

ARTICLE OPEN



Towards practical and massively parallel quantum computing emulation for quantum chemistry

Honghui Shang^{1,6}, Yi Fan^{2,6}, Li Shen², Chu Guo³, Jie Liu², Xiaohui Duan⁴, Fang Li⁵ and Zhenyu Li²

Quantum computing is moving beyond its early stage and seeking for commercial applications in chemical and biomedical sciences. In the current noisy intermediate-scale quantum computing era, the quantum resource is too scarce to support these explorations. Therefore, it is valuable to emulate quantum computing on classical computers for developing quantum algorithms and validating quantum hardware. However, existing simulators mostly suffer from the memory bottleneck so developing the approaches for large-scale quantum chemistry calculations remains challenging. Here we demonstrate a high-performance and massively parallel variational quantum eigensolver (VQE) simulator based on matrix product states, combined with embedding theory for solving large-scale quantum computing emulation for quantum chemistry on HPC platforms. We apply this method to study the torsional barrier of ethane and the quantification of the protein–ligand interactions. Our largest simulation reaches 1000 qubits, and a performance of 216.9 PFLOP/s is achieved on a new Sunway supercomputer, which sets the state-of-the-art for quantum computing emulation for quantum chemistry.

npj Quantum Information (2023)9:33 | <https://doi.org/10.1038/s41534-023-00696-7>

INTRODUCTION

Computation is revolutionizing chemistry and materials science. Computing the electronic structure by approximately solving the Schrödinger equation enables us to explore chemicals and materials at the atomic scale. However, the pursuit for chemical accuracy in numerical simulations of quantum many-body systems is a longstanding problem since the computational complexity grows exponentially with the system size. For example, even with the help of supercomputers, the exact solution of the Schrödinger equation is limited to a complete active space problem of (24 electrons, 24 orbitals), which corresponds to a diagonalization problem of size 7.3 trillion¹. Richard Feynman suggested quantum computing as a potential solution for simulating quantum systems, as he marked ‘if you want to make a simulation of nature, you’d better make it quantum mechanical’².

Significant advances in quantum computing technologies over the past two decades are turning Feynman’s vision into reality. As a milestone, quantum advantage in the random circuit sampling (RCS) problem has been demonstrated on noisy intermediate-scale quantum (NISQ) computers^{3–5}. Toward practical applications, the ground-state energies of diamonds have been estimated with the quantum Monte Carlo (QMC) method using 16 qubits and 65 circuit depths, which is the largest quantum chemistry calculation using a quantum computer⁶. However, the quantum resource used in this experiment is far away from that required to realize the quantum advantage in quantum chemistry, which is expected to appear at around 38 to 68 qubits (under the assumption of error-corrected qubits)⁷. Besides, the variational quantum eigensolver (VQE) is an appealing candidate for solving quantum chemistry problems on NISQ devices⁸, which has great flexibility in choosing quantum circuit ansatzes and mitigating errors⁹.

However, compared to the RCS and QMC experiments, the VQE simulations with tens of qubits would be significantly more challenging for quantum hardware in that: (1) the circuit depth scales quickly up to 10^3 or even more as the number of qubits increases¹⁰ and (2) the nonlinear optimization with a large number of parameters remarkably increases the computational cost. As such, the largest VQE experiment performed on a quantum computer has only used 12 qubits¹¹, and the current VQE emulation with classical simulators is also mostly limited to relatively small molecules with 10–20 qubits, as shown in Table 1 for the typical simulations of chemical and material systems using classical simulators.

To explore practical applications of quantum computing in quantum chemistry, one can resort to the development of quantum technologies, e.g. advanced quantum algorithms in combination with error mitigation techniques or fault-tolerant quantum computers as a long-term target. Another way is the combination of state-of-the-art simulators with high-performance computing (HPC), which enable us to emulate large-scale quantum computation of the electronic structure on classical computers. In the current stage, simulators are expected to play a fundamental role in algorithm design or verification. In the RCS experiments, classical simulators are used for both calibrating the fidelity of individual gate operation and the whole random quantum circuit and extrapolating the fidelity of simpler quantum circuits to the most difficult ones^{3–5}. In most quantum algorithm designs, simulators are employed as the numerical emulating platform to benchmark new algorithms.

Classical simulators suffer from the notorious exponential wall when the many-body systems are simulated exactly. As such, approximation algorithms are often used to realize large-scale emulations of quantum chemistry calculation. For example, the

¹Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. ²Hefei National Laboratory, University of Science and Technology of China, Hefei, China. ³Key Laboratory of Low-Dimensional Quantum Structures and Quantum Control of Ministry of Education, Department of Physics and Synergetic Innovation Center for Quantum Effects and Applications, Hunan Normal University, Changsha, China. ⁴School of Software, Shandong University, Jinan, China. ⁵National Research Center of Parallel Computer Engineering and Technology, Beijing, China. ⁶These authors contributed equally: Honghui Shang, Yi Fan. [✉]email: shanghui.ustc@gmail.com; guochu604b@gmail.com; liujie86@ustc.edu.cn

Table 1. Typical simulations of molecular and material systems with classical simulators.

Work	System	N_a	N_q	N_{CNOT}	Reference
Microsoft QDK	H ₂	2	4	696	41
Cirq	CH ₂ O	4	6	1.8×10^3	42
Qulacs	He crystal	1	8	1.6×10^3	43
Qiskit	N ₂	2	16	1.9×10^4	44
Yao.jl	C ₁₈	18	16	5.4×10^4	14
VQChem	H chain	2	16	5.4×10^4	45
QCQC	Si crystal	2	16	1.1×10^5	46
Tequila	BH	2	22	6.2×10^3	47
HiQ	C ₂ H ₄	6	28	1.2×10^5	13
iQCC-VQE	Ir ^{III} complexes	~60	72	~96	12
MPS-VQE	H ₂	2	92	1.4×10^5	This work
MPS-VQE	C ₂ H ₆	8	32	4.4×10^5	
MPS-VQE (one shot)	H ₂ chain	500	1000	1.0×10^6	
DMET-MPS-VQE	Atazanavir	103	16	1.8×10^6	

Number of atoms (N_a), number of qubits (N_q), and the estimated number of CNOT gates (N_{CNOT}) are listed for comparison.

excited states of iridium complexes have been computed with up to 72 qubits¹², which is the largest classical emulation of the VQE in terms of the number of qubits up to date. However, to achieve such a large emulation scale, a very shallow quantum circuit ansatz was employed to reduce the computational cost. Additionally, a 28-qubit VQE emulation of the C₂H₄ molecule has been reported by using point symmetry to significantly reduce the total number of gate operations¹³. A classical emulation of the C₁₈ molecule (a model system consisting of 144 spin molecular orbitals and 72 electrons) has been reported by combining VQE with the density matrix embedding theory (DMET), where DMET is used to break the molecule into small fragments and the VQE is used as the solver for the electronic structure of each fragment. While the maximum number of qubits used in the VQE calculations is only 16¹⁴.

In this work, we demonstrate a high-performance and massively parallel VQE simulator using the matrix product state (MPS) representation of the quantum state, as illustrated in Fig. 1. Our simulator maximally utilizes the power of tensor network methods and supercomputers in order to overcome the exponential memory bottleneck and realize the largest classical emulation of quantum computational chemistry. The major computational bottleneck of the MPS-VQE algorithm (see the section “MPS algorithm for quantum circuit simulation” for more details) on HPC is the implementation of high-level linear algebra solvers, such as singular value decomposition (SVD) (see the section “SVD and Jacobi-based method”). Here, we overcome this bottleneck with the optimized SVD and tensor operation algorithm. As discussed in the section “Speedup and scaling with MPS-VQE simulator”, our one-sided Jacobi SVD is more than 60 times faster than the non-optimized version on average for matrix sizes from 100 to 500. As a result, our largest simulation which uses the MPS-VQE simulator scales up to 1000 qubits for one-shot energy evaluation and to 92 qubits for fully converged VQE emulation, with a two-qubit gate count of up to 10^5 . In combination with DMET (see section “The DMET method” for more details), our simulator is applied to study practical quantum chemistry systems containing 103 atoms and achieves comparable accuracy with state-of-the-art computational methods.

RESULTS

Optimization strategies

Emulating quantum computing on a classical computer is difficult due to the exponential runtime and memory requirement. Such difficulties can be leveraged with tensor network methods and by utilizing many-core and multi-node computers. Heterogeneous many-core systems are efficient for handling runtime issues but have limited total accessible memory space. Meanwhile, the memory of a multi-node computer can be scaled to the petabytes order, but its bandwidth for access from host computers (CPUs) is narrow. To simultaneously accelerate simulations and enlarge the total memory space, the heterogeneous parallelization approach¹⁵ (see sections “Heterogeneous parallelization strategy” and section “Julia programming language” for more details) can be adopted. Our simulator allocates memory to each computation node and then accelerates simulations by utilizing the full capabilities of the heterogeneous many-core processors.

The new-generation Sunway supercomputer that is the successor of the Sunway TaihuLight supercomputer is used for performance assessment in this work. Similar to the Sunway TaihuLight system, the new Sunway supercomputer adopts a new generation of domestic high-performance heterogeneous many-core processors (SW26010Pro) and interconnection network chips in China. The architecture of the SW26010Pro processor is shown in Fig. 2a. Each processor contains 6 core groups (CGs), with 65 cores in each CG, making a total number of 390 cores. Each CG contains one management processing element (MPE), one cluster of computing processing elements (CPEs), and one memory controller. Each CPE has a 32 KB L1 instruction cache, and a 256 kB scratch pad memory (SPM, also called the Local Data Memory (LDM)), which serves the same function as the L1 cache. Data transfer between LDM and main memory can be realized by direct memory access (DMA).

The hotspots of our simulator are mainly the tensor contractions and SVD functions. In the tensor contraction, the first step is the index permutation of the tensors, followed by one of the BLAS (basic linear algebra subprograms)¹⁶ routine that performs matrix–matrix multiplications (ZGEMM) to accomplish the calculation. Here we use the fused permutation and multiplication technique¹⁷. For the ZGEMM calculation, we perform matrix–matrix multiplications based on the optimization strategies, including a balanced block that we choose optimized block for the matrix A and B to make balanced computations with CPEs, and diagonal broadcasting method where we use CPEs on the diagonal to perform a broadcast to forward its data to its corresponding row or column, to realize efficient parallel computing for matrix multiplications, matrix transpose multiplications and conjugate transpose multiplications on the Sunway many-core system. First, we need to decompose the matrices A and B into smaller blocks to fit into the computing size of the kernel. Second, we transmit the blocks of the input matrix into the LDM from the main memory. If we need to permute the input matrix, we should load the data that need to be transposed to the LDM of each CPE in blocks by DMA_get, and the data stored on its own LDM using the single instruction multiple data (SIMD) ‘vshuff’ instruction (the interface of the shuffle between two vectors); A diagonal broadcast optimization method is used to greatly reduce the memory access overhead to ensure the overall performance of matrix multiplication. Third, SIMD is used to implement eight 64-bit double-precision floating-point operations at a time. One SIMD instruction is equivalent to a small loop, so the number of instructions can be reduced, thereby reducing the requirement for bandwidth, and reducing the number of loops caused by induced control-related time overhead, as shown in Fig. 2b.

For the SVD calculation, there are mainly two classes of algorithms. The first class of the SVD algorithms is the QR-based two-phase approach¹⁸, in which the matrix A is transformed into a

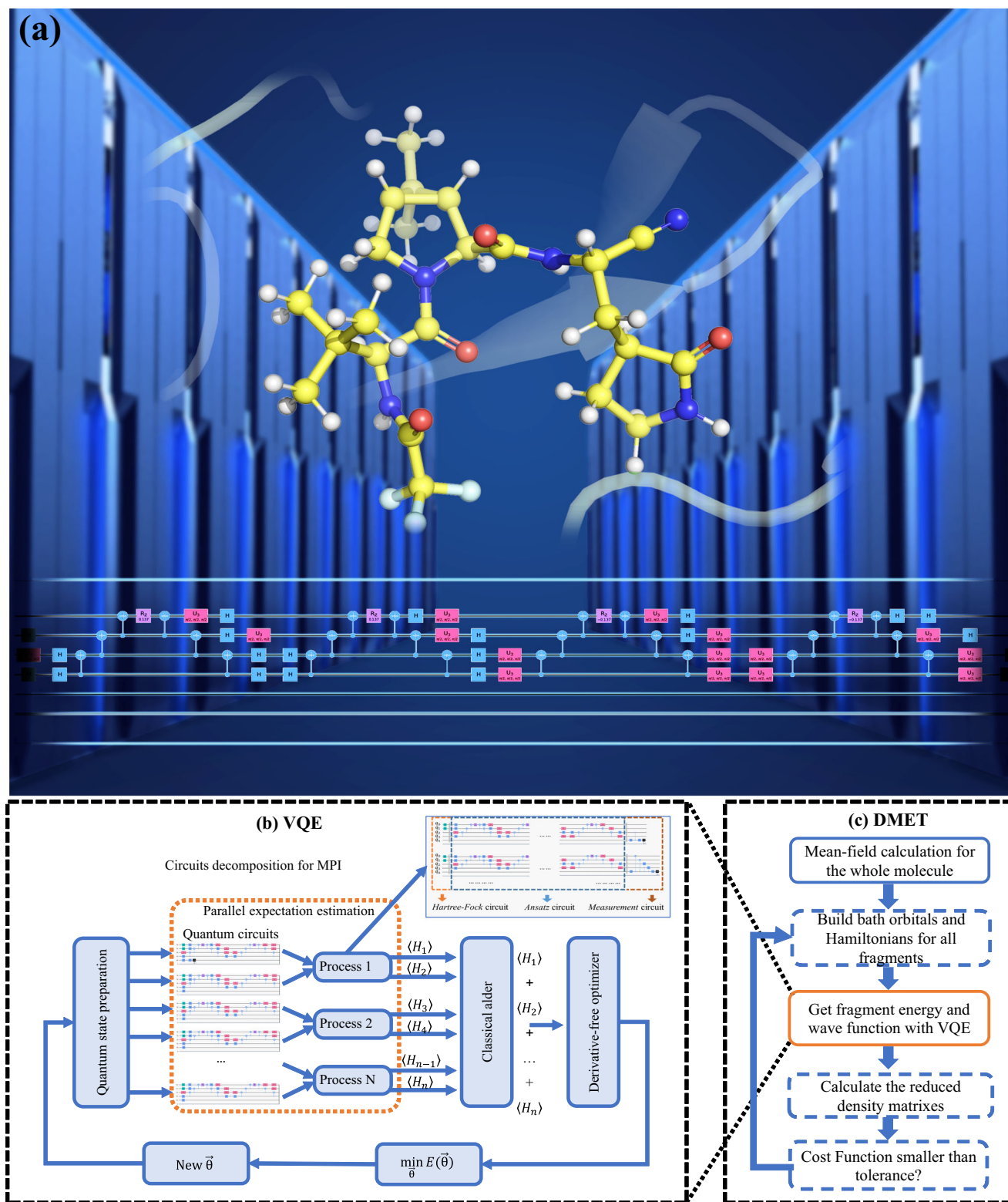


Fig. 1 Framework of our quantum computational chemistry simulator. **a** The conceptual illustration of the quantum computing emulation for quantum chemistry. **b** The VQE simulator using the matrix product states (MPS) representation of the quantum state for each fragment within DMET. **c** The DMET calculation procedures for the realistic chemical systems.

bidiagonal matrix using an orthogonal transformation, and then the bidiagonal matrix is diagonalized using the bidiagonal divide-and-conquer method or the QR algorithm. The complete SVD is then determined during the backward transformation. This

method is efficient for large matrices while suffering from loss of relative accuracy¹⁹. The second class of the SVD algorithms is the Jacobi-based algorithm, which has recently attracted a lot of attention because it has a higher degree of potential

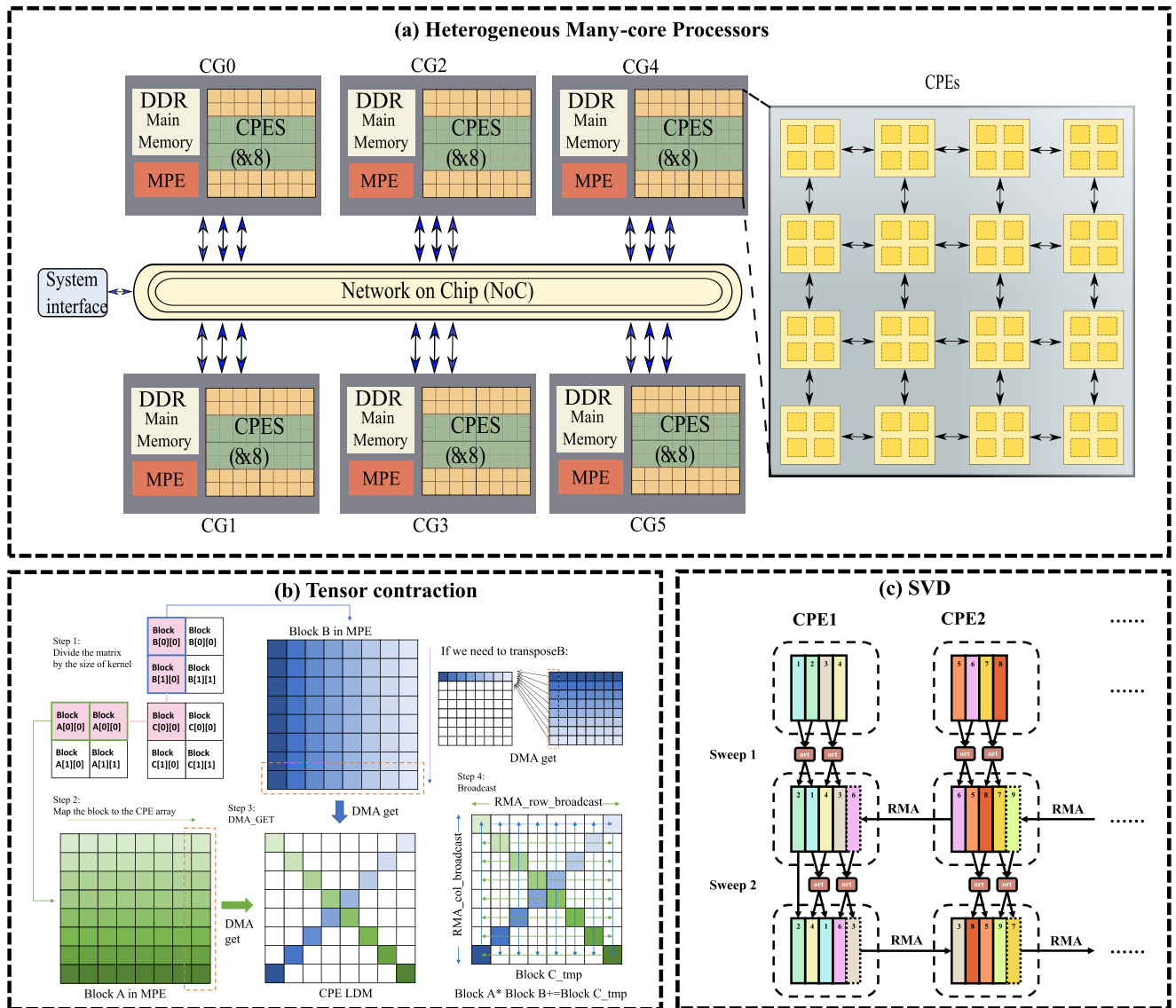


Fig. 2 Algorithm details for linear algebra routines. **a** Architecture of the SW26010Pro processor. **b** Matrix multiplication on the Sunway many-core processor. **c** One-sided Jacobi SVD algorithm on the Sunway many-core processor.

parallelism^{20–22}. There are two varieties of the Jacobi-based algorithm (see section “SVD and Jacobi-based method”), one-sided and two-sided algorithms. The one-sided Jacobi algorithm is computationally more efficient than the two-sided algorithm²³ and suitable for vector pipeline computing. Thus, to achieve efficient parallel SVD computation on Sunway heterogeneous many-core architectures, the best choice is the Hestenes one-sided Jacobi transformation method²⁴, where all pairs of columns are repeatedly orthogonalized in sweeps using Jacobi rotations²⁵ until all columns are mutually orthogonal. When the convergence is reached, the right singular vectors can be computed by accumulating the rotations, the left singular vectors are the normalized columns of the modified matrix, and the singular values are the norms of those columns. Since each pair of columns can be orthogonalized independently, the method is also easy to parallelize over the CPEs, as shown in Fig. 2c. It should be noted that another scalable SVD algorithm called cross-product SVD²⁶ is also widely used in the principal component analysis. However, numerical issues may appear since the condition number is squared in the intermediate step to orthogonalize $A^T A$. To simulate

quantum systems in which the superposition of states is quite arbitrary, the cross-product SVD may be not as stable as other approaches.

Validation results with MPS-VQE simulator (92 qubits)

As a pilot application, Fig. 3 shows the potential energy curves (PECs) of the hydrogen molecule computed with the MPS-VQE simulator. The unitary coupled cluster with single and double excitations (UCCSD) ansatz that is able to accurately describe this two-electron system is employed for single-point energy calculations. The implementation of the UCCSD ansatz with MPS is described in the “Methods” section (see the section “The implementation of UCCSD with matrix produce states” for more details). The STO-3G, cc-pVDZ, cc-pVTZ, and aug-cc-pVTZ basis sets are used to extend these emulations from 4 to 92 qubits. The BOBYQA optimizer is used for the variational optimization, with a convergence threshold set to 10^{-6} for the minimum allowed value of the trust region radius. Note that the hydrogen molecule can be simulated without supercomputer resources even in aug-cc-pVTZ basis since only two electrons are involved. However, this 92-qubit

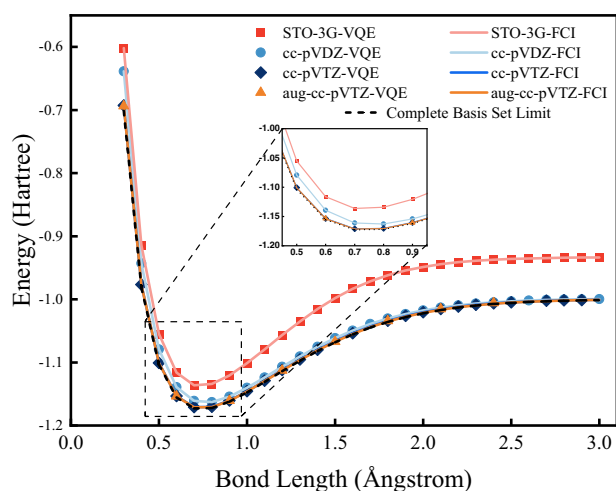


Fig. 3 Potential energy curves in unit Hartree of the hydrogen molecule computed with UCCSD. The basis sets are STO-3G, cc-pVDZ, cc-pVTZ, and aug-cc-pVTZ, which correspond to 4, 20, 56, and 92 qubits, respectively. The results of full configuration interaction calculations at the complete basis set limit are provided for comparison.

Table 2. Wall time per VQE iteration in seconds and number of iterations to converge for different basis sets.

Basis set	STO-3G	cc-pVDZ	cc-pVTZ	aug-cc-pVTZ
Wall time per iteration	0.12	3.67	190.63	1564.52
Number of steps	18	303	459	677

The data are collected from the geometry with the lowest energy of each basis set in Fig. 3.

case involves 1.4×10^5 CNOT gates (161 variational parameters), which is the largest quantum circuit simulation up to date in terms of the number of qubits and circuit depth. The simulations are carried out using 512 processes, and the computation times are given in Table 2. The results from MPS-VQE are in excellent agreement with the full configuration interaction (FCI) results as shown in Table 3. For all four basis sets, chemical accuracy is achieved with a maximum error of $0.82 \text{ kcal mol}^{-1}$ at $R(\text{H-H}) = 2.4 \text{ \AA}$; for the aug-cc-pVTZ results. We also show results obtained with FCI in the complete basis set (CBS) limit, which can be considered as the exact potential energy curve of the hydrogen molecule. The results of aug-cc-pVTZ show an average deviation of $1.42 \text{ kcal mol}^{-1}$ from the complete basis set limit. We can see that using a larger basis set makes the potential energy curve much closer to the exact dissociation limit.

Speedup and scaling with MPS-VQE simulator

One major bottleneck of the MPS-VQE simulator is the SVD function (technical details shown in the section “SVD and Jacobi-based method”), which takes around 85% of the CPU time on average. In Fig. 4, we show the performance improvement of the two optimized versions of SVD, including the QR-based method implemented in SW_xmath (QR_SW_xmath) and the optimized one-sided Jacobi in this work (one-sided-Jacobi_SW), compared to the QR-based SVD method running on MPE (QR_MPE), for different matrix sizes. We use the performance of the QR_MPE as the baseline, which we set as 1 in Fig. 4b. We can see that the optimized SVD using the one-sided Jacobi method produces an overall speedup ranging from $1.5 \times$ to $62.2 \times$ compared to QR_MPE, and achieves a speedup of $2 \times$ to $6 \times$

Table 3. The mean absolute errors (MAE) and maximum absolute errors (MAX) (in kcal/mol) of the potential energy surfaces for H_2 computed with the UCCSD-VQE method using different Gaussian basis sets.

Basis set	STO-3G	cc-pVDZ	cc-pVTZ	aug-cc-pVTZ
MAE	9.4×10^{-13}	2.7×10^{-3}	8.1×10^{-2}	3.3×10^{-1}
MAX	6.3×10^{-12}	1.3×10^{-2}	1.8×10^{-1}	8.2×10^{-1}

The FCI results are taken as the reference values.

compared to QR_SW_xmath version. For the one-sided Jacobi SVD (one-sided-Jacobi_SW), we use the Athread library routines provided by the Sunway architecture for the many-core acceleration, and we use 64 threads for the actual computation. The Jacobi-based method for SVD used in this work has potentially better accuracy than other methods. For example, if the SVD routine in the MPS simulator is replaced with cross-product SVD²⁶, the energy error with respect to FCI will raise from 1.1×10^{-2} to $1.5 \times 10^{-1} \text{ kcal mol}^{-1}$ for the simplest H_2 molecule (cc-pVTZ basis set) even if more than 2.5 times the number of VQE steps are performed.

For the tensor contraction using the optimization method listed in the section “Optimization strategies” (SW_zgemm), we can get an overall speedup of around $1.3 \times$ to $7.2 \times$ compared with the SW_xmath version (a vendor-provided linear algebra library on the Sunway supercomputer), as shown in Fig. 4a.

Figure 4c shows the computational time of the MPS-VQE simulator for implementing the VQE circuits of the hydrogen chain using 512 processes, where the detailed information are given in Table 4. The maximally allowed bond dimension is set to be $D = 128$, as explained in the section “The wave function ansatz for hydrogen chain simulations”. The one-shot energy estimation means that only one step of energy evaluation is performed instead of performing optimization of variational parameters until convergence. In the one-shot energy evaluation, the parameters are set as random numbers in order to keep the bond dimension at the upper limit value ($D = 128$) during the circuit evolution. The number of electrons/atoms ranges from 12 to 500, and the corresponding number of qubits ranges from 24 to 1000. The scaling exponents of the computation time (as a function of the total number of atoms N) for each VQE iteration are fitted by the polynomial scaling formula $t = cN^\alpha$ (α is the exponent). We find the exponent $\alpha \approx 1.6$ for all of the VQE circuits. This is because the number of terms in the Hamiltonian approximately scales as $N^{1.5}$ for the hydrogen chain.

Peak performance with DMET-MPS-VQE

We use the hydrogen chain to assess the scalability and performance of our DMET-MPS-VQE simulator. The wave function ansatz is adaptively built in order to reduce the circuit depth (see section “The wave function ansatz for hydrogen chain simulations” for more details). The system is divided into fragments with the DMET method. A brief introduction of the DMET method used in this work can be found in the section “The DMET method”. We record the computational time with an increasing number of fragments (2048 processes per fragment). The number of floating point operations for tensor contractions is measured by counting all the floating point arithmetic instructions needed for matrix multiplications. For SVD, the number of floating-point operations is measured using the profiler LWPF²⁷ which can monitor the floating-point operation hardware counters in the processor. The quantum circuits containing CNOT gates acting on each pair of neighbouring qubits. This building block serves as the entanglement block in the hardware-efficient ansatz²⁸. Evolving the circuit requires to perform SVDs for $N_q - 3$ matrices of size $2D \times 2D$ and $3 \times (N_q - 3)$ matrix-matrix multiplications. The results are shown in

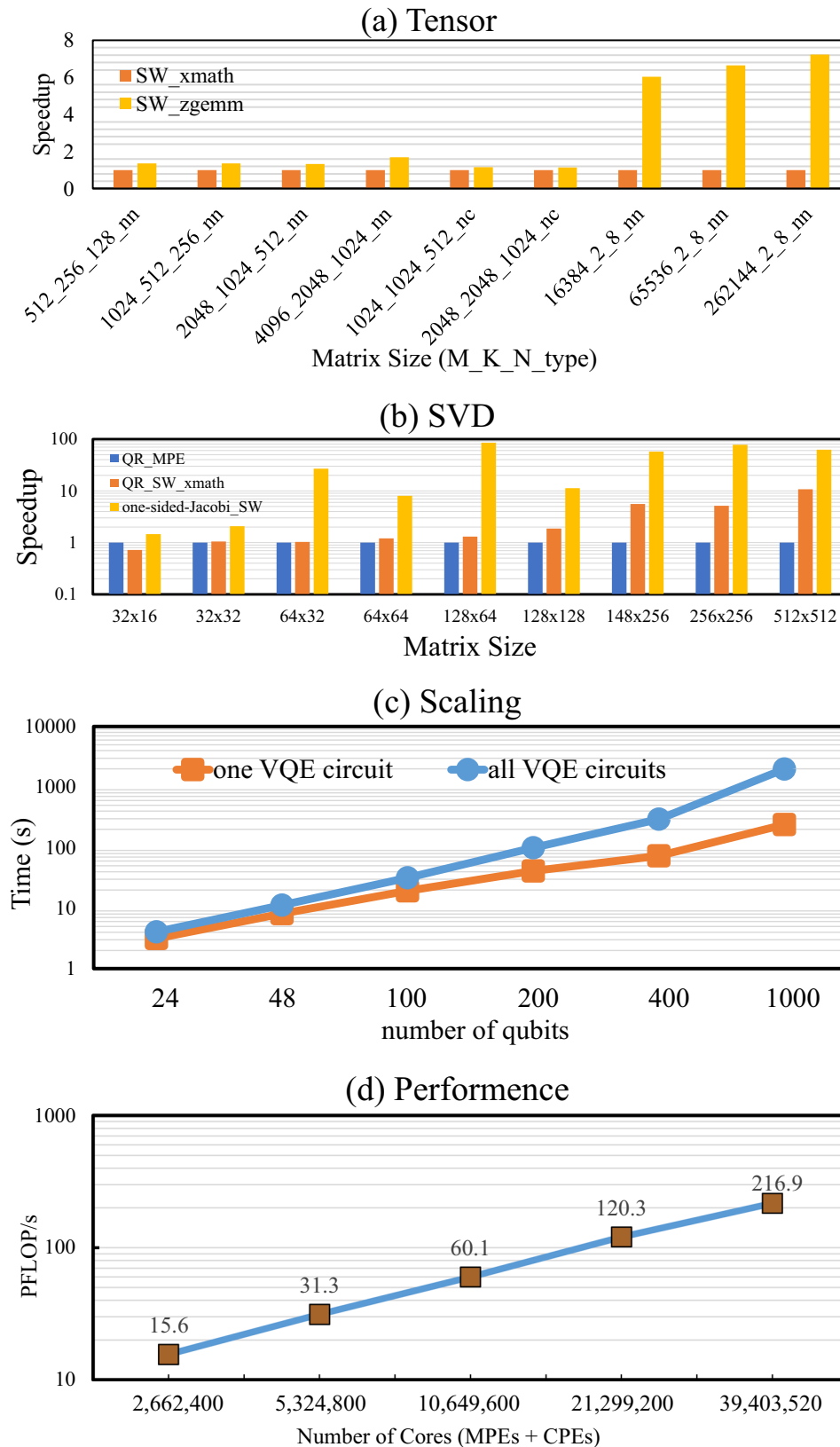


Fig. 4 Performance results of linear algebra routines. **a** The performance comparison for tensor contraction with respect to the matrix size. **b** The performance comparison for SVD with respect to the matrix size, which is evaluated on one CG that contains 1 MPE and 64 CPEs. **c** The computational time of the hydrogen chain with the MPS-VQE simulator. The blue line refers to the computational time for all circuits with 512 processes. **d** The performance (PFLOP/s) and strong scaling of the MPS-VQE simulator integrated with DMET on the new-generation Sunway supercomputer.

Fig. 4d. We can see that a nearly linear scaling is obtained. Sustained performance of 216.9 PFLOPS is achieved in double precision with 606,208 processes (39,403,520 cores) for the system with 2368 qubits.

Implications

In this section, we discuss applications of our MPS-VQE and DMET-MPS-VQE simulators to study realistic chemical systems. One

Table 4. The computational time per VQE iteration using 512 cores for the hydrogen chain with the MPS-VQE simulator (without DMET).

System	N_a	N_q	N_c	Wall time (s)	CPU Time (core · s)
(H ₂) ₃	6	12	1811	1.23	559.64
(H ₂) ₆	12	24	15,905	4.20	2133.08
(H ₂) ₁₂	24	48	60,723	10.67	5443.94
(H ₂) ₂₅	50	100	193,607	29.90	14,923.02
(H ₂) ₅₀	100	200	544,549	86.74	43,520.58
(H ₂) ₁₀₀	200	400	1,426,637	304.25	154,234.77
(H ₂) ₂₅₀	500	1000	5,059,403	1961.03	999,432.92

The number of atoms (N_a), number of qubits (N_q), the estimated number of circuits (N_c) are listed in the table. The bond dimension D is set to be 128.

example is the torsional barrier of ethane, which is one of the most fundamental problems in biomacromolecule configuration analysis. Figure 5 shows the results obtained by the MPS-VQE simulator for the torsional barrier of the ethane molecule. The bond lengths of C–C and C–H are set to be 1.512 and 1.153 Å, respectively. The STO-3G basis set with all 16 orbitals is used (32 qubits). The obtained torsional barrier is 0.29 eV which is higher than the experimental value 0.13 eV. Using the 6-31G(d) basis set will lower the barrier to 0.20 eV even if a small active space of only 6-orbital-6-electron is used. Therefore, It is expected that using a larger basis set could further improve the simulation accuracy.

As an anticipated application, we apply the DMET-MPS-VQE simulator to study the quantification of the protein–ligand interactions, which is a large-scale practical biochemical problem. Compared to classical calculations, quantum mechanical calculations can automatically include the effects of polarization, charge transfer, charge penetration, and the coupling of the various terms, thus offering more accurate and detailed information on the nature of the protein–ligand interactions. This is highly important in high-accuracy binding affinity prediction as well as in drug design. The SARS-CoV-2 is the coronavirus behind the COVID-19 pandemic, and its main protease (M^{pro}) is an enzyme that cleaves the viral polyproteins into individual proteins required for viral replication, so it is important to develop drugs targeting at M^{pro} for SARS-CoV-2. In quantum mechanical studies, the

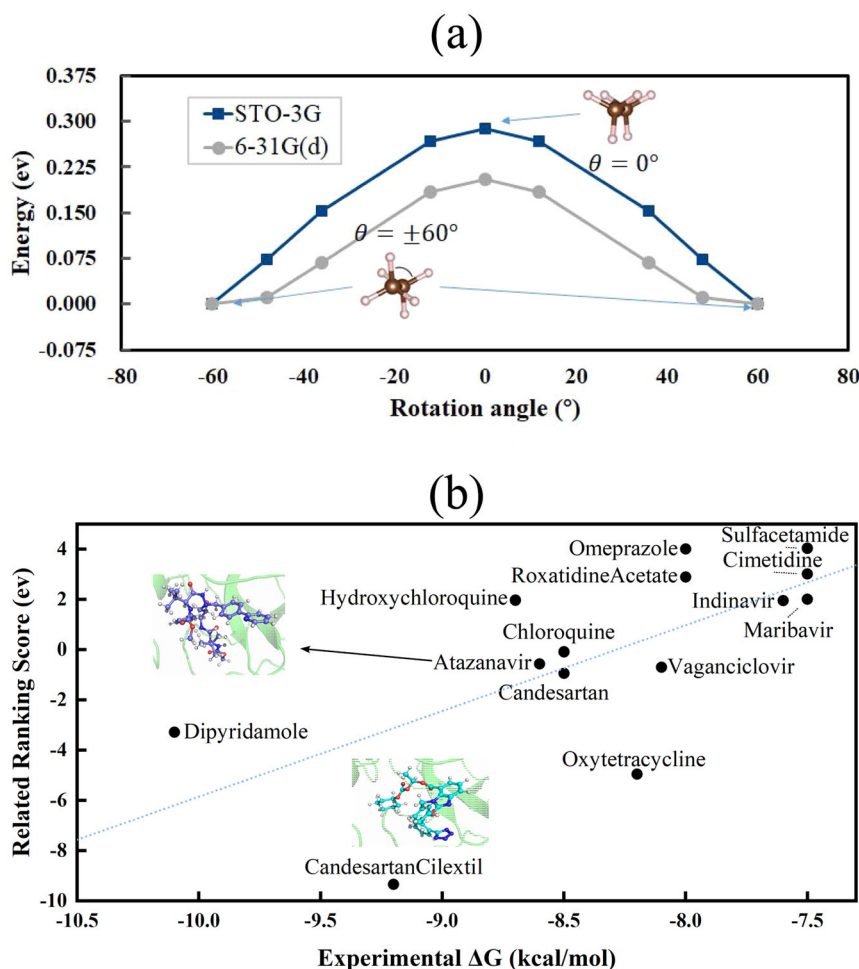


Fig. 5 Simulated results for chemical applications. **a** Torsional barrier of the ethane molecule simulated with MPS-VQE using STO-3G (32 qubits) and 6-31G(d) (12 qubits using a (6e,6o) active space) basis set. **b** Binding energy ranking score versus experimental binding free energies. The overall R^2 value for all points is 0.44. The results are computed with MPS-VQE integrated with DMET.

protein–ligand binding energy is calculated by $E_b = E_{\text{complex}} - E_{\text{protein}} - E_{\text{ligand}}$, where E_{complex} is the energy of the complex, E_{protein} is the energy of the protein and E_{ligand} is the energy of the unbound ligand. The energy of the complex, protein and ligand bounded in the complex are calculated using density functional theory with the PBE+MDB functional to account for many-body van der Waals interactions, which is important to obtain accurate potential-energy surfaces²⁹. After that, the energy differences between bounded and unbounded geometries of ligands are estimated with DMET-VQE³⁰. We use the geometries of the 14 neutral ligands from ref.³¹, and then we optimize the geometries of the ligands at the Hartree–Fock level to account for the geometric distortion needed for the ligand to occupy the active site. Similar to ref.³⁰, we use STO-3G basis set in the DMET-VQE calculation. We plot the ranking score against the experimental binding free energies in a correlational plot as shown in Fig. 4. The ranking score is defined as the difference between the binding energy and the average value of 14 ligands. Ideally, the simulated ranking score should reproduce the experimental trends. We use the coefficients of determination, denoted as R^2 , of the simulated ranking score and the experimentally measured free energy to access the quality of our simulation. It can be seen the correlation between our simulation and the experiment is fairly good, with R^2 of 0.44, which is better than the FEP-based approach (with R^2 of 0.29)³². The dipyrindamole falls off the correlation line, but the fact that candesartan cilexetil binds best to the protein agrees with the experiment. By removing dipyrindamole and hydroxychloroquine from the set, we get an R^2 of 0.59. However, we are fully aware of the necessity to consider the basis set, environment, and temperature effects, as well as DMET subsystem size when applying the DMET-MPS-VQE to drug design in the following studies. The largest molecule we calculated is Atazanavir which contains 103 atoms and 378 electrons, this is the largest system that has been investigated with simulators to our knowledge.

DISCUSSIONS

As a heuristic quantum algorithm, the accuracy and performance of VQE should be verified in practical applications. The problems that VQE aims to solve, namely finding the ground state of a quantum many-body Hamiltonian, have a computational complexity growing exponentially with the problem size in general. Therefore, small-scale simulations for simple molecules using around 20 qubits are hard to demonstrate the powerfulness of VQE in practical applications. In this work, the MPS-VQE simulator scales up to 1000 qubits for one-shot energy evaluation and to 92 qubits for converged VQE emulation, moreover, the DMET-MPS-VQE simulator scales up to 39 million cores on the New Sunway supercomputer. The quantification of the protein–ligand interactions for SARS-CoV-2 is studied with the DMET-MPS-VQE as an application in drug discovery. Particularly, we can obtain decent results using VQE, which are comparable with the experimental observations.

The development of quantum computers requires the intertwining and contribution of classical supercomputers, which enables us to benefit from much more mature classical computing. The simulation scale we have reached in this work, in terms of both the number of qubits and the circuit depths, is far beyond the simulations that have been done in existing literature, and the capability of existing quantum computers. Although we have limited ourselves to the physically motivated UCCSD ansatz, our simulator could also be straightforwardly used with any other circuit ansatz, such as those hardware-efficient ones, which are more friendly to current quantum computers. Our simulator would be an excellent benchmark and validation tool for the development of next-generation quantum computers, as well as a flexible platform for quantum researchers to explore industrially related applications with tens of qubits.

METHODS

Unitary coupled cluster ansatz

The electronic Hamiltonian \hat{H} of a chemical system is written in the second-quantized form as $\hat{H} = \sum_{pq} h_{pq}^p a_p^\dagger a_q + \frac{1}{2} \sum_{pqrs} g_{pqrs}^p a_p^\dagger a_q^\dagger a_r a_s$, where h_{pq}^p and g_{pqrs}^p are one- and two-electron integrals in the Hartree–Fock orbital basis. In the framework of the VQE, the total energy is calculated by measuring the expectation values of the qubit Hamiltonian obtained by Fermion-to-Qubit transformations, such as Jordan–Wigner or Bravyi–Kitaev, of the fermionic Hamiltonian. One of the most widely used wave function ansatz is the unitary coupled cluster⁹ in the form of $|\Psi(\theta)\rangle = e^{\hat{T}(\theta)-\hat{T}^\dagger(\theta)}|\Phi_0\rangle$. Here, $|\Phi_0\rangle$ is the Hartree–Fock state, which can be easily prepared on a quantum computer. When the UCC operator is truncated to the single and double excitations (UCCSD), namely

$$\hat{T}(\theta) = \sum_{ai} \theta_i^a \hat{a}_a^\dagger \hat{a}_i + \frac{1}{4} \sum_{abij} \theta_{ij}^{ab} \hat{a}_a^\dagger \hat{a}_b^\dagger \hat{a}_i \hat{a}_j, \quad (1)$$

where $\{i, j, \dots\}$, $\{a, b, \dots\}$ and $\{p, q, \dots\}$ denote the occupied, virtual, and general spin molecular orbitals, respectively. The UCCSD ansatz does not have an exact finite truncation of the Baker–Campbell–Hausdorff expansion such that an approximation should be introduced in its classical implementation.

The UCCSD ansatz can be implemented on a quantum platform with a parametric quantum circuit generated from Suzuki–Trotter decomposition of the unitary exponential operator into one- and two-qubit gates³³. In such a case, the UCCSD ansatz can be mapped to a W-shaped ansatz circuit with a quartic number of two-qubit gates. For example, restricting to the minimal basis set, the number of CNOT gates of a full UCC circuit reaches 8.6×10^5 for the simple C_2H_4 molecule, which is usually far beyond the capability of current NISQ devices and can hardly be simulated on most of existing quantum circuit simulators.

We note that UCCSD is inadequate for describing many strongly correlated systems. Here, we focus on exhibiting the performance of our simulator. The accuracy of the wave function ansatzes can be improved by introducing adaptive VQE algorithms³⁴.

MPS algorithm for quantum circuit simulation

The correlated wave function in quantum chemistry considering all configuration states can be written as

$$|\Psi\rangle = \sum_{i_1 \dots i_N} c_{i_1 i_2 i_3 \dots i_N} |i_1 i_2 i_3 \dots i_N\rangle \quad (2)$$

where $|i_1 i_2 i_3 \dots i_N\rangle$ refers to the computation basis, $c_{i_1 i_2 i_3 \dots i_N}$ is a rank- N tensor of 2^N complex numbers. This state can be represented with matrix product states (MPS), decompose the correlated wave function into a set of low-rank tensors:

$$c_{i_1 i_2 i_3 \dots i_N} = \sum_{a_0 \dots a_N} B_{a_0 a_1}^{i_1} B_{a_1 a_2}^{i_2} B_{a_2 a_3}^{i_3} \dots B_{a_{N-1} a_N}^{i_N}, \quad (3)$$

where $i_n \in \{0, 1\}$ refers to “physical” indices and a_n the “virtual” index related to the partition entanglement entropy. a_0 and a_N at the boundaries are trivial indices added for notational convenience.

In our MPS simulator, we keep the tensors to be right-canonical, namely the site tensors of the MPS in Eq. (3) satisfy:

$$\sum_{i_n, a_n} \left(B_{a_{n-1} a_n}^{i_n} \right)^* B_{a_{n-1} a_n}^{i_n} = \delta_{a_{n-1}, a'_{n-1}}. \quad (4)$$

A single-qubit gate operation acting on the n th qubit, denoted as $Q_{i_n i'_n}$ can be simply applied onto the MPS as

$$\tilde{B}_{a_{n-1} a_n}^{i_n} = \sum_{i'_n} Q_{i_n i'_n} B_{a_{n-1} a_n}^{i'_n}. \quad (5)$$

The new site tensor $\tilde{B}_{a_{n-1} a_n}^{i_n}$ satisfies Eq. (4) since $Q_{i_n i'_n}$ is unitary and $B_{a_{n-1} a_n}^{i'_n}$ satisfies Eq. (4). For the operation of a two-qubit gate on qubits

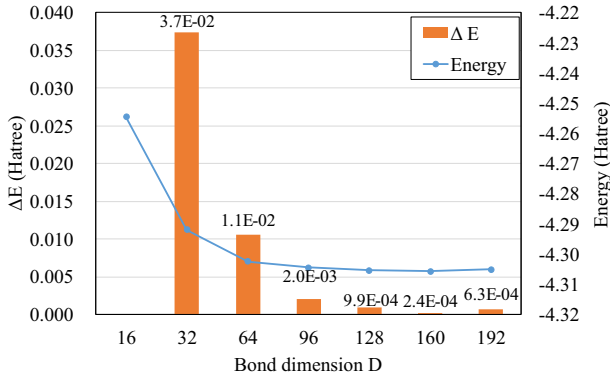


Fig. 6 The MPS-VQE optimized energies of H_8 molecule using different bond dimension settings. The energy different ΔE is calculated by $\Delta E = |E_{D_i} - E_{D_{i+1}}|$, where $D_i \in \{16, 32, 64, 96, 128, 160, 192\}$.

n and $n+1$ (the n th bond), denoted as $Q_{i_n, i_{n+1}}^{j_n, j_{n+1}}$, we use the technique from ref. 35 to keep the underlying MPS in the right-canonical form, which is shown in the following. We first contract the two-site tensors $B_{a_n, a_{n+1}}^{i_n, i_{n+1}}$ and $B_{a_{n-1}, a_n}^{j_{n-1}, j_n}$ with $Q_{i_n, i_{n+1}}^{j_n, j_{n+1}}$ to get a two-site tensor

$$C_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}} = \sum_{a_n, i_n, i_{n+1}} Q_{i_n, i_{n+1}}^{j_n, j_{n+1}} B_{a_n, a_{n+1}}^{i_n, i_{n+1}} B_{a_{n-1}, a_n}^{j_{n-1}, j_n}, \quad (6)$$

then we contract $C_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}}$ with the singular matrix formed by the singular values at the $n-1$ th bond (denoted as $\lambda_{a_{n-1}}$) to get a new two-site tensor as

$$\tilde{C}_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}} = \lambda_{a_{n-1}} C_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}}, \quad (7)$$

We perform singular value decomposition onto the tensor $\tilde{C}_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}}$ and get

$$\text{SVD}(\tilde{C}_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}}) = \sum_{a_n} U_{a_{n-1}, a_n}^{i_n} \tilde{\lambda}_{a_n} V_{a_n, a_{n+1}}^{j_{n+1}}, \quad (8)$$

during which we will also truncate the small singular values below a certain threshold or simply reserve the largest few singular values to control the memory overhead. Finally the new site tensors $\tilde{B}_{a_{n-1}, a_n}^{i_n}$ and $\tilde{B}_{a_n, a_{n+1}}^{j_{n+1}}$ can be obtained as

$$\tilde{B}_{a_{n-1}, a_n}^{i_n} = \sum_{i_{n+1}, a_{n+1}} C_{a_{n-1}, a_{n+1}}^{i_n, i_{n+1}} (V_{a_n, a_{n+1}}^{j_{n+1}})^*; \quad (9)$$

$$\tilde{B}_{a_n, a_{n+1}}^{j_{n+1}} = V_{a_n, a_{n+1}}^{j_{n+1}}, \quad (10)$$

and the new singular values $\tilde{\lambda}_{a_n}$ is used to replace the old λ_{a_n} at the n th bond. Since $\sum_{a_n} \tilde{B}_{a_{n-1}, a_n}^{i_n} \tilde{B}_{a_n, a_{n+1}}^{j_{n+1}} = C_{a_{n-1}, a_{n+1}}^{i_n, j_{n+1}}$, they indeed represent the correct site tensors after the two-qubit gate operation. $\tilde{B}_{a_n, a_{n+1}}^{j_{n+1}}$ is right-canonical by the definition of SVD. Moreover, one can verify that $\tilde{B}_{a_{n-1}, a_n}^{i_n}$ is also right-canonical by substituting Eqs. (7), (8) into Eq. (9):

$$\tilde{B}_{a_{n-1}, a_n}^{i_n} = U_{a_{n-1}, a_n}^{i_n} \tilde{\lambda}_{a_n} / \tilde{\lambda}_{a_{n-1}}, \quad (11)$$

The above equation transforms a left-canonical site tensor $U_{a_{n-1}, a_n}^{i_n}$ into a right-canonical site tensor $\tilde{B}_{a_{n-1}, a_n}^{i_n}$.

The implementation of UCCSD with matrix product states

As discussed in section "Unitary coupled cluster ansatz", the implementation of the UCCSD ansatz in this work includes three steps:

- We perform the Jordan–Wigner transformation of the cluster operator. Here, the Hartree–Fock state is employed as a reference state. The cluster operator is defined as a linear combination of single and double excitations from occupied

orbitals to virtual orbitals (see Eq. (1)).

- We perform a Suzuki–Trotter decomposition of the unitary exponential operator into one- and two-qubit gates. Because the excitation operators are not commutative, we use first-order Trotter decomposition to approximate the UCCSD ansatz as products of exponential operators, which can be further decomposed into products of one- and two-qubit gates.
- We apply these quantum gates to a reference wave function. The intermediate wave functions after applying quantum gates to the initial wave function are represented by matrix product states.

Steps 1 and 2 are done using the Q²Chemistry package³⁶. Step 3 is one of the most important parts of this work. Applying a single qubit gate to an MPS can be done without approximation by multiplying the gate with a single MPS tensor. To apply a two-qubit gate to qubits n and $n+1$, we first perform tensor contractions of the corresponding gates and tensors and then apply the gate to the contracted state. To restore the MPS form, the resulting tensor is decomposed with an SVD truncated to keep the largest X singular values, and the matrix of singular values is multiplied into one of the unitary factors X or Y .

With a right-canonical form of MPS, there is a very efficient way to compute the expectation of a single Pauli string. Taking the expectation value of a single-qubit observable O_{i_n, i_n} as an example, it can be simply computed as

$$\sum_{a_{n-1}, a_n, i_n, i_n'} \lambda_{a_{n-1}}^2 O_{i_n, i_n'} B_{a_{n-1}, a_n}^{i_n, i_n'} (B_{a_{n-1}, a_n}^{i_n, i_n})^*, \quad (12)$$

while a generic two-qubit observable $O_{i_m, i_n}^{j_m, j_n}$ (assuming $m < n$) can be computed as

$$\sum_{a_{n-m-1}, i_n, m, i_n'} \lambda_{a_{n-m-1}}^2 O_{i_m, i_n}^{j_m, j_n} B_{a_{n-m-1}, a_n}^{i_m, i_n'} (B_{a_{n-m-1}, a_n}^{i_m, i_n'})^* \times \dots \times B_{a_{n-1}, a_n}^{i_n, i_n'} (B_{a_{n-1}, a_n}^{i_n, i_n'})^*, \quad (13)$$

where we have used $x_{ji} = \{x_j, x_{j+1}, \dots, x_i\}$ as an abbreviation for a list of indices. The expectation value of a general n -qubit Pauli string could be computed similarly.

The wave function ansatz for hydrogen chain simulations

When hydrogen chains containing hundreds of atoms are studied, it is impossible to implement a full UCCSD ansatz even with a supercomputer. As such, we construct approximate wave function ansatzes to perform such large-scale simulations using our simulator. The ansatzes are constructed following four steps:

- The generalized single and double (GSD) excitation operators are generated using every 5 consecutive orbitals. For example, if there are 100 Hartree–Fock orbitals obtained from the Hartree–Fock calculation, we first build GSD excitation operators using orbital 1–5, and then orbital 2–6, etc.
- After the fermionic operator pool has been constructed, the Jordan–Wigner transformation is used to generate an initial operator pool $\{P\}$ in the form of Pauli strings.
- All the Pauli-Zs are removed from the Pauli strings in order to reduce the quantum circuit depth. Because the Hamiltonian is real, all Pauli strings with an even number of Pauli-Ys are removed from $\{P\}$.
- The parametric circuit is adaptively constructed as a product of the exponential of Pauli strings $\prod_j \exp(i\theta_j P_j)$, where $P_j \in \{P\}$ and $\{\theta\}$ are variational parameters to be optimized. Here, we follow the strategy suggested in the qubit-ADAPT-VQE method³⁷. While we did not iteratively build the wave function ansatz until convergence, high accuracy can be

achieved if more iterations are performed to improve the wave function ansatz.

The above steps are performed by interfacing our MPS-VQE simulator with the Q²Chemistry package^{15,36}. In this way, an approximate wave function ansatz that entangles every neighbouring 5 orbitals (10 qubits) is constructed for the hydrogen chain simulations. Another important factor that affects the simulation accuracy is the maximum allowed bond dimension of the MPS simulator. In order to choose a reasonable bond dimension, we performed a benchmark on the converged energy with respect to different bond dimension settings using a smaller molecule (H₈, 16 qubits). The results are given in Fig. 6 and the bond dimension is selected such that $\Delta E = |E_{D_i} - E_{D_{i+1}}| < 1.0 \times 10^{-3}$ Hartree which is slightly more strict than chemical accuracy (1.6×10^{-3} Hartree).

The DMET method

In DMET, a high-level calculation for each fragment (e.g. VQE) is carried out individually until the self-consistency criterion has been met: the sum of the number of electrons of all of the fragments agrees with the number of electrons for the entire system. The DMET energy for the fragment is calculated using the 1-RDM and 2-RDM, that is,

$$E_A = \sum_{p \in A} \left(\sum_q^{N_{\text{orb}}^A + N_{\text{orb}}^B} \left(h_{pq} + \frac{1}{2} \sum_{rs}^{N_{\text{orb}}} [(pq|rs) - (ps|rq)] D_{rs}^{\text{env},A} \right) D_{qp}^A \right. \\ \left. + \frac{1}{2} \sum_{qrs}^{N_{\text{orb}}^A + N_{\text{orb}}^B} (pq|rs) P_{qp}^A \right), \quad (14)$$

where h_{pq} are the one-electron integrals, $(pq|rs)$ are two-electron integrals, N_{orb}^A is the number of orbitals in the fragment, N_{orb}^B is the number of the bath orbitals, N_{orb} is the total number of the orbitals in the entire molecule and p, q, r, s are orbital indices. $D_{qp}^A = \langle \hat{a}_p^\dagger \hat{a}_q \rangle$ is 1-RDM and $P_{qp}^A = \langle \hat{a}_p^\dagger \hat{a}_q^\dagger \hat{a}_r \hat{a}_s \rangle$ is 2-RDM, which are evaluated with VQE method in this work. The number of electrons in fragment A is calculated as $N^A = \sum_{p \in A} D_{pp}^A$ and the DMET total energy is the sum of the fragment energies

$$E^{\text{total}} = \sum_A E_A \quad (15)$$

The DMET cycle iterates until the number of electrons $N^{\text{DMET}} = \sum_A N^A$ converges to the total number of electrons in molecule (N).

Heterogeneous parallelization strategy

For the DMET-MPS-VQE simulator, three levels of parallelization are adopted: (1) The calculation of different fragments can be performed in an embarrassingly parallel manner, that we split the whole CPU pool into different sub-groups and sub-communicators, and there is no communication between different fragment calculations; (2) within each sub-group, the total energy of each fragment is calculated with the MPS-VQE method. We adopted the parallel simulation algorithm based on distributed memory over the circuits, just “mimic” the actual quantum computers, so our method can offer a good reference for VQE running on the quantum computers; (3) within the simulations of a single quantum circuit, we use low-level multi-threaded parallelism on the CPEs to further boost the performance for the tensor contraction and singular value decomposition. We refer the reader to ref. ¹⁵ for more details.

Julia programming language

The Julia script language is used as the main programming language in this study. Julia has the performance of a statically

compiled language while providing interactive dynamic behavior and productivity³⁸. The codes written in Julia can be highly extensible due to its type system and the multiple dispatch mechanism. In addition to its JIT feature and meta-programming ability, its powerful foreign function interface (FFI) makes it easy to use external libraries written in other languages. In this study, the electronic structure libraries Pyscf³⁹ and OpenFermion⁴⁰ are linked to Julia through PyCall.jl, and the optimized SVD routines written in C is called using the LLVM.jl package which provides a high-level wrapper to the LLVM C API.

Our parallel algorithm implemented in Julia is based on the parallel libraries MPI.jl. MPI.jl is a basic Julia wrapper for the Message Passing Interface (MPI). On the Sunway architecture, the MPI libraries are versatile and highly optimized. MPI.jl can call this MPI library through interfaces of Julia that are almost identical to the C language, and provides similar performance.

SVD and Jacobi-based method

The singular value decomposition of a Matrix $A_{m \times n}$ can be written as

$$A = U \Sigma V^T \quad (16)$$

where the matrix $A_{m \times n}$ is decomposed into three matrices. Matrix $U_{m \times m}$ and $V_{n \times n}$ are complex unitary matrices, and $V_{n \times n}^T$ is the conjugate transpose of $V_{n \times n}$. Matrix $\Sigma_{m \times n}$ is a rectangular diagonal matrix with the singular values of matrix $A_{m \times n}$ on the diagonal.

There are two classes of Jacobi-based SVD algorithms: one-sided and two-sided. Two-sided Jacobi iteration algorithm transforms a symmetric matrix into a diagonal matrix by a sequence of two-sided Jacobi rotations (J).

$$J(i, j, \theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & -s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ i \\ \\ j \\ \\ \end{matrix} \quad (17)$$

Based on two-sided Jacobi algorithm, one-sided Jacobi SVD calculates singular value decomposition with only one-sided Jacobi rotations that modifies columns only. Algorithm 1 describes the one-sided Jacobi method. The parameters c and s of the Jacobi rotation matrix can be calculated by t and τ .

$$c = \frac{1}{\sqrt{1+t^2}} \quad (18)$$

$$s = t \times c \quad (19)$$

$$t = \frac{\text{sign}(\tau)}{|\tau| + \sqrt{1+\tau^2}} \quad (20)$$

$$\tau = \frac{a_i^\dagger a_i - a_j^\dagger a_j}{2a_i^\dagger a_j} \quad (21)$$

The algorithm converges when all rotations in a sweep are skipped. Since each pair of columns can be orthogonalized independently, the method is also easily parallelized over the CPEs. The simplicity and inherent parallelism of the method make it an attractive first choice for implementation on the many-core system.

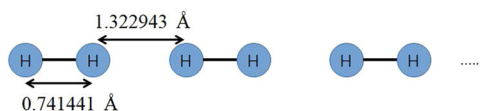


Fig. 7 The geometry of the one-dimensional hydrogen molecule chain. The hydrogen atoms are placed with alternate bond lengths of 0.741441 and 1.322943 Å.

Algorithm 1. One-sided Jacobi SVD method for $m \times n$ matrix A , $m \geq n$.

```

function one_sided_jacobi_svd(A)
  V = I
  for i_sweep = 1, ..., N_sweep do
    pass = true
    for each column pair(i, j), i < j do
      bii = Ai†Ai
      bjj = Aj†Aj
      bij = Ai†Aj
      if |bij| ≥ ε√(biibjj) then
        get Jacobi rotation matrix J
        A = AJ
        V = VJ
        pass = false
      end
    end
    if(pass) break
  end
  for i = 1, ..., n do
    σi = √(Ai†Ai)
    ui = ai/σi
  end
  sort Σ and apply same permutation to columns of
  U and V
  return (U, Σ, V)

```

The quantum simulation time of hydrogen chain with MPS-VQE

The quantum simulation time of the hydrogen chain using the MPS-VQE simulator is tested. The number of atoms (N_a), number of qubits (N_q), and the estimated number of circuits (N_c) are listed in Table 4. The geometry of the hydrogen molecule chain is set as follows: the H₂ moieties with R(H–H) = 0.741 Å were aligned, and the distance between the closest atoms of different H₂ fragments was 1.322 Å, as shown in Fig. 7. For all the calculations, we use 512 cores (8 nodes × 64 cores per node).

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding authors upon reasonable request.

Received: 15 October 2022; Accepted: 9 March 2023;
Published online: 07 April 2023

REFERENCES

- Vogiatzis, K. D., Ma, D., Olsen, J., Gagliardi, L. & de Jong, W. A. Pushing configuration-interaction to the limit: towards massively parallel mcsf calculations. *J. Chem. Phys.* **147**, 184111 (2017).
- Feynman, R. P. Simulating physics with computers. In Hey, A. J. G. (ed) *Feynman and Computation* 133–153 (CRC Press, 2018).
- Arute, F. et al. Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
- Wu, Y. et al. Strong quantum computational advantage using a superconducting quantum processor. *Phys. Rev. Lett.* **127**, 180501 (2021).
- Zhu, Q. et al. Quantum computational advantage via 60-qubit 24-cycle random circuit sampling. *Sci. Bull.* **67**, 240–245 (2022).
- Huggins, W. J. et al. Unbiasing fermionic quantum Monte Carlo with a quantum computer. *Nature* **603**, 416–420 (2022).
- Elfvig, V. E. et al. How will quantum computers provide an industrially relevant computational advantage in quantum chemistry? Preprint at <https://arxiv.org/abs/2009.12472> (2020).
- Peruzzo, A. et al. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **5**, 4213 (2014).
- McArdle, S., Endo, S., Aspuru-Guzik, A., Benjamin, S. C. & Yuan, X. Quantum computational chemistry. *Rev. Mod. Phys.* **92**, 015003 (2020).
- 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.* **15**, 4764–4780 (2019).
- Quantum, G. A. et al. Hartree-Fock on a superconducting qubit quantum computer. *Science* **369**, 1084–1089 (2020).
- Genin, S. N. et al. Estimating phosphorescent emission energies in Ir^{III} complexes using large-scale quantum computing simulations. *Angew. Chem. Int. Ed.* **61**, e202116175 (2022).
- Cao, C. 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* **105**, 062452 (2022).
- Li, W. et al. Toward practical quantum embedding simulation of realistic chemical systems on near-term quantum computers. *Chem. Sci.* **13**, 8953–8962 (2021).
- Shang, H. et al. Large-scale simulation of quantum computational chemistry on a new sunway supercomputer. In *Proc. International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '22 (IEEE Press, 2022).
- Blackford, L. et al. An updated set of basic linear algebra subprograms (blas). https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=50982 (2002)
- Liu, Y. A. et al. Closing the “quantum supremacy” gap: Achieving real-time simulation of a random quantum circuit using a new sunway supercomputer. In *Proc. International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '21. (Association for Computing Machinery, New York, NY, USA, 2021)
- Gu, M., Demmel, J. & Dhillon, I. S. Efficient computation of the singular value decomposition with applications to least squares problems. Technical Report LBL-36201 (Lawrence Berkeley National Laboratory, 1994).
- Demmel, J. & Veselić, K. Jacobi’s method is more accurate than qr. *SIAM J. Matrix Anal. Appl.* **13**, 1204–1245 (1992).
- Bečka, M., Okša, G. & Vajteršič, M. New dynamic orderings for the parallel one-sided block-Jacobi svd algorithm. *Parallel Process. Lett.* **25**, 1550003 (2015).
- Novaković, V. & Singer, S. A GPU-based hyperbolic SVD algorithm. *BIT Numer. Math.* **51**, 1009–1030 (2011).
- Lahabar, S. & Narayanan, P. J. Singular value decomposition on gpu using cuda. In *2009 IEEE International Symposium on Parallel & Distributed Processing* 1–10 (2009).
- Dongarra, J. et al. The singular value decomposition: anatomy of optimizing an algorithm for extreme scale. *SIAM Rev.* **60**, 808–865 (2018).
- Hastens, M. R. Inversion of matrices by biorthogonalization and related results. *SIAM J. Appl. Math.* **6**, 51–90 (1958).
- de Rijk, P. P. M. A one-sided Jacobi algorithm for computing the singular value decomposition on a vector computer. *SIAM J. Sci. Comput.* **10**, 359–371 (1989).
- Schmidt, D. A survey of singular value decomposition methods for distributed tall/skinny data. In *2020 IEEE/ACM 11th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (Scala)* 27–34 (2020).
- Duan, X. et al. Tuning a general purpose software cache library for TaihuLight’s SW26010 processor. *CCF Trans. High Perform. Comput.* **2**, 164–182 (2020).
- Miki, T., Okita, R., Shimada, M., Tsukayama, D. & Shirakashi, J.-i. Variational ansatz preparation to avoid cnot-gates on noisy quantum devices for combinatorial optimizations. *AIP Adv.* **12**, 035247 (2022).
- Hoja, J., Reilly, A. M. & Tkatchenko, A. First-principles modeling of molecular crystals: structures and stabilities, temperature and pressure. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **7**, e1294 (2017).
- Kirsopp, J. J. M. et al. Quantum computational quantification of protein–ligand interactions. *Int. J. Quantum Chem.* **122**, e26975 (2021).

31. Wang, Y., Murlidaran, S. & Pearlman, D. A. Quantum simulations of SARS-CoV-2 main protease Mpro enable high-quality scoring of diverse ligands. *J. Comput. Aided Mol. Design* **35**, 963–971 (2021).
32. Li, Z. et al. Identify potent sars-cov-2 main protease inhibitors via accelerated free energy perturbation-based virtual screening of existing drugs. *Proc. Natl Acad. Sci. USA* **117**, 27 381–27 387 (2020).
33. Poulin, D. et al. The trotter step size required for accurate quantum simulation of quantum chemistry. *Quantum Inf. Comput.* **15**, 361 (2015).
34. Grimsley, H. R., Economou, S. E., Barnes, E. & Mayhall, N. J. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nat. Commun.* **10**, 1–9 (2019).
35. Hastings, M. B. Light-cone matrix product. *J. Math. Phys.* **50**, 095207 (2009).
36. Fan, Y. et al. Q²chemistry: a quantum computation platform for quantum chemistry. *J. Univ. Sci. Technol. China* **52**, 2 (2022).
37. Tang, H. L. et al. Qubit-adapt-vqe: an adaptive algorithm for constructing hardware-efficient ansätze on a quantum processor. *PRX Quantum* **2**, 020310 (2021).
38. Bezanson, J., Edelman, A., Karpinski, S. & Shah, V. B. Julia: A Fresh Approach to Numerical Computing. *SIAM Rev.* **59**, 65–98 (2017).
39. Sun, Q. et al. Pyscf: the python-based simulations of chemistry framework. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **8**, e1340 (2018).
40. McClean, J. R. et al. Openfermion: the electronic structure package for quantum computers. *Quantum Sci. Technol.* **5**, 034014 (2020).
41. Bylaska, E. J. et al. Quantum solvers for plane-wave hamiltonians: abridging virtual spaces through the optimization of pairwise correlations. *Front. Chem.* **9**, 26 (2021).
42. Yalouz, S. et al. A state-averaged orbital-optimized hybrid quantum–classical algorithm for a democratic description of ground and excited states. *Quantum Sci. Technol.* **6**, 024004 (2021).
43. Manrique, D. Z., Khan, I. T., Yamamoto, K., Wichitwechkarn, V. & Ramo, D. M. Momentum-space unitary coupled cluster and translational quantum subspace expansion for periodic systems on quantum computers. Preprint at <https://arxiv.org/abs/2008.08694> (2021).
44. Xia, R. & Kais, S. Qubit coupled cluster singles and doubles variational quantum eigensolver ansatz for electronic structure calculations. *Quantum Sci. Technol.* **6**, 015001 (2020).
45. Liu, J., Wan, L., Li, Z. & Yang, J. Simulating periodic systems on a quantum computer using molecular orbitals. *J. Chem. Theory Comput.* **16**, 6904–6914 (2020).
46. Fan, Y., Liu, J., Li, Z. & Yang, J. Equation-of-motion theory to calculate accurate band structures with a quantum computer. *J. Phys. Chem. Lett.* **12**, 8833–8840 (2021).
47. Kottmann, J. S., Schleich, P., Tamayo-Mendoza, T. & Aspuru-Guzik, A. Reducing qubit requirements while maintaining numerical precision for the variational quantum eigensolver: a basis-set-free approach. *J. Phys. Chem. Lett.* **12**, 663–673 (2021).

ACKNOWLEDGEMENTS

H.S. acknowledges support from National Natural Science Foundation of China (Grant No. T2222026, 22003073). L.S. acknowledges support from National Key Research and Development Program of China (Grant No. 2018YFB0204200). C.G. acknowledges support from National Natural Science Foundation of China (Grant No. 11805279). J.L. acknowledges National Natural Science Foundation of China (Grant No. 22073086), Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0303306), the Fundamental Research Funds for the Central Universities (Grant No. WK2060000018). Computational resources were provided by the new Sunway supercomputer.

AUTHOR CONTRIBUTIONS

The project was conceived by H.S. The manuscript was written by H.S., J.L., and C.G. The numerical simulations were performed by Y.F., H.S., L.S., F.L., X.D., and Z.L.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Correspondence and requests for materials should be addressed to Honghui Shang, Chu Guo or Jie Liu.

Reprints and permission information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2023