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,[‡] Linfeng Zhang,[*,¶] and Kuang Yu[*,‡,§]

[†]*DP Technology, Beijing 100080, P. R. China*

[‡]*Tsinghua-Berkley Shenzhen Institute, Shenzhen 518055, Guangdong, P. R. China*

[¶]*AI for Science Institute, Beijing 100080, P. R. China*

[§]*Tsinghua Shenzhen International Graduate School, Shenzhen 518055, Guangdong, P. R. China*

E-mail: zhanglf@dp.tech; yu.kuang@sz.tsinghua.edu.cn

Abstract

In the simulation of molecular systems, the underlying force field (FF) model plays an extremely important role, 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

[#] These authors contribute equally.

FF development platform based on 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 force field optimization workflow. DMFF also eases the evaluation of forces and virial tensors for complicated advance force fields, 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`.

I. Introduction

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, instead of being computed ab initioly on-the-fly. Therefore, the quality of the underlying force field limits the accuracy and the predictive power of the MM simulation.

To date, conventional force fields 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 intra-molecular 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 (VDW) 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 in 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, and many-body dispersion etc. 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 interests. Such parameterization process relies heavily on human intervention, 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, such as ForceBalance and OpenFF,[7–9] but the differentiation of macroscopic properties with respect to force field parameters is a nontrivial task. In these works, such differentiation is still 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,[10] DeepPotential,[11] PhysNet,[12] and EANN[13] etc. 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 the full advantage of the bonding topology information, and a more accurate treatment to the long-range interactions is also in need. Therefore, for molecular systems, ML methods need to be combined with physics-driven models,[14,15] 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.[16–18] 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 force field parameters or system geometry. Such 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 force field parameters, combined with modern gradient descent algorithms, enables automatic optimization workflows for force field 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 engine, including: TorchMD,[19] JAX-MD,[20] and SPONGE.[21] However, as discussed in a previous study,[22] 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 cost in backward propagation. On the other hand, a comprehensive support to more force field function forms and more types of object functions is a more urgent need. For most molecular force field developers, functions such as topological scaling (e.g., the 1-3 scaling), flexible atom typification, long-range Ewald, and polarization, etc. are critical, but are still missing in the aforementioned platforms. Currently, the only differentiable framework dedicated to force field development is JAX-ReaxFF,[23] which is however specific to ReaxFF. So far, there is still no differentiable platform designed for molecular force field 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 force fields and advanced multipolar polarizable models. It features a user-friendly OpenMM-like interface for convenient molecular force field 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 force field models, DMFF also provides convenient differentiable estimators for not only energies and forces, but also thermal dynamic quantities including free energies and 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.[24] 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 compare 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 in Github.[25] Here we will introduce the overall structure of DMFF from four aspects: model preparation, model implementation, model optimization, and model deployment.

A. 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 force field 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.[26] By doing so, we can share the well-established open-source OpenMM ecosystem for the convenient definition of molecular force field. All parameters can be defined in the format of OpenMM force field XML files and a core class named "Hamiltonian" is devised to overload the "ForceField" class in OpenMM. Within the Hamiltonian object, the force field parameters are read and stored in a highly organized and hierarchical data structure. A demo for the preparation of a DMFF force field function and parameters can be found in Listing 1.

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

```python
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 force field parameters based on the molecular topology, 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.

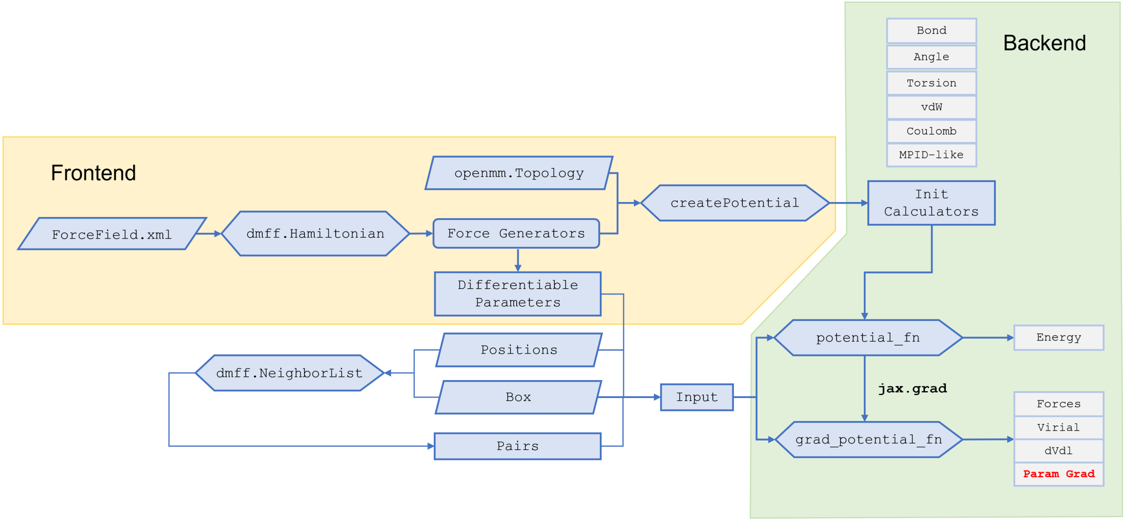

Figure 1: The overall code structure of the DMFF force field frontend and backend.

B. Model Implementation with 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 force field 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 DMFF backend provides a complete implementation of conventional molecular force fields, and a multipolar polarizable potential module (named Auto Differentiable Multipolar Polarizable module, or ADMP) that replicates the MPID[17] behavior up to quadrupole moments. For long-range interactions, Particle Meshed Ewald (PME)[27] 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.

C. Model Optimization

The biggest advantage of the differentiable implementation of molecular force field is that now we can employ gradient descent algorithms to optimize force fields 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[28] or jaxopt.[29]

The most trivial estimators that come naturally with the DMFF force field 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; and 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 etc. 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, Thaler and Zavadlav[22] 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[30] 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 briefly introduce the basic formulation of the reweighting MBAR estimator in DMFF.

In the MBAR theory, it is assumed that there are K ensembles defined by effective energies $u_i(x), (i = 1, 2, \ldots, K)$. The Boltzmann weight, the partition function, and the probability in each ensemble are defined as:

$$\begin{cases} w_i(x) = \exp\left(-\beta \cdot u_i(x)\right) \\ c_i = \int dx \cdot w_i(x) \\ p_i(x) = \frac{w_i(x)}{c_i} \end{cases} \tag{1}$$

From each ensemble i, we draw N_i samples, labeled as $\{x_{in}\}, (n = 1, 2, \ldots, N_i)$, and the whole sample set can be labeled as $\{x_n\}, n = (1, 2, \ldots, N)$, 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:[30]

$$\hat{c}_i = \sum_{n=1}^{N} \frac{w_i(x_n)}{\sum_{k=1}^{K} N_k \hat{c}_k^{-1} w_k(x_n)} \tag{2}$$

Note that Eqn.2 regulates the ratio between $\hat{c}_i$ and $\hat{c}_j$, so the free energy difference $\Delta f_{ij} = -\ln(c_j/c_i)$ can be solved.

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:

$$\begin{cases} w_j(x) = A(x)w_i(x) \\ c_j = \int dx \cdot A(x)w_i(x) \end{cases} \tag{3}$$

Then Eqn.2 can be used to solve the ratio between c_j and c_i, which is the esnemble average of A:

$$\frac{\hat{c}_j}{\hat{c}_i} = \frac{\int dx \cdot A(x)w_i(x)}{\int dx \cdot w_i(x)} = \langle A \rangle_i \tag{4}$$

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, Eqn.2 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 $m = 1, 2, \ldots, M$), 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 are always zero. Consequently, in Eqn.2, 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:

$$\hat{c_p} = \sum_{n=1}^{N} \frac{w_p(x_n)}{\sum_{m=1}^{M} N_m \hat{c}_m^{-1} w_m(x_n)} \tag{5}$$

In practice, each time when resample happens, $\hat{c}_m$ are updated by solving Eqn.2 iteratively, and are stored as constants within the estimator until next resampling. Then during the parameter optimization, $\hat{c}_p$, or their ratio $\hat{c}_p/\hat{c}_q$, is computed using Eqn.5, and returned as a differentiable 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:

$$\langle A \rangle_p = \frac{\hat{c}_q}{\hat{c}_p} = \frac{\sum_{n=1}^{N} A(x_n) \exp(-\beta \Delta u_{p0}(x_n))}{\sum_{n=1}^{N} \exp(-\beta \Delta u_{p0}(x_n))} \tag{6}$$

With $\Delta u_{p0}(x) = u_p(x) - u_0(x)$. This result recovers the Eqn.3 and 4 in reference [22], 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. (5). While for averaged properties, we reformulate Eq. (5) as:

$$\begin{aligned}\langle A \rangle_p &= \sum_{n=1}^{N} \frac{\left[\sum_{m=1}^{M} N_m \exp\left(\hat{f}_m - \beta \Delta U_{mp}(x_n)\right)\right]^{-1}}{\sum_{n=1}^{N} \left[\sum_{m=1}^{M} N_m \exp\left(\hat{f}_m - \beta \Delta U_{mp}(x_n)\right)\right]^{-1}} A(x_n) \\ &= \sum_{n=1}^{N} W_n A(x_n)\end{aligned} \tag{7}$$

with:

$$\begin{cases} \Delta U_{mp}(x_n) = U_m(x_n) - U_p(x_n) \\ W_n = \frac{\left[\sum_{m=1}^{M} N_m \exp\left(\hat{f}_m - \beta\Delta U_{mp}(x_n)\right)\right]^{-1}}{\sum_{n=1}^{N}\left[\sum_{m=1}^{M} N_m \exp\left(\hat{f}_m - \beta\Delta U_{mp}(x_n)\right)\right]^{-1}} \end{cases} \tag{8}$$

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 the Kish's effective sample size, which could be used to determine when a dataset is redundant thus can be removed, and when new samples are needed:

$$n_{\mathrm{eff}} = \frac{\left(\sum_{n=1}^{N} W_n\right)^2}{\sum_{n=1}^{N} W_n^2} \tag{9}$$

The gradient can be obtained by differentiating the estimator, and be fed to an external optimizer (e.g., optax[28] or jaxopt[29]) to update the force field parameters. A typical automatic workflow for a thermodynamic fitting using the reweighting MBAR estimator is given in Figure 2.

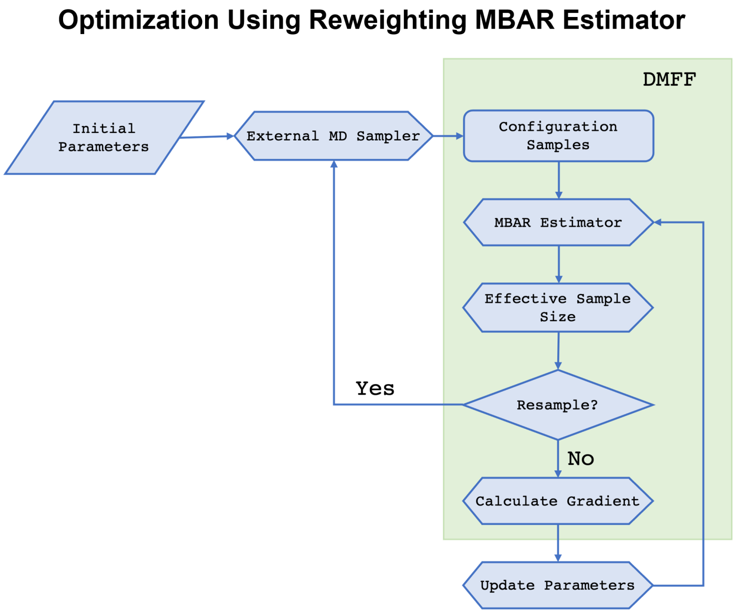

Figure 2: The workflow of thermodynamic properties optimization using reweighting MBAR estimator.

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 or dynamic properties (e.g., diffusion coefficients, electrical conductivities, and spectroscopy etc.) are also under development. Nevertheless, the current capability of DMFF already makes it a powerful tool for force field parameterization, as we will demonstrate in the results section.

D. Model Deployment

Once the force field has been optimized using DMFF, it can be deployed in MD simulations. On one hand, for conventional force field, DMFF is not as efficient as the highly optimized C++ code in the mainstream MD programs like LAMMPS[31] or OpenMM. In the current stage, DMFF does not pursue extreme performance in large-scale simulations, but rather focuses on fast force field optimization. Therefore, for conventional force fields, 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 readily 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[32] interface for DMFF, such that MD simulations can be run directly using the DMFF force engine. In future, interfaces with other more efficient MD programs (such as LAMMPS) is also to be developed.

3. Results and Discussions

As discussed in the introduction section, the automatic differentiation technique can be beneficial to both fast implementation of force/virial tensors, as well as force field 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 appendix.

A. Application in Advanced MD Simulation

In modern force field development, one often has to design 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 polarizable force field to compute the electrostatic energies.[15]

2. Similarly, one can use ML models to predict electronegativities and hardness, then use them in charge equilibration models.[33] 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 process is extremely tedious and requires highly skillful developers, 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 force field development. Meanwhile, it is noticed that some of the computational kernels (such as PME, pairwise interactions, minimization with respect to charges and dipoles, etc.) 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 force field development is not feasible due to the needs of force and virial tensor calculations: differentiating through different modules can be cubersome. The automatic differentiable DMFF package is designed to solve this problem. All commonly used molecular force field 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[34] is implemented. This water model is based on a multipolar polarizable potential, which is quite similar to AMOEBA and MPID.[34] 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 essential 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 the DMFF package. We can then interface it with MD packages such as i-PI[35] 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.

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

B. Bottom-Up Fitting

In the bottom-up force field development, we need to fit the energies or the forces to ab initio results. The object functions are typically the root mean square error (RMSE) or weighted RMSE of energies and forces, which can be directly computed using the DMFF force field calculator. The JAX implementation of the molecular force fields enables us to utilize the state-of-the-art optimizers developed in the ML community, designed to perform

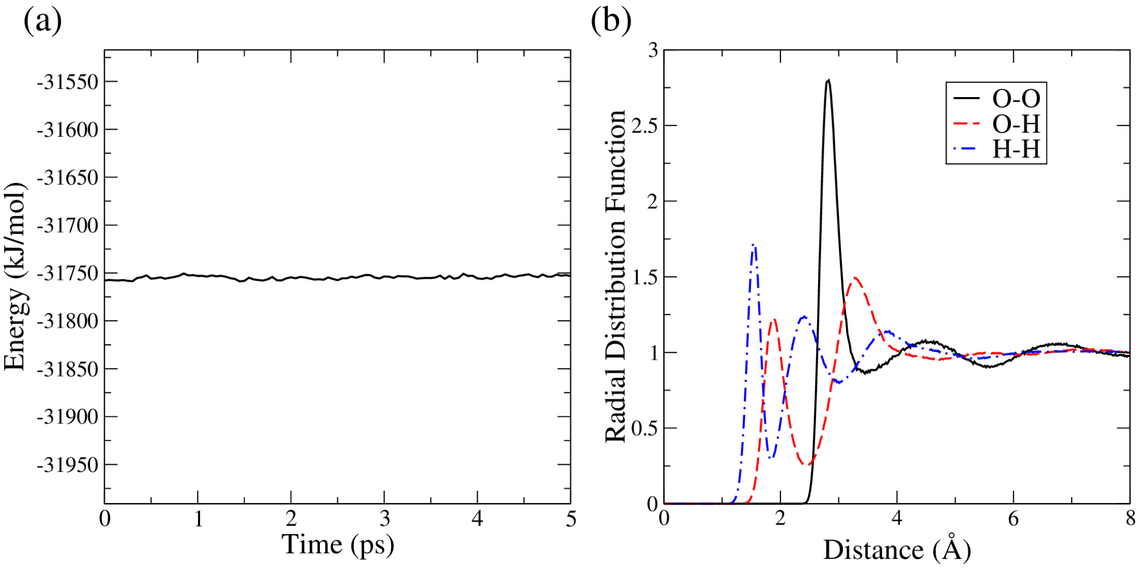

Figure 3: Simulation results of polarizable water force field with fluctuating charges and dispersion coefficients: (a) the energy conservation in NVE simulation, the range of the y-axis of the figure is set to be the thermal fluctuation of total energy ($\pm\sqrt{kT^2C_V}$); (b) the radial distribution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

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.[36] 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)[37] method, and the short-range parts of them are then fitted using the following functions:

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

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 target function is a weighted average of the RMSEs of all components and the total energy:

$$\begin{aligned}
L = \lambda_{tot} \sum \left(V_{tot} - V_{tot}^{ref}\right)^2 + \lambda_{ex} \sum \left(V_{ex} - V_{ex}^{ref}\right)^2 + \lambda_{es} \sum \left(V_{es}^{sr} - V_{sr,es}^{ref}\right)^2 \\
+ \lambda_{ind} \sum \left(V_{ind}^{sr} - V_{sr,ind}^{ref}\right)^2 + \lambda_{dhf} \sum \left(V_{dhf} - V_{dhf}^{ref}\right)^2 + \lambda_{disp} \sum \left(V_{disp} - V_{sr,disp}^{ref}\right)^2
\end{aligned} \tag{11}$$

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 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).

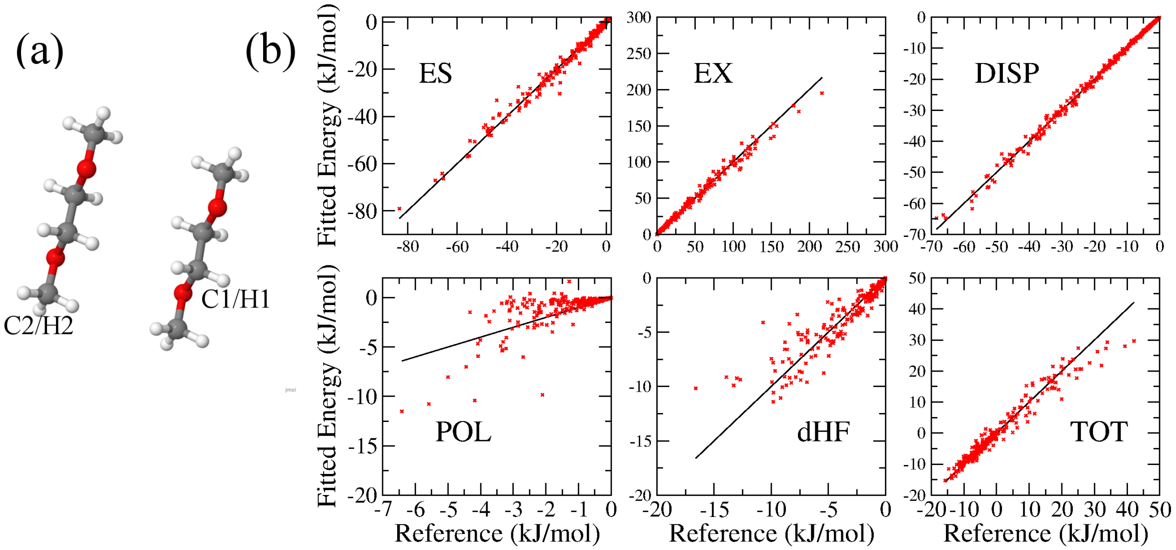

Figure 4: Bottom-up fitting result of COCCOC dimer: (a) the dimer structure and atom type definition; (b) the fitting results of each component and the total energy.

In this example, we show how ML optimizers can be conveniently utilized in developing complicated physics-driven force fields, using DMFF. This is a relatively straightforward way to take the advantage of an automatic differentiable force field 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.

C. Ensemble Averages

In classical force field parameterization, we always need to make the force field 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 Lennard-Jones interaction of the liquid tetrachloromethane and benzene 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), respectively. 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 tried to parameterize Lennard-Jones parameters of liquid tetrachloromethane to reproduce the density at 300K. The optimization target function is the squared error of density:

$$L_{\text{density}} = \left(\sum_{n=1}^{N} W_n \rho_n - \rho_{\text{ref}} \right)^2 \tag{12}$$

On the other hand, vector properties can also be estimated using the reweighting MBAR estimator. As an example, We built 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

$$L_{\text{RDF}} = \log \| \sum_{n=1}^{N} W_n \boldsymbol{v}_n - \boldsymbol{v}_{\text{ref}} \| \tag{13}$$

in which $\boldsymbol{v}_n$ is the vector of RDF curve in each sample frame, $\boldsymbol{v}_{\text{ref}}$ is the reference RDF curve.

The changes of density and RDF along the optimization process are shown in Figure (5a)

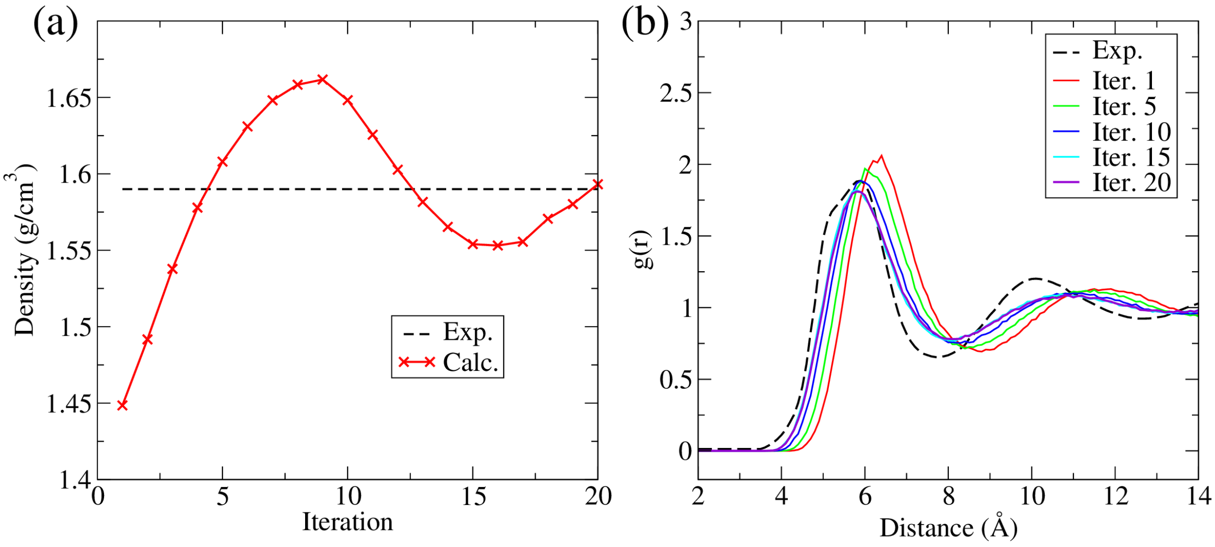

Figure 5: The change of (a) density of liquid tetrachloromethane and (b) radial distribution function of liquid benzene along optimization iteration.

and (5b), in which both density and RDF are closer to the target after about 20 iterations. In classical force field 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 process can be largely automated. Moreover, it can be done in a larger scale, with more parameters and more training data from multiple systems, thus improving the generality of the resulting force field.

D. Free Energies

Comparing to ensemble averaged properties such as density and RDF, fitting free energy difference is an important, yet even more challenging task. For example, the MD free energy calculation is one of the most common techniques used to predict protein-ligand binding affinities, which is the key goal in computer-aided drug design (CADD). Comparing to other methods that rely on empirical scoring functions (e.g., molecular docking), free-energy based techniques such as free energy perturbation (FEP) is physically rigorous and can potentially reach the chemical accuracy (i.e., within 1 kcal/mol of error).[38,39] However, it posts a great challenge to develop a general, yet accurate force field for FEP that can cover the diversified chemical space of drug-like molecules. An accurate description to the electrostatic interactions is particularly difficult. Due to the heavy computational cost of FEP, such calculations are still primarily conducted using conventional fixed charge models, including OPLS,[40–42] GAFF,[3] CGenFF,[43] and OpenFF.[44] In these models polarization effects are not explicitly included, so atomic charges need to be carefully tuned to give a balanced description to the electrostatics.

Currently, AM1/CM1A-BCC models are widely employed to assign atomic partial charges and were reported to reach fair results on FEP benchmark sets.[45,46] In this model, atomic charges are firstly determined using the Mulliken charges computed at semi-empirical level of theory (e.g., AM1 or CM1A). Then, they are further adjusted using bond charge correction (BCC), which is a correction only depends on the molecular topology. The BCC parameters are initially fitted to electrostatic potentials (ESP), and then refined to reproduce the experimental solvation free energies of some model molecules. Protein-ligand binding free energies, however, to the best of our knowledge, have not been used as a direct fitting target. The protein-ligand interaction involves many protein residues and ligand functional groups, thus are affected by a large number of parameters. Manually fine-tunning all these parameters simultaneously could be a difficult task, even for experienced experts with excellent intuitions. But with DMFF, this process now can be conducted automatically using gradient-based optimization algorithms.

To fit the FEP results, we note the Helmholtz free energy A can be calculated as:

$$A = -\beta^{-1} \ln Q = -\beta^{-1} \ln \int dx \; e^{-\beta U(x;\theta)} \tag{14}$$

where $\beta = 1/k_B T$, k_B is the Boltzmann factor, x is particles' coordinates and momenta, U is the energy function defined by the force field, and θ refers to force field parameters. The parametric gradient of the free energy can be evaluated as the ensemble average of the parametric gradient of the potential energy, which is given by DMFF.

$$\frac{\partial A}{\partial \theta} = -\beta^{-1}\left(\frac{1}{Q}\frac{\partial Q}{\partial \theta}\right) \tag{15}$$

$$= -\beta^{-1}\left(\frac{1}{Q}\int d\mathbf{q}\cdot\frac{\partial U}{\partial \theta}\cdot e^{-\beta U(\mathbf{q};\theta)}\right) \tag{16}$$

$$= \left\langle \frac{\partial U}{\partial \theta} \right\rangle \tag{17}$$

To demonstrate the capability of DMFF, six relative free energies ($\Delta\Delta G$) are selected from TNKS2, which is one of the most popular protein-ligand binding affinity benchmark sets.[39] As a proof of concept, we try to optimize the GAFF2 BCC parameters against the selected data points. The loss function is defined in Eqn. 18 and the steepest gradient descent algorithm is utilized for the optimization.

$$L = \frac{1}{2M}\sum_{i=1}^{M}(\Delta\Delta G_{\mathrm{exp}} - \Delta\Delta G_{\mathrm{pred}})^2 \tag{18}$$

After only 5 iterations, the overall R^2 has increased from 0.53 to 0.87 and the root of mean squared error (RMSE) dropped from 1.26 kcal/mol to 0.73 kcal/mol (as shown in Figure 6 and Table 1). This performance boost can be attributed to the significant improvement of outliers, such as 5a→5c. This result clearly demonstrates the advantage brought by DMFF, essentially allowing us to further improve the state-of-the-art model without any chemical intuition.

5. Conclusions

To summarize, in this work 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 force fields. It also includes a variety of efficient backend kernels that support both conventional point charge models and

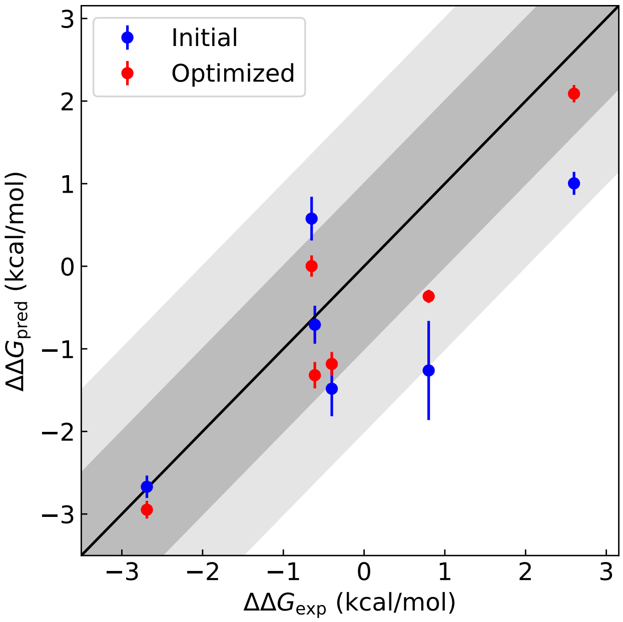

Figure 6: Correlation between the experimental and the FEP-predicted relative binding free energies with initial BCC parameters (blue) and optimized BCC parameters (red). The light and the dark gray areas mark the regions with an error smaller than 2 kcal/mol and 1 kcal/mol, respectively.

advanced multipolar polarizable potentials. The potential functions defined in DMFF are easy to be extended and recombined to form new advanced force fields. The implementation of forces and virial tensors can be largely simplified, so the MD validation of complex force fields can be performed with much less efforts. Based on the differentiable force field engine, a variety of object functions can be used for force field 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 force field optimizations can be performed in an automatic fashion. Parameters can be obtained with better performances in both bulk organic liquid simulations and FEP calculations. Therefore, DMFF serves as an excellent tool for the development of next-generation molecular force fields.

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, interface with more MD engines, are to be developed. All activities

Table 1: Experimental and FEP-predicted relative binding free energies with initial BCC parameters and optimized BCC parameters for studied TYK2 cases. All units are in kcal/-mol.

Ligand	$\Delta\Delta G_{\mathrm{exp}}$	$\Delta\Delta G_{\mathrm{pred}}$ (Initial BCC)	$\Delta\Delta G_{\mathrm{pred}}$ (Optimized BCC)	Improved
5a→5c	0.80	-1.26	-0.36	Yes
5e→5a	-0.65	0.58	0.00	Yes
3a→7	2.60	1.00	2.09	Yes
5e→5p	-0.40	-1.48	-1.18	Yes
5e→5n	-0.61	-0.71	-1.32	No
7→5j	-2.69	-2.67	-2.95	No
	RMSE	1.26	**0.73**	Yes
	$\mathbf{R^2}$	0.53	**0.87**	Yes

will be conducted in the DeepModeling community (`https://github.com/deepmodeling/`). In addition, any issues, pull requests, suggestions, and potential collaborations are welcome.

Acknowledgement

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

References

(1) de Pablo, J. J. et al. New frontiers for the materials genome initiative. *npj Computational Materials* **2019**, *5*, 1–23, Number: 1 Publisher: Nature Publishing Group.

(2) Surabhi, S.; Singh, B. K. COMPUTER AIDED DRUG DESIGN: AN OVERVIEW. *Journal of Drug Delivery and Therapeutics* **2018**, *8*, 504–509, Number: 5.

(3) Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. Development and testing of a general amber force field. *Journal of Computational Chemistry* **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. *Journal of the American Chemical Society* **1992**, *114*, 10024–10035, Publisher: American Chemical Society.

(5) Vanommeslaeghe, K.; MacKerell, A. D. CHARMM additive and polarizable force fields for biophysics and computer-aided drug design. *Biochimica et biophysica 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. *Journal of the American Chemical Society* **1996**, *118*, 11225–11236.

(7) Wang, L.-P.; Chen, J.; Van Voorhis, T. Systematic Parametrization of Polarizable Force Fields from Quantum Chemistry Data. *Journal of Chemical Theory and Computation* **2013**, *9*, 452–460, Publisher: American Chemical Society.

(8) Wang, L.-P.; Martinez, T. J.; Pande, V. S. Building Force Fields: An Automatic, Systematic, and Reproducible Approach. *The Journal of Physical Chemistry Letters* **2014**, *5*, 1885–1891, Publisher: American Chemical Society.

(9) 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. *Journal of Chemical Theory and Computation* **2022**, *18*, 3566–3576, Publisher: American Chemical Society.

(10) Behler, J.; Parrinello, M. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Physical Review Letters* **2007**, *98*, 146401.

(11) Zhang, L.; Han, J.; Wang, H.; Car, R.; E, W. Deep Potential Molecular Dynamics: a scalable model with the accuracy of quantum mechanics. *Physical Review Letters* **2018**, *120*, 143001, arXiv: 1707.09571.

(12) Unke, O. T.; Meuwly, M. PhysNet: A Neural Network for Predicting Energies, Forces, Dipole Moments, and Partial Charges. *Journal of Chemical Theory and Computation* **2019**, *15*, 3678–3693, Publisher: American Chemical Society.

(13) Zhang, Y.; Hu, C.; Jiang, B. Embedded Atom Neural Network Potentials: Efficient and Accurate Machine Learning with a Physically Inspired Representation. *The Journal of Physical Chemistry Letters* **2019**, *10*, 4962–4967.

(14) 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. *The Journal of Physical Chemistry Letters* **2021**, *12*, 7982–7987, Publisher: American Chemical Society.

(15) 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. *Journal of Chemical Theory and Computation* **2022**, *18*, 1711–1725, Publisher: American Chemical Society.

(16) Shi, Y.; Xia, Z.; Zhang, J.; Best, R.; Wu, C.; Ponder, J. W.; Ren, P. Polarizable Atomic Multipole-Based AMOEBA Force Field for Proteins. *Journal of Chemical Theory and Computation* **2013**, *9*, 4046–4063, Publisher: American Chemical Society.

(17) 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. *The Journal of Chemical Physics* **2017**, *147*, 161702, Publisher: American Institute of Physics.

(18) 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. *Journal of Chemical Theory and Computation* **2019**, *15*, 5001–5013, Publisher: American Chemical Society.

(19) 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. *Journal of Chemical Theory and Computation* **2021**, *17*, 2355–2363, Publisher: American Chemical Society.

(20) Schoenholz, S.; Cubuk, E. D. JAX MD: A Framework for Differentiable Physics. Advances in Neural Information Processing Systems. 2020; pp 11428–11441.

(21) 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. *Chinese Journal of Chemistry* **2022**, *40*, 160–168, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cjoc.202100456.

(22) Thaler, S.; Zavadlav, J. Learning neural network potentials from experimental data via Differentiable Trajectory Reweighting. *Nature Communications* **2021**, *12*, 6884, Number: 1 Publisher: Nature Publishing Group.

(23) 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. *Journal of Chemical Theory and Computation* **2022**, Publisher: American Chemical Society.

(24) 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. 2018; `http://github.com/google/jax`.

(25) DMFF Project. `https://github.com/deepmodeling/DMFF`.

(26) 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 Computational Biology* **2017**, *13*, e1005659, Publisher: Public Library of Science.

(27) Essmann, U.; Perera, L.; Berkowitz, M. L.; Darden, T.; Lee, H.; Pedersen, L. G. A smooth particle mesh Ewald method. *The Journal of Chemical Physics* **1995**, *103*, 8577–8593, Publisher: American Institute of Physics.

(28) Babuschkin, I. et al. The DeepMind JAX Ecosystem. 2020; `http://github.com/deepmind`.

(29) Blondel, M.; Berthet, Q.; Cuturi, M.; Frostig, R.; Hoyer, S.; Llinares-López, F.; Pedregosa, F.; Vert, J.-P. Efficient and Modular Implicit Differentiation. 2022; `http://arxiv.org/abs/2105.15183`, arXiv:2105.15183 [cs, math, stat].

(30) Shirts, M. R.; Chodera, J. D. Statistically optimal analysis of samples from multiple equilibrium states. *The Journal of Chemical Physics* **2008**, *129*, 124105, Publisher: American Institute of Physics.

(31) Thompson, A. P.; Aktulga, H. M.; Berger, R.; Bolintineanu, D. S.; Brown, W. M.; Crozier, P. S.; Veld, P. J. i. t.; 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. *Comp. Phys. Comm.* **2022**, *271*, 108171.

(32) Wang, X. I-Pi-Driver, Github Repository. 2020; `https://github.com/WangXinyan940/i-pi-driver`.

(33) 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. *The Journal of Physical Chemistry B* **2022**, *126*, 6271–6280, Publisher: American Chemical Society.

(34) 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. **2022**,

(35) 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.

(36) 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. *Journal of Chemical Theory and Computation* **2016**, *12*, 3851–3870.

(37) Williams, H. L.; Chabalowski, C. F. Using KohnSham Orbitals in Symmetry-Adapted Perturbation Theory to Investigate Intermolecular Interactions. *The Journal of Physical Chemistry A* **2001**, *105*, 646–659.

(38) Wang, L.; Wu, Y.; Deng, Y.; Kim, B.; Pierce, L.; Krilov, G.; Lupyan, D.; Robinson, S.; Dahlgren, M. K.; Greenwood, J., et al. Accurate and reliable prediction of relative ligand binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. *Journal of the American Chemical Society* **2015**, *137*, 2695–2703.

(39) Schindler, C. E.; Baumann, H.; Blum, A.; Bose, D.; Buchstaller, H.-P.; Burgdorf, L.; Cappel, D.; Chekler, E.; Czodrowski, P.; Dorsch, D., et al. Large-scale assessment of binding free energy calculations in active drug discovery projects. *Journal of Chemical Information and Modeling* **2020**, *60*, 5457–5474.

(40) Harder, E.; Damm, W.; Maple, J.; Wu, C.; Reboul, M.; Xiang, J. Y.; Wang, L.; Lupyan, D.; Dahlgren, M. K.; Knight, J. L., et al. OPLS3: a force field providing

broad coverage of drug-like small molecules and proteins. *Journal of Chemical Theory and Computation* **2016**, *12*, 281–296.

(41) Roos, K.; Wu, C.; Damm, W.; Reboul, M.; Stevenson, J. M.; Lu, C.; Dahlgren, M. K.; Mondal, S.; Chen, W.; Wang, L., et al. OPLS3e: Extending force field coverage for drug-like small molecules. *Journal of Chemical Theory and Computation* **2019**, *15*, 1863–1874.

(42) Lu, C.; Wu, C.; Ghoreishi, D.; Chen, W.; Wang, L.; Damm, W.; Ross, G. A.; Dahlgren, M. K.; Russell, E.; Von Bargen, C. D., et al. OPLS4: Improving force field accuracy on challenging regimes of chemical space. *Journal of Chemical Theory and Computation* **2021**, *17*, 4291–4300.

(43) Vanommeslaeghe, K.; Hatcher, E.; Acharya, C.; Kundu, S.; Zhong, S.; Shim, J.; Darian, E.; Guvench, O.; Lopes, P.; Vorobyov, I., et al. CHARMM general force field: A force field for drug-like molecules compatible with the CHARMM all-atom additive biological force fields. *Journal of Computational Chemistry* **2010**, *31*, 671–690.

(44) Qiu, Y.; Smith, D. G.; Boothroyd, S.; Jang, H.; Hahn, D. F.; Wagner, J.; Bannan, C. C.; Gokey, T.; Lim, V. T.; Stern, C. D., et al. Development and Benchmarking of Open Force Field v1. 0.0—the Parsley Small-Molecule Force Field. *Journal of Chemical Theory and Computation* **2021**, *17*, 6262–6280.

(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. *Journal of Computational Chemistry* **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. *Journal of Computational Chemistry* **2002**, *23*, 1623–1641.

(47) Kapil, V. et al. i-PI 2.0: A universal force engine for advanced molecular simulations. *Computer Physics Communications* **2019**, *236*, 214–223.

(48) Werner, H.-J. et al. *MOLPRO, version 2019.2, a package of ab initio programs*; 2019.

(49) Beauchamp, K. A.; Chodera, J. D.; Naden, L. N.; Shirts, M. R. The PyMBAR program. 2013; `https://github.com/choderalab/pymbar`.

Appendix

A. Details in Water MD Simulation

All water simulations were carried out using a cubic box consisting of 1018 H_2O molecules, with the density set to be 0.03338 molecule per $Å^3$. Each atom was represented by a single bead in classical MD and by 32 beads in path-integral molecular dynamics (PIMD) simulations. The equations of motion were propagated using a multiple time step integration scheme with a time step of 0.5 fs. All PIMD simulations were carried out in NVT ensemble and the temperature was controlled by path-integral langevin thermostat. The classical MD simulation were carried out in NVE ensemble. All MD simulations were performed using the i-PI 2.0 program,[47] with forces evaluated using DMFF.

B. Details in COCCOC Energy Fitting

In the COCCOC fitting example, dimer structures were first generated using a NVT MD simulation at 300K. The MD sampling was conducted using the OpenMM program,[26] with OPLS-AA force field.[6] A constrain potential was applied to bind the centers of mass of the two molecules within a distance of 6 Å. For each dimer, one of the monomers are then shifted along the direction that connects the centers of mass of the two monomers, generating a dimer scan. DFT-SAPT[37] calculations were then performed using AVTZ basis set to obtain the interaction energy between the two monomers, as well as all the energy components. An even tempered basis set (5s5p3d2f) was placed at the center of the two monomers, to improve the accuracy of the dispersion energy. All SAPT calculations were performed using the MOLPRO program.[48] The dimer interaction energies were then partitioned into the short and the long range parts, with the long-range part computed using the exactly same method as in our previous work.[14] The short range part is then fitted using DMFF. In total, 50 dimer scans were generated for fitting, with each scan containing 12 different geometries.

C. Details in Ensemble Average Fitting

In the physical properties fitting example, the force field parameters were initialized using intra-molecular parameters from GAFF force field. The MD sampling was conducted using the OpenMM program with 1.1 nm as the real space cutoff. In the sampling process, a 400 ps trajectory was sampled with the first 200 ps as equilibration and the last 200 ps as production. The MBAR free energies in DMFF MBAR estimator was solved using PyMBAR program.[49] The optimization of physical properties were implemented using optax package.[28]

D. AM1-BCC model

In practice of GAFF and OPLS force fields, methods that based on bond charge corrections have been employed to assign high-quality atomic charges quickly and efficiently. In this scheme, the partial charges vector $\mathbf{q} \in \mathbb{R}^N$ (N is the number of atoms consisting a molecule) is constructed from two terms:

$$\mathbf{q} = \mathbf{q}^{\text{pre}} + \mathbf{q}^{\text{corr}} \tag{19}$$

The first term, $\mathbf{q}^{\text{pre}}$, is Mulliken charges under semi-empirical level. In GAFF2, Austin Model 1 (AM1) is used, and CM1A model in OPLS. It is expected that primary properties of electron distribution including delocalization and conformer-dependency can be captured by these non-expensive calculations. However, the resulting charges are not suitable for applications in condensed-phase environment, so a bond charge correction (BCC) term $\mathbf{q}^{\text{corr}}$, defined based on molecular topology (equation 20), is applied:

$$\mathbf{q}^{\text{corr}} = \mathbf{T}\mathbf{p} \tag{20}$$

where $T \in \mathbb{R}^{N \times N_{\text{bcc}}}$ is named topology matrix (or bond connectivity template matrix), and $\mathbf{p} \in \mathbb{R}^{N_{\text{bcc}}}$ is a vector of pre-tabulated BCC parameters, N_{bcc} is the number of bond-type based parameters. Based on equation 20, we can extend the back propagation from atomic charges to BCC parameters:

$$\frac{\partial \Delta G}{\partial \mathbf{p}} = \mathbf{T}^{\mathrm{T}} \cdot \frac{\partial \Delta G}{\partial \mathbf{q}} \tag{21}$$

E. Details in free energy optimization

FEP calculations with both initial and optimized BCC parameters were set up and carried out using the FEP module of Hermite® Platform (https://hermite.dp.tech, DP Technology). The initial structures of the protein and ligands were obtained from the original paper.[39] AMBER99SB force field was used to paramterize the protein. 16 lambdas windows were set and the value of lambdas were adapted on-the-fly. After energy minimization, a standard 5-stage equilibrium protocol was carried out to relax the system with all parameters set to default. Then, 5 ns production simulation was performed and Langevin thermostat and Parrinello-Rahman barostat were adopted to keep the system at 300 K and 1 bar and other parameters were set to default. In the optimization process, steep gradient descent algorithm was used and the learning rate was set to 0.0005.