Computing, Baidu Research, Beijing 100193, China*
[2]*Centre for Quantum Software and Information, University of Technology Sydney, NSW 2007, Australia*

(Dated: October 27, 2020)

Singular value decomposition is central to many problems in engineering and scientific fields. Several quantum algorithms have been proposed to determine the singular values and their associated singular vectors of a given matrix. Although these algorithms are promising, the required quantum subroutines and resources are too costly on near-term quantum devices. In this work, we propose a variational quantum algorithm for singular value decomposition (VQSVD). By exploiting the variational principles for singular values and the Ky Fan Theorem, we design a novel loss function such that two quantum neural networks (or parameterized quantum circuits) could be trained to learn the singular vectors and output the corresponding singular values. Furthermore, we conduct numerical simulations of VQSVD for random matrices as well as its applications in image compression of handwritten digits. Finally, discuss the applications of our algorithm in recommendation systems and polar decomposition. Our work explores new avenues for quantum information processing beyond the conventional protocols that only works for Hermitian data, and reveals the capability of matrix decomposition on near-term quantum devices.

I. INTRODUCTION

Matrix decompositions are integral parts of many algorithms in optimization [1], machine learning [2], and recommendation systems [3]. One crucial approach is the singular value decomposition (SVD). Mathematical applications of the SVD include computing the pseudoinverse, matrix approximation, and estimating the range and null space of a matrix. SVD has also been successfully applied to many areas of science and engineering industry, such as data compression, noise reduction, and image processing. The goal of SVD is to decompose a square matrix M to $UDV^\dagger$ with diagonal matrix $D = \operatorname{diag}(d_1, \cdots, d_r)$ and unitaries U and V, where r denotes the rank of matrix M.

Quantum computing is believed to deliver new technology to speed up computation, and it already promises speedups for integer factoring [4] and database search [5] in theory. Enormous efforts have been made in exploring the possibility of using quantum resources to speed up other important tasks, including linear system solvers [6–10], convex optimizations [11–14], and machine learning [15–17]. Quantum algorithms for SVD have been proposed in [18, 19], which leads to applications in solving linear systems of equations [9] and developing quantum recommendation systems [18]. However, these algorithms above are too costly to be convincingly validated for near-term quantum devices, which only support a restricted number of physical qubits and limited gate fidelity.

Hence, an important direction is to find useful algorithms that could work on noisy intermediate-scale quantum (NISQ) devices [20]. The leading strategy to solve various problems using NISQ devices are called variational quantum algorithms [21]. These algorithms can be implemented on shallow-depth quantum circuits that depend on external parameters (e.g., angle θ in rotation gates $R_y(\theta)$), which are also known as parameterized quantum circuits or quantum neural networks (QNNs). These parameters will be optimized externally by a classical computer with respect to certain loss functions. Various variational algorithms using QNNs have been proposed for Hamiltonian ground and excited states preparation [22–28], quantum state fidelity estimation [29], quantum Gibbs state preparation [30–32], quantum compiling [33–36] etc. Furthermore, unlike the strong need of error correction in fault-tolerant quantum computation, noise in shallow quantum circuits can be suppressed via error mitigation [37–40], indicating the feasibility of quantum computing with NISQ devices.

In this paper, we formulate the task of SVD as an optimization problem and derive a variational quantum algorithm for singular value decomposition (VQSVD) that can be implemented on near-term quantum computers. The core idea is to construct a novel loss function inspired by the variational principles and properties of singular values. We theoretically show that the optimized quantum neural networks based on this loss function could learn the singular vectors of a given matrix. That is, we could train two quantum neural networks $U(\alpha)$ and $V(\beta)$ to learn the singular vectors of a matrix M in the sense that $M \approx U(\alpha)DV(\beta)^\dagger$, where the diagonal matrix D provides us the singular values. Our approach generalizes the conventional methods of Hamiltonian diagonalization [26, 41] to a non-Hermitian regime, extending the capabilities of matrix decomposition on near-term quantum computers. As a proof of principle, we conduct numerical simulations to estimate the SVD of random 8×8 matrices. Furthermore, we explore the possibility of applying VQSVD to compress images of size 32×32 pixel, including the famous MNIST dataset. Finally, we showcase the applications of VQSVD in recommendation systems and polar decomposition.

II. MAIN RESULTS

A. Variational quantum singular value decomposition

In this section, we present a variational quantum algorithm for singular value decomposition of $n \times n$ matrices, and it can be naturally generalized for $n \times m$ complex matrices. For given $n \times n$ matrix $M \in \mathbb{R}^{n\times n}$, there exists a decomposition of the form $M = UDV^\dagger$, where $U, V \in \mathbb{R}^{n\times n}$ are unitary operators and D is a diagonal matrix with r positive entries $d_1, \cdots, d_r$ and r is the rank of M. Alternatively, we write $M = \sum_{j=1}^{r} d_j |u_j\rangle\langle v_j|$, where $|u_j\rangle$, $|v_j\rangle$, and d_j are the sets of left and right orthonormal singular vectors, and singular values of M, respectively.

A vital issue in NISQ algorithm is to choose a suitable loss function. A desirable loss function here should be able to output the target singular values and vectors after the optimization and in particular should be implementable on near-term devices. As a key step, we design such a desirable loss function for quantum singular value decomposition (cf. Section II B).

The input of our VQSVD algorithm is a decomposition of the matrix M into a linear combination of K unitaries of the form $M = \sum_{k=1}^{K} c_k A_k$ with real numbers c_k. For example, we could assume that M can be decomposed into a linear combination of Pauli terms. After taking in the inputs, our VQSVD algorithm enters a hybrid quantum-classical optimization loop to train the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in the parameterized quantum circuits $U(\boldsymbol{\alpha})$ and $V(\boldsymbol{\beta})$ via a designed loss $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$ (cf. Section II B). This loss function can be computed on quantum computers via the Hadamard tests. We then feeds the value of the loss function or its gradients (in gradient-based optimization) to a classical computer, which adjusts the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for the next round of the loop. The goal is to find the global minimum of $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$, i.e., $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^* := \arg\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} L(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

In practice, one will need to set some termination condition (e.g., convergence tolerance) on the optimization loop. After the hybrid optimization, one will obtain values $\{m_j\}_{j=1}^{T}$ and optimal parameters $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$. The outputs $\{m_j\}_{j=1}^{T}$ approximate the singular values of M, and approximate singular vectors of M are obtained by inputting optimal parameters $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$ into the parameterized circuits U and V in VQSVD and then applying to the orthonormal vectors $|\psi_j\rangle$ for all $j = 1, ..., T$. The detailed VQSVD algorithm is included in Algorithm 1. A schematic diagram is shown in Fig. 1.

Algorithm 1 Variational quantum singular value decomposition (VQSVD)

```
1: Input: $\{c_k, A_k\}_{k=1}^{K}$, desired rank $T$, parametrized circuits $U(\boldsymbol{\alpha})$ and $V(\boldsymbol{\beta})$ with initial parameters of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and tolerance $\varepsilon$;
2: Prepare positive numbers $q_1 > \cdots q_T > 0$;
3: Choose computational basis $|\psi_1\rangle, \cdots, |\psi_T\rangle$;
4: for $j = 1, \cdots, T$ do
5:     Apply $U(\boldsymbol{\alpha})$ to state $|\psi_j\rangle$ and obtain $|u_j\rangle = U(\boldsymbol{\alpha})|\psi_j\rangle$;
6:     Apply $V(\boldsymbol{\beta})$ to state $|\psi_j\rangle$ and obtain $|v_j\rangle = V(\boldsymbol{\beta})|\psi_j\rangle$ ;
7:     Compute $m_j = \mathrm{Re}\langle u_j|M|v_j\rangle$ via Hadamard tests;
8: end for
9: Compute the loss function $L(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^{T} q_j m_j$;
10: Perform optimization to maximize $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$, update parameters of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$;
11: Repeat 4-10 until the loss function $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$ converges with tolerance $\varepsilon$;
12: Output $\{m_j\}_{j=1}^{T}$ as the largest $T$ singular values, output $U(\boldsymbol{\alpha}^*)$ and $V(\boldsymbol{\beta}^*)$ as corresponding unitary operators ($\langle\psi_j|U(\boldsymbol{\alpha}^*)^\dagger$ and $V(\boldsymbol{\beta}^*)|\psi_j\rangle$ are left and right singular vectors, respectively).
```

B. Loss function

In this section, we provide more details and intuitions of the loss function in VQSVD. The key idea is to exploit the variational principles in matrix computation, which have great importance in analysis for error bounds of matrix analysis . In particular, the singular values satisfy a subtler variational property that incorporates both left and right singular vectors at the same time. For a given $n \times n$ matrix M, the largest singular value of M can be characterized by

$$d_1 = \max_{|u\rangle, |v\rangle} \frac{|\langle u|M|v\rangle|}{\|u\|\|v\|} = \max_{|u\rangle, |v\rangle \in \mathcal{S}} \mathrm{Re}[\langle u|M|v\rangle], \qquad (1)$$

where $\mathcal{S}$ is the set of pure states (normalized vectors) and Re means taking the real part. Moreover, by denoting the optimal singular vectors as $|u_1\rangle, |v_1\rangle$, the remaining singular values ($d_2 \geq \cdots \geq d_r$) can be deduced using similar methods by restricting the unit vectors to be orthogonal to previous singular value vectors.

For a given $n \times n$ matrix M, the largest singular value of M can be characterized by

$$d_1 = \max_{|u\rangle, |v\rangle \in \mathcal{S}} \mathrm{Re}[\langle u|M|v\rangle], \qquad (2)$$

where $\mathcal{S}$ is the set of pure states (normalized vectors) and $\mathrm{Re}[\cdot]$ means to take the real part. Moreover, by denoting the optimal singular vectors as $|u_1\rangle, |v_1\rangle$, the remaining singular values ($d_2 \geq \cdots \geq d_r$) can be deduced as follows

$$\begin{aligned} d_k = \max\ & \mathrm{Re}[\langle u|M|v\rangle] \\ \text{s.t.}\ & |u\rangle, |v\rangle \in \mathcal{S}, \\ & |u\rangle \perp \mathrm{span}\{|\mathrm{u}_1\rangle, \cdots, |\mathrm{u}_{\mathrm{k}-1}\rangle\}, \\ & |v\rangle \perp \mathrm{span}\{|\mathrm{v}_1\rangle, \cdots, |\mathrm{v}_{\mathrm{k}-1}\rangle\}. \end{aligned} \qquad (3)$$

Another useful fact (Ky Fan Theorem, cf. [42, 43]) is that

$$\sum_{j=1}^{T} d_j = \max_{\text{orthonormal } \{u_j\}, \{v_j\}} \sum_{j=1}^{T} \langle u_j|M|v_j\rangle. \qquad (4)$$

For a given matrix M, the loss function in our VQSVD algorithm is defined as

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^{T} q_j \times \mathrm{Re}\langle\psi_j|U(\boldsymbol{\alpha})^\dagger M V(\boldsymbol{\beta})|\psi_j\rangle, \qquad (5)$$

where $q_1 > \cdots > q_T > 0$ are real weights and $\{\psi_j\}_{j=1}^{T}$ is a set of orthonormal states.

Theorem 1 *For a given matrix M, the loss function $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is maximized if only if*

$$\langle\psi_j|U(\boldsymbol{\alpha})^\dagger M V(\boldsymbol{\beta})|\psi_j\rangle = d_j,\ \forall\, 1 \leq j \leq T, \qquad (6)$$

where $d_1 \geq \cdots \geq d_T$ are the largest T singular values of M and $|\psi_1\rangle, \cdots, |\psi_T\rangle$ are orthonormal vectors. Moreover, $\langle\psi_j|U(\boldsymbol{\alpha})^\dagger$ and $V(\boldsymbol{\beta})|\psi_j\rangle$ are left and right singular vectors, respectively.

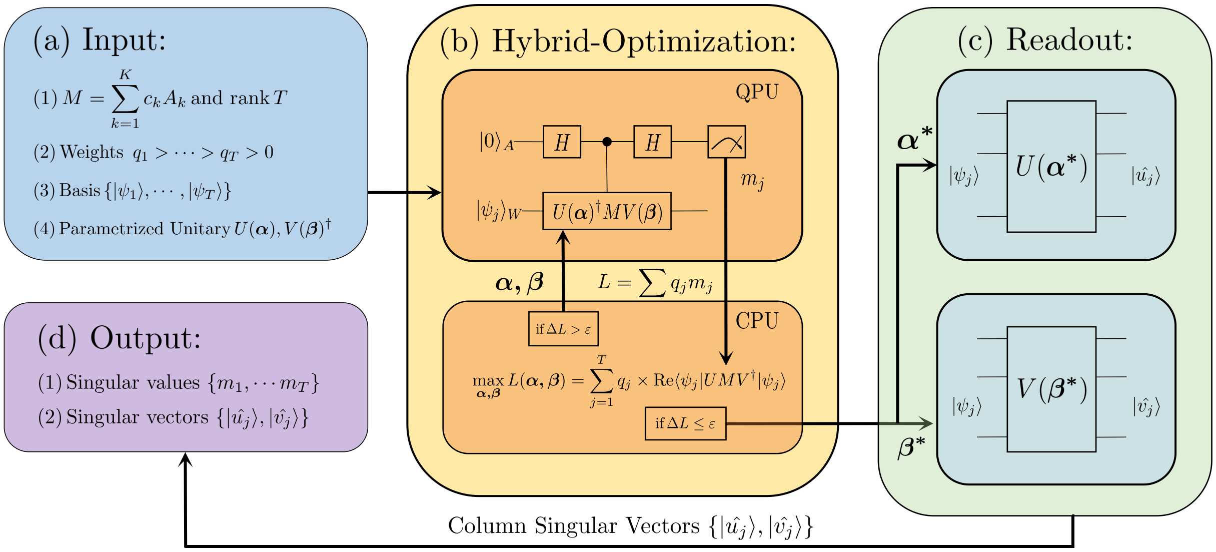
(a) Input:
(1) $M = \sum_{k=1}^{K} c_k A_k$ and rank T
(2) Weights $q_1 > \cdots > q_T > 0$
(3) Basis $\{|\psi_1\rangle, \cdots, |\psi_T\rangle\}$
(4) Parametrized Unitary $U(\boldsymbol{\alpha}), V(\boldsymbol{\beta})^\dagger$
(b) Hybrid-Optimization:
QPU
$|0\rangle_A$
H
H
m_j
$|\psi_j\rangle_W$
$U(\boldsymbol{\alpha})^\dagger M V(\boldsymbol{\beta})$
$\boldsymbol{\alpha}, \boldsymbol{\beta}$
$L = \sum q_j m_j$
CPU
if $\Delta L > \varepsilon$
$\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} L(\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_{j=1}^{T} q_j \times \mathrm{Re}\langle\psi_j|UMV^\dagger|\psi_j\rangle$
if $\Delta L \leq \varepsilon$
(c) Readout:
$\boldsymbol{\alpha}^*$
$|\psi_j\rangle$
$U(\boldsymbol{\alpha}^*)$
$|\hat{u}_j\rangle$
$\boldsymbol{\beta}^*$
$|\psi_j\rangle$
$V(\boldsymbol{\beta}^*)$
$|\hat{v}_j\rangle$
(d) Output:
(1) Singular values $\{m_1, \cdots m_T\}$
(2) Singular vectors $\{|\hat{u}_j\rangle, |\hat{v}_j\rangle\}$
Column Singular Vectors $\{|\hat{u}_j\rangle, |\hat{v}_j\rangle\}$

FIG. 1: Schematic diagram including the steps of VQSVD algorithm. (a) The unitary decomposition of matrix M is provided as the first input. This can be achieved through Pauli decomposition with tensor products of $\{X, Y, Z, I\}$. The algorithm also requires the desired number of singular values T, orthonormal input states $\langle\psi_i|\psi_j\rangle = \delta_{ij}$ and two parametrized unitary matrices $U(\boldsymbol{\alpha}), V(\boldsymbol{\beta})$. Finally, the weight is usually set to be integers $q_j = T + 1 - j$. The former two information will be sent to the hybrid-optimization loop in (b) where the quantum computer (QPU) will estimate each singular value $m_j = \mathrm{Re}\langle\psi_j|U^\dagger MV|\psi_j\rangle$ via Hadamard test. These estimations are sent to a classical computer (CPU) to evaluate the loss function until it converges to tolerance ε. Once we reach the global minimum, the singular vectors $\{|\hat{u}_j\rangle, |\hat{v}_j\rangle\}$ can be produced in (c) by applying the learned unitary matrices $U(\boldsymbol{\alpha}^*)$ and $V(\boldsymbol{\beta}^*)$ on orthonormal basis $\{|\psi_j\rangle\}_{j=1}^{T}$ to extract the column vectors.

Proof Let assume that $\langle\psi_j|U(\boldsymbol{\alpha})^\dagger MV(\boldsymbol{\beta})|\psi_j\rangle = m_j$ are real numbers for simplicity, which could be achieved after the ideal maximization process.

Then we have

$$L(\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_{j=1}^{T} q_j \times m_j \tag{7}$$

$$\leq \sum_{j=1}^{T} q_j \times m_j^{\downarrow} \tag{8}$$

$$= \sum_{j=1}^{T} (q_j - q_{j+1}) \sum_{t=1}^{j} m_t^{\downarrow} \tag{9}$$

$$\leq \sum_{j=1}^{T} (q_j - q_{j+1}) \sum_{t=1}^{j} d_t. \tag{10}$$

Assume $q_{T+1} = 0$. The first inequality Eq. (8) follows due to the rearrangement inequality. The second inequality Eq. (10) follows due to property of singular values in Eq. (4). Note that the upper bound in Eq. (10) could be achieved if and only if $\sum_{t=1}^{j} d_t = \sum_{t=1}^{j} m_t$ for all j, which is equivalent to

$$m_j = d_j, \quad \forall\, 1 \leq j \leq T. \tag{11}$$

Therefore, the loss function is maximized if and only if $\langle\psi_j|U(\boldsymbol{\alpha})^\dagger MV(\boldsymbol{\beta})|\psi_j\rangle$ extracts the singular values d_j for each j from 1 to T. Further, due to the variational property of singular values in Eq. (2) and Eq. (3), we conclude that the quantum neural networks U and V learn the singular vectors in the sense that $\langle\psi_j|U(\boldsymbol{\alpha})^\dagger$ and $V(\boldsymbol{\beta})|\psi_j\rangle$ extract the left and right singular vectors, respectively. ■

Proposition 2 *The loss function $L(\boldsymbol{\alpha},\boldsymbol{\beta})$ can be estimated on near-term quantum devices.*

For $M = \sum_{k=1}^{K} c_k A_k$ with unitaries A_k and real numbers c_k, the quantity $\mathrm{Re}\langle\psi_j|U(\boldsymbol{\alpha})^\dagger MV(\boldsymbol{\beta})|\psi_j\rangle$ can be decomposed to

$$\sum_{k=1}^{K} c_k \times \mathrm{Re}\langle\psi_j|U(\boldsymbol{\alpha})^\dagger A_k V(\boldsymbol{\beta})|\psi_j\rangle. \tag{12}$$

To estimate the quantity in Eq. (12), we could use quantum subroutines for estimating the quantity $\mathrm{Re}\langle\psi|U|\psi\rangle$ for a general unitary U. One of these subroutines is to utilize the well-known Hadamard test [44], which requires only one ancillary qubit, one copy of state $|\psi\rangle$, and one controlled unitary operation U, and hence it can be experimentally implemented on near term quantum hardware. To be specific, Hadamard test (see Fig. 2) starts with state $|+\rangle_A|\psi\rangle_W$, where A denotes the ancillary qubit, and W denotes the work register, and then apply a controlled unitary U, conditioned on the qubit in register A, to prepare state $\frac{1}{\sqrt{2}}(|0\rangle_A|\psi\rangle_W + |1\rangle_A U|\psi\rangle_W)$, at last, apply a Hadamard gate on the ancillary qubit, and measure. If the measurement outcome is 0, then let the output be 1; otherwise, let the output be -1, and the expectation of output is $\mathrm{Re}\langle\psi|U|\psi\rangle$. As for the imaginary part $\mathrm{Im}\langle\psi|U|\psi\rangle$, it also can be estimated via Hadamard test by starting with state $\frac{1}{\sqrt{2}}(|0\rangle_A + i|1\rangle_A)|\psi\rangle_W$.

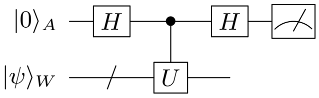

FIG. 2: Quantum circuit for implementing Hadamard test

C. As a generalization of the VQE family

In this subsection, we discuss the connection between our VQSVD algorithm and the variational quantum eigensolver (VQE) [22], which estimates the ground-state energy given a molecular electronic-structure Hamiltonian H. This is a central problem to quantum chemistry as many properties of the molecule can be calculated once we determine the eigenstates of H. Several related algorithms have been proposed to improve the spectrum learning capability of VQE (i.e., SSVQE [28]) such that the properties of excited states can also be explored. We consider these algorithms as a unified family and promising applications for quantum chemistry.

For practical reasons, one has to discretize the aforementioned Hamiltonian H into a series of Pauli tensor products $\sigma_i \otimes \cdots \otimes \sigma_j$ to work on quantum devices. Many physical models naturally fit into this scheme including the quantum Ising model and the Heisenberg Model. In particular, if the input of VQSVD in Eq. (5) is a Hamiltonian in a discretized form (i.e., a linear combination of Pauli tensor products or equivalently a Hermitian matrix), VQSVD could be naturally applied to diagonalize this Hamiltonian and prepare the eigenstates. Therefore, VQSVD can be seen as a generalization of the VQE family that works not only for Hamiltonians, but also for more general non-Hermitian matrices.

III. OPTIMIZATION OF THE LOSS FUNCTION

Finding optimal parameters $\{\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*\}$ is a significant part of variational hybrid quantum-classical algorithms. Both gradient-based and gradient-free methods could be used to do the optimization. Here, we provide analytical details on the gradient-based approach, and we refer to [45] for more information on the optimization subroutines in variational quantum algorithms. Reference about gradients estimation via quantum devices can be found in Ref. [27, 46, 47].

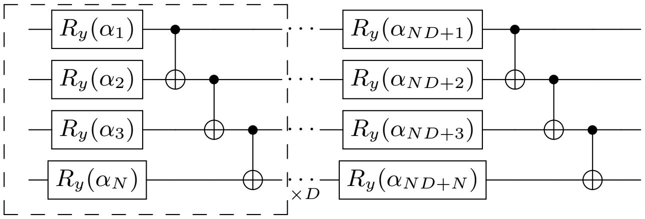

FIG. 3: Variational quantum circuit used in the simulation for both $U(\boldsymbol{\alpha})$ and $V(\boldsymbol{\beta})$. The parameters are optimized to minimize the loss function $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$. D denotes the number of repetitions of the same block (denoted in the dashed-line box) consists of a column of single-qubit rotations about the y-axis $R_y(\alpha_j)$ following by a layer of CNOT gates which only connects the adjacent qubits (hardware-efficient). Notice the number of parameters in this ansatz increases linearly with the circuit depth D and the number of qubits N.

A. Gradients estimation

Here, we discuss the computation of the gradient of the global loss function $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$ by giving an analytical expression, and show that the gradients can be estimated by shifting parameters of the circuit used for evaluating the loss function. In Algorithm 1, to prepare states $|u_j\rangle$ and $|v_j\rangle$, we apply gate sequences $U = U_{\ell_1}...U_1$ and $V = V_{\ell_2}...V_1$ in turn to state $|\psi_j\rangle$, where each gate U_l and V_k are either fixed, e.g., C-NOT gate, or parameterized, for all $l = 1, ..., \ell_1$ and $k = 1, ..., \ell_2$. The parameterized gates U_l and V_k have forms $U_l = e^{-iH_l\alpha_l/2}$ and $V_k = e^{-iQ_k\beta_k/2}$, respectively, where α_l and β_k are real parameters, and H_l and Q_k are tensor products of Pauli matrices. Hence the gradient of loss function L is dependent on parameters $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ and the following proposition shows it can be computed on near-term quantum devices.

Proposition 3 *The gradient of loss function $L(\boldsymbol{\alpha}, \boldsymbol{\beta})$ can be estimated on near-term quantum devices and its explicitly form is defined as follows,*

$$\nabla L(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \left(\frac{\partial L}{\partial \alpha_1}, ..., \frac{\partial L}{\partial \alpha_{\ell_1}}, \frac{\partial L}{\partial \beta_1}, ..., \frac{\partial L}{\partial \beta_{\ell_2}}\right). \tag{13}$$

Particularly, the derivatives of L with respect to α_l and β_k can be computed using following formulas, respectively,

$$\frac{\partial L}{\partial \alpha_l} = \frac{1}{2} L(\boldsymbol{\alpha}_l, \boldsymbol{\beta}), \tag{14}$$

$$\frac{\partial L}{\partial \beta_k} = \frac{1}{2} L(\boldsymbol{\alpha}, \boldsymbol{\beta}_k), \tag{15}$$

where notations $\boldsymbol{\alpha}_l$ and $\boldsymbol{\beta}_k$ denote parameters $\boldsymbol{\alpha}_l = (\alpha_1, ..., \alpha_l + \pi, ..., \alpha_{\ell_1})$ and $\boldsymbol{\beta}_k = (\beta_1, ..., \beta_k - \pi, ..., \beta_{\ell_2})$.

Proof Notice that the partial derivatives of loss function are given by Eqs. (14) (15), and hence the gradient is computed by shifting the parameters of circuits that are used to evaluate the loss function. Since the loss function can be estimated on near term quantum devices, claimed in Proposition 2, thus, the gradient can be calculated on near-term quantum devices.

The derivations of Eqs. (14) (15) use the fact that $\partial_\theta(\mathrm{Re} z(\boldsymbol{\theta})) = \mathrm{Re}(\partial_\theta z(\boldsymbol{\theta}))$, where $z(\boldsymbol{\theta})$ is a parameterized complex number, and more details of derivation are deferred to Appendix A. ■

B. Barren plateaus

It has been shown that when employing hardware-efficient ansatzes (applying 2-qubit gates only to adjacent qubits) [48], the global cost functions like the expectation value $\langle \hat{O} \rangle = \mathrm{Tr}[\hat{O} U(\boldsymbol{\theta}) \rho U(\boldsymbol{\theta})^\dagger]$ of an observable $\hat{O}$ are untrainable for large problem sizes since they exhibit exponentially vanishing gradients which makes the optimization landscape flat (i.e., barren plateaus [49]). Such barren plateaus happens even when the ansatz is short depth [50] and the work [51] showed that the barren plateau phenomenon could also arise in the architecture of dissipative quantum neural networks [52–54].

Such barren plateau issues can be avoided by either implementing identity-block initialization strategy or employing the technique of local cost [50], where the local cost function is defined such that one firstly construct a local observable $\hat{O}_L$, calculating the expectation value with respect to each individual qubit rather than gathering information in a global sense, and finally adding up all the local contributions. The latter strategy has been verified to extend the trainable circuit depth up to $D \in \mathcal{O}(poly(log(N)))$ where N denotes qubit number. Recently, a new method of constructing adaptive Hamiltonians during the optimization loop has been proposed to resolve similar issues [55]. Moreover, one may also employ gradient-free optimization methods [56] such as genetic algorithms, simulated annealing, and Nelder–Mead method [57] to address the barren plateau challenges.

IV. NUMERICAL EXPERIMENTS AND APPLICATIONS

Here we numerically simulate the VQSVD algorithm with randomly generated 8×8 non-Hermitian real matrices as a proof of concept. Then, to demonstrate the possibility of scaling VQSVD to larger and more exciting applications, we simulate VQSVD to compress some standard gray-scale 32×32 images. Fig. 3 shows the variational ansatz used. We choose the input states to be $\{|\psi_j\rangle\} = \{|000\rangle, |001\rangle, \cdots, |111\rangle\}$ and the circuit depth D is set to be $D = 20$. The parameters $\{\alpha, \beta\}$ are initialized randomly from an uniform distribution $[0, 2\pi]$. All simulations and optimization loop are implemented via Paddle Quantum [58] on the PaddlePaddle Deep Learning Platform [59, 60].

A. Three-Qubit example

The VQSVD algorithm described in Section II A can find T largest singular values of matrix $M_{n\times n}$ at once. Here, we choose the weight to be positive integers $(q_1, \cdots, q_T) = (T, T-1, \cdots, 1)$. Fig. 4 shows the learning process. One can see that this approach successfully find the desired singular values and one can reconstruct the given matrix M with inferred singular values and vectors $M_{re}^{(T)} = \sum_{j=1}^{T} m_j |\hat{u_j}\rangle\langle\hat{v_j}|$. The distance between M_{re} and the original matrix M is taken to be the matrix 2-norm $||A_{n\times n}||_2 = \sqrt{\sum_{i,j=1}^{n} |a_{ij}|^2}$ where a_{ij} are the matrix elements. As illustrated in Fig. 4, the distance decreases as more and more singular values being used.

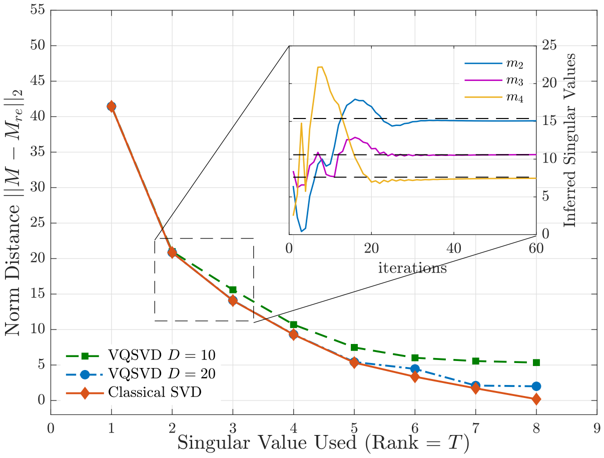

FIG. 4: Distance measure between the reconstructed matrix M_{re} and the original matrix M via VQSVD with different circuit depth $D = \{10, 20\}$ and compare with the classical SVD method. Specifically, we plot the learning process (subspace) of selected singular values $\{m_2, m_3, m_4\}$ when using the layered hardware-efficient ansatz illustrated in Fig. 3 with circuit depth $D = 20$.

B. Image compression

Next, we apply the VQSVD algorithm to compress a 32×32 pixel handwritten digit image taken from the MNIST dataset with only 7.81% (choose rank to be $T = 5$) of its original information. By comparing with the classical SVD method, one can see that the digit #7 is successfully reconstructed with some background noise. Notice the circuit structure demonstrated in Fig. 3 is ordinary and it is a not well-studied topic for circuit architecture. Although we don't know how to load such images into quantum data at this stage, this simulation shows the scaling potential for the VQSVD algorithm. Future studies are needed to efficiently load classical information into NISQ devices.

V. SOLUTION QUALITY ESTIMATION

After obtaining the results (singular values and vectors) from Algorithm 1, it is natural and desirable to have a method to benchmark or verify these results. In this section, we further introduce a procedure for verification purposes. Particularly, we propose a variational quantum algorithm for estimating the sum of largest T squared singular values, i.e.,

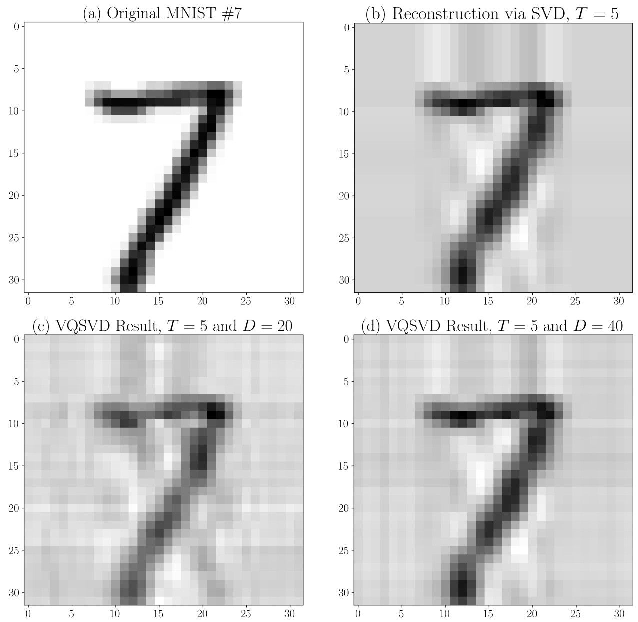

FIG. 5: Performance of simulated 5-qubit VQSVD for image compression. (a) shows the original handwritten digit $\#7$ and it is compressed via classical SVD up to the largest 5 singular values in (b). The performance of VQSVD is presented in (c) and (d) with the same rank $T = 5$ but different circuit depth $D = \{20, 40\}$. An experimental trick to reach better performance is setting the desired rank slightly higher $T = 7$ during the learning process and later use only $T = 5$ subspace to reconstruct the information. This works because the more weights we put on the better accuracy we will get. Here, we choose the weight to be $(32, 28, \cdots)$.

$\sum_{j=1}^T d_j^2$, of a matrix $M \in \mathbb{C}^{m\times n}$ as a subroutine. In the following, we first define the error of the inferred singular values and singular vectors, and then show its estimation via a tool provided in Algorithm 3.

Let $\{m_j\}_{j=1}^T$ denote the inferred singular values of the matrix M that are arranged in descending order, and let $\{|\hat{u}_j\rangle\}_{j=1}^T$ $\left(\{|\hat{v}_j\rangle\}_{j=1}^T\right)$ denote the associated inferred left (right) singular vectors. The error of inferred singular values is defined as follows,

$$\epsilon_d = \sum_{j=1}^T (d_j - m_j)^2, \tag{16}$$

where d_js are the exact singular values of matrix M and also arranged in descending order. And the error ϵ_v of inferred singular vectors is defined below,

$$\epsilon_v = \sum_{j=1}^T \parallel H|\hat{e}_j^+\rangle - m_j|\hat{e}_j^+\rangle \parallel^2 + \sum_{j=1}^T \parallel H|\hat{e}_j^-\rangle + m_j|\hat{e}_j^-\rangle \parallel^2, \tag{17}$$

where H is a Hermitian of the form $H = |0\rangle\langle 1|\otimes M + |1\rangle\langle 0|\otimes M^\dagger$, and $|\hat{e}_j^\pm\rangle = (|0\rangle|\hat{u}_j\rangle \pm |1\rangle|\hat{v}_j\rangle)/\sqrt{2}$. It is worth pointing out that when inferred vectors $|\hat{e}_j^\pm\rangle$ approximate the eigenvectors $|e_j^\pm\rangle$ of H, i.e., $\epsilon_v \to 0$, inferred singular vectors $|\hat{u}_j\rangle$ and $|\hat{v}_j\rangle$ approximate the singular vectors $|u_j\rangle$ and $|v_j\rangle$ respectively, and vice versa.

Now we are ready to introduce the procedure for verifying the quality of VQSVD outputs. To quantify the errors, we exploit the fact that errors ϵ_d and ϵ_v are upper bounded. Specifically speaking,

$$\epsilon_d \le \sum_{j=1}^T d_j^2 - \sum_{j=1}^T m_j^2, \tag{18}$$

$$\epsilon_v \le 2(\sum_{j=1}^T d_j^2 - \sum_{j=1}^T m_j^2), \tag{19}$$

We refer the detailed proofs for Eqs. (18), (19) to Lemma 4 included in Appendix B.

Thus, we only need to evaluate the sum $\sum_{j=1}^T d_j^2$, which is the Frobenius norm of matrix M, and the sum $\sum_{j=1}^T m_j^2$ independently. The latter can be directly computed from the outputs of VQSVD while the former quantity can be estimated through Algorithm 3 presented in Appendix B.

VI. APPLICATIONS

In this section, we give a brief discussion about the application of VQSVD in recommendation systems, and some other problems such as polar decomposition.

a. Recommendation systems The goal of a recommendation system is to provide personalized items to customers based on their ratings of items, or other information. Usually, the preferences of customers are modeled by an $m \times n$ matrix $\mathbf{A}$, where we assume there are m customers and n items. The element $\mathbf{A}_{lj}$ indicates the level that customer l rates the item j. In particular, finding suitable recommendations means to locate the large entries of rows.

One commonly used method for locating the large entries of rows is matrix reconstruction. Given an arbitrary matrix and integer k, the goal is to output a small k-rank matrix as an approximate matrix. In recommendation system, such an low-rank approximate matrix $\mathbf{A}_k$ of the preference matrix $\mathbf{A}$ is usually used to recommend items. In this sense, our VQSVD algorithm finds its application in recommendation system by learning such a small rank matrix. To be more specific, the information about its singular values and vectors can be obtained.

After obtaining the low rank matrix $\mathbf{A}_k$, the process of recommendation for customer l proceeds as follows: project the lth row vector of $\mathbf{A}$ onto the space spanned by the right singular vectors of $\mathbf{A}_k$. Specifically, denote the right singular vectors of $\mathbf{A}_k$ by $|v_1\rangle, ..., |v_k\rangle$, and projection operator by $\Pi := \sum_{t=1}^k |v_t\rangle\langle v_t|$, then apply Π to the lth normalized row vectors $|\mathbf{b}\rangle$ of $\mathbf{A}$. Specially, let $|\widehat{\mathbf{b}}\rangle$ denote the normalized vector after projection, given below:

$$|\widehat{\mathbf{b}}\rangle = \frac{\Pi|\mathbf{b}\rangle}{\parallel \Pi|\mathbf{b}\rangle \parallel} = \frac{1}{G}\sum_{t=1}^k \xi_t|v_t\rangle, \tag{20}$$

where the right singular vectors $\{|v_t\rangle\}$ form an orthogonal basis, ξ_t are the corresponding coefficients, and G denotes the normalization factor, i.e., $G = \sqrt{\sum_{t=1}^k (\xi_t)^2}$.

The state $|\widehat{\mathbf{b}}\rangle$ can be efficiently prepared via paramerized circuit $V(\boldsymbol{\beta})$ as long as we can prepare the state $\frac{1}{G}\sum_{t=1}^k \xi_t|\psi_t\rangle$. Particularly, there are k coefficients ξ_t in Eq. (20), determining $|\widehat{\mathbf{b}}\rangle$. Further, we present a procedure for estimating these coefficients in Algorithm 2 (shown below).

Algorithm 2

```
1: Input: matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, desired rank $k$, parametrized circuits $U(\boldsymbol{\alpha})$ and $V(\boldsymbol{\beta})$ with initial parameters of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and tolerance $\varepsilon$, and a circuit $U_b$ to prepare state $|\mathbf{b}\rangle \in \mathbb{R}^n$;
2: Run VQSVD in Algorithm 1 with $\mathbf{A}$, $k$, $U(\boldsymbol{\alpha})$, $V(\boldsymbol{\beta})$, $\varepsilon$, and return the optimal parameters $\boldsymbol{\alpha}^*$, $\boldsymbol{\beta}^*$, and $\{\sigma_j\}_{j=1}^k$ as singular values;
3: for $t = 1, \cdots, k$ do
4:     Apply $V(\boldsymbol{\beta})$ to state $|\psi_t\rangle$ and obtain $|v_t\rangle = V(\boldsymbol{\beta}^*)|\psi_t\rangle$;
5:     Apply $U_b$ to state $|\mathbf{0}\rangle$ and obtain $|\mathbf{b}\rangle = U_b|\mathbf{0}\rangle$;
6:     Compute $\xi_t = \langle v_t|\mathbf{b}\rangle$ via Hadamard test;
7: end for
8: Output $\{\xi_t\}$ as coefficients in Eq. (20).
```

Finally, measure $|\widehat{\mathbf{b}}\rangle$ in the computational basis and output the outcome.

b. Polar decomposition and other applications Another important application of our VQSVD algorithm is finding the polar decomposition which has many applications in linear algebra [61, 62]. Recently, Lloyd et al [63] proposed a quantum polar decomposition algorithm that performs in a deterministic way. For a given a complex matrix $M \in \mathbb{C}^{n\times n}$, the right polar decomposition is defined as follows,

$$M = WP, \tag{21}$$

where W is a unitary and P is a positive-semidefinite Hermitian matrix. Particularly, suppose the singular value decomposition of M is calculated through the VQSVD algorithm $M = UDV^\dagger$, then $W = UV^\dagger$ and $P = VDV^\dagger$. It is interesting to explore whether our VQSVD could be applied to polar decomposition.

For Hermitian matrices, the VQSVD algorithm can be applied as an eigensolver since singular value decomposition reduces to spectral decomposition in this case. Recently, some work has been proposed to extract eigenvalues and reconstruct eigenvectors of Hermitian matrices [28], density operators [64]. Our VQSVD algorithm may also be applied to Schmidt decomposition of bipartite quantum states since SVD is the core subroutine. We also note that a recent work Bravo-Prieto et al. [65] introduces a quantum singular value decomposer in the sense that it could output Schmidt coefficients and associated orthonormal vectors of a bipartite pure state.

VII. DISCUSSION AND OUTLOOK

To summarize, we have presented a variational quantum algorithm for singular value decomposition with NISQ devices. One key contribution is to design a loss function that could be used to train the quantum neural networks to learn the left and right singular vectors and output the target singular values. Further improvements on the performance of our VQSVD algorithm may be done for sparse matrices together with more sophisticated ansatzes. We have numerically verified our algorithm for singular value decomposition of random matrices and image compression and proposed extensive applications in solving linear systems of equations. As a generalization of the family of VQE algorithms on Hamiltonian diagonalization (or spectral decomposition), the VQSVD algorithm may have potential applications in quantum chemistry, quantum machine learning. and quantum optimization in the NISQ era.

One future direction is to develop near-term quantum algorithms for non-negative matrix factorization [3] which have various applications and broad interests in machine learning. See [66] as an example of the quantum solution. Another interesting direction is to develop near-term quantum algorithms for higher-order singular value decomposition [67].

Acknowledgements

We would like to thank Runyao Duan for helpful discussions. This work was done when Z. S. was a research intern at Baidu Research. Y. W. acknowledged support from the Baidu-UTS AI Meets Quantum project and the Australian Research Council (Grant No: DP180100691).

[1] S. Boyd and L. Vandenberghe, *Convex optimization* (Cambridge university press, 2004).

[2] K. P. Murphy, *Machine learning: a probabilistic perspective* (MIT press, 2012).

[3] Y. Koren, R. Bell, and C. Volinsky, Computer **42**, 30 (2009).

[4] P. W. Shor, SIAM Journal on Computing **26**, 1484 (1997), arXiv:9508027 [quant-ph] .

[5] L. K. Grover, in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96* (ACM Press, New York, New York, USA, 1996) pp. 212–219, arXiv:9605043 [quant-ph] .

[6] A. W. Harrow, A. Hassidim, and S. Lloyd, Physical Review Letters **103**, 150502 (2009).

[7] B. D. Clader, B. C. Jacobs, and C. R. Sprouse, Physical Review Letters **110**, 250504 (2013), arXiv:1301.2340 .

[8] A. M. Childs, R. Kothari, and R. D. Somma, SIAM Journal on Computing **46**, 1920 (2017), arXiv:1511.02306 .

[9] L. Wossnig, Z. Zhao, and A. Prakash, Physical Review Letters **120**, 050502 (2018), arXiv:1704.06174 .

[10] Y. Subasi, R. D. Somma, and D. Orsucci, Physical Review Letters **122**, 060504 (2018), arXiv:1805.10549 .

[11] F. G. Brandao and K. M. Svore, in *2017 IEEE 58th Annual Sym-*

posium on Foundations of Computer Science (FOCS) (IEEE, 2017) pp. 415–426, arXiv:1609.05537 .

[12] S. Chakrabarti, A. M. Childs, T. Li, and X. Wu, Quantum **4**, 221 (2020).

[13] F. G. S. L. Brandão, A. Kalev, T. Li, C. Y.-Y. Lin, K. M. Svore, and X. Wu, "Quantum SDP Solvers: Large Speed-Ups, Optimality, and Applications to Quantum Learning," (2019).

[14] J. van Apeldoorn, A. Gilyén, S. Gribling, and R. de Wolf, Quantum **4**, 220 (2020).

[15] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature **549**, 195 (2017).

[16] M. Schuld and F. Petruccione, *Supervised Learning with Quantum Computers*, Quantum Science and Technology (Springer International Publishing, Cham, 2018).

[17] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **474**, 20170551 (2018).

[18] I. Kerenidis and A. Prakash, arXiv preprint arXiv:1603.08675 , 1 (2016), arXiv:1603.08675 .

[19] P. Rebentrost, A. Steffens, I. Marvian, and S. Lloyd, Physical Review A **97**, 012327 (2018), arXiv:1607.05404 .

[20] J. Preskill, Quantum **2**, 79 (2018), arXiv:1801.00862 .

[21] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, New Journal of Physics **18**, 023023 (2016), arXiv:1509.04279 .

[22] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien, Nature Communications **5**, 4213 (2014).

[23] J. R. McClean, M. E. Kimchi-Schwartz, J. Carter, and W. A. de Jong, Physical Review A **95**, 042308 (2017), arXiv:1603.05681 .

[24] O. Higgott, D. Wang, and S. Brierley, Quantum **3**, 156 (2019), arXiv:1805.08138 .

[25] P. K. Barkoutsos, G. Nannicini, A. Robert, I. Tavernelli, and S. Woerner, Quantum **4**, 256 (2020), arXiv:1907.04769 .

[26] K. M. Nakanishi, K. Mitarai, and K. Fujii, Physical Review Research **1**, 033062 (2019).

[27] M. Ostaszewski, E. Grant, and M. Benedetti, arXiv:1905.09692 , 1 (2019), arXiv:1905.09692 .

[28] K. M. Nakanishi, K. Mitarai, and K. Fujii, Physical Review Research **1**, 033062 (2019).

[29] M. Cerezo, A. Poremba, L. Cincio, and P. J. Coles, arXiv:1906.09253 (2019), 10.22331/q-2020-03-26-248, arXiv:1906.09253 .

[30] J. Wu and T. H. Hsieh, Physical Review Letters **123**, 220502 (2019), arXiv:1811.11756 .

[31] A. N. Chowdhury, G. H. Low, and N. Wiebe, , 1 (2020), arXiv:2002.00055 .

[32] Y. Wang, G. Li, and X. Wang, arXiv preprint arXiv:2005.08797 (2020), arXiv:2005.08797 .

[33] S. Khatri, R. LaRose, A. Poremba, L. Cincio, A. T. Sornborger, and P. J. Coles, Quantum **3**, 140 (2019), arXiv:1807.00800 .

[34] K. Heya, Y. Suzuki, Y. Nakamura, and K. Fujii, (2018), arXiv:1810.12745 .

[35] T. Jones and S. C. Benjamin, , 1 (2018), arXiv:1811.03147 .

[36] K. Sharma, S. Khatri, M. Cerezo, and P. J. Coles, New Journal of Physics **22**, 043006 (2020).

[37] K. Temme, S. Bravyi, and J. M. Gambetta, Physical Review Letters **119**, 180509 (2017), 1612.02058 .

[38] S. McArdle, X. Yuan, and S. Benjamin, Physical Review Letters **122**, 180501 (2019), arXiv:1807.02467 .

[39] A. Strikis, D. Qin, Y. Chen, S. C. Benjamin, and Y. Li, arXiv:2005.07601 , 1 (2020), arXiv:2005.07601 .

[40] A. Kandala, K. Temme, A. D. Córcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta, Nature **567**, 491 (2019), arXiv:1805.04492 .

[41] R. Larose, A. Tikku, É. O. Neel-judy, L. Cincio, and P. J. Coles, npj Quantum Information (2019), 10.1038/s41534-019-0167-6.

[42] K. Fan, Proceedings of the National Academy of Sciences of the United States of America **37**, 760 (1951).

[43] Z. Zhang, arXiv:1510.08532 (2015), arXiv:1510.08532 .

[44] D. Aharonov, V. Jones, and Z. Landau, Algorithmica **55**, 395 (2009), arXiv:0511096 [quant-ph] .

[45] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Quantum Science and Technology **4**, 043001 (2019), arXiv:1906.07682 .

[46] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Physical Review A **98**, 032309 (2018), arXiv:1803.00745 .

[47] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Physical Review A **99**, 032331 (2019).

[48] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature **549**, 242 (2017), arXiv:1704.05018 .

[49] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Nature Communications **9**, 4812 (2018), arXiv:1803.11173 .

[50] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, arXiv:2001.00550 (2020), arXiv:2001.00550 .

[51] K. Sharma, M. Cerezo, L. Cincio, and P. J. Coles, arXiv:2005.12458 (2020), arXiv:2005.12458 .

[52] K. Poland, K. Beer, and T. J. Osborne, arXiv:2003.14103 (2020), arXiv:2003.14103 .

[53] D. Bondarenko and P. Feldmann, Physical Review Letters **124**, 130502 (2020), arXiv:1910.09169 .

[54] K. Beer, D. Bondarenko, T. Farrelly, T. J. Osborne, R. Salzmann, D. Scheiermann, and R. Wolf, Nature Communications **11**, 808 (2020).

[55] M. Cerezo, K. Sharma, A. Arrasmith, and P. J. Coles, arXiv:2004.01372 , 1 (2020), arXiv:2004.01372 .

[56] Journal of Global Optimization **56**, 1247 (2013).

[57] J. A. Nelder and R. Mead, The computer journal **7**, 308 (1965).

[58] "Paddle Quantum," (2020).

[59] "https://github.com/paddlepaddle/paddle," .

[60] Y. Ma, D. Yu, T. Wu, and H. Wang, Frontiers of Data and Domputing **1**, 105 (2019).

[61] N. J. Higham, SIAM Journal on Scientific and Statistical Computing **7**, 1160 (1986).

[62] J. C. Gower, G. B. Dijksterhuis, *et al.*, *Procrustes problems*, Vol. 30 (Oxford University Press on Demand, 2004).

[63] S. Lloyd, S. Bosch, G. De Palma, B. Kiani, Z.-W. Liu, M. Marvian, P. Rebentrost, and D. M. Arvidsson-Shukur, arXiv preprint arXiv:2006.00841 (2020).

[64] R. LaRose, A. Tikku, É. O'Neel-Judy, L. Cincio, and P. J. Coles, npj Quantum Information **5**, 1 (2019).

[65] C. Bravo-Prieto, D. Garcia-Martin, and J. I. Latorre, arXiv:1905.01353 (2019), arXiv:1905.01353 .

[66] Y. Du, T. Liu, Y. Li, R. Duan, and D. Tao, in *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (2018) pp. 2093–2099.

[67] T. G. Kolda and B. W. Bader, SIAM review **51**, 455 (2009).

[68] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, arXiv:1909.05820 (2019), arXiv:1909.05820 .

Appendix A: Proof details for Proposition 3

In this section, we show a full derivation on the gradients of the loss function in our VQSVD algorithm.

$$L(\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_{j=1}^{T} q_j \times \mathrm{Re}\langle\psi_j|U^\dagger(\boldsymbol{\alpha})MV(\boldsymbol{\beta})|\psi_j\rangle \tag{A1}$$

The real part can be estimated via Hadamard test. Equivalently saying that,

$$\mathrm{Re}\langle\psi_j|\hat{O}|\psi_j\rangle = \frac{1}{2}\Big[\langle\psi_j|\hat{O}|\psi_j\rangle + \langle\psi_j|\hat{O}^\dagger|\psi_j\rangle\Big] \tag{A2}$$

Consider the parametrized quantum circuit $U(\boldsymbol{\alpha}) = \Pi_{i=n}^{1}U_i(\alpha_i)$ and $V(\boldsymbol{\beta}) = \Pi_{j=m}^{1}V_j(\beta_j)$. For convenience, denote $U_{i:j} = U_i\cdots U_j$. We can write the cost function as:

$$\begin{aligned}L(\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{1}{2}\sum_{j=1}^{T} q_j \times \Big[&\langle\psi_j|U^\dagger_{1:n}(\alpha_{1:n})MV_{m:1}(\beta_{m:1})|\psi_j\rangle \\ &+ \langle\psi_j|V^\dagger_{1:m}(\beta_{1:m})M^\dagger U_{n:1}(\alpha_{n:1})|\psi_j\rangle\Big]\end{aligned} \tag{A3}$$

Absorb most gates into state $|\psi_j\rangle$ and matrix M,

$$\begin{aligned}L(\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{1}{2}\sum_{j=1}^{T} q_j \times \Big[&\langle\phi_j|U^\dagger_\ell(\alpha_\ell)GV_k(\beta_k)|\varphi_j\rangle \\ &+ \langle\varphi_j|V^\dagger_k(\beta_k)G^\dagger U_\ell(\alpha_\ell)|\phi_j\rangle\Big]\end{aligned} \tag{A4}$$

where $|\varphi_j\rangle = V_{k-1:1}|\psi_j\rangle$, $|\phi_j\rangle = U_{\ell-1:1}|\psi_j\rangle$ and $G \equiv U^\dagger_{\ell+1:n}MV_{m:k+1}$. We assume $U_\ell = e^{-i\alpha_\ell H_\ell/2}$ is generated by a Pauli product H_ℓ and same for $V_k = e^{-i\beta_k Q_k/2}$. The derivative with respect to a certain angle is

$$\frac{\partial U_{1:n}}{\partial\alpha_\ell} = -\frac{i}{2}U_{1:\ell-1}(H_\ell U_\ell)U_{\ell+1:n} \tag{A5}$$

$$\frac{\partial V_{1:m}}{\partial\beta_k} = -\frac{i}{2}V_{1:k-1}(Q_k V_k)V_{k+1:m} \tag{A6}$$

Thus the gradient is calculated to be

$$\begin{aligned}\frac{\partial L(\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial\alpha_\ell} = \frac{1}{2}\sum_{j=1}^{T} q_j \times \Big[&\frac{i}{2}\langle\phi_j|H^\dagger_\ell U^\dagger_\ell(\alpha_\ell)GV_k(\beta_k)|\varphi_j\rangle \\ &- \frac{i}{2}\langle\varphi_j|V^\dagger_k(\beta_k)G^\dagger H_\ell U_\ell(\alpha_\ell)|\phi_j\rangle\Big]\end{aligned} \tag{A7}$$

With the following property, we can absorb the Pauli product H_ℓ by an rotation on $\alpha_\ell \rightarrow \alpha_\ell + \pi$

$$\begin{aligned}U_\ell(\pm\pi) &= e^{\mp i\pi H_\ell/2} \\ &= \cos(\frac{\pi}{2})I \mp i\sin(\frac{\pi}{2})H_\ell \\ &= \mp iH_\ell\end{aligned} \tag{A8}$$

Plug it back and we get,

$$\begin{aligned}\frac{\partial L(\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial\alpha_\ell} = \frac{1}{4}\sum_{j=1}^{T} q_j \times \Big[&\langle\phi_j|U^\dagger_\ell(\alpha_\ell+\pi)GV_k(\beta_k)|\varphi_j\rangle \\ &+ \langle\varphi_j|V^\dagger_k(\beta_k)G^\dagger U_\ell(\alpha_\ell+\pi)|\phi_j\rangle\Big]\end{aligned} \tag{A9}$$

This can be further simplified as

$$\begin{aligned}\frac{\partial L(\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial\alpha_\ell} &= \frac{1}{2}\sum_{j=1}^{T} q_j \times \mathrm{Re}\langle\psi_j|U^\dagger(\alpha_\ell+\pi)MV(\boldsymbol{\beta})|\psi_j\rangle \\ &= \frac{1}{2}L(\alpha_\ell+\pi,\boldsymbol{\beta})\end{aligned} \tag{A10}$$

Similarly, for β_k we have

$$\begin{aligned}\frac{\partial L(\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial\beta_k} = \frac{1}{2}\sum_{j=1}^{T} q_j \times \Big[&\frac{i}{2}\langle\phi_j|U^\dagger_\ell(\alpha_\ell)GQ_kV_k(\beta_k)|\varphi_j\rangle \\ &- \frac{i}{2}\langle\varphi_j|Q^\dagger_k V^\dagger_k(\beta_k)G^\dagger U_\ell(\alpha_\ell)|\phi_j\rangle\Big]\end{aligned} \tag{A11}$$

This can be further simplified as

$$\begin{aligned}\frac{\partial L(\boldsymbol{\alpha},\boldsymbol{\beta})}{\partial\beta_k} &= \frac{1}{2}\sum_{j=1}^{T} q_j \times \mathrm{Re}\langle\psi_j|U^\dagger(\boldsymbol{\alpha})MV(\beta_k-\pi)|\psi_j\rangle \\ &= \frac{1}{2}L(\boldsymbol{\alpha},\beta_k-\pi).\end{aligned} \tag{A12}$$

One can see from the above derivation that calculating the analytical gradient of VQSVD algorithm simply means rotating a specific gate parameter (angle α_ℓ or β_k) by $\pm\pi$ which can be easily implemented on near-term quantum devices.

Appendix B: Supplemental material for verification of the solution quality

In this section, we provide the necessary proofs in Sec. V and detailed discussions on variational quantum Frobenius norm estimation algorithm.

1. Definitions

Recall that the error of inferred singular values is defined as follows,

$$\epsilon_d = \sum_{j=1}^{T}(d_j - m_j)^2, \tag{B1}$$

where d_j are the exact singular values of matrix M and also arranged in descending order. And the error ϵ_v of inferred singular vectors is defined below,

$$\epsilon_v = \sum_{j=1}^{T} \parallel H|\hat{e}_j^+\rangle - m_j|\hat{e}_j^+\rangle \parallel^2 + \sum_{j=1}^{T} \parallel H|\hat{e}_j^-\rangle + m_j|\hat{e}_j^-\rangle \parallel^2, \tag{B2}$$

where H is a Hermitian of the form $H = |0\rangle\langle 1| \otimes M + |1\rangle\langle 0| \otimes M^\dagger$, and $|\hat{e}_j^\pm\rangle = (|0\rangle|\hat{u}_j\rangle \pm |1\rangle|\hat{v}_j\rangle)/\sqrt{2}$. The quantity $\|H|\hat{e}_j^\pm\rangle \mp m_j|\hat{e}_j^\pm\rangle\|^2$ quantifies the component of $H|\hat{e}_j^\pm\rangle$ that is perpendicular to $|\hat{e}_j^\pm\rangle$, which follows from $(I - |\hat{e}_j^\pm\rangle\langle\hat{e}_j^\pm|)H|\hat{e}_j^\pm\rangle$.

It is worth pointing out that when inferred vectors $|\hat{e}_j^\pm\rangle$ approximate the eigenvectors $|e_j^\pm\rangle$, where $|e_j^\pm\rangle = (|0\rangle|u_j\rangle \pm |1\rangle|v_j\rangle)/\sqrt{2}$, of H, i.e., $\epsilon_v \to 0$, inferred singular vectors $|\hat{u}_j\rangle$ and $|\hat{v}_j\rangle$ approximate the singular vectors $|u_j\rangle$ and $|v_j\rangle$ respectively, and vice versa. On the other hand, the error ϵ_v which is used to quantify the extent that vectors $|\hat{e}_j^\pm\rangle$ approximate eigenvector $|e_j^\pm\rangle$ can quantify the extent that inferred vectors $\{u_j\}$ and $\{v_j\}$ approximate the singular vectors. Specifically, these distances have an equal relation, which is depicted in the following equation.

$$D(\{|u_j\rangle, |v_j\rangle\}, \{|\hat{u}_j\rangle, |\hat{v}_j\rangle\}) = D(\{|e_j^+\rangle, |e_j^-\rangle\}, \{|\hat{e}_j^+\rangle, |\hat{e}_j^-\rangle\}), \tag{B3}$$

where D denotes the distance between vectors.

Here, we give the explicit forms of the distances . (B3). The distances between $\{|u_j\rangle, |v_j\rangle\}$ and $\{|\hat{u}_j\rangle, |\hat{v}_j\rangle\}$ are defined in the following form,

$$D(\{|u_j\rangle, |v_j\rangle\}, \{|\hat{u}_j\rangle, |\hat{v}_j\rangle\}) \tag{B4}$$
$$\equiv \parallel |u_j\rangle - |\hat{u}_j\rangle \parallel^2 + \parallel |v_j\rangle - |\hat{v}_j\rangle \parallel^2 . \tag{B5}$$

And the distances between $|e_j^\pm\rangle$ and $|\hat{e}_j^\pm\rangle$ are defined below,

$$D(\{|e_j^+\rangle, |e_j^-\rangle\}, \{|\hat{e}_j^+\rangle, |\hat{e}_j^-\rangle\}) \tag{B6}$$
$$\equiv \parallel |e_j^+\rangle - |\hat{e}_j^+\rangle \parallel^2 + \parallel |e_j^-\rangle - |\hat{e}_j^-\rangle \parallel^2 . \tag{B7}$$

Notice that both of the Right-Hand-Sides of Eqs. (B5), (B7) are equivalent to $4 - 2(\mathrm{Re}\,\langle u_j|\hat{u}_j\rangle + \mathrm{Re}\,\langle v_j|\hat{v}_j\rangle)$, and then the relation in Eq. (B3) follows.

2. Error analysis

After running VQSVD, it would be ideal if we can verify the quality of the outputs from VQSVD. For achieving this purpose, we show that these error ϵ_d and ϵ_v are upper bounded and give the explicit form of upper bounds. We present the derivation for upper bounds on errors in the following lemma.

Lemma 4 *Given a matrix $M \in \mathbb{C}^{n\times n}$, let ϵ_d and ϵ_v denote the errors of the inferred singular values and singular vectors in Eqs.* (B1), (B2)*, respectively, then both of them are upper bounded. To be more specific,*

$$\epsilon_d \le \sum_{j=1}^{T} d_j^2 - \sum_{j=1}^{2} m_j^2,$$
$$\epsilon_v \le 2(\sum_{j=1}^{T} d_j^2 - \sum_{j=1}^{T} m_j^2),$$

where d_js are singular values of matrix M, and m_js are inferred singular values from Algorithm 1.

Proof Recall the definitions of ϵ_d and ϵ_v in Eqs. (B1), (B2), and notice that

$$\epsilon_d = \sum_{j=1}^{T} d_j^2 - 2\sum_{j=1}^{T} d_j m_j + \sum_{j=1}^{T} m_j^2. \tag{B8}$$

Since the dot product with decreasingly ordered coefficients is Schur-convex and $\{d_j\}$ majorize $\{m_j\}$, i.e., $\sum_{j=1}^{\ell} d_j \ge \sum_{j=1}^{\ell} m_j$ for all $\ell = 1, ..., T$, then $\sum_{j=1}^{T} d_j m_j \ge \sum_{j=1}^{T} m_j^2$, which results an upper bound on error ϵ_d.

Note that the error ϵ_v can be rewritten as

$$\epsilon_v = \sum_{j=1}^{T} (\langle\hat{e}_j^+|H^2|\hat{e}_j^+\rangle + \langle\hat{e}_j^-|H^2|\hat{e}_j^-\rangle - 2\sum_{j=1}^{T} m_j^2, \tag{B9}$$

and eigenvectors $\{|\hat{e}_j^\pm\rangle\}$ can be expanded into a basis of the space, then we have

$$\sum_{j=1}^{T} (\langle\hat{e}_j^+|H^2|\hat{e}_j^+\rangle + \langle\hat{e}_j^-|H^2|\hat{e}_j^-\rangle \tag{B10}$$
$$\le \mathrm{Tr}(H^2) = 2\sum_{j=1}^{T} d_j^2. \tag{B11}$$

■

3. A tool for solution quality estimation

The goal of this section is to provide a tool that computes the sum of largest T squared singular values, which is used in Sec. V to analyze the accuracy of outputs of VQSVD. In the following, we mainly show the correctness analysis of Algorithm 3. The detailed discussions on loss evaluation and gradients estimation are omitted, since we can employ the same methods introduced in Ref. [68] to loss evaluation and gradients derivation. The differences of our method from that in Ref. [68] occur in input states. Specifically, we input computational states $|\psi_j\rangle$ for all j into the circuit, while they input state $|\mathbf{0}\rangle$. For more information on loss evaluation and gradients derivation, we refer the interested readers to Ref. [68].

Algorithm 3

```
1:  Input: ${c_k, A_k}_{k=1}^K$, desired rank $T$, parametrized circuits $U(\boldsymbol{\alpha})$
    and $V(\boldsymbol{\beta})$ with initial parameters of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and tolerance $\varepsilon$;
2:  Choose computational basis $|\psi_1\rangle, \cdots, |\psi_T\rangle$;
3:  for $j = 1, \cdots, T$ do
4:  Apply $U(\boldsymbol{\alpha})$ to state $|\psi_j\rangle$ and obtain $|u_j\rangle = U(\boldsymbol{\alpha})|\psi_j\rangle$;
5:  Apply $V(\boldsymbol{\beta})$ to state $|\psi_j\rangle$ and obtain $|v_j\rangle = V(\boldsymbol{\beta})|\psi_j\rangle$ ;
6:  Compute $o_j = |\langle u_j|M|v_j\rangle|^2$ via Hadamard test;
7:  end for
8:  Compute the loss function $F(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^T o_j$;
9:  Perform optimization to maximize $F(\boldsymbol{\alpha}, \boldsymbol{\beta})$, update parameters
    of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$;
10: Repeat 4-10 until the loss function $F(\boldsymbol{\alpha}, \boldsymbol{\beta})$ converges with tol-
    erance $\varepsilon$;
11: Output $F(\boldsymbol{\alpha}, \boldsymbol{\beta})$ as Frobenius norm.
```

a. Correctness analysis The validity of Algorithm 3 follows from a fact that, for arbitrary matrix, its squared singular values majorize the squared norms of diagonal elements. Specifically, the sum of the largest T squared singular values is larger than the sum of squared norms of the largest T diagonal elements. We summarize this fact in the lemma below and further provide a proof.

Lemma 5 *For arbitrary matrix $M \in \mathbb{C}^{N\times N}$, let singular values of M be σ_1, σ_2,...,σ_N, which are arranged in descending order. Then for any $k \in [N]$, we have the following inequality,*

$$\sum_{j=1}^{k} \sigma_j^2 \geq \sum_{j=1}^{k} |D_j^{\downarrow}|^2, \tag{B12}$$

where D is the diagonal vector of M, i.e., $D = \mathrm{diag}(M)$, the notation $\downarrow$ means that the elements are arranged in descending order. The equality in Eq. (B12) holds if and only if M is diagonal.

Proof The core of the proof is to build relationships between singular values and diagonal elements, to be specific,

$$\sum_{j=1}^{k} \sigma_j^2 \geq \sum_{j=1}^{k} \vec{M}_j^{\downarrow}, \tag{B13}$$

$$\sum_{j=1}^{k} \vec{M}_j^{\downarrow} \geq \sum_{j=1}^{k} |D_j^{\downarrow}|^2, \tag{B14}$$

where $\vec{M}$ is the diagonal vector of $MM^\dagger$.

Next, we prove inequalities in Eqs. (B13) (B14). First, recall that eigenvalues of a Hermitian matrix majorize its diagonal elements, then, for any $k \in [N]$, we have

$$\sum_{j=1}^{k} \sigma_j^2 \geq \sum_{j=1}^{k} \vec{M}_j^{\downarrow}, \tag{B15}$$

since σ_j^2s are the eigenvalues of $MM^\dagger$.

Second, note that that diagonal elements of $MM^\dagger$, i.e., $\vec{M}_j$, can be expressed in the following form,

$$\vec{M}_j = \sum_{l=1}^{N} |M_{jl}|^2. \tag{B16}$$

Then, from Eq. (B16), we can easily derive an inequality below

$$\sum_{j=1}^{k} \vec{M}_j^{\downarrow} \geq \sum_{l\in S} \vec{M}_l \geq \sum_{j}^{k} |D_j^{\downarrow}|^2, \tag{B17}$$

where the first inequality is due to the rearrangement inequality and S is the index set that includes all the indices of elements appearing in $\sum_{j=1}^{k} |D_j^{\downarrow}|^2$.

Note that the equality in Eq. (B14) holds only when $\vec{M}_j^{\downarrow} = D_j$ which implies that matrix M is diagonal. And if the matrix M is diagonal, then the equality in Eq. (B13) also follows. Overall, the equality in Eq. (B12) holds only for diagonal matrices. ■

b. Loss evaluation We consider the evaluation of o_j in VQFNE, which can be rewritten as

$$o_j = \langle u_j|M|v_j\rangle\langle v_j|M^\dagger|u_j\rangle \tag{B18}$$

$$= \sum_{k_1,k_2} c_{k_1} c_{k_2} \langle u_j|A_{k_1}|v_j\rangle\langle v_j|A_{k_2}^\dagger|u_j\rangle. \tag{B19}$$

In principle, these inner products in Eq. (B19) can be efficiently estimated via Hadamard test and a little classical post-processing. Actually, there are other methods named Hadamard-overlap test for estimating o_j. Hadamard-overlap test was introduced in Ref. [68] to compute a quantity of the form $\langle \mathbf{0}|\mathbf{U}^\dagger \mathbf{A_l} \mathbf{V}|\mathbf{0}\rangle\langle \mathbf{0}|\mathbf{V}^\dagger \mathbf{A}_{\mathbf{l'}}^\dagger \mathbf{U}|\mathbf{0}\rangle$, while in VQFNE, we substitute state $|\mathbf{0}\rangle$ with state $|\psi_j\rangle$, which makes no difference in loss evaluation and gradients derivation. Particularly, instead of estimating each inner product $\langle u_j|A_{k_1}|v_j\rangle$ and $\langle v_j|A_{k_2}^\dagger|u_j\rangle$, the values $\langle u_j|A_{k_1}|v_j\rangle\langle v_j|A_{k_2}^\dagger|u_j\rangle$ can be estimated via Hadamard-overlap test at the expense of doubling the number of qubits.

c. Gradients The gradient of the loss function $F(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is given below,

$$\nabla F(\boldsymbol{\alpha}, \boldsymbol{\beta}) = (\frac{\partial F}{\partial \alpha_1}, ..., \frac{\partial F}{\partial \alpha_{h_1}}, \frac{\partial F}{\partial \beta_1}, ..., \frac{\partial F}{\partial \beta_{h_2}}), \tag{B20}$$

where

$$\frac{\partial F}{\partial \alpha_l} = \sum_j \frac{\partial o_j}{\partial \alpha_l} = \sum_j \sum_{k_1 k_2} c_{k_1} c_{k_2} \frac{\partial R_{j,k_1,k_2}}{\partial \alpha_l} \tag{B21}$$

$$\frac{\partial F}{\partial \beta_t} = \sum_j \frac{\partial o_j}{\partial \beta_t} = \sum_j \sum_{k_1 k_2} c_{k_1} c_{k_2} \frac{\partial R_{j,k_1,k_2}}{\partial \beta_t}. \tag{B22}$$

where $R_{j,k_1,k_2} = \langle u_j|A_{k_1}|v_j\rangle\langle v_j|A_{k_2}^\dagger|u_j\rangle$.

More details on deriving gradients in Eqs. (B21), (B22) can be found in Ref. [68].