JCTC Journal of Chemical Theory and Computation

pubs.acs.org/JCTC Article

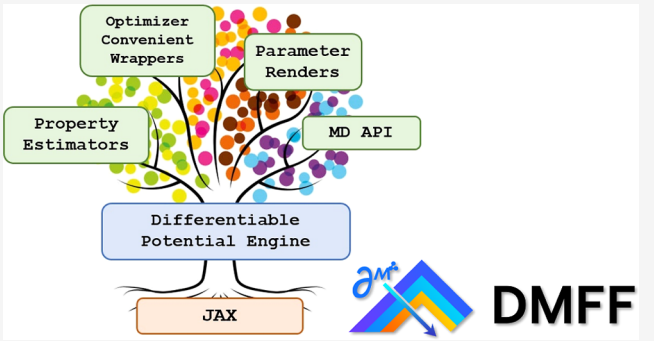
DMFF: An Open-Source Automatic Differentiable Platform for Molecular Force Field Development and Molecular Dynamics Simulation

Xinyan Wang,[⊥] Jichen Li,[⊥] Lan Yang, Feiyang Chen, Yingze Wang, Junhan Chang, Junmin Chen, Wei Feng, Linfeng Zhang,* and Kuang Yu*

Cite This: *J. Chem. Theory Comput.* 2023, 19, 5897−5909 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: In the simulation of molecular systems, the underlying force field (FF) model plays an extremely important role in determining the reliability of the simulation. However, the quality of the state-of-the-art molecular force fields is still unsatisfactory in many cases, and the FF parameterization process largely relies on human experience, which is not scalable. To address this issue, we introduce DMFF, an open-source molecular FF development platform based on an automatic differentiation technique. DMFF serves as a powerful tool for both top-down and bottom-up FF development. Using DMFF, both energies/forces and thermodynamic quantities such as ensemble averages and free energies can be evaluated in a differentiable way, realizing an automatic, yet highly flexible FF optimization workflow. DMFF also eases the evaluation of forces and virial tensors for complicated advanced FFs, helping the fast validation of new models in molecular dynamics simulation. DMFF has been released as an open-source package under the LGPL-3.0 license and is available at https://github.com/deepmodeling/DMFF.



1. INTRODUCTION

The organic molecular system is an important subject of study in a variety of research fields including materials science and biology. As one of the most important computer simulation tools, molecular mechanics (MM) plays an essential role in the studies of organic materials and biomolecules.[1,2] All MM simulations rely on a potential energy surface (PES), which describes the energies and the forces of the simulated atoms. For the sake of computational efficiency, the PES is typically approximated using a classical model, namely molecular force field (FF), instead of being computed ab initio on-the-fly. Therefore, the quality of the underlying FF limits the accuracy and the predictive power of the MM simulation.

To date, conventional FFs such as GAFF,[3] UFF,[4] CHARMM,[5] and OPLS-AA[6] have been the main workhorses in materials science and biological simulations. All these force fields partition the total energy into inter- and intramolecular parts based on the bonding topology of the molecule. Intermolecular interactions are described by the atomic point charges and simple pairwise-additive van der Waals potentials. The intramolecular interactions are decomposed into a direct sum of contributions from all internal coordinates (i.e., bond lengths, angles, dihedrals, etc.). These force fields can be computed at a fast speed, making them primary models to use in industry to study large molecular systems. However, the mathematical form of the conventional force fields is heavily approximated, neglecting many important physics including polarization, charge penetration, many-body dispersion, and so forth. Therefore, conventional force fields are hard to generalize and are essentially effective models that need to be tailored for specific systems or applications. The force field parameters are often obtained through a top-down approach, fitted to reproduce the experimental macroscopic properties of interest. Such a parameterization process relies heavily on human intervention and thus cannot be done in high throughput. Consequently, force field parameterization is often the bottleneck of research when studying new systems with a large number of parameters. There are efforts trying to automate this process[7] and tools recently developed such as ForceBalance and OpenFF[8−10] to serve this purpose. However, the differentiation of macroscopic properties with respect to force field parameters is a nontrivial task. In these works, such differentiation is still

Received: December 22, 2022
Published: August 17, 2023

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

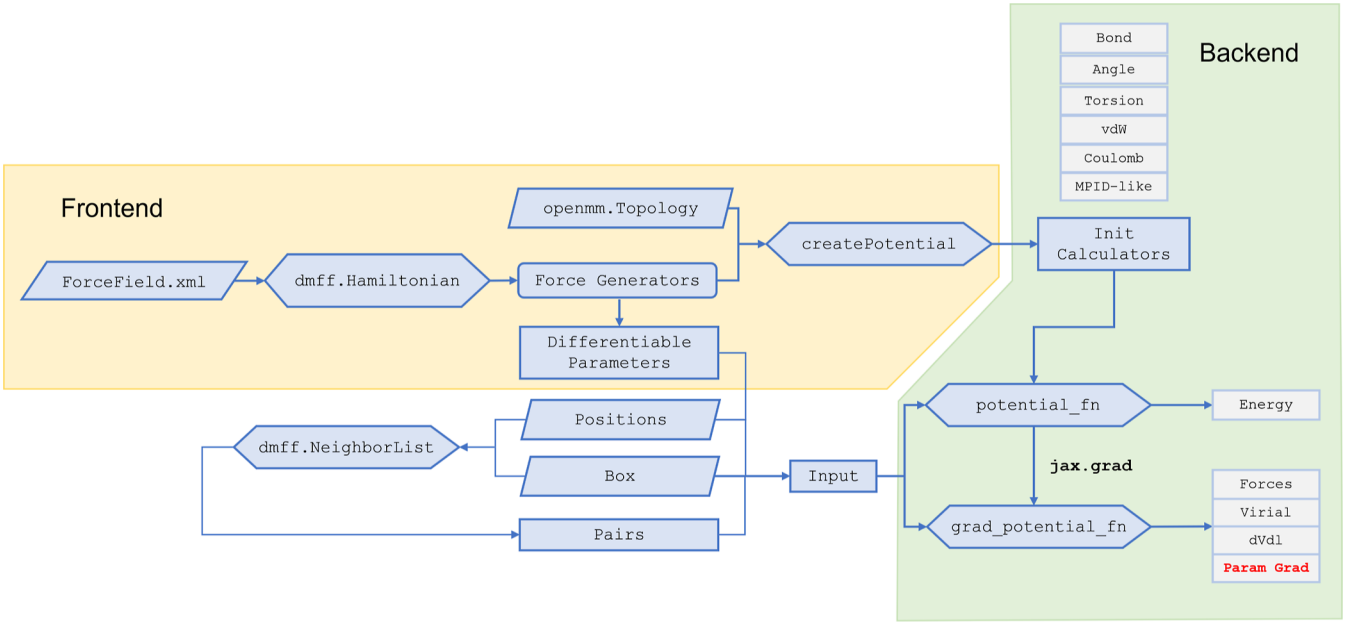


Figure 1. Overall code structure of the DMFF FF frontend and backend. The arrows in the figure represent the logical sequence of function calls and the directions of data flow.

conducted using the finite difference method, resulting in limited efficiency and poor scalability. Considering how widely conventional force fields are used in modern industry, their automatic optimization remains an important challenge to be solved.

Meanwhile, more advanced force fields are being developed using function forms beyond conventional force fields. These force fields are typically developed using the bottom-up strategy, in which ab initio energies and forces are the fitting targets. The new generation of force fields can be categorized into two types: the data-driven models and the physics-driven models. The data-driven models include a variety of machine learning (ML) models such as BPNN,[11] DeepPotential,[12] PhysNet,[13] and EANN.[14] These ML models are becoming increasingly popular recently and achieved great success in homogeneous and hard material systems. However, the performances of such pure ML methods are less reliable in bulk organic molecular systems. The current ML frameworks do not take full advantage of the bonding topology information, and a more accurate treatment for the long-range interactions is also in need. Therefore, for molecular systems, ML methods need to be combined with physics-driven models,[15,16] leading to hybrid strategies. For physics-driven models, new terms such as multipolar interactions, polarization effects, and charge transfer effects are introduced into the model analytically.[17−19] Due to the complexity of these interactions, the implementation of the corresponding forces and the virial tensors is an extremely tedious task, which impedes the fast validation of the model in MD. The situation becomes even worse for hybrid ML/physics-driven models, which are not well supported in all the currently available MD programs.

Here, we note that all the tasks mentioned above can be formulated as differentiating the model with respect to either FF parameters or system geometry. Such a task can be easily performed using the automatic differentiation technique. Being the basis of all modern ML platforms (e.g., tensorflow, torch, JAX, etc.), automatic differentiation allows the developers to focus on the computing process itself, while leaving the differentiation to computers. The differentiation of macroscopic properties with respect to FF parameters, combined with modern gradient descent algorithms, enables automatic optimization workflows for FF development. Meanwhile, the automatic differentiation capability naturally eases the implementation of forces and virial tensors, leading to fast MD validation. Based on different platforms, researchers have built several end-to-end differentiable MD engines including TorchMD,[20] JAX-MD,[21] and SPONGE.[22] However, as discussed in a previous study,[23] for almost all thermodynamic properties, end-to-end differentiation through the entire MD trajectory is not necessary. On the contrary, the deep computation graph of MD leads to extra issues such as gradient vanishing and exploding, as well as huge memory costs in backward propagation. On the other hand, a comprehensive support for more FF function forms and more types of object functions is a more urgent need. For most molecular FF developers, functions such as topological scaling (e.g., the 1−3 scaling), flexible atom typification, long-range Ewald, polarization, and so forth are critical but are still not supported in the aforementioned works.[20−23] Currently, the only differentiable framework dedicated to FF development is JAX-ReaxFF,[24] which is however specific to ReaxFF. So far, there is still no differentiable platform designed for molecular FF development, which is the object of this work.

In this work, we build an automatic differentiable molecular force field platform, namely, DMFF (Deep Modelling Force Field, or Differentiable Molecular Force Field). DMFF is a comprehensive implementation of both conventional molecular FFs and advanced multipolar polarizable models. It features a user-friendly OpenMM-like interface for convenient molecular FF definition and parameter file generation. It also possesses an extensible backend structure, such that new potential forms (e.g., various ML potentials and customized pairwise potentials) can be easily added. Based on the comprehensive implementation of the FF models, DMFF also provides convenient differentiable estimators for not only energies and forces but also thermal dynamic quantities including free energies and

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

ensemble-averaged properties. Based on these estimators, the corresponding object functions can be defined, enabling automatic bottom-up and top-down optimization workflows. DMFF is built on the basis of the JAX framework.[25] As a function-oriented framework, JAX allows the users to recombine and decorate the DMFF functions, allowing easy customization of new potentials and object functions. Such features are extremely important in force field development, saving the developers from re-implementing the same computation kernels (such as Ewald sum) repeatedly. The XLA-backed just-in-time (JIT) function in JAX provides GPU support and also makes DMFF run much more efficiently compared to typical Python programs. Relying on these features, DMFF is designed to be a both flexible and efficient tool for the development of molecular force fields.

2. PROGRAM STRUCTURE AND THEORETICAL BACKGROUND

The entire DMFF package is available on GitHub.[26] Here, we will introduce the overall structure of DMFF from four aspects: model preparation, model implementation, model optimization, and model deployment.

2.1. Model Preparation with DMFF Frontend. The core function of DMFF is a differentiable molecular force field calculator, the structure of which is represented in Figure 1. DMFF FF implementations can be divided into two parts: the frontend interfaces and the backend kernels. The DMFF frontend is in charge of model preparation and is currently designed to resemble the behavior of the OpenMM API.[27] By doing so, we can share the well-established open-source OpenMM ecosystem for the convenient definition of the molecular FF. All parameters can be defined in the format of OpenMM FF XML files and a core class named "Hamiltonian" is devised to overload the "ForceField" class in OpenMM. Within the Hamiltonian object, the FF parameters are read and stored in a highly organized and hierarchical data structure. A demo for the preparation of a DMFF FF function and parameters can be found in Listing 1.

Listing 1: A code demo for DMFF force field frontend interface.

```
import openmm as mm
import openmm.app as app
import openmm.unit as unit
from dmff.api import Hamiltonian
app.Topology.loadBondDefinitions("lig-top.xml")
pdb = app.PDBFile("lig.pdb")
H = Hamiltonian("gaff-2.11.xml", "lig-prm.xml")
# Generate the potential function
pots = H.createPotential(pdb.topology,
                         nonbondedMethod=app.PME)
potential_func = pots.getPotentialFunc()
# Access force field parameters
params = H.getParameters()
# Render new parameter file
H.render('forcefield.xml')
```

Within the DMFF frontend, for each potential term (i.e., each node in the xml file), a "generator" class is defined. The generator class is in charge of dispatching FF parameters based on the molecular topology and then invokes the backend kernels to return a differentiable potential function to users. The atom typification process is done in the frontend, so we can keep the maximal flexibility of the backend computational kernels, which can be further wrapped to obtain more advanced potentials if necessary.

2.2. Model Implementation with the DMFF Backend. In the backend, we define different forces, each of which corresponds to a potential term (i.e., harmonic bond, harmonic angles, electrostatics, etc.). Each force object encloses an environment for a potential calculation, which is defined by all the parameters that do not need to be differentiable (e.g., distance cutoff and the energy threshold in an Ewald calculation). The force object then returns a potential in the form of a Python function, which exposes differentiable variables (system geometries and FF parameters) as inputs. The function then can be further modified by the JIT or the automatic gradient decorators provided in JAX. The current version of the DMFF backend provides a complete implementation of conventional molecular FFs and a multipolar polarizable potential module (named Auto Differentiable Multipolar Polarizable module, or ADMP) that replicates the MPID[18] behavior up to quadrupole moments. For long-range interactions, particle−mesh Ewald (PME)[28] is supported for both multipolar electrostatics and dispersion interactions. As stated above, all backend functions do not recognize atom types and only accept parameters per atom as inputs. The DMFF backend forces are organized in a modular style, so they can be added, assembled, and extended at will to generate more complex potentials such as fluctuating charge models.

2.3. Model Optimization. The biggest advantage of the differentiable implementation of molecular FF is that now we can employ gradient descent algorithms to optimize FFs

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

automatically in high throughput. Due to the flexibility of JAX, we can define the object function as a combination of various properties, thus optimizing them simultaneously. DMFF aims to build a variety of differentiable property estimators, so they can be combined by the users to define highly customizable object functions. Then, the object functions can be optimized using the state-of-the-art optimizers implemented in external packages such as optax[29] or jaxopt.[30]

The most trivial estimators that come naturally with the DMFF FF calculators are energies and forces estimators. Both energies and forces are the most important fitting targets in the bottom-up force field development. DMFF offers a systematic tool to conduct energy and force fittings, which is advantageous in two ways: (1) it offers a clean API interfacing with the external optimization programs, so the most advanced optimizers developed in the ML community can be directly employed with minimal coding efforts; (2) it is now convenient to combine energies and forces with other targets, even realizing a mixed bottom-up and top-down strategy. Overall speaking, as we will show later, DMFF really makes it simple to optimize a large number of nonlinear parameters with respect to ab initio energy and force targets.

Meanwhile, in top-down force field development, developers aim to reproduce macroscopic properties including density, radial distribution function (RDF), evaporation enthalpy, free energy difference, and so forth. To compute these properties, one needs to perform long MD simulations, the differentiation of which is nontrivial to do. Previous studies often differentiate through the entire MD trajectory, which is extremely expensive in both computational time and memory cost. Meanwhile, following the spirit of several previous works,[31,32] Thaler and Zavadlav[23] showed that for ensemble averages, such end-to-end differentiation can be avoided by a trajectory reweighting scheme. In DMFF, we put the reweighting algorithm into a more general context of the MBAR[33] method, and introduce differentiable estimators for both averaged properties and free energies. While the differentiable evaluation of dynamic quantities remains a challenge, the reweighting MBAR estimator makes the fittings of thermodynamic properties relatively easy. We now introduce the basic formulation of the reweighting MBAR estimator in DMFF in brief.

In the MBAR theory, it is assumed that there are K ensembles defined by effective energies $u_i(x)$, $(i = 1, 2, ..., K)$. Note that $u(x)$ takes different forms in different ensembles. If $U(x)$ is the potential energy, then

$$u(x) = \begin{cases} U(x) \text{ for } NVT \text{ ensemble} \\ U(x) + pV(x) \text{ for } NPT \text{ ensemble} \\ U(x) - \mu N(x) \text{ for } \mu VT \text{ ensemble} \end{cases} \quad (1)$$

The Boltzmann weight, the partition function, and the probability in each ensemble are defined as

$$\begin{cases} w_i(x) = \exp(-\beta \cdot u_i(x)) \\ c_i = \int \mathrm{d}x \cdot w_i(x) \\ p_i(x) = \dfrac{w_i(x)}{c_i} \end{cases} \quad (2)$$

From each ensemble i, we draw N_i samples, labeled as $\{x_{in}\}$, $(n = 1, 2, ..., N_i)$, and the whole sample set can be labeled as $\{x_n\}$, $n = (1, 2, ..., N)$, with N being the total sample size: $N = \sum_{i=1}^{K} N_i$. MBAR theory states that the statistically optimal estimators $(\hat{c}_i)$ for the partition functions c_i of these K ensembles satisfy the following equation[33]

$$\hat{c}_i = \sum_{n=1}^{N} \frac{w_i(x_n)}{\sum_{k=1}^{K} N_k \hat{c}_k^{-1} w_k(x_n)} \quad (3)$$

Note that eq 3 only regulates the ratio between different $\hat{c}_i$, so the free energy difference between two different ensembles i and j, $\Delta f_{ij} = -\ln(c_j/c_i)$ can be solved, where c_j means the partition function of ensemble j.

Meanwhile, to compute a property $A(x)$ averaged in ensemble i, we can define a fictitious ensemble j with a fictitious Boltzmann weight and partition function

$$\begin{cases} w_j(x) = A(x) w_i(x) \\ c_j = \int \mathrm{d}x \cdot A(x) w_i(x) \end{cases} \quad (4)$$

Then, eq 3 can be used to solve the ratio between c_j and c_i, which is the ensemble average of A

$$\frac{\hat{c}_j}{\hat{c}_i} = \frac{\int \mathrm{d}x \cdot A(x) w_i(x)}{\int \mathrm{d}x \cdot w_i(x)} = \langle A \rangle_i \quad (5)$$

Only this time no samples are drawn from the fictitious ensemble j, so N_j is always 0. Therefore, MBAR gives a unified theory for the evaluation of free energies and ensemble-averaged properties from multiple samplings.

In the original MBAR theory, eq 3 needs to be solved iteratively as it is a set of coupled nonlinear equations for all $\hat{c}_i$. However, in the spirit of reweighting, the computation of the estimator can be largely simplified. Noticing that in a gradient descent training process, the parameters are only slightly perturbed in each training cycle, thus the samplings from the previous cycles can be reused. A resample is not necessary until the current ensemble (i.e., the ensemble being optimized) deviates from the sampling ensembles significantly. Therefore, in our implementation of reweighting MBAR, we define two types of ensembles: the sampling ensembles, from which all samples are drawn (assuming there are M of them, labeled by index $m = 1, 2, ..., M$), and the target ensembles (labeled by indices p, q), which are the ensembles subject to optimization. The sampling ensembles are only updated when necessary and they do not need to be differentiable, so their data can be generated using external packages such as OpenMM. Since all samples are drawn from the sampling ensembles, the N_p for the target ensembles is always zero. Consequently, in eq 3, the $\hat{c}_p$ is well decoupled from $\hat{c}_m$ and each other and can be computed from $\hat{c}_m$ in one shot

$$\hat{c_p} = \sum_{n=1}^{N} \frac{w_p(x_n)}{\sum_{m=1}^{M} N_m \hat{c}_m^{-1} w_m(x_n)} \quad (6)$$

In practice, each time when resample happens, $\hat{c}_m$ are updated by solving eq 3 iteratively and are stored as constants within the estimator until the next resampling. Then, during the parameter optimization, $\hat{c}_p$, or their ratio $\hat{c}_p/\hat{c}_q$, is computed using eq 6, and returned as a differentiable estimator.

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

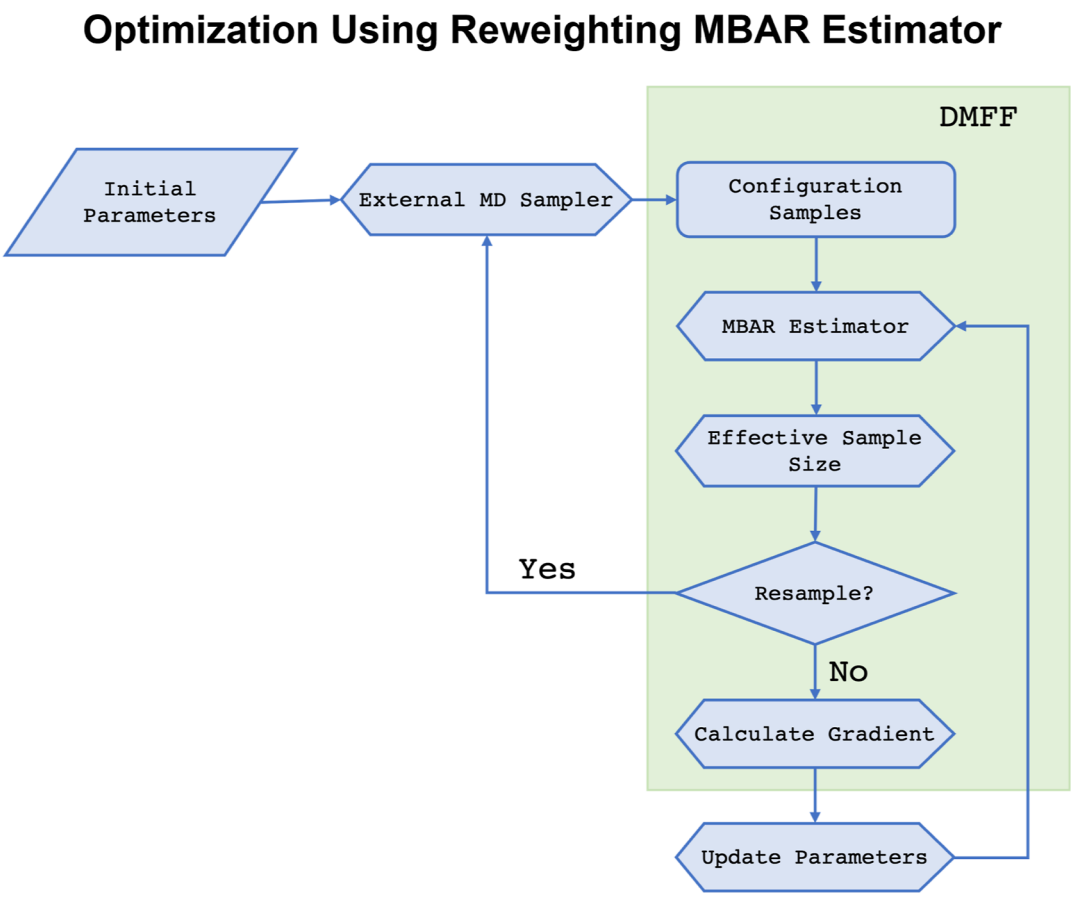


Figure 2. Workflow of thermodynamic properties optimization using the reweighting MBAR estimator.

If we set the sampling ensembles to be a single ensemble $w_0(x)$, and the target ensembles to be $w_p(x)$ and $w_q(x) = A(x)\,w_p(x)$, then it is straightforward to show that

$$\langle A\rangle_p = \frac{\hat{c}_q}{\hat{c}_p} = \frac{\sum_{n=1}^{N} A(x_n)\exp(-\beta\Delta u_{p0}(x_n))}{\sum_{n=1}^{N}\exp(-\beta\Delta u_{p0}(x_n))} \tag{7}$$

With $\Delta u_{p0}(x) = u_p(x) - u_0(x)$. This result recovers the eqs 3 and 4 in ref 23, showing that the trajectory reweighting method is indeed a special case of the reweighting MBAR estimator.

In practice, for convenience, DMFF provides two APIs for free energy and averaged property evaluations, respectively. For free energies, $\Delta\hat{f}_{pq} = -\ln\left(\frac{\hat{c}_p}{\hat{c}_q}\right)$ is directly computed and returned using eq 6. While for averaged properties, we reformulate eq 6 as

$$\begin{aligned}\langle A\rangle_p &= \sum_{n=1}^{N} \frac{\left[\sum_{m=1}^{M} N_m \exp(\hat{f}_m - \beta\Delta u_{mp}(x_n))\right]^{-1}}{\sum_{n'=1}^{N}\left[\sum_{m=1}^{M} N_m \exp(\hat{f}_m - \beta\Delta u_{mp}(x_{n'}))\right]^{-1}} A(x_n) \\ &= \sum_{n=1}^{N} W_n A(x_n)\end{aligned} \tag{8}$$

with

$$\begin{cases}\Delta u_{mp}(x_n) = u_m(x_n) - u_p(x_n) \\ W_n = \dfrac{\left[\sum_{m=1}^{M} N_m \exp(\hat{f}_m - \beta\Delta u_{mp}(x_n))\right]^{-1}}{\sum_{n'=1}^{N}\left[\sum_{m=1}^{M} N_m \exp(\hat{f}_m - \beta\Delta u_{mp}(x_{n'}))\right]^{-1}}\end{cases} \tag{9}$$

The estimator gives the MBAR weights of each sample (W_n), using which the user can construct their own property estimator easily. The estimator also evaluates Kish's effective sample size, which could be used to determine when a data set is redundant and thus can be removed, and when new samples are needed

$$n_{\text{eff}} = \frac{\left(\sum_{n=1}^{N} W_n\right)^2}{\sum_{n=1}^{N} W_n^2} \tag{10}$$

The gradient can be obtained by differentiating the estimator, and be fed to an external optimizer (e.g., optax[29] or jaxopt[30]) to update the FF parameters. A typical automatic workflow for a thermodynamic fitting using the reweighting MBAR estimator is given in Figure 2.

In summary, for model optimization, DMFF currently supports a variety of object functions including energies, forces, ensemble-averaged properties, and free energy differences. Other estimators such as electrostatic potentials (ESP) or dynamic properties (e.g., diffusion coefficients, electrical conductivities, spectroscopy, etc.) are also under development. Nevertheless, the current capability of DMFF already makes it a powerful tool for FF parameterization, as we will demonstrate in the Results and Discussion section.

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

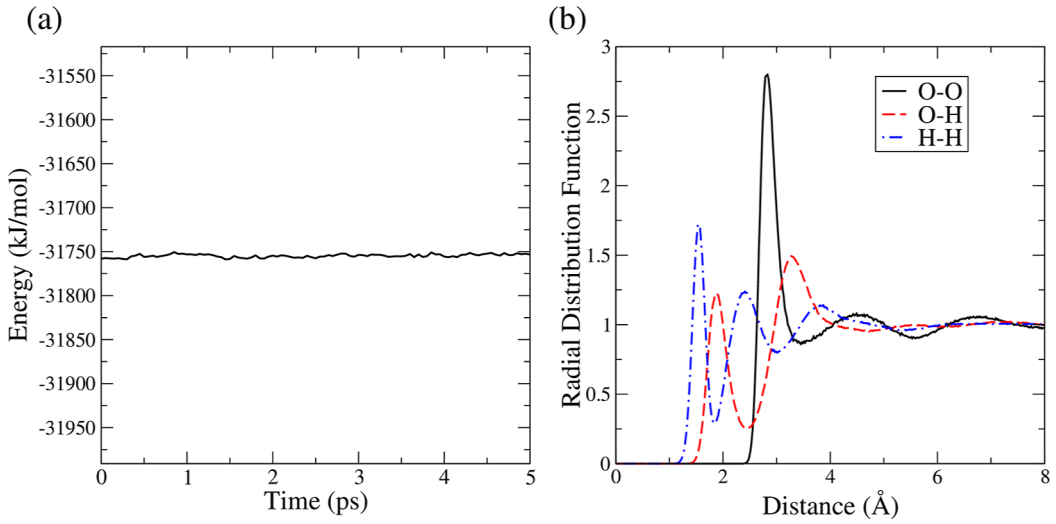


Figure 3. Simulation results of polarizable water FF with fluctuating charges and dispersion coefficients: (a) energy conservation in NVE simulation, the range of the *y*-axis of the figure is set to be the thermal fluctuation of total energy ($\pm\sqrt{kT^2C_V}$); (b) radial distribution functions from *NVT* path-integral MD simulation.

2.4. Model Deployment. Once the FF has been optimized using DMFF, it can be deployed in MD simulations. On one hand, for the conventional FF, DMFF is not as efficient as the highly optimized C++ code in mainstream MD programs like LAMMPS[34] or OpenMM. In the current stage, DMFF does not pursue extreme performance in large-scale simulations but rather focuses on fast FF optimization. Therefore, for conventional FFs, it is recommended to deploy the model directly in OpenMM. For this purpose, DMFF provides a "render" function that can export the parameter object into an xml file, which is ready to be used in OpenMM. On the other hand, for developers who are exploring complex function forms that are not well supported by the mainstream MD codes, DMFF can be also used as the force computation engine. In these scenarios, as we discussed in the Introduction section, we often need to perform small-scale validation runs, so the bottleneck is often the coding process, rather than the production run. For this purpose, we implemented an i-PI[35] interface for DMFF, such that MD simulations can be run directly using the DMFF force engine. In the future, interfaces with other more efficient MD programs (such as LAMMPS) are also to be developed.

3. RESULTS AND DISCUSSION

As discussed in the Introduction section, the automatic differentiation technique can be beneficial to both fast implementation of force/virial tensors as well as FF optimization. In the next few sections, we will give a few application examples of DMFF, demonstrating its capability. The computational details for all examples can be found in the Supporting Information.

3.1. Application in Advanced MD Simulation. In modern FF development, one often has to design a complex model to predict atomic parameters and use it in conjunction with conventional algorithms such as PME. A couple of examples include:

1. Considering the geometry dependence of charges and polarizabilities, it is natural to use ML models to predict these parameters and feed them into a polarizable FF to compute the electrostatic energies.[16]
2. Similarly, one can use ML models to predict electronegativities and hardness, then use them in charge equilibration models.[36] Similar methods can also be utilized to perform constant potential simulations for electrodes.

Currently, each research group that develops a new atomic parameter model typically has to implement it in MD from scratch. Such a process is extremely tedious and requires highly skillful developers, and thus can only be done by a few specialized research groups. Therefore, there is an essential barrier between building the model and validating it in real MD simulation, which limits the throughput of FF development. Meanwhile, it is noticed that some of the computational kernels (such as PME, pairwise interactions, and minimization with respect to charges and dipoles) are quite common but still need to be implemented repeatedly in different scenarios. It would be ideal if we can design the most commonly used modules in advance and assemble or extend them at will when exploring new potential forms. Previously such modular FF development is not feasible due to the needs of force and virial tensor calculations: differentiating through different modules can be cumbersome. The automatic differentiable DMFF package is designed to solve this problem. All commonly used molecular FF kernels are implemented with a user-friendly interface, which can be recombined easily to form new potentials.

To show the convenience brought by DMFF, we give a proof-of-concept example, in which a new water model[37] is implemented. This water model is based on a multipolar polarizable potential, which is quite similar to AMOEBA and MPID.[37] But the atomic charges and leading dispersion coefficients are not constants but can fluctuate linearly depending on the bond lengths and bond angles. While this combined fluctuating charge and polarizable model can significantly improve the description of the intermolecular electrostatic energy, its implementation requires an intensive modification to the existing MD program. However, by wrapping the DMFF PME backend with the charge/dispersion coefficients computation kernel, the force and virial tensor of this model can be computed in a few tens of lines of Python code. A sample of this code is included in the examples folder of

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

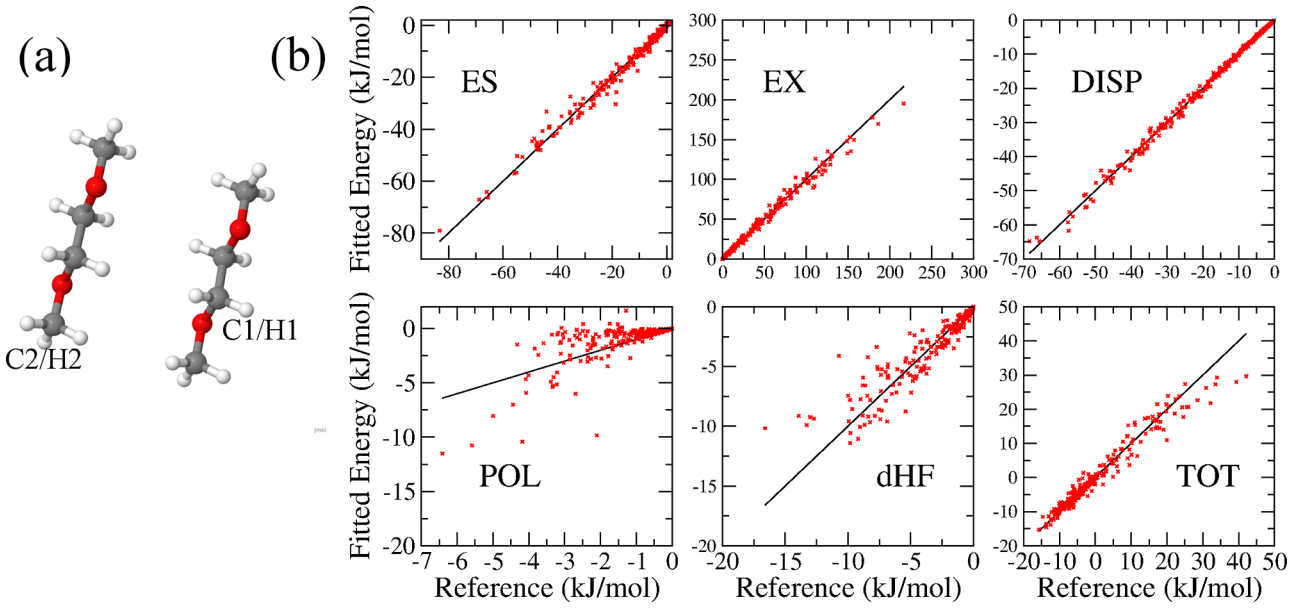


Figure 4. Bottom-up fitting result of COCCOC dimer: (a) dimer structure and atom-type definition: C2/H2 (terminal) and C1/H1 (internal) labels different atom types according to their chemical environments; (b) fitting results of each component and the total energy. The energy components include ES (electrostatics), EX (exchange), DISP (dispersion), POL (polarization), dHF (dhf), and TOT (total) energy, arranged in left-to-right, top-to-bottom order.

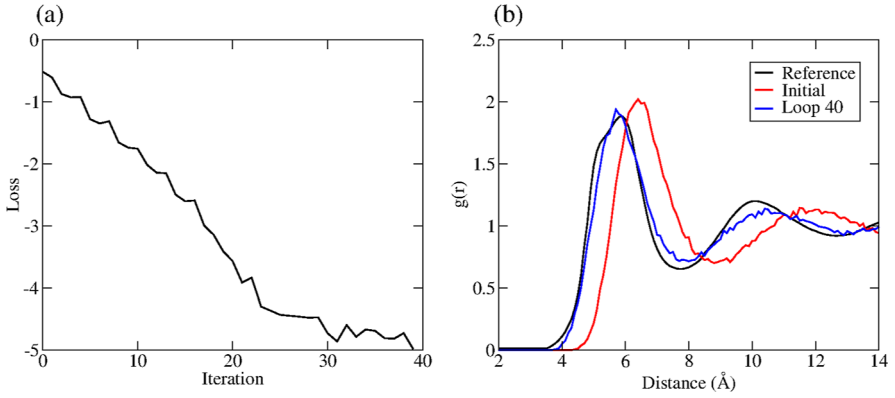


Figure 5. Change in the (a) loss function of the benzene RDF optimization and (b) RDF of liquid benzene before and after the optimization, plotted along the side with the experimental reference.

the DMFF package. We can then interface it with MD packages such as i-PI[38] to perform MD simulations, the results of which are shown in Figure 3. In the NVE simulation, the energy is well conserved within 0.02% of precision, verifying the smoothness of the potential and the correctness of the force evaluation. And the RDFs from the *NVT* path-integral MD simulation are also shown, demonstrating the capability of DMFF in bulk simulations. DMFF allows the developers to really focus on the part they are developing (i.e., the charge model), not concerning themselves with reinventing the wheel (i.e., the multipolar PME kernel) every time.

As discussed in Section 2.4, we note that DMFF does not pursue extreme performance in simulation speed, and the differentiable JAX implementation is indeed slower compared to the conventional C++ codes. Using a water box with 1017 water molecules and the MPID multipolar polarizable FF[18] as an example, we compare the efficiency of the DMFF implementation with the c++ MPID plugin in OpenMM, and DMFF is overall four times slower in this case. It is noted that the relative performance of DMFF depends on the potential as well as the system size, but a 3−10 times slow down can be generally observed. We note that more detailed and systematic profiling is required to identify the computational bottlenecks of DMFF in different scenarios. The possible hot spots that we may improve in the future include: (1) accelerating the neighbor list construction, which by nature is hard to JIT and runs particularly slow in Python; and (2) optimizing the interface between the DMFF force engine and mainstream MD programs to minimize data communication between CPU and GPU. Nevertheless, when validating complex advance FFs, the bottleneck is often the implementation of the potential, and a 3−10 times slowdown in MD simulation is often acceptable.

Besides being used to compute forces for new potentials, DMFF is also a powerful tool for FF parameterization. It can be used in both bottom-up and top-down FF development with different types of object functions, as we will show in the next few sections.

3.2. Bottom-Up Fitting. In the bottom-up FF development, we need to fit the energies or the forces to ab initio results. The object functions are typically the root mean square error

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

(RMSE) or weighted RMSE of energies and forces, which can be directly computed using the DMFF FF calculator. The JAX implementation of the molecular FFs enables us to utilize the state-of-the-art optimizers developed in the ML community, designed to perform robustly when fitting a large number of nonlinear parameters. Here, we show a complicated energy fitting example, the goal of which is to fit the short-range and damping parts of an ab initio potential.[39] The total energy is decomposed into several physically meaningful terms, including exchange (ex), electrostatic (es), dispersion (disp), polarization (pol), and dHF (dhf) interactions. All these terms can be computed using the ab initio Symmetry Adapted Perturbation Theory (SAPT)[40] method, and the short-range parts of them are then fitted using the following functions

$$
\begin{aligned}
V_{\mathrm{ex}} &= \sum_{i<j} A_{ij}^{\mathrm{ex}} P(B_{ij}, r_{ij}) \exp(-B_{ij} r_{ij}) \\
V_{\mathrm{es}}^{\mathrm{sr}} &= \sum_{i<j} \left\{ -A_{ij}^{\mathrm{es}} P(B_{ij}, r_{ij}) \exp(-B_{ij} r_{ij}) + [f_1(B_{ij} r_{ij}) - 1] \frac{q_i q_j}{r_{ij}} \right\} \\
V_{\mathrm{disp}}^{\mathrm{sr}} &= \sum_{i<j} \left\{ -A_{ij}^{\mathrm{disp}} P(B_{ij}, r_{ij}) \exp(-B_{ij} r_{ij}) + \sum_{n=6,8,10} [1 - f_n(x)] \frac{C_{ij}^n}{r_{ij}^n} \right\} \\
V_{\mathrm{pol}}^{\mathrm{sr}} &= \sum_{i<j} -A_{ij}^{\mathrm{pol}} P(B_{ij}, r_{ij}) \exp(-B_{ij} r_{ij}) \\
V_{\mathrm{dhf}} &= \sum_{i<j} -A_{ij}^{\mathrm{dhf}} P(B_{ij}, r_{ij}) \exp(-B_{ij} r_{ij}) \\
A_{ij} &= A_i A_j \\
B_{ij} &= \sqrt{B_i B_j} \\
C_{ij}^n &= \sqrt{C_i^n C_j^n} \\
f_n(x) &= 1 - e^{-x} \sum_{k=1}^{n} \frac{x^k}{k!} \\
P(B_{ij}, r_{ij}) &= \frac{1}{3} (B_{ij} r_{ij})^2 + B_{ij} r_{ij} + 1 \\
x &= B_{ij} r_{ij} - \frac{2 B_{ij}^2 r_{ij} + 3 B_{ij}}{B_{ij}^2 r_{ij}^2 + 3 B_{ij} r_{ij} + 3} r_{ij}
\end{aligned}
\tag{11}
$$

Each short-range term depends on its own prefactor A_i, and all terms share the same damping exponent B_i, which is in principle the reciprocal of the atom size. The prefactors and the exponents are strongly coupled, and both parameters are difficult to optimize due to their nonlinearity. The final loss function is a weighted average of the RMSEs of all components and the total energy

$$
\begin{aligned}
L = {} & \lambda_{\mathrm{tot}} \sum (V_{\mathrm{tot}} - V_{\mathrm{tot}}^{\mathrm{ref}})^2 + \lambda_{\mathrm{ex}} \sum (V_{\mathrm{ex}} - V_{\mathrm{ex}}^{\mathrm{ref}})^2 \\
& + \lambda_{\mathrm{es}} \sum (V_{\mathrm{es}}^{\mathrm{sr}} - V_{\mathrm{sr,es}}^{\mathrm{ref}})^2 \\
& + \lambda_{\mathrm{ind}} \sum (V_{\mathrm{ind}}^{\mathrm{sr}} - V_{\mathrm{sr,ind}}^{\mathrm{ref}})^2 + \lambda_{\mathrm{dhf}} \sum (V_{\mathrm{dhf}} - V_{\mathrm{dhf}}^{\mathrm{ref}})^2 \\
& + \lambda_{\mathrm{disp}} \sum (V_{\mathrm{disp}}^{\mathrm{sr}} - V_{\mathrm{sr,disp}}^{\mathrm{ref}})^2
\end{aligned}
\tag{12}
$$

Using DMFF and the ADAM optimizer in optax, we conduct such a fitting on the COCCOC dimer system, the results of which are shown in Figure 4.The major terms such as electrostatic, exchange, and dispersion interactions are well fitted, and the fitting quality of the total energy is also reasonable. The polarization and dHF terms are less accurate but they are much less important in this system anyway. More importantly, the obtained parameters show a quite reasonable physical trend: for example, the B_i of H1/H2 (41.9 and 45.1 nm^{-1}) is larger than that of O (36.6 nm^{-1}), which is then larger than that of C1/C2 (34.4 and 33.1 nm^{-1}). This is exactly the reverse order of the atom sizes of the three elements (i.e., C > O > H).

In this example, we show how ML optimizers can be conveniently utilized in developing complicated physics-driven FFs, using DMFF. This is a relatively straightforward way to take advantage of an automatic differentiable FF calculator. Meanwhile, the reweighting MBAR estimator implemented in DMFF makes it an even more powerful tool in top-down parameter optimization, as we will show in the next two sections.

3.3. Ensemble Averages. In classical FF parameterization, we always need to make the FF model able to reproduce the experimentally measured thermodynamic properties, such as density, vaporization enthalpy, or RDF. To verify the validity of the reweighting MBAR estimator for physical property estimation, we attempted to optimize the coarse grain potential of the liquid benzene and dimethyl carbonate (DMC) systems using the DMFF optimization framework. The intramolecular terms were initialized using GAFF[3] with AM1-BCC charge. The optimization targets are the experimental density and the center-of-mass (COM) RDF from neutron diffraction at 30 °C (303 K). During the optimization, we keep monitoring the effective sample size of the estimator, and resampling will be run if the sample size is smaller than a predetermined threshold.

As an example of estimating scalar property, we define the optimization loss function for density as

$$
L_{\mathrm{density}} = \left(\sum_{n=1}^{N} W_n \rho_n - \rho_{\mathrm{ref}} \right)^2 \tag{13}
$$

On the other hand, vector properties, such as various types of structural distribution histograms, can also be optimized using the reweighting MBAR estimator. As an example, we first aim to build a coarse-grained benzene model with three beads per molecule and then parameterized the Lennard-Jones interaction between particles to reproduce the experimental RDF of liquid benzene. The loss function is the natural log of the L2-norm of the vector difference

$$
L_{\mathrm{RDF}} = \log \left\| \sum_{n=1}^{N} W_n \boldsymbol{\nu}_n - \boldsymbol{\nu}_{\mathrm{ref}} \right\| \tag{14}
$$

in which $\boldsymbol{\nu}_n$ is the vector of the RDF curve in each sample frame and $\boldsymbol{\nu}_{\mathrm{ref}}$ is the reference RDF curve. The loss function used in the real optimization procedure is a combination of density and RDF losses

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

$$L_{\text{tot}} = L_{\text{RDF}} + \alpha L_{\text{density}} \tag{15}$$

With $\alpha = 10\ \text{g}^{-2}\ \text{cm}^6$ in the benzene case to keep the units of the two terms consistent. It was found that the addition of density penalty is critical when optimizing RDF. Otherwise, the optimizer would occasionally explore nonphysical regions in the parameter space, leading to a liquid−gas phase transition and the failure of the optimization. Such behavior highlights the importance of a carefully chosen loss function, which is the key to the numerical stability and the transferability of the optimized results. The design of the most appropriate loss function is probably a system- and application-dependent problem and is a subject of our future research. While DMFF, as a tool, provides the flexibility to allow users to define their targets freely, it is highly adaptable to different application scenarios.

The changes in the loss function and RDF along the optimization process are shown in Figure 5a,b. The density of the system changes from 0.526 ± 0.002 to 0.770 ± 0.003 g/mL, which is much closer to the experimental reference of 0.876 g/mL. The optimized RDF also matches the experimental data with excellent precision. In classical FF optimizations, the Lennard-Jones parameters were always manually tuned to reproduce densities, which is not only low-efficient but also unreliable. With the help of DMFF, such a process can be largely automated. Moreover, it can be done on a larger scale, with more parameters and more training data from multiple systems, thus improving the generality of the resulting FF.

Moreover, we also selected DMC as an additional case for coarse-grained (CG) model fitting as shown in Figure 6. We can

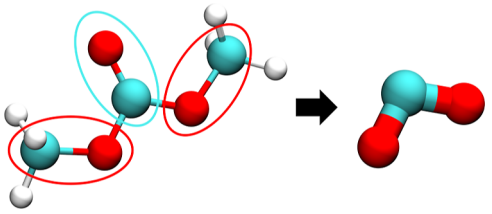


Figure 6. Scheme of the 3-site CG model for dimethyl carbonate.

parameterize the model by adjusting four parameters to reproduce the RDF obtained from the OPLS-AA all-atom simulation, including two σ and two ϵ. The charge contribution was computed with PME using OpenMM (in sampling) and DMFF (in reweighting) but not optimized. Compared to the CG model of benzene, the CG DMC is more challenging as it involves more atom types with more adjustable parameters. As a widely used organic solvent in electrolytes, it also represents a more application-oriented test case. In every iteration of the optimization process, molecular dynamics simulations were conducted for a duration of 1 ns. The initial 250 ps trajectory was disregarded to account for equilibration, and subsequently, conformations were stored at 5 ps intervals. As a result, 150 samples were obtained in each resampling stage. If Kish's effective sample size fell below 75 samples, which corresponds to half of the total size, the resampling process was repeated and the trajectory data were updated accordingly. In the case of DMC, for each iteration of estimating the loss using reweighting MBAR, we also performed plain MD sampling to evaluate the reference loss and gradients (see Figure 7). Compared with the loss and gradient estimated by reweighting samples from other potential functions, the loss and gradient estimated by sampling using the current potential function exhibit the highest accuracy and can be regarded as reference values.

To assess the consistency between the gradient of loss estimated via reweighting with respect to parameter vector $\mathbf{v}$ $\left(\frac{\partial L_{\text{reweight}}}{\partial \mathbf{v}}\right)$ and the gradient obtained from resampling $\left(\frac{\partial L_{\text{ref}}}{\partial \mathbf{v}}\right)$, we can compute the projection length of $\frac{\partial L_{\text{reweight}}}{\partial \mathbf{v}}$ onto the direction of $\frac{\partial L_{\text{ref}}}{\partial \mathbf{v}}$.

$$\mathbf{g}_{\mathbf{v}} = \frac{\partial L_{\text{reweight}}}{\partial \mathbf{v}}$$

$$\mathbf{g}_{\mathbf{v}}^{\text{ref}} = \frac{\partial L_{\text{ref}}}{\partial \mathbf{v}}$$

$$\mathbf{g}_{\mathbf{v}}^{\parallel} = \mathbf{g}_{\mathbf{v}} \cdot \mathbf{g}_{\mathbf{v}}^{\text{ref}} / |\mathbf{g}_{\mathbf{v}}^{\text{ref}}| \tag{16}$$

The results are also shown in Figure 7.

Figure 7a shows that the loss estimated by reweighting from previous samples is comparable to the loss from the resampled values at each step. In Figure 7b, after 60 optimization rounds, the peaks of RDF become significantly closer to the reference curve. Figures 7c,d reveal that the gradients estimated by reweighting are mostly consistent with resampled gradients in terms of direction. Moreover, the reweighting-derived gradients with respect to σ are often lower than those from resampling, possibly due to the distribution difference in reweighting. The frequency of resample can be also seen clearly in Figure 7c,d: resample happens in 24 out of the 60 optimization iterations, and it happens less frequently in the later stage of the optimization (only 10 times in iterations 30−60) when the change of parameters is smaller. Both the gradients of σ and ϵ show a general trend of decay, indicating the optimization approaching a local minimum in the parameter space.

3.4. Free Energies. Compared to ensemble-averaged properties such as density and RDF, fitting to free energy difference is an important, yet even more challenging task.[41] For example, hydration-free energy (HFE) is a critical physiochemical property that is of great interest to drug discovery communities. The HFE calculation is a primary step in the thermodynamic cycle used in the calculation of relative or absolute protein−ligand binding free energy, which is the primary target to optimize drug design projects. Therefore, HFEs of small organic molecules are often used to validate the accuracy and transferability of force fields for organic (or drug-like) molecules, such as OPLS[42−44] or GAFF.[3] In these fixed charge models, polarization effects are not explicitly included, so atomic charges need to be carefully tuned to give a balanced description of the electrostatics.

In practice, models based on bond charge correction (BCC) are widely employed to efficiently assign atomic partial charges and were reported to reach fair results on benchmark sets of protein−ligand binding affinities.[45,46] In this approach, atomic charges are firstly determined using the Mulliken charges computed at the semi-empirical level of theory (e.g., AM1 or CM1A). Then, they are further adjusted using BCC, which is a set of parameters that only depends on the molecular topology. The BCC parameters were initially fitted to emulate vacuum ESP from QM calculation at HF/6-31G* level, and then the parameters associated with some functional groups were refined to reproduce the experimental HFEs of some model molecules containing these functional groups. Traditionally, such fine-tuning had to be conducted manually by experienced computa-

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

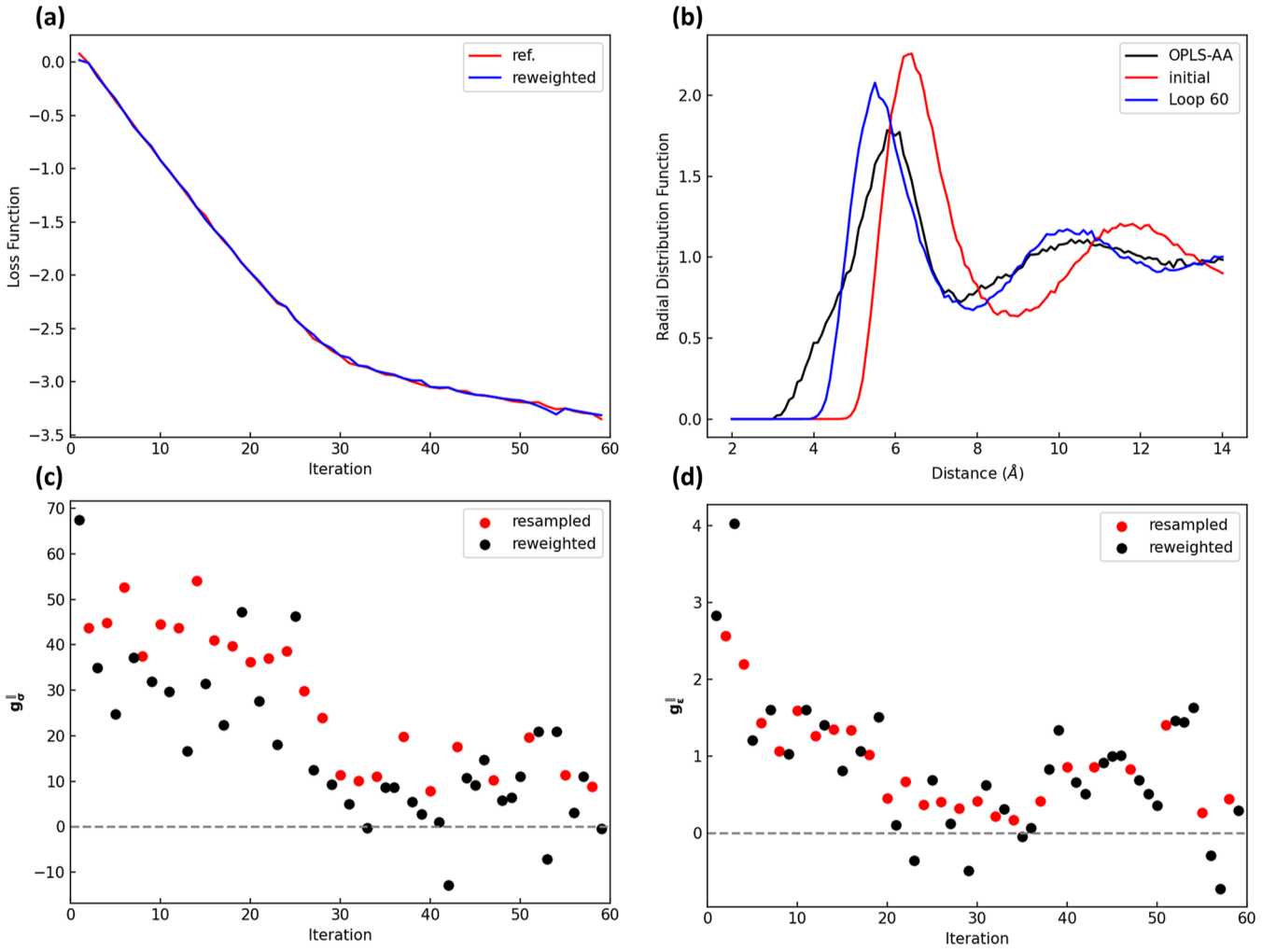


Figure 7. (a) Variation of the loss function during the optimization of the CG model for DMC. The blue line represents the loss calculated using the reweighted MBAR estimator, while the red line serves as a reference, corresponding to the loss computed based on the plain MD sampling with the current parameters. (b) RDF before and after optimization. (c,d) Length of $\mathbf{g}_\sigma$ (c) and $\mathbf{g}_\epsilon$ (d) projected in the $\mathbf{g}_\sigma^{ref}$ and $\mathbf{g}_\epsilon^{ref}$ direction. Positive for the same direction and negative for the opposite direction.

tional chemists with excellent intuitions, which lacks automation and reproducibility. However, herein we demonstrated that with DMFF, this process now can be conducted automatically using gradient-based optimization algorithms.

Note that the Gibbs free energy G of state k can be calculated as

$$G_k = -\beta^{-1}\ln Q = -\beta^{-1}\ln \int \mathrm{d}\mathbf{x} e^{-\beta[U_k(\mathbf{x};\theta)+PV]} \tag{17}$$

where $\beta = 1/k_B T$, k_B is the Boltzmann factor, $\mathbf{x}$ is particles' coordinates and momenta, Q is the partition function, U_k is the energy function of state k defined by the force field, P is the pressure, V is the volume, and θ refers to force field parameters. The parametric gradient of the free energy can be evaluated as the ensemble average of the gradient of the potential energy with respect to force field parameters, which can be calculated by DMFF with an automatic differentiation technique

$$\begin{aligned}\frac{\partial G_k}{\partial \theta} &= -\beta^{-1}\left(\frac{1}{Q}\frac{\partial Q}{\partial \theta}\right) \\ &= \frac{1}{Q}\int \mathrm{d}\mathbf{x}\cdot\frac{\partial U}{\partial \theta}\cdot e^{-\beta[U_k(\mathbf{x};\theta)+PV]} \\ &= \left\langle \frac{\partial U_k}{\partial \theta}\right\rangle_k\end{aligned} \tag{18}$$

In terms of HFE, the calculation follows an alchemical pathway, in which the interaction between solute and water is gradually turned off by a parameter λ, and the free energy is estimated by thermodynamic integration or MBAR.

$$U(\lambda;\theta) = (1-\lambda)U_0(\theta) + \lambda U_1(\theta) \tag{19}$$

$$\Delta G(\theta) = G(\lambda = 1;\theta) - G(\lambda = 0;\theta) \tag{20}$$

Two subscripts "0" and "1" denotes the initial ("decoupled") state and the final ("coupled") state, in which the solute−water interaction is totally turned off and on, respectively. The gradient of HFE with respect to force field parameters can be derived by combining eqs 18 and 20.

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

$$\frac{\partial \Delta G}{\partial \theta} = \left\langle \frac{\partial U_1}{\partial \theta} \right\rangle_1 - \left\langle \frac{\partial U_0}{\partial \theta} \right\rangle_0 \quad (21)$$

As a proof of concept, we optimized four BCC parameters in the GAFF2 model related to esters, one functional group which presents a well-known issue where partial charges derived only from QM ESP failed to accurately predict the hydration-free energies.[43] We selected 27 aliphatic ester molecules with experimental data from the FreeSolv v0.52 database[47] as the training set. In order to minimize sampling error and avoid overfitting problems, these molecules are chosen to be as simple as possible and contain only the ester group, i.e., with formula R_1COOR_2 where R_1 and R_2 are alkyl groups. A full list of the molecules and their 2D topologies can be found in the Supporting Information. The mean squared deviation defined in eq 22 is chosen as the loss function and the steepest gradient descent algorithm with a learning rate of 0.005 is utilized for the optimization. In this equation, the symbol α represents the prefactor for free energy loss and is 1 $mol^2/kcal^2$. Details about Alchemica MD simulations are described in the Supporting Information.

$$L = \frac{\alpha}{2M} \sum_{m=1}^{M} (\Delta G_{\mathrm{expt},m} - \Delta G_{\mathrm{calc},m})^2 \quad (22)$$

As illustrated in Figure 8, the optimization successfully reached convergence after 8 iterations and the RMSE of

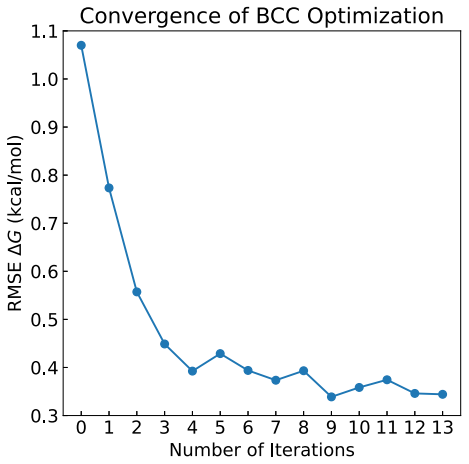


Figure 8. RMSE of hydration-free energies plotted against the number of optimization iterations.

hydration-free energies decreased by 60% to around 0.4 kcal/mol from 1.07 kcal/mol. Note that in order to reduce the computational cost, only 200 ps simulations in each window were performed during the optimization process. We assumed short simulations were adequate to provide a correct direction of gradient descent, rather than giving accurate free energy outcomes. This assumption is further verified by three independent trials of longer (5 ns) MD simulations with optimized BCC parameters. As shown in Table 1 and Figure 9, the initial BCC parameters with the GAFF2 model systematically overestimated the HFE by having a mean singed error (MSE) of 0.90 kcal/mol, but this trend is fixed after the optimization by having an MSE very close to zero. Also, the RMSE with initial BCC parameters is up to 1.08 kcal/mol but decreases to 0.36 kcal/mol after optimization using DMFF. The HFEs of all molecules except one got improved after BCC

Table 1. Comparison of Metrics of Calculated Hydration-Free Energies with Respect to Experimental Values[a]

metric	initial BCC	optimized BCC
MSE	0.90	−0.03
MAE	0.90	0.24
RMSE	1.08	0.36

[a]MSE stands for mean signed error. MAE stands for mean absolute error. RMSE stands for the root of mean squared error. All values in kcal/mol unit.

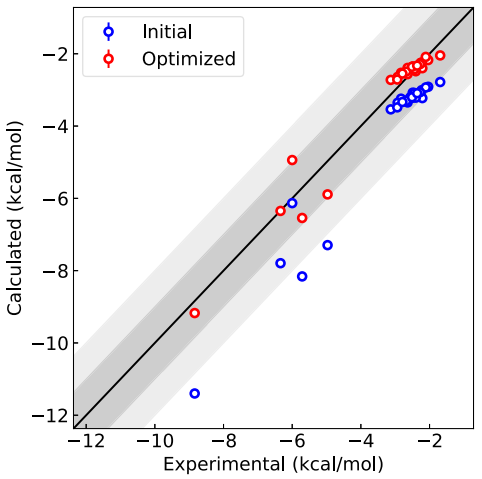


Figure 9. Correlation between the experimental and the calculated hydration-free energies with initial BCC parameters (blue) and optimized BCC parameters (red). The light and the dark-gray areas mark the regions with an error smaller than 2 and 1 kcal/mol, respectively.

refinement, which demonstrates the ability of DMFF to optimize parameters against experimental free energy data. The only exception is "mobely_2402487", a 1,3-dicarbonyl compound, whose primary electronic structure is different from normal esters, which suggests that the existing BCC parameters for esters are not in good compatibility with such compounds and thus more BCC parameters should be added to cover this type of chemical space. A list of calculated hydration-free energies of all 27 molecules and the optimized BCC values can be found in the Supporting Information.

4. CONCLUSIONS AND FUTURE PERSPECTIVES

To summarize, in this study, we present DMFF, a JAX-based differentiable molecular force field development tool. DMFF features a user-friendly frontend API that replicates the OpenMM behavior, allowing a convenient definition of molecular FFs. It also includes a variety of efficient backend kernels that support both conventional point charge models and advanced multipolar polarizable potentials. The potential functions defined in DMFF are easy to be extended and recombined to form new advanced FFs. The implementation of forces and virial tensors can be largely simplified, so the MD validation of complex FFs can be performed with much less effort. Based on the differentiable FF engine, a variety of object functions can be used for FF optimization, including energies, forces, and macroscopic properties such as free energies and ensemble-averaged quantities. It is demonstrated that using DMFF, both bottom-up and top-down FF optimizations can be performed in an automatic fashion. Parameters can be obtained with better performances in both bulk organic liquid simulations

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909

and HFE calculations. Therefore, DMFF serves as an excellent tool for the development of next-generation molecular FFs.

We note that DMFF is currently still under active development. Being open-source, we expect it to become an ever-improving useful tool with a community effort. In the near future, more flexible support for the function forms of the FF model, optimization schemes using dynamical properties, and interface with more MD engines, are to be developed. As a beginning, we have incorporated an implementation of the DMFF potential function using the TensorFlow C++ interface within OpenMM (https://github.com/dingye18/openmm_dmff_plugin). This implementation will be ported to general molecular dynamics engines such as GROMACS in the future. Meanwhile, it is also part of the plan to release the C++ interface as a standalone API that can be used by any MD code developer. All activities will be conducted in the DeepModeling community (https://github.com/deepmodeling/). In addition, any issues, pull requests, suggestions, and potential collaborations are welcome.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jctc.2c01297.

Initial and final parameters of the benzene and DMC fitting cases (ZIP)

Computational details and additional results (PDF)

■ AUTHOR INFORMATION

Corresponding Authors

Linfeng Zhang − *AI for Science Institute, Beijing 100080, P. R. China*; orcid.org/0000-0002-8470-5846; Email: zhanglf@dp.tech

Kuang Yu − *Tsinghua-Berkley Shenzhen Institute, Shenzhen, Guangdong 518055, P. R. China; Tsinghua Shenzhen International Graduate School, Shenzhen, Guangdong 518055, P. R. China*; orcid.org/0000-0001-9142-5263; Email: yu.kuang@sz.tsinghua.edu.cn

Authors

Xinyan Wang − *DP Technology, Beijing 100080, P. R. China*

Jichen Li − *DP Technology, Beijing 100080, P. R. China*

Lan Yang − *Tsinghua-Berkley Shenzhen Institute, Shenzhen, Guangdong 518055, P. R. China*; orcid.org/0000-0002-8664-8929

Feiyang Chen − *DP Technology, Beijing 100080, P. R. China*

Yingze Wang − *DP Technology, Beijing 100080, P. R. China*

Junhan Chang − *DP Technology, Beijing 100080, P. R. China*

Junmin Chen − *Tsinghua-Berkley Shenzhen Institute, Shenzhen, Guangdong 518055, P. R. China*; orcid.org/0000-0002-6069-9162

Wei Feng − *DP Technology, Beijing 100080, P. R. China*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jctc.2c01297

Author Contributions

⊥X.W. and J.L. contribute equally.

Notes

The authors declare no competing financial interest.

A tutorial for beginner users is avaliable at https://bohrium.dp.tech/notebook/f305009510dc49bea9f10cc3c92754aa.

■ ACKNOWLEDGMENTS

The authors thank the National Natural Science Foundation of China (Grant Number: 22103048), Shenzhen Bay Laboratory (grant number: SZBL2021080601013), and Tsinghua Shenzhen International Graduate School (grant number: HW2020009) for their financial support of this work. The authors also thank Lei Wang from the Institute of Physics, Chinese Academy of Sciences, and Han Wang of the Beijing Institute of Applied Physics and Computational Mathematics for helpful discussions.

■ REFERENCES

(1) de Pablo, J. J.; Jackson, N. E.; Webb, M. A.; Chen, L.-Q.; Moore, J. E.; Morgan, D.; Jacobs, R.; Pollock, T.; Schlom, D. G.; Toberer, E. S.; Analytis, J.; Dabo, I.; DeLongchamp, D. M.; Fiete, G. A.; Grason, G. M.; Hautier, G.; Mo, Y.; Rajan, K.; Reed, E. J.; Rodriguez, E.; Stevanovic, V.; Suntivich, J.; Thornton, K.; Zhao, J.-C. New frontiers for the materials genome initiative. *npj Comput. Mater.* **2019**, *5*, 41.

(2) Surabhi, S.; Singh, B. K. Computer Aided Drug Design: An Overview. *J. Drug Deliv. Therapeut.* **2018**, *8*, 504−509.

(3) Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. Development and testing of a general amber force field. *J. Comput. Chem.* **2004**, *25*, 1157−1174.

(4) Rappe, A. K.; Casewit, C. J.; Colwell, K. S.; Goddard, W. A.; Skiff, W. M. UFF, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *J. Am. Chem. Soc.* **1992**, *114*, 10024−10035.

(5) Vanommeslaeghe, K.; MacKerell, A. D. CHARMM additive and polarizable force fields for biophysics and computer-aided drug design. *Biochim. Biophys. Acta* **2015**, *1850*, 861−871.

(6) Jorgensen, W. L.; Maxwell, D. S.; TiradoRives, J. Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. *J. Am. Chem. Soc.* **1996**, *118*, 11225−11236.

(7) Fröhlking, T.; Bernetti, M.; Calonaci, N.; Bussi, G. Toward empirical force fields that match experimental observables. *J. Chem. Phys.* **2020**, *152*, 230902.

(8) Wang, L.-P.; Chen, J.; Van Voorhis, T. Systematic Parametrization of Polarizable Force Fields from Quantum Chemistry Data. *J. Chem. Theory Comput.* **2013**, *9*, 452−460.

(9) Wang, L.-P.; Martinez, T. J.; Pande, V. S. Building Force Fields: An Automatic, Systematic, and Reproducible Approach. *J. Phys. Chem. Lett.* **2014**, *5*, 1885−1891.

(10) Boothroyd, S.; Wang, L.-P.; Mobley, D. L.; Chodera, J. D.; Shirts, M. R. Open Force Field Evaluator: An Automated, Efficient, and Scalable Framework for the Estimation of Physical Properties from Molecular Simulation. *J. Chem. Theory Comput.* **2022**, *18*, 3566−3576.

(11) Behler, J.; Parrinello, M. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Phys. Rev. Lett.* **2007**, *98*, 146401.

(12) Zhang, L.; Han, J.; Wang, H.; Car, R.; E, W. Deep Potential Molecular Dynamics: a scalable model with the accuracy of quantum mechanics. *Phys. Rev. Lett.* **2018**, *120*, 143001.

(13) Unke, O. T.; Meuwly, M. PhysNet: A Neural Network for Predicting Energies, Forces, Dipole Moments, and Partial Charges. *J. Chem. Theory Comput.* **2019**, *15*, 3678−3693.

(14) Zhang, Y.; Hu, C.; Jiang, B. Embedded Atom Neural Network Potentials: Efficient and Accurate Machine Learning with a Physically Inspired Representation. *J. Phys. Chem. Lett.* **2019**, *10*, 4962−4967.

(15) Wang, X.; Xu, Y.; Zheng, H.; Yu, K. A Scalable Graph Neural Network Method for Developing an Accurate Force Field of Large Flexible Organic Molecules. *J. Phys. Chem. Lett.* **2021**, *12*, 7982−7987.

(16) Kumar, A.; Pandey, P.; Chatterjee, P.; MacKerell, A. D. Deep Neural Network Model to Predict the Electrostatic Parameters in the Polarizable Classical Drude Oscillator Force Field. *J. Chem. Theory Comput.* **2022**, *18*, 1711−1725.

(17) Shi, Y.; Xia, Z.; Zhang, J.; Best, R.; Wu, C.; Ponder, J. W.; Ren, P. Polarizable Atomic Multipole-Based AMOEBA Force Field for Proteins. *J. Chem. Theory Comput.* **2013**, *9*, 4046−4063.
(18) Huang, J.; Simmonett, A. C.; Pickard, F. C.; MacKerell, A. D.; Brooks, B. R. Mapping the Drude polarizable force field onto a multipole and induced dipole model. *J. Chem. Phys.* **2017**, *147*, 161702.
(19) Das, A. K.; Urban, L.; Leven, I.; Loipersberger, M.; Aldossary, A.; Head-Gordon, M.; Head-Gordon, T. Development of an Advanced Force Field for Water Using Variational Energy Decomposition Analysis. *J. Chem. Theory Comput.* **2019**, *15*, 5001−5013.
(20) Doerr, S.; Majewski, M.; Pérez, A.; Krämer, A.; Clementi, C.; Noe, F.; Giorgino, T.; De Fabritiis, G. TorchMD: A Deep Learning Framework for Molecular Simulations. *J. Chem. Theory Comput.* **2021**, *17*, 2355−2363.
(21) Schoenholz, S.; Cubuk, E. D. JAX MD: A Framework for Differentiable Physics. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 11428−11441.
(22) Huang, Y.-P.; Xia, Y.; Yang, L.; Wei, J.; Yang, Y. I.; Gao, Y. Q. SPONGE: A GPU-Accelerated Molecular Dynamics Package with Enhanced Sampling and AI-Driven Algorithms. *Chin. J. Chem.* **2022**, *40*, 160−168.
(23) Thaler, S.; Zavadlav, J. Learning neural network potentials from experimental data via Differentiable Trajectory Reweighting. *Nat. Commun.* **2021**, *12*, 6884.
(24) Kaymak, M. C.; Rahnamoun, A.; O'Hearn, K. A.; van Duin, A. C. T.; Merz, K. M.; Aktulga, H. M. JAX-ReaxFF: A Gradient-Based Framework for Fast Optimization of Reactive Force Fields. *J. Chem. Theory Comput.* **2022**, *18*, 5181−5194.
(25) Bradbury, J.; Frostig, R.; Hawkins, P.; Johnson, M. J.; Leary, C.; Maclaurin, D.; Necula, G.; Paszke, A.; VanderPlas, J.; Wanderman-Milne, S.; Zhang, Q. *JAX: Composable Transformations of Python +NumPy Programs*; GitHub, 2018. http://github.com/google/jax.
(26) *DMFF Project*; GitHub, 2022. https://github.com/deepmodeling/DMFF.
(27) Eastman, P.; Swails, J.; Chodera, J. D.; McGibbon, R. T.; Zhao, Y.; Beauchamp, K. A.; Wang, L.-P.; Simmonett, A. C.; Harrigan, M. P.; Stern, C. D.; Wiewiora, R. P.; Brooks, B. R.; Pande, V. S. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. *PLoS Comput. Biol.* **2017**, *13*, No. e1005659.
(28) Essmann, U.; Perera, L.; Berkowitz, M. L.; Darden, T.; Lee, H.; Pedersen, L. G. A smooth particle mesh Ewald method. *J. Chem. Phys.* **1995**, *103*, 8577−8593.
(29) Babuschkin, I.; Baumli, K.; Bell, A.; Bhupatiraju, S.; Bruce, J.; Buchlovsky, P.; Budden, D.; Cai, T.; Clark, A.; Danihelka, I.; Fantacci, C.; Godwin, J.; Jones, C.; Hemsley, R.; Hennigan, T.; Hessel, M.; Hou, S.; Kapturowski, S.; Keck, T.; Kemaev, I.; King, M.; Kunesch, M.; Martens, L.; Merzic, H.; Mikulik, V.; Norman, T.; Quan, J.; Papamakarios, G.; Ring, R.; Ruiz, F.; Sanchez, A.; Schneider, R.; Sezener, E.; Spencer, S.; Srinivasan, S.; Wang, L.; Stokowiec, W.; Viola, F. *The DeepMind JAX Ecosystem*; GitHub, 2020. http://github.com/deepmind.
(30) Blondel, M.; Berthet, Q.; Cuturi, M.; Frostig, R.; Hoyer, S.; Llinares-López, F.; Pedregosa, F.; Vert, J.-P. Efficient and Modular Implicit Differentiation. 2022, arXiv:2105.15183 [cs, math, stat]. http://arxiv.org/abs/2105.15183.
(31) Norgaard, A. B.; Ferkinghoff-Borg, J.; Lindorff-Larsen, K. Experimental Parameterization of an Energy Function for the Simulation of Unfolded Proteins. *Biophys. J.* **2008**, *94*, 182−192.
(32) Bottaro, S.; Lindorff-Larsen, K.; Best, R. B. Variational Optimization of an All-Atom Implicit Solvent Force Field To Match Explicit Solvent Simulation Data. *J. Chem. Theory Comput.* **2013**, *9*, 5641−5652.
(33) Shirts, M. R.; Chodera, J. D. Statistically optimal analysis of samples from multiple equilibrium states. *J. Chem. Phys.* **2008**, *129*, 124105.
(34) Thompson, A. P.; Aktulga, H. M.; Berger, R.; Bolintineanu, D. S.; Brown, W. M.; Crozier, P. S.; in 't Veld, P. J.; Kohlmeyer, A.; Moore, S. G.; Nguyen, T. D.; Shan, R.; Stevens, M. J.; Tranchida, J.; Trott, C.; Plimpton, S. J. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. *PLoS Comput. Biol.* **2022**, *271*, 108171.
(35) Wang, X. *I-Pi-Driver*; Github Repository, 2020. https://github.com/WangXinyan940/i-pi-driver.
(36) Dajnowicz, S.; Agarwal, G.; Stevenson, J. M.; Jacobson, L. D.; Ramezanghorbani, F.; Leswing, K.; Friesner, R. A.; Halls, M. D.; Abel, R. High-Dimensional Neural Network Potential for Liquid Electrolyte Simulations. *J. Phys. Chem. B* **2022**, *126*, 6271−6280.
(37) Yang, L.; Li, J.; Chen, F.; Yu, K. A transferrable range-separated force field for water: Combining the power of both physically-motivated models and machine learning techniques. *J. Chem. Phys.* **2022**, *157*, 214108.
(38) Ceriotti, M.; More, J.; Manolopoulos, D. E. i-PI: A Python interface for ab initio path integral molecular dynamics simulations. *Comput. Phys. Commun.* **2014**, *185*, 1019−1026.
(39) Van Vleet, M. J.; Misquitta, A. J.; Stone, A. J.; Schmidt, J. R. Beyond Born−Mayer: Improved Models for Short-Range Repulsion in ab Initio Force Fields. *J. Chem. Theory Comput.* **2016**, *12*, 3851−3870.
(40) Williams, H. L.; Chabalowski, C. F. Using Kohn-Sham Orbitals in Symmetry-Adapted Perturbation Theory to Investigate Intermolecular Interactions. *J. Phys. Chem. A* **2001**, *105*, 646−659.
(41) Cournia, Z.; Chipot, C.; Roux, B.; York, D. M.; Sherman, W. *Free Energy Methods in Drug Discovery: Current State and Future Directions*; ACS Symposium Series 1397; American Chemical Society, 2021; Vol. *1397*; pp 1−38, Section: 1.
(42) Harder, E.; Damm, W.; Maple, J.; Wu, C.; Reboul, M.; Xiang, J. Y.; Wang, L.; Lupyan, D.; Dahlgren, M. K.; Knight, J. L.; Kaus, J. W.; Cerutti, D. S.; Krilov, G.; Jorgensen, W. L.; Abel, R.; Friesner, R. A. OPLS3: a force field providing broad coverage of drug-like small molecules and proteins. *J. Chem. Theory Comput.* **2016**, *12*, 281−296.
(43) Roos, K.; Wu, C.; Damm, W.; Reboul, M.; Stevenson, J. M.; Lu, C.; Dahlgren, M. K.; Mondal, S.; Chen, W.; Wang, L.; Abel, R.; Friesner, R. A.; Harder, E. D. OPLS3e: Extending force field coverage for drug-like small molecules. *J. Chem. Theory Comput.* **2019**, *15*, 1863−1874.
(44) Lu, C.; Wu, C.; Ghoreishi, D.; Chen, W.; Wang, L.; Damm, W.; Ross, G. A.; Dahlgren, M. K.; Russell, E.; Von Bargen, C. D.; Abel, R.; Friesner, R. A.; Harder, E. D. OPLS4: Improving force field accuracy on challenging regimes of chemical space. *J. Chem. Theory Comput.* **2021**, *17*, 4291−4300.
(45) Jakalian, A.; Bush, B. L.; Jack, D. B.; Bayly, C. I. Fast, efficient generation of high-quality atomic charges. AM1-BCC model: I. Method. *J. Comput. Chem.* **2000**, *21*, 132−146.
(46) Jakalian, A.; Jack, D. B.; Bayly, C. I. Fast, efficient generation of high-quality atomic charges. AM1-BCC model: II. Parameterization and validation. *J. Comput. Chem.* **2002**, *23*, 1623−1641.
(47) Mobley, D. L.; Guthrie, J. P. FreeSolv: a database of experimental and calculated hydration free energies, with input files. *J. Comput.-Aided Mol. Des.* **2014**, *28*, 711−720.

https://doi.org/10.1021/acs.jctc.2c01297
J. Chem. Theory Comput. 2023, 19, 5897−5909