Quantum autoencoders for efficient compression of quantum data

Jonathan Romero,[1] Jonathan P. Olson,[1] and Alan Aspuru-Guzik[1,*]

[1]*Department of Chemistry and Chemical Biology, Harvard University, Cambridge, Massachusetts 02138, United States*

(Dated: February 14, 2017)

Classical autoencoders are neural networks that can learn efficient low dimensional representations of data in higher dimensional space. The task of an autoencoder is, given an input x, is to map x to a lower dimensional point y such that x can likely be recovered from y. The structure of the underlying autoencoder network can be chosen to represent the data on a smaller dimension, effectively compressing the input. Inspired by this idea, we introduce the model of a quantum autoencoder to perform similar tasks on quantum data. The quantum autoencoder is trained to compress a particular dataset of quantum states, where a classical compression algorithm cannot be employed. The parameters of the quantum autoencoder are trained using classical optimization algorithms. We show an example of a simple programmable circuit that can be trained as an efficient autoencoder. We apply our model in the context of quantum simulation to compress ground states of the Hubbard model and molecular Hamiltonians.

I. INTRODUCTION

Quantum technologies, ranging from quantum computing to quantum cryptography, have been found to have increasingly powerful applications for a modern society. Quantum simulators for chemistry, for example, have been recently shown to be capable of efficiently calculating molecular energies for small systems [1]; the capability for larger scale simulations promises to have deep implications for materials design, pharmacological research, and an array of other potentially life-changing functions [2]. A limiting factor for nearly all of these applications, however, is the amount of quantum resources that can be realized in an experiment. Therefore, for experiments now and in the near future, any tool which can reduce the experimental overhead in terms of these resources is especially valuable.

For classical data processing, machine learning via an autoencoder is one such tool for dimensional reduction [3–5], as well as having application in generative data models [6]. A classical autoencoder is a function whose parameters are optimized across a training set of data which, given an $(n+k)$-bit input string x, attempts to reproduce x. Part of the specification of the circuit, however, is to erase some k bits during the process. If an autoencoder is successfully trained to reproduce x at the output at least approximately, then the remaining n bits immediately after the erasure (referred to as the *latent space*) represent a compressed encoding of the string x. Thus, the circuit "learns" to encode information that is close to the training set.

In this paper, we introduce the concept of a quantum autoencoder which is inspired by this design for an input of $n+k$ qubits. Because quantum mechanics is able to generate patterns with properties (e.g. superposition and entanglement) that is beyond classical physics, a quantum computer should also be able to *recognize* patterns that are beyond the capabilities of classical machine learning. Thus, the motivation for a *quantum* autoencoder is simple; such a model allows us to perform analogous machine learning tasks for quantum systems without exponentially costly classical memory, for instance, in dimension reduction of quantum data. A related work proposing a quantum autoencoder model establishes a formal connection between classical and quantum feedforward neural networks where a particular setting of parameters in the quantum network reduces to a classical neural network exactly [7]. In this work, we provide a simpler model which we believe more easily captures the essence of an autoencoder, and apply it to ground states of the Hubbard model and molecular Hamiltonians.

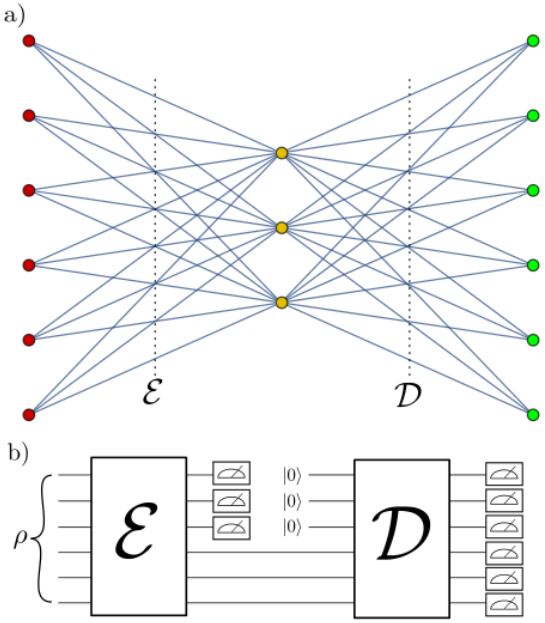

Figure 1. a) A graphical representation of a 6-bit autoencoder with a 3-bit latent space. The map $\mathcal{E}$ encodes a 6-bit input (red dots) into a 3-bit intermediate state (yellow dots), after which the decoder $\mathcal{D}$ attempts to reconstruct the input bits at the output (green dots). b) Circuit implementation of a 6-3-6 quantum autoencoder.

* Corresponding author: aspuru@chemistry.harvard.edu

II. QUANTUM AUTOENCODER MODEL

In analogy with the model of classical autoencoders, the quantum network has a graphical representation consisting of an interconnected group of nodes. In the graph of the quantum network, each node represents a qubit, with the first layer of the network representing the input register and the last layer representing the output register. In our representation, the edges connecting adjacent layers represent a unitary transformation from one layer to the next. Autoencoders, in particular, shrink the space between the first and second layer, as depicted in Figure 1a.

For a quantum circuit to embody an autoencoder network, the information contained in some of the input nodes must be discarded after the initial "encoding" $\mathcal{E}$. We imagine this takes place by tracing over the qubits representing these nodes (in Figure 1b, this is represented by a measurement on those qubits). Fresh qubits (initialized to some reference state) are then prepared and used to implement the final "decoding" evolution $\mathcal{D}$, which is then compared to the initial state.

The learning task for a quantum autoencoder is to find unitaries which preserve the quantum information of the input through the smaller intermediate latent space. To this end, it is important to quantify the deviation from the initial input state, $|\psi_i\rangle$, to the output, ρ_i^{out}. Here, we will use the expected fidelity [8] $F(|\psi_i\rangle, \rho_i^{out}) = \langle\psi_i| \rho_i^{out} |\psi_i\rangle$. We thus describe a successful autoencoding as one in which $F(|\psi_i\rangle, \rho_i^{out}) \approx 1$ for all the input states.

A more formal description of a quantum autoencoder follows: Let $\{p_i, |\psi_i\rangle_{AB}\}$ be an ensemble of pure states on $n+k$ qubits, where subsystems A and B are comprised of n and k qubits, respectively. Let $\{U^{\vec{p}}\}$ be a family of unitary operators acting on $n+k$ qubits, where $\vec{p} = \{p_1, p_2, \dots\}$ is some set of parameters defining a unitary quantum circuit. Also let $|a\rangle_{B'}$ be some fixed pure reference state of k qubits. Using classical learning methods, we wish to find the unitary $U^{\vec{p}}$ which maximizes the average fidelity, which we define to be the cost function,

$$C_1(\vec{p}) = \sum_i p_i \cdot F(|\psi_i\rangle, \rho_{i,\vec{p}}^{out}), \tag{1}$$

where,

$$\rho_{i,\vec{p}}^{out} = (U^{\vec{p}})^\dagger_{AB'} \operatorname{Tr}_B \left[U^{\vec{p}}_{AB} \left[\psi_{i_{AB}} \otimes a_{B'} \right] (U^{\vec{p}}_{AB})^\dagger \right] (U^{\vec{p}})_{AB'}, \tag{2}$$

and we have abbreviated $|\psi_i\rangle\langle\psi_i|_{AB} = \psi_{i_{AB}}$ and $|a\rangle\langle a|_{B'} = a_{B'}$. Equivalently, the goal is to find the best unitary $U^{\vec{p}}$ which, on average, best preserves the input state of the circuit in Figure 2 where instead of tracing over the B system, we employ a swap gate and trace over the B' system.

To prove this, consider the fidelity of the input and output of the entire system of Figure 2 for some fixed $\vec{p}$ where we

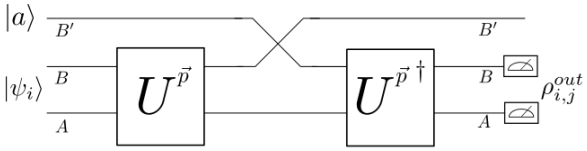

Figure 2. A quantum autoencoder circuit. The goal is to find $\vec{p}$ such that the averaged $F(|\psi\rangle_i, \rho_{i,\vec{p}}^{out})$ is maximized.

have denoted the swap operation by the unitary V,

$$\begin{aligned} &F(|\psi_i\rangle_{AB} \otimes |a\rangle_{B'}, U_{AB}^\dagger V_{BB'} U_{AB} |\psi_i\rangle_{AB} \otimes |a\rangle_{B'}) = \\ &F(U_{AB} |\psi_i\rangle_{AB} \otimes |a\rangle_{B'}, V_{BB'} U_{AB} |\psi_i\rangle_{AB} \otimes |a\rangle_{B'}) = \\ &F(|\psi_i'\rangle_{AB} \otimes |a\rangle_{B'}, V_{BB'} |\psi_i'\rangle_{AB} \otimes |a\rangle_{B'}) = \\ &F(|\psi_i'\rangle_{AB} \otimes |a\rangle_{B'}, |\psi_i'\rangle_{AB'} \otimes |a\rangle_B), \end{aligned} \tag{3}$$

where we have denoted $U|\psi_i\rangle = |\psi_i'\rangle$. The terms in the cost function are found by tracing out over the B' system,

$$\begin{aligned} &F(\operatorname{Tr}_{B'}[\psi'_{i_{AB}} \otimes a_{B'}], \operatorname{Tr}_{B'}[\psi'_{i_{AB'}} \otimes a_B]) = \\ &F(\psi'_{i_{AB}}, \rho'_A \otimes a_B), \end{aligned} \tag{4}$$

where $\rho'_A = \operatorname{Tr}_{B'} |\psi_i'\rangle\langle\psi_i'|_{AB'}]$. However, consider instead tracing over the AB system and looking at the "trash system" of B',

$$\begin{aligned} &F(\operatorname{Tr}_{AB}\left[\psi'_{i_{AB}} \otimes a_{B'}\right], \operatorname{Tr}_{AB}\left[\psi'_{i_{AB'}} \otimes a_B\right]) = \\ &F(|a\rangle_{B'}, \rho'_{B'}), \end{aligned} \tag{5}$$

where $\rho'_{B'} = \operatorname{Tr}_A[|\psi_i'\rangle\langle\psi_i'|_{AB'}]$. We henceforth refer to $\rho'_{B'}$ as the "trash state" of the circuit. It is straightforward to see in the above circuit that perfect fidelity (i.e. $C_1 = 1$) can be achieved by a unitary U if and only if, for all i:

$$U|\psi_i\rangle_{AB} = |\psi_i^c\rangle_A \otimes |a\rangle_B . \tag{6}$$

where $|\psi_i^c\rangle_A$ is some compressed version of $|\psi_i\rangle$. This follows because, if the B and B' systems are identical when the swap occurs, the entire circuit reduces to the identity map. However, this occurs precisely when the trash state is equal to the reference state, i.e., $F(|a\rangle_{B'}, \rho'_{B'}) = 1$. This implies that it is possible to accomplish the learning task of finding the ideal $U^{\vec{p}}$ by training *only on the trash state*. Furthermore, because Eq. (6) is completely independent of $U^\dagger$, this suggests that the circuit of Figure 2 can be reduced further. We then consider an alternative definition of the cost function in terms of the trash state fidelity,

$$C_2(\vec{p}) = \sum_i p_i \cdot F(\operatorname{Tr}_A \left[U^{\vec{p}} |\psi_i\rangle\langle\psi_i|_{AB} (U^{\vec{p}})^\dagger \right], |a\rangle_B), \tag{7}$$

Note, however, that the cost functions of Eq. (1) and Eq. (7) are not in general the same (in fact, $C_1 \leq C_2$). However, in practice, one must consider resource limitations; it is not hard to see that preparing copies of a fixed reference state would be easier than requiring identical copies of the input state to use in a SWAP test on the entire output state. For some applications of a quantum autoencoder, it may also be the case that

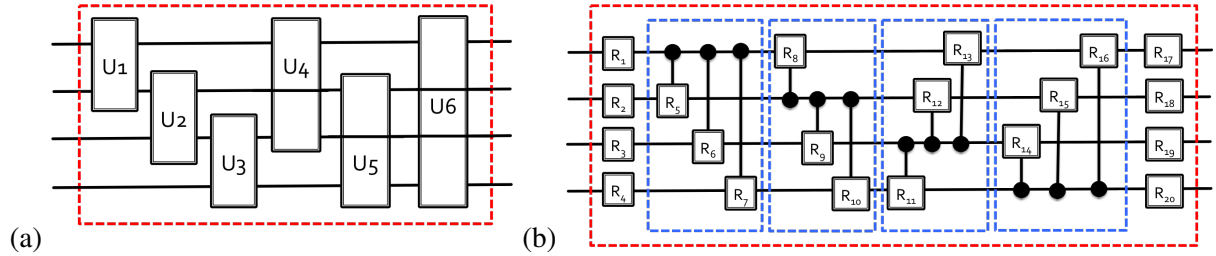

Figure 3. Two programmable circuits employed as autoencoder models: a) Circuit A: a network of all the possible two-qubit gates (denoted by U_i) among the qubits. b) Circuit B: a network comprising all the possible controlled general single-qubit rotations (denoted by R_i) in a qubit set, plus single qubit rotations at the beginning and at the end of the unit-cell. All the circuits are depicted in the case of a four-qubit input. The unit-cell is delimited by the red dotted line.

one has limited access to or limited knowledge of the input state.

It is interesting to note that, if we only care about circuits where $C_2 \approx 1$, we can re-imagine the problem to being one of finding a particular disentangling. It has been shown that, employing a circuit of exponential depth, one is always able to perform a disentangling operation [9], but to perform this operation in constant or polynomial depth is hard, and so classical heuristics are often used to find quantum circuits that are as close to optimal as possible. Also, information-theoretic bounds have been explored in this context before, both in the context of one-shot compression and one-shot decoupling [10, 11]. However, because the heuristics involved in choosing efficient-to-implement families of unitaries are largely ad-hoc, it is difficult to say if these bounds are meaningful in the context of a quantum autoencoder.

III. IMPLEMENTATION OF THE QUANTUM AUTOENCODER MODEL

To implement the quantum autoencoder model on a quantum computer we must define the form of the unitary, $U^{\vec{p}}$ (Eq. (2)) and decompose it into a quantum circuit suitable for optimization. For the implementation to be efficient, the number of parameters and the number of gates in the circuit should scale polynomially with the number of input qubits. This requirement immediately eliminates the possibility of using a $(n+k)$-qubit general unitary as $U^{\vec{p}}$ due to the exponential scaling in the number of parameters needed to generate them.

One alternative for the generation of $U^{\vec{p}}$ is to employ a programmable quantum circuit [12, 13]. This type of circuit construction consists of a fixed networks of gates, where a polynomial number of parameters associated to the gates i.e. rotation angles, constitute $\vec{p}$. The pattern defining the network of gates is regarded as a unit-cell. This unit-cell can ideally be repeated to increase the flexibility of the model. For the numerical assessment presented in this work, we employed two simple programmable circuits illustrated in Figure 3.

Circuit A has a unit-cell comprising a network of general two-qubit gates where we have considered all the possible pairings between qubits, as illustrated in Figure 3a for the four-qubit case. Accordingly, this model requires $15n(n-1)/2$ training parameters per unit-cell. A network of arbitrary two qubit gates can be easily implemented using state of the art superconducting qubit technologies [14] and the standard decomposition of a two-qubit gate into three CNOT gates and single-qubit rotations [15]. Arbitrary two qubit-gates have been also implemented using ion traps [16] and quantum dots [17].

Circuit B has a unit-cell comprising all the possible controlled one-qubit rotations among a set of qubits, complemented with a set of single qubit rotations at the beginning and at the end of the unit-cell, as shown in Figure 3b for the four-qubit case. We start considering the rotations controlled by the first qubit, followed by the rotations controlled by the second qubit and so on. Accordingly, our second model comprises $3n(n-1)+6n$ training parameters per unit-cell and can be implemented in state of the art quantum hardware using the standard decomposition of controlled unitaries into two CNOT gates and single-qubit rotations [18]. This model is also general and can be modified by adding constraints to the parameters. For instance, one could consider the initial and final layers of rotations to be all the same.

Once the circuit model has been chosen, we must train the network by maximizing the autoencoder cost function Eq. (7), in close analogy to classical autoencoders. Our training procedure adopts a quantum-classical hybrid scheme in which the state preparation and measurement are performed on the quantum computer while the optimization is realized via an optimization algorithm running on a classical computer. Such hybrid schemes have been proposed in the context of quantum machine learning [19, 20] and variational algorithms for quantum simulation [21–24]. In the later case, several experimental demonstrations have been successfully carried out [1, 21, 25].

As described in Section II, the cost function of the quantum autoencoder is defined as the weighted average of fidelities between the trash state produced by the compression, and the reference state. These fidelities can be measured via a SWAP test [26] between the reference state and the trash state. Accordingly, our quantum register must comprise the input state, $|\psi_i\rangle$, and the reference state. In a single iteration of our training algorithm, we perform the following steps for each of the states in the training set:

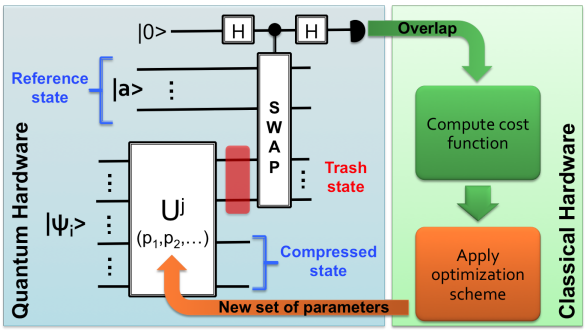

Figure 4. Schematic representation of the hybrid scheme for training a quantum autoencoder. After preparation of the input state, $|\psi_i\rangle$, the state is compressed by the application of the parameterized unitary, $U^{\vec{p}}$. The overlap between the reference state and the trash state produced by the compression is measured via a SWAP test. The results for all the states in the training set are collected to compute the cost function that is minimized using a classical optimization algorithm. The process is repeated until achieving convergence on the cost function and/or the values of the parameters, $\vec{p} = (\mathrm{p}_1, \mathrm{p}_2, \dots)$.

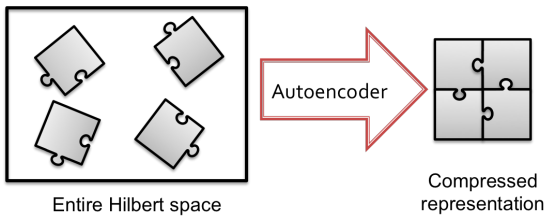

Figure 5. Graphical representation of Hilbert space compression. Given that the states of interest have support on only a subset $\mathcal{S}$ of the Hilbert space (gray pieces), the quantum autoencoder finds an encoding that uses a space of size $|\mathcal{S}|$.

1. Prepare the input state, $|\psi_i\rangle$, and the reference state. We assume these preparations to be efficient.

2. Evolve under the encoding unitary, $U^{\vec{p}}$, where $\vec{p}$ is the set of parameters at a given optimization step.

3. Measure the fidelity between the trash state and the reference state via a SWAP test.

With the estimates of all the fidelities, the cost function (Eq. (7)) is computed and fed into a classical optimization routine that returns a new set of parameters for our compression circuit. These steps are repeated until the optimization algorithm converges. Given that the value of the cost function is upper bounded by 1, we performed the optimization by minimizing the value of the function $\log_{10}(1 - C_2)$. This procedure is widely used in machine learning applications and helps prevent numerical instabilities [27]. A graphical summary of the hybrid scheme for training a quantum autoencoder is shown in Figure 4.

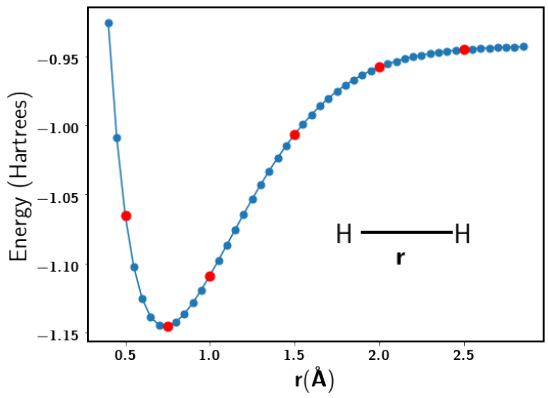

Figure 6. Potential energy surface for the hydrogen molecule using a STO-6G basis set. The ground states at the red dots where used as training set for the quantum autoencoder. The ground states at the blue dots were used for testing.

IV. APPLICATION TO QUANTUM SIMULATION

Consider a set of states, $\{|\psi_i\rangle\}$, with support on a subset of a Hilbert space $\mathcal{S} \subset \mathcal{H}$. Using a quantum autoencoder, we could find an encoding scheme that employs only $\log_2 |\mathcal{S}|$ qubits to represent the states instead of $\log_2 |\mathcal{H}|$, with a trash state of size $\log_2 |\mathcal{H} - \mathcal{S}|$. This idea is graphically depicted in Figure 5. This situation is usually encountered for eigenstates of many-body systems due to special symmetries.

Fermionic wavefunctions, for instance, are eigenfunctions of the particle number operator, same as the fermionic state vectors. Consequently, an eigenstate of a system with η particles is spanned exclusively by the subspace of fermionic state vectors with the same number of particles [28], that has size $\binom{N}{\eta}$ with N the number of fermionic modes. This result has direct implications for the design of quantum algorithms for simulation, suggesting that the number of qubits required to store fermionic wavefunctions could be reduced up to $\log \binom{N}{\eta}$ if an appropriate mapping can be found. The same situation is encountered for the spin projection operator, thus reducing the size of the subspace spanning a specific fermionic wavefunction even further.

Generally, the number of particles of the system is part of the input when finding eigenstates of many-body systems. In quantum chemistry simulations, the spin projection of the target state is also known. Many classical algorithms for simulating quantum systems take advantage of these constraints to reduce their computational cost [28]. However, the standard transformations employed to map fermionic systems to qubits, namely the Bravyi-Kitaev (BK) and the Jordan-Wigner (JW) mappings [29, 30], do not exploit these symmetries and thus employ more qubits than formally needed.

In this scenario, a quantum autoencoder could be trained to compress fermionic wavefunctions obtained using a quantum simulation algorithm that has been run using the standard transformations. The compression schemes obtained through this procedure could be employed to reduce the use of quan-

tum memory, if the wavefunction needs to be stored. It also could save quantum resources for the simulation of systems with similar symmetries. To illustrate this idea, we simulated a quantum autoencoder applied to molecular wavefunctions.

Within the Born-Oppenheimer approximation, the non-relativistic molecular Hamiltonian can be written as

$$H = h_{nuc} + \sum_{pq} h_{pq} a_p^\dagger a_q + \frac{1}{2} \sum_{pqrs} h_{pqrs} a_p^\dagger a_q^\dagger a_r a_s \qquad (8)$$

where h_{nuc} corresponds to the classical electrostatic repulsion between nuclei, and the constants h_{pq} and h_{pqrs} correspond to the one- and two-electron integrals (see Appendix A). The operators $a_p^\dagger$ and a_p creates and annihilates an electron in the spin-orbital p. After applying either the JW or the BK transformation, the molecular Hamiltonian can be expressed as $H = \sum_i^M c_i H_i$, with M scaling as $O(N^4)$. In this case, the operators H_i correspond to tensor products of Pauli matrices and the real coefficient c_i are linear combinations of the one- and two-electron integrals. For a fixed set of nuclei and a given number of electrons, the molecular integrals as well as the coefficients c_i are functions of the internal coordinates of the molecule, $\vec{R}$.

For instance, consider the Hamiltonian of molecular hydrogen in the STO-6G minimal basis set [28]. Using the JW transformation, the corresponding Hamiltonian acting on four qubits adopts the generic form [29]:

$$\begin{aligned} H = {} & c_0 I + c_1(Z_0 + Z_1) + c_2(Z_2 + Z_3) + c_3 Z_0 Z_1 + \\ & c_4(Z_0 Z_2 + Z_1 Z_3) + c_5(Z_1 Z_2 + Z_0 Z_3) + c_6 Z_2 Z_3 \\ & + c_7(Y_0 X_1 X_2 Y_3 - X_0 X_1 Y_2 Y_3 - Y_0 Y_1 X_2 X_3 + X_0 Y_1 Y_2 X_3) \end{aligned} \qquad (9)$$

In this case, the coefficients c_i are a function of the inter-nuclear distance, r. By solving the Schrödinger equation for the Hamiltonians at different values of r, we can obtain the ground state energy for molecular hydrogen and construct the potential energy surface (PES) shown in Figure 6. We expect that the ground state wavefunctions along the PES conserve the same number of particles and projection spin symmetries, turning this set of states into an excellent target for compression.

To illustrate the previous idea, we classically simulated a quantum autoencoder taking six ground states of the hydrogen molecule at different values of r, $\{|\Psi(r_i)\rangle\}_{i=1}^6$, as our training set. In this case, the weights of the states are chosen to be all equal. In real applications, we can imagine that the ground states are obtained using a quantum algorithm such as the quantum variational eigensolver [21]. We trained the circuit model described in Figure 3 to compress the training set of four-qubit states to two qubits and to one qubit, using $|0\rangle^{\otimes 2}$ and $|0\rangle^{\otimes 3}$ as reference states, respectively. Once the circuits were trained we tested them on 44 ground states corresponding to values of r different from those of the training set. This selection of the training and testing sets is shown in Figure 6.

The classical simulation was performed using a Python script supplemented with the QuTiP library [31, 32]. To simulate general two-qubit gates we employed the decomposition described in Ref.[16]. Arbitrary single-qubit rotations were implemented by decomposing them into Pauli-Z and Pauli-Y rotations, $R = R_z(\theta_1)R_y(\theta_2)R_z(\theta_3)$, ignoring global phases [18]. The optimization was performed using the SciPy implementation of the Basin-Hopping (BS) algorithm [33]. We also employed the L-BFGS-B method [34] with a numerical gradient (central finite difference formula with step size $h = 10^{-8}$). In the optimization of both circuit models, the parameters were constrained to the range $[0, 4\pi)$. The optimization of each circuit was initialized by setting the parameters to randomly selected values.

Table I. Average fidelity ($\mathcal{F}$) error after one cycle of compression and decompression using the quantum autoencoder trained from ground states of the Hydrogen molecule. We also report the error in the energy of the decoded state. (Maximum and minimum errors shown within parenthesis). 6 states were used for training and 44 more were used for testing. These results were obtained with the L-BFGS-B optimization.

Circuit	Final size (# qubits)	Set	$-\log_{10}(1-\mathcal{F})$ MAE	-$\log_{10}$ Energy MAE (Hartrees)
Model A	2	Training	6.96(6.82-7.17)	6.64(6.27-7.06)
	2	Testing	6.99(6.81-7.21)	6.76(6.18-7.10)
	1	Training	6.92(6.80-7.07)	6.60(6.23-7.05)
	1	Testing	6.96(6.77-7.08)	6.72(6.15-7.05)
Model B	2	Training	6.11(5.94-6.21)	6.00(5.78-6.21)
	2	Testing	6.07(5.91-6.21)	6.03(5.70-6.21)
	1	Training	3.95(3.53-5.24)	3.74(3.38-4.57)
	1	Testing	3.81(3.50-5.38)	3.62(3.35-4.65)

* MAE: Mean Absolute Error. Log chemical accuracy in Hartrees ≈-2.80

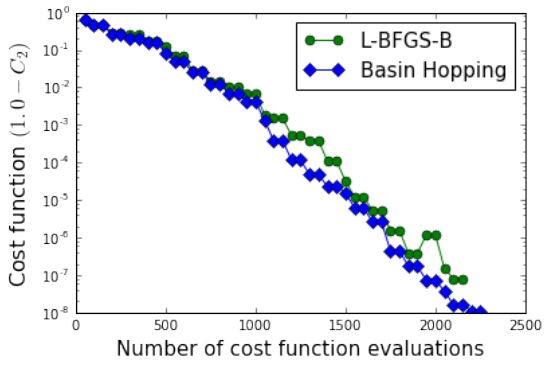

Figure 7. A plot of the cost function versus the number of cost function evaluations during the training process. This example corresponds to a quantum autoencoder for compression of wavefunction of H_2 from 4 to 2 qubits using circuit A with a training set of six ground states. We compared the L-BFGS-B and the Basin-Hopping algorithms for optimization.

Table I shows the average error in the fidelities and the energies obtained after a cycle of compression and decompression through the optimal quantum autoencoder. We observe that both circuit models are able to achieve high fidelities for

the encoding, producing decoded wavefunctions with energies that are close to the original values within chemical accuracy (1kcal/mol $\equiv 1.6 \times 10^{-3}$ Hartrees $\equiv 43.4$ meV). This accuracy requirement assures that quantum chemistry predictions have enough quality to be used for the estimation of thermochemical properties such as reaction rates [35].

Figure 7 illustrates the optimization process of a quantum autoencoder. We compared two different optimization algorithms, L-BFGS-B and Basin-Hopping. The parameters were initialized at random and the same guess was employed for both optimizations. As observed in Figure 7, both algorithms required a similar number of cost function evaluations to achieve similar precision and exhibit a monotonic reduction of the difference between the cost function and its ideal value with the number of function evaluations. The implementation of the quantum autoencoder in state of the art architectures can benefit from the use of algorithms that do not require gradient evaluations and have a larger tolerance to the presence of noise in the hardware, such as Basin-Hopping.

To gain insight into the compression process, we plotted the density matrices of the compressed states and compared them with the density matrix of the original state in Figure 8, for three different values of r. The sparsity of the original input density matrix is due to the symmetry of the Hamiltonian for molecular hydrogen, whose eigenvectors have support on only 2 computational basis states, allowing for a compression up to a single qubit. Although the quantum autoencoder achieves high fidelity with both types of circuit, the structure of the density matrices indicates that the forms of the compressed space and therefore the forms of the encoding unitaries differ between the two circuit models. As the values of r change, the relation between the features of the input space, here represented by the elements of the density matrix, and the features of the compressed space become apparent.

In addition to the example of molecular hydrogen, we tested the autoencoder compression scheme with ground states of the Hubbard model and the H_4 molecule. We considered half-filled Hubbard models with 2-sites and 4-sites arranged in a two-leg ladder (2×1 and 2×2 lattices, respectively). The Hamiltonian for these systems is given by

$$H = -t \sum_{<i,j>} \sum_{\sigma} a^{\dagger}_{i,\theta} a_{j,\sigma} + U \sum_{i} a^{\dagger}_{i,\uparrow} a_{i,\uparrow} a^{\dagger}_{i,\downarrow} a_{i,\downarrow} \quad (10)$$

where $a^{\dagger}_{i,\theta}$ and $a_{i,\sigma}$ create and annihilate an electron at site i with spin σ, respectively. The summation in the first term runs over all the interacting sites, denoted as $<i,j>$. We used periodic boundary conditions along the horizontal direction and open boundary conditions in the vertical direction. As in the case of molecular Hamiltonians, Hubbard Hamiltonians can be mapped to qubits using either the JW or the BK transformation, requiring two qubits per site.

We trained the two circuits of Figure 3 to compress the ground states of the Hubbard Hamiltonians obtained by setting t to six different values equally spaced between 0.9 and 1.1, with $U = 2.0$. The optimization process was repeated three times starting at randomly selected values. The same procedure was applied to the ground states of the H_4 system at six different values of the bond distance d (0.6, 1.4, 2.2, 3.0, 3.8 and 4.6 atomic units) for the geometry described in Figure 9.

Table II. Final error in the cost function (Eq. (7)) obtained after training a quantum autoencoder for compression of the ground states of two-sites and four-sites Hubbard models and the H_4 molecule along a parallel path. Six ground states were used for training each system. These results were obtained with the L-BFGS-B optimization.

Circuit	System	Compression rate* $n_o \to n_l$	$-\log_{10}(1-C_2)$
Model A	Hubbard 2x1 sites	$4 \to 3$	7.52
		$4 \to 2$	1.15
		$4 \to 1$	1.13
	Hubbard 2x2 sites	$8 \to 7$	2.28
		$8 \to 6$	1.42
	H_4	$8 \to 7$	1.53
		$8 \to 6$	1.6
Model B	Hubbard 2x1 sites	$4 \to 3$	6.82
		$4 \to 2$	3.92
		$4 \to 1$	4.02
	Hubbard 2x2 sites	$8 \to 7$	2.29
		$8 \to 6$	2.31
	H_4	$8 \to 7$	4.33
		$8 \to 6$	1.15

* n_o and n_l stand for the number of qubits in the original register and the number of qubits in the latent space, respectively.

Table II shows the lowest errors obtained for the compression of the Hubbard models and the H_4 system. Errors are quantified as the difference between the final value of the cost function (Eq. (7)) and the ideal value of 1. Recall that the cost function corresponds to the average fidelity over the training set. We observe that the ground states of the two-sites Hubbard model can be compressed from 4 to 3 qubits using both circuit types. However, only circuit B is able to compress these these states from 4 to 2 qubits and 4 to 1 qubits with an error below 10^{-3}. The same circuit-type achieves an error smaller than 10^{-4} when compressing the ground states of the H_4 system from 8 to 7 qubits. In contrast, circuit A is unable to obtain errors below 10^{-3} for H_4. In the case of the 4-sites Hubbard model, none of the circuit models proposed here was able to obtain errors below 10^{-3}.

The differences between the performances of the two circuit models described above exemplifies how the ansatz employed for the autoencoder unitary impacts the degree of compression achievable with the autoencoder model. Compression of a particular set of states could be achieved more easily with a dedicated ansatz designed for that purpose. One form of unitary that can serve as a template to design such dedicated ansatzes is given by the expression

$$U(\vec{\alpha}) = e^{-i \sum_i \alpha_i H_i} \quad (11)$$

where the real numbers α_i are the parameters for optimization and the terms H_i are local interactions consisting of tensor products of Pauli matrices. The experimental implementation

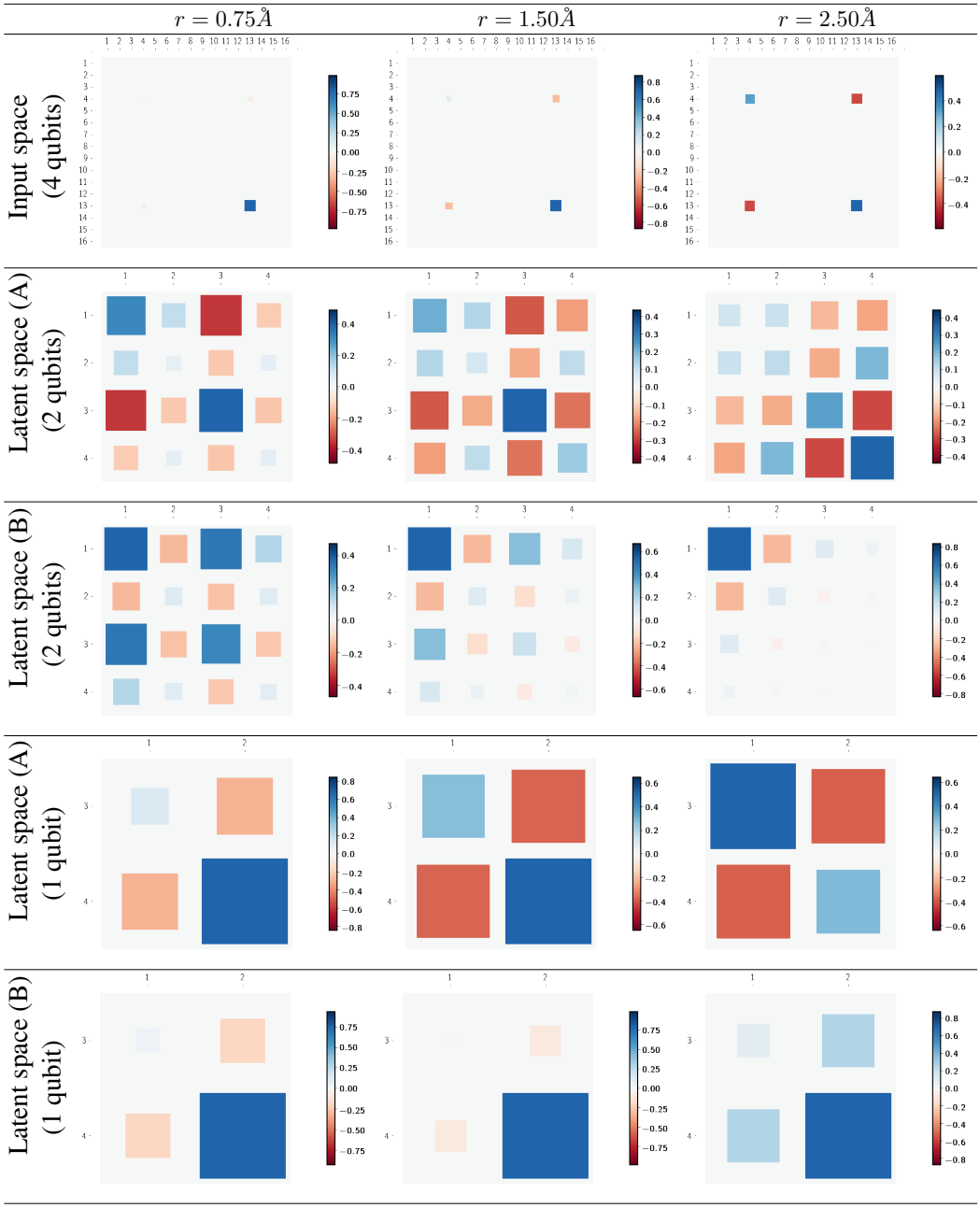

Figure 8. Visualization of the input space and the latent (compressed) spaces for three different testing instances of the H_2 compression, corresponding to three different bond distances, r. The input and latent spaces are characterized as the density matrices of the input and compressed states. Letters (A) and (B) denote the type of circuit employed to construct the encoding unitary. The size of the register (in qubits) appears within parenthesis. Integer labels starting at 1 denote the computational basis states in ascending order from $|00\cdots0\rangle$ to $|11\cdots1\rangle$.

of Eq. (11) would benefit from the techniques developed for quantum simulation algorithms [36].

Finally, we point out that the maximum rate of lossless compression achievable with a quantum autoencoder is predetermined by the size of the subspace spanning the training set. Consequently, a given set of states might only admit a small or null compression rate. For instance, consider a fermionic system with 8 fermionic modes and 4 particles, such as a half-filled 4-sites Hubbard model or the H_4 molecule in a minimal basis set studied here. Based solely on the con-

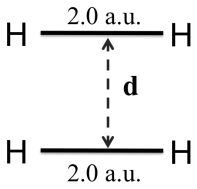

Figure 9. H_4 molecule in a parallel configuration, with the hydrogen atoms forming a rectangle. We obtained the ground states of this system at different values of d, with the bond distance in the two hydrogen molecules fixed to 2 atomic units (a.u.).

strain in the number of particles, these 8-qubits systems could be compressed to $\log_2 \binom{8}{4} \approx 7$ qubits. Compression beyond this point could be achieved if an extra symmetry constraint is present. In general, we expect fermionic systems where the number of fermionic modes is considerably larger than the number of particles to be good candidates for compression.

V. DISCUSSION

We have introduced a general model for a quantum autoencoder – a quantum circuit augmented with learning via classical optimization – and have shown that it is capable of learning a unitary circuit which can facilitate compression of quantum data, particularly in the context of quantum simulations. We imagine that the model can have other applications, such as compression protocols for quantum communication, error-correcting circuits, or perhaps to solve particular problems directly. A natural application for quantum autoencoders is state preparation. Once a quantum autoencoder has been trained to compress a specific set of states, the decompression unitary ($U^\dagger$) can be used to generate states similar to those originally used for training. This is achieved by preparing a state of the form $|\Psi_I\rangle \otimes |a\rangle$ and evolving it under $U^\dagger$, where $|\Psi_I\rangle$ has the size of the latent space and $|a\rangle$ is the reference state used for training.

Autoencoders as state preparation tools have direct application in the context of quantum variational algorithms [21–24]. These algorithms approximate the energy or the time evolution of an eigenstate by performing measurements on a quantum state prepared according to a parameterized ansatz. A quantum autoencoder could be trained with states of size n_o qubits, obtained from a given ansatz, and later be used as a state preparation tool as described above. Because the autoencoder parameters are fixed after training, the only active parameters in the variational algorithm would be those associated to the preparation of a state in the latent space (n_l). Since $n_l < n_o$, the state preparation with autoencoders would require fewer parameters than the original ansatz.

In our specification of the autoencoder, we define the input states to be an ensemble of pure states, and the evolution of those states to be unitary. The most generalized picture of the autoencoder, however, would allow for inputs to be ensembles of mixed states and the set $\{U^{\vec{p}}\}$ to be a set of quantum channels. In the case of mixed state inputs, we remark that this formulation can in principle be captured by our model already. More specifically, one can consider the case where a set of ancillas (denoted A') representing a purification of the mixed state is input into the autoencoder along with the original input. Ulhmann's theorem [8] guarantees that there exists a purification whose fidelity is identical to that of the mixed state generated from tracing out the purification; namely, it is a maximum over a unitary V acting on the purification alone (although finding this unitary can be a difficult computational problem itself). Consider then the encoding $U^{\vec{p}}_{AB} \otimes V_{A'}$, where the original latent space is expanded to contain all of A' (i.e. none of the ancilla qubits are traced out). This purified system will recover the behavior of the mixed system. The autoencoder structure as defined here cannot completely capture the structure for general quantum channels, though we expect other computational tasks may be solved by considering specific channel instances.

Are there any obvious limitations to the quantum autoencoder model? One consideration is that the von-Neumann entropy [8] of the density operator representing the ensemble $\{p_i, |\psi_i\rangle_{AB}\}$ limits the number of qubits to which it can be noiselessly compressed. However, finding the entropy of this density operator is not trivial – in fact, given a circuit that constructs a density operator ρ, it is known that, in general, even estimating the entropy of ρ is QSZK-complete [37]. This then opens the possibility for quantum autoencoders to efficiently give *some* estimate of the entropy of a density operator. In a similar vein, the unitary of the autoencoder could be defined to include the action of a quantum channel, and the autoencoder used to give both an encoding for and some lower bound for the capacity of a quantum communication channel (although the trash state may no longer be useful for training the autoencoder in some of these cases).

It is natural to consider whether the quantum autoencoder structure we have defined is actually a generalization of its classical counterpart, as in the construction of [7]. It may certainly be possible that some particular family of unitaries, together with certain choices for n and k, can be constructed so that a mapping exists. However, it is unclear that such a correspondence would even be desirable. Rather, we believe the value of autoencoders in general lies in the relatively simple structure of forcing a network to preserve information in a smaller subspace, as we have defined here.

Another topic of interest for any quantum computing model is the computational complexity exhibited by the device. For our construction, it is clear that any complexity result would be dependent upon the family of unitaries that is chosen for the learning to be optimized over. As the training itself is based on classical optimization algorithms (with no clear 'optimal' learning method), this further obfuscates general statements regarding the complexity of the model.

ACKNOWLEDGEMENTS

The authors would like to thank Yudong Cao, Peter Johnson, Ian Kivlichan, Nicolas Sawaya, Libor Veis, and Dominic

Berry for very insightful discussions and suggestions. JR and AAG acknowledge the Air Force Office of Scientific Research for support under Award: FA9550-12-1-0046. AA-G and JO acknowledge support from the Vannevar Bush Faculty Fellowship program sponsored by the Basic Research Office of the Assistant Secretary of Defense for Research and Engineering and funded by the Office of Naval Research through grant N00014-16-1-2008. AA-G acknowledges the Army Research Office under Award: W911NF-15-1-0256. The authors thank the Harvard Odyssey cluster facility where the numerical simulations presented in this work were carried out.

Appendix A: Molecular integrals

Using atomic units, where the electron mass m_e, the electron charge e, Bohr radius a_0, Coulomb's constant and $\hbar$ are unity, we can write the nuclear repulsion, h_{nuc}, and one-electron and two-electron integrals, h_{pq} and h_{pqrs}, as

$$h_{pq} = \int d\sigma \varphi_p^*(\sigma) \left(-\frac{\nabla_{\vec{r}}^2}{2} - \sum_i \frac{Z_i}{|\vec{R}_i - \vec{r}|} \right) \varphi_q(\sigma) \quad \text{(A1)}$$

$$h_{pqrs} = \int d\sigma_1 \, d\sigma_2 \, \frac{\varphi_p^*(\sigma_1)\varphi_q^*(\sigma_2)\varphi_s(\sigma_1)\varphi_r(\sigma_2)}{|\vec{r}_1 - \vec{r}_2|} \quad \text{(A2)}$$

$$h_{nuc} = \frac{1}{2} \sum_{i \neq j} \frac{Z_i Z_j}{|\vec{R}_i - \vec{R}_j|} \quad \text{(A3)}$$

Where Z_i represents the nuclear charge, $\vec{r}$ and $\vec{R}$ denote electronic and nuclear spatial coordinates, respectively, and σ is now a spatial and spin coordinate with $\sigma_i = (\vec{r}_i; s_i)$. Summations run over all nuclei. $\varphi(\sigma)$ represent the spin-orbitals (one-electron functions), that are generally obtained from a self-consistent field (SCF) Hartree-Fock (HF) calculation.

[1] P. J. J. O'Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, et al., Phys. Rev. X **6**, 031007 (2016).

[2] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, Science **309**, 1704 (2005).

[3] C.-Y. Liou, J.-C. Huang, and W.-C. Yang, Neurocomputing **71**, 3150 (2008).

[4] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, Neurocomputing **139**, 84 (2014).

[5] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, arXiv preprint quant-ph/1611.09347 (2016).

[6] R. Gómez-Bombarelli, D. Duvenaud, J. M. Hernández-Lobato, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, arXiv preprint cs.LG/1610.02415 (2016).

[7] K. H. Wan, O. Dahlsten, H. Kristjánsson, R. Gardner, and M. Kim, arXiv preprint cs.AI/1612.00104 (2016).

[8] M. M. Wilde, *Quantum Information Theory* (Cambridge University Press, 2012).

[9] V. Shende, S. Bullock, and I. Markov, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **25**, 1000 (2006).

[10] N. Datta, J. M. Renes, R. Renner, and M. M. Wilde, IEEE Transactions on Information Theory **59**, 8057 (2013).

[11] F. Dupuis, M. Berta, J. Wullschleger, and R. Renner, Commun. Math. Phys. **328**, 251284 (2014).

[12] P. B. Sousa and R. V. Ramos, arXiv preprint quant-ph/0602174 (2006).

[13] A. Daskin, A. Grama, G. Kollias, and S. Kais, J. Chem. Phys **137**, 234112 (2012).

[14] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, Y. Campbell Chen, et al., Nature **508**, 500 (2014).

[15] B. Kraus and J. Cirac, Phys. Rev. A **63**, 062309 (2001).

[16] D. Hanneke, J. Home, J. Jost, J. Amini, D. Leibfried, and D. Wineland, Nat. Phys. **6**, 13 (2010).

[17] M. Veldhorst, C. Yang, J. Hwang, W. Huang, J. Dehollain, J. Muhonen, S. Simmons, A. Laucht, F. Hudson, K. M. Itoh, et al., Nature **526**, 410 (2015).

[18] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2010).

[19] S. Gammelmark and K. Mølmer, New. J. Phys **11**, 033017 (2009).

[20] J. Bang, J. Ryu, S. Yoo, M. Pawłowski, and J. Lee, New. J. Phys **16**, 073017 (2014).

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

[22] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, New. J. Phys **18**, 023023 (2016).

[23] D. Wecker, M. B. Hastings, and M. Troyer, Phys. Rev. A **92**, 042303 (2015).

[24] Y. Li and S. C. Benjamin, arXiv preprint quant-ph/1611.09301 (2016).

[25] R. Santagati, J. Wang, A. Gentile, S. Paesani, N. Wiebe, J. McClean, S. Short, P. Shadbolt, D. Bonneau, J. Silverstone, et al., arXiv preprint quant-ph/1611.03511 (2016).

[26] H. Buhrman, R. Cleve, J. Watrous, and R. De Wolf, Phys. Rev. Lett. **87**, 167902 (2001).

[27] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006), ISBN 0387310738.

[28] T. Helgaker, J. Olsen, and P. Jorgensen, *Molecular Electronic Structure Theory* (Wiley, 2013).

[29] J. T. Seeley, M. J. Richard, and P. J. Love, J. Chem. Phys. **137**, 224109 (2012).

[30] A. Tranter, S. Sofia, J. Seeley, M. Kaicher, J. McClean, R. Babbush, P. V. Coveney, F. Mintert, F. Wilhelm, and P. J. Love, International Journal of Quantum Chemistry **115**, 1431 (2015).

[31] J. Johansson, P. Nation, and F. Nori, Comput. Phys. Commun **183**, 1760 (2012).

[32] J. Johansson, P. Nation, and F. Nori, Comput. Phys. Commun **184**, 1234 (2013).
[33] D. J. Wales and J. P. Doye, J. Phys. Chem. A **101**, 5111 (1997).
[34] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, SIAM. J. Sci. Comput **16**, 1190 (1995).
[35] K. A. Peterson, D. Feller, and D. A. Dixon, Theor. Chem. Acc. **131**, 1 (2012).
[36] I. Georgescu, S. Ashhab, and F. Nori, Rev. Mod. Phys. **86**, 153 (2014).
[37] A. Ben-Aroya and A. Ta-Shma, arXiv preprint quant-ph/0702129 (2007).