IOP Publishing Mach. Learn.: Sci. Technol. 4 (2023) 025034 https://doi.org/10.1088/2632-2153/acdb2f

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
16 February 2023

REVISED
12 May 2023

ACCEPTED FOR PUBLICATION
2 June 2023

PUBLISHED
23 June 2023

PAPER

Scalable neural quantum states architecture for quantum chemistry

Tianchen Zhao[1], **James Stokes**[2] **and Shravan Veerapaneni**[1,2,*]

[1] Department of Mathematics, University of Michigan, Ann Arbor, MI 48109, United States of America
[2] Flatiron Institute, Simons Foundation, New York, NY 10010, United States of America
* Author to whom any correspondence should be addressed.

E-mail: shravan@umich.edu

Keywords: neural quantum states, quantum chemistry, high-performance computing, variational Monte Carlo

Abstract

Variational optimization of neural-network representations of quantum states has been successfully applied to solve interacting fermionic problems. Despite rapid developments, significant scalability challenges arise when considering molecules of large scale, which correspond to non-locally interacting quantum spin Hamiltonians consisting of sums of thousands or even millions of Pauli operators. In this work, we introduce scalable parallelization strategies to improve neural-network-based variational quantum Monte Carlo calculations for *ab-initio* quantum chemistry applications. We establish GPU-supported local energy parallelism to compute the optimization objective for Hamiltonians of potentially complex molecules. Using autoregressive sampling techniques, we demonstrate systematic improvement in wall-clock timings required to achieve coupled cluster with up to double excitations baseline target energies. The performance is further enhanced by accommodating the structure of resultant spin Hamiltonians into the autoregressive sampling ordering. The algorithm achieves promising performance in comparison with the classical approximate methods and exhibits both running time and scalability advantages over existing neural-network based methods.

1. Introduction

Quantum many-body systems describe a vast category of physical problems at microscopic scales. In the context of *ab-initio* quantum chemistry (QC), the central topic is to unravel the quantum effects determining the structure and properties of molecules by solving the many-body time-independent Schrödinger equation describing the interaction of heavy nuclei with orbiting electrons. However, the exponential complexity with respect to the number of particles makes the analytical computations about the system impractical [1].

Classical strategies discretize the problem using a finite number of basis functions, expanding the full many-body state in a basis of anti-symmetric Slater determinants. Because of the exponential scaling of the determinant space, however, exact approaches that systematically consider all electronic configurations such as the full configuration interaction (FCI) method, are typically restricted to small molecules and basis sets. Coupled cluster (CC) techniques [2, 3] are approximate methods routinely adopted in QC electronic calculations, however, the accuracy of CC is intrinsically limited in the presence of strong quantum correlations, in turn restricting the applicability of the method to regimes of relative weak correlations. Recently neural-network quantum state (NQS) methods [4, 5] have proven to be successful variational ansatze for finding the ground state of molecular systems up to 30 qubits (Li_2O). However, significant scalability challenges of the NQS approach arise when considering molecules of larger scales. The scalability issue stems from two sources, which we refer to as local energy parallelism and sampling parallelism. (i) The complexity of the computation of local energy scales linearly with respect to the number of terms in the molecular Hamiltonian, which induces out-of-memory (OOM) issues for larger problems. (ii) In order to achieve satisfactory performance, one needs to sample exact and accurate configurations from the targeted

distribution, which becomes expensive or even impossible for existing approaches such as Markov chain Monte-Carlo (MCMC) in high dimensions.

In order to address local energy parallelism for large-scale molecules, we use an efficient tensor representation of the second quantized spin Hamiltonian generated from chemical data so that the computation of the local energy is efficiently supported by GPUs. We further employ identical copies of the model across the computing units to generate only a few samples per unit and combine the independent samples from all these units to construct an accurate expectation estimate. In addition, memory consumption is reduced through gradient accumulation, which splits the batch of samples used for training the model into several mini-batches of samples that can be distributed across the devices and processed independently. The proposed parallelization scheme for large batch computation is particularly important for chemical molecules with a large number of terms in their electronic Hamiltonian formulation, where the calculation of local energy becomes prohibitive.

The basis of our approach to achieving sampling parallelism is our utilization of a wavefunction based on Masked Autoencoder for Distribution Estimation (MADE) [6]. Using MADE as our base model enables the development of a parallelization scheme based on [7]. Unlike restricted Boltzmann machines (RBMs) [8] and NADE [9], which have formed the basis of previous *ab-initio* studies, MADE is known to be lightweight and scalable to high-dimensional inputs. On the one hand, it overcomes the asymptotical convergence issues of MCMC by using autoregressive sampling. On the other hand, it bypasses the inherently sequential nature of neural autoregressive distribution estimator (NADE) with the minor additional cost of simple masking operations. In addition, we further improve the performance of our model through the autoregressive sampling of the state entries in an order that roughly matches the entanglement hierarchy among the involved qubits. Our experiments demonstrate that the proposed algorithm effectively works for molecules up to 76 qubits with millions of terms in its electronic Hamiltonian.

The paper is organized as follows. In section 2, we begin by summarizing the existing state-of-art in NQS research as applied to *ab initio* QC. Section 3 provides mathematical details about the nature of the Hamiltonians under consideration as well as stochastic approximation strategy based on neural networks. The proposed NQS architecture is elaborated in section 4 and the parallel evaluation strategies are subsequently detailed in section 5. Experimental results for molecules up to 76 qubits are described in section 6, including ablative studies on the relevant factors controlling the performance of the algorithm.

2. Related work

Variational Monte-Carlo and autoregressive quantum states. The idea of utilizing NQSs to overcome the curse of dimensionality in high-dimensional VQMC simulations was first introduced by Carleo and Troyer [10], who concentrated on RBMs applied to two-dimensional quantum spin models. However, RBM relies on approximate sampling procedures like MCMC, whose convergence time remains undetermined, which often results in the generation of highly correlated samples and deterioration in performance. Such sampling approximations can be avoided by using autoregressive models [11] that estimate the joint distribution by decomposing it into a product of conditionals by the probability chain rule, making both the density estimation and generation process tractable. To this end, Larochelle and Murray [9] proposed NADE as feed-forward architectures. MADE [6] improves the efficiency of models with minor additional costs for simple masking operations. Sharir *et al* [12, 13] and Hibat-Allah [14] introduced NQSs based on the autoregressive assumption inspired, respectively by PixelCNN [15] and recurrent neural networks, and demonstrated significantly improved performance compared to RBMs.

Application in QC. Many approximate methods specific to the QC problem [16, 17] have been discovered by researchers. The Hartree–Fock (HF) approximation treats each electron in the molecule as an independent particle that moves under the influence of the Coulomb potential due to the nuclei, and a mean-field generated by all other electrons. It calculates the expansion coefficients of the linear combination of atomic orbitals. Configuration interaction methods [18] use a linear combination of configuration state functions built from spin orbitals to restrict the active space to configuration strings. CC approaches [2, 3] construct multi-electron wavefunctions using the exponential cluster operator to account for electron correlation, but cannot parameterize arbitrary superpositions and occasionally lead to unphysical solutions. Choo *et al* [4] proposed an RBM-based NQS variational ansatz, leveraging the power of artificial neural networks that have recently emerged in the more general context of interacting many-body quantum matter [10]. This approach provides a compact, variational parameterization of the many-body wave function. Barrett *et al* [5] subsequently proposed a novel autoregressive NQS architecture for *ab-initio* QC based on NADE [9] with hard-coded pre-and post-processing that enables exact sampling of physically viable states. These advances ultimately allow their approach to scale to systems up to 30 qubits, surpassing what was previously possible using RBMs, while still falling short of large-scale applications. Empirically, we have

found that the approach of [5] encounters significant scalability challenges in generalizing to molecular systems of yet larger size.

3. Problem formulation

We will work under the assumption of the so-called Born-Oppenheimer approximation [19], which treats the nuclei as fixed point charges. Indeed, the specification of a molecular geometry implicitly assumes such an approximation. When the positions of the nuclei are specified, the electronic structure problem can be restated as finding the ground eigenstate of the electronic Hamiltonian operator and consequently the ground state electronic energy becomes a parametric function of atomic positions. Here, the molecule's electronic Hamiltonian is commonly represented using the second-quantization formalism, with a chosen basis of atomic orbitals which describe the wave function of electrons in the molecule. In order to identify the Hamiltonian for a compound, we start by fetching the information required to build the target molecule object, then employ established solvers to compute the second-quantized Hamiltonian and store the result in a string format.

In the second-quantized formulation, the Hamiltonian is represented as a complex conjugate-symmetric (Hermitian) matrix H of side length 2^n where n represents the number of orbitals, and the basic problem is to determine the ground energy of H and a description of an associated eigenvector. Since storage and manipulation of such matrices is prohibitive, a common practice is to exploit the fact that H admits an efficient description in terms of superpositions of tensor products of 2×2 matrices drawn from the set $\{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$, where

$$\sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \sigma_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \sigma_2 = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \qquad \sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{1}$$

In fact, any Hermitian matrix $H \in \mathbb{C}^{2^n \times 2^n}$ admits a unique decomposition of the form

$$H = \sum_{\mathbf{p} \in \{0,1,2,3\}^n} \alpha_{\mathbf{p}} P_{\mathbf{p}} \tag{2}$$

where $P_{\mathbf{p}} := \sigma_{p_1} \otimes \cdots \otimes \sigma_{p_n}$ denotes a tensor product of Pauli matrices, and the real-valued coefficients entering the sum are determined by the formula $\alpha_{\mathbf{p}} = \frac{1}{2^n} \operatorname{tr}(HP_{\mathbf{p}})$. Since each matrix $P_{\mathbf{p}}$ is row-sparse with exactly one nonzero entry per row, the number of nonzero entries per row of H is bounded by the number of nonzero coefficients K entering the above sum, which could be as large as 4^n for typical matrices. The matrices of relevance to QC are far from typical, however, and consequently enjoy a high level of row sparsity, which will be a crucial property in our subsequent algorithm development.

Beyond the assumption of row-sparsity, additional features of H can be determined from the structure of the constituent Pauli strings $\mathbf{p}$, which in turn depend on the choice of encoding map from the chemical Hamiltonian to the qubit representation. Efficient qubit encoding maps are an active field of research, and the most common examples in usage are Jordan–Wigner [20] and Bravyi–Kitaev [21], which are equivalent isospectrally. In this work, we will adopt the Jordan–Wigner encoding because of its simplicity and the fact that it has shown success in prior work on VMC applied to *ab-initio* QC [4, 5]. In the particular case of Jordan–Wigner transformation, we point out that many of the tensor factors of terms appearing in the sum (2) are the identity matrix and are moreover organized in a hierarchical structure.

Following the standard neural-network variational Monte Carlo procedure, we postulate a family of trial wavefunctions whose complex amplitudes relative to the standard basis are computed by the output of a neural network, parametrized by variational parameters $\theta \in \mathbb{R}^d$. Thus, given a function of the form

$$f : \{0,1\}^n \times \mathbb{R}^d \longrightarrow \mathbb{C} \tag{3}$$

which is differentiable in the second argument, we define an associated family of trial quantum states $|f_\theta\rangle$, which are differentiably parametrized by $\theta \in \mathbb{R}^d$, as follows

$$|f_\theta\rangle := \sum_{x \in \{0,1\}^n} f(x, \theta)|x\rangle \tag{4}$$

where $|x\rangle := |x_1\rangle \otimes \cdots \otimes |x_n\rangle$ is a shorthand to denote the standard basis vectors for $\mathbb{C}^{2^n}$. In this work, following [5], we assume that the function f is chosen in such a way that $|f_\theta\rangle$ is unit-normalized for all $\theta \in \mathbb{R}^d$, and that exact sampling from the probability distribution $\pi_\theta := |f(\cdot, \theta)|^2$ is computationally tractable. Using the unit vector (4) as a trial vector in the Rayleigh quotient for H, we obtain a differentiable

objective function $\mathcal{L}:\mathbb{R}^d \longrightarrow \mathbb{R}$, which upper bounds the minimal eigenvalue $\lambda_{\min}(H)$ as a consequence of the Rayleigh–Ritz principle,

$$\mathcal{L}(\theta) := \frac{1}{2}\langle f_\theta | H | f_\theta \rangle \tag{5}$$

and whose value can be estimated at the Monte Carlo rate using the following estimator

$$\mathcal{L}(\theta) = \frac{1}{2}\mathbb{E}[l_\theta(x)] \quad , \qquad l_\theta(x) := \frac{\langle x|H|f_\theta\rangle}{f(x,\theta)} \quad , \qquad x \sim \pi_\theta \tag{6}$$

where l_θ is referred to as the local energy. Minimization of $\mathcal{L}$ is performed using stochastic gradient-based optimization techniques. In particular, the gradient of $\mathcal{L}$ can be estimated at the Monte Carlo rate using,

$$\nabla\mathcal{L}(\theta) = \mathrm{Re}\,\mathbb{E}\left[(l_\theta(x)\mathbb{1} - b)\,\overline{\sigma_\theta}(x)\right] \quad , \qquad \sigma_\theta(x) := \frac{\nabla_\theta f(x,\theta)}{f(x,\theta)} \quad , \qquad x \sim \pi_\theta \tag{7}$$

where $b \in \mathbb{R}^{d\times d}$ is an arbitrary baseline matrix that can be set to $b = \mathbb{E}[l_\theta(x)]$.

Equation (7) demonstrates that an integral component of the algorithm is the computation of the local energy, which is required both to perform stochastic gradient updates of the model and to estimate the value of the objective function. The computational complexity of computing the mapping $x \longmapsto l_\theta(x)$ for a minibatch of size B is evidently $O(BK)$ where recall that K denotes the number of terms in the Hamiltonian expansion (2). Despite the fact that the sparsity parameter satisfies $K \ll 4^n$, the computation of local energy still suffers from severe OOM issues in practice since 4^n can be extraordinarily large even for modestly sized molecules. For example, in the case of Methanol with $n = 28$ orbitals we have $4^n \approx 7.2 \times 10^{16}$ while $K = 52\,887$. This implies that for Methanol, given a modest batch size $B = 1024$, the local energy needs a forward pass of 54M samples of input size 28. This computational bottleneck is inevitable as large batch sizes are essential to ensure the precision of the Monte-Carlo approximation of the energy objective in equation (6), which directly influences the performance of the algorithm. Therefore, efficient sampling and local energy computation become the key issue for scaling the neural-network variational approach to large compounds.

4. Autoregressive modeling of molecular quantum states

Motivated by the fact that solving high dimensional molecular quantum systems is still a difficult problem with existing variational techniques, we consider the base model from Zhao *et al* [7], which was shown to be capable of solving quantum systems of very high dimensions. Adaptation of this work to chemistry problems involves generalizing from real-valued to complex-valued wave functions, as well as adjusting the sampling process to handle larger numbers of configurations and accommodate domain priors. In addition, following [5], we enforce constraints necessary to ensure that the generated samples correspond to physical electronic states.

Architecture. Since the ground state of the targeted problem is in general complex-valued, we consider the complex generalization of the model in [7] by splitting the output of the wave function into modulus and phase parts, learned by two sub-models separately as follows

$$\begin{aligned}\text{Modulus sub-model: } \textit{Input} &\xrightarrow{[B,N]} \texttt{MaskedFC1} \xrightarrow{[B,h]} \texttt{ReLU} \\ &\xrightarrow{[B,h]} \texttt{MaskedFC2} \xrightarrow{[B,N]} \texttt{Sigmoid} \xrightarrow{[B,N]} \textit{Output},\\ \text{Phase sub-model: } \textit{Input} &\xrightarrow{[B,N-2]} \texttt{MaskedFC1} \xrightarrow{[B,h]} \texttt{ReLU} \xrightarrow{[B,h]} \texttt{MaskedFC2} \xrightarrow{[B,4]} \textit{Output}.\end{aligned}$$

The modulus model predicts n conditional probabilities for each configuration, and the phase model predicts the phases for each of the four configurations that are identical in the first $n-2$ entries, which is a $n-2$ dimensional vector being fed as the input. Here B is the batch size, n is the number of dimensions, h is the hidden layer size and `MaskedFC` is the masked fully connected layer, which removes the connections in the computational path of MADE. The outputs of modulus sub-model are the conditional probabilities $\{\pi_i(x_i|x_{i-1},\ldots,x_1)\}_{i=1}^n$, which together define a joint probability function $\pi : \{0,1\}^n \longrightarrow [0,\infty)$ for input strings x. Normalization follows automatically from the autoregressive assumption,

$$\pi(x) = \prod_{i=1}^{n} \pi_i(x_i|x_{i-1},\ldots,x_1) \quad . \tag{8}$$

On the other hand, we model the phase $\phi : \{0,1\}^n \longrightarrow [0, 2\pi)$ of the wavefunction directly by feeding the input x into a two-layer MLP. It follows that the complex logarithm of the model output $f(x,\theta)$ can be written in a computationally tractable form as

$$f(x,\theta) = e^{i\phi(x)} \prod_{i=1}^{n} \sqrt{\pi_i(x_i|x_{i-1},\ldots,x_1)}, \tag{9}$$

$$\log f(x,\theta) = i\phi(x) + \frac{1}{2}\sum_{i=1}^{n} \log \pi_i(x_i|x_{i-1},\ldots,x_1) \tag{10}$$

where $\theta \in \mathbb{R}^d$ denotes the concatenation of parameters describing the neural networks ϕ and π.

Sampling. Standard autoregressive sampling techniques such as NADE [9] have a sequential nature that updates a batch of randomly initialized states entry by entry following a pre-fixed order. In practice, this approach is highly inefficient as the sampled batch usually contains repeated samples. Instead, we keep track of a sample buffer throughout the sampling process, associated with a counter storing the number of occurrences for each sample in the buffer. We start with a buffer containing only one random initialized sample. At each of the n iterations, we first double the size of the buffer by alternating ± 1 value for existing samples at a fixed entry. Then, we update the counter for all samples in the buffer through Bernoulli sampling with probabilities computed by forward pass. Finally, we eliminate the samples in the batch that have the lowest numbers of occurrences in the counter, to avoid the exponential growth of the buffer size.

In addition, as a consequence of the Jordan–Wigner encoding, many of the tensor factors appearing with high qubit index are given by the identity factor, which leads to the expectation that high index qubits are comparatively less correlated compared to those with at low indices. Motivated by this observation, in an effort to ease the training of the model we performed autoregressive sampling in a reversed order beginning with the nth qubit.

Constraints. Recall that the unconstrained model assigns nonzero probability mass to all 2^n possible bit-strings representing possible states of a multi-electron system. However, in quantum chemistry problems we considered, only $n_e \leqslant n$ out of the n single-electron spin-orbitals are occupied with electrons, and the net charge C of the molecular system determines the number of unpaired electrons. It follows that the numbers of electrons with up-spins and down-spins $n_\uparrow, n_\downarrow$ respectively should satisfy

$$n_\uparrow + n_\downarrow = n_e, \quad n_\uparrow - n_\downarrow = C. \tag{11}$$

This corresponds to applying Hamming weight constraints to the bit-strings, which effectively reduces the total number of candidate samples from 2^n to $\binom{n/2}{n_\uparrow}\binom{n/2}{n_\downarrow}$, which significantly reduces the complexity of the problem. We adopted techniques from similar work [5, 14] to enforce the constraints. Define the hamming weight constraint of the configuration $x \in \{0,1\}^n$ to be

$$\sum_{i=1}^{n/2} x_{2i-1} = n_\uparrow, \quad \sum_{i=1}^{n/2} x_{2i} = n_\downarrow, \tag{12}$$

where the even and odd indices correspond to the up-spin and down-spin electrons in the orbitals. The idea to enforce the constraint in the autoregressive sampling process is to assign nonzero probabilities only to samples that satisfy equation (12), for which the sufficient and necessary condition is that the kth entry x_k must satisfy

$$n_\uparrow - (n/2 - \lceil k/2 \rceil) \leqslant \sum_{i=1}^{\lceil k/2 \rceil} x_{2i-1} \leqslant n_\uparrow, \quad n_\downarrow - (n/2 - \lceil k/2 \rceil) \leqslant \sum_{i=1}^{\lceil k/2 \rceil} x_{2i} \leqslant n_\downarrow, \tag{13}$$

for all $k \in 1, 2, \ldots, N$. Condition (13) can be enforced at every iteration k during the sampling process by introducing the following modified probability distribution,

$$\widehat{\pi}(x_k = 1|x_{k-1},\ldots,x_1) = \begin{cases} 1, & \text{if } \sum_{i=1}^{\lceil k/2 \rceil} x_{2i-1} < n_\uparrow - (n/2 - \lceil k/2 \rceil) \\ 1, & \text{if } \sum_{i=1}^{\lceil k/2 \rceil} x_{2i} < n_\downarrow - (n/2 - \lceil k/2 \rceil) \\ 0, & \text{if } \sum_{i=1}^{\lceil k/2 \rceil} x_{2i-1} \geqslant n_\uparrow \\ 0, & \text{if } \sum_{i=1}^{\lceil k/2 \rceil} x_{2i} \geqslant n_\downarrow \\ \pi(x_k = 1|x_{k-1},\ldots,x_1), & o.w., \end{cases} \tag{14}$$

which has the property that for any x violating the constraints, we have $\widehat{\pi}(x_k = x|x_{k-1},\ldots,x_1) = 0$ and $\widehat{\pi}(x_k = 1 - x|x_{k-1},\ldots,x_1) = 1$. The modified probabilities are still normalize to one by design.

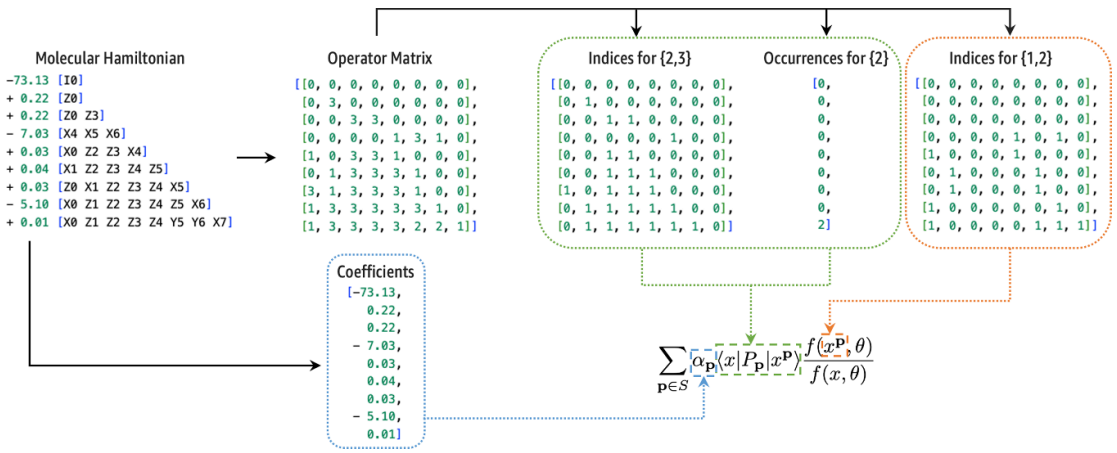

Figure 1. Local energy computational paradigm. The Hamiltonian is parsed into an operator matrix and a coefficient vector, from which incidence matrices $F^{\{1,2\}}, F^{\{2,3\}} \in \{0,1\}^{K\times n}$ are extracted for subsequent calculation of the local energy. This formulation provides significant computational speed-up for molecular Hamiltonian with a huge number of terms.

Algorithm 1. Parallel tensor computation of local energy (presented for batch size 1).

Input: Bit string $x \in \{0,1\}^n$, coefficient vector, incidence matrices $F^{\{1,2\}}, F^{\{2,3\}} \in \{0,1\}^{K\times n}$
Output: Local energy
Compute $\{x^{\mathbf{p}}\} \in \{0,1\}^{K\times n}$ as $\{x^{\mathbf{p}}\} = X \oplus F^{\{1,2\}}$, where $X \in \{0,1\}^{K\times n}$ is the K-fold replication of x
Compute amplitudes $f(x) \in \mathbb{C}$ and $\{f(x^{\mathbf{p}})\} \in \mathbb{C}^K$
Compute $\{\langle x|P_{\mathbf{p}}|x^{\mathbf{p}}\} \in \mathbb{C}^K$ using equation (16)
Evaluate the sum in equation (15)

5. Parallelization

We parallelize the training by distributing the input batch across the GPUs, where the model parameters are replicated on each GPU, which handles a portion of the full batch. During the backward pass, gradients from each node are averaged. Locally within each process, we tensorize the entire computational pipeline so that the computation of local energies is fully GPU-supported.

5.1. Tensorized computation of local energy

The form of the local energy has the property that it can easily be parallelized, which becomes increasingly important with increasing molecular system size since the Hamiltonian can potentially involve a large number of terms. We implemented an efficient tensor representation of the second quantized spin Hamiltonian generated from chemical data, which capitalizes on the fact that the matrix corresponding to an arbitrary product of Pauli operators $P_{\mathbf{p}}$ is extremely sparse. In particular, for each row index $x \in \{0,1\}^n$ there is exactly one column index $x^{\mathbf{p}} \in \{0,1\}^n$, for which the corresponding matrix entry $\langle x|P_{\mathbf{p}}|x^{\mathbf{p}}\rangle$ is nonzero[3]. If we denote the subset of Pauli strings with nonzero coefficient by $S = \{\mathbf{p} \in \{0,1,2,3\}^n : \alpha_{\mathbf{p}} \neq 0\}$, then the local energy simplifies to

$$l_\theta(x) = \sum_{\mathbf{p}\in S} \alpha_{\mathbf{p}} \langle x|P_{\mathbf{p}}|x^{\mathbf{p}}\rangle \frac{f(x^{\mathbf{p}},\theta)}{f(x,\theta)} \quad . \tag{15}$$

The goal now is to efficiently compute equation (15), in which the number of summands $K = |S|$ is very large. The idea, depicted in figure 1, is to extract the key information required to perform the computation from the molecular Hamiltonian and store the information as tensors that directly support GPU computation. To this end, we constructed a string parser that computes an Operator Matrix along with coefficients. For Pauli string, we track the indices of Pauli $\sigma_1, \sigma_2, \sigma_3$ operators as tensors, which are later utilized to compute $x^{\mathbf{p}}$ and the corresponding matrix element $\langle x|P_{\mathbf{p}}|x^{\mathbf{p}}\rangle$.

In practice, $x^{\mathbf{p}}$ is computed by flipping the bits of x based corresponding to the locations of σ_1 and σ_2 in $P_{\mathbf{p}}$. To improve the efficiency, we collect the indices of σ_1, σ_2 operators prior to the training and only keep a

[3] The formula for the associated matrix entry in terms of x is given in [4].

buffer of unique flippings and their corresponding number of occurrences. This approach can effectively reduce the input size to the forward pass by removing the repeated samples, which is particularly helpful for large-sized problems. The matrix element $\langle x|P_{\mathbf{p}}|x^{\mathbf{p}}\rangle$ admits the formula

$$\langle x|P_{\mathbf{p}}|x^{\mathbf{p}}\rangle = \langle x|\sigma_{p_1}\otimes\cdots\otimes\sigma_{p_n}|x^{\mathbf{p}}\rangle = (-i)^r \prod_{k:p_k\in\{2,3\}} (-1)^{x_k} \tag{16}$$

where r is the number of occurrences for the σ_2 operator, i.e. $r=\sum_{k=1}^{n}\mathbb{1}_{\{p_k=2\}}$. Similar to before, the indices of σ_2,σ_3 operators are collected prior to the training, and the product can be calculated by the Hadamard product between the indices and x, followed by the production of all entries. Note that all computations in this subsection can be performed in parallel for all terms with GPU.

5.2. Parallel algorithm implementation

In applications such as QC, the Hamiltonian of even the smallest molecules contain thousands of terms, which leads to severe OOM issues for the existing VMC platforms. Our proposed pipeline tensorizes the information in the molecular Hamiltonian to maximize memory efficiency. In addition, the computation of local energy is conducted term-wise with no interaction between the terms, which motivates embarrassingly parallel algorithms for Hamiltonians consisting of a large number of terms. We take a step toward addressing the bottleneck by applying our sampling parallelization strategy to this problem, where we use identical copies of the model across the computing units to generate only a few samples per unit and combine the independent samples from all these units to construct an accurate expectation estimate. In addition, we remove the replicated configurations as described in the previous section before the forward pass locally for each GPU to save more memory.

In more detail, recall that the energy expectation is approximated as the following double sum

$$\frac{\langle f_\theta|H|f_\theta\rangle}{\langle f_\theta|f_\theta\rangle} \approx \frac{1}{B}\sum_{i=1}^{B}\sum_{\mathbf{p}\in S}\alpha_{\mathbf{p}}\langle x_i|P_{\mathbf{p}}|x_i^{\mathbf{p}}\rangle\frac{f(x_i^{\mathbf{p}},\theta)}{f(x_i,\theta)} \quad . \tag{17}$$

We distribute the BK summands in the above sum across L GPUs in order to perform the forward and backward passes of the model with a mini-batch size of BK/L. Locally, each GPU has access to the necessary ingredients required to compute the corresponding partial sums of size BK/L. We compute local gradients with forward and backward passes within each GPU and update the model parameters with the averages of local gradients obtained by averaging over BK/L elements. In addition, we use gradient accumulation [22], splitting the batch into several mini-batches before a single update to avoid potential OOM issues for molecules with larger sizes.

6. Experiments

We now investigate the performance of our algorithm and the running time efficiency of the proposed parallelization paradigm. We first demonstrate our main results by comparing our algorithm with HF and CC with up to double excitations (CCSD) baselines, where our performance is either on par or superior to the classical approximate methods over a wide list of chemical molecules. Our performance is also close to the ground truth FCI energies up to molecular systems with 28 qubits, where the results for larger molecules become increasingly hard to obtain. We then perform ablation studies to examine our algorithm over various aspects. First, we show that increasing batch size improves the performance, at the cost of increased algorithmic complexity. On the other hand, our parallelization strategy can effectively reduce the running time, achieving near-optimal weak scaling. Our proposed sampling trick also improves the performance of our model by a noticeable margin. At last, we show that our model architecture exhibits superior running time efficiency compared against RBM [4] and NADE [5].

6.1. Experimental set-ups

Given a target molecule ID, we fetch its corresponding PubChem compound identifier (CID) from the official PubChem website [23]. CID is recognized by `PubChemPy`, a software that provides a way to interact with *PubChem* in Python, allowing depiction and retrieval of chemical properties, such as the geometry of atoms, number of unpaired electrons, total charge, *etc*. The mapping from second-quantized Hamiltonian to interacting spinning model is done by transforming fermion operators into qubit operators with Jordan–Wigner [20] using `OpenFermion-Psi4` [24]. Note that these solvers can also be used to estimate the ground state energies including HF, CCSD, and FCI. The whole data processing pipeline is automatic without further human interference.

Table 1. Best molecular ground-state energies obtained by different methods as described in the main text over five trials. Molecules have been sorted according to the number of qubits used in the Jordan–Wigner representation. In addition, the numbers $(N_\uparrow, N_\downarrow)$ up- and down-spin electrons and the number K of terms in the Hamiltonian are reported. Our method exhibits superior performance in comparison with classical approximate methods such as HF and CCSD and comes close to the FCI ground truth, which is only available up to molecules of size 28 qubits.

Name	MF	n	$N_\uparrow + N_\downarrow$	K	HF Energy	CCSD	Ours	FCI
Hydrogen	H2	4	$1+1=2$	15	−1.066 108 64	**−1.101 150**	**−1.101 150**	−1.101 150
Lithium Hydride	LiH	12	$2+2=4$	631	−7.767 362 13	−7.784 455	**−7.784 460**	−7.784 460
Water	H2O	14	$5+5=10$	1390	−74.964 4475	−75.015 409	**−75.015 511**	−75.015 530
Methylene	CH2	14	$5+3=8$	2058	−37.484 6329	−37.504 411	**−37.504 419**	−37.504 435
Beryllium Hydride	BeH2	14	$3+3=6$	2074	−14.443 2411	−14.472 713	**−14.472 922**	−14.472 947
Ammonia	NH3	16	$5+5=10$	4929	−55.454 7926	−55.520 931	**−55.521 037**	−55.521 150
Methane	CH4	18	$5+5=10$	8480	−39.726 5817	−39.806 022	**−39.806 170**	−39.806 259
Diatomic Carbon	C2	20	$6+6=12$	2239	−74.248 3215	−74.484 727	**−74.486 037**	−74.496 388
Fluorine	F2	20	$9+9=18$	2951	−195.638 041	**−195.661 086**	−195.661 067	−195.66 108
Nitrogen	N2	20	$7+7=14$	2239	−107.498 967	−107.656 080	**−107.656 763**	−107.660 206
Oxygen	O2	20	$9+7=16$	2879	−147.631 948	−147.747 738	−147.749 953	−147.750 235
Lithium Fluoride	LiF	20	$6+6=12$	5849	−105.113 709	−105.159 235	**−105.165 270**	−105.166 172
Hydrochloric Acid	HCl	20	$9+9=18$	5851	−455.135 968	**−455.156 189**	**−455.156 189**	−455.156 189
Hydrogen Sulfide	H2S	22	$9+9=18$	9558	−394.311 379	−394.354 556	**−394.354 592**	−394.354 623
Formaldehyde	CH2O	24	$8+8=16$	20 397	−112.354 197	−112.498 567	**−112.500 944**	−112.501 253
Phosphine	PH3	24	$9+9=18$	24 369	−338.634 114	−338.698 165	**−338.698 186**	−338.698 400
Lithium Chloride	LiCL	28	$10+10=20$	24 255	−460.827 258	−460.847 580	**−460.848 109**	−460.849 618
Methanol	CH4O	28	$9+9=18$	52 887	−113.547 027	**−113.665 485**	**−113.665 485**	−113.666 485
Lithium Oxide	Li2O	30	$7+7=14$	20 558	−87.795 5672	−87.885 514	**−87.885 637**	—
Ethylene Oxide	C2H4O	38	$12+12=24$	137 218	−150.927 608	−151.120 474	**−151.120 486**	—
Propene	C3H6	42	$12+12=24$	161 620	−115.657 941	−115.885 123	**−115.886 571**	—
Acetic Acid	C2H4O2	48	$16+16=32$	461 313	−224.805 400	**−225.050 896**	−225.042 9767	—
Sulfuric Acid	H2O4S	62	$25+25=50$	1235 816	−689.262 656	−689.498 410	**−689.505 237**	—
Sodium Carbonate	CNa2O3	76	$26+26=52$	1625 991	−575.016 102	−575.299 810	**−575.299 820**	—

We train the model over 10 K iterations with Adam optimizer [25] by default at a learning rate of 1×10^{-3} with standard decay rates for the first- and second-moment estimates of $\beta_1 = 0.9$ and $\beta_2 = 0.99$, respectively; no learning rate scheduler is applied. The batch size for the number of unique samples is fixed to be 1024 throughout our experiments unless specified otherwise. For scalability experiments, each GPU is distributed with a constant mini-batch size mB, and the effective batch size is $mB \times L$, where L is the total number of GPUs available. All experiments in this work use a single set of hyperparameters and identical training procedures, and the results are the best energies obtained across exactly 5 seeds. Throughout the experiments, the timing benchmarks are performed on Tesla V100-SXM2-16GB and Intel(R) Xeon(R) CPU @ 2.30 GHz processors on Google Cloud Platform.

6.2. Performance benchmark

We report the performance of our implementation over a wide variety of molecules in table 1. The molecules are sorted according to the number of qubits in their Jordan–Wigner representation. We consider HF and CCSD energies as classical baseline approximate methods to compare against. The FCI ground truth energy is also provided for smaller molecules for reference.

Our model exhibits consistently strong performance on all molecules considered. In particular, the computed energies match the ground-truth FCI result closely on all molecules with up to 20 electrons and 28 spin-orbitals and out-perform other approximate methods on the majority of the molecules we considered. Notice that as the size of the molecule increases, the number of terms in the electronic Hamiltonian formulation also grows quickly, which leads to severe computational issues such as running time and memory consumption. As a result, our proposed algorithm is capable of scaling up to system size with 76 qubits of which the electronic Hamiltonian is consisted of 1.6 million terms, and achieves state-of-the-art performance.

6.3. Ablation studies

In this section, we perform ablation studies to test the effectiveness of the ingredients in our contribution. We start by validating the fact that increasing batch size is capable of improving the performance for larger-scale problems, which justifies the motivation of our parallelization scheme that enables large-batch training for large molecules. We also examine our proposed reverse sampling trick by comparing the performance with

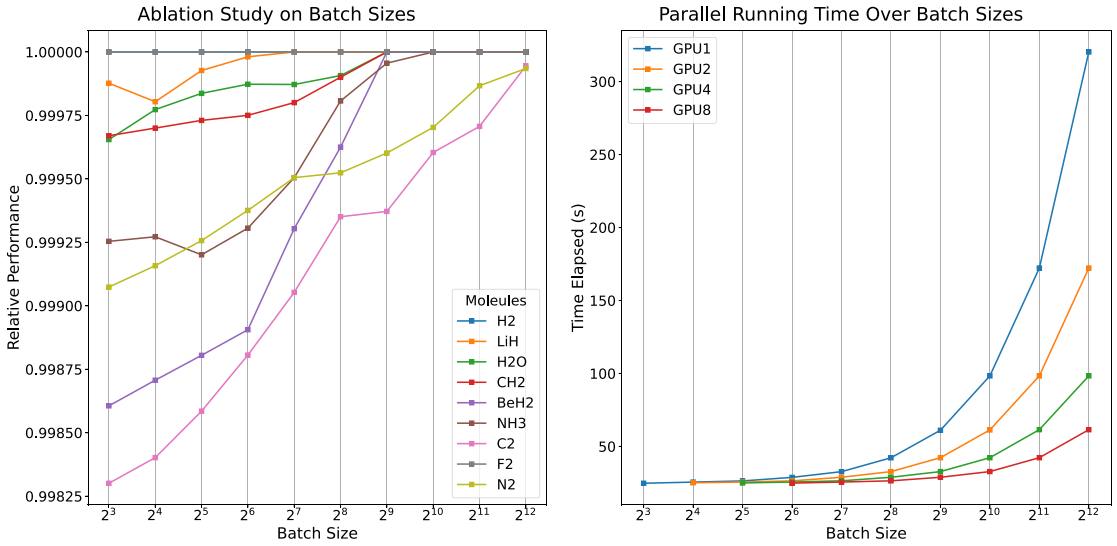

Figure 2. Left: the performance of our algorithm with increasing batch size for different molecules. The relative performance score is obtained by normalizing the estimated energy with respect to the corresponding FCI ground truth. Better performance is obtained by training with larger batch sizes, and the improvements become substantial for larger molecules. Right: demonstration of near-optimal weak scaling obtained by running the algorithm on the C2 molecule for 10^3 iterations with different batch sizes up to 4096 and reporting the time elapsed in seconds. Upon distributing the batch over multiple GPUs, the running time is significantly reduced. In addition, the running times for the training with a fixed batch size per GPU are close across different settings.

different sampling orders. Finally, we tried different model architectures under the same training framework; our model achieves the overall best running time efficiency in comparison with RBM [4] and NADE [5].

Parallelization. As discussed in section 3, the total number of input samples for the forward pass required to compute the local energy scales with the number of terms K in the Hamiltonian and with the batch size B, which leads to a heavy computational bottleneck as both of these factors increase. However, a sufficiently large batch size is essential to guarantee the performance of the algorithm for molecules of larger sizes. This claim is validated in the left panel of figure 2, where we train our model with batches of varying sizes for various molecules. Since the ground state energies for different molecules differ from each other, we report the performance relative the FCI ground truth.

To examine the effectiveness of our parallelization scheme, in the right panel of figure 2 we illustrate the running time as a function of batch size for C2 molecule using 10^3 iterations. We observe that the running time scales inversely with respect to the number of GPUs. In particular, doubling the number of GPUs roughly corresponds to half the time usage. We conducted additional experiments by saturating the memory on each GPU and observe that the running time for different numbers of GPUs remains constant, indicating that our approach achieves near-optimal weak scaling.

Reverse order sampling. We proposed to perform autoregressive sampling in a reversed order to improve the training. To examine the effectiveness of this approach as well as the impact of the sampling order on QC problems in general, we perform ablation studies on the sampling order. In table 2, we employ three different sampling orders: forward sampling from 1 to n, reverse sampling from n to 1, and random sampling by any pre-determined order between 1 to n. The results illustrate that reverse sampling indeed improves the performance effectively as it achieves the best results consistently across the list of molecules.

Model architecture. Our model architecture offers significant parallelization advantages compared to existing architectures based on RBM and NADE. Despite the sequential nature of both MCMC and autoregressive sampling, the latter can be executed on GPUs in a straightforward fashion and moreover exhibits superior running time efficiency. In addition, the distribution of the MCMC samples only converges to the distribution of interest asymptotically, whereas autoregressive sampling yields exact samples under a known number of iterations. NADE [9] requires n forward passes through the network to evaluate the probability, with n submodules for each entry. The main disadvantage of NADE is its sequential nature in its forward pass, which contributes to running time as the input dimension grows. In addition, computation with NADE is slower in practice compared to MADE even in low dimensions, especially for the model of high depth, due to its multi-module architectural design. In table 2, we directly compare the time taken for NADE and MADE to run for 3×10^4 iterations, using a custom implementation of NADE based on the publicly available source code [5]. In addition, we measure the time taken for each architecture to reach the performance of CCSD for each molecule. We did not include the results for RBM because the running times

Table 2. Ablation study on reverse sampling and time efficiency tests over different architectures. NADE refers to our custom implementation of the neural autoregressive density estimation proposed for QC in [5].

Molecule	H2	H2O	NH3	C2	N2	O2	HCL
Energy	Ablation study on the sampling order						
Forward	**−1.101 150**	−75.015 449	−55.515 394	−74.4849 249	−107.634 908	−147.723 681	−454.927 860
Reverse	**−1.101 150**	**−75.015 511**	**−55.521 037**	**−74.486 0377**	**−107.656 763**	**−147.749 953**	**−455.156 189**
Random	**−1.101 150**	−75.014 553	−55.519 741	−74.4851 979	−107.606 417	−147.732 876	−455.012 498
Running time (s)	Running time for 30 K iterations						
NADE	303.76	1087.52	4474.94	4574.30	2959.28	2821.60	1593.21
MADE	**282.64**	**838.60**	**3188.31**	**2922.25**	**2295.90**	**2122.21**	**1112.70**
Hitting time (s)	Hitting time to the CCSD performance						
NADE	117.28	**352.78**	2007.14	1754.10	1029.64	824.37	489.88
MADE	**100.14**	364.32	**782.46**	**827.41**	**986.27**	**648.27**	**186.38**

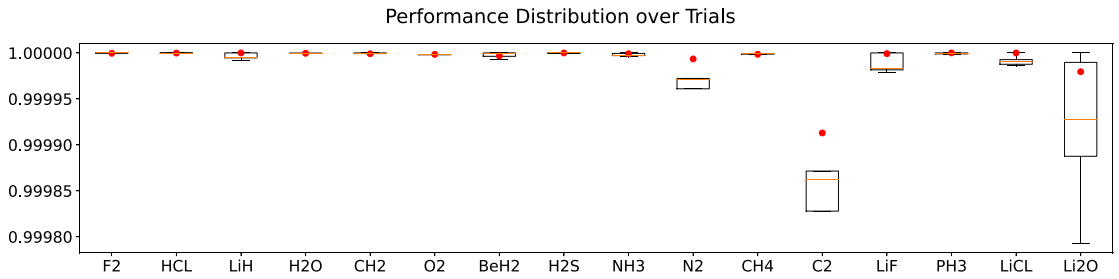

Figure 3. We report our performance on different molecules for 5 trials in the form of a box plot. For illustration purposes, we divide the results for all molecules by the corresponding FCI ground truths, so that the reported value is normalized with a maximum value of 1. In addition, we directly cite the numbers in table 1 from [5] and mark them in the form of red dots in the figure for comparison purposes. We notice that for certain molecules, the performance of our method fluctuates due to different random seeds. Nevertheless, competitive results can be obtained after a sufficient number of trials.

for RBM for 30 K iterations are exceedingly large, e.g. about four hours for H2 molecule, and therefore the direct comparison results against the other two architectures have no practical interest. We also directly cite the numbers from table 1 in [5] and compare the results with ours in figure 3 for a sanity check.

7. Conclusions

We proposed a scalable parallelization strategy to improve the VMC algorithm in the application of *ab-initio* QC. Our local energy parallelism enables the optimization for Hamiltonians of more complex molecules and our autoregressive sampling techniques out-perform the CCSD baseline and exhibit an advantage against other neural-network based algorithms in terms of running time efficiency and scalability. We further improve the performance of our model through the sampling order of the state entries to match the entanglement hierarchy among the molecule qubits. Our algorithm effectively works for molecules up to 76 qubits with millions of terms in its electronic Hamiltonian. It would be of significant interest to undertake systematic comparison of the results obtained here to continuum methods based on first quantization [26], especially in the regime $n \gg n_e$ necessary for extrapolation to experimental data.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

ORCID iD

Shravan Veerapaneni https://orcid.org/0000-0002-2294-7233

References

[1] Troyer M and Wiese U-J 2005 Computational complexity and fundamental limitations to fermionic quantum Monte Carlo simulations *Phys. Rev. Lett.* **94** 170201
[2] Coester F and Kümmel H 1960 Short-range correlations in nuclear wave functions *Nucl. Phys.* **17** 477–85
[3] Bartlett R J and Musiał M 2007 Coupled-cluster theory in quantum chemistry *Rev. Mod. Phys.* **79** 291
[4] Choo K, Mezzacapo A and Carleo G 2020 Fermionic neural-network states for *ab-initio* electronic structure *Nat. Commun.* **11** 1–7
[5] Barrett T D, Malyshev A and Lvovsky A I 2022 Autoregressive neural-network wavefunctions for *ab initio* quantum chemistry *Nat. Mach. Intell.* **4** 351–8
[6] Germain M, Gregor K, Murray I and Larochelle H 2015 Made: masked autoencoder for distribution estimation *Int. Conf. on Machine Learning* pp 881–9
[7] Zhao T, Stokes J, Knitter O, Chen B and Veerapaneni S 2021 Overcoming barriers to scalability in variational quantum Monte Carlo *The Int. Conf. for High Performance Computing, Networking, Storage and Analysis*
[8] Salakhutdinov R and Murray I 2008 On the quantitative analysis of deep belief networks *Int. Conf. on Machine Learning* pp 872–9
[9] Larochelle H and Murray I 2011 The neural autoregressive distribution estimator *Proc. 14th Int. Conf. on Artificial Intelligence and Statistics* pp 29–37
[10] Carleo G and Troyer M 2017 Solving the quantum many-body problem with artificial neural networks *Science* **355** 602–6
[11] Bengio Y and Bengio S 2000 Modeling high-dimensional discrete data with multi-layer neural networks *Advances in Neural Information Processing Systems* vol 12 pp 400–6
[12] Sharir O, Levine Y, Wies N, Carleo G and Shashua A 2020 Deep autoregressive models for the efficient variational simulation of many-body quantum systems *Phys. Rev. Lett.* **124** 020503
[13] Sharir O, Levine Y, Wies N, Carleo G and Shashua A 2020 FlowKet: an open-source library based on tensorflow for running variational Monte-Carlo simulations on GPUs (available at: https://github.com/HUJI-Deep/FlowKet)
[14] Hibat-Allah M, Ganahl M, Hayward L E, Melko R G and Carrasquilla J 2020 Recurrent neural network wave functions *Phys. Rev. Res.* **2** 023358
[15] van den Oord A, Kalchbrenner N, Espeholt L, Kavukcuoglu K, Vinyals O and Graves A 2016 Conditional image generation with PixelCNN decoders *Advances in Neural Information Processing Systems* vol 29, ed D Lee, M Sugiyama, U Luxburg, I Guyon and R Garnett (Curran Associates, Inc.)
[16] Hammond B L, Lester W A and Reynolds P J 1994 *Monte Carlo Methods in Ab Initio Quantum Chemistry* vol 1 (Singapore: World Scientific)
[17] Langhoff S 2012 *Quantum Mechanical Electronic Structure Calculations With Chemical Accuracy* vol 13 (Dordrecht: Springer)
[18] David Sherrill C and Schaefer H F III 1999 The configuration interaction method: advances in highly correlated approaches *Adv. Quantum Chem.* **34** 143–269
[19] Born M and Oppenheimer R 1927 Zur quantentheorie der molekeln *Ann. Phys., Lpz.* **389** 457–84
[20] Jordan P and Wigner E P 1993 Über das Paulische Äquivalenzverbot *Z. Physik* **47** 631–51
[21] Bravyi S B and Kitaev A Y 2002 Fermionic quantum computation *Ann. Phys., NY* **298** 210–26
[22] Lin Y, Han S, Mao H, Wang Y and Dally W J 2017 Deep gradient compression: reducing the communication bandwidth for distributed training *Int. Conf. on Learning Representations*
[23] Wang Y, Xiao J, Suzek T O, Zhang J, Wang J and Bryant S H 2009 PubChem: a public information system for analyzing bioactivities of small molecules *Nucleic Acids Res.* **37** W623–33
[24] McClean J R *et al* 2020 Openfermion: the electronic structure package for quantum computers *Quantum Sci. Technol.* **5** 034014
[25] Kingma D P and Ba J 2015 Adam: a method for stochastic optimization *Int. Conf. on Learning Representations*
[26] Pfau D, Spencer J S, Matthews A G D G and Foulkes W M C 2020 *Ab initio* solution of the many-electron Schrödinger equation with deep neural networks *Phys. Rev. Res.* **2** 033429