JCTC Journal of Chemical Theory and Computation

pubs.acs.org/JCTC Article

Graphics Processing Unit-Accelerated Semiempirical Born Oppenheimer Molecular Dynamics Using PyTorch

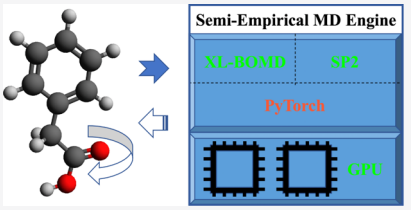
Guoqing Zhou, Ben Nebgen,* Nicholas Lubbers, Walter Malone, Anders M. N. Niklasson, and Sergei Tretiak

Cite This: *J. Chem. Theory Comput.* 2020, 16, 4951−4962

Read Online

ACCESS | Metrics & More | Article Recommendations

ABSTRACT: A new open-source high-performance implementation of Born Oppenheimer molecular dynamics based on semiempirical quantum mechanics models using PyTorch called PYSEQM is presented. PYSEQM was designed to provide researchers in computational chemistry with an open-source, efficient, scalable, and stable quantum-based molecular dynamics engine. In particular, PYSEQM enables computation on modern graphics processing unit hardware and, through the use of automatic differentiation, supplies interfaces for model parameterization with machine learning techniques to perform multiobjective training and prediction. The implemented semiempirical quantum mechanical methods (MNDO, AM1, and PM3) are described. Additional algorithms include a recursive Fermi-operator expansion scheme (SP2) and extended Lagrangian Born Oppenheimer molecular dynamics allowing for rapid simulations. Finally, benchmark testing on the nanostar dendrimer and a series of polyethylene molecules provides a baseline of code efficiency, time cost, and scaling and stability of energy conservation, verifying that PYSEQM provides fast and accurate computations.



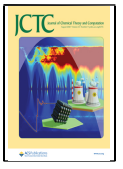
1. INTRODUCTION

Semiempirical quantum mechanics (SEQM) models are widely used to efficiently study ground- and excited-state electronic properties of chemical systems and materials.[1,2] Although high-level methods such as configuration interaction (CI), coupled cluster (CC), and density functional theory (DFT) can have a higher accuracy than SEQM, the former techniques scale cubically to exponential with the system size and have a large computational overhead, typically limiting them to systems with 10−100 atoms.[3,4] SEQM models utilize very minimal basis sets and parameterize various integrals in the electronic Hamiltonian with experimental or ab initio references,[5−9] resulting in methods that are much faster than traditional wave function or DFT frameworks while maintaining a reasonable level of accuracy. The increased speed of SEQM techniques allows them to be applied to systems traditionally outside the reach of quantum mechanical methods such as proteins,[2] nanotubes,[10] and many noncovalent complexes.[11,12] However, despite the many decades of research into SEQM methods,[1,5,6,8,9,13,14] the limitations of the functional form for the underlining reduced electronic Hamiltonian models leave much room for improvement in their accuracy.[8,14]

Over the last several decades, several distinct groups of semiempirical methods have been developed, including modified neglect of diatomic overlap (MNDO),[5] Austin model 1 (AM1),[6] MNDO/d,[13] Parametric model 6 (PM6),[8] and PM7.[14] These approaches were developed in series to improve the description of core−core interactions and to produce more accurate parameter schemes. To accurately describe noncovalent bond interaction, Hobza and co-workers extended PM6 with models such as PM6-DH,[15] PM-DH2,[16] and PM-DH+.[17] Moreover, a series of recently developed orthogonalization-corrected methods (OMx) reach higher accuracy and have better descriptions of noncovalent bonds by taking into account repulsive orthogonalization, attractive penetration effects, and repulsive core−valence and dispersion interactions.[9] Although these developments have resulted in significant improvements in effective Hamiltonian models, they still lack the accuracy and generality compared to conventional high-level quantum methods.[18,19] This poses a need for further improvements of SEQM using modern data science approaches as was demonstrated for the parameterization of reduced density functional tight-binding (DFTB) models.[20]

In the last couple of decades, graphics processing units (GPUs) have demonstrated their ability to accelerate numerically intense computations. The matrix operations that comprise many parts of SEQM methods can strongly benefit from such GPU offloading.[1] For example, the most time-consuming component of SEQM for mean-field ground-state

Received: March 12, 2020
Published: July 1, 2020



calculations, the self-consistent field (SCF) procedure, can be sped up significantly through the use of novel numerical methods, such as the single-particle density matrix expansion algorithm with second-order spectral projection polynomials (SP2).[21] GPU implementations of these approaches, which used to require highly specialized knowledge of the layout and structure of GPUs, have become much simpler through the use of high-level packages such as PyTorch,[22] a package originally designed for machine learning (ML) applications. Furthermore, the incorporation of reverse-mode automatic differentiation,[22] which can compute the gradient of a quantity with respect to all parameters at the same computational cost as computing the quantity itself, has facilitated the solution of complex optimization problems. All of these developments have contributed to the rising popularity of ML in many areas including but not limited to physics, chemistry, and material science. Applications in these fields range from predicting various properties including atomic partial charge,[23] molecular dipoles,[24] atomization energy,[25] generating fast and highly accurate potential energy surfaces and forces,[26] modeling chemical reactions,[27] and discovering new materials.[28]

Unfortunately, many of these applications suffer from the "black box" problem, where a deep neural network (DNN) makes an accurate prediction but provides essentially no explanation as to why that prediction was made. A possible solution to this transparency problem is to build quantum mechanical models on top of a DNN for modeling molecular systems. Yaron et al.[20] demonstrated a DNN linked with a self-consistent DFTB layer and successfully reduced the error in DFTB by 40−60% when predicting total molecular energy and electronic dipole moment. Zubatyuk and co-workers linked a simple extended Hückel model with HIPNN[29] to dynamically generate parameters for the semiempirical model which facilitated electronic structure predictions comparable to DFT calculations.[30] Future work similar to these developments would tremendously benefit from a general software platform combining an efficient implementation of model Hamiltonians in a programming platform which also natively supports ML techniques; SEQM aims to fill this role.

In this article, we present an implementation of various SEQM methods utilizing PyTorch (PYSEQM) in an open-source software package.[31] PYSEQM supports a variety of SEQM Hamiltonians including MNDO, AM1, PM3, PM6, and extended Hückel theory.[22] Additionally, it contains a simple molecular dynamics (MD) driver to facilitate the simulation of dynamic properties and nonequilibrium configurations for molecular systems. Furthermore, PYSEQM is capable of processing multiple molecules simultaneously on a GPU (i.e., run multiple MD trajectories in parallel) to get energy and gradient information, which may be of use in future ML applications or in various theoretical methodologies requiring ensemble averaging of a swarm of trajectories.[32] Because PyTorch supports a wide range of execution options on GPUs, PYSEQM can easily be deployed on a wide range of GPU devices to achieve high performance. Finally, we have implemented extended Lagrangian Born Oppenheimer molecular dynamics (XL-BOMD)[33−37] for accelerated MD simulations that avoid the costly SCF algorithm and the recursive Fermi-operator density matrix expansion algorithm (SP2) for a rapid SCF convergence utilizing GPU devices.[21,38−40] The largest limitations of PYSEQM are that it currently only works for closed shelled systems and that it only supports s and p orbitals, although an extension to d orbitals is currently under development. Additionally, the SP2 scheme currently implemented in PYSEQM will fail with bond breaking; adding newer versions of the scheme would address this shortcoming.[41]

We detail the implementations of these semiempirical models as well as XL-BOMD and SP2 in Section 2. In Section 3, we show the performance of PYSEQM on two groups of systems: the nanostar dendrimer[42] and various lengths of polyethylene chains. The testing on the nanostar demonstrates the performance of PYSEQM on large systems in addition to the enforcement of energy conservation with the XL-BOMD scheme. The second test set with polyethylenes shows the scaling of this code with system size and demonstrates situations where PYSEQM outperforms conventional semiempirical codes. Finally, we summarize the results and conclude in Section 4.

2. METHODS

In this section, we detail the features of the PYSEQM package. First, we examine the form of the three unique semiempirical methods implemented in PYSEQM: MNDO, AM1, and PM3 for elements from H to Cl covering organic molecular systems. Although not explored in this paper, PM6 with d orbitals is also under development. These effective Hamiltonians are constructed with a set of empirical parameters fit to experimental data to compensate for the neglect of electron-correlation inherent in all single reference (mean-field) methods. Second, we detail the SP2 method for rapid density matrix determination directly from a Hamiltonian matrix, bypassing matrix diagonalization. SP2 expands the density matrix in terms of the Hamiltonian Operator with an iterative scheme, which is much more efficient on modern GPU architectures than traditional diagonalization and density matrix construction procedures.[21] Finally, we outline the stable and time-reversible XL-BOMD algorithm, which ensures energy conservation and dissipates numerical error while simultaneously avoiding the overhead of an iterative SCF optimization required in regular BOMD.[34]

2.1. Semiempirical Quantum Chemistry Methods. All semiempirical methods implemented in PYSEQM use atomic orbitals (AOs, $\phi_\mu(\boldsymbol{r})$) as basis functions, with most elemental basis sets only containing valence shell AOs. Slater-type AOs are used by all of these methods and are detailed here

$$\phi_{nlm} = R_{nl}(r) y_{lm}(\theta, \phi) \tag{1}$$

$$R_{nl}(r) = (2\zeta_{nl})^{n+1/2}[(2n)!]^{-1/2} r^{n-1} e^{-\zeta_{nl} r} \tag{2}$$

where n, l, and m are the quantum numbers for $\phi_\mu(\boldsymbol{r})$, $R_{nl}(\boldsymbol{r})$ is the radial function with the orbital exponent ζ_{nl}, and $y_{lm}(\theta,\phi)$ is the real spherical harmonic function.

The molecular orbitals (MOs, $\psi_i(\boldsymbol{r})$) are expressed as a linear combination of AOs

$$\psi_i(r) = \sum_\mu C_{i\mu} \phi_\mu(r) \tag{3}$$

The expansion coefficients $C_{i\mu}$ are obtained by solving the Roothaan−Hall equation[43]

$$\boldsymbol{FC} = \boldsymbol{SCE} \tag{4}$$

where $\boldsymbol{F}$ is the Fock Matrix, $\boldsymbol{C}$ is the matrix of $C_{i\mu}$, $\boldsymbol{E}$ is the diagonal matrix with MO energies, and $\boldsymbol{S}$ is the overlap matrix. These semiempirical methods make the zero-differential

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

overlap (ZDO) approximation,[1] setting $\boldsymbol{S} = \boldsymbol{I}$ and reducing the equation to

$$\boldsymbol{FC} = \boldsymbol{CE} \tag{5}$$

$\boldsymbol{F}$ is composed of one-electron ($\boldsymbol{h}$) and two-electron ($\boldsymbol{G}$) parts, which for closed shell systems are expressed as

$$\boldsymbol{F}(\boldsymbol{D}) = \boldsymbol{h} + \boldsymbol{G}(\boldsymbol{D}) \tag{6}$$

$$h_{\mu\nu} = \left\langle \mu \middle| -\frac{1}{2}\nabla^2 \middle| \nu \right\rangle - \sum_A Z_A \left\langle \mu \middle| \frac{1}{R_A} \middle| \nu \right\rangle \tag{7}$$

$$G_{\mu\nu} = \sum_{\lambda\sigma} D_{\lambda\sigma}\left[(\mu\nu, \lambda\sigma) - \frac{1}{2}(\mu\lambda, \nu\sigma)\right]$$

$$(\mu\nu, \lambda\sigma) = e^2 \int\int \phi_\mu(\boldsymbol{r}_1)\phi_\nu(\boldsymbol{r}_1)\frac{1}{r_{12}}\phi_\lambda(\boldsymbol{r}_2)\phi_\sigma(\boldsymbol{r}_2)\mathrm{d}\boldsymbol{r}_1\mathrm{d}\boldsymbol{r}_2 \tag{8}$$

where Z_A is the effective charge for nuclei A with core shell electrons, $1/R_A$ is the one-electron potential energy operator, and $-\frac{1}{2}\nabla^2$ is the kinetic energy operator. The density matrix $D_{\lambda\sigma}$ can be computed from the molecular orbital coefficients of the occupied states in the following way

$$D_{\lambda\sigma} = 2\sum_i C_{i\lambda}C_{i\sigma} \tag{9}$$

The MNDO, AM1, and PM3 methods utilize the neglect of differential-diatomic overlap (NDDO) approximation, where integrals $\phi_\mu(\boldsymbol{r})\phi_\nu(\boldsymbol{r})\mathrm{d}\boldsymbol{r}$ are not vanishing only if $\phi_\mu(\boldsymbol{r})$ and $\phi_\nu(\boldsymbol{r})$ are centered on the same atom. With NDDO, all three- and four-center integrals in $\boldsymbol{G}$ are ignored, reducing the total number of integrals from $O(N^4)$ in $\boldsymbol{G}$ to $O(N^2)$, where N is the total number of atoms.

With the NDDO approximation in the MNDO-based model, the one-electron matrix $\boldsymbol{h}$ is further approximated as

$$h_{\mu\nu} = \begin{cases} U_{\mu\mu} - \sum_{B\neq A} Z_B(\mu\mu, s_B s_B) & \mu = \nu \\ -\sum_{B\neq A} Z_B(\mu\nu, s_B s_B) & \mu, \nu \text{ centered on atom A} \\ \beta_{\mu\upsilon} = \frac{(\beta_\mu + \beta_\nu)}{2} S_{\mu\nu} & \text{otherwise} \end{cases} \tag{10}$$

where $|s_B\rangle$ is the s orbital of the valence shell for atom B and the two-center one-electron integral from the interaction with the core shell of atom B is evaluated as $\langle\mu|1/R_B|\nu\rangle = (\mu\nu, s_B s_B)$ if μ and ν are centered on the same atom. $U_{\mu\mu} = \langle\mu|-\nabla^2/2|\mu\rangle - Z_A\langle\mu|1/R_A|\mu\rangle$ is the onsite energy for orbital ϕ_μ on atom A and is parameterized for each element. The resonance integrals $\beta_{\mu\nu}$ are approximated as the average of each orbital's β parameter multiplied by the corresponding term in the overlap matrix.

With NDDO, the two-electron integral $(\mu\nu, \lambda\sigma)$ in $\boldsymbol{G}$ (eq 8) is not to be ignored only if both pairs of orbitals (ϕ_μ, ϕ_ν) and $(\phi_\lambda, \phi_\sigma)$ are located on the same atoms, respectively. For evaluating $(\mu\nu, \lambda\sigma)$, a classical approximation is used. $(\mu\nu, \lambda\sigma)$ describes the electron interaction energy between charge distribution $e\phi_\mu\phi_\nu$ on atom A and $e\phi_\lambda\phi_\sigma$ on atom B. $e\phi_\mu\phi_\nu$ is represented by a set of multipoles $\{M_{lm}^A\}$, and each multipole is represented by a configuration of a set of point charges with net charge e and multipole M_{lm}^A.[44] The integrals are evaluated as

$$(\mu\nu, \lambda\sigma) = \sum_{l_1}\sum_{l_2}\sum_m [M_{l_1m}^A, M_{l_2m}^B] = \frac{e^2}{2^{l_1+l_2}}\sum_{i=1}^{2^{l_1}}\sum_{j=1}^{2^{l_2}} f(R_{ij}) \tag{11}$$

where l is the multipole order, m is the multipole tensor index, and $f(R)$ is the Coulomb interaction between each unit point charge. To satisfy the asymptotic behavior with $R \to 0^+, +\infty$, Dewar–Sabelli–Klopman (DSK) approximation[5,44] is used

$$f(R_{ij}) = [{R_{ij}}^2 + (\rho_{l_1}^A + \rho_{l_2}^B)^2]^{-1/2} \tag{12}$$

where ρ_l^A is the additive term for atom A with the multipole order l. The value of ρ_l^A is derived for each atomic species in the limit $R \to 0^+$ for the same type of atom A and B to yield the correct direct and exchange integrals $g_{\mu\nu}$, $h_{\mu\nu}$

$$\lim_{R_{AB}\to 0^+}(\mu_A\mu_A, \nu_B\nu_B) = (\mu_A\mu_A, \nu_A\nu_A) = g_{\mu\nu} \tag{13}$$

$$\lim_{R_{AB}\to 0^+}(\mu_A\nu_A, \mu_B\nu_B) = (\mu_A\nu_A, \mu_A\nu_A) = h_{\mu\nu} \tag{14}$$

here $g_{\mu\nu}$ and $h_{\mu\nu}$ are parameterized from experimental or ab initio calculation data and they are smaller than the analytic values as to compensate for the neglect of there- and four-center Coulomb integrals and the exclusion of electron correlation in HF.[5,6]

Apart from the parameterization in the Hamiltonian, the only difference between MNDO, AM1, and PM3 is the nuclear–nuclear interaction term. The nuclear energy between atom A and B in MNDO is

$$E_{\mathrm{nuc}}^{AB} = Z_A Z_B(s^A s^A, s^B s^B)[1 + \mathrm{e}^{-\alpha_A R_{AB}} + \mathrm{e}^{-\alpha_B R_{AB}}] \tag{15}$$

where α_A and α_B are atomic parameters for atom A and B, respectively. R_{AB} is simply the distance between the two atoms. In contrast, AM1 and PM3 have the following nuclear–nuclear interaction term

$$E_{\mathrm{nuc}}^{AB} = Z_A Z_B(s^A s^A, s^B s^B)[\mathrm{e}^{-\alpha_A R_{AB}} + \mathrm{e}^{-\alpha_B R_{AB}} + F_A(R_{AB}) + F_B(R_{AB})] \tag{16}$$

$$F_A(R_{AB}) = \sum_i^{m_A} K_A^i \exp[L_A^i(R_{AB} - M_A^i)^2] \tag{17}$$

where K_A^i, L_A^i, and M_A^i are sequences of atom-type-dependent parameters for the Gaussian expansion of $F_A(R_{AB})$. In AM1, there are 2–4 terms in this sum, depending on the element, while for PM3, there are only 2. With this, the total energy of the system is then

$$E_{\mathrm{tot}} = E_{\mathrm{elec}} + \sum_{A<B} E_{\mathrm{nuc}}^{AB} \tag{18}$$

where the electronic energy is

$$E_{\mathrm{elec}}(\boldsymbol{R}, \boldsymbol{D}) = \frac{1}{2}\mathrm{Tr}[\boldsymbol{D} \times (\boldsymbol{h} + \boldsymbol{F}(\boldsymbol{D}))] \tag{19}$$

Equation 5 is solved iteratively using an SCF loop. A trial density matrix $\boldsymbol{D}^0$ is constructed with the valence charge of each atom evenly distributed on its valence shell orbitals. Other strategies of initialization $\boldsymbol{D}^0$ adopted in other packages

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951–4962

include using a faster QM method first, like the Hückel model, or even distribution of the electron density with a small noise. During each step k, $\boldsymbol{D}^{k-1}$ is used to form the Fock matrix $\boldsymbol{F}(\boldsymbol{D}^{k-1})$ based on eq 6, and the density $\boldsymbol{D}^k$ is constructed either by diagonalizing $\boldsymbol{F}(\boldsymbol{D}^{k-1})$ or by some expansion scheme as discussed below. The new density for the next step $\boldsymbol{D}^k$ is formed by linear mixing of $\boldsymbol{D}^k$, $\boldsymbol{D}^{k-1}$, $\boldsymbol{D}^{k-2}$, ...$\boldsymbol{D}^{k-m}$, with the mixing coefficients being evaluated by linear interpolation algorithms such as adaptive mixing[45] and Pulay.[46] Alternatively, it can be constructed with algorithms such as Camp-King[47] or SOSCF.[48] The loop can be stopped with a variety of different convergence criterion, such as a small ΔE_{tot} between subsequent iterations, a small $\Delta \boldsymbol{D}$ between subsequent iterations, or when the density matrix and the Fock matrix commute to within some threshold $\varepsilon([\boldsymbol{D}^k, \boldsymbol{F}(\boldsymbol{D}^k)] = \boldsymbol{S}\boldsymbol{D}^k F(\boldsymbol{D}^k) - \boldsymbol{F}(\boldsymbol{D}^k)\boldsymbol{D}^k\boldsymbol{S} < \varepsilon)$.

The forces acting on the atoms are further computed as the negative gradient of total energy with respect to the nuclear coordinates. For BOMD, the electrons are assumed to stay in the ground electronic state, and because of Hellmann–Feynman theorem, the gradient on the density matrix is assumed to be zero. Using PyTorch's Automatic Differentiation features,[22] the analytic gradient is automatically obtained as the total energy is computed. Automatic differentiation does not rely on finite differences to compute gradients, instead constructing a computational graph and saving the required intermediate variables for the backward gradient calculations, using the chain rule to calculate gradients in an efficient manner.

2.2. SP2 Algorithm. For each SCF loop, a new density matrix $\boldsymbol{D}^k$ must be constructed from the Fock matrix $\boldsymbol{F}^k$. Traditionally, this is done by diagonalizing $\boldsymbol{F}^k$ and forming $\boldsymbol{D}^k$ using the eigenvectors based on eq 9. As an alternative approach, we can directly form $\boldsymbol{D}^k$ from $\boldsymbol{F}^k$ without matrix diagonalization using the property that $\boldsymbol{D}^k$ (in an orthogonalized matrix representation) is given by a step function of $\boldsymbol{F}^k$ (in an orthogonalized matrix representation) with the step formed at the chemical potential, as is described in eq 20 below. A step function expansion can then be performed using various recursive Fermi-operator schemes that avoid any explicit diagonalization. The simplest and possibly most efficient scheme is based on second-order spectral projection polynomials, known as the SP2 algorithm. The computational kernel of the SP2 scheme that dominates the cost is a generalized matrix–matrix multiplication, which can make efficient use of hardware acceleration, such as GPUs.[39] To begin, $\boldsymbol{D}^k$ can be rewritten as

$$\boldsymbol{D}^k = \theta(\mu \boldsymbol{I} - \boldsymbol{F}^k), \quad \mathrm{Tr}[\boldsymbol{D}^k] = N_{\text{occ}} \tag{20}$$

where $\theta(x)$ is the Heaviside step function and μ and N_{occ} are the chemical potential and the occupation number of the system, respectively. Generally, $\theta(x-\mu)$ can be expanded and approximated by a serial polynomial expansion, for example, with Chebyshev polynomials, with the step centered on a predefined μ.[49] The SP2 algorithm approximates the step function with recursive expansion based on second-order spectral projection polynomials given in eq 23, which have the stationary end points at 0 and 1, without requiring prior knowledge of μ.

In the SP2 scheme, the Fock matrix $\boldsymbol{F}^k$ is first scaled so the eigen spectrum lies on the interval $[0,1]$

$$\boldsymbol{X}^0 = \frac{(\varepsilon_N \boldsymbol{I} - \boldsymbol{F}^k)}{(\varepsilon_N - \varepsilon_1)} \tag{21}$$

where ε_1 and ε_N are the estimates of the smallest and largest eigenvalues of $\boldsymbol{F}$, respectively. ε_1 and ε_N can be estimated, for example, with Gershgorin Circle Theorem,[49] which dictates that all the eigenvalues of the Hermitian matrix $\boldsymbol{F}$ lie in the interval on the real axis

$$[\min_i(R_i - F_{ii}), \max_i(R_i + F_{ii})], \qquad R_i = \sum_{j \neq i} |F_{ij}| \tag{22}$$

where R_i and F_{ii} are the radius and center of Gershgorin Disc i, respectively. The estimated bounds are used to replace ε_1 and ε_N in the scaling of $\boldsymbol{F}$ as $\varepsilon_{1/N}$ are not known a priori.

A quadratic form of the spectral projection (or purification) polynomials $P^{(a)}$ and $P^{(b)}$ is defined as

$$P^{(a)}(\boldsymbol{X}) = \boldsymbol{X}^2, \qquad P^{(b)}(\boldsymbol{X}) = 2\boldsymbol{X} - \boldsymbol{X}^2 \tag{23}$$

These projection polynomials are applied iteratively to $\boldsymbol{X}^{(0)}$ to generate a converging sequence of $\boldsymbol{X}^{(n)}$ under the following rules

$$\boldsymbol{X}^{(n+1)} = \begin{cases} P^{(a)}(\boldsymbol{X}^{(n)}), & |\mathrm{Tr}[P^{(a)}(\boldsymbol{X}^{(n)})] - N_{\text{occ}}| \\ & \quad < |\mathrm{Tr}[P^{(b)}(\boldsymbol{X}^{(n)})] - N_{\text{occ}}| \\ P^{(b)}(\boldsymbol{X}^{(n)}), & \text{otherwise} \end{cases} \tag{24}$$

In this way, the eigenvalues of $\boldsymbol{X}^{(n)}$ converge to the step function of H as $n \to \infty$, while $\mathrm{Tr}(\boldsymbol{X}^{(n)})$ converges to the occupied number of orbitals N_{occ}. The iterative procedure is stopped when the error in total electron count at step n and $n-1$ is smaller than 10^{-4}. The error in the electron number at step n is defined as

$$\mathrm{err}(n) = |\mathrm{Tr}(\boldsymbol{X}^{(n)}) - N_{\text{occ}}| \tag{25}$$

which is used as a convergence test together with the idempotency error that can be estimated by $\mathrm{Tr}[P(I-P)]$. Typically, 20–30 iterations are required before a sufficient convergence is achieved. Finally, the density matrix can be formed from the converged $\boldsymbol{X}^{(n)}$

$$\boldsymbol{D}^k = 2 \lim_{n \to \infty} \boldsymbol{X}^{(n)} \tag{26}$$

where the factor of 2 accounts for the spin degeneracy in a closed shell system.

2.3. Extended Lagrangian BOMD. In conventional Born Oppenheimer molecular dynamics, the motion of the nuclei is described by the Lagrangian

$$\mathcal{L}^{\text{BO}}(\boldsymbol{R}, \dot{\boldsymbol{R}}) = \frac{1}{2}\sum_k m_k \dot{R}_k^{\,2} - U(\boldsymbol{R}, \boldsymbol{D}) \tag{27}$$

Here, $U(\boldsymbol{R}, \boldsymbol{D})$ is the potential energy surface with the ground-state density matrix $\boldsymbol{D}$. For a given $\boldsymbol{R}$, U is the total energy defined in eq 18. In regular BOMD, an extrapolation of ground-state density matrices from previous time steps is used as an initial guess to the SCF optimization procedure. This extrapolation, followed by an SCF optimization, which in practice is never complete, breaks the time-reversibility of the underlying propagation of the electronic degrees of freedom.

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951–4962

This common form of direct BOMD simulation typically leads to an unphysical systematic drift in the total energy and phase space.[50] The problem can be reduced or removed by tightening the SCF convergence or by ignoring the extrapolations and instead starting each new SCF optimization from scratch using, for example, overlapping atomic densities in each new time step. This may substantially increase the computational cost. To avoid the shortcomings of regular direct BOMD, a time-reversible formulation can be constructed by introducing an auxiliary electronic density matrix $\boldsymbol{P}$ as a dynamical tensor variable that evolves in addition to the nuclear coordinates and their velocities. This can be achieved using XL-BOMD,[33−37] which is defined through an extended Lagrangian

$$\mathcal{L}^{\mathrm{XLBO}}(\boldsymbol{R}, \dot{\boldsymbol{R}}, \boldsymbol{P}, \dot{\boldsymbol{P}}) = \frac{1}{2}\sum_k m_k \dot{R}_k^{\,2} - U(\boldsymbol{R}, \boldsymbol{P}) + \frac{\mu}{2}\mathrm{Tr}[\dot{\boldsymbol{P}}^2] - \frac{\mu\omega^2}{2}\mathrm{Tr}[(\boldsymbol{D}-\boldsymbol{P})^T\boldsymbol{K}^T\boldsymbol{K}(\boldsymbol{D}-\boldsymbol{P})] \tag{28}$$

Here, μ is a fictitious electron mass parameter, ω is the frequency parameters for the electronic degree of freedom, and $\boldsymbol{K}^T\boldsymbol{K}$ is a metric tensor of the harmonic well. These variables dictate how the auxiliary electron density matrix $\boldsymbol{P}$ fluctuates through a harmonic oscillator located around an optimized electron density matrix $\boldsymbol{D}$. The modified "shadow" potential energy surface $U(\boldsymbol{R},\boldsymbol{P})$ is defined through a constrained density matrix minimization as

$$U(\boldsymbol{R}, \boldsymbol{P}) = E^{(1)}_{\mathrm{elec}}(\boldsymbol{R}, \boldsymbol{P}) + E_{\mathrm{nuc}}(\boldsymbol{R}) \tag{29}$$

$$E^{(1)}_{\mathrm{elec}}(\boldsymbol{R}, \boldsymbol{P}) = \frac{1}{2}(2\mathrm{Tr}[\boldsymbol{h}\boldsymbol{D}(\boldsymbol{P})] + \mathrm{Tr}[(2\boldsymbol{D}(\boldsymbol{P}) - \boldsymbol{P})\boldsymbol{G}(\boldsymbol{P})]) \tag{30}$$

The P dependency of $D(P)$ is dropped in most part of the text for simplicity. It is shown by Niklasson and Cawkwell[36] that the difference between $\boldsymbol{D}$ and $\boldsymbol{P}$ scales as $(\boldsymbol{D}-\boldsymbol{P}) = \boldsymbol{O}(\Omega^2/\omega^2)$, where Ω is the highest characteristic frequency of nuclear motion. In an adiabatic limit where $\mu \to 0$, with $\mu\omega$ being constant, assuming $\omega \gg \Omega$, and with $\boldsymbol{K}$ chosen as a superoperator given by the inverse Jacobian[36] of the residual $\boldsymbol{D}-\boldsymbol{P}$, the Euler−Lagrange equations of motion for $\mathcal{L}^{\mathrm{XLBO}}$ that govern the time evolution of $\boldsymbol{P}$ and $\boldsymbol{R}$ are given by

$$m_k\ddot{R}_k = -\frac{\partial U(\boldsymbol{R}, \boldsymbol{P})}{\partial R_k} = \frac{1}{2}\mathrm{Tr}\left[2\frac{\partial \boldsymbol{h}}{\partial R_k}\boldsymbol{D} + (2\boldsymbol{D}-\boldsymbol{P})\frac{\partial \boldsymbol{G}(\boldsymbol{P})}{\partial R_k}\right] \tag{31}$$

$$\ddot{\boldsymbol{P}} = -\omega^2\boldsymbol{K}(\boldsymbol{D}-\boldsymbol{P}) \tag{32}$$

where $\boldsymbol{D}(t)$ is obtained at the constrained minimization, as the lowest stationary solution of the linearized energy expression in eq 30. Thanks to the linearization, the minimization is achieved in a single step and no iterative SCF optimization is involved. In all of our calculations, we will use a scaled delta function approximation of the kernel $\boldsymbol{K}$, where it is replaced by $-\mathrm{c}\times\boldsymbol{I}$, with c in the interval [0,1]. The initial value of $\boldsymbol{P}(t)$ is given by a full regular SCF optimization of $\boldsymbol{D}(0)$, that is, where $\boldsymbol{P}(0) = \boldsymbol{D}(0)$ with $\dot{\boldsymbol{P}}(0) = 0$. The nuclear equations of motions are integrated with a symplectic velocity Verlet scheme.[51] Eq 32 shows that $\boldsymbol{P}$ oscillates on a harmonic surface centered around the optimized density matrix $\boldsymbol{D}$. In the adiabatic limit, $(\boldsymbol{D}-\boldsymbol{P}) = \boldsymbol{O}(\Omega^2/\omega^2)$. The equations of motion for the electronic degrees of freedom, $\boldsymbol{P}$, can be integrated using the standard Verlet scheme[51]

$$\boldsymbol{P}(t+\Delta t) = 2\boldsymbol{P}(t) - \boldsymbol{P}(t-\Delta t) + \Delta t^2\ddot{\boldsymbol{P}} \tag{33}$$

As the integration is time-reversible, the SCF optimization in eq 33 keeps the time-reversal symmetry in the evolution of $\boldsymbol{D}(t)$. The systematic drift of total energy and in phase space due to breaking time-reversal symmetry in BOMD can thus be eliminated with the XL-BOMD framework. However, the time-reversible integration of XL-BOMD introduces a problem because of internal numerical noise from an approximate matrix algebra based on finite floating-point operations.[34] A time-reversible integration scheme does not dissipate any numerical noise. Numerical error therefore accumulates and may eventually lead to a significant deviation of $\boldsymbol{P}$ from $\boldsymbol{D}$. This subsequently results in a drift of the potential energy surface $U(\boldsymbol{R}, \boldsymbol{P})$ when performing MD simulations for long periods of time. To avoid this noise accumulation, a dampening force term is added to the Verlet integration of $\boldsymbol{P}(t)$

$$\boldsymbol{P}(t+\Delta t) = 2\boldsymbol{P}(t) - \boldsymbol{P}(t-\Delta t) + \Delta t^2\ddot{\boldsymbol{P}} + \alpha\sum_{k=0}^{K} c_k\boldsymbol{P}(t-k\Delta t) \tag{34}$$

The dissipation is introduced by the last summation term that dampens out noise accumulation. Here, α and c_k are the parameters with their values chosen such that the error is reduced while keeping the time-reversibility to a higher odd order, $\mathrm{O}(\Delta t^{(2K-3)})$, of the integration time step. Their optimized values can be found in ref 34 together with the value of $\kappa = \Delta t^2\omega^2$. The integration scheme dissipates numerical errors and generates stable MD trajectories with well-controlled long-term energy conservation.

3. RESULTS AND DISCUSSION

To test the models and algorithms implemented in the PYSEQM package and benchmark performance, we run conventional BOMD and XL-BOMD on two model systems. The first system, a "nanostar" phenylene−ethynylene dendrimer with 884 atoms,[42] is used as a showcase molecule. The second system, 8 polyethylene molecules with sizes ranging from H-$(C_2H_4)_1$-H to H-$(C_2H_4)_{128}$-H, is used to illustrate the scaling of the code with a system size. Benchmarks are performed on both central processing unit (CPU) and GPU architectures with double precision. To put these benchmarks in perspective of conventional implementation of SEQM, the results are compared with a standard quantum chemistry package (ORCA).[52] As PyTorch relies on an asynchronous execution when utilizing GPU, synchronizing calls are used to enforce accurate GPU timings, and based on the performance, adding synchronizing slows down the performance by around 2%.

3.1. Nanostar Dendrimer. The nanostar tested here is a dendrimer with two unique chemical environments, a core ethynylperylene chromophore and phenylene-ethynylene terminals.[42] Because of potential wide applications and fruitful experimental and theoretical studies on this type of

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

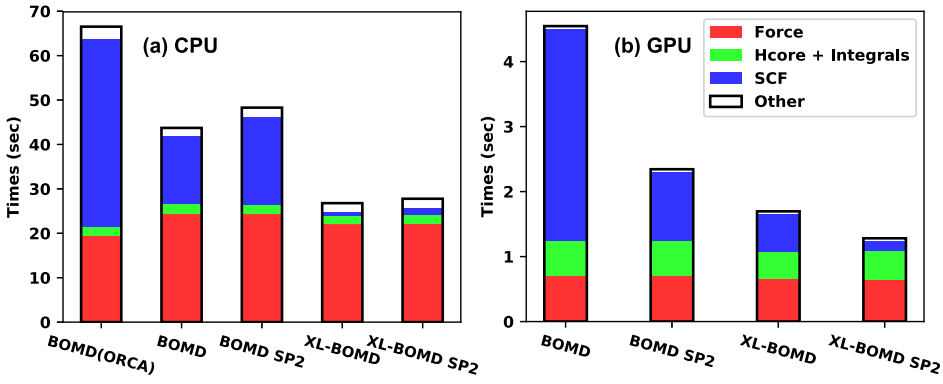


Figure 1. Stacked histogram of the average time spent computing forces (red), Hcore and integrals (green), the SCF loop (blue), and other codes (white) per MD step with the nanostar dendrimer. This is both with and without SP2 and the XL method on an Intel Xeon E5 CPU (a) and a Nvidia Tesla V100 (b). The left most column is a BOMD computed with ORCA on the CPU for comparison.

dendrimer,[42,53,54] the nanostar, containing 460 C and 424 H atoms in total, makes a good benchmark system because of its size. The structure was initially equilibrated at 300 K for 50 ps using the AIREBO force field[55] in the LAMMPS molecular dynamics (MD) package.[56] Initializing from the equilibrated structure and randomly assigned velocities, we further run MD for 200 steps with a 0.2 fs time step. Simulations for all combinations of BOMD/XL-BOMD, SP2/conventional diagonalization, and CPU/GPU are performed to get a comprehensive measure of code performance under a variety of different simulation conditions. The simulations are tested on one Intel Xeon E5-2660_v3 CPU core and one Nvidia Tesla V100-SXM2 GPU. The convergence criteria for the SCF procedure in BOMD are set to be 10^{-6} E_h (Hartree Energy). Finally, a BOMD simulation performed with ORCA is run for a timing comparison.

Figure 1 shows the average time cost per MD step in both BOMD and XL-BOMD. Additionally, these timings are reported separately for CPU and GPU execution. Modern GPUs are capable of operating in the "single-instruction multiple data" (SIMD) mode across thousands of cores on a single GPU, resulting in a factor of 10−20 speed up when operating on a GPU versus a CPU, as shown in Figure 1a,b. In Figure 1a, we report the execution time of a BOMD simulation performed on a CPU in both PYSEQM and ORCA software. The PYSEQM calculation is approximately 1.5 times faster than ORCA, even though ORCA utilizes a more sophisticated SCF algorithm that generally requires fewer iterations to achieve convergence than the SCF algorithm used in PYSEQM. The improvement is mainly coming from the SCF procedure, as computing force, Hcore, and Coulombic integrals take approximately the same amount of time between the two codes. This is due to implementation details: ORCA utilizes the second-order SCF orbital optimization (SOSCF)[57] algorithm, which is more numerically stable (see our discussion in the next section) and costly. The time spent in the SCF cycle is greatly reduced by the XL-BOMD algorithm because it only requires a single matrix diagonalization to obtain the correct density. This leads to approximately a 95% reduction in computation time for CPU calculations and an 85% reduction for GPU computations for the SCF loop. This translates to an overall factor of 2 speedup for the full MD propagation. The SP2 algorithm performs differently on the CPU and GPU: it is slightly slower on the CPU but 3 times faster on the GPU. The SP2 algorithm performs much better on GPU architectures because the dominant operations in this algorithm are matrix−matrix multiplications, which benefit greatly from GPU acceleration. *Overall, PYSEQM utilizing the XL-BOMD and SP2 algorithms running on a GPU is more than 50 times faster than ORCA running on a CPU for the nanostar system.*

It is critical that quantum-based MD simulations conserve total energy when simulating an NVE ensemble. In practice, most of such simulations experience some drift in the total energy over time in part because of the finite convergence of the SCF procedure. To verify that PYSEQM is capable of conserving energy to within normal tolerances, we performed a 0.6 ps NVE MD simulation on the nanostar system with a variety of algorithms and both 0.1 and 0.2 fs time steps. The total energy of these simulations is shown in Figure 2. Table 1

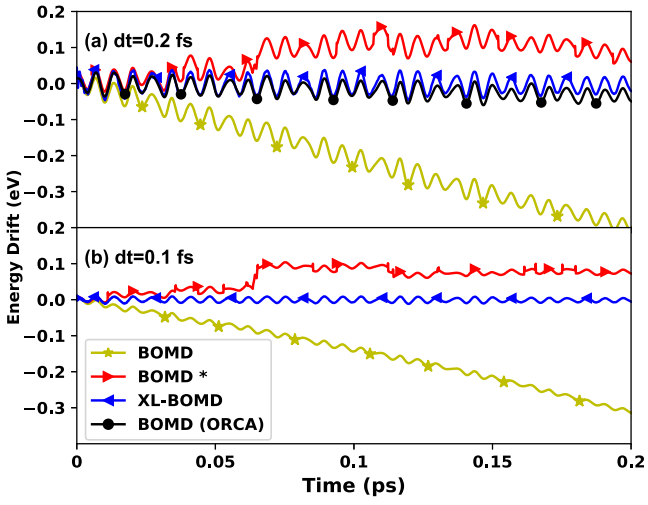


Figure 2. Total energy drift of the nanostar system during 0.6 ps simulations under various simulation conditions, 0.2 ps is shown here. The time steps (a) dt = 0.2 fs and (b) 0.1 fs are used with BOMD (yellow star), BOMD without reusing density matrix in consecutive MD steps (read right triangle), XL-BOMD (blue left triangle), and BOMD on ORCA (black dot).

shows the drifting coefficient, which is an average per atom unit of energy change per unit time, and energy fluctuation, which is the standard deviation of total energy after removing the drift. When restarting the SCF procedure with the density matrix obtained from the previous MD step, a large drift in total energy is observed as seen in the yellow trace of Figure 2. This can be improved in BOMD simulations with

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

Table 1. Energy Drift and Fluctuation from Figure 2 for BOMD and XL-BOMD Done With Package PYSEQM and ORCA[a]

	BOMD	BOMD	XL-BOMD	XL-BOMD	BOMD(ORCA)
timestep (fs)	0.2	0.1	0.2	0.1	0.2
drift (meV/ps/atom)	−2.13	−2.03	-1.21×10^{-2}	-2.94×10^{-3}	-1.95×10^{-1}
fluctuation (eV)	1.63×10^{-2}	1.17×10^{-2}	1.77×10^{-2}	4.42×10^{-3}	1.49×10^{-2}

[a]BOMD results here are for the cases with reusing the density matrix in consecutive MD steps.

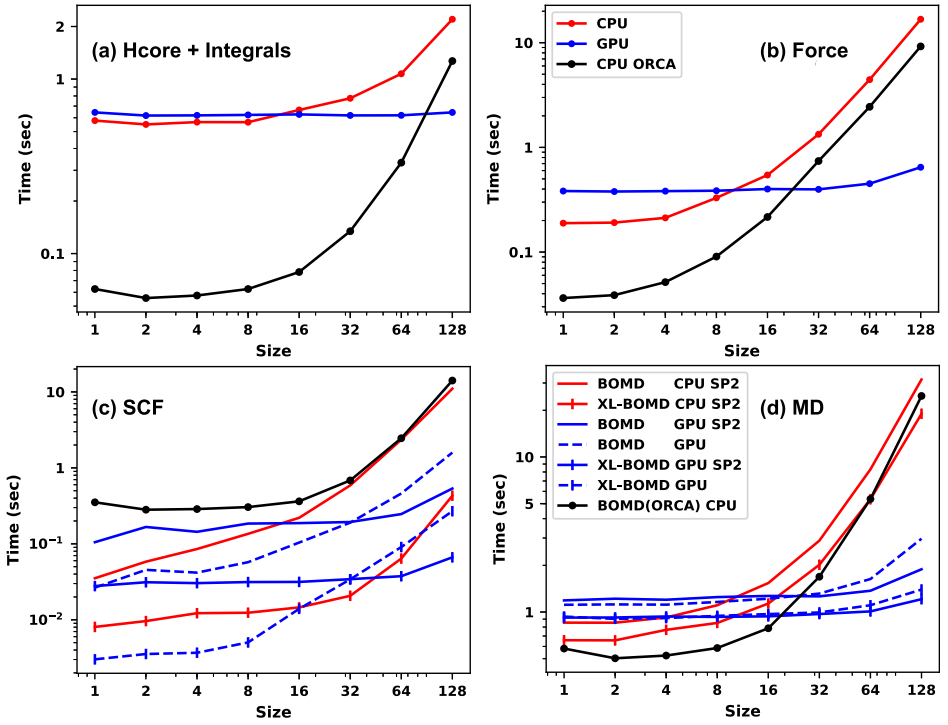


Figure 3. Average time spending per MD step on computing (a) Hcore and Coulombic Integrals, (b) Force, (c) SCF procedure, and (d) MD for eight molecules H-$(C_2H_4)_n$-H with size n = 1, 2,..., 128. The timing is done on the GPU (blue), CPU (red), CPU with ORCA (black dot) with BOMD (no marker), and XL-BOMD (vertical marker I) with SP2 (solid line) or conventional diagonalization (dash line). Only the results with SP2 on the CPU are shown here for clearness of figures as the difference with or without using SP2 on the CPU is small.

sophisticated SCF algorithms such as SOSCF and[57] direct inversion in the iterative subspace (DIIS),[58] which lead to much better energy conservation, for example, in ORCA. In PYSEQM, the drift can also be fixed through the use of the XL-BOMD algorithm, which for this particular system, has a drift of 2.94×10^{-3} meV/ps/atom with dt = 0.1 fs. This amount of drift is comparable to a classical MD simulation performed with LAMMPS using the AIREBO (not shown),[55,56] which has a drift around 1.0×10^{-3} meV/ps/atom with dt = 0.1 fs. The energy drift in PYSEQM BOMD can also be reduced by restarting the SCF procedure from a new guess at each time step (Figure 2, red right triangle trace) rather than reusing the previous time step's converged density matrix, although this is not used in practice for two reasons. First, starting from a guess density at each time step greatly increases the number of SCF iterations required at each point in the MD trajectory. Second, the density matrix may become stuck in local minima during the SCF procedure, resulting in a small energy jump, as shown in Figure 2 (red right triangle trace). The flat plateau of the curves in Figure 2 (red right triangle trace) shows that energy conservation in BOMD simulations can be improved by not reusing the density matrix when compared to the results in Figure 2 (yellow star trace). In practice, a generally acceptable drift is around 1 meV/ps/atom over long time scales, and the discussed strategy is not adapted for efficiency.

Another issue with respect to the energy conservation is the fluctuation of energy. This is coming from the discretization of propagation. Under the velocity Verlet scheme, the fluctuation is in the order of $O(\Delta t^2)$.[59,60] Because of this, the fluctuation can be suppressed by reducing the time step used for propagation. However, in practice for efficiency, the time step is frequently chosen to be 0.05−0.01 times the time scale for the highest characteristic vibrational frequency of the system. Generally, this guideline results in a time step of 0.1−1 fs for organic systems. Here, with reducing time step from 0.2 to 0.1 fs, the energy fluctuation is reduced by around 50% with BOMD and more than 75% with XL-BOMD.

3.2. Polyethylene. As the simplest polymer, polyethylene is a good system with which to check the scaling of PYSEQM with molecular size. We create 8 polyethylene chains H-$(C_2H_4)_n$-H with n = 1, 2, 4,..., 128, with the largest having a total of 770 atoms (256 C and 512 H). Similar to the nanostar, these molecules are first relaxed and equilibrated at 300 K with LAMMPS using the AIREBO force field for 50 ps. Then, we perform simulations for 20 steps with a time step of 0.2 fs, with all combinations of BOMD/XL-BOMD, SP2/conventional diagonalization, and CPU (Intel Xeon E5)/GPU (Nvidia TITAN_V) for these 8 molecules. This gives a good benchmark to quantify the scaling of PYSEQM with system size.

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

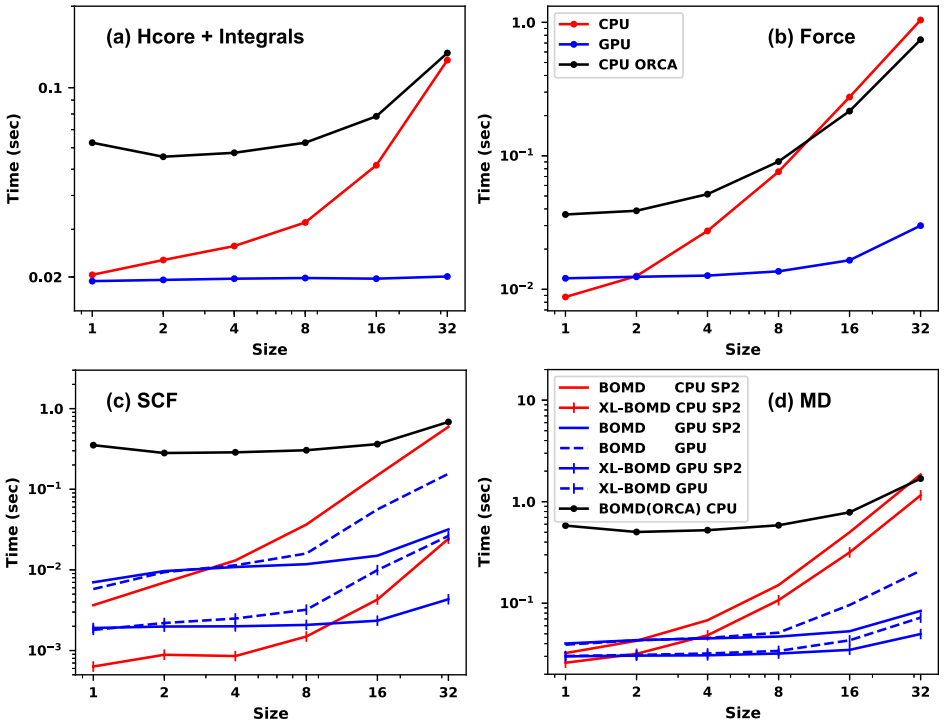


Figure 4. Average time needed per MD step per molecule with the batch mode enabled when computing (a) Hcore and Coulombic integrals, (b) force, (c) SCF procedure, and (d) MD for each of six H-$(C_2H_4)_n$-H with size n = 1, 2,..., 32. The same labeling and coloring are used here as in Figure 3.

Figure 3 shows the scaling of the Hcore and Coulombic integrals, Force, SCF, and MD computations. When running on a CPU, ORCA is faster for small molecules but it scales worse than PYSEQM, as shown in Figure 3a−d, with ORCA's performance becoming equal to that of PYSEQM for molecules $n \geq 32$, as shown in Figure 3d. It is reasonable that PYSEQM is slower for smaller systems because the overhead for function calls dominate the time cost in PYSEQM for small systems. This overhead in python is the main reason why it is slower than other low-level programming languages such as C/C++ used by ORCA. However, for larger systems, where overhead takes a smaller fraction of the total time of simulation, PYSEQM outperforms ORCA because it uses a vectorization strategy for computing Hcore, Integrals, and Force, as shown in Figure 3a,b. Figure 3a,b shows results from using PYSEQM on the CPU and GPU and ORCA on the CPU. For small molecules, ORCA performs worse than PYSEQM in the SCF procedure, which is because ORCA uses a systematic combination of SCF algorithms: in general, it is more stable and robust but slower. The difference with using SP2 or conventional diagonalization on the CPU is negligible and the results are not shown in Figure 3. Combined with the results on the nanostar molecule, PYSEQM is faster than ORCA for larger molecules when using a CPU.

However, the true strength of PYSEQM is its ability to run on a GPU. This results in a significant gain in performance when running on large systems. For small systems, the GPU mode is less efficient. Modern GPUs have thousands of slow cores, which only perform well with large amounts of data being processed by the same operations. As a result, for small systems, the GPU hardware is not fully utilized. In turn, the GPU mode of PYSEQM shows poor scaling for small systems, with the total time required to perform a calculation nearly independent of system size for small molecules, as shown in Figure 3. However, the ability of PYSEQM to run multiple simulations in a batch mode (discussed below) compensates for this drawback. Once the system size is large enough to use a significant fraction of the GPU, PYSEQM on a GPU can easily outperform PYSEQM and ORCA on the CPU. There is a similar trend for using the SP2 algorithm on a GPU. For smaller systems, SP2 is slower than conventional diagonalization, but for large systems, the matrix multiplications required by SP2 can fully utilize GPU cores and outperform conventional diagonalization algorithms.

Because of our desire to interface PYSEQM with ML algorithms, it has the ability to run in the "batched mode," where simulations are performed simultaneously on an ensemble of molecules to fully take advantage of the parallel GPU architecture. In addition to being useful for ML applications, many simulations in physics and chemistry require ensemble averaging such as computing the decay time of photoexcited electrons,[61] or estimating quantum yields,[62] making this "batched mode" directly relevant for applications as well. Thus, in Figure 4, we benchmark PYSEQM in the batched mode. We run the MD simulations with 32 molecules where trajectories were initiated with different initial velocities and starting from the same nuclear coordinates for each of the 8 polyethylene systems. The simulations here are done with the same setting as for the single-molecule mode.

Figure 4 shows the time cost per MD step per molecule when running with batches while Figure 5 shows the relative performance boost of the batch mode versus running the simulations in serial. Even running MD with 32 molecules at the same time, the occupancy on the GPU is still relatively low for small systems and the scaling is poor, as shown in Figure 4. However, the time cost per molecule is greatly reduced not only with the GPU but also when running on the CPU. For

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

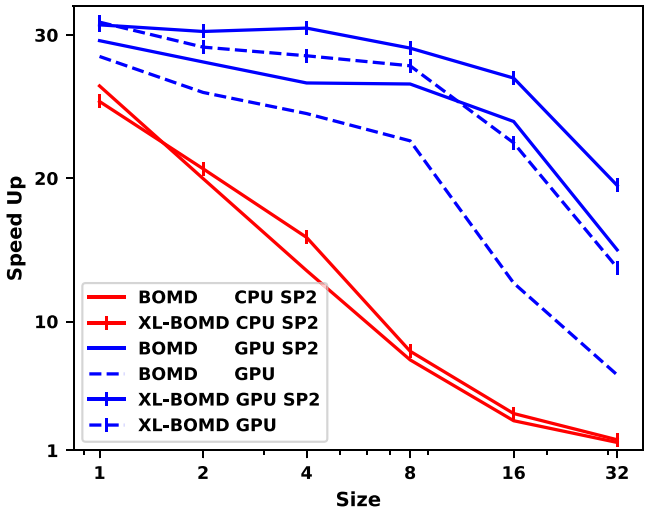


Figure 5. Performance boost when running 32 molecules simultaneously in the batch mode compared with running in serial. Simulations are performed on polyethylene H-$(C_2H_4)_n$-H, n = 1, 2,..., 32. On the CPU (red), the difference is small, and the results with conventional diagonalization are omitted. When computing on the GPU (blue), speedups of a factor of 30 are possible for small molecules.

small systems, the performance of PYSEQM in the batched mode on the CPU is 10−20 times better than that in the serial mode. As system size grows, the performance boost is quickly degraded on the CPU because PyTorch switches to perform operations in series for large operations such as matrix diagonalizations or multiplications. This can be seen in the scaling of computing forces and the SCF procedure in Figure 4b,c. The differences with SP2 and conventional diagonalization are relatively small and thus are not shown in Figures 4 and 5.

On the GPU, the performance boost with the batch mode is higher and degrades much slower, as shown in Figure 5. It is around 20−30 times faster than conventional serial simulations on the CPU. Additionally, the poor scaling for small systems indicates that further performance gains can be achieved by increasing the batch size to further saturate the GPU. SP2 shows its advantages for $n \geq 8$ for our polyethylene systems compared to conventional diagonalization. In our testing with random symmetric matrices, SP2 is faster than direct diagonalization almost for all sizes of the matrix. However, as PYSEQM is implemented to be able to deal with ensembles of systems simultaneously, preprocessing is required to use SP2, in which the overhead slows down SP2 for small systems. For larger systems, SP2 is around 5 times faster when compared with using direct diagonalization, as shown in Figure 4c, resulting in 2−3 times faster MD performance, as shown in Figure 4d. Overall, from the results shown in Figure 4a−c, PYSEQM on the GPU with the batch mode is systematically faster than conventional ORCA calculations by about 20 times.

4. CONCLUSIONS

In this work, we present the PYSEQM software package, which provides Born Oppenheimer molecular dynamics with semiempirical quantum mechanics methods (MNDO, AM1, PM3, and PM6) in the PyTorch framework, enabling GPU acceleration and a native interface with ML codes. To ensure rapid ground-state SCF computations at each MD steps, PYSEQM implements the SP2 density matrix expansion algorithm. To avoid energy drift present in conventional BOMD based on the quantum methods, PYSEQM further implements the XL-BOMD method, which both accelerates the calculations and enforces energy conservation because an iterative SCF optimization that typically is only approximate is avoided prior to the force evaluations. In particular, this allows achieving the energy conservation 100−1000 times better than traditional semiempirical quantum mechanical molecular dynamics and at a significant reduction in the computational cost. Finally, to make full use of the highly parallel GPU architecture, PYSEQM supports a "batched mode" where many molecules can be computed simultaneously. This batch feature, where many independent MD trajectories are run simultaneously, is generally absent in many packages and it is very beneficial for simulations requiring statistical averaging. Overall, the current PYSEQM implementation can treat systems with more than 1000 atoms and with elements from H to Cl, which cover all organic molecules.

We further use the nanostar dendrimer and several polyethylene chains to document performance of the PYSEQM code. Various PYSEQM applications to these molecular systems are compared to the ORCA computational package featuring conventional implementation to show for the end user PYSEQM's performance advantages and disadvantages. When running PYSEQM on the CPU, its performance is on par with a conventional implementation for smaller systems and is faster for large molecules owing the use of SP2 and XL-BOMD algorithms. Subsequently, here, the PYSEQM package provides a valuable alternative to other conventional codes on the CPU capable of semiempirical BOMD simulations. In contrast, the main focus of the PYSEQM design is its ability to efficiently utilize the GPU. We show that while it is slower to use PYSEQM for small systems on the GPU, this can be overcome utilizing the "batch mode". For large molecular systems (over 500 atoms), PYSEQM becomes very efficient on the GPU in the serial mode as well. Altogether, we demonstrate a factor of 50 speedup when performing BOMD calculations on the GPU when compared to conventional CPU calculations.

In summary, in the future, we envision multiple applications for and further development of the PYSEQM package. First, this software may be useful for an end user to perform effective BOMD simulations on the GPU using tried-and-true old semiempirical models (AM1, MNDO, PM3, and PM6), particularly, for large systems. Second, our future work will include extensions to other semiempirical methods (overviewed in the Introduction), which will both increase the accuracy and facilitate the treatment of more systems.[8] One specific target is extension of the implementation to nonadiabatic excited-state molecular dynamics[63−65] which will enable rapid ensemble propagation using the "batched mode" for calculations such as fewest switches surface hopping[63] (FSSH) or Multiconfigurational Ehrenfest with Ab Initio Multiple Cloning[66] (MCE-AIMC). Finally and most importantly, utilization of the PyTorch framework facilitates the construction of interfaces to ML methods, such as HIPNN,[29] ANI,[26,67] or SchNet.[68−70] By training such a neural network to generate custom parameter sets for the semiempirical Hamiltonians, the accuracy of these methods can be greatly increased. This will allow on-demand generation of reduced quantum-mechanical models for targeted molecular families able to accurately describe desired properties and dynamics of

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

both ground and excited states. Of particular interest for this method are Hamiltonians utilizing d oribitals, where we think significant improvements in accuracy can be obtained.[8,9,14] Another desirable future development is implementation of open-shell capability to the PYSEQM package to treat, for example, spin states and chemical reactions. Here, the SP2 scheme can easily be adapted for spin-polarized systems using a simple generalization starting with a block-diagonal Hamiltonian matrix with one block for the spin up and one for the spin down channel generating the corresponding idempotent block-diagonal single-particle density matrix.[41] All these extensions will lead to a wide application of the implementation presented here and be beneficial to researchers working in theoretical computation and ML in chemistry, physics, and material science. The described PYSEQM package is now released as open-source software and we invite the community to participate in its further development.[31]

AUTHOR INFORMATION

Corresponding Author

Ben Nebgen − *Los Alamos National Laboratory, Los Alamos, New Mexico 87545, United States;* orcid.org/0000-0001-5310-3263; Email: bnebgen@lanl.gov

Authors

Guoqing Zhou − *Department of Physics and Astronomy, University of Southern California, Los Angeles, California 90089, United States;* orcid.org/0000-0002-4000-8467

Nicholas Lubbers − *Los Alamos National Laboratory, Los Alamos, New Mexico 87545, United States*

Walter Malone − *Los Alamos National Laboratory, Los Alamos, New Mexico 87545, United States;* orcid.org/0000-0001-8245-322X

Anders M. N. Niklasson − *Los Alamos National Laboratory, Los Alamos, New Mexico 87545, United States;* orcid.org/0000-0003-1856-4982

Sergei Tretiak − *Los Alamos National Laboratory, Los Alamos, New Mexico 87545, United States;* orcid.org/0000-0001-5547-3647

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jctc.0c00243

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

We acknowledge support from the Los Alamos National Laboratory (LANL) Directed Research and Development funds (LDRD). This work was performed in part at the Center for Nonlinear Studies (CNLS) and the Center for Integrated Nanotechnology (CINT), a U.S. Department of Energy and Office of Basic Energy Sciences user facility. This research used resources provided by the LANL Institutional Computing Program. A.M.N.N. was supported by the U.S. Department of Energy Office of Basic Energy Sciences through FWP LANLE8AN.

REFERENCES

(1) Thiel, W. Semiempirical quantum−chemical methods. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2014**, *4*, 145−157.

(2) Bikadi, Z.; Hazai, E. Application of the PM6 semi-empirical method to modeling proteins enhances docking accuracy of AutoDock. *J. Cheminformatics* **2009**, *1*, 15.

(3) Kohn, W.; Sham, L. J. Self-consistent equations including exchange and correlation effects. *Phys. Rev.* **1965**, *140*, A1133.

(4) Engel, E.; Dreizler, R. M. *Density Functional Theory*; Springer: Berlin, 2013.

(5) Dewar, M. J. S.; Thiel, W. Ground states of molecules. 38. The MNDO method. Approximations and parameters. *J. Am. Chem. Soc.* **1977**, *99*, 4899−4907.

(6) Dewar, M. J. S.; Zoebisch, E. G.; Healy, E. F.; Stewart, J. J. P. Development and use of quantum mechanical molecular models. 76. AM1: a new general purpose quantum mechanical molecular model. *J. Am. Chem. Soc.* **1985**, *107*, 3902−3909.

(7) Stewart, J. J. P. Optimization of parameters for semiempirical methods II. Applications. *J. Comput. Chem.* **1989**, *10*, 221−264.

(8) Stewart, J. J. P. Optimization of parameters for semiempirical methods V: modification of NDDO approximations and application to 70 elements. *J. Mol. Model.* **2007**, *13*, 1173−1213.

(9) Dral, P. O.; Wu, X.; Thiel, W. Semiempirical quantum-chemical methods with Orthogonalization and dispersion corrections. *J. Chem. Theory Comput.* **2019**, *15*, 1743−1760.

(10) McNamara, J. P.; Sharma, R.; Vincent, M. A.; Hillier, I. H.; Morgado, C. A. The non-covalent functionalisation of carbon nanotubes studied by density functional and semi-empirical molecular orbital methods including dispersion corrections. *Phys. Chem. Chem. Phys.* **2008**, *10*, 128−135.

(11) Sure, R.; Grimme, S. Comprehensive benchmark of association (free) energies of realistic host−guest complexes. *J. Chem. Theory Comput.* **2015**, *11*, 3785−3801.

(12) Dobeš, P.; Rezác, J.; Fanfrlík, J.; Otyepka, M.; Hobza, P. Semiempirical quantum mechanical method PM6-DH2X describes the geometry and energetics of CK2-inhibitor complexes involving halogen bonds well, while the empirical potential fails. *J. Phys. Chem. B* **2011**, *115*, 8581−8589.

(13) Thiel, W.; Voityuk, A. A. Extension of MNDO to d orbitals: parameters and results for silicon. *J. Mol. Struct.: THEOCHEM* **1994**, *313*, 141−154.

(14) Stewart, J. J. P. Optimization of parameters for semiempirical methods VI: more modifications to the NDDO approximations and re-optimization of parameters. *J. Mol. Model.* **2013**, *19*, 1−32.

(15) Řezáč, J.; Fanfrlík, J.; Salahub, D.; Hobza, P. Semiempirical quantum chemical PM6 method augmented by dispersion and H-bonding correction terms reliably describes various types of noncovalent complexes. *J. Chem. Theory Comput.* **2009**, *5*, 1749−1760.

(16) Korth, M.; Pitoňák, M.; Řezáč, J.; Hobza, P. A transferable H-bonding correction for semiempirical quantum-chemical methods. *J. Chem. Theory Comput.* **2010**, *6*, 344−352.

(17) Korth, M. Third-generation hydrogen-bonding corrections for semiempirical QM methods and force fields. *J. Chem. Theory Comput.* **2010**, *6*, 3808−3816.

(18) Dral, P. O.; Wu, X.; Spörkel, L.; Koslowski, A.; Weber, W.; Steiger, R.; Scholten, M.; Thiel, W. Semiempirical quantum-chemical orthogonalization-corrected methods: theory, implementation, and parameters. *J. Chem. Theory Comput.* **2016**, *12*, 1082−1096.

(19) Grimme, S.; Steinmetz, M. Effects of London dispersion correction in density functional theory on the structures of organic molecules in the gas phase. *Phys. Chem. Chem. Phys.* **2013**, *15*, 16031−16042.

(20) Li, H.; Collins, C.; Tanha, M.; Gordon, G. J.; Yaron, D. J. A density functional tight binding layer for deep learning of chemical Hamiltonians. *J. Chem. Theory Comput.* **2018**, *14*, 5764−5776.

(21) Niklasson, A. M. Expansion algorithm for the density matrix. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2002**, *66*, 155115.

(22) Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic Differentiation in Pytorch. *NIPS 2017 Workshop Autodif*, 2017.

(23) Nebgen, B.; Lubbers, N.; Smith, J. S.; Sifain, A. E.; Lokhov, A.; Isayev, O.; Roitberg, A. E.; Barros, K.; Tretiak, S. Transferable dynamic molecular charge assignment using deep neural networks. *J. Chem. Theory Comput.* **2018**, *14*, 4687−4698.

(24) Sifain, A. E.; Lubbers, N.; Nebgen, B. T.; Smith, J. S.; Lokhov, A. Y.; Isayev, O.; Roitberg, A. E.; Barros, K.; Tretiak, S. Discovering a transferable charge assignment model using machine learning. *J. Phys. Chem. Lett.* **2018**, *9*, 4495−4501.
(25) Smith, J. S.; Nebgen, B. T.; Zubatyuk, R.; Lubbers, N.; Devereux, C.; Barros, K.; Tretiak, S.; Isayev, O.; Roitberg, A. E. Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning. *Nat. Chem.* **2019**, *10*, 2903.
(26) Smith, J. S.; Isayev, O.; Roitberg, A. E. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **2017**, *8*, 3192−3203.
(27) Ahneman, D. T.; Estrada, J. G.; Lin, S.; Dreher, S. D.; Doyle, A. G. Predicting reaction performance in C−N cross-coupling using machine learning. *Science* **2018**, *360*, 186−190.
(28) Raccuglia, P.; Elbert, K. C.; Adler, P. D. F.; Falk, C.; Wenny, M. B.; Mollo, A.; Zeller, M.; Friedler, S. A.; Schrier, J.; Norquist, A. J. Machine-learning-assisted materials discovery using failed experiments. *Nature* **2016**, *533*, 73.
(29) Lubbers, N.; Smith, J. S.; Barros, K. Hierarchical modeling of molecular energies using a deep neural network. *J. Chem. Phys.* **2018**, *148*, 241715.
(30) Zubatyuk, T.; Nebgen, B.; Lubbers, N.; Smith, J. S.; Zubatyuk, R.; Zhou, G.; Koh, C.; Barros, K.; Isayev, O.; Tretiak, S. Machine Learned Hückel Theory: Interfacing Physics and Deep Neural Networks, **2019**. arXiv:1909.129633.
(31) Zhou, G. PYSEQM. https://github.com/lanl/PYSEQM (accessed June 2020).
(32) Tapavicza, E.; Bellchambers, G. D.; Vincent, J. C.; Furche, F. Ab initio non-adiabatic molecular dynamics. *Phys. Chem. Chem. Phys.* **2013**, *15*, 18336.
(33) Niklasson, A. M. Extended born-oppenheimer molecular dynamics. *Phys. Rev. Lett.* **2008**, *100*, 123004.
(34) Niklasson, A. M. N.; Steneteg, P.; Odell, A.; Bock, N.; Challacombe, M.; Tymczak, C. J.; Holmström, E.; Zheng, G.; Weber, V. Extended Lagrangian Born−Oppenheimer molecular dynamics with dissipation. *J. Chem. Phys.* **2009**, *130*, 214109.
(35) Souvatzis, P.; Niklasson, A. M. N. First principles molecular dynamics without self-consistent field optimization. *J. Chem. Phys.* **2014**, *140*, 044117.
(36) Niklasson, A. M. N.; Cawkwell, M. J. Generalized extended lagrangian born-oppenheimer molecular dynamics. *J. Chem. Phys.* **2014**, *141*, 164123.
(37) Niklasson, A. M. N. Next generation extended Lagrangian first principles molecular dynamics. *J. Chem. Phys.* **2017**, *147*, 054103.
(38) Mniszewski, S. M.; Cawkwell, M. J.; Wall, M. E.; Mohd-Yusof, J.; Bock, N.; Germann, T. C.; Niklasson, A. M. N. Efficient parallel linear scaling construction of the density matrix for Born−Oppenheimer molecular dynamics. *J. Chem. Theory Comput.* **2015**, *11*, 4644−4654.
(39) Cawkwell, M. J.; Wood, M. A.; Niklasson, A. M. N.; Mniszewski, S. M. Computation of the density matrix in electronic structure theory in parallel on multiple graphics processing units. *J. Chem. Theory Comput.* **2014**, *10*, 5391−5396.
(40) Rubensson, E. H.; Niklasson, A. M. N. Interior eigenvalues from density matrix expansions in quantum mechanical molecular dynamics. *SIAM J. Sci. Comput.* **2014**, *36*, B147−B170.
(41) Mniszewski, S. M.; Perriot, R.; Rubensson, E. H.; Negre, C. F. A.; Cawkwell, M. J.; Niklasson, A. M. N. Linear Scaling Pseudo Fermi-Operator Expansion for Fractional Occupation. *J. Chem. Theory Comput.* **2018**, *15*, 190−200.
(42) Palma, J. L.; Atas, E.; Hardison, L.; Marder, T. B.; Collings, J. C.; Beeby, A.; Melinger, J. S.; Krause, J. L.; Kleiman, V. D.; Roitberg, A. E. Electronic spectra of the nanostar dendrimer: Theory and experiment. *J. Phys. Chem. C* **2010**, *114*, 20702−20712.
(43) Roothaan, C. C. J. New developments in molecular orbital theory. *Rev. Mod. Phys.* **1951**, *23*, 69.
(44) Dewar, M. J. S.; Thiel, W. A semiempirical model for the two-center repulsion integrals in the NDDO approximation. *Theor. Chim. Acta* **1977**, *46*, 89−104.
(45) Badziag, P.; Solms, F. An improved SCF iteration scheme. *Comput. Chem.* **1988**, *12*, 233−236.
(46) Pulay, P. Convergence acceleration of iterative sequences. The case of SCF iteration. *Chem. Phys. Lett.* **1980**, *73*, 393−398.
(47) Camp, R. N.; King, H. F. An interpolation method for forcing SCF convergence. *J. Chem. Phys.* **1981**, *75*, 268−274.
(48) Chaban, G.; Schmidt, M. W.; Gordon, M. S. Approximate second order method for orbital optimization of SCF and MCSCF wavefunctions. *Theor. Chem. Acc.* **1997**, *97*, 88−95.
(49) Daniels, A. D.; Scuseria, G. E. What is the best alternative to diagonalization of the Hamiltonian in large scale semiempirical calculations? *J. Chem. Phys.* **1999**, *110*, 1321−1328.
(50) Niklasson, A. M.; Tymczak, C.; Challacombe, M. Time-reversible Born-Oppenheimer molecular dynamics. *Phys. Rev. Lett.* **2006**, *97*, 123001.
(51) Verlet, L. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Phys. Rev.* **1967**, *159*, 98.
(52) Neese, F. The ORCA program system. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2012**, *2*, 73−78.
(53) Fernandez-Alberti, S.; Kleiman, V. D.; Tretiak, S.; Roitberg, A. E. Nonadiabatic molecular dynamics simulations of the energy transfer between building blocks in a phenylene ethynylene dendrimer. *J. Phys. Chem. A* **2009**, *113*, 7535−7542.
(54) Bradshaw, D. S.; Andrews, D. L. Mechanisms of light energy harvesting in dendrimers and hyperbranched polymers. *Polymers* **2011**, *3*, 2053−2077.
(55) Stuart, S. J.; Tutein, A. B.; Harrison, J. A. A reactive potential for hydrocarbons with intermolecular interactions. *J. Chem. Phys.* **2000**, *112*, 6472−6486.
(56) Plimpton, S. Fast parallel algorithms for short-range molecular dynamics. *J. Comput. Phys.* **1995**, *117*, 1−19.
(57) Neese, F. Approximate second-order SCF convergence for spin unrestricted wavefunctions. *Chem. Phys. Lett.* **2000**, *325*, 93−98.
(58) Hamilton, T. P.; Pulay, P. Direct inversion in the iterative subspace (DIIS) optimization of open-shell, excited-state, and small multiconfiguration SCF wave functions. *J. Chem. Phys.* **1986**, *84*, 5728−5734.
(59) Toxvaerd, S.; Heilmann, O. J.; Dyre, J. C. Energy conservation in molecular dynamics simulations of classical systems. *J. Chem. Phys.* **2012**, *136*, 224106.
(60) Lippert, R. A.; Bowers, K. J.; Dror, R. O.; Eastwood, M. P.; Gregersen, B. A.; Klepeis, J. L.; Kolossvary, I.; Shaw, D. E. A common, avoidable source of error in molecular dynamics integrators. *J. Chem. Phys.* **2007**, *126*, 046101.
(61) Zhou, G.; Cen, C.; Wang, S.; Deng, M.; Prezhdo, O. V. Electron−Phonon Scattering Is Much Weaker in Carbon Nanotubes than in Graphene Nanoribbons. *J. Phys. Chem. Lett.* **2019**, *10*, 7179.
(62) Callis, P. R.; Vivian, J. T. Understanding the variable fluorescence quantum yield of tryptophan in proteins using QM-MM simulations. Quenching by charge transfer to the peptide backbone. *Chem. Phys. Lett.* **2003**, *369*, 409−414.
(63) Tully, J. C. Molecular dynamics with electronic transitions. *J. Chem. Phys.* **1990**, *93*, 1061−1071.
(64) Parandekar, P. V.; Tully, J. C. Mixed quantum-classical equilibrium. *J. Chem. Phys.* **2005**, *122*, 094102.
(65) Tapavicza, E.; Bellchambers, G. D.; Vincent, J. C.; Furche, F. Ab initio non-adiabatic molecular dynamics. *Phys. Chem. Chem. Phys.* **2013**, *15*, 18336−18348.
(66) Shalashilin, D. V. Nonadiabatic dynamics with the help of multiconfigurational Ehrenfest method: Improved theory and fully quantum 24D simulation of pyrazine. *J. Chem. Phys.* **2010**, *132*, 244111.
(67) Gao, X.; Ramezanghorbani, F.; Isayev, O.; Smith, J. S.; Roitberg, A. E. TorchANI: A Free and Open Source PyTorch Based

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962

Deep Learning Implementation of the ANI Neural Network Potentials. *J. Chem. Inf. Model.* **2020**, *60* (7), 3408−3415.
(68) Schütt, K. T.; Sauceda, H. E.; Kindermans, P.-J.; Tkatchenko, A.; Müller, K.-R. SchNet−A deep learning architecture for molecules and materials. *J. Chem. Phys.* **2018**, *148*, 241722.
(69) Schütt, K.; Kindermans, P.-J.; Felix, H. E. S.; Chmiela, S.; Tkatchenko, A.; Müller, K.-R. In Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in Neural Information Processing Systems*; Curran Associates, Inc., 2017; pp 991−1001.
(70) Schütt, K. T.; Kessel, P.; Gastegger, M.; Nicoli, K. A.; Tkatchenko, A.; Müller, K.-R. SchNetPack: A deep learning toolbox for atomistic systems. *J. Chem. Theory Comput.* **2019**, *15*, 448−455.

https://dx.doi.org/10.1021/acs.jctc.0c00243
J. Chem. Theory Comput. 2020, 16, 4951−4962