npj | Quantum Information

www.nature.com/npjqi

Corrected: Publisher Correction

ARTICLE OPEN

Variational quantum state diagonalization

Ryan LaRose[1,2], Arkin Tikku[1,3], Étude O'Neel-Judy[1], Lukasz Cincio[1] and Patrick J. Coles[1]

Variational hybrid quantum-classical algorithms are promising candidates for near-term implementation on quantum computers. In these algorithms, a quantum computer evaluates the cost of a gate sequence (with speedup over classical cost evaluation), and a classical computer uses this information to adjust the parameters of the gate sequence. Here we present such an algorithm for quantum state diagonalization. State diagonalization has applications in condensed matter physics (e.g., entanglement spectroscopy) as well as in machine learning (e.g., principal component analysis). For a quantum state ρ and gate sequence U, our cost function quantifies how far $U\rho U^\dagger$ is from being diagonal. We introduce short-depth quantum circuits to quantify our cost. Minimizing this cost returns a gate sequence that approximately diagonalizes ρ. One can then read out approximations of the largest eigenvalues, and the associated eigenvectors, of ρ. As a proof-of-principle, we implement our algorithm on Rigetti's quantum computer to diagonalize one-qubit states and on a simulator to find the entanglement spectrum of the Heisenberg model ground state.

npj Quantum Information (2019)5:57 ; https://doi.org/10.1038/s41534-019-0167-6

INTRODUCTION

The future applications of quantum computers, assuming that large-scale, fault-tolerant versions will eventually be realized, are manifold. From a mathematical perspective, applications include number theory,[1] linear algebra,[2–4] differential equations,[5,6] and optimization.[7] From a physical perspective, applications include electronic structure determination[8,9] for molecules and materials and real-time simulation of quantum dynamical processes[10] such as protein folding and photo-excitation events. Naturally, some of these applications are more long-term than others. Factoring and solving linear systems of equations are typically viewed as longer term applications due to their high resource requirements. On the other hand, approximate optimization and the determination of electronic structure may be nearer term applications, and could even serve as demonstrations of quantum supremacy in the near future.[11,12]

A major aspect of quantum algorithms research is to make applications of interest more near term by reducing quantum resource requirements including qubit count, circuit depth, numbers of gates, and numbers of measurements. A powerful strategy for this purpose is algorithm hybridization, where a fully quantum algorithm is turned into a hybrid quantum-classical algorithm.[13] The benefit of hybridization is two-fold, both reducing the resources (hence allowing implementation on smaller hardware) as well as increasing accuracy (by outsourcing calculations to "error-free" classical computers).

Variational hybrid algorithms are a class of quantum-classical algorithms that involve minimizing a cost function that depends on the parameters of a quantum gate sequence. Cost evaluation occurs on the quantum computer, with speedup over classical cost evaluation, and the classical computer uses this cost information to adjust the parameters of the gate sequence. Variational hybrid algorithms have been proposed for Hamiltonian ground state and excited state preparation,[8,14,15] approximate optimization,[7] error correction,[16] quantum data compression,[17,18] quantum simulation,[19,20] and quantum compiling.[21] A key feature of such algorithms is their near-term relevance, since only the subroutine of cost evaluation occurs on the quantum computer, while the optimization procedure is entirely classical, and hence standard classical optimization tools can be employed.

In this work, we consider the application of diagonalizing quantum states. In condensed matter physics, diagonalizing states is useful for identifying properties of topological quantum phases —a field known as entanglement spectroscopy.[22] In data science and machine learning, diagonalizing the covariance matrix (which could be encoded in a quantum state[2,23]) is frequently employed for principal component analysis (PCA). PCA identifies features that capture the largest variance in one's data and hence allows for dimensionality reduction.[24]

Classical methods for diagonalization typically scale polynomially in the matrix dimension.[25] Similarly, the number of measurements required for quantum state tomography—a general method for fully characterizing a quantum state—scales polynomially in the dimension. Interestingly, Lloyd et al. proposed a quantum algorithm for diagonalizing quantum states that can potentially perform exponentially faster than these methods.[2] Namely, their algorithm, called quantum principal component analysis (qPCA), gives an exponential speedup for low-rank matrices. qPCA employs quantum phase estimation combined with density matrix exponentiation. These subroutines require a significant number of qubits and gates, making qPCA difficult to implement in the near term, despite its long-term promise.

Here, we propose a variational hybrid algorithm for quantum state diagonalization. For a given state ρ, our algorithm is composed of three steps: (i) Train the parameters $\boldsymbol{\alpha}$ of a gate sequence $U_p(\boldsymbol{\alpha})$ such that $\tilde{\rho} = U_p(\boldsymbol{\alpha}_{\text{opt}})\rho U_p(\boldsymbol{\alpha}_{\text{opt}})^\dagger$ is approximately

[1]Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA; [2]Department of Computational Mathematics, Science, and Engineering & Department of Physics and Astronomy, Michigan State University, East Lansing, MI 48823, USA and [3]Department of Physics, Blackett Laboratory, Imperial College London, Prince Consort Road, London SW7 2AZ, UK
Correspondence: Lukasz Cincio (lcincio@lanl.gov)

Received: 30 November 2018 Accepted: 29 May 2019
Published online: 26 June 2019

Published in partnership with The University of New South Wales

npj nature partner journals

diagonal, where $\boldsymbol{\alpha}_{\text{opt}}$ is the optimal value of $\boldsymbol{\alpha}$ obtained (ii) Read out the largest eigenvalues of ρ by measuring in the eigenbasis (i.e., by measuring $\tilde{\rho}$ in the standard basis), and (iii) Prepare the eigenvectors associated with the largest eigenvalues. We call this the variational quantum state diagonalization (VQSD) algorithm. VQSD is a near-term algorithm with the same practical benefits as other variational hybrid algorithms. Employing a layered ansatz for $U_p(\boldsymbol{\alpha})$ (where p is the number of layers) allows one to obtain a hierarchy of approximations for the eigevalues and eigenvectors. We therefore think of VQSD as an approximate diagonalization algorithm.

We carefully choose our cost function C to have the following properties: (i) C is faithful (i.e, it vanishes if and only if $\tilde{\rho}$ is diagonal), (ii) C is efficiently computable on a quantum computer, (iii) C has operational meanings such that it upper bounds the eigenvalue and eigenvector error (see Sec. IIA), and (iv) C scales well for training purposes in the sense that its gradient does not vanish exponentially in the number of qubits. The precise definition of C is given in Sec. IIA and involves a difference of purities for different states. To compute C, we introduce short-depth quantum circuits that likely have applications outside the context of VQSD.

To illustrate our method, we implement VQSD on Rigetti's 8-qubit quantum computer. We successfully diagonalize one-qubit pure states using this quantum computer. To highlight future applications (when larger quantum computers are made available), we implement VQSD on a simulator to perform entanglement spectroscopy on the ground state of the one-dimensional (1D) Heisenberg model composed of 12 spins.

Our paper is organized as follows. Section II outlines the VQSD algorithm and presents its implementation. In Sec. III, we give a comparison to the qPCA algorithm, and we elaborate on future applications. Section IV presents our methods for quantifying diagonalization and for optimizing our cost function.

RESULTS

The VQSD algorithm

Overall structure. Figure 1 shows the structure of the VQSD algorithm. The goal of VQSD is to take, as its input, an n-qubit density matrix ρ given as a quantum state and then output approximations of the m-largest eigenvalues and their associated eigenvectors. Here, m will typically be much less than 2^n, the matrix dimension of ρ, although the user is free to increase m with increased algorithmic complexity (discussed below). The outputted eigenvalues will be in classical form, i.e., will be stored on a classical computer. In contrast, the outputted eigenvectors will be in quantum form, i.e., will be prepared on a quantum computer. This is necessary because the eigenvectors would have 2^n entries if they were stored on a classical computer, which is intractable for large n. Nevertheless, one can characterize important aspects of these eigenvectors with a polynomial number of measurements on the quantum computer.

Similar to classical eigensolvers, the VQSD algorithm is an approximate or iterative diagonalization algorithm. Classical eigenvalue algorithms are necessarily iterative, not exact. This can be seen by noting that computing eigenvalues is equivalent to computing roots of a polynomial equation (namely the characteristic polynomial of the matrix) and that no closed-form solution exists for the roots of general polynomials of degree greater than or equal to five.[25] Iterative algorithms are useful in that they allow for a trade-off between run-time and accuracy. Higher degrees of accuracy can be achieved at the cost of more iterations (equivalently, longer run-time), or short run-time can be achieved at the cost of lower accuracy. This flexibility is desirable in that it allows the user of the algorithm to dictate the quality of the solutions found.

The iterative feature of VQSD arises via a layered ansatz for the diagonalizing unitary. This idea similarly appears in other variational hybrid algorithms, such as the Quantum Approximate Optimization Algorithm.[7] Specifically, VQSD diagonalizes ρ by

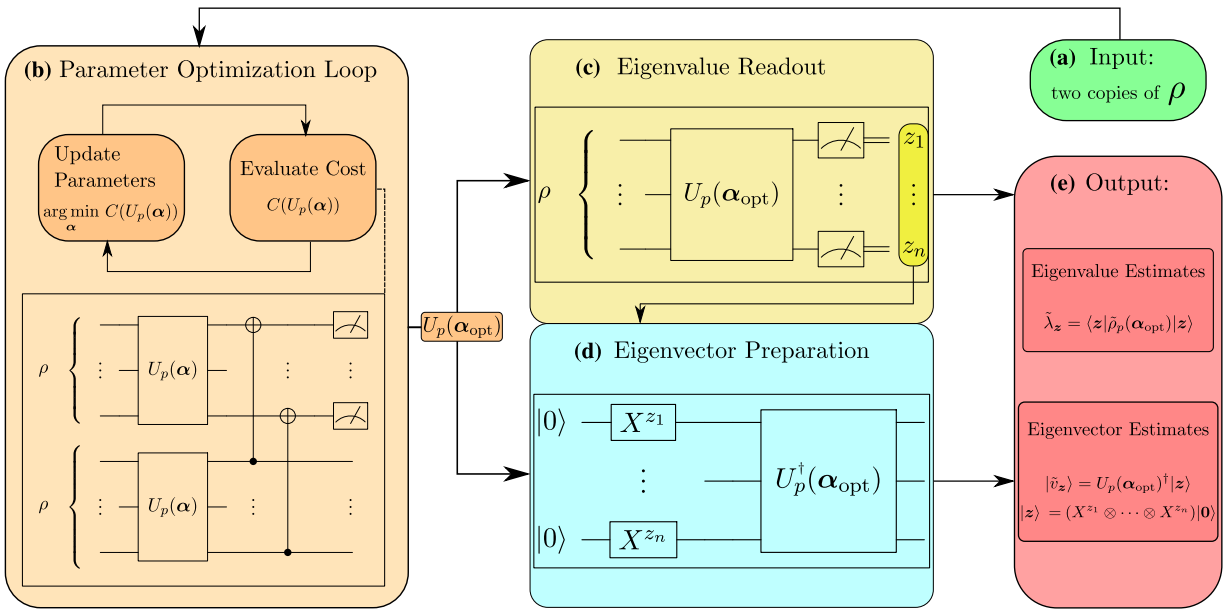

Fig. 1 Schematic diagram showing the steps of the VQSD algorithm. (**a**) Two copies of quantum state ρ are provided as an input. These states are sent to the parameter optimization loop (**b**) where a hybrid quantum-classical variational algorithm approximates the diagonalizing unitary $U_p(\boldsymbol{\alpha}_{\text{opt}})$. Here, p is a hyperparameter that dictates the quality of solution found. This optimal unitary is sent to the eigenvalue readout circuit (**c**) to obtain bitstrings $\boldsymbol{z}$, the frequencies of which provide estimates of the eigenvalues of ρ. Along with the optimal unitary $U_p(\boldsymbol{\alpha}_{\text{opt}})$, these bitstrings are sent to the eigenvector preparation circuit (**c**) to prepare the eigenstates of ρ on a quantum computer. Both the eigenvalues and eigenvectors are the outputs (**d**) of the VQSD algorithm

variationally updating a parameterized unitary $U_p(\boldsymbol{\alpha})$ such that

$$\tilde{\rho}_p(\boldsymbol{\alpha}) := U_p(\boldsymbol{\alpha})\rho U_p^\dagger(\boldsymbol{\alpha}) \quad (1)$$

is (approximately) diagonal at the optimal value $\boldsymbol{\alpha}_{\text{opt}}$. (For brevity we often write $\tilde{\rho}$ for $\tilde{\rho}_p(\boldsymbol{\alpha})$). We assume a layered ansatz of the form

$$U_p(\boldsymbol{\alpha}) = L_1(\boldsymbol{\alpha}_1)L_2(\boldsymbol{\alpha}_2)\cdots L_p(\boldsymbol{\alpha}_p). \quad (2)$$

Here, p is a hyperparameter that sets the number of layers $L_i(\boldsymbol{\alpha}_i)$, and each $\boldsymbol{\alpha}_i$ is a set of optimization parameters that corresponds to internal gate angles within the layer. The parameter $\boldsymbol{\alpha}$ in (1) refers to the collection of all $\boldsymbol{\alpha}_i$ for $i = 1, \ldots, p$. Once the optimization procedure is finished and returns the optimal parameters $\boldsymbol{\alpha}_{\text{opt}}$, one can then run a particular quantum circuit (shown in Fig. 1c and discussed below) N_{readout} times to approximately determine the eigenvalues of ρ. The precision (i.e, the number of significant digits) of each eigenvalue increases with N_{readout} and with the eigenvalue's magnitude. Hence for small N_{readout} only the largest eigenvalues of ρ will be precisely characterized, so there is a connection between N_{readout} and how many eigenvalues, m, are determined. The hyperparameter p is a refinement parameter, meaning that the accuracy of the eigensystem (eigenvalues and eigenvectors) typically increases as p increases. We formalize this argument as follows.

Let C denote our cost function, defined below in (10), which we are trying to minimize. In general, the cost C will be non-increasing (i.e., will either decrease or stay constant) in p. One can ensure that this is true by taking the optimal parameters learned for p layers as the starting point for the optimization of $p+1$ layers and by setting $\boldsymbol{\alpha}_{p+1}$ such that L_{p+1} $(\boldsymbol{\alpha}_{p+1})$ is an identity. This strategy also avoids barren plateaus[26,27] and helps to mitigate the problem of local minima, as we discuss in Appendix B of Supplementary Material (SM).

Next, we argue that C is closely connected to the accuracy of the eigensystem. Specifically, it gives an upper bound on the eigensystem error. Hence, one obtains an increasingly tighter upper bound on the eigensystem error as C decreases (equivalently, as p increases). To quantify eigenvalue error, we define

$$\Delta_\lambda := \sum_{i=1}^{d} (\lambda_i - \tilde{\lambda}_i)^2, \quad (3)$$

where $d = 2^n$, and $\{\lambda_i\}$ and $\{\tilde{\lambda}_i\}$ are the true and inferred eigenvalues, respectively. Here, i is an index that orders the eigenvalues in decreasing order, i.e., $\lambda_i \geq \lambda_{i+1}$ and $\tilde{\lambda}_i \geq \tilde{\lambda}_{i+1}$ for all $i \in \{1, \ldots, d-1\}$. To quantify eigenvector error, we define

$$\Delta_v := \sum_{i=1}^{d} \langle \delta_i | \delta_i \rangle, \quad \text{with } |\delta_i\rangle = \rho|\tilde{v}_i\rangle - \tilde{\lambda}_i|\tilde{v}_i\rangle = \Pi_i^\perp \rho |\tilde{v}_i\rangle. \quad (4)$$

Here, $|\tilde{v}_i\rangle$ is the inferred eigenvector associated with $\tilde{\lambda}_i$, and $\Pi_i^\perp = 1 - |\tilde{v}_i\rangle\langle\tilde{v}_i|$ is the projector onto the subspace orthogonal to $|\tilde{v}_i\rangle$. Hence, $|\delta_i\rangle$ is a vector whose norm quantifies the component of $\rho|\tilde{v}_i\rangle$ that is orthogonal to $|\tilde{v}_i\rangle$, or in other words, how far $|\tilde{v}_i\rangle$ is from being an eigenvector of ρ.

As proven in Sec. IV A, our cost function upper bounds the eigenvalue and eigenvector error up to a proportionality factor β,

$$\Delta_\lambda \leq \beta C, \quad \text{and} \quad \Delta_v \leq \beta C. \quad (5)$$

Because C is non-increasing in p, the upper bound in (5) is non-increasing in p and goes to zero if C goes to zero.

We remark that Δ_v can be interpreted as a weighted eigenvector error, where eigenvectors with larger eigenvalues are weighted more heavily in the sum. This is a useful feature since it implies that lowering the cost C will force the eigenvectors with the largest eigenvalues to be highly accurate. In many applications, such eigenvectors are precisely the ones of interest. (See Sec. II B for an illustration of this feature).

The various steps in the VQSD algorithm are shown schematically in Fig. 1. There are essentially three main steps: (1) an optimization loop that minimizes the cost C via back-and-forth communication between a classical and quantum computer, where the former adjusts $\boldsymbol{\alpha}$ and the latter computes C for $U_p(\boldsymbol{\alpha})$, (2) a readout procedure for approximations of the m largest eigenvalues, which involves running a quantum circuit and then classically analyzing the statistics, and (3) a preparation procedure to prepare approximations of the eigenvectors associated with the m largest eigenvalues. In the following subsections, we elaborate on each of these procedures.

Parameter optimization loop. Naturally, there are many ways to parameterize $U_p(\boldsymbol{\alpha})$. Ideally one would like the number of parameters to grow at most polynomially in both n and p. Figure 2 presents an example ansatz that satisfies this condition. Each layer L_i is broken down into layers of two-body gates that can be performed in parallel. These two-body gates can be further broken down into parameterized one-body gates, for example, with the construction in ref. [28]. We discuss a different approach to parameterize $U_p(\boldsymbol{\alpha})$ in Appendix B of SM.

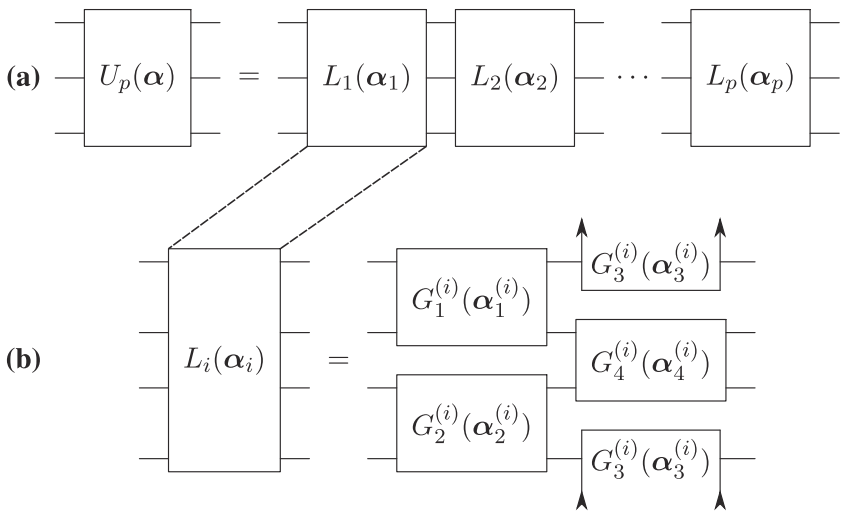

Fig. 2 (**a**) Layered ansatz for the diagonalizing unitary $U_p(\boldsymbol{\alpha})$. Each layer L_i, $i = 1, \ldots, p$, consists of a set of optimization parameters $\boldsymbol{\alpha}_i$. (**b**) The two-qubit gate ansatz for the ith layer, shown on four qubits. Here we impose periodic boundary conditions on the top/bottom edge of the circuit so that G_3 wraps around from top to bottom. Appendix B of SM discusses an alternative approach to the construction of $U_p(\boldsymbol{\alpha})$, in which the ansatz is modified during the optimization process

For a given ansatz, such as the one in Fig. 2, parameter optimization involves evaluating the cost C on a quantum computer for an initial choice of parameters and then modifying the parameters on a classical computer in an iterative feedback loop. The goal is to find

$$\boldsymbol{\alpha}_{\text{opt}} := \arg\min_{\boldsymbol{\alpha}} C(U_p(\boldsymbol{\alpha})). \tag{6}$$

The classical optimization routine used for updating the parameters can involve either gradient-free or gradient-based methods. In Sec. IV B, we explore this further and discuss our optimization methods.

In Eq. (6), $C(U_p(\boldsymbol{\alpha}))$ quantifies how far the state $\tilde{\rho}_p(\boldsymbol{\alpha})$ is from being diagonal. There are many ways to define such a cost function, and in fact there is an entire field of research on coherence measures that has introduced various such quantities.[29] We aim for a cost that is efficiently computable with a quantum-classical system, and hence we consider a cost that can be expressed in terms of purities. (It is well known that a quantum computer can find the purity $\text{Tr}(\sigma^2)$ of an n-qubit state σ with complexity scaling only linearly in n, an exponential speedup over classical computation[30,31]). Two such cost functions, whose individual merits we discuss in Sec. IV A, are

$$C_1(U_p(\boldsymbol{\alpha})) = \text{Tr}(\rho^2) - \text{Tr}(\mathcal{Z}(\tilde{\rho})^2), \tag{7}$$

$$C_2(U_p(\boldsymbol{\alpha})) = \text{Tr}(\rho^2) - \frac{1}{n}\sum_{j=1}^{n} \text{Tr}(\mathcal{Z}_j(\tilde{\rho})^2). \tag{8}$$

Here, $\mathcal{Z}$ and $\mathcal{Z}_j$ are quantum channels that dephase (i.e., destroy the off-diagonal elements) in the global standard basis and in the local standard basis on qubit j, respectively. Importantly, the two functions vanish under the same conditions:

$$C_1(U_p(\boldsymbol{\alpha})) = 0 \Leftrightarrow C_2(U_p(\boldsymbol{\alpha})) = 0 \Leftrightarrow \tilde{\rho} = \mathcal{Z}(\tilde{\rho}). \tag{9}$$

So the global minima of C_1 and C_2 coincide and correspond precisely to unitaries $U_p(\boldsymbol{\alpha})$ that diagonalize ρ (i.e., unitaries such that $\tilde{\rho}$ is diagonal).

As elaborated in Sec. IV A, C_1 has operational meanings: it bounds our eigenvalue error, $C_1 \geq \Delta_\lambda$, and it is equivalent to our eigenvector error, $C_1 = \Delta_v$. However, its landscape tends to be insensitive to changes in $U_p(\boldsymbol{\alpha})$ for large n. In contrast, we are not aware of a direct operational meaning for C_2, aside from its bound on C_1 given by $C_2 \geq (1/n)C_1$. However, the landscape for C_2 is more sensitive to changes in $U_p(\boldsymbol{\alpha})$, making it useful for training $U_p(\boldsymbol{\alpha})$ when n is large. Due to these contrasting merits of C_1 and C_2, we define our overall cost function C as a weighted average of these two functions

$$C(U_p(\boldsymbol{\alpha})) = qC_1(U_p(\boldsymbol{\alpha})) + (1-q)C_2(U_p(\boldsymbol{\alpha})), \tag{10}$$

where $q \in [0, 1]$ is a free parameter that allows one to tailor the VQSD method to the scale of one's problem. For small n, one can set $q \approx 1$ since the landscape for C_1 is not too flat for small n, and, as noted above, C_1 is an operationally relevant quantity. For large n, one can set q to be small since the landscape for C_2 will provide the gradient needed to train $U_p(\boldsymbol{\alpha})$. The overall cost maintains the operational meaning in (5) with

$$\beta = n/(1 + q(n-1)). \tag{11}$$

Appendix B illustrates the advantages of training with different values of q.

Computing C amounts to evaluating the purities of various quantum states on a quantum computer and then doing some simple classical post-processing that scales linearly in n. This can be seen from Eqs. (7) and (8). The first term, $\text{Tr}(\rho^2)$, in C_1 and C_2 is independent of $U_p(\boldsymbol{\alpha})$. Hence, $\text{Tr}(\rho^2)$ can be evaluated outside of the optimization loop in Fig. 1 using the Destructive Swap Test (see Sec. IV A for the circuit diagram). Inside the loop, we only need to compute $\text{Tr}(\mathcal{Z}(\tilde{\rho})^2)$ and $\text{Tr}(\mathcal{Z}_j(\tilde{\rho})^2)$ for all j. Each of these terms are computed by first preparing two copies of $\tilde{\rho}$ and then implementing quantum circuits whose depths are constant in n. For example, the circuit for computing $\text{Tr}(\mathcal{Z}(\tilde{\rho})^2)$ is shown in Fig. 1b, and surprisingly it has a depth of only one gate. We call it the Diagonalized Inner Product (DIP) Test. The circuit for computing $\text{Tr}(\mathcal{Z}_j(\tilde{\rho})^2)$ is similar, and we call it the Partially Diagonalized Inner Product (PDIP) Test. We elaborate on both of these circuits in Sec. IV A.

Eigenvalue readout. After finding the optimal diagonalizing unitary $U_p(\boldsymbol{\alpha}_{\text{opt}})$, one can use it to readout approximations of the eigenvalues of ρ. Figure 1c shows the circuit for this readout. One prepares a single copy of ρ and then acts with $U_p(\boldsymbol{\alpha}_{\text{opt}})$ to prepare $\tilde{\rho}_p(\boldsymbol{\alpha}_{\text{opt}})$. Measuring in the standard basis $\{|\boldsymbol{z}\rangle\}$, where $\boldsymbol{z} = z_1 z_2 \ldots z_n$ is a bitstring of length n, gives a set of probabilities $\{\tilde{\lambda}_{\boldsymbol{z}}\}$ with

$$\tilde{\lambda}_{\boldsymbol{z}} = \langle \boldsymbol{z}|\tilde{\rho}_p(\boldsymbol{\alpha}_{\text{opt}})|\boldsymbol{z}\rangle. \tag{12}$$

We take the $\tilde{\lambda}_{\boldsymbol{z}}$ as the inferred eigenvalues of ρ. We emphasize that the $\tilde{\lambda}_{\boldsymbol{z}}$ are the diagonal elements, not the eigenvalues, of $\tilde{\rho}_p(\boldsymbol{\alpha}_{\text{opt}})$.

Each run of the circuit in Fig. 1c generates a bitstring $\boldsymbol{z}$ corresponding to the measurement outcomes. If one obtains $\boldsymbol{z}$ with frequency $f_{\boldsymbol{z}}$ for N_{readout} total runs, then

$$\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}} = f_{\boldsymbol{z}}/N_{\text{readout}} \tag{13}$$

gives an estimate for $\tilde{\lambda}_{\boldsymbol{z}}$. The statistical deviation of $\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}}$ from $\tilde{\lambda}_{\boldsymbol{z}}$ goes with $1/\sqrt{N_{\text{readout}}}$. The relative error $\epsilon_{\boldsymbol{z}}$ (i.e., the ratio of the statistical error on $\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}}$ to the value of $\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}}$) then goes as

$$\epsilon_{\boldsymbol{z}} = \frac{1}{\sqrt{N_{\text{readout}}}\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}}} = \frac{\sqrt{N_{\text{readout}}}}{f_{\boldsymbol{z}}}. \tag{14}$$

This implies that events $\boldsymbol{z}$ with higher frequency $f_{\boldsymbol{z}}$ have lower relative error. In other words, the larger the inferred eigenvalue $\tilde{\lambda}_{\boldsymbol{z}}$, the lower the relative error, and hence the more precisely it is determined from the experiment. When running VQSD, one can pre-decide on the desired values of N_{readout} and a threshold for the relative error, denoted $\epsilon_{\max}$. This error threshold $\epsilon_{\max}$ will then determine m, i.e., how many of the largest eigenvalues that get precisely characterized. So $m = m(N_{\text{readout}}, \epsilon_{max}, \{\tilde{\lambda}_{\boldsymbol{z}}\})$ is a function of N_{readout}, $\epsilon_{\max}$, and the set of inferred eigenvalues $\{\tilde{\lambda}_{\boldsymbol{z}}\}$. Precisely, we take $m = |\tilde{\lambda}^{\text{est}}|$ as the cardinality of the following set:

$$\tilde{\lambda}^{\text{est}} = \{\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}} : \epsilon_{\boldsymbol{z}} \leq \epsilon_{max}\}, \tag{15}$$

which is the set of inferred eigenvalues that were estimated with the desired precision.

Eigenvector preparation. The final step of VQSD is to prepare the eigenvectors associated with the m-largest eigenvalues, i.e., the eigenvalues in the set in Eq. (15). Let $\boldsymbol{Z} = \{\boldsymbol{z} : \tilde{\lambda}^{\text{est}}_{\boldsymbol{z}} \in \tilde{\lambda}^{\text{est}}\}$ be the set of bitstrings $\boldsymbol{z}$ associated with the eigenvalues in $\tilde{\lambda}^{\text{est}}$. (Note that these bitstrings are obtained directly from the measurement outcomes of the circuit in Fig. 1c, i.e., the outcomes become the bitstring $\boldsymbol{z}$). For each $\boldsymbol{z} \in \boldsymbol{Z}$, one can prepare the following state, which we take as the inferred eigenvector associated with our estimate of the inferred eigenvalue $\tilde{\lambda}^{\text{est}}_{\boldsymbol{z}}$,

$$|\tilde{v}_{\boldsymbol{z}}\rangle = U_p(\boldsymbol{\alpha}_{\text{opt}})^\dagger|\boldsymbol{z}\rangle \tag{16}$$

$$= U_p(\boldsymbol{\alpha}_{\text{opt}})^\dagger (X^{z_1} \otimes \cdots \otimes X^{z_n})|\boldsymbol{0}\rangle. \tag{17}$$

The circuit for preparing this state is shown in Fig. 1d. As noted in (17), one first prepares $|\boldsymbol{z}\rangle$ by acting with X operators raised to the appropriate powers, and then one acts with $U_p(\boldsymbol{\alpha}_{\text{opt}})^\dagger$ to rotate from the standard basis to the inferred eigenbasis.

Once they are prepared on the quantum computer, each inferred eigenvector $|\tilde{v}_{\boldsymbol{z}}\rangle$ can be characterized by measuring

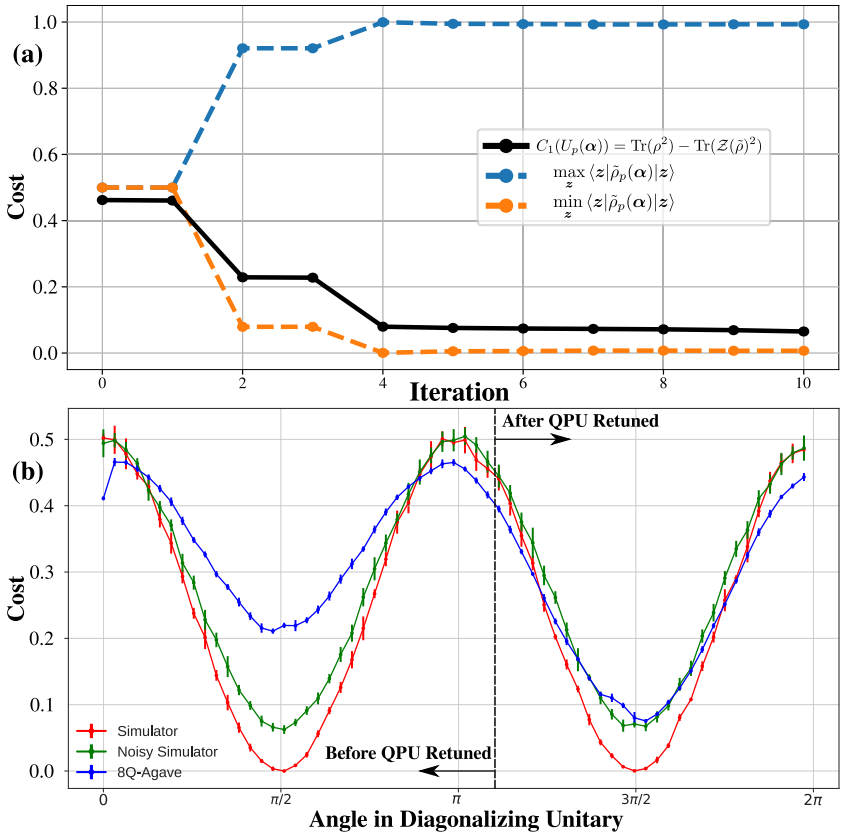

Fig. 3 The VQSD algorithm run on Rigetti's 8Q-Agave quantum computer for $\rho = |+\rangle\langle+|$. (**a**) A representative run of the parameter optimization loop, using the Powell optimization algorithm (see Sec. IV B for details and Appendix B for data from additional runs). Cost versus iteration is shown by the black solid line. The dotted lines show the two inferred eigenvalues. After four iterations, the inferred eigenvalues approach {0, 1}, as required for a pure state. (**b**) The cost landscape on a noiseless simulator, Rigetti's noisy simulator, and Rigetti's quantum computer. Error bars show the standard deviation (due to finite sampling) of multiple runs. The local minima occur roughly at the theoretically predicted values of $\pi/2$ and $3\pi/2$. During data collection for this plot, the 8Q-Agave quantum computer retuned, after which its cost landscape closely matched that of the noisy simulator

expectation values of interest. That is, important physical features such as energy or entanglement (e.g., entanglement witnesses) are associated with some Hermitian observable M, and one can evaluate the expectation value $\langle \tilde{v}_{\boldsymbol{z}}|M|\tilde{v}_{\boldsymbol{z}}\rangle$ to learn about these features.

Implementations

Here we present our implementations of VQSD, first for a one-qubit state on a cloud quantum computer to show that it is amenable to currently available hardware. Then, to illustrate the scaling to larger, more interesting problems, we implement VQSD on a simulator for the 12-spin ground state of the Heisenberg model. See Appendices A and B of SM for further details. The code used to generate some of the examples presented here and in SM can be accessed from ref. [32]

One-qubit state. We now discuss the results of applying VQSD to the one-qubit plus state $\rho = |+\rangle\langle+|$ on the 8Q-Agave quantum computer provided by Rigetti.[33] Because the problem size is small ($n = 1$), we set $q = 1$ in the cost function (10). Since ρ is a pure state, the cost function is

$$C(U_p(\boldsymbol{\alpha})) = C_1(U_p(\boldsymbol{\alpha})) = 1 - \mathrm{Tr}(\mathcal{Z}(\tilde{\rho})^2). \tag{18}$$

For $U_p(\boldsymbol{\alpha})$, we take $p = 1$, for which the layered ansatz becomes an arbitrary single qubit rotation.

The results of VQSD for this state are shown in Fig. 3. In Fig. 3a, the solid curve shows the cost versus the number of iterations in the parameter optimization loop, and the dashed curves show the inferred eigenvalues of ρ at each iteration. Here we used the Powell optimization algorithm, see Section IV B for more details. As can be seen, the cost decreases to a small value near zero and the eigenvalue estimates simultaneously converge to the correct values of zero and one. Hence, VQSD successfully diagonalized this state.

Figure 3b shows the landscape of the optimization problem on Rigetti's 8Q-Agave quantum computer, Rigetti's noisy simulator, and a noiseless simulator. Here, we varied the angle α in the diagonalizing unitary $U(\alpha) = R_x(\pi/2)R_z(\alpha)$ and computed the cost at each value of this angle. The landscape on the quantum computer has local minima near the optimal angles $\alpha = \pi/2$, $3\pi/2$ but the cost is not zero. This explains why we obtain the correct eigenvalues even though the cost is nonzero in Fig. 3a. The nonzero cost can be due to a combination of decoherence, gate infidelity, and measurement error. As shown in Fig. 3b, the 8Q-Agave quantum computer retuned during our data collection, and after this retuning, the landscape of the quantum computer matched that of the noisy simulator significantly better.

Heisenberg model ground state. While current noise levels of quantum hardware limit our implementations of VQSD to small problem sizes, we can explore larger problem sizes on a simulator. An important application of VQSD is to study the entanglement in condensed matter systems, and we highlight this application in

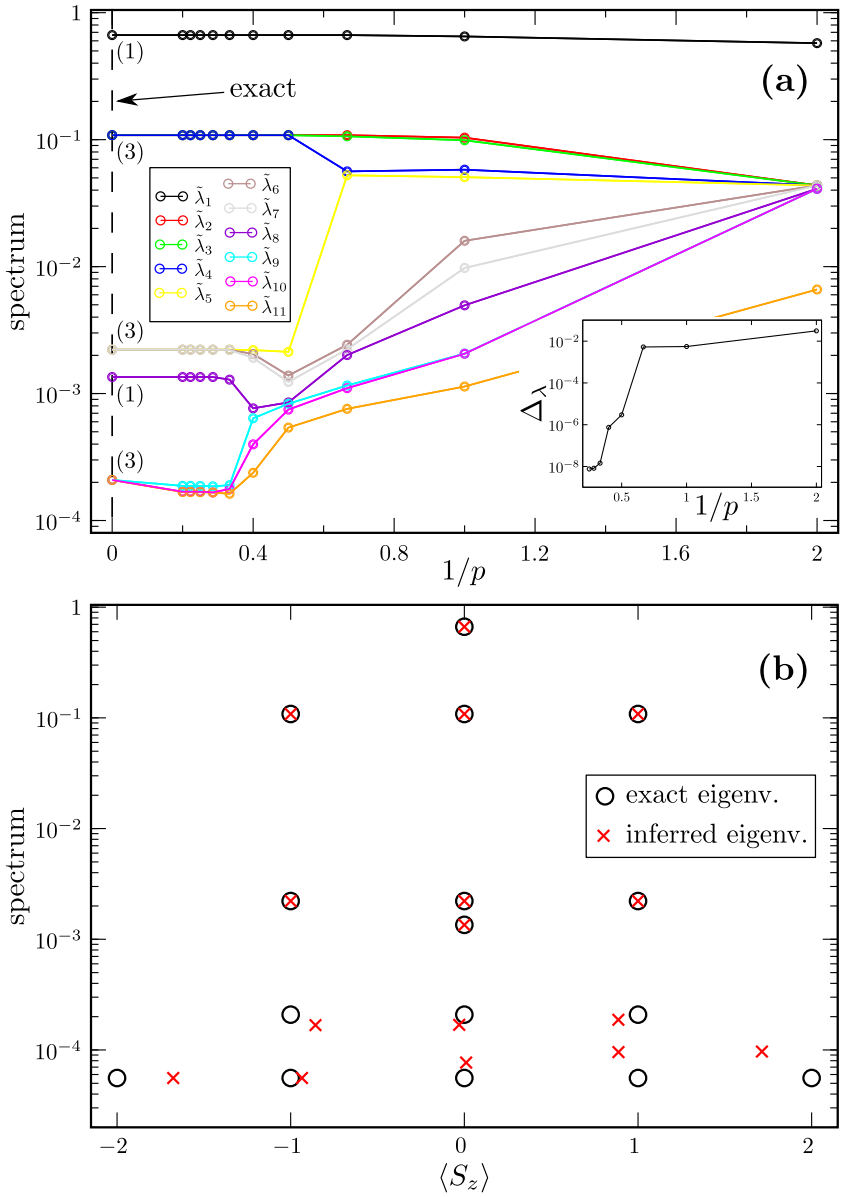

Fig. 4 Implementation of VQSD with a simulator for the ground state of the 1D Heisenberg model, diagonalizing a four-spin subsystem of a chain of eight spins. We chose $q = 1$ for the cost in (10) and employed a gradient-based method to find $\boldsymbol{\alpha}_{\text{opt}}$. (**a**) Largest inferred eigenvalues $\tilde{\lambda}_j$ versus $1/p$, where p is the number of layers in our ansatz, which in this example takes half-integer values corresponding to fractions of layers shown in Fig. 2. The exact eigenvalues are shown on the y-axis (along $1/p = 0$ line) with their degeneracy indicated in parentheses. One can see the largest eigenvalues converge to their correct values, including the correct degeneracies. Inset: overall eigenvalue error Δ_λ versus $1/p$. (**b**) Largest inferred eigenvalues resolved by the inferred $\langle S_z \rangle$ quantum number of their associated eigenvector, for $p = 5$. The inferred data points (red X's) roughly agree with the theoretical values (black circles), particularly for the largest eigenvalues. Appendix B of SM discusses Heisenberg chain of 12 spins

the following example.

Let us consider the ground state of the 1D Heisenberg model, the Hamiltonian of which is

$$H = \sum_{j=1}^{2n} \boldsymbol{S}^{(j)} \cdot \boldsymbol{S}^{(j+1)}, \tag{19}$$

with $\boldsymbol{S}^{(j)} = (1/2)(\sigma_x^{(j)}\hat{x} + \sigma_y^{(j)}\hat{y} + \sigma_z^{(j)}\hat{z})$ and periodic boundary conditions, $\boldsymbol{S}^{(2n+1)} = \boldsymbol{S}^{(1)}$. Performing entanglement spectroscopy on the ground state $|\psi\rangle_{AB}$ involves diagonalizing the reduced state $\rho = \text{Tr}_B(|\psi\rangle\langle\psi|_{AB})$. Here we consider a total of eight spins ($2n = 8$). We take A to be a subset of four nearest-neighbor spins, and B is the complement of A.

The results of applying VQSD to the four-spin reduced state ρ via a simulator are shown in Fig. 4. Panel (a) plots the inferred eigenvalues versus the number of layers p in our ansatz (see Fig. 2). One can see that the inferred eigenvalues converge to their theoretical values as p increases. Panel (b) plots the inferred eigenvalues resolved by their associated quantum numbers (z-component of total spin). This plot illustrates the feature we noted previously that minimizing our cost will first result in minimizing the eigenvector error for those eigenvectors with the largest eigenvalues. Overall our VQSD implementation returned roughly the correct values for both the eigenvalues and their quantum numbers. Resolving not only the eigenvalues but also their quantum numbers is important for entanglement spectroscopy,[22] and clearly VQSD can do this.

In Appendix B of SM we discuss an alternative approach employing a variable ansatz for $U_p(\boldsymbol{\alpha})$, and we present results of applying this approach to a six-qubit reduced state of the 12-qubit ground state of the Heisenberg model.

DISCUSSION

We emphasize that VQSD is meant for states ρ that have either low rank or possibly high rank but low entropy $H(\rho) = -\mathrm{Tr}(\rho \log \rho)$. This is because the eigenvalue readout step of VQSD would be exponentially complex for states with high entropy. In other words, for high entropy states, if one efficiently implemented the eigenvalue readout step (with N_{readout} polynomial in n), then very few eigenvalues would get characterized with the desired precision. In Appendix B of SM we discuss the complexity of VQSD for particular example states.

Examples of states for which VQSD is expected to be efficient include density matrices computed from ground states of 1D, local, gapped Hamiltonians. Also, thermal states of some 1D systems in a many-body localized phase at low enough temperature are expected to be diagonalizable by VQSD. These states have rapidly decaying spectra and are eigendecomposed into states obeying a 1D area law.[34–36] This means that every eigenstate can be prepared by a constant depth circuit in alternating ansatz form,[35] and hence VQSD will be able to diagonalize it.

Comparison to literature

Diagonalizing quantum states with classical methods would require exponentially large memory to store the density matrix, and the matrix operations needed for diagonalization would be exponentially costly. VQSD avoids both of these scaling issues.

Another quantum algorithm that extracts the eigenvalues and eigenvectors of a quantum state is qPCA.[2] Similar to VQSD, qPCA has the potential for exponential speedup over classical diagonalization for particular classes of quantum states. Like VQSD, the speedup in qPCA is contingent on ρ being a low-entropy state.

We performed a simple implementation of qPCA to get a sense for how it compares to VQSD, see Appendix B in SM for details. In particular, just like we did for Fig. 3, we considered the one-qubit plus state $\rho = |+\rangle\langle+|$. We implemented qPCA for this state on Rigetti's noisy simulator (whose noise is meant to mimic that of their 8Q-Agave quantum computer). The circuit that we implemented applied one controlled-exponential-swap gate (in order to approximately exponentiate ρ, as discussed in ref. [2]). We employed a machine-learning approach[37] to compile the controlled-exponential-swap gate into a short-depth gate sequence (see Appendix B in SM). With this circuit we inferred the two eigenvalues of ρ to be approximately 0.8 and 0.2. Hence, for this simple example, it appears that qPCA gave eigenvalues that were slightly off from the true values of 1 and 0, while VQSD was able to obtain the correct eigenvalues, as discussed in Fig. 3.

Future applications

As noted in ref. [2], one application of quantum state diagonalization is benchmarking of quantum noise processes, i.e., quantum process tomography. Here one prepares the Choi state by sending half of a maximally entangled state through the process of interest. One can apply VQSD to the resulting Choi state to learn about the noise process, which may be particular useful for benchmarking near-term quantum computers.

A special case of VQSD is variational state preparation. That is, if one applies VQSD to a pure state $\rho = |\psi\rangle\langle\psi|$, then one can learn the unitary $U(\boldsymbol{\alpha})$ that maps $|\psi\rangle$ to a standard basis state. Inverting this unitary allows one to map a standard basis state (and hence the state $|0\rangle^{\otimes n}$) to the state $|\psi\rangle$, which is known as state preparation. Hence, if one is given $|\psi\rangle$ in quantum form, then VQSD can potentially find a short-depth circuit that approximately prepares $|\psi\rangle$. Variational quantum compiling algorithms that were very recently proposed[21,38] may also be used for this same purpose, and hence it would be interesting to compare VQSD to these algorithms for this special case. Additionally, in this special case one could use VQSD and these other algorithms as an error mitigation tool, i.e., to find a short-depth state preparation that achieves higher accuracy than the original state preparation.

In machine learning, PCA is a subroutine in supervised and unsupervised learning algorithms and also has many direct applications. PCA inputs a data matrix X and finds a new basis such that the variance is maximal along the new basis vectors. One can show that this amounts to finding the eigenvectors of the covariance matrix $E[XX^T]$ with the largest eigenvalues, where E denotes expectation value. Thus PCA involves diagonalizing a positive-semidefinite matrix, $E[XX^T]$. Hence VQSD can perform this task provided one has access to QRAM[23] to prepare the covariance matrix as a quantum state. PCA can reduce the dimension of X as well as filter out noise in data. In addition, nonlinear (kernel) PCA can be used on data that is not linearly separable. Very recent work by Tang[39] suggests that classical algorithms could be improved for PCA of low-rank matrices, and potentially obtain similar scaling as qPCA and VQSD. Hence future work is needed to compare these different approaches to PCA.

Perhaps the most important near-term application of VQSD is to study condensed matter physics. In particular, we propose that one can apply the variational quantum eigensolver[8] to prepare the ground state of a many-body system, and then one can follow this with the VQSD algorithm to characterize the entanglement in this state. Ultimately this approach could elucidate key properties of condensed matter phases. In particular, VQSD allows for entanglement spectroscopy, which has direct application to the identification of topological order.[22] Extracting both the eigenvalues and eigenvectors is useful for entanglement spectroscopy,[22] and we illustrated this capability of VQSD in Fig. 4. Finally, an interesting future research direction is to check how the discrepancies in preparation of multiple copies affect the performance of the diagonalization.

METHODS

Diagonalization test circuits

Here we elaborate on the cost functions C_1 and C_2 and present short-depth quantum circuits to compute them.

C_1 and the DIP test. The function C_1 defined in Eq. (7) has several intuitive interpretations. These interpretations make it clear that C_1 quantifies how far a state is from being diagonal. In particular, let $D_{\mathrm{HS}}(A,B) := \mathrm{Tr}\left((A-B)^\dagger(A-B)\right)$ denote the Hilbert–Schmidt distance. Then we can write

$$C_1 = \min_{\sigma\in\mathcal{D}} D_{\mathrm{HS}}(\tilde{\rho},\sigma) \tag{20}$$

$$= D_{\mathrm{HS}}(\tilde{\rho},\mathcal{Z}(\tilde{\rho})) \tag{21}$$

$$= \sum_{\boldsymbol{z},\,\boldsymbol{z}'\neq\boldsymbol{z}} |\langle\boldsymbol{z}|\tilde{\rho}|\boldsymbol{z}'\rangle|^2 . \tag{22}$$

In other words, C_1 is (1) the minimum distance between $\tilde{\rho}$ and the set of diagonal states $\mathcal{D}$, (2) the distance from $\tilde{\rho}$ to $\mathcal{Z}(\tilde{\rho})$, and (3) the sum of the absolute squares of the off-diagonal elements of $\tilde{\rho}$.

C_1 can also be written as the eigenvector error in Eq. (4) as follows. For an inferred eigenvector $|\tilde{v}_{\boldsymbol{z}}\rangle$, we define $|\delta_{\boldsymbol{z}}\rangle = \rho|\tilde{v}_{\boldsymbol{z}}\rangle - \tilde{\lambda}_{\boldsymbol{z}}|\tilde{v}_{\boldsymbol{z}}\rangle$ and write the

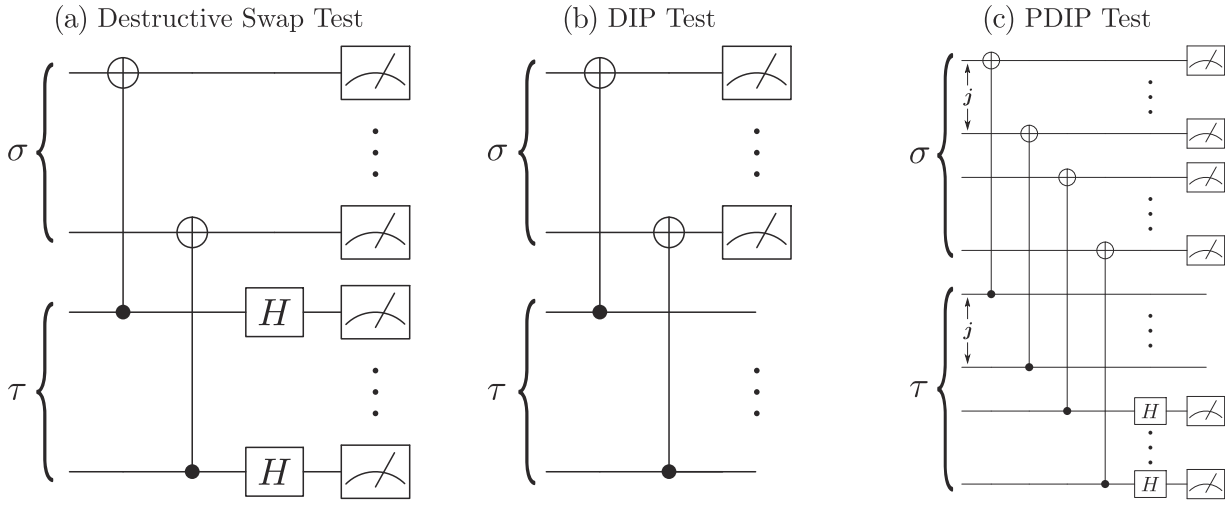

Fig. 5 Diagonalization test circuits used in VQSD. (**a**) The Destructive Swap Test computes $\mathrm{Tr}(\sigma\tau)$ via a depth-two circuit. (**b**) The Diagonalized Inner Product (DIP) Test computes $\mathrm{Tr}(\mathcal{Z}(\sigma)\mathcal{Z}(\tau))$ via a depth-one circuit. (**c**) The Partially Diagonalized Inner Product (PDIP) Test computes $\mathrm{Tr}(\mathcal{Z}_{\boldsymbol{j}}(\sigma)\mathcal{Z}_{\boldsymbol{j}}(\tau))$ via a depth-two circuit, for a particular set of qubits $\boldsymbol{j}$. While the DIP test requires no postprocessing, the postprocessing for the Destructive Swap Test and the Partial DIP Test scales linearly in n

eigenvector error as

$$\langle\delta_{\boldsymbol{z}}|\delta_{\boldsymbol{z}}\rangle = \langle\tilde{v}_{\boldsymbol{z}}|\rho^2|\tilde{v}_{\boldsymbol{z}}\rangle + \tilde{\lambda}_{\boldsymbol{z}}^2 - 2\tilde{\lambda}_{\boldsymbol{z}}\langle\tilde{v}_{\boldsymbol{z}}|\rho|\tilde{v}_{\boldsymbol{z}}\rangle \tag{23}$$

$$= \langle\tilde{v}_{\boldsymbol{z}}|\rho^2|\tilde{v}_{\boldsymbol{z}}\rangle - \tilde{\lambda}_{\boldsymbol{z}}^2, \tag{24}$$

since $\langle\tilde{v}_{\boldsymbol{z}}|\rho|\tilde{v}_{\boldsymbol{z}}\rangle = \tilde{\lambda}_{\boldsymbol{z}}$. Summing over all $\boldsymbol{z}$ gives

$$\Delta_v = \sum_{\boldsymbol{z}}\langle\delta_{\boldsymbol{z}}|\delta_{\boldsymbol{z}}\rangle = \sum_{\boldsymbol{z}}\langle\tilde{v}_{\boldsymbol{z}}|\rho^2|\tilde{v}_{\boldsymbol{z}}\rangle - \tilde{\lambda}_{\boldsymbol{z}}^2 \tag{25}$$

$$= \mathrm{Tr}(\rho^2) - \mathrm{Tr}(\mathcal{Z}(\tilde{\rho})^2) = C_1, \tag{26}$$

which proves the bound in Eq. (5) for $q = 1$.

In addition, C_1 bounds the eigenvalue error defined in Eq. (3). Let $\tilde{\lambda} = (\tilde{\lambda}_1, \ldots, \tilde{\lambda}_d)$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_d)$ denote the inferred and actual eigenvalues of ρ, respectively, both arranged in decreasing order. In this notation we have

$$\Delta_\lambda = \boldsymbol{\lambda}\cdot\boldsymbol{\lambda} + \tilde{\lambda}\cdot\tilde{\lambda} - 2\boldsymbol{\lambda}\cdot\tilde{\lambda} \tag{27}$$

$$C_1 = \boldsymbol{\lambda}\cdot\boldsymbol{\lambda} - \tilde{\lambda}\cdot\tilde{\lambda} \tag{28}$$

$$= \Delta_\lambda + 2(\boldsymbol{\lambda}\cdot\tilde{\lambda} - \tilde{\lambda}\cdot\tilde{\lambda}). \tag{29}$$

Since the eigenvalues of a density matrix majorize its diagonal elements, $\boldsymbol{\lambda} \succ \tilde{\lambda}$, and the dot product with an ordered vector is a Schur convex function, we have

$$\boldsymbol{\lambda}\cdot\tilde{\lambda} \geq \tilde{\lambda}\cdot\tilde{\lambda}. \tag{30}$$

Hence from Eq. (29) and Eq. (30) we obtain the bound

$$\Delta_\lambda \leq C_1, \tag{31}$$

which corresponds to the bound in Eq. (5) for the special case of $q = 1$.

For computational purposes, we use the difference of purities interpretation of C_1 given in Eq. (7). The $\mathrm{Tr}(\rho^2)$ term is independent of $U_p(\boldsymbol{\alpha})$. Hence it only needs to be evaluated once, outside of the parameter optimization loop. It can be computed via the expectation value of the swap operator S on two copies of ρ, using the identity

$$\mathrm{Tr}(\rho^2) = \mathrm{Tr}((\rho\otimes\rho)S). \tag{32}$$

This expectation value is found with a depth-two quantum circuit that essentially corresponds to a Bell-basis measurement, with classical post-processing that scales linearly in the number of qubits.[37,40] This is shown in Fig. 5a. We call this procedure the Destructive Swap Test, since it is like the Swap Test, but the measurement occurs on the original systems instead of on an ancilla.

Similarly, the $\mathrm{Tr}(\mathcal{Z}(\tilde{\rho})^2)$ term could be evaluated by first dephasing $\tilde{\rho}$ and then performing the Destructive Swap Test, which would involve a depth-three quantum circuit with linear classical post-processing. This approach was noted in ref. [41]. However, there exists a simpler circuit, which we call the Diagonalized Inner Product (DIP) Test. The DIP Test involves a depth-one quantum circuit with no classical post-processing. An abstract version of this circuit is shown in Fig. 5b, for two states σ and τ. The proof that this circuit computes $\mathrm{Tr}(\mathcal{Z}(\sigma)\mathcal{Z}(\tau))$ is given in Appendix B of SM. For our application we will set $\sigma = \tau = \tilde{\rho}$, for which this circuit gives $\mathrm{Tr}(\mathcal{Z}(\tilde{\rho})^2)$.

In summary, C_1 is efficiently computed by using the Destructive Swap Test for the $\mathrm{Tr}(\rho^2)$ term and the DIP Test for the $\mathrm{Tr}(\mathcal{Z}(\tilde{\rho})^2)$ term.

C_2 and the PDIP test. Like C_1, C_2 can also be rewritten in terms of of the Hilbert–Schmidt distance. Namely, C_2 is the average distance of $\tilde{\rho}$ to each locally dephased state $\mathcal{Z}_j(\tilde{\rho})$:

$$C_2 = \frac{1}{n}\sum_{j=1}^{n} D_{\mathrm{HS}}(\tilde{\rho}, \mathcal{Z}_j(\tilde{\rho})). \tag{33}$$

where $\mathcal{Z}_j(\cdot) = \sum_z (|z\rangle\langle z|_j \otimes 1_{k\neq j})(\cdot)(|z\rangle\langle z|_j \otimes 1_{k\neq j})$. Naturally, one would expect that $C_2 \leq C_1$, since $\tilde{\rho}$ should be closer to each locally dephased state than to the fully dephased state. Indeed this is true and can be seen from:

$$C_2 = C_1 - \frac{1}{n}\sum_{j=1}^{n}\min_{\sigma\in D} D_{HS}(\mathcal{Z}_j(\tilde{\rho}), \sigma) \tag{34}$$

However, C_1 and C_2 vanish under precisely the same conditions, as noted in Eq. (9). One can see this by noting that C_2 also upper bounds $(1/n)C_1$ and hence we have

$$C_2 \leq C_1 \leq nC_2. \tag{35}$$

Combining the upper bound in Eq. (35) with the relations in Eq. (26) and Eq. (31) gives the bounds in Eq. (5) with β defined in Eq. (11). The upper bound in Eq. (35) is proved as follows. Let $\boldsymbol{z} = z_1 \ldots z_n$ and $\boldsymbol{z}' = z_1' \ldots z_n'$ be n-dimensional bitstrings. Let $\mathcal{S}$ be the set of all pairs $(\boldsymbol{z}, \boldsymbol{z}')$ such that $\boldsymbol{z} \neq \boldsymbol{z}'$, and let $\mathcal{S}_j$ be the set of all pairs $(\boldsymbol{z}, \boldsymbol{z}')$ such that $z_j \neq z_j'$. Then we have $C_1 = \sum_{(\boldsymbol{z},\boldsymbol{z}')\in\mathcal{S}} |\langle\boldsymbol{z}|\tilde{\rho}|\boldsymbol{z}'\rangle|^2$, and

$$nC_2 = \sum_{j=1}^{n}\sum_{(\boldsymbol{z},\boldsymbol{z}')\in\mathcal{S}_j} |\langle\boldsymbol{z}|\tilde{\rho}|\boldsymbol{z}'\rangle|^2 \tag{36}$$

$$\geq \sum_{(\boldsymbol{z},\boldsymbol{z}')\in\mathcal{S}^U} |\langle\boldsymbol{z}|\tilde{\rho}|\boldsymbol{z}'\rangle|^2 = C_1, \tag{37}$$

where $\mathcal{S}^U = \bigcup_{j=1}^{n}\mathcal{S}_j$ is the union of all the $\mathcal{S}_j$ sets. The inequality in Eq. (37) arises from the fact that the $\mathcal{S}_j$ sets have non-trivial intersection with each other, and hence we throw some terms away when only considering the union $\mathcal{S}^U$. The last equality follows from the fact that $\mathcal{S}^U = \mathcal{S}$, i.e, the set of

all bitstring pairs that differ from each other ($\mathcal{S}$) corresponds to the set of all bitstring pairs that differ for at least one element ($\mathcal{S}^U$).

Writing C_2 in terms of purities, as in Eq. (8), shows how it can be computed on a quantum computer. As in the case of C_1, the first term in Eq. (8) is computed with the Destructive Swap Test. For the second term in Eq. (8), each purity $\mathrm{Tr}(\mathcal{Z}_j(\tilde{\rho})^2)$ could also be evaluated with the Destructive Swap Test, by first locally dephasing the appropriate qubit. However, we present a slightly improved circuit to compute these purities that we call the PDIP Test. The PDIP Test is shown in Fig. 5c for the general case of feeding in two distinct states σ and τ with the goal of computing the inner product between $\mathcal{Z}_j(\sigma)$ and $\mathcal{Z}_j(\tau)$. For generality we let l, with $0 \leq l \leq n$, denote the number of qubits being locally dephased for this computation. If $l > 0$, we define $\boldsymbol{j} = (j_1, \ldots, j_l)$ as a vector of indices that indicates which qubits are being locally dephased. The PDIP Test is a hybrid of the Destructive Swap Test and the DIP Test, corresponding to the former when $l = 0$ and the latter when $l = n$. Hence, it generalizes both the Destructive Swap Test and the DIP Test. Namely, the PDIP Test performs the DIP Test on the qubits appearing in $\boldsymbol{j}$ and performs the Destructive Swap Test on the qubits not appearing in $\boldsymbol{j}$. The proof that the PDIP Test computes $\mathrm{Tr}(\mathcal{Z}_j(\sigma)\mathcal{Z}_j(\tau))$, and hence $\mathrm{Tr}(\mathcal{Z}_j(\tilde{\rho})^2)$ when $\sigma = \tau = \tilde{\rho}$, is given in Appendix B of SM.

C_1 versus C_2. Here we discuss the contrasting merits of the functions C_1 and C_2, hence motivating our cost definition in Eq. (10).

As noted previously, C_2 does not have an operational meaning like C_1. In addition, the circuit for computing C_1 is more efficient than that for C_2. The circuit in Fig. 5b for computing the second term in C_1 has a gate depth of one, with n CNOT gates, n measurements, and no classical post-processing. The circuit in Fig. 5c for computing the second term in C_2 has a gate depth of two, with n CNOT gates, $n-1$ Hadamard gates, $2n-1$ measurements, and classical post-processing whose complexity scales linearly in n. So in every aspect, the circuit for computing C_1 is less complex than that for C_2. This implies that C_1 can be computed with greater accuracy than C_2 on a noisy quantum computer.

On the other hand, consider how the landscape for C_1 and C_2 scale with n. As a simple example, suppose $\rho = |0\rangle\langle 0| \otimes \cdots \otimes |0\rangle\langle 0|$. Suppose one takes a single parameter ansatz for U, such that $U(\theta) = R_X(\theta) \otimes \cdots \otimes R_X(\theta)$, where $R_X(\theta)$ is a rotation about the X-axis of the Bloch sphere by angle θ. For this example,

$$C_1(\theta) = 1 - \mathrm{Tr}(\mathcal{Z}(\tilde{\rho})^2) = 1 - x(\theta)^n \tag{38}$$

where $x(\theta) = \mathrm{Tr}(\mathcal{Z}(R_X(\theta)|0\rangle\langle 0|R_X(\theta)^\dagger)^2) = (1 + \cos^2\theta)/2$. If θ is not an integer multiple of π, then $x(\theta) < 1$, and $x(\theta)^n$ will be exponentially suppressed for large n. In other words, for large n, the landscape for $x(\theta)^n$ becomes similar to that of a delta function: it is zero for all θ except for multiples of π. Hence, for large n, it becomes difficult to train the unitary $U(\theta)$ because the gradient vanishes for most θ. This is just an illustrative example, but this issue is general. Generally speaking, for large n, the function C_1 has a sharp gradient near its global minima, and the gradient vanishes when one is far away from these minima. Ultimately this limits C_1's utility as a training function for large n.

In contrast, C_2 does not suffer from this issue. For the example in the previous paragraph,

$$C_2(\theta) = 1 - x(\theta), \tag{39}$$

which is independent of n. So for this example the gradient of C_2 does not vanish as n increases, and hence C_2 can be used to train θ. More generally, the landscape of C_2 is less barren than that of C_1 for large n. We can argue this, particularly, for states ρ that have low rank or low entropy. The second term in Eq. (8), which is the term that provides the variability with $\boldsymbol{\alpha}$, does not vanish even for large n, since (as shown in Appendix B of SM):

$$\mathrm{Tr}(\mathcal{Z}_j(\tilde{\rho})^2) \geq 2^{-H(\rho)-1} \geq \frac{1}{2r}. \tag{40}$$

Here, $H(\rho) = -\mathrm{Tr}(\rho \log_2 \rho)$ is the von Neumann entropy, and r is the rank of ρ. So as long as ρ is low entropy or low rank, then the second term in C_2 will not vanish. Note that a similar bound does not exist for second term in C_1, which does tend to vanish for large n.

Optimization methods

Finding $\boldsymbol{\alpha}_{\mathrm{opt}}$ in Eq. (6) is a major component of VQSD. While many works have benchmarked classical optimization algorithms (e.g., ref. [42]), the particular case of optimization for variational hybrid algorithms[43] is limited and needs further work.[44] Both gradient-based and gradient-free methods are possible, but gradient-based methods may not work as well with noisy data. Additionally, ref. [26] notes that gradients of a large class of circuit ansatze vanish when the number of parameters becomes large. These and other issues (e.g., sensitivity to initial conditions, number of function evaluations) should be considered when choosing an optimization method.

In our preliminary numerical analyses (see Appendix B in SM), we found that the Powell optimization algorithm[45] performed the best on both quantum computer and simulator implementations of VQSD. This derivative-free algorithm uses a bi-directional search along each parameter using Brent's method. Our studies showed that Powell's method performed the best in terms of convergence, sensitivity to initial conditions, and number of correct solutions found. The implementation of Powell's algorithm used in this paper can be found in the open-source Python package SciPy Optimize.[46] Finally, Appendix B of SM shows how our layered ansatz for $U_p(\boldsymbol{\alpha})$ as well as proper initialization of $U_p(\boldsymbol{\alpha})$ helps in mitigating the problem of local minima.

DATA AVAILABILITY

Data generated and analyzed during current study are available from the corresponding author upon reasonable request.

CODE AVAILABILITY

The code used to generate some of the examples presented here and in Supplementary Material can be accessed from ref. [32]

ACKNOWLEDGEMENTS

We thank Rigetti for providing access to their quantum computer. The views expressed in this paper are those of the authors and do not reflect those of Rigetti. R. L., E.O.N-J, and A.T. acknowledge support from the U.S. Department of Energy through a quantum computing program sponsored by the LANL Information Science & Technology Institute. R.L. acknowledges support from an Engineering Distinguished Fellowship through Michigan State University. L.C. was supported by the U.S. Department of Energy through the J. Robert Oppenheimer fellowship. P.J.C. was supported by the LANL ASC Beyond Moore's Law project. LC and P.J.C. were also supported by the LDRD program at Los Alamos National Laboratory, by the U.S. Department of Energy (DOE), Office of Science, Office of Advanced Scientific Computing Research, and also by the U.S. DOE, Office of Science, Basic Energy Sciences, Materials Sciences and Engineering Division, Condensed Matter Theory Program.

AUTHOR CONTRIBUTIONS

R.L., A.T., and L.C. implemented the algorithms and performed numerical analysis. L.C. and P.J.C. designed the project. P.J.C. proposed the cost function and proved the analytical results. R.L., A.T., É.O.N.-J., L.C., and P.J.C. contributed to data analysis, as well as writing and editing the final paper.

ADDITIONAL INFORMATION

Supplementary Information accompanies the paper on the *npj Quantum Information* website (https://doi.org/10.1038/s41534-019-0167-6).

Competing interests: The authors declare no competing interests.

REFERENCES

1. Shor, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **41**, 303–332 (1999).
2. Lloyd, S., Mohseni, M. & Rebentrost, P. Quantum principal component analysis. *Nat. Phys.* **10**, 631–633 (2014).
3. Harrow, A. W., Hassidim, A. & Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **103**, 150502 (2009).
4. Rebentrost, P., Steffens, A., Marvian, I. & Lloyd, S. Quantum singular-value decomposition of nonsparse low-rank matrices. *Phys. Rev. A* **97**, 012327 (2018).
5. Leyton, S. K. & Osborne, T. J. A quantum algorithm to solve nonlinear differential equations. *arXiv:0812.4423*, https://arxiv.org/abs/0812.4423 (2008).

6. Berry, D. W. High-order quantum algorithm for solving linear differential equations. *J. Phys. A* **47**, 105301 (2014).
7. Farhi, E., Goldstone, J. & Gutmann, S. A quantum approximate optimization algorithm. *arXiv:1411.4028*, https://arxiv.org/abs/1411.402 (2014).
8. Peruzzo, A. et al. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **5**, 4213 (2014).
9. Kandala, A. et al. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* **549**, 242 (2017).
10. Berry, D. W., Childs, A. M., Cleve, R., Kothari, R. & Somma, R. D. Simulating Hamiltonian dynamics with a truncated taylor series. *Phys. Rev. Lett.* **114**, 090502 (2015).
11. Preskill, J. Quantum computing and the entanglement frontier. *arXiv:1203.5813*, https://arxiv.org/abs/1203.5813 (2012).
12. Harrow, A. W. & Montanaro, A. Quantum computational supremacy. *Nature* **549**, 203 (2017).
13. Bravyi, S., Smith, G. & Smolin, J. A. Trading classical and quantum computational resources. *Phys. Rev. X* **6**, 021043 (2016).
14. Higgott, O., Wang, D. & Brierley, S. Variational quantum computation of excited states. *arXiv:1805.08138*, https://arxiv.org/abs/1805.08138 (2018).
15. Endo, S., Jones, T., McArdle, S., Yuan, X. & Benjamin, S. Variational quantum algorithms for discovering Hamiltonian spectra. *arXiv:1806.05707*, https://arxiv.org/abs/1806.05707 (2018).
16. Johnson, P. D., Romero, J., Olson, J., Cao, Y. & Aspuru-Guzik, A. QVECTOR: an algorithm for device-tailored quantum error correction. *arXiv:1711.02249*, https://arxiv.org/abs/1711.02249 (2017).
17. Romero, J., Olson, J. P. & Aspuru-Guzik, A. Quantum autoencoders for efficient compression of quantum data. *Quantum Sci. Technol.* **2**, 045001 (2017).
18. Khoshaman, A., Vinci, W., Denis, B., Andriyash, E. & Amin, M. H. Quantum variational autoencoder. *Quantum Sci. Technol.* **4**, 014001 (2018).
19. Li, Y. & Benjamin, S. C. Efficient variational quantum simulator incorporating active error minimization. *Phys. Rev. X* **7**, 021050 (2017).
20. Kokail, C. et al. Self-verifying variational quantum simulation of the lattice schwinger model. *Nature* **569**, 355 (2019).
21. Khatri, S. et al. Quantum-assisted quantum compiling. *Quantum* **3**, 140 (2019).
22. Li, H. & Haldane, F. D. M. Entanglement spectrum as a generalization of entanglement entropy: identification of topological order in non-abelian fractional quantum hall effect states. *Phys. Rev. Lett.* **101**, 010504 (2008).
23. Giovannetti, V., Lloyd, S. & Maccone, L. Quantum random access memory. *Phys. Rev. Lett.* **100**, 160501 (2008).
24. Pearson, K. On lines and planes of closest fit to systems of points in space. *Lond., Edinb., Dublin Philos. Mag. J. Sci.* **2**, 559–572 (1901).
25. Trefethen, L. N. & Bau, D. *Numerical Linear Algebra* (SIAM, Philadelphia, PA, 1997).
26. McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R. & Neven, H. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.* **9**, 4812 (2018).
27. Grant, E., Wossnig, L., Ostaszewski, M. & Benedetti, M. An initialization strategy for addressing barren plateaus in parametrized quantum circuits. *arXiv:1903.05076*, https://arxiv.org/abs/1903.05076 (2019).
28. Vatan, F. & Williams, C. Optimal quantum circuits for general two-qubit gates. *Phys. Rev. A* **69**, 032315 (2004).
29. Baumgratz, T., Cramer, M. & Plenio, M. B. Quantifying coherence. *Phys. Rev. Lett.* **113**, 140401 (2014).
30. Buhrman, H., Cleve, R., Watrous, J. & De Wolf, R. Quantum fingerprinting. *Phys. Rev. Lett.* **87**, 167902 (2001).
31. Gottesman, D. & Chuang, I. Quantum digital signatures. *quant-ph/0105032*, https://arxiv.org/abs/quant-ph/0105032 (2001).
32. VQSD source code. https://github.com/rmlarose/vqsd.
33. Smith, R. S., Curtis, M. J. & Zeng, W. J. A Practical Quantum Instruction Set Architecture. *arXiv:1608.03355*, https://arxiv.org/abs/1608.03355 (2016).
34. Hastings, M. B. An area law for one-dimensional quantum systems. *J. Stat. Mech.: Theory Exp.* **2007**, 08024 (2007).
35. Bauer, B. & Nayak, C. Area laws in a many-body localized state and its implications for topological order. *J. Stat. Mech.: Theory Exp.* **2013**, 09005 (2013).
36. Grover, T. Certain general constraints on the many-body localization transition. *arXiv:1405.1471*, https://arxiv.org/abs/1405.1471 (2014).
37. Cincio, L., Subas, Y., Sornborger, A. T. & Coles, P. J. Learning the quantum algorithm for state overlap. *New J. Phys.* **20**, 113022 (2018).
38. Jones, T. & Benjamin, S. C. Quantum compilation and circuit optimisation via energy dissipation. *arXiv:1811.03147*, https://arxiv.org/abs/1811.03147 (2018).
39. Tang, E. Quantum-inspired classical algorithms for principal component analysis and supervised clustering. *arXiv:1811.00414*, https://arxiv.org/abs/1811.00414 (2018).
40. Garcia-Escartin, J. C. & Chamorro-Posada, P. Swap test and Hong-Ou-Mandel effect are equivalent. *Phys. Rev. A* **87**, 052330 (2013).
41. Smith, G. et al. Quantifying coherence and entanglement via simple measurements. *arXiv:1707.09928*, https://arxiv.org/abs/1707.09928 (2017).
42. Rios, L. M. & Sahinidis, N. V. Derivative-free optimization: a review of algorithms and comparison of software implementations. *J. Glob. Optim.* **56**, 1247–1293 (2013).
43. Guerreschi, G. G. & Smelyanskiy, M. Practical optimization for hybrid quantum-classical algorithms. *arXiv:1701.01450*, https://arxiv.org/abs/1701.01450 (2017).
44. McClean, J. R., Romero, J., Babbush, R. & Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.* **18**, 023023 (2016).
45. Powell, M. J. D. A fast algorithm for nonlinearly constrained optimization calculations. In *Numerical Analysis*, Lecture Notes in Mathematics (ed. Watson, G. A.) 144–157 (Springer, Berlin, 1978).
46. Scipy optimization and root finding. https://docs.scipy.org/doc/scipy/reference/optimize.html (2018).