Open Source Variational Quantum Eigensolver Extension of the Quantum Learning Machine (QLM) for Quantum Chemistry

Mohammad Haidar, Marko J. Rančić, Thomas Ayral, Yvon Maday, Jean-Philip Piquemal

Received: 22 August 2022 | Revised: 28 November 2022 | Accepted: 23 December 2022

DOI: 10.1002/wcms.1664

SOFTWARE FOCUS

WIREs COMPUTATIONAL MOLECULAR SCIENCE WILEY

Open source variational quantum eigensolver extension of the quantum learning machine for quantum chemistry

Mohammad Haidar[1,2,3] | **Marko J. Rančić**[3] | **Thomas Ayral**[4] | **Yvon Maday**[2,5] | **Jean-Philip Piquemal**[1]

[1]Sorbonne Université, Laboratoire de Chimie Théorique (UMR-7616-CNRS), Paris, France

[2]Sorbonne Université, CNRS, Université Paris Cité, Laboratoire Jacques Louis Lions (LJLL), Paris, France

[3]TotalEnergies, Tour Coupole La Défense, Paris, France

[4]Atos Quantum Laboratory, Les Clayes-sous-Bois, France

[5]Institut Universitaire de France, Paris, France

Correspondence
Mohammad Haidar, Sorbonne Université, Laboratoire de Chimie Théorique (UMR-7616-CNRS) and Laboratoire Jacques Louis Lions (UMR-7598-CNRS), 4 place Jussieu, 75005 Paris, France.
TotalEnergies, Tour Coupole La Défense, 2 Pl. Jean Millier, 92078 Paris, France.
Email: mohammad.haidar@upmc.fr; Mohammadhaidar2016@outlook.com

Marko J. Rančić, TotalEnergies, Tour Coupole La Défense, 2 Pl. Jean Millier, 92078 Paris, France.
Email: marko.rancic@totalenergies.com

Jean-Philip Piquemal, Sorbonne Université, Laboratoire de Chimie Théorique (UMR-7616-CNRS), 4 place Jussieu, 75005 Paris, France.
Email: jean-philip.piquemal@sorbonne-universite.fr

Funding information
H2020 European Research Council, Grant/Award Number: EMC2 / grant agreement No 81036; Horizon 2020 Framework Programme, Grant/Award Number: ⟨NE|AS|QC⟩ project

Edited by: Peter R. Schreiner, Editor-in-Chief

Abstract

Quantum chemistry (QC) is one of the most promising applications of quantum computing. However, present quantum processing units (QPUs) are still subject to large errors. Therefore, noisy intermediate-scale quantum (NISQ) hardware is limited in terms of qubit counts/circuit depths. Variational quantum eigensolver (VQE) algorithms can potentially overcome such issues. Here, we introduce the OpenVQE open-source QC package. It provides tools for using and developing chemically-inspired adaptive methods derived from unitary coupled cluster (UCC). It facilitates the development and testing of VQE algorithms and is able to use the Atos Quantum Learning Machine (QLM), a general quantum programming framework enabling to write/optimize/simulate quantum computing programs. We present a specific, freely available QLM open-source module, myQLM-fermion. We review its key tools for facilitating QC computations (fermionic second quantization, fermion-spin transforms, etc.). OpenVQE largely extends the QLM's QC capabilities by providing: (i) the functions to generate the different types of excitations beyond the commonly used UCCSD ansatz; (ii) a new Python implementation of the "adaptive derivative assembled pseudo-Trotter method" (ADAPT-VQE). Interoperability with other major quantum programming frameworks is ensured thanks to the myQLM-interop package, which allows users to build their own code and easily execute it on existing QPUs. The combined OpenVQE/myQLM-fermion libraries facilitate the implementation, testing and development of variational quantum algorithms, while offering access to large molecules as the noiseless Schrödinger-style dense simulator can reach up to 41 qubits for any circuit. Extensive benchmarks are provided for molecules associated to qubit counts

ranging from 4 to 24. We focus on reaching chemical accuracy, reducing the number of circuit gates and optimizing parameters and operators between "fixed-length" UCC and ADAPT-VQE ansätze.

This article is categorized under:
Software > Quantum Chemistry
Quantum Computing > Algorithms

KEYWORDS

quantum chemistry, quantum computing, quantum learning machine, unitary coupled cluster, variational quantum eigensolver, unitary coupled cluster

1 | INTRODUCTION

Solving the Schrödinger equation to obtain the many-electron wavefunction is the central problem of modern quantum chemistry.[1,2] In practice, it is possible to achieve full accuracy for systems that contain few electrons through methods like the full configuration interaction (FCI) and the full coupled-cluster (FCC), which include all the electron configurations. However, the computational cost grows exponentially—through the dimension of the FCI and FCC wave functions—when the number of electrons increases. For example, the number of possible Slater determinants increases as a function of the number of electrons n_e and of orbitals n_o[3]:

$$\dim\mathcal{H} \approx \left(\frac{n_o!}{(n_o - n_e/2)!(n_e/2)!}\right)^2. \tag{1}$$

Attempts to reach full accuracy on large systems clearly faces the so-called "exponential wall" that limits the applicability of the most accurate methods to more complex chemical systems. So far, the largest calculations performed with classical supercomputers have only included tens of billions of determinants[4] with 20 electrons and 20 orbitals, with hopes to solve problems of a size close to a trillion determinants (24 electrons, 24 orbitals) in a near future thanks to the advances of massively parallel supercomputer architectures.[5] Given such constraints, other classes of methods have to be used to approximate the ground state wavefunctions for larger many-electron systems. They include: (i) Density-Functional Theory (DFT), which relies on the use of a single Slater determinant, and has been shown to be extremely successful while failing to describe strongly correlated systems[6–8]; (ii) post Hartree–Fock methodologies such as the truncated coupled-cluster (CC) and the configuration interaction (CI) approaches that, even though being still operational beyond the single Slater determinant, cannot be applied to large-size molecules, due to their extreme computational requirements in terms of Slater determinants.[9–16] A good example is provided by the "gold standard" methods denoted coupled-cluster single-, double-, plus perturbative triple-excitations CCSD(T). Indeed, CCSD(T) is able to handle a few thousand basis functions, but at the cost of an enormous operation count that is limited by the large requirements in term of data storage.[17] No matter the choice of chemical basis sets (STO-3G, 6-31G, cc-pVDZ, beyond, etc.), these methods are insufficient to reach accurate enough results for large molecules.

A paradigm shift proposed by Feynman[18,19] is to use quantum computers for simulating quantum systems. This has prompted the community to use quantum computers in order to solve quantum chemistry wavefunction problems. Intuitively, the advantage comes from the fact that quantum computers can process "exponentially" more information than classical computers.[20] Recent reviews provide background material about strategies for developing quantum algorithms dedicated to quantum chemistry. These approaches include techniques such as quantum phase estimation (QPE), variational quantum eigensolvers (VQEs), or quantum imaginary time evolution (QITE).[21–24] All methods generally include three key steps: (i) transforming the fermionic Hamiltonian and wavefunction into a qubit representation; (ii) constructing circuits with one and two-qubit quantum gates; (iii) using the circuits to generate a relevant wavefunction and measure the expectation value of a given Hamiltonian. Importantly, currently available quantum computers remain in the noisy intermediate-scale quantum (NISQ) era and are limited by two main resources: the

number of qubits and the circuit depth (i.e., number of quantum gates).[25–28] Among all available strategies, the VQE family of techniques is a very promising algorithm that can be applied to NISQ hardware.[21,29–31] Indeed, it is better suited to such devices than other algorithms such as QPE, thanks to its more modest requirements in terms of quantum processor coherence times. It has already been successfully executed on real NISQ quantum computers based on various technologies, including superconducting qubits,[32,33] photons[29] and trapped ions.[34] The performance of VQE algorithms is known to be highly dependent on the type of wavefunction ansatz. In practice, several types of ansätze can be used including: unitary coupled-cluster (UCC) wavefunctions,[35,36] hardware-efficient,[33,37] qubit coupled cluster (QCC),[38] deep multi-scale entanglement renormalization ansatz (DMERA)[39] and Hamiltonian variational.[40] Furthermore, there are recent hybrid approaches such as adaptive ansätze[41] and the subspace expansion method,[42] which also utilize the basic VQE scheme to build the wavefunction of the system.

In this article, we provide a newly-designed software package named "OpenVQE" that extends the Atos Quantum Learning Machine (QLM). QLM is a quantum computing platform that allows to write hardware-agnostic quantum programs, compile them in compliance with hardware constraints, and either emulate them classically with realistic noise models or execute them on actual quantum hardware. QLM is a mix between commercial tools and non-commercial ones provided by the Atos company. OpenVQE is fully compatible with the freely available open source component of QLM called "myQLM," which is available online.[43] myQLM is a quantum software stack for writing, simulating, optimizing, and executing quantum programs. It provides, through a Python interface: (i) powerful semantics for manipulating quantum circuits, with support for universal as well as custom gate sets, abstract parameters, advanced linking options, and so forth; (ii) a versatile execution stack for running quantum jobs, including an easy handling of observables, special plugins for carrying out NISQ-oriented variational methods (such as VQE or QPE), and an easy application programming interface (API) for writing customized plugins (e.g., for compilation or error mitigation), as well as for connecting to any quantum processing unit (QPU); (iii) a seamless interface to available quantum processors and major quantum programming frameworks.[44] The capacity of QLM/myQLM to interoperate with other packages such as Qiskit,[45] Cirq[46] and PyQuil[47] allows researchers from different horizons to communicate, interact, exchange, and work more efficiently and faster. Furthermore, myQLM acts as a universal platform for users who want to access various quantum computers (IBM, Google, etc.), providing them simple tools to run directly their jobs. myQLM comes with fermionic second quantization tools, which are helpful for solving quantum chemistry problems. These tools are now included in a new specialized open-source module dedicated to quantum chemistry called "myQLM-fermion." myQLM-fermion will be described in the present work; OpenVQE is designed to work synergistically with it. myQLM-fermion includes the key QLM resources that are important for quantum chemistry developments. More specifically, we use two important tools provided by myQLM-fermion which are useful for quantum resource reductions[35]: (i) an active space (AS) selection approach that is useful for the reduction of the number of qubits; (ii) an MP2 pre-screening approach that involves an implementation of second order Møller-Plesset perturbation (MP2) amplitudes as an input guess that is useful for faster parameter optimization. Therefore, by building on top of the initial myQLM blocks, we were able to design our OpenVQE package, which provides the community with advanced VQE tools dedicated to solve quantum chemistry problems. OpenVQE especially focuses on giving access to modules enabling the use of the UCC family and to adaptive ansatz algorithms. These modules include various features such as:

1. Different types of UCC generators (truncated to single and double excitations): (i) Unitary Coupled Cluster Singles and Doubles (UCCSD[48]) approaches; (ii) Unitary Pair CC with Generalized Singles and Doubles Product (*k*-UpCCGSD[49]) techniques; (iii) the possibility of including excitations that form spin-complemented pair interactions[41] and excitations that conserve only singlet spin symmetry of UCCGSD (see appendix of reference 50); (iv) Qubit Unitary Coupled Cluster Singles and Doubles QUCCSD.[51]
2. Adaptive VQE algorithms (namely Adaptive Derivative-Assembled Pseudo-Trotter-VQE (ADAPT-VQE)), with different operator pools including: (i) Unitary fermionic operators (fermionic-ADAPT-VQE[41]); (ii) Anti-Hermitian Pauli operators (qubit-ADAPT-VQE[50,52]).

These modules are structured in simple Python classes that require only to write a few lines of code and provide therefore examples enabling a rapid implementation of additional algorithms on the basis of myQLM basic tools.

The goal of this article is to showcase the novel OpenVQE/myQLM-fermion combined quantum simulator package that facilitates the test, development and measurement of the computational requirements of the different VQE flavors toward their future potential implementation on real supercomputers. In particular, it allows to compare the accuracy

of the total molecular energies obtained thanks to various variants of the UCC method with popular quantum chemistry methods on classical computers. The GitHub websites of OpenVQE[53] and myQLM-Fermion[54] are public, under "MIT" and "Apache-2.0" licenses, respectively.

The article is organized as follows: we review the theory of UCC ansätze including the different types of excitations, the ADAPT-VQE approach and the different steps needed to implement it (Section 2). We then introduce the QLM library, specifically its myQLM-fermion open source extension, with a focus on its quantum chemistry tools, which we use develop OpenVQE (see Section 3). Then, based on myQLM-fermion components, we describe the OpenVQE package, with its implementation of advanced UCC ansätze (static or adaptive) (Section 4). Finally, executing the OpenVQE library on our in-house HPC architecture (a quantum learning machine at TotalEnergies), we simulate a full set of molecules ranging from 4 to 24 qubits. We first show the properties of the simulator applied to a set of molecules. Second, we describe how OpenVQE/myQLM-fermion can use active space selection and MP2 pre-screening initial guesses for different test molecules. Third, using our ADAPT-VQE module, we compare the fermionic and qubit-ADAPT-VQE results obtained on a set of molecules in terms of chemical accuracy, number of variational parameters, operators and quantum gates. Finally, we compare the group of "fixed-length" ansätze (using UCC modules) to the ADAPT-VQE method (Section 5). We conclude with a summary table comparing the features of the OpenVQE/myQLM-fermion packages with those implemented in other software packages, and we give some perspectives toward the development of new types of UCC ansätze and/or new variational algorithms within OpenVQE.

2 | REVIEW: UNITARY COUPLED CLUSTER AND ADAPTIVE DERIVATIVE-ASSEMBLED PSEUDO-TROTTER (ADAPT) WITHIN THE VQE ALGORITHM

We will use the following notations in the article: the indices i and j refer to the occupied spin-orbitals in the Hartree–Fock (HF) state; indices a and b refer to the unoccupied (or virtual) spin-orbitals in the HF state. Indices p,q,r and s are always related to spin-orbitals unless mentioned otherwise. When the orbitals are associated to an α-like symbol (p_α, q_α...) they refer to spin-up electrons, and those with a β-like (p_β, q_β...) refer to spin-down electrons. n_e and n_o (as given in the introduction) represent the number of electrons and orbitals, respectively. We also denote by η and N_A the number of active electrons and number of active spin-orbitals, respectively, and by n the number of qubits. In qubit representation, $|\psi_{\text{ref}}\rangle$ denotes the n-qubit initial state. It is usually a computational basis state $|q_0,q_1,q_2,q_3,...q_n\rangle$. We call $|\psi(\theta)\rangle$ the trial wave function and θ its variational parameter, while $U(\theta)$ denotes a parameterized unitary operator.

2.1 | The VQE with the unitary coupled cluster ansatz

2.1.1 | VQE-UCC in a nutshell

The VQE has been first developed by Peruzzo et al.[29] This hybrid quantum classical algorithm aims at finding the ground state energy E_0 of a given Hamiltonian H. In the context of quantum chemistry, the Hamiltonian is often written in the following fermionic second-quantized form:

$$H = \sum_{pq} h_{pq} c_p^\dagger c_q + \frac{1}{2} \sum_{pqrs} h_{pqrs} c_p^\dagger c_q^\dagger c_r c_s, \tag{2}$$

where $c_p^\dagger$ (c_q) are anti-commuting operators that create (annihilate) electrons in spin-orbital $p(q)$, respectively. The symbols h_{pq} and h_{pqrs} denote the one- and two-body integrals of the corresponding operators and spin-orbitals in Dirac notation, respectively. These integrals can be easily computed on a classical computer.

The VQE method is based on quantum variational theory, whose starting point is the Rayleigh-Ritz variational principle, which states that

$$\langle \psi(\theta)|H|\psi(\theta)\rangle \geq E_0, \tag{3}$$

namely the energy of the normalized "trial" or "ansatz" wave function $|\psi(\theta)\rangle$ provides an upper bound for the ground state energy E_0. Variational methods leverage this principle through an iterative process that minimizes the expectation value $E(\theta) = \langle\psi(\theta)|H|\psi(\theta)\rangle$ with respect to the parameterized trial state $|\psi(\theta)\rangle$. This process is supposed to converge to an optimal parameter set θ^* and hence give an approximation $E(\theta^*)$ of E_0.

One of the core challenges of VQE is the choice of the ansatz $|\psi(\theta)\rangle$.

One popular ansatz is the unitary coupled cluster (UCC) approach. It is inspired from the classical coupled cluster (CC) computational method,[9,15,55,56] which has been shown to be a good choice for reaching accurate results. The unitary version of CC, UCC, was originally introduced in the field of quantum chemistry computations.[14] It was later brought to the field of quantum computing[29,57] because of its unitary character (as opposed to the CC method). A comprehensive review of the UCC method is given in references 35,36. UCC yields variational states prepared by using unitary evolution under a sum of fermionic terms representing excitations of the initial Hartree–Fock (HF) state. The UCC wavefunction can be expressed as follows:

$$|\psi_{\mathrm{UCC}}\rangle = e^{T-T^\dagger}|\psi_{\mathrm{HF}}\rangle, \tag{4}$$

where T is the so-called cluster operator and $T^\dagger$ is its Hermitian conjugate. As T is the sum of the excitations at different levels $T = T_1 + T_2 + \cdots + T_x$ until x excitations, then

$$T - T^\dagger = \sum_{a,i}^{n_o-1} \theta_{ai}\left(c_a^\dagger c_i - c_i^\dagger c_a\right) + \sum_{a,b,i,j}^{n_o-1} \theta_{abij}\left(c_a^\dagger c_b^\dagger c_i c_j - c_j^\dagger c_i^\dagger c_b c_a\right) + \cdots, \tag{5}$$

where $a,b \in virt, i,j \in occ$, θ_{ai} and θ_{abij} are variational parameters.

This form of variational state ensures that UCC applied within VQE can systematically approximate the true ground eigenstate of the system. This is not guaranteed in the case for example of the hardware efficient ansätze,[33,58–60] whose form is not inspired by chemistry considerations. Moreover, the hardware-efficient ansatz includes many optimization parameters, and can therefore suffer from the "barren plateau"[61,62] problem. The UCC optimization is not straightforward on a classical computer even for low-order cluster operators, due to the non-truncation of the Baker–Campbell–Hausdorff (BCH) series.[14] On the other hand, a UCC state can be easily prepared on a quantum device, with first experimental demonstrations on a trapped-ion quantum simulator.[57,63] There, UCC-VQE was shown to be able to reach a final state with a good overlap $|\langle\psi_{UCC}|\psi_0\rangle|$ between the converged UCC solution and the FCI solution $|\psi_0\rangle$ (see, e.g., fig. 3 in reference 57).

The implementation of the UCC wavefunction on a quantum computer is an important step in the VQE process. Further implementation aspects are described in Section 3. In particular, Figure 1 summarizes all the VQE process.

The UCC wavefunction in Equation (4) is implemented on a quantum device by first constructing a circuit corresponding to the qubit representation of the HF state, by using (for instance) the Jordan Wigner (JW) representation,[64] $|\psi_{HF}\rangle_Q = |11110000...\rangle$ where $|0/1\rangle$ refers to a single qubit being in the state 0 or 1. In the multi-qubit register, we assign each qubit to a spin-orbital, with orbital energies ranging from low to high. The $|0\rangle$ state refers to an unoccupied spin-orbital, while $|1\rangle$ refers to an occupied spin-orbital. The JW transformation relates spin-1/2 operators to fermionic creation and annihilation operators. We will use JW throughout the article. After having prepared $|\psi_{HF}\rangle_Q$ on the device, the unitary operator $U(\theta) = e^{T-T^\dagger}$ has to be implemented as well. This is done following these steps: $U(\theta)$ must be decomposed into operations that can be implemented as circuits compatible with present available quantum computers. To do this, in practice, $U(\theta)$ is approximated using the Trotter decomposition,[65–67] which breaks up the exponential of a sum as an ordered product of individual exponentials. Such transformation provides a reasonable approximation to the full unitary operator that is disassembled into local unitary operators. The formula of the trotterized $U(\theta)$ with $T(\theta) - T(\theta)^\dagger$ (Equation 5) is:

$$U(\theta) = \left(\prod_\rho e^{\frac{\theta_\rho}{t}\left(T_\rho - T_\rho^\dagger\right)}\right)^t + \mathcal{O}\left(\frac{1}{t}\right), \tag{6}$$

where t is the number of Trotter steps and ρ corresponds to the elements of excitations introduced in Equation (5). Using $t > 1$ can, on paper, decrease the trotterization error. Yet, for the current available NISQ devices, it is not

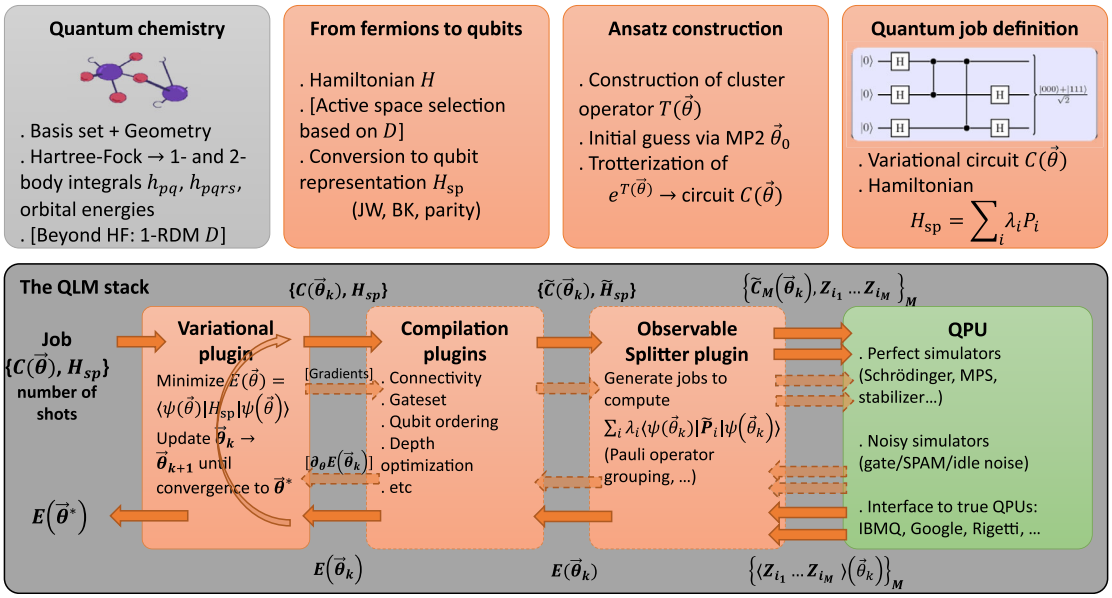


FIGURE 1 The QLM workflow for quantum chemistry. Top row: Steps to prepare a variational quantum job containing a parameterized circuit and the Hamiltonian whose ground state energy one wants to approximate. The leftmost (gray) box uses standard third-party quantum chemistry modules. Orange boxes stand for QLM libraries. Bottom row: QLM stack, with plugins (orange boxes) that pre- and post-process the job and results, and a QPU (green box) that executes the quantum job and returns a result.

recommended since it would involve additional gates induced by the UCC ansatz translation into a quantum circuit.[36,58] For instance, in reference 58 (Figure 5), increasing the number of Trotter steps t using a UCCSD ansatz for the H_2 molecule does not improve the convergence and the UCCSD energy. In the present work, we use a UCC ansatz truncated at the single and double excitation levels with a single Trotter step. Once these approximations are performed, each exponential of a fermionic excitation operator (see Equation 6) can be directly implemented as a sequence of gates. This is done using the standard "CNOT staircase method"[21,25,34] or a more advanced method[68–70] to reduce the CNOT counts. Further details about the implementation of the VQE-UCC method will be given in Section 3.2.

2.1.2 | Unitary coupled cluster approaches truncated at single and double excitations levels

In this subsection, we list the different versions of UCC that are truncated at the single and double excitation levels, by defining the excitation generator in each UCC version. We also review their recent use within variational algorithms. We first focus on the UCCSD and QUCCSD ansätze to compare their gate complexity and CNOT counts. We then focus on a second group of methods that includes UCCGSD and k-UpUCCGSD which, as shown in some reviews, perform better than the first group of ansätze in terms of chemical accuracy. However, such approaches require an increased circuit depth compared to the first group of techniques. These features have motivated us to implement these versions of UCC together with ADAPT-VQE (presented in the next section) in our OpenVQE package (see Section 4).

Unitary Coupled Cluster Single and Double was the first commonly used ansatz by the quantum chemistry community for quantum computing, and it was successfully tested experimentally on quantum computers using VQE.[34,57] Chemically, the UCCSD ansatz includes few excitations of fermionic single and double operators, which occur only from occupied to unoccupied (virtual) spin-orbitals on the top of the HF state as given in Equation (5). By assuming a single-step Trotter approximation (Equation 6), UCCSD is given by the product of single and double fermionic evolutions

$$U(\theta) = \prod_{a,i} e^{\left(\theta_{ai}\left(c_a^\dagger c_i - c_i^\dagger c_a\right)\right)} \prod_{a,b,i,j} e^{\left(\theta_{abij}\left(c_a^\dagger c_b^\dagger c_i c_j - c_j^\dagger c_i^\dagger c_b c_a\right)\right)}. \quad (7)$$

The Qubit Unitary Coupled Cluster Single and Double method is a simplified version of the UCCSD ansatz (Equation 7) in which the single and double excitations $q_a^\dagger$ (q_i) used to construct the trial wavefunction

$$U(\theta) = \prod_{a,i} e^{\left(\theta_{ai}\left(q_a^\dagger q_i - q_i^\dagger q_a\right)\right)} \prod_{a,b,i,j} e^{\left(\theta_{abij}\left(q_a^\dagger q_b^\dagger q_i q_j - q_j^\dagger q_i^\dagger q_b q_a\right)\right)}, \tag{8}$$

are the creation (annihilation) operators without the inclusion of Pauli-Z terms after JW transformation (see their expressions in chapter two, sec. 2.2 (eqs. 2.11 and 2.12) of reference 71). These excitations can be achieved by particle-preserving exchange rotation gates in the qubit space. The actual role of Pauli-Z is to conserve the parity from fermionic creation (annihilation) $c^\dagger$ (c) operators, while preserving the time and particle symmetries. By comparing QUCCSD to UCCSD ansatz VQE performance for a few simple molecules, it has been shown[51] that the two ansätze could achieve a nearly identical accuracy in the estimation of the ground state energies. Even though the parity property is missed in the qubit evolution excitations, it was found, as is described in figs. 1 and 2 of reference 51, that the advantage of using QUCCSD is to reduce the CNOT counts with respect to the CNOT staircase circuit to implement fermionic evolutions. This demonstrates that QUCCSD is sufficient to approximate the exact FCI wavefunction with a smaller number of gates in the circuit and it might therefore be more favorable to use on current NISQ devices. However, it has been shown that the operators in UCCSD and QUCCSD ansätze may not be ordered in practice and could decrease the precision in energy. Attempts for keeping ordered operators have been recently proposed.[35,38,41,72] Moreover it has been shown that an unnecessarily large number of parameters and operators occur when these ansätze were used, especially when system size gets larger. This is because the information about the specific chemical system such as point group symmetry were not taken into account. For example, as is shown in tab. I of reference 73, when the point group symmetry is used in UCCSD, the number of parameters reduce by at least 20%, which accelerates the optimization procedure and even brings little improvement, compared to UCCSD energies calculated without parameter reduction.

The Unitary Coupled Cluster Generalized Single and Double (UCCGSD) ansatz state has been first mentioned in reference 48, then later in reference 49 for using it in quantum computing applications. It consists into excitations that occur from occupied to occupied, and occupied to unoccupied levels, as is expressed in Equation (7), or from unoccupied to unoccupied levels:

$$\sum_{pq} \theta_{qp}\left(c_q^\dagger c_p - c_p^\dagger c_q\right) + \sum_{pqrs} \theta_{rspq}\left(c_r^\dagger c_s^\dagger c_q c_p - c_p^\dagger c_q^\dagger c_s c_r\right). \tag{9}$$

k-UpCCGSD stands for Unitary Pair Coupled Cluster with Generalized Single and Double Product,[49] where k denotes the products of unitary paired generalized double excitations, along with the full set of generalized single excitations. In other words, in k-UpCCGSD, single excitation operators are fully generalized without any specific constraint on the choice of occupied and virtual orbitals, but the double excitation operators are generalized and restricted to transitions of pair electrons acting on the same orbital:

$$T_2 - T_2^\dagger = \sum_{q_\alpha q_\beta p_\alpha p_\beta} \theta_{q_\alpha q_\beta p_\alpha p_\beta}\left(c_{q_\alpha}^\dagger c_{q_\beta}^\dagger c_{p_\beta} c_{p_\alpha} - c_{p_\alpha}^\dagger c_{p_\beta}^\dagger c_{q_\beta} c_{q_\alpha}\right). \tag{10}$$

It was recently demonstrated[49,74] that the UCCGSD ansatz leads to more accurate results for ground state energies than the simpler UCCSD. In the same work, it was shown that starting from $k = 3$ (i.e., 3-UpCCGSD) brings better accuracy than the UCCSD ansatz (see, e.g., tab. 4 for H_4 molecule using 6-31G basis set). Indeed, such approaches were also engineered not only for ground states but also to tackle excited states. They have been used in the variational quantum deflation algorithm (VQD), which is used to determine the energies of the ground and excited states.[75]

2.2 | Adaptive Derivative-Assembled Pseudo-Trotter-VQE

Instead of a fixed-length ansatz, that is, where one chooses UCCSD, UCCSDT or UCCSDTQ anszatz (length equal 2, 3 or 4, respectively), variational algorithms with a variable-length ansatz have been proposed by the community[76–80] There are two reasons for that: (1) the fixed-length ansätze use unnecessary excitations that can be considered as redundant terms as they do not contribute to a better approximation of the FCI wavefunction. This creates longer circuits and consumes a greater number of variational parameters; (2) fixed-length ansätze such as UCCSD or QUCCSD cannot

provide a good chemical accuracy for strongly correlated systems[51] (i.e., see, e.g., the H_6 molecule in Section 5). This problem occurs at long bond lengths and could be solved by including higher-order excitations and/or using multiple-step trotterization in the UCCSD or QUCCSD ansatz. Yet, both suggestions are problematic since they would necessarily deepen the circuits and increase the number of variational parameters. Fixed-length ansätze like 3-UpCCGSD (or $k>3$) and UCCGSD might also provide very good accuracy at these bond lengths, because the excitations considered are general. However, this yields many redundant terms during optimization, and as the system size increases, the number of operators and parameters, as well as the circuit depth, increase. The ADAPT-VQE approach is an important step toward solving these issues.[41,50]

The ADAPT-VQE algorithm constructs the molecular system's wavefunction dynamically and can in principle avoid redundant terms. It is grown iteratively in the form of a disentangled UCC ansatz as given in Equation (11). At each step, an operator or a few operators are chosen from a pool: the operator(s) contributing to the largest energy deviations is (are) chosen and added gradually to the ansatz until the exact FCI wavefunction has been reached

$$\prod_{k=1}^{\infty}\prod_{pq}\left(e^{\theta_{pq}(k)\widehat{A}_{p,q}}\prod_{rs}e^{\theta_{pqrs}(k)\widehat{A}_{pq,rs}}\right)|\psi_{\mathrm{HF}}\rangle, \tag{11}$$

which is in the form of a long product of one-body $\widehat{A}_{p,q}$ and two-body $\widehat{A}_{pq,rs}$ operators generated from the pool of excitations, where each of the variational parameters $\{\theta_{pq},\theta_{pqrs}\}$ is associated to an operator.

Let us describe briefly the procedure to implement the ADAPT-VQE (see details in reference 41). A reference Hartree–Fock $|\psi_{HF}\rangle$ state needs to be chosen first in the qubit representation that preserves the number of electrons of the system, which is an essential component for implementing the algorithm. At this stage, it is then possible to construct the ansatz by starting the ADAPT-VQE-iteration at $k=1$. The circuit is initialized by the identity $U^{(0)}(\theta)=I$, corresponding to the HF state as an initial state. Then to start constructing the ADAPT wavefunction, the algorithm starts (by a "while loop" until exit):

1. Then the trial state with the current ansatz on the quantum simulator $|\psi^{(k-1)}\rangle=U^{k-1}(\theta_{k-1})|\psi_{HF}\rangle$, where θ_{k-1} comes from the previous VQE iteration, has to be prepared;
2. In order to obtain the gradient (i.e., the energy derivative with respect to θ_{k-1}), one measures the commutator (in the qubit representation) between the Hamiltonian H (Equation 14) and each of the operators in the pool, of size m, which is defined by $A_m=\{A_m(p,q),A_m(p,q,r,s)\}$. The energy gradient formula reads

$$\frac{\partial E^{(k-1)}}{\partial\theta_m}=\langle\psi(\theta_{k-1})|[H,A_m]|\psi(\theta_{k-1})\rangle. \tag{12}$$

3. If the norm of the gradient vector $\left\|g^{(k-1)}\right\|=\sqrt{\left(\frac{\partial E^{(k-1)}}{\partial\theta_1}\right)^2+\ldots+\left(\frac{\partial E^{(m)}}{\partial\theta_m}\right)^2}$ becomes smaller than a threshold, ε, then the algorithm exits the loop. If not, the operator with the largest gradient $\max_{\theta_i}\left(\left|\frac{\partial E^{(1)}}{\partial\theta_1}\right|,\ldots,\left|\frac{\partial E^{(k-1)}}{\partial\theta_m}\right|\right)$ is selected, and added to the left end of the ansatz, with a new variational parameter $\theta_k=\theta_i$. The wavefunction corresponding to ADAPT-VQE is given by

$$|\psi^{(k)}\rangle=e^{\theta_kA_K}|\psi^{(k-1)}\rangle=e^{\theta_kA_k}\cdots e^{\theta_3A_3}e^{\theta_2A_2}e^{\theta_1A_1}|\psi_{HF}\rangle. \tag{13}$$

4. Perform a VQE experiment to re-optimize all parameters $\{\theta_k,\theta_{k-1},\ldots,\theta_2,\theta_1\}$ in the ansatz and when this is over we go back to step 1.

Historically, fermionic-ADAPT-VQE[41] was the first algorithm in the family of adaptive ansätze. The name "fermionic" comes from the excitation pool operators formed as spin-complement pairs of single and double fermionic evolutions (see Equations S1 and S2 are given in the Supplementary Material[81]). The fermionic-ADAPT-VQE has been shown to bring accurate chemical results for several molecules such as for H_4, LIH and H_6 molecules (as shown in fig. 2 of reference 41) by using an ansatz constituted of fewer parameters and shorter circuit depth than for the corresponding UCCSD results.

After this fermionic version, qubit-ADAPT-VQE[50,52] was introduced to reduce the number of gates compared to fermionic-ADAPT-VQE. In qubit-ADAPT-VQE, the pool used is a collection of anti-Hermitian operators where each operator is a tensor product of Pauli matrices. Each such operator is transpiled into a defined number of CNOT gates using the CNOT staircase method, but unlike the fermionic-ADAPT-VQE, the qubit-ADAPT-VQE produces a smaller number of gates at the end of the process. Qubit-ADAPT-VQE has been tested for several molecules[50] and lowers the gate counts by one order of magnitude compared to fermionic-ADAPT-VQE. However, qubit-ADAPT-VQE requires a higher number of parameters and iterations in order to reach a given level of accuracy.

The cost of fermionic- and qubit-ADAPT-VQE is the number of measurements needed to compute the energy gradients. It scales as $O(n^8)$[41,50] (where n represents the number of qubits). The choice of pool in qubit-ADAPT-VQE has also been important in the research to reach and approximate the FCI wavefunction, with a view to not only reduce the circuit depth compared to fermionic-ADAPT-VQE but also to minimize the number of operators in the pool without losing the completeness properties of operators. Reducing the pool size from $\mathcal{O}(n^4)$ into $2\,n-2$ operators and maintaining the necessary and sufficient conditions for completeness has been achieved in a recent work.[52] This new pool size reduces the number of measurements from $O(n^8)$ to $O(n^5)$ for a given molecular system. In the same reference, it was shown that incorporating symmetries into the pool solves the convergence problems caused by the lack of symmetries.

3 | ALGORITHMS FOR QUANTUM CHEMISTRY ON THE QLM

In this section, we briefly review the tools of the QLM that are relevant to quantum chemistry computations. We will then describe in the next section (Section 4) how we built an advanced chemistry module upon these tools.

3.1 | A quantum programming environment and a powerful simulator

QLM is a complete environment designed for quantum software programmers, engineers and researchers.[82] It includes a wide variety of low-level tools useful for writing, compiling and optimizing quantum circuits.[83–87] These tools come in the form of so-called "plugins" that can be combined or stacked upon another to construct a quantum compilation chain.

QLM can simulate quantum circuits in a noiseless or noisy fashion. The noiseless simulators come in different flavors, with "Schrödinger style" simulators (storing the wavefunction either in a dense fashion or using Matrix Product States[88]), a "Feynman-style" simulator,[89] a binary decision diagram simulator,[90] a Clifford simulator, and so forth. The Schrödinger-style dense simulator can reach up to 41 qubits for any circuit, while the other simulators can reach much larger qubit counts depending on the circuit properties (entanglement, gateset, etc.). The noisy simulators enable the emulation of realistic quantum noise, a crucial tool in the current NISQ era. Gate noise, state preparation and measurement (SPAM) noise and idling noise can be taken into account via density-matrix (resp. stochastic) simulations that can handle circuit with up to 20 (resp. 40) qubits. Most importantly, the interface of the simulators (also called "Quantum Processing Units" or QPUs) is such that they can easily be swapped for actual experimental QPUs without modifying the quantum program. QLM provides an interface to various hardware processors including those constructed by IBM, Rigetti, and so forth. Its circuits are also interoperable[44] with other quantum circuit descriptions like Qiskit,[45] Cirq[46] and PyQuil,[47] as described in more detail in Section SI in the Supplementary Material.[81] Moreover, QLM comes with quantum application libraries that enable the easy exploration of potential use cases of quantum computing. These libraries help generate quantum programs in fields ranging from combinatorial optimization to quantum chemistry and condensed-matter physics. A summary of the QLM workflow for quantum chemistry is provided in Figure 1. The top row represents the steps one has to go through to handle a quantum chemistry problem using `myQLM-fermion` (bracketed terms represent optional steps), including the transformation of the electronic structure Hamiltonian (Equation 2) to a spin representation:

$$H_{\mathrm{sp}} = \sum_j \lambda_j P_j, \tag{14}$$

with $P_j = \prod_{i=0}^{n_o-1} \sigma_i^j$, and σ_i a Pauli matrix applied on qubit i. At the end of these steps, one obtains a "job" comprising a parameterized quantum circuit (that implements a UCC ansatz) $C\left(\vec{\theta}\right)$ and a Hamiltonian H_{sp} represented as a sum of Pauli terms. This job can then be fed to the QLM stack.

This stack, represented in the bottom row, can be constructed by the user at will depending on the QPU they want to execute the job on. A minimal stack consists of a QPU (without plugins), which can handle minimal jobs containing native quantum circuits (with gates native to the hardware and compliant with its connectivity) and Z-axis observables. To handle more sophisticated jobs, one can stack "plugins" on top of the QPU. Each plugin has a well-defined role, such as compiling the circuit to a given gateset, rewriting it to comply with a given connectivity (see "Compilation plugins"), or generating all the elementary jobs required to compute the expectation value of a general observable (see "Observable Splitter plugin"). Finally, some plugins can handle variational jobs corresponding to the VQE algorithm. Such "variational plugins" handle the update of the variational parameters based on the result of the previous steps. They can optionally generate jobs to compute the gradient of a given expectation value if gradient-based optimization is asked for.

In practice, the QLM environment is composed of software modules and a powerful classical HPC hardware appliance, as described in Figure 2. Among the software modules, some are available and downloadable for free as Python packages. They come under the name of `myQLM`.[43] `myQLM` includes modules for programming quantum circuits, with quantum libraries (for performing, e.g., quantum arithmetic operations, etc.), for describing quantum observables and analog quantum schedules. It also comes with a basic noiseless simulator for quantum circuits, basic plugins (e.g., for splitting observables depending on the commutation relations of the Pauli operators), and modules for interacting with other quantum frameworks like Qiskit, Cirq, and so forth. Finally, `myQLM` contains the aforementioned `myQLM-fermion` module.

`myQLM` can be installed on a laptop or on a local cluster. The size of the circuits that can be simulated with the QPU simulator of `myQLM` is limited by the available Random Access Memory (RAM) (for a standard laptop, this corresponds to about 25 qubits). The quantum learning machine is also a powerful HPC appliance. It comes with extra software in addition to `myQLM`: advanced noiseless simulators, modules for describing hardware models, noisy simulators (both gate-based and analog), advanced compilers, tomography and error mitigation modules, among others. The appliance itself can simulate circuits with up to 41 qubits when using state vector simulators, and more when using more specific representations of the wave function (like Matrix Product States or stabilizers). The software module we present below, `OpenVQE`, relies only on `myQLM` modules, which means it can be used by simply downloading `myQLM` on one's personal machine. Therefore, the results shown for small qubit counts can be (up to the run times) reproduced on a personal laptop. In practice, to execute the simulations, we used the QLM advanced noiseless statevector simulator so as to reach large qubit counts and gain speed.

3.2 | myQLM-fermion: An open source QLM module for fermionic many-body problems

The tools for quantum chemistry are collected into an open-source module called "myQLM-fermion," which is part of myQLM. `myQLM-fermion` provides general tools for dealing with fermionic problems: representations of fermionic Hamiltonians (including usual Hamiltonians like the electronic structure Hamiltonian [Equation 2], the Hubbard and Anderson Hamiltonians), transformations from fermion operators to qubit operators (Jordan-Wigner, Bravyi-Kitaev, parity), generation of quantum evolution circuits via trotterization, implementation of the QPE algorithm, as well as various modules for variational algorithms such as VQE (see Section 2.1): libraries of variational ansätze (like hardware-efficient ansätze, matchgate circuits, low-depth circuit ansatz [LDCA] circuits), various optimization plugins (including the standard optimizers implemented in scipy[91] [COBYLA, BFGS, etc.] as well as the simultaneous perturbation stochastic algorithm [SPSA] and particle-swarm optimizer [PSO]).

A submodule of `myQLM-fermion`[54] is specifically devoted to quantum chemistry. It provides tools for selecting active spaces (based on natural-orbital occupation numbers), generating cluster operators (and thus, via the aforementioned trotterization tools, UCC-type ansätze), and initial guesses for their variational parameters. The architecture of QLM and `myQLM-fermion` allows for experts in a given field to construct their own advanced modules with the QLM building blocks.The key building block of quantum chemistry computations is the Hamiltonian, Equation (2). On QLM, it is described by an object `ElectronicStructureHamiltonian`

```
from qat.fermion import ElectronicStructureHamiltonian
hamiltonian = ElectronicStructureHamiltonian(h, g)
```

where `h` and `g` are the tensors h_{pq} and h_{pqrs} of Equation (2). Such an object also describes cluster operators such as the ones described in Equation (5) truncated at $x = 2$. For instance, the following snippet.

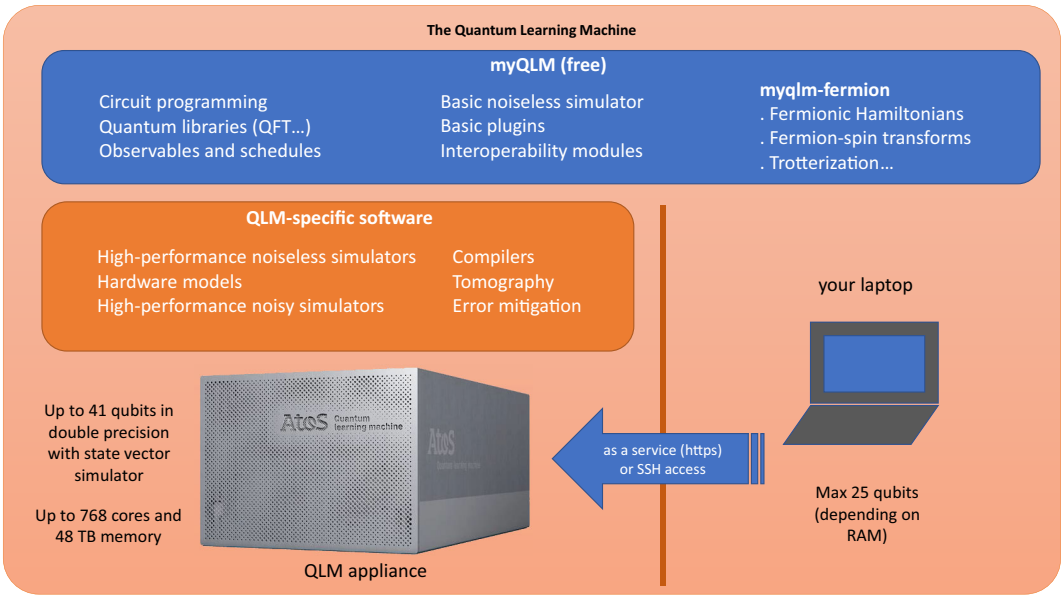


FIGURE 2 Overview of the QLM environment. Documentation of my-QLM/myQLM-fermion is given in references 43,54.

```
from qat.fermion import get_cluster_ops
cluster_ops = get_cluster_ops(n_electrons, nqbits=nqbits)
```

creates the list containing the sets of single excitations $\{c_k^\dagger c_i - c_i^\dagger c_k, k \in virt, i \in occ\}$ and double excitations $\{c_l^\dagger c_k^\dagger c_j c_i - c_j^\dagger c_i^\dagger c_l c_k, k,l \in virt, i,j \in occ\}$.

These objects can be readily converted to a spin (or qubit) representation using various fermion-spin transforms:

```
# Jordan-Wigner
from qat.fermion.transforms import transform_to_jw_basis
hamiltonian_jw = transform_to_jw_basis(hamiltonian)
cluster_ops_jw = [transform_to_jw_basis(t_o) for t_o in cluster_ops]

# Bravyi-Kitaev
from qat.fermion.transforms import transform_to_bk_basis
hamiltonian_bk = transform_to_bk_basis(hamiltonian)
cluster_ops_bk = [transform_to_bk_basis(t_o) for t_o in cluster_ops]
```

The transformed objects are now in the form of Equation (14). With these qubit operators, one can then easily construct a simple UCCSD ansatz via trotterization red (as given Equation 6) of the exponential of the parametric cluster operator defined as `cluster_ops_jw`:

```
from qat.lang.AQASM import Program, X
from qat.fermion.trotterisation import make_trotterisation_routine

prog = Program()
reg = prog.qalloc(nqbits)
# Create Hartree-Fock state (assuming JW representation)
for qb in range(n_electrons):
    prog.apply(X, reg[qb])

# Define the full cluster operator with its parameters
theta_list = [prog.new_var(float, "\\theta_{%s}" % i) for i in range(len(cluster_ops_jw))]
cluster_op = sum([theta * T for theta, T in zip(theta_list, cluster_ops_jw)])

# Trotterize the Hamiltonian (with 1 trotter step)
qrout = make_trotterisation_routine(cluster_op, n_trotter_steps=1, final_time=1)
prog.apply(qrout, reg)
circ = prog.to_circ()
```

The circuit we constructed, `circ`, is a variational circuit that creates a variational wavefunction $|\psi(\theta)\rangle$. Its parameters can be optimized to minimize the variational energy $E(\theta) = \sum_j h_j \langle \psi(\theta) \prod_{i=0}^{n_o-1} \sigma_i^j \psi(\theta) \rangle$. This is done by a simple VQE loop:

```
# create a quantum job containing the variational circuit and the Hamiltonian
job = circ.to_job(observable=hamiltonian_jw, nbshots=0)

# import a plugin to perform the optimization
from qat.plugins import scipyMinimizePlugin
optimizer_scipy = scipyMinimizePlugin(method="COBYLA", tol=1e-3, options={"maxiter": 1000}, x0=theta_init)

# import a QPU to execute the quantum circuit
from qat.qpus import get_default_qpu

# define the quantum stack
stack = optimizer_scipy | get_default_qpu()

# submit the job and read the result
result = stack.submit(job)

print("Minimum energy =", result.value)
```

The simulated QPU used in the previous code snippet can be readily replaced by an actual QPU by using the `qat-interop` module. For instance, in the following snippet, we use a transmon QPU by IBM:

```
from qat.interop.qiskit import BackendToQPU
qpu= get_default_qpu()
qpu = BackendToQPU(token=MY_IBM_TOKEN, ibmq_backend="ibmq_guadalupe")
```

Conversely, circuits created using other quantum programming frameworks can be converted to the QLM format. For instance, here we convert a circuit written in the Google Cirq format to a QLM circuit (that can then be fed to a QLM simulator):

```
from qat.interop.cirq import cirq_to_qlm
qlm_circ = cirq_to_qlm(your_google_circ)
```

4 | THE OpenVQE PACKAGE

OpenVQE facilitates the development of new algorithms on quantum computers. It consists of a couple of new modules that extend the main myQLM/myQLM-fermion implementations (see Section 3). Those modules allow to build normal UCCSD algorithm, its variants and ADAPT-VQE algorithms as reviewed in Section 2. To explain in another fashion, OpenVQE allows to perform calculations using new types of UCC methods that are different from the QLM predefined ones such as the regular UCCSD ansatz.

OpenVQE consists of two main modules that are inside the "openvqe" folder:

- **UCC Family** denoted in the code by "ucc_family": this module includes different classes and functions to generate the fermionic cluster operators (fermionic pool) and the qubit pools, and to get the VQE optimized energies in the cases of active and non-active orbital selections.
- **ADAPT** denoted by "adapt": it includes two sub-modules:
  - **Fermionic-ADAPT**: containing functions performing the fermionic-ADAPT-VQE algorithmic steps in the active and non-active space selections;
  - **Qubit-ADAPT**: containing functions that perform the qubit-ADAPT-VQE algorithmic steps calculation in the active and non-active space orbital selections.

Additionally, "openvqe" contains a subfolder named "common_files" that stores all the internal functions needed to be imported for executing the two modules. OpenVQE also contains a "notebooks" folder that allows the user to run and test the above two modules (which are theoretically described in the review Section 2). A brief sketch of the OpenVQE implementation is explained in the flow chart presented in Figure 3. We remind the reader that we will describe below the code of OpenVQE for the full space selections and some internal function examples such as those related to "generator_excitations" and "fermionic_adapt_vqe." The codes related to active space selections can be found in the GitHub website[53] inside "notebooks" with prefix "active_space."

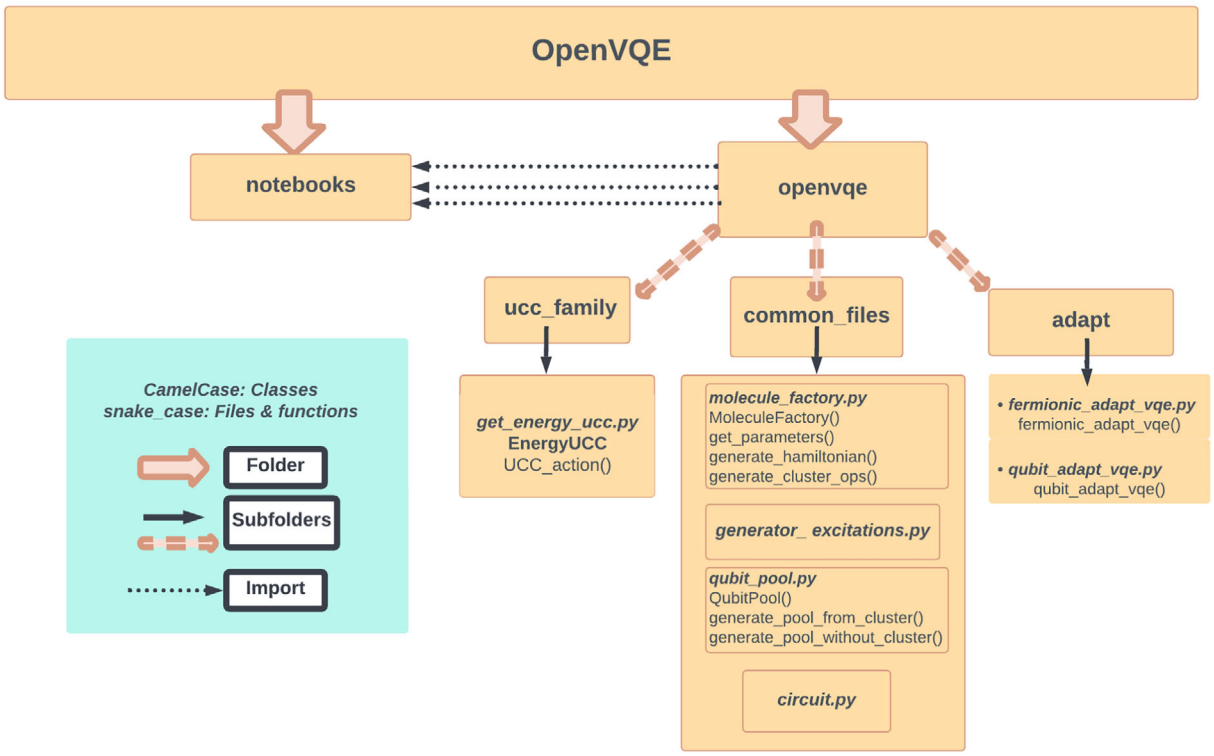


FIGURE 3 Flow chart of the OpenVQE package. The code is given in our Github repository and documentation.[53]

In order to develop a new algorithm using our OpenVQE package, it is important to list the different parameters and return variables contained in its main functions.The first crucial parameters are the ones specifying the characteristics of the molecules such as the symbol of the molecule; the type of excitation generator (generators to produce the excitations, see Section 2.1.2): UCCSD (Equation 7), QUCCSD (Equation 8), UCCGSD (Equation 9), *k*-UpCCGSD (Equation 10) and spin-complemented pair (Equations S1 and S2 in the Supplementary Material[81]); the spin transformation mapping (JW, Bravi-Kitaev, parity basis), choice of active or non-active space selection, denoted respectively in our code as `molecule_symbol`, `type_of_generator`, `transform`, and `active`.

```
# molecule examples: H2, H4, H6, LiH, H2O, CO, CO2, NH3 etc...
molecule_symbol = 'H2O'
# suppose user choose the spin-complemented generalized singlet and doublet  denoted as spin_complement_gsd
type_of_generator = 'spin_complement_gsd'
# user can type other type of generators, such as: uccsd, quccsd, uccgsd,  k-upccgsd, etc...
# we type Jordan wigner (JW),  user can  also type Bravyi-Kitaev or Parity basis
transform = 'JW'
# no active space selection, set:
active = False
# for active space (AS), switch to  "active = True"
```

The user specifies these parameters in a class called `MoleculeFactory`. This class takes those parameters as input:

```
from openvqe.common_files.molecule_factory import MoleculeFactory

# returns the properties of a molecule:
r, geometry, charge, spin, basis = MoleculeFactory.get_parameters(molecule_symbol = 'H2O')
```

define a function named as `generate_hamiltonian` that generates the electronic structure Hamiltonian (`hamiltonian`) and other properties such as the spin Hamiltonian (e.g., `hamiltonian_jw`), number of electrons (`n_els`), the list contains the number of natural orbital occupation numbers (`noons_full`), the list of orbital energies (`orb_energies_full`) where the orbital energies are doubled due to spin degeneracy and `info` which is a dictionary that stores energies of some classical methods(such as Hartree–Fock, CCSD and FCI):

```
Hamiltonian, hamiltonian_jw, n_els, noons_full, orb_energies_full, info = MoleculeFactory.generate_hamiltonian(
    molecule_symbol='H2O', active=False, transform='JW')
```

In addition to that, we define another function named as `generate_cluster_ops()` that takes as input the name of excitation generator user need (e.g., UCCSD, QUCCSD, UCCGSD, etc.) and internally it calls the file name *generator_excitations.py* which allows `generate_cluster_ops()` to return as output the size of pool excitations, fermionic operators, and JW transformed operators denoted in our code respectively as pool_size, `cluster_ops` and `cluster_ops_jw`:

```
from .generator_excitations import (uccsd, quccsd, singlet_gsd, singlet_sd,singlet_upccgsd, spin_complement_gsd,
    spin_complement_gsd_twin)
pool_size,cluster_ops,cluster_ops_jw = MoleculeFactory.generate_cluster_ops(molecule_symbol='H2O',
    type_of_generator='spin_complement_gsd', transform='JW', active=False)
# in our example 'spin_complement_gsd':
def generate_cluster_ops():
    pool_size,cluster_ops,cluster_ops_jw= None, None, None
    if type_of_generator == 'spin_complement_gsd':
        pool_size,cluster_ops,cluster_ops_jw   = spin_complement_gsd(n_el,n_orb,'JW')
#  elif for other excitations (uccsd, quccsd, singlet_upccgsd...)
#    ::::
    return  pool_size,cluster_ops,cluster_ops_jw
```

Once these are generated, we import them as input to the UCC-family and ADAPT modules.In the example of fermionic-ADAPT sub-module, we call the function `fermionic_adapt_vqe()` that takes as parameters the fermionic cluster operators, spin Hamiltonian, maximum number of gradients to be taken per iteration, the type of optimizer, tolerance, threshold of norm (ε) and the maximum number of adaptive iterations:

```
from openvqe.adapt.fermionic_adapt_vqe import fermionic_adapt_vqe

# choose maximum number of gradients needed (1,2,3....)
n_max_grads = 1
# choose optimizer needed (COBYLA, BFGS, SLSQP, Nelder-Mead etc...)
optimizer = 'COBYLA'
tolerance = 10**(-6)
# according to a given norm value we stop the ADAPT-VQE loop
type_conver = 'norm'
threshold_needed = 1e-2
# the maximum external number of iterations to complete the ADAPT-VQE under a given threshold_needed
max_iterations = 35
fci = info['FCI']
# sparse the Hamiltonian and cluster operators using myQLM-fermion tools obtained from MoleculeFactory, which are
    explicitly:
hamiltonian_sparse = hamiltonian_jw.get_matrix(sparse=True)
cluster_ops_sparse = cluster_ops.get_matrix(sparse=True)
# reference_ket  and hf_init_sp can be obtained from class MoleculeFactory():
reference_ket, hf_init_sp = MoleculeFactory.get_reference_ket(hf_init, nbqbits, 'JW')
# when all these parameters are satisfied, then fermionic-ADAPT-VQE function is:
fermionic_adapt_vqe(cluster_ops, hamiltonian_sparse, cluster_ops_sparse, reference_ket, h_sp, cluster_ops_jw,
    hf_init_sp, n_max_grads, fci, optimizer,  tolerance,type_conver = type_conver, threshold_needed =
    threshold_needed, max_external_iterations = max_iterations)
```

`fermionic_adapt_vqe()` calls internally other functions allowing the execution of the steps from 1th to 4th given in Section 2.2: (1) prepare the trial state through `prepare_state()`; (2) compute analytically the commutator between the Hamiltonian and the fermionic operator through `compute_gradient()` (or numerically by using the command `hamiltonian_jw | cluster_ops_sp`); (3) collect and arrange the gradients in a list from maximum to minimum (with avoiding the zeros) through `sorted_gradient()`; (4) if the norm is less than the (ε), the program exits returning the generated values, else it will continue calculating the maximum gradient(s) of operator(s) depending on the number of maximum gradient in the input, after that, we append the operator(s) associated with their parameter(s) to the left of the previous trial state (step (1)) during which we apply VQE using function `ucc_action()`, in order to optimize the parameter(s) until satisfying the threshold's condition; `fermionic_adapt_vqe()` returns at the end: (i) the number of classical parameters for the final ansatz, (ii) the number of CNOT gates in the ansatz circuit, (iii) number of other gates, (iv) list of optimized energies corresponding to external iterations and (v) energy subtracted from the FCI. The qubit-ADAPT sub-module is globally similar to the fermionic-ADAPT in terms of the code structure, some key steps are different and makes it unique in its nature. Those steps can be summarized in

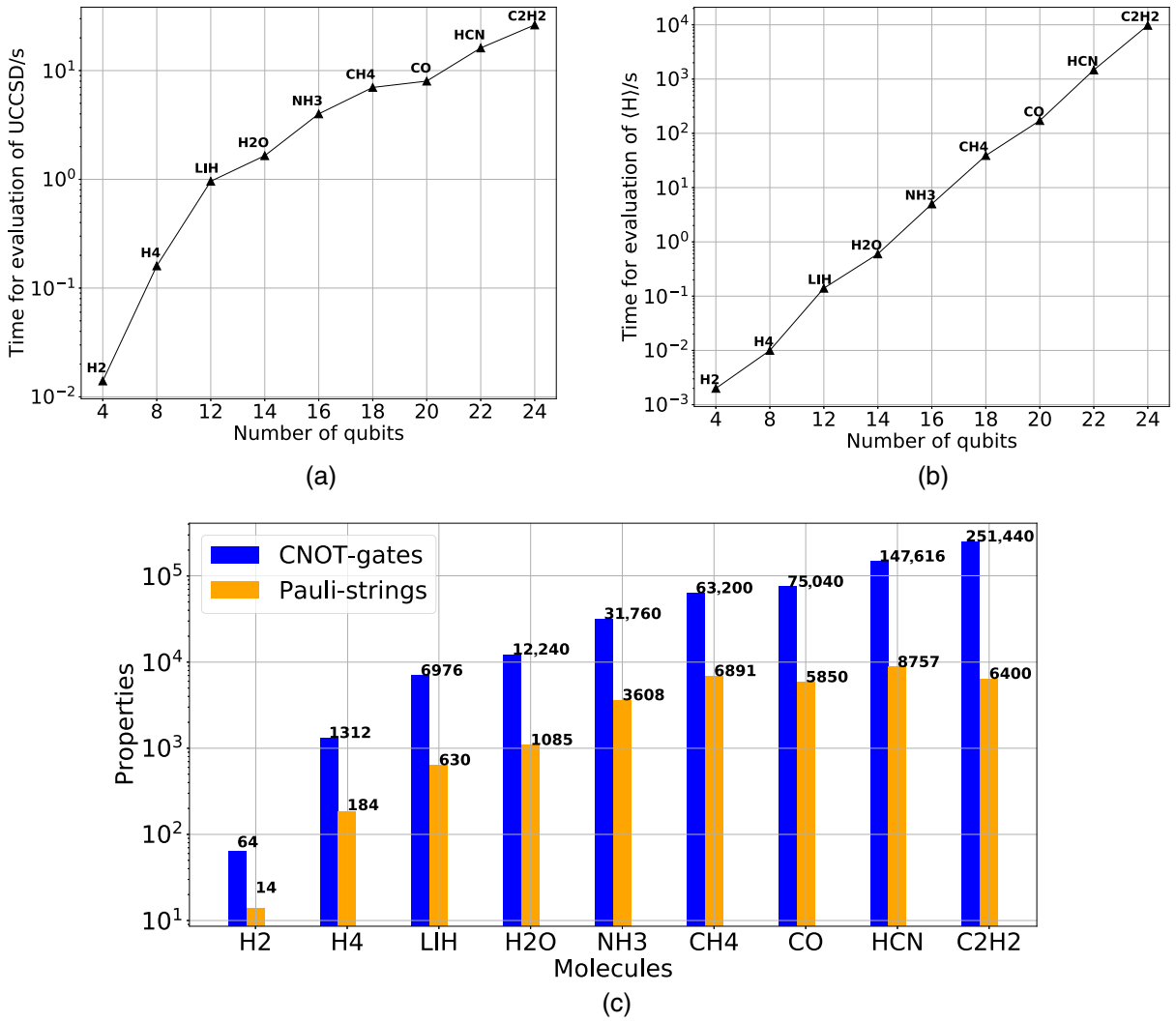


FIGURE 4 (a) Timings for the application of a UCCSD circuit by OpenVQE while increasing the number of qubits. (b) Timings for the evaluation of increasing size molecular Hamiltonians (summed over Pauli strings and using JW transformation) for the following molecules: H_2, H_4, LiH, H_2O, NH_3, CH_4, CO, HCN, C_2H_2. (c) The number of CNOT gates required to complete the UCCSD circuit and the number of Pauli strings in the Hamiltonian.

the following sequence: (i) we use qubit pool generators using `QubitPool` class instead of fermionic ones; (ii) the preparation of the trial state is different from the fermionic-ADAPT one; (iii) the gradients calculation is not the same. It returns the same properties as that of fermionic-ADAPT-VQE.

5 | SYSTEMATIC BENCHMARKING OF THE PERFORMANCES OF THE UCC AND ADAPT-VQE ALGORITHMS ON SEVERAL MOLECULES USING THE OpenVQE PACKAGE

In this section, we used OpenVQE to perform numerical estimations of ground state energies of various molecules using the VQE-UCC-family and ADAPT-VQE modules. The studied molecules require a number of qubits ranging from 4 to 24 (see Figure 4). We perform only noiseless simulations in order to minimize computational runtime. All simulations were performed using TotalEnergies's in-house HPC computing platform.* We validated our results by comparing some of them with other works obtained with similar methodologies. For each numerical simulation in the UCC-family or ADAPT-VQE modules, we use the following common functions implemented in OpenVQE to calculate: (i) the molecular orbital integrals using the PYSCF package[92] with the STO-3G, 6-31G and cc-pVDZ

basis sets; (ii) the molecular Hamiltonian mapping, obtained by applying the Jordan-Wigner transformation; (iii) the ansatz circuits, based on the CNOT staircase method, except for QUCCSD version for which we use the two circuits representing the single and double qubit evolutions (see figs. 1 and 2 in reference 70, respectively). For optimization, we used both optimzers BFGS and SLSQP from the *scipy.optimize* library.[91] We also set the commonly accepted threshold for chemical accuracy, which is 1 kcal/mol (i.e., 1.59×10^{-3} Hartees [Ha]). This accuracy represents the error difference between the predicted energy and FCI energy. We use this value as a standard reference in all our results.

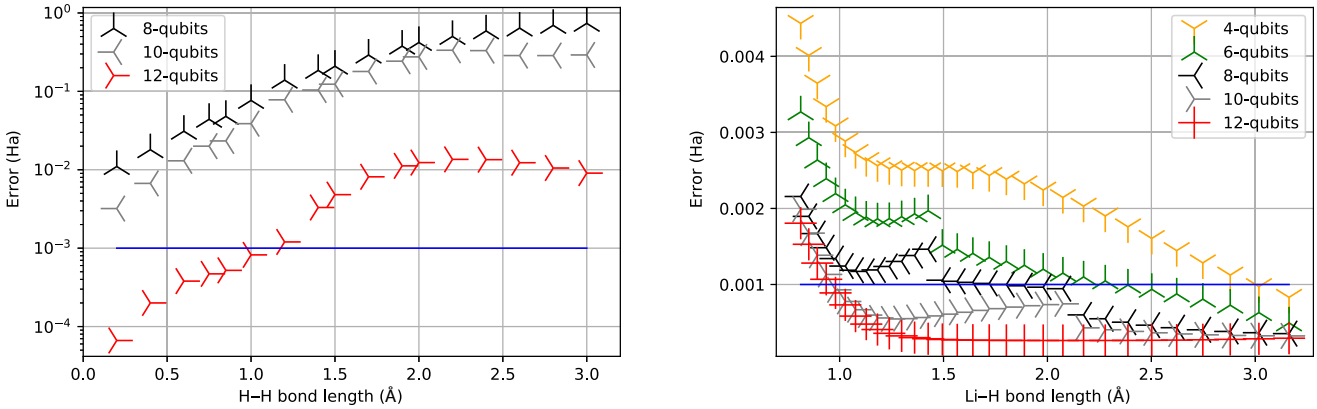
In order to show how openVQE performs, we split our results into four main subsections: (i) Section 5.1 displays the timings of the QLM simulator for constructing the UCCSD ansatz on a simulator as well as for obtaining the expectation value of the electronic structure Hamiltonian for a set of molecules (STO-3G basis set) ranging from 4 to 24 qubits; (ii) Section 5.2, dedicated to the UCC-family module, describes the performance of UCCSD-VQE for H_6 with the STO-3G basis set and for LiH using 6-31G basis set, with different sizes of active spaces, using Møller-Plesset wavefunction in second order (MP2 pre-screening) as initial guess; (iii) Section 5.3 uses the ADAPT-VQE module. It describes the comparison between the fermionic and qubit-ADAPT-VQEs, in terms of the number of operators as well as the chemical accuracy, by benchmarking the three molecules H_4, LiH and H_6, in the STO-3G orbital basis set, by assuming a full space selection approximation in each molecule. (iv) Section 5.4 introduces a brief table summarizing the comparison of UCC-family methods involving fermionic excitations (UCCSD and UCCGSD) with the fermionic-ADAPT-VQE approach. Several molecules are considered and the results are discussed in terms of number of parameters, CNOT counts, chemical accuracy, and computational time.

In addition to the results presented below in this section, there are also additional interesting results obtained using the OpenVQE UCC family module. Indeed, we simulated several other test molecules, using a larger number of qubits and different chemical basis sets assuming both full and active space selections. Such computations were performed in order to demonstrate further the capabilities of our module to perform large calculations and to reach chemical accuracy despite the molecular size increase. These results are detailed in Section S2 of the Supplementary Materials.[81]

5.1 | The standard UCCSD ansatz: Representative computational performances with OpenVQE

To illustrate the performance of our OpenVQE package on CPUs, we chose to use a standard laptop† to evaluate the time to solution required to perform two operations that are common to most of the variational algorithms: (i) the application of the ansatz; (ii) the computation of the expectation value of a molecular Hamiltonian or another observable. Examples of such operations are: the implementation of a hardware efficient ansatz, or a unitary coupled cluster ansatz. Observables are for example the electronic structure Hamiltonian, a commutator operator, Pauli string terms, and so forth, which can be applied with either of these ansätze to measure their expectation value. As a test, we chose here the UCCSD method to evaluate the timing associated to various computations. It is important to note that, in practice, for the particular case of the simple UCCSD, OpenVQE and the native QLM (QLM simulator version: 1.2.1) timings are identical. To measure such timings, we conducted benchmarks with a set of molecules using OpenVQE and the STO-3G basis set (H_2, H_4, LiH, H_2O, NH_3, CH_4, CO, HCN, and C_2H_2).

Figure 4a displays the timings associated to the generation of a state-vector corresponding to the UCCSD ansatz (see Equation 7) from OpenVQE. The computational cost for this generation grows sub-exponentially with the number of qubits. We observe that the package can compute the UCCSD wavefunction within less than a second for a molecule requiring 4 to 12 qubits, between a second and 10 s for molecules associated to a 14–20 qubits range and up to a few minutes for larger molecules (22–24 qubits). Figure 4b represents the timings required to measure the expectation value of the JW transformed Hamiltonian (see Equation 14) of each of these molecules using UCCSD ansatz. This measurement increases linearly with the number of qubits and is performed using the UCCSD circuit together with the "Job" class which will be submitted to QPU to obtain the expectation value. The process is described by the code function called `UCC_action()` which can be found in the UCC module openVQE.[53] We observe that the computational cost of performing the measurement increases exponentially with the number of qubits. In particular, we notice that molecular systems up to 14 qubits like H_2O (10 electrons) using OpenVQE, can be performed within a second for a UCCSD



(a) Errors for the H6 molecule ground state

(b) Errors for the LiH molecule ground state.

FIGURE 5 VQE results for the H_6 molecule using the STO-3G basis set (a) and for LiH with the 6-31G basis (b). The dissociation profile is showed, using different active space selections and are calculated by using the first order trotterization UCCSD ansatz. The size of qubits represents the number of active spin-orbitals. The blue line indicates the chemical accuracy.

circuit. This circuit involves 12,240 CNOT gates and 1083 Pauli strings, as shown in Figure 4c. This figure also shows the details of the number of CNOT gates and Pauli strings associated to each molecule.

5.2 | UCCSD: Active space selections and MP2 pre-screening

We test the active space (AS) selection approach by simulating H_6 linear geometry type within STO-3G basis set and LiH within 6-31g basis set, for a range of bond lengths for both molecules. We use MP2 pre-screening as initial guesses in the BFGS optimizer. For the H_6 molecule, which has a Hilbert space spanned by 12 Hartree–Fock orbitals, 6 occupied and 6 virtual (i.e., unoccupied) we choose two different active spaces: (i) First 2 spin orbitals where the lowest energies are always filled and 2 spin orbitals where the highest energies are always empty, as these 4 spin-orbitals are not considered in the active space selection, the system consists of 8 active spin orbitals with 4 active electrons, which means a 8-qubit Hamiltonian; (ii) when we consider that only the first two spin-orbitals are always filled, the system has then 4 active electrons and 10 spin-orbitals corresponding to 10 qubit Hamiltonian. Figure 5a shows the error of ground state energies of the H_6 molecule at different dissociation profiles, with different sizes of the active spaces ranging from a minimum of 8 qubits to the full space of 12 qubits. This error corresponds to absolute value of VQE-UCCSD results substracted from FCI energy. By increasing the active space from 8 to 10 qubits, we observe only little improvement of the accuracy and both sizes are still far from approaching the chemical accuracy (see the blue line which was set at 10^{-3} Ha). However, in the full space case, the VQE results approach the chemical accuracy before a bond length equal to 1.5 Å. Interestingly, for bond length ≥ 1.5 Å, even with full space, the error deviates well beyond the chemical accuracy but it seems to be curving again toward the blue line beyond 3.0 Å. This error means that triple excitations are probably required. Results comparable to our findings for H_6 can be seen in fig. 7 of reference 51, but this work only considered the non-active case. Figure 5b shows the same energy profile but for the LiH molecule. Using the 6-31G basis set, LiH has a Hilbert space spanned by 22 HF orbitals, 4 occupied and 18 virtual (i.e., unoccupied). But five active space selections are chosen, leading to 4, 6, 8, 10 and 12 qubit Hamiltonians. For all choices of active spaces here, the error decreases steadily as the bond length increases. The 10 and 12 qubit Hamiltonians are below the chemical accuracy for bond length > 1.0 Å, while the largest deviation measured is in the intermediate bond lengths range between 1 and 2.4 Å. Interestingly, the 8, 10 and 12 qubit results fall within a narrow range, at bond length before 1.0 and at bond length beyond 2.2 Å. Surprisingly, we noticed some jumps in the accuracy for the three cases 6, 8 and 10 qubits at some bond lengths, for example around 2.2 Å. A similar analysis has been made in reference 58 for two different molecules than ours, by changing the number of qubits in different basis set: (6-31G) in H_2 and (STO-3G) in H_2O. In particular, they observed similar jumps we also observed for H_2O at intermediate range, which as mentioned there it might correspond to geometries close to the so-called Coulson-Fisher point where spin-symmetry breaking can occur.[93]

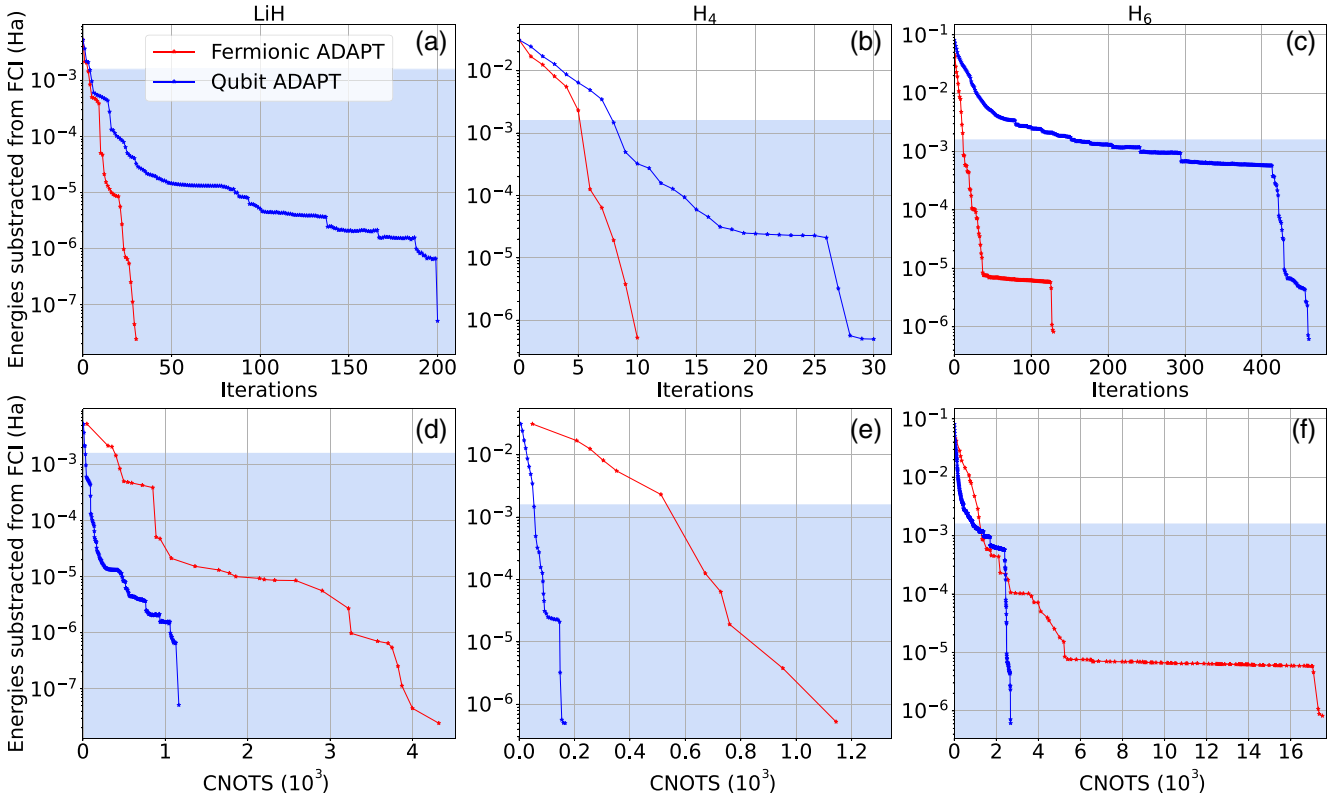


FIGURE 6 Energy convergence plots for the near equilibrium ground states of LiH, H_4 and H_6 using the STO-3G orbital basis set, at bond distances $r_{Li-H} = 1.45$ Å, $r_{H-H} = 0.85$ Å, and $r_{H-H} = 1.0$ Å. The red plots are obtained with a fermionic-ADAPT-VQE algorithm and the blue plots are obtained with qubit-ADAPT-VQE. Top panels (a–c) present the energy accuracy as a function of the ansatz parameters (note that for both ADAPT-VQE, the number of parameters are the same as the number of operators taken per iteration). Bottom panels (d–f) present the energy accuracy as a function of the CNOT count of the ansatz circuit. The blue area corresponds to the chemical accuracy.

5.3 | Performance of ADAPT-VQE module in OpenVQE

In Figure 6, we represent the energy convergence for LiH, H_4 and H_6, which are (8–12) qubit systems, obtained with the fermionic-ADAPT-VQE and qubit-ADAPT-VQE algorithms. The three molecules are at bond lengths $r_{Li-H} = 1.45$ Å, $r_{H-H} = 0.85$ Å, and H_6 at bond distance $r_{H-H} = 1.0$ Å, respectively. All convergence plots are terminated when the norm value of the gradient is below the threshold ε (see Section 2.2), which is between 10^{-3} and 10^{-4} for fermionic-ADAPT-VQE, and between 10^{-3} and 10^{-6} for qubit-ADAPT-VQE, depending on the molecule. Since we find SLSQP and BFGS optimizers behave comparably for energy convergence (after using MP2 initial guess) as is noticed in Figure S3 in the Supplementary Material,[81] we used SLSQP optimizer for fermionic-ADAPT-VQE and the BFGS optimizer for qubit-ADAPT-VQE. The plots in Figure 6 present the two ADAPT algorithms in terms of two important conditions, required to construct an ansatz and to achieve a specific chemical accuracy: (i) the number of iterations (or parameters); (ii) the number of CNOT gates. Figure 6a–c shows that the fermionic-ADAPT-VQE converges faster than the qubit-ADAPT-VQE, requiring systematically fewer number of iterations and thus variational parameters. However as it is shown in Figure 6d–f, the CNOT counts from the fermionic-ADAPT-VQE required to reach convergence are higher by more than one-order of magnitude compared to qubit-ADAPT-VQE. Such results make sense with the fermionic-ADAPT-VQE approach since the type of operators is spin-complemented pairs (see Equations S1 and S2 in the Supplementary Material). Indeed, the JW transformation of single and double operators will lead to sets of two and eight Pauli strings, respectively, where each Pauli string is a tensor product of Pauli matrices. According to theory, there are four CNOT staircases (for a single excitation) and 16 (for a double excitation).[69] The qubit-ADAPT-VQE anszatz (see Section 2.2) states that the pool consists of many individual Pauli strings. This means the qubit-ADAPT-VQE needs more ansatz elements (i.e., operators or parameters) to reach convergence. However, at the same time, the CNOT counts are strongly reduced compared to fermionic-ADAPT-VQE. Similar CNOT counts results, number of parameters and chemical accuracy for fermionic-ADAPT-VQE involving LiH and H_6 molecules are presented in fig.

TABLE 1 LiH (r_{Li-H}) = 1.45 Å molecule at the STO-3G basis set level.

Method	Parameters	CNOT-gates	Energy (Ha)	Error (Ha)	Computational time
UCCSD	44	6080	−7.880973466	8.85×10^{-6}	1 h, 11 min
UCCGSD	330	97,280	−7.880973130	9.18×10^{-6}	1 day, 12 h
Fermionic-ADAPT-VQE	30	4624	−7.880982281	2.43×10^{-8}	2 h, 43 min

Note: Comparison between UCCSD and UCCGSD fixed-length ansätze with fermionic-ADAPT-VQE in terms of number of parameters, CNOT-gates and energies. Accuracy is displayed as the absolute value of the difference between the ansatz Energy and the FCI energy in Hartree (Ha).

TABLE 2 Linear H_6 (r_{H-H}) = 1.0 Å molecule at the STO-3G basis set level.

Method	Parameters	CNOT-gates	Energy (Ha)	Error (Ha)	Computational time
UCCSD	54	8544	−3.235451570	6.14×10^{-4}	3 h, 10 min
UCCGSD	330	97,280	−3.2360518902	1.43×10^{-5}	5 days, 7 h
Fermionic-ADAPT-VQE	130	21,648	−3.236065546	7.33×10^{-7}	7 days, 7 h

Note: Comparison between UCCSD and UCCGSD fixed-length ansätze and fermionic-ADAPT-VQE in terms of number of parameters, CNOT-gates and Energy. Accuracy is displayed as the absolute value of the difference between the ansatz Energy and the FCI energy in Hartree (Ha).

2 of reference 41. For the LiH molecule, we match these results. For H_6 some differences occur since we have converged our computations beyond the 10^{-3} threshold. This added additional parameters to the ansatz, providing therefore a better accuracy. In reference 50 (Figure 1) results of fermionic and qubit-ADAPT-VQE involving LiH ($r = 2.0$ Å), H_4 ($r = 1.5$ Å) and H_6 ($r = 2.0$ Å) are also comparable to our findings despite some small differences are noted due to slightly different geometries in the three molecules, and convergence threshold.

5.4 | Comparison of fixed-length Ansätze with ADAPT-VQE methods

In this subsection, we want to determine how many parameters (i.e., CNOT gates) are needed and which energy accuracy is reached by the "fixed-length ansätze" as opposed to the fermionic-ADAPT-VQE. To do so, we choose the LiH (1.45 Å) and H_6 (1.0 Å) molecules as benchmarks. In Figure 6a,b,d,f, and in figures (S4.a and S4.c in the Supplementary Material[81]), the attempt to reach chemical accuracy is displayed alongside with the number of CNOTs for both the fermionic-ADAPT-VQE and UCCSD, respectively. The UCCGSD (see Equation 9) method remains to be characterized. We do so use the BFGS optimizer in a similar VQE loop. Tables 1 and 2 summarize the results obtained from calculations with each of the methods for LiH and H_6, respectively. We observe from Table 1 that fermionic-ADAPT-VQE brings the most accurate values for the calculated energy by around two orders of magnitude compared with that of UCCSD and UCCGSD. A comparable accuracy between UCCSD and ADAPT-VQE occurs when a norm threshold 10^{-2} is used in ADAPT-VQE (which could be also noticed in fig. 2b of reference 41). This means that as the norm threshold decreases, the ADAPT-VQE approach reaches a better accuracy. Remarkably, the number of parameters/CNOTs (30/4624) is still small compared to UCCSD and UCCGSD even though the considered threshold is 10^{-3}. Results displayed in Table 2 demonstrate again that ADAPT-VQE yields to the most accurate energy for the H_6 molecule, by at least two orders of magnitude as compared to UCCGSD and UCCSD. Although UCCSD requires a fewer number of parameters and CNOTs than with ADAPT-VQE, the latter brings better accuracy. This is again due to the fact that we controlled the norm threshold to 10^{-4}. A comparable accuracy between UCCSD and ADAPT-VQE in H_6 occurs when a norm threshold 10^{-1} is used in ADAPT-VQE as noticed in fig. 2h of reference 41. It shows the fact that, at this level of threshold, ADAPT-VQE decreases the counts of CNOTs and parameters. For UCCGSD, its energy lies between UCCSD and ADAPT-VQE ones, while its number of CNOTs and parameters remain very high.

With those results, one can deduce that ADAPT-VQE brings more accurate results than UCCSD by several order of magnitudes, depending on how the user can control the threshold value, but the cost associated to the use of ADAPT-VQE with and increased threshold is linked to the associated higher number of iterations and CNOTs. These latter increase with the number of qubits, which consequently leads to higher computational requirements in term of gradient computations which could be a practical problem on a real QPU. For UCCGSD (together with UCCSD), the analysis of numerical simulations applied to H_4, H_2O and N_2 were shown in reference 49, illustrating the fact that although the

TABLE 3 Available implementations of individual approaches in a list of most recent open sources quantum chemistry simulation packages (table referred from reference 36) including OpenVQE and myQLM-fermion packages.

Package\ansatz	UCCSD	*k*-UpCCGSD	ADAPT	PNO-UpCCGSD	SPA	QPE
adapt-vqe	×	×	94	×	×	×
TEQUILA	95	96	95	95,97	95,98	×
PennyLane	99,100	101	99,102	×	×	×
OpenFermion	103	×	×	×	×	×
QEBAB	104	104	75,104	×	×	×
QFORTE	105	×	105,106	×	×	105,107
Qiskit	108	×	108	×	×	109
XACC	110	×	×	×	×	×
QDK	111	×	×	×	×	111
InQuanto	112	×	112	×	×	×
OpenVQE/myQLM-fermion	53,54	53	53	×	×	54

Note: PNO-UpCCGSD stands for pair-natural orbitals-UpCCGSD and SPA for Separable Pair Approximation. The × notation means that the method is NOT implemented in a given package.

UCCGSD ansatz yields very accurate results, the problem in using this method is linked to its high circuit depth. This is the opposite to the UCCSD case (or even *k*-UpCCGSD with at least $k > 2$, see reference 49).

6 | CONCLUSION AND FUTURE WORK

In this article, we discussed in detail our newly introduced open source package, OpenVQE. OpenVQE allows one to perform quantum chemistry computations based on the quantum learning machine (QLM) thanks to the newly introduced myQLM-fermion open-source library that gather the key QLM resources that are important for quantum chemical developments. The OpenVQE/myQLM-fermion combined quantum libraries facilitate the implementation, testing and overall development of variational quantum algorithms dedicated to quantum chemistry problems and provide state-of-the-art quantum computing methods (see Table 3 which summarizes the GitHub links of the open-source quantum chemistry packages available to the community). We have shown that OpenVQE enables users to construct modules in a few lines of Python code using simple class structures related to the adaptive UCC family ansätze that we reviewed in the article. We also presented how to easily implement efficient circuits related to fermionic and qubit evolutions with the goal of reducing the number of CNOT gates. Using these modules, we generated benchmarks on a range of molecular computations from 4 to 22 qubits and obtained reliable results that appeared consistent with those previously obtained in reference literature (see references 35,71,113,114 for the UCC and references 41,50,52,71 for the ADAPT-VQE approaches). It is worth noting that technically the QLM machine is designed to simulate up to 41 Qubits on a large memory classical computer systems. OpenVQE is thus designed to simplify the researchers' work, providing simple code modules facilitating the construction of new algorithms and ansätze.

Furthermore, OpenVQE benefits from the myQLM-fermion tools and allow to exchange/interact with other software, like quantum chemistry-oriented quantum computing packages (see Figure S1 in the Supplementary Material[81]), thus enhancing the community's capacity to work collectively and faster. We also intend to forge an efficient module that could merge the few keys tools required to run jobs on quantum computers (IBM, Rigetti, etc.). For example, Figure S2 shows examples of simple code structure classes that involve the interoperability between different open source packages and QPUs.

We plan to further extend OpenVQE/myQLM-fermion tools toward reducing the quantum resources by building new modules describing functions that enable: (i) to taper off the number of qubits in VQE (alongside active space selection) through using penalty functions,[115] using point-group symmetry in small molecules[116] and in large molecules[117] and studying new functions that could preserve spatial symmetry of molecules (rotational and translational); (ii) to reduce the number of CNOT gates (alongside with MP2 guesses, efficient circuits, fermionic and qubit-ADAPT-VQE) through implementing other adaptive algorithms such as permute qubit-VQE.[118]

Concerning chemical accuracy, we want to implement further methods to go beyond the chemically inspired UCC ansätze described in reference 36 (i.e., the implemented UCCSD, UCCGSD, spin-complement-gsd, *k*-UpCCGSD, etc.). Equally we are interested in other types of UCC ansätze that involve higher-order correlation effects, such as MP3 or MP4, or triple excitation UCCSD(T), which could be important for strongly correlated systems and could further improve the accuracy as mentioned earlier.[49] We are also motivated to implement additional algorithms going beyond VQE, such as VQD (ADAPT-VQD) and QITE (ADAPT-QITE) that allow to determine excited state energies.[74,75,119–121]

Concerning optimization effects in the UCC family of methods, since we have demonstrated in the present work that some convergence problems might appear in ADAPT-VQE when targeting certain levels of accuracy, we want to make use of more efficient optimizer methods to overcome this problem. Of course, the practical use of VQE algorithms on present quantum computers will require to make Darwinian choices in link with qubit/operator counts and circuits depths restrictions: not all algorithms will "make the cut" to the real quantum computing world. We believe that the present OpenVQE/myQLM-Fermion approach will help the community to study and design new accurate VQE algorithmic "champions" able to efficiently run on present and near-future NISQ quantum computers toward performing accurate quantum chemical computations on complex molecular systems.

AUTHOR CONTRIBUTIONS

Mohammad Haidar: Conceptualization (equal); methodology (lead); software (lead); validation (equal); writing – original draft (lead); writing – review and editing (equal). **Marko J. Rančić:** Conceptualization (equal); methodology (equal); resources (lead); software (supporting); supervision (equal); validation (equal); writing – original draft (supporting); writing – review and editing (equal). **Thomas Ayral:** Conceptualization (equal); methodology (equal); software (lead); supervision (equal); validation (equal); writing – original draft (equal); writing – review and editing (equal). **Yvon Maday:** Conceptualization (equal); formal analysis (equal); methodology (equal); software (supporting); supervision (equal); validation (equal); writing – original draft (supporting); writing – review and editing (equal). **Jean-Philip Piquemal:** Conceptualization (equal); methodology (equal); resources (equal); software (supporting); supervision (lead); validation (equal); writing – original draft (equal); writing – review and editing (lead).

ACKNOWLEDGMENTS

Marko J. Rančić is grateful to the European Union for the Horizon 2020 (H2020) research grant within the ⟨NE|AS|QC⟩ project (grant agreement No. 951821). This work has also received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 810367), project EMC2 (Jean-Philip Piquemal, Yvon Maday). Support from the PEPR EPiQ and HQI programs is acknowledged. We would like to thank TotalEnergies for providing HPC ressources. Special thanks also to Diata Traore (LCT) for discussions on the classical quantum chemistry computations; to Baptiste Anselme Martin (TotalEnergies) and Cesar Feniou (LCT/Qubit Pharmaceuticals) for helpful quantum computing discussions.

CONFLICT OF INTEREST STATEMENT

The authors have declared no conflicts of interest for this article.

OPEN RESEARCH BADGES

This article has earned an Open Data badge for making publicly available the digitally-shareable data necessary to reproduce the reported results. The data is available at https://github.com/OpenVQE/OpenVQE.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created or analyzed in this study

ORCID

Mohammad Haidar https://orcid.org/0000-0003-2216-0702
Jean-Philip Piquemal https://orcid.org/0000-0001-6615-9426

RELATED WIREs ARTICLES

Emerging quantum computing algorithms for quantum chemistry

FURTHER READING

For further information, we encourage the reader to visit the documentation of the packages: myQLM[43] and myQLM-fermion[54] and OpenVQE.[53] They cover many subjects discussed here and the tutorials are detailed step-by-step for the user.

ENDNOTES

* The server we used for our calculations has the following properties: it is Linux machine named "skelling," with 162 cores, each core containing two threads and a total memory of 3094000 MiB.

† Laptop properties are: Intel(R) i5-10310U, Clock speed 2.21 GHz processor, OP windows 10.

REFERENCES

1. Aspuru-Guzik A, Dutoi AD, Love PJ, Head-Gordon M. Simulated quantum computation of molecular energies. Science. 2005; 309(5741):1704–7.
2. Bassman L, Urbanek M, Metcalf M, Carter J, Kemper AF, Jong dWA. Simulating quantum materials with digital quantum computers. Quantum Sci Technol. 2021;6(4):043002.
3. Lehtola S, Tubman NM, Whaley KB, Head-Gordon M. Cluster decomposition of full configuration interaction wave functions: a tool for chemical interpretation of systems with strong correlation. J Chem Phys. 2017;147(15):154105.
4. Gan Z, Grant DJ, Harrison RJ, Dixon DA. The lowest energy states of the group-IIIA–group-VA heteronuclear diatomics: BN, BP, AlN, and AlP from full configuration interaction calculations. J Chem Phys. 2006;125(12):124311.
5. Vogiatzis KD, Ma D, Olsen J, Gagliardi L, De Jong WA. Pushing configuration-interaction to the limit: towards massively parallel MCSCF calculations. J Chem Phys. 2017;147(18):184111.
6. Hohenberg P, Kohn W. Inhomogeneous electron gas. Phys Rev. 1964;136(3B):B864–71.
7. Kohn W, Sham LJ. Self-consistent equations including exchange and correlation effects. Phys Rev. 1965;140(4A):A1133–8.
8. Schütt O, VandeVondele J. Machine learning adaptive basis sets for efficient large scale density functional theory simulation. J Chem Theory Comput. 2018;14(8):4168–75.
9. Bartlett RJ, Kucharski SA, Noga J. Alternative coupled-cluster ansätze II. The unitary coupled-cluster method. Chem Phys Lett. 1989; 155(1):133–40.
10. Kutzelnigg W. Error analysis and improvements of coupled-cluster theory. Theor Chim Acta. 1991;80(4):349–86.
11. Harrison RJ. Approximating full configuration interaction with selected configuration interaction and perturbation theory. J Chem Phys. 1991;94(7):5021–31.
12. Olsen J, Jo/rgensen P, Koch H, Balkova A, Bartlett RJ. Full configuration–interaction and state of the art correlation calculations on water in a valence double-zeta basis with polarization functions. J Chem Phys. 1996;104(20):8007–15.
13. Peris G, Planelles J, Malrieu JP, Paldus J. Perturbatively selected CI as an optimal source for externally corrected CCSD. J Chem Phys. 1999;110(24):11708–16.
14. Taube AG, Bartlett RJ. New perspectives on unitary coupled-cluster theory. Int J Quantum Chem. 2006;106(15):3393–401.
15. Bartlett RJ, Musiał M. Coupled-cluster theory in quantum chemistry. Rev Mod Phys. 2007;79(1):291–352.
16. Schriber JB, Evangelista FA. Communication: an adaptive configuration interaction approach for strongly correlated electrons with tunable accuracy. J Chem Phys. 2016;144(16):161106.
17. Nagy PR, Kállay M. Approaching the basis set limit of CCSD (T) energies for large molecules with local natural orbital coupled-cluster methods. J Chem Theory Comput. 2019;15(10):5275–98.
18. Feynman RP. Simulating physics with computers. Boca Raton, Florida, USA: CRC Press; 2018. p. 133–53.
19. Benioff P. The computer as a physical system: a microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. J Stat Phys. 1980;22(5):563–91.
20. Reiher M, Wiebe N, Svore KM, Wecker D, Troyer M. Elucidating reaction mechanisms on quantum computers. Proc Natl Acad Sci. 2017;114(29):7555–60.
21. McArdle S, Endo S, Aspuru-Guzik A, Benjamin SC, Yuan X. Quantum computational chemistry. Rev Mod Phys. 2020;92(1): 015003.
22. Cao Y, Romero J, Olson JP, Degroote M, Johnson PD, Kieferová M, et al. Quantum chemistry in the age of quantum computing. Chem Rev. 2019;119(19):10856–915.
23. Bauer B, Bravyi S, Motta M, Chan GKL. Quantum algorithms for quantum chemistry and quantum materials science. Chem Rev. 2020; 120(22):12685–717.
24. Yeter-Aydeniz K, Gard BT, Jakowski J, Majumder S, Barron GS, Siopsis G, et al. Benchmarking quantum chemistry computations with variational, imaginary time evolution, and Krylov space solver algorithms. Adv Quantum Technol. 2021;4(7):2100012.
25. Whitfield JD, Biamonte J, Aspuru-Guzik A. Simulation of electronic structure Hamiltonians using quantum computers. Mol Phys. 2011;109(5):735–50.
26. Preskill J. Quantum computing in the NISQ era and beyond. Quantum. 2018;2:79.

27. Elfving VE, Broer BW, Webber M, Gavartin J, Halls MD, Lorton KP, et al. How will quantum computers provide an industrially relevant computational advantage in quantum chemistry?. arXiv Preprint arXiv:2009.12472 2020.
28. Bharti K, Cervera-Lierta A, Kyaw TH, Haug T, Alperin-Lea S, Anand A, et al. Noisy intermediate-scale quantum algorithms. Rev Mod Phys. 2022;94(1):015004.
29. Peruzzo A, McClean J, Shadbolt P, Yung MH, Zhou XQ, Love PJ, et al. A variational eigenvalue solver on a photonic quantum processor. Nat Commun. 2014;5(1):1–7.
30. McClean JR, Romero J, Babbush R, Aspuru-Guzik A. The theory of variational hybrid quantum-classical algorithms. New J Phys. 2016;18(2):023023.
31. Cerezo M, Arrasmith A, Babbush R, Benjamin SC, Endo S, Fujii K, et al. Variational quantum algorithms. Nat Rev Phys. 2021;3(9):625–44.
32. Arute F, Arya K, Babbush R, Bacon D, Bardin JC, Barends R, et al. Hartree–Fock on a superconducting qubit quantum computer. Science. 2020;369(6507):1084–9.
33. Kandala A, Mezzacapo A, Temme K, Takita M, Brink M, Chow JM, et al. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature. 2017;549(7671):242–6.
34. Hempel C, Maier C, Romero J, McClean J, Monz T, Shen H, et al. Quantum chemistry calculations on a trapped-ion quantum simulator. Phys Rev X. 2018;8(3):031022.
35. Romero J, Babbush R, McClean JR, Hempel C, Love PJ, Aspuru-Guzik A. Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. Quantum Sci Technol. 2018;4(1):014008.
36. Anand A, Schleich P, Alperin-Lea S, WK Jensen P, Sim S, Diaz-Tinoco M, et al. A quantum computing view on unitary coupled cluster theory. Chem Soc Rev. 2022;51:1659–1684.
37. Gard BT, Zhu L, Barron GS, Mayhall NJ, Economou SE, Barnes E. Efficient symmetry-preserving state preparation circuits for the variational quantum eigensolver algorithm. Npj Quantum Inf. 2020;6(1):1–9.
38. Ryabinkin IG, Yen TC, Genin SN, Izmaylov AF. Qubit coupled cluster method: a systematic approach to quantum chemistry on a quantum computer. J Chem Theory Comput. 2018;14(12):6317–26.
39. Kim IH, Swingle B. Robust entanglement renormalization on a noisy quantum computer; 2017. arXiv Preprint arXiv:1711.07500.
40. Wecker D, Hastings MB, Troyer M. Progress towards practical quantum variational algorithms. Phys Rev A. 2015;92(4):042303.
41. Grimsley HR, Economou SE, Barnes E, Mayhall NJ. An adaptive variational algorithm for exact molecular simulations on a quantum computer. Nat Commun. 2019;10(1):1–9.
42. McClean JR, Kimchi-Schwartz ME, Carter J, De Jong WA. Hybrid quantum-classical hierarchy for mitigation of decoherence and determination of excited states. Phys Rev A. 2017;95(4):042308.
43. myQLM package [cited 22 Aug 2022]. Available from: https://myqlm.github.io
44. Interoperability with myQLM [cited 22 Aug 2022]. Available from: https://myqlm.github.io/myqlm_specific/interoperability.html
45. Wille R, Van Meter R, Naveh Y. IBM's Qiskit tool chain: working with and developing for real quantum computers. Piscataway, New Jersey, United States: IEEE; 2019. p. 1234–40.
46. Developers C. Cirq. See full list of authors on Github; [cited 22 Aug 2022]. Available from: https://github.com/quantumlib/Cirq/graphs/contributors
47. Smith RS, Curtis MJ, Zeng WJ. A practical quantum instruction set architecture; 2016. arXiv Preprint arXiv:1608.03355.
48. Nooijen M. Can the eigenstates of a many-body Hamiltonian be represented exactly using a general two-body cluster expansion? Phys Rev Lett. 2000;84(10):2108–11.
49. Lee J, Huggins WJ, Head-Gordon M, Whaley KB. Generalized unitary coupled cluster wave functions for quantum computation. J Chem Theory Comput. 2018;15(1):311–24.
50. Tang HL, Shkolnikov V, Barron GS, Grimsley HR, Mayhall NJ, Barnes E, et al. Qubit-adapt-vqe: an adaptive algorithm for constructing hardware-efficient ansätze on a quantum processor. PRX Quantum. 2021;2(2):020310.
51. Xia R, Kais S. Qubit coupled cluster singles and doubles variational quantum eigensolver ansatz for electronic structure calculations. Quantum Sci Technol. 2020;6(1):015001.
52. Shkolnikov V, Mayhall NJ, Economou SE, Barnes E. Avoiding symmetry roadblocks and minimizing the measurement overhead of adaptive variational quantum eigensolvers; 2021. arXiv Preprint arXiv:2109.05340.
53. Open-VQE package [cited 22 Aug 2022]. Available from: Repository: https://github.com/OpenVQE/OpenVQE.git; and documentation: https://openvqe.github.io/OpenVQE/
54. myqlm-fermion [cited 22 Aug 2022]. Available from: Repository: https://github.com/myQLM/myqlm-fermion; and documentation: https://myqlm.github.io/qat-fermion.html
55. Čížek J. On the correlation problem in atomic and molecular systems. Calculation of wavefunction components in Ursell-type expansion using quantum-field theoretical methods. J Chem Phys. 1966;45(11):4256–66.
56. Cizek J, Paldus J. Coupled cluster approach. Physica Scripta. 1980;21(3–4):251–4.
57. Yung MH, Casanova J, Mezzacapo A, McClean J, Lamata L, Aspuru-Guzik A, et al. From transistor to trapped-ion computers for quantum chemistry. Sci Rep. 2014;4(1):1–7.
58. Barkoutsos PK, Gonthier JF, Sokolov I, Moll N, Salis G, Fuhrer A, et al. Quantum algorithms for electronic structure calculations: particle-hole Hamiltonian and optimized wave-function expansions. Phys Rev A. 2018;98(2):022322.

59. Ganzhorn M, Egger DJ, Barkoutsos P, Ollitrault P, Salis G, Moll N, et al. Gate-efficient simulation of molecular eigenstates on a quantum computer. Phys Rev Appl. 2019;11(4):044092.
60. Sennane W, Piquemal JP, Rančić MJ. Calculating the ground state energy of benzene under spatial deformations with noisy quantum computing. Phys Rev A. 2023;107:012416.
61. McClean JR, Boixo S, Smelyanskiy VN, Babbush R, Neven H. Barren plateaus in quantum neural network training landscapes. Nat Commun. 2018;9(1):1–6.
62. Cerezo M, Sone A, Volkoff T, Cincio L, Coles PJ. Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nat Commun. 2021;12(1):1–12.
63. Shen Y, Zhang X, Zhang S, Zhang JN, Yung MH, Kim K. Quantum implementation of the unitary coupled cluster for simulating molecular electronic structure. Phys Rev A. 2017;95(2):020501.
64. Fradkin E. Jordan-Wigner transformation for quantum-spin systems in two dimensions and fractional statistics. Phys Rev Lett. 1989; 63(3):322–5.
65. Evangelista FA, Chan GKL, Scuseria GE. Exact parameterization of fermionic wave functions via unitary coupled cluster theory. J Chem Phys. 2019;151(24):244112.
66. Grimsley HR, Claudino D, Economou SE, Barnes E, Mayhall NJ. Is the trotterized uccsd ansatz chemically well-defined? J Chem Theory Comput. 2019;16(1):1–6.
67. Hatano N, Suzuki M. Finding exponential product formulas of higher orders. Berlin-Heidelberg: Springer; 2005. p. 37–68.
68. Yordanov YS, Barnes CH. Implementation of a general single-qubit positive operator-valued measure on a circuit-based quantum computer. Phys Rev A. 2019;100(6):062317.
69. Yordanov YS, Arvidsson-Shukur DR, Barnes CH. Efficient quantum circuits for quantum computational chemistry. Phys Rev A. 2020; 102(6):062612.
70. Yordanov YS, Armaos V, Barnes CH, Arvidsson-Shukur DR. Qubit-excitation-based adaptive variational quantum eigensolver. Commun Phys. 2021;4(1):1–11.
71. Yordanov Y. Quantum computational chemistry methods for early-stage quantum computers. PhD thesis. Cambridge, UK: University of Cambridge; 2021.
72. Huggins WJ, Lee J, Baek U, O'Gorman B, Whaley KB. A non-orthogonal variational quantum eigensolver. New J Phys. 2020;22(7): 073009.
73. Cao C, Hu J, Zhang W, Xu X, Chen D, Yu F, et al. Progress toward larger molecular simulation on a quantum computer: simulating a system with up to 28 qubits accelerated by point-group symmetry. Phys Rev A. 2022;105(6):062452.
74. Greene-Diniz G, Muñoz RD. Generalized unitary coupled cluster excitations for multireference molecular states optimized by the variational quantum eigensolver. Int J Quantum Chem. 2021;121(4):e26352.
75. Chan HHS, Fitzpatrick N, Segarra-Martí J, Bearpark MJ, Tew DP. Molecular excited state calculations with adaptive wavefunctions on a quantum eigensolver emulation: reducing circuit depth and separating spin states. Phys Chem Chem Phys. 2021;23(46): 26438–50.
76. Rattew AG, Hu S, Pistoia M, CFR C, Wood S. A domain-agnostic, noise-resistant evolutionary variational quantum eigensolver for hardware-efficient optimization in the Hilbert space. DeepAI. 2019. arXiv Preprint arXiv:1910.09694.
77. Ryabinkin IG, Lang RA, Genin SN, Izmaylov AF. Iterative qubit coupled cluster approach with efficient screening of generators. J Chem Theory Comput. 2020;16(2):1055–63.
78. Lang RA, Ryabinkin IG, Izmaylov AF. Unitary transformation of the electronic Hamiltonian with an exact quadratic truncation of the Baker-Campbell-Hausdorff expansion. J Chem Theory Comput. 2020;17(1):66–78.
79. Sim S, Romero J, Gonthier JF, Kunitsa AA. Adaptive pruning-based optimization of parameterized quantum circuits. Quantum Sci Technol. 2021;6(2):025019.
80. Liu J, Li Z, Yang J. An efficient adaptive variational quantum solver of the Schrödinger equation based on reduced density matrices. J Chem Phys. 2021;154(24):244112.
81. Haidar M, Rančić MJ, Ayral T, Maday Y, Piquemal JP. See Supplemental Material for openVQE package: interoperability discussion and numerical results.
82. Quantum Learning Machine [cited 22 Aug 2022]. Available from: https://atos.net/en/solutions/quantum-learning-machine
83. Martiel S, Brugière dTG. Architecture aware compilation of quantum circuits via lazy synthesis. Quantum. 2022;6:729. https://doi.org/10.22331/q-2022-06-07-729
84. De Brugière TG, Baboulin M, Valiron B, Martiel S, Allouche C. Gaussian elimination versus greedy methods for the synthesis of linear reversible circuits. ACM Transactions on Quantum Computing. 2021;2(3):26. https://doi.org/10.1145/3474226
85. Vandaele V, Martiel S, Brugière GT. Phase polynomials synthesis algorithms for NISQ architectures and beyond. Quantum Sci Technol. 2022;7:045027.
86. TGd B, Baboulin M, Valiron B, Martiel S, Allouche C. Reducing the depth of linear reversible quantum circuits. IEEE Trans Quantum Eng. 2021;2:1–22. https://doi.org/10.1109/TQE.2021.3091648
87. Brugière dTG, Baboulin M, Valiron B, Martiel S, Allouche C. Decoding techniques applied to the compilation of CNOT circuits for NISQ architectures. ScienceDirect. 2022;214(February):1–31. https://doi.org/10.1016/j.scico.2021.102726
88. Vidal G. Efficient classical simulation of slightly entangled quantum computations. Phys Rev Lett. 2003;91:147902. https://doi.org/10.1103/PhysRevLett.91.147902

89. Rudiak-Gould B. The sum-over-histories formulation of quantum computing; 2006. arXiv Preprint arXiv:Quant-Ph/0607151.
90. Miller DM, Thornton MA. QMDD: a decision diagram structure for reversible and quantum circuits. Piscataway, New Jersey, United States: IEEE; 2006. p. 30.
91. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods. 2020;17:261–72. https://doi.org/10.1038/s41592-019-0686-2
92. Sun Q, Berkelbach TC, Blunt NS, Booth GH, Guo S, Li Z, et al. PySCF: the python-based simulations of chemistry framework. WIREs Comput Mol Sci. 2018;8(1):e1340.
93. Gunnarsson O, Lundqvist BI. Exchange and correlation in atoms, molecules, and solids by the spin-density-functional formalism. Phys Rev B. 1976;13(10):4274–98.
94. ADAPT package [cited 22 Aug 2022]. Available from: https://github.com/mayhallgroup/adapt-vqe.git
95. Kottmann JS, Alperin-Lea S, Tamayo-Mendoza T, Cervera-Lierta A, Lavigne C, Yen TC, et al. Tequila: a platform for rapid development of quantum algorithms. Quantum Sci Technol. 2021;6(2):024009.
96. Pair-natural orbitals in tequila package [cited 22 Aug 2022]. Available from: https://github.com/aspuru-guzik-group/tequila
97. Kottmann JS, Schleich P, Tamayo-Mendoza T, Aspuru-Guzik A. Reducing qubit requirements while maintaining numeri-cal precision for the variational quantum eigensolver: a basis-set-free approach. J Phys Chem Lett. 2021;12(1):663–73.
98. Kottmann JS, Aspuru-Guzik A. Optimized low-depth quantum circuits for molecular electronic structure using a separablepair approximation. Phys Rev A. 2022;105(3):032449.
99. Bergholm V, Izaac J, Schuld M, Gogolin C, Ahmed S, Ajith V, et al. Pennylane: automatic differentiation of hybrid quantum-classical computations; 2018. arXiv Preprint arXiv:1811.04968.
100. Arrazola JM, Di Matteo O, Quesada N, Jahangiri S, Delgado A, Killoran N. Universal quantum circuits for quantum chemistry; 2021. arXiv Preprint arXiv:2106.13839.
101. PennyLaneAI package [cited 22 Aug 2022]. Available from: https://github.com/PennyLaneAI/pennylane.git
102. Delgado A, Arrazola JM, Jahangiri S, Niu Z, Izaac J, Roberts C, et al. Variational quantum algorithm for molecular geometry optimization. Phys Rev A. 2021;104(5):052402.
103. McClean JR, Sung KJ, Kivlichan ID, Cao Y, Dai C, Fried ES, et al. OpenFermion: the electronic structure package for quantum computers. Quantum Sci Technol. 2020;5(3):034014.
104. QEBAB package [cited 22 Aug 2022]. Available from: https://github.com/hanschanhs/QEBAB.git
105. Stair NH, Evangelista FA. Qforte: an efficient state simulator and quantum algorithms library for molecular electronic structure; 2021. arXiv Preprint arXiv:2108.04413.
106. Stair NH, Evangelista FA. Simulating many-body systems with a projective quantum eigensolver. PRX Quantum. 2021;2(3):030301.
107. Stair NH, Huang R, Evangelista FA. A multireference quantum krylov algorithm for strongly correlated electrons. J Chem Theory Comput. 2020;16(4):2236–45.
108. Aleksandrowicz G, Alexander T, Barkoutsos P, Bello L, Ben-Haim Y, Bucher D, et al. Qiskit: an open-source framework for quantum computing; 2019. https://doi.org/10.5281/zenodo.2562111
109. Quantum phase estimation algorithm in Qiskit package [cited 22 Aug 2022]. Available from: https://qiskit.org/textbook/ch-algorithms/quantum-phase-estimation.html
110. McCaskey AJ, Lyakh DI, Dumitrescu EF, Powers SS, Humble TS. XACC: a system-level software infrastructure for heterogeneous quantum–classical computing. Quantum Sci Technol. 2020;5(2):024002.
111. Quantum Development Kit [cited 22 Aug 2022]. Available from: https://github.com/microsoft/quantum.git
112. InQuanto tket package [cited 22 Aug 2022]. Available from: https://github.com/CQCL/tket.git
113. Harsha G, Shiozaki T, Scuseria GE. On the difference between variational and unitary coupled cluster theories. J Chem Phys. 2018;148(4):044107.
114. Kühn M, Zanker S, Deglmann P, Marthaler M, Weiß H. Accuracy and resource estimations for quantum chemistry on a near-term quantum computer. J Chem Theory Comput. 2019;15(9):4764–80.
115. Kuroiwa K, Nakagawa YO. Penalty methods for a variational quantum eigensolver. Phys Rev Res. 2021;3(1):013197.
116. Setia K, Chen R, Rice JE, Mezzacapo A, Pistoia M, Whitfield JD. Reducing qubit requirements for quantum simulations using molecular point group symmetries. J Chem Theory Comput. 2020;16(10):6091–7.
117. Cao C, Hu J, Zhang W, Xu X, Chen D, Yu F, et al. Towards a larger molecular simulation on the Quantum computer: up to 28 qubits systems accelerated by point group symmetry; 2021. arXiv Preprint arXiv:2109.02110.
118. Tkachenko NV, Sud J, Zhang Y, Tretiak S, Anisimov PM, Arrasmith AT, et al. Correlation-informed permutation of qubits for reducing ansatz depth in the variational quantum eigensolver. PRX Quantum. 2021;2(2):020337.
119. Ollitrault PJ, Kandala A, Chen CF, Barkoutsos PK, Mezzacapo A, Pistoia M, et al. Quantum equation of motion for computing molecular excitation energies on a noisy quantum processor. Phys Rev Res. 2020;2(4):043140.
120. Ville JL, Morvan A, Hashim A, Naik RK, Lu M, Mitchell B, et al. Leveraging randomized compiling for the QITE algorithm; 2021. arXiv Preprint arXiv:2104.08785.
121. Kamakari H, Sun SN, Motta M, Minnich AJ. Digital quantum simulation of open quantum systems using quantum imaginary–time evolution. PRX Quantum. 2022;3(1):010320.

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

How to cite this article: Haidar M, Rančić MJ, Ayral T, Maday Y, Piquemal J-P. Open source variational quantum eigensolver extension of the quantum learning machine for quantum chemistry. WIREs Comput Mol Sci. 2023;13(5):e1664. https://doi.org/10.1002/wcms.1664