The Open Force Field Evaluator: An automated, efficient, and scalable framework for the estimation of physical properties from molecular simulation

Simon Boothroyd[1,2,5], **Lee-Ping Wang**[3], **David L. Mobley**[4], **John D. Chodera**[1], **Michael R. Shirts**[2]

[1]Computational and Systems Biology Program, Sloan Kettering Institute, Memorial Sloan Kettering Cancer Center, New York, NY 10065; [2]Department of Chemical & Biological Engineering, University of Colorado Boulder, Boulder, CO, USA 80309; [3]Department of Chemistry, The University of California at Davis, Davis, California 95616; [4]Departments of Pharmaceutical Sciences and Chemistry, The University of California at Irvine, Irvine, CA, USA 92617; [5]Present address - Boothroyd Scientific Consulting Ltd, 71-75 Shelton Street, London, Greater London, United Kingdom, WC2H 9JQ

***For correspondence:**
michael.shirts@colorado.edu (MRS)

Abstract

Developing accurate classical force field representations of molecules is key to realizing the full potential of molecular simulations, both as a powerful route to gaining fundamental insight into a broad spectrum of chemical and biological phenomena, and for predicting physicochemical and mechanical properties of substances. The Open Force Field Consortium is an industry-funded open science effort to this end, developing open source tools for rapidly generating new, high-quality small molecule force fields. An integral aspect of this is the parameterization and assessment of force fields against high-quality, condensed-phase physical property data, curated from open data sources such the NIST ThermoML Archive, alongside quantum chemical data. The quantity of such experimental data in open data archives alone would require an onerous amount of human and compute resources to both curate and estimate manually, especially when estimations must be made for numerous sets of force field parameters. Here we present an entirely automated, highly scalable framework for evaluating physical properties and their gradients in terms of force field parameters. It is written as a modular and extensible Python framework, which employs an intelligent multiscale estimation approach that allows for the automated estimation of properties from simulation and cached simulation data, and a pluggable API for estimation of new properties. In this study we demonstrate the utility of the framework by benchmarking the OpenFF 1.0.0 small molecule force field, GAFF 1.8 and GAFF 2.1 force fields against a test set of binary density and enthalpy of mixing measurements curated using the frameworks utilities. Further, we demonstrate the framework's utility as part of force field optimization by using it alongside ForceBalance, a framework for systematic force field optimization, to retrain a set of non-bonded van der Waals parameters against a training set of density and enthalpy of vaporization measurements.

1 Introduction

The development of accurate and transferable molecular force fields is a necessary step to achieving the full potential of molecular simulation [1–4]. Molecular simulation offers both a powerful route to gaining deep insight into a range of biological and chemical phenomena and as a tool for predicting the physicochemical and mechanical properties of substances.

While the bonded terms of a force field are often fit and assessed directly against quantum chemical data, the non-bonded terms are generally indirectly inferred by fitting against experimentally measured condensed phase physical property data [5–7]. While there is a substantial amount of experimentally measured physical property data available from open data sources (including the NIST ThermoML archive [8–12], the FreeSolv data set [13, 14], and BindingDB [15]) the data is often stored in a diverse range of file and storage formats which are not always documented, and in cases, not readily machine readable. Furthermore, the large amount of data, often containing many duplicate (or erroneously corrupted) data points [16], makes it prohibitively time consuming to manually curate training and test sets. Even once the training and test sets have been curated, the estimation of those sets using a given force field often requires a significant amount of human time to prepare the required input files and to perform analysis on the results, and requires significant compute time to perform the needed simulations for any estimated properties.

Here, we report on our OpenFF Evaluator framework, which was designed to overcome these issues. In particular, it is an automated, scalable, Python framework for the curation of physical property data sets from open data sources, and the estimation of properties of such data sets using a combination of molecule simulation and cached molecular simulation data.

Two core philosophies underlie the framework's design. The first is that the framework should be readily scalable for any required calculations from running on a single machine up to running across hundreds of high performance compute nodes, and potentially even into the cloud. Secondly, it is constructed so that every aspect is user extendable via a flexible plugin system. This includes everything from the extraction of properties of data sources into Python objects, up to defining the workflows for how physical properties should be estimated.

Here we describe the general architecture of the framework and its features, and demonstrate its ability to both assess the performance of three common small molecule force fields (OpenFF 1.0.0 [17], GAFF 1.8 [6] and GAFF 2.1 [18]) as well as train the non-bonded vdW parameters of the OpenFF 1.0.0 force field against data sets of physical property data curated using the framework's tools.

A more complete overview of the technical features of the framework, as well as installation instructions and getting started tutorials, can be found in the framework's documentation [19].

2 Framework Architecture

The framework's architecture complements the full workflow for force field development, from the curation of the testing and training sets from open data sources, evaluating the optimization objective function (and its gradient with respect to force field parameters) through integration with optimization frameworks such as ForceBalance [20–22], and the assessment of force fields against large data sets of even more complex physical properties including solvation free energies and host-guest binding affinities (Figure 1).

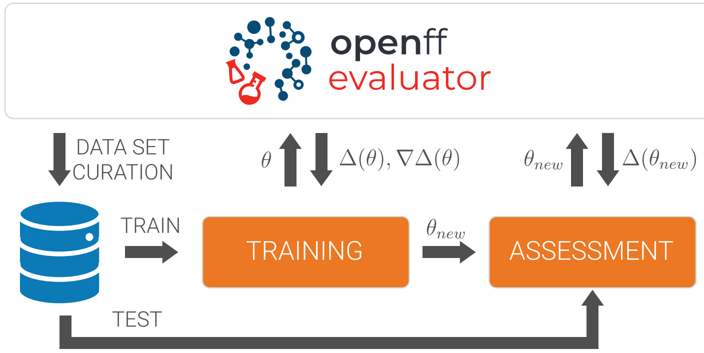

Figure 1. The Evaluator framework integrates into each step of optimising and assessing force fields against physical property data. The framework provides tools for extracting and curating training and test data sets from open data sets, can estimate the deviations of properties from the experimentally values ($\Delta(\theta)$) for a given set of force field parameters θ, as well as the gradient of those deviations with respect to the parameters $\nabla(\Delta(\theta))$ (i.e evaluate an optimization objective function and the gradient of the objective function).

In order to accommodate such a workflow, the framework was designed so as to:

- be able to **directly import data from different open data sources**, where the data from each data source may be in a different storage or file format, and store it in a common data object.
- **provide a unified set of utilities** for analysing, filtering, converting and curating training and test sets from imported data.
- be able to **apply force field parameters** from a wide range of different file formats and engines to benchmark the broad spectrum of commonly used force fields.
- **readily allow new properties to be defined** by users so that they may rapidly be used as both fitting and benchmarking targets.
- be able to **scale across available compute resources**, whether that be a local machine (e.g. via MPI), a compute cluster, or the cloud.
- **allow for different approaches for computing properties** (or sets of properties), such that users can take advantage of large amounts of cached simulation data to speed up their calculations.
- **be readily integrated into other software** requiring the estimation of properties.

The framework handles these demands by implementing a highly modular design, whereby each of these specific requirements are handled by independent modules which may readily be extended or replaced entirely with custom implementations (Figure 2).

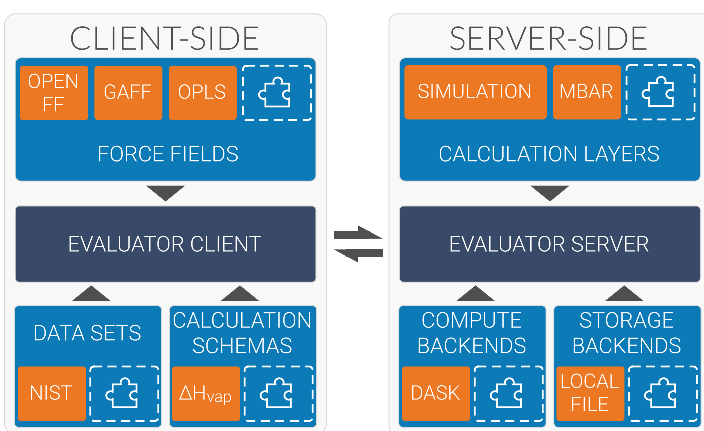

Figure 2. The framework is composed of modular components which may be extended or replaced by user defined plugins. The core functionality of the framework is entirely modularised into clearly abstracted components (blue) which can readily be swapped out with built-in implementations (shown in orange), or user-created plugins (represented by the dashed-box "puzzle pieces").

The framework is implemented as a client-server architecture. This design allows users to launch Evaluator server instances on whichever compute resources they may have available, from a single machine up to a large HPC cluster. Evaluator clients, run on modest hardware such as a user's laptop, may then connect a running sever to both request that a physical property data set be estimated, and to query and retrieve the results of those estimation requests.

The 'client' portion of the framework implements the logic for curating and sourcing the data sets, loading the force field parameters into uniform Python objects, and defining calculation schemas for how a class of physical property (e.g. mass densities or solvation free energies) should be estimated. Conversely, the 'server' side implements the logic required for scheduling and performing the calculations required to estimated a data set as requested by a client.

The server has three core componenents: calculation layers, storage backends, and compute backends. A "compute backend" is an abstraction around a library or framework which is able to distribute a set of tasks to perform, such as building the coordinates of a molecular system, across a number of available compute resources. These may be as simple as wrappers around Python's multi-processing libraries, or more complex such as the 'dask-jobqueue' library [23] which is able to distribute graphs of tasks across high performance compute (HPC) resources. A "storage backend" is another abstraction whose purpose is to both store cached simulation data (for example on a remote storage platform, or in a database structure) and also query and retrieve stored simulation data. The currently implemented local file backend stores all data on the local file disk. However, in the future, more sophisticated options, such as storing data within a SQL or NoSQL database or on a remote server, may be supported. Finally, calculation layers (as discussed in more detail in Section 2.2) are implementations of a particular approach for estimating a set of physical properties, such as via molecular simulation or evaluating a surrogate model which has been training on previously generated simulation data.

The 'server-client' model in particular allows the framework to be trivially integrated into other applications, as the user will mostly never need to consider how to schedule and run their calculations, but rather, use the API to submit and re-query the results of their request [19].

2.1 Curation of Experimental Data Sets

The framework has built-in support for constructing data sets for force field optimization and assessment via two main routes. Data sets may be manually transcribed by a user by directly creating the data set objects, typically requiring the user to enter common information about a property such as the state for which it was measured, the composition of the measured system, provenance information, and so forth. More usefully for large-scale projects, data may be automatically imported from certain sources. The framework currently supports importing data directly from the FreeSolv data set [14], and from the NIST ThermoML archive [12].

The NIST ThermoML archive in particular contains a wealth of experimental measurements for a diverse range of physical properties (Table 1). This diversity and range of data, combined with the framework's ability to seamlessly extract, curate, and then estimate those properties, makes the archive a valuable source of data for both training and assessing force fields.

Table 1. An estimate of the number of measurements that may be imported from the NIST ThermoML archive using the framework's built-in utilities as of 03/08/2021.

Property	Number of Measurements Points (in Thousands)		
	Pure	Binary	Ternary
Mass Density	176.6	364.9	119.4
Excess Molar Volume	-	11.7	3.1
Enthalpy of Mixing	-	32.9	4.9
Enthalpy of Vaporization	0.5	-	-
Vapor Pressure	44.6	75.4	10.2
Activity Coefficient	28.4	1.3	-
Osmotic Coefficient	-	2.0	0.6
Speed of Sound	21.5	55.0	15.4
Dielectric Constant	1.7	3.0	0.4
Liquid Gas Surface Tension	3.5	6.5	0.9

More than just offering utilities for importing experimental measurements, the framework offers a full suite of components aimed at making the curation of training and testing data sets as quick and painless as possible. In particular it contains components to filter out unwanted data points, ranging from filtering out data points that were measured outside of a particular temperature, to filtering by the characteristics of the substances the measurement was made for, such as only retaining measurements made for molecules containing alcohol or ester functionalities. Moreover, there are components available to:

- convert between property types where commensurate data is available, such as converting between excess molar volume data and density data when the densities of the pure components are available.
- select a fixed number of data points where were measured at states close to a target set of target states (e.g. selecting data points measured at close to ambient conditions).
- select data points measured for a diverse range of molecules which contain a target set of functionalities (e.g. data points measured for ketones, alcohols or alkanes).

A full list of the available curation components can be found in the framework's documentation [19].

2.2 Calculation Layers

A core aspect of the framework is its ability to employ a hierarchy of different approaches to compute a data set of physical properties, ranging from very rapid but less robust approaches such as evaluating surrogate models which have been trained on simulation data, to more robust approaches such as estimation by direct molecular simulation. Such a hierarchy enables the framework to automatically attempt to select the fastest approach which is able to estimate a given data set to within a user defined accuracy (Figure 3a).

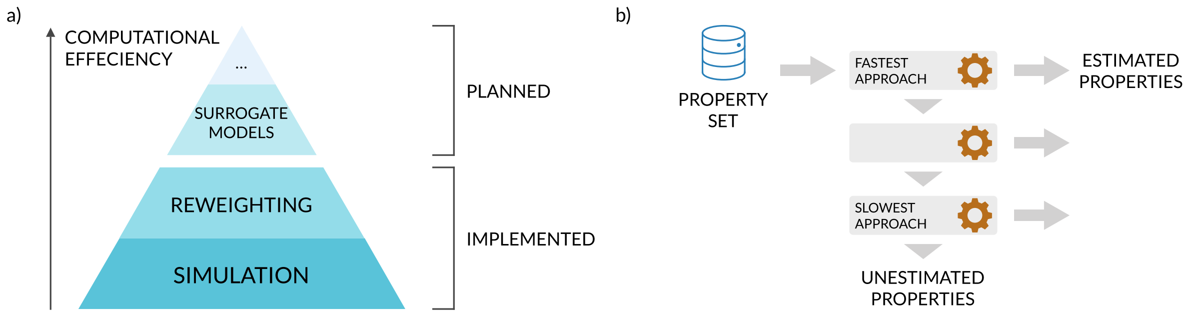

Figure 3. Automated selection of the fastest estimation approach optimisation can reduce computation effort. a) The framework employs a hierarchy of calculation approaches which currently includes estimation by direct simulation, and by reweighting cached simulation data. In the future, this may be extended to include both training of and estimation using surrogate models. b) Properties are cascaded through the calculation approaches, whereby those properties which could be estimated are returned, or those which couldn't be estimated with sufficient accuracy by this layer are moved to the next layer. This continues until either the full set of physical properties have been estimated using the specified force field parameters, or there are no more approaches left to attempt to estimate the set in which case the remaining properties are marked as unestimated and returned to the user.

In practice, each different calculation approach is implemented as a specific 'calculation layer'. Each layer acts as a black box that must take as input a set of physical properties to estimate and a calculation schema that controls how they should be estimated (e.g. how long simulations should be run for), and must return those properties which it was able to estimate as well as the uncertainty in those values. These calculations layers are then 'stacked' together, whereby the framework will first attempt to estimate the data set using the fastest layer at the top of the stack. Any properties which are estimated to within the specified uncertainty are then returned back to the user. Any properties which could not be estimated, for example, when an approach does not yet support estimated a particular type of property or the approach not being able to estimate a property to within the specified uncertainty, are then used as input for the next fastest layer. This process is then repeated until either all properties have been estimated, or there are no remaining calculations layers left to attempt (Figure 3b).

Currently the framework implements two calculation layers: a simulation layer which employs direct molecular simulations to estimate the property set, and a reweighting layer, which employ the Multistate Bennett Acceptance Ratio (MBAR) [24] technique to re-evaluate cached simulation data generated at one state, or using one particular set of force field parameters, to yield a property estimate at a new state or set of parameters [25].

The simulation layer is the 'fallback layer' which should always be able to estimate the data set of properties if the user has chosen to enable it. It reports the statistical uncertainty in the simulated properties, by default calculated by bootstrapping the sampled data to yield a estimated distribution of results. The layer is able to automatically extend all simulations until the uncertainty in the estimated properties has been reduced to within the set tolerance. A maximum simulation length is enforced to stop simulations from running indefinitely in the case of very noisy or extremely slow to converge properties.

The reweighting layer is in principle a much more rapid layer than the simulation layer, in that it does not need to run a new simulation to estimate the property, but rather it simply reprocesses existing decorrelated simulation data. The reweighting layer has two confidence metrics: the 'effective number of samples' and the uncertainty in the estimated properties. The effective number of samples describes the amount of information contained about the ensemble with new parameters that is contained in the original simulation. It must be above a user-defined threshold, with a default of 50, to be generally sufficient to generated accurate uncertainties in reweighted observables. [25]. The uncertainty in the estimated properties may also be requested to be below a user defined threshold. This uncertainty can either be an absolute threshold, or a threshold defined relative to each property in the data set's reported uncertainty.

2.3 Workflow Engine

To facilitate computing a diverse range of physical properties using a variety of different computation approaches, each of which may require performing distinct calculation steps, the framework facilitates the creation of lightweight property estimation workflows. The built-in workflow engine is for the most part a wrapper around more established workflow engines, delegating the actual execution and scheduling of the workflow to the external engine (currently Dask [26]). The built-in components focus instead on defining and exposing the possible set of workflow tasks (here referred to as protocols) and outlining how those tasks are coupled together through the construction of JSON serializable workflow schemas.

The framework implements many individual modular components of simulation workflows such as for building coordinates, for applying force fields parameters, performing bootstrapped analysis of simulation results, and even setting up and running full free energy simulations via Yank and OpenMM [27, 28]; we refer to these modular components as "protocols". These protocols can be chained together to form a larger workflow. Each individual protocol must define the set of inputs that they require as well as the outputs which they will produce. The protocols may then be chained together at a granular level, whereby individual outputs of a previous protocol may be used as inputs to protocols further along in the workflow, allowing diverse and complex workflows to be constructed from a limited set of simple protocol building blocks (Figure 4). A full list of protocols and guidance on combining them to form property estimation workflows is provided in the frameworks documentation [19].

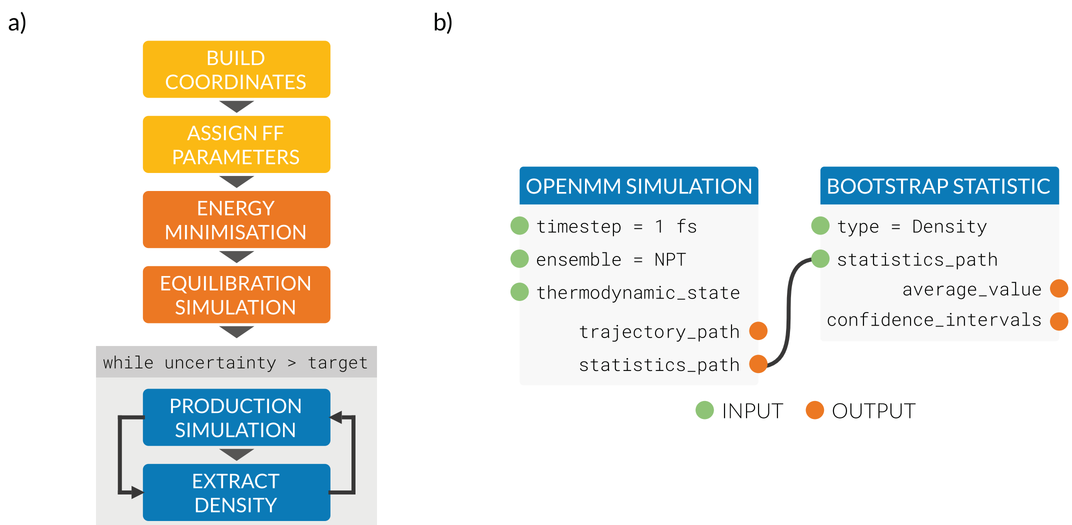

Figure 4. Physical properties are estimated using modular, lightweight workflows. a) An example workflow to estimate the density of a substance, composed of built-in workflow protocols chained together. b) Each protocol has a number of well-defined inputs that can either take their values from the output of other protocols, or by having their value set directly.

Each protocol which may be used in the workflow engine is defined as a Python object which is completely decoupled from the workflow engine and hence may be used outside of workflows. An example of initializing a protocol which will perform a simulation, and one which will then analyze the output of that simulation is shown in Figure 5.

```
run_simulation = OpenMMSimulation("run_simulation")
run_simulation.timestep = 1.0 * unit.femtosecond
run_simulation.ensemble = Ensemble.NPT

analysis = ExtractAverageStatistic("extract_density")
analysis.statistics = ProtocolPath("statistics_path", "run_simulation")
```

Figure 5. Pseudocode for initializing and chaining together workflow protocols. Each workflow protocol is described by a unique Python object, which has a number of attributes flagged as inputs, and a number flagged as outputs. Inputs and outputs of protocols are connected together using 'ProtocolPath' objects, which are essentially pointers to the output of another protocol in the workflow as identified by its unique id and the name of its output attribute (Figure 4b). These pointer objects will be automatically replaced with the actual output value of the reference protocol by the workflow manager once the previous protocol has been executed.

In addition to simply chaining together individual protocols into larger workflows, the workflow engine offers a number of more advanced features. In particular it is able to:

- detect when multiple workflows contain protocols that receive an identical set of inputs and remove these redundant steps before executing.
- parallelize parts of a workflow for a list of inputs. This is useful, for example, when defining part of a workflow which estimates the enthalpy of a particular component which should then be repeated for each component in a particular system.
- be executed using any one of the built-in, or user defined, calculation backends, thus allowing workflows to be scaled from running on a single laptop up to being parallelized across multiple nodes on a HPC cluster.

2.4 Supported Properties and Derivatives

A key goal of the framework is to enable the seamless estimation of data sets of physical properties using a variety of different calculation approaches without user intervention. This is accomplished in the framework through the definition of 'calculation schemas' that encode the exact workflow that must be followed to compute a particular property using a particular calculation approach.

For calculation approaches which make use of the built-in workflow engine, which includes the built-in simulation and cached data reweighting approaches, the calculation schema predominantly defines which protocols are required how they are chained together. Defining properties in this way enables new properties to be readily added to the framework, either directly or through the flexible plug-in system.

The properties which have built-in calculation schemas are summaries in Table 2 and are detailed in full in the frameworks documentation [19].

Table 2. The types of physical property which are by default supported by the framework: the mass density (ρ), dielectric constant (ϵ), enthalpies of vaporization and mixing (ΔH_{vap} and ΔH_{mix} respectively), excess molar volume (ΔV_{ex}) and solvation free energy (ΔG_{solv}). New physical properties are readily supported through user created plugins.

		Direct Simulation		**MBAR Reweighting**	
		Supported	**Derivatives**	**Supported**	**Derivatives**
Mass Density	ρ	✓	✓	✓	✓
Dielectric Constant	ϵ	✓	✓	✓	✓
Enthalpy of Vaporization	ΔH_{vap}	✓	✓	✓	✓
Enthalpy of Mixing	ΔH_{mix}	✓	✓	✓	✓
Excess Molar Volume	ΔV_{ex}	✓	✓	✓	✓
Solvation Free Energy	ΔG_{solv}	✓	✓	-	-

The derivatives of almost all properties with respect to force field parameters may be optionally estimated alongside the value of the property itself. From version 0.3.0 of the framework onwards, all such derivatives are computed using the fluctuation formula [29] according to

$$\frac{\mathrm{d}\langle X\rangle}{\mathrm{d}\theta_i} = \left\langle \frac{\mathrm{d}X}{\mathrm{d}\theta_i} \right\rangle - \beta \left[\left\langle X \frac{\mathrm{d}U}{\mathrm{d}\theta_i} \right\rangle - \left\langle \frac{\mathrm{d}U}{\mathrm{d}\theta_i} \right\rangle \langle X \rangle \right] \tag{1}$$

where X is the observable of interest, θ_i is the force field parameter the derivative is being taken with respect to, U is the system energy and $\langle \cdot \rangle$ is used to represent an ensemble average.

While future versions of the framework will aim to support differentiable simulation engines (such as timemachine [30]) which can compute $\frac{\mathrm{d}U}{\mathrm{d}\theta_i}$ directly, currently most common simulation engines do not directly support computing this quantity. Until such support is added, the framework employs a central finite difference approach, whereby

$$\frac{\mathrm{d}U}{\mathrm{d}\theta_i} \approx \frac{U\left(\theta_i + h\right) - U\left(\theta_i - h\right)}{2h} \tag{2}$$

and U is computed by re-evaluating the energy of each configuration generated during a simulation using the perturbed force field parameters. Although more expensive than computing either the forward or backwards derivative, the central difference method should give a more accurate estimate of the gradient at the minima, maxima and transition points. By default a value of $h = \theta_i \times 10^{-4}$ is used.

3 Applications

3.1 Force Field Assessment

The framework offers a scalable platform for assessing the performance of common force fields against physical property data sets, being able to seamlessly distribute the individual steps needed to estimate a particular property across many compute nodes and graphical processing units. Moreover, the framework has built-in support for estimating physical properties using most of the commonly available force fields, including SMIRNOFF based force fields through integration with the OpenFF toolkit [31], GAFF and GAFF2 force fields through integration with LEaP [32] and the publicly available OPLS force fields through integration with LigParGen [33, 34], enabling comparison of different force fields by changing a single line of Python.

Of particular value is the framework's ability to automatically detect redundant calculations when estimating data sets of physical properties. Consider the case of estimating the excess molar volume and enthalpy of mixing of the same substance at the same state. The framework will automatically detect that the density and enthalpy of the mixture, and that of each of the components, can be computed using the same simulation without human intervention, thus in cases drastically reducing the cost of the assessment.

To demonstrate this ability, the OpenFF 1.0.0 (openff-1.0.0), GAFF 1.8 (gaff-1) and GAFF 2.1 (gaff-2) force fields were assessed against a data set of 103 density $\rho(x)$, 101 enthalpy of mixing $\Delta H_{mix}(x)$ and 100 excess molar volume $V_{excess}(x)$ data points measured at ambient conditions for a set of binary systems each at three different compositions (25%, 50% and 75%). It contains a total of 36 unique binary mixtures of 39 unique components, and all data points were sourced directly from the ThermoML archive using the framework's built in parsers. All calculation were performed using v0.3.1 of the framework and using the default calculation schemas as described in the documentation [19].

Such a data set would naively require a total of 706 simulations to be performed and analyzed: three for each $\Delta H_{mix}(x)$ and $V_{excess}(x)$ data point, and one for each $\rho(x)$ data point. If all the data points in the set were measured at identical state points (i.e. the same temperature, pressure and composition) then the same data set could in principle be estimated using only 142 simulations if redundant simulations were removed. 38 simulations would be required to compute the density and enthalpy of each of the individual components, while 104 simulations would be required to compute the same for each binary mixture at the three different compositions. In practice, due to certain data points being measured at slightly different conditions (e.g. at 308.15 K rather than 298.15 K) and concentrations, the data set used for this study required a total of 246 simulations after redundant calculations have been removed. Still, this is roughly a third of the simulations which would have been required had the redundant ones not been removed.

The results of this assessment of the three force fields are presented in Figure 6. In general the performance of the three different force fields are roughly comparable. This is consistent with with expectations; the largest differences between these force fields are in valence parameters, which typically are thought not to play a dramatic role in calculations of the physical properties considered here.

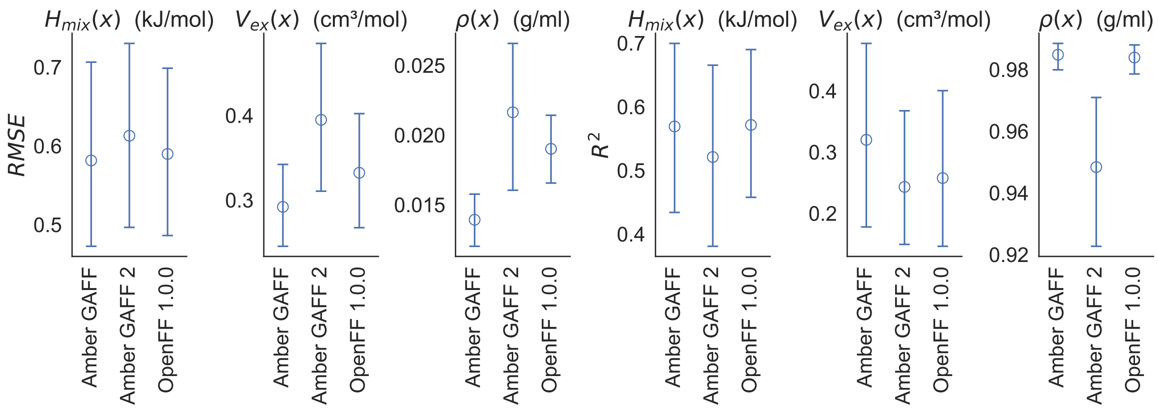

Figure 6. An assessment of the OpenFF 1.0.0, GAFF 1.8, and GAFF 2.1 force fields against a set of 304 $\rho(x)$, $\Delta H_{mix}(x)$ and $V_{ex}(x)$ data points measured for binary systems. In general the different force fields show a similar level of performance for the current test set. All errors in the RMSE and R^2 are shown as 95% confidence intervals computed by bootstrapping the physical property measurements.

3.2 Force Field Training

The framework offers a powerful, flexible route to estimating large data sets of physical properties as well as their first derivatives with respect to the force field parameters used in the estimations. This readily allows for the training of such parameters against the physical property data without requiring human intervention at each training epoch through integration with the ForceBalance optimization package. Moreover, the framework's ability to automatically employ reweighting of cached simulations is designed to enable a speed up of successive optimization epochs provided the changes in parameters are sufficiently small. We demonstrate these abilities here by retraining the non-bonded van der Waals (vdW) parameters of the OpenFF 1.0.0 (openff-1.0.0) force field against a total of 114 liquid density and enthalpy of vaporization measurements made at ambient conditions for a set of alcohols, acids, esters, ethers, ketones and alkanes.

The selected training set exercises a total of 18 vdW force field parameters (8 hydrogen parameters, 4 carbon parameters and 6 oxygen parameters) all of which were optimized. The training was initially performed using a combination of both molecular simulations and cached simulation data to estimate the data set at each epoch, and then was repeated using only molecular simulation so as to determine what speed up (if any) is provided by the cached data reweighting. A regularized least squares objective function as implemented by the ForceBalance software package was used, where the contribution of the physical properties was computed by:

$$\sum_{n}^{N} \frac{1}{M_n} \sum_{m}^{M_n} \frac{1}{d_n^2} \left(y_m^{ref} - y_m\left(\vec{\theta}\right) \right)^2 \tag{3}$$

where $\vec{\theta}$ is a vector of the parameters being trained, N is the number of types of physical property, M_n is the number of data points of type n, d_n is a weight associated with a particular property type with the same units as the property, y^{ref} is the value of the experimental data point and y_m is the estimated value. The training hyperparameters as required by ForceBalance are provided in Table 3, and are described more fully in [20]. All properties were computed using the default density and enthalpy of vaporization schemas but the number of molecules included in the simulation box when performing the simulations was reduced from 1000 to 500. This was done to increase the likelihood that the cached data reweighting would be employed when estimating the physical properties, given that the degree of overlap between two states decreases as the system size increases. By default only the four most recent pieces of cached simulation data are chosen for reweighting. This limits the overhead associated with attempting to reweight data which does not sufficiently overlap with the current state, which if uncapped would increase linearly with the number of training iterations performed.

Table 3. The key hyperparameters used as input to ForceBalance for each of the training runs.

Hyperparameter	Value
d_ρ	0.05 g / ml
$d_{\Delta H_{vap}}$	25.5 kJ / mol
ϵ prior	0.1 kcal / mol
$\frac{r_{min}}{2}$ prior	1.0 Å

The objective function at each training iteration is shown in Figure 7. For the two training runs performed, both with and without reweighting, the least squares objective function was found to decrease rapidly after the first iteration to a similar minimum value before fluctuating around a close to constant minimum. This fluctuation is observed due to noise in the estimated physical properties and hence also in their first derivatives with respect to the force field parameters being trained. The reweighting of cached simulation data therefore enables a sufficiently comparable estimation of both the objective function and its derivative with respect to the force field parameters being trained to be used as part of the parameter training as an appropriate replacement to the full simulation approach.

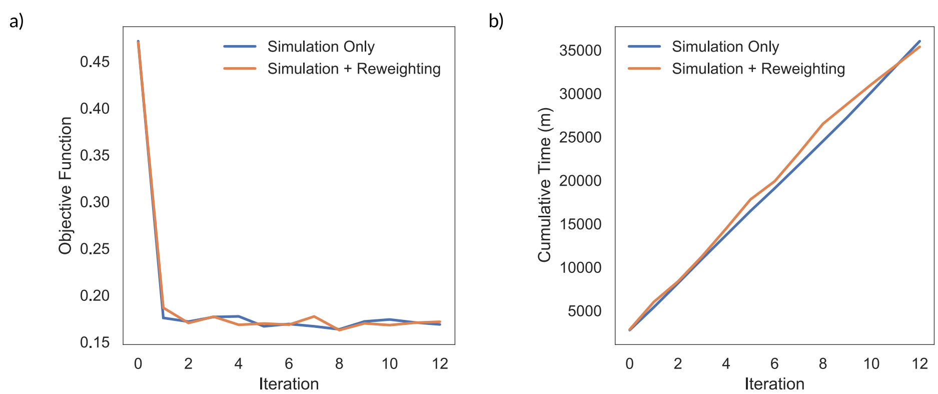

Figure 7. Employing a combination of cached data reweighting and molecular simulation did not significantly speed up the training compared to only employing molecular simulation. a) The objective function decreases to a similar value whether cached simulation data reweighting was employed or not. b) The use of cached simulation data reweighting did not significantly speed up the training of the force field parameters.

The cumulative time taken to reach the end of each training iteration is also shown in Figure 7. While hypothesized, based on previous use of reweighting in Bayesian inference of parameters [35], that employing reweighting of cached simulation data should enable a large speed up once enough data has been stored to facilitate the technique with sufficient accuracy, in this application it does not appear to be faster than simply estimating the objective function using only molecular simulation.

There are several possible reasons for why the cached data reweighting did not speed up the training of the force field parameters. A breakdown for which percentage of the different types of properties were able to be computed from cached simulation data, as well as a breakdown of how much time was needed to estimate those properties by either simulation or reweighting cached simulation data, is shown in Figure 8.

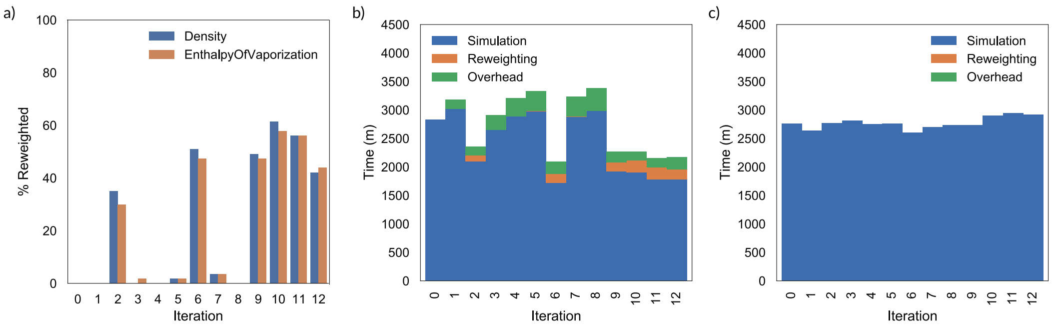

Figure 8. A breakdown of how often cached data reweighting is employed over direct molecular simulation. a) The percentage of training data points of each property type which were estimated using the two available approaches for each training iteration. b) The time spent by each calculation approach when estimating the data set at each iteration. The overhead associated with attempting to reweight data points which then ultimately had to be simulated is included in green. c) The total time to complete each iteration when only employing direct simulations.

As the training progresses and more simulation data is cached, a point is reached where there is a sufficient amount of cached data to accurately begin estimated a number of physical properties using reweighting. Although it was observed that reweighting was able to estimate the physical properties faster (on average roughly 5 minutes per property) than by direct simulation (on average roughly 25 minutes per property) the overhead (green bars in Figure 8) associated with attempting to reweight when there is not enough cached simulation data to yield an accurate estimate of a data point (less the 50 effective samples) is somewhat large. In these cases a new simulation must be performed instead in addition to the failed attempt at reweighting. There is currently no way to detect whether there will be a sufficient amount of cached data to reweight until reweighting has actually been attempted, and hence this overhead will always be present.

A further, and likely the biggest issue, is that the number of properties which may accurately estimated using cached simulation data reweighting is on average less than 50% of the total number of properties to estimate. This is a consequence of the optimizer performing, in a sense, too well, and the force field parameters varying by too large an amount at each new iteration compared to the previous iteration, such that there are an insufficient number of effective samples at the new state. While the step size of the algorithm could be reduced in order to ensure that reweighting is employed more frequently, it is not clear that this would always be optimal. It can be seen in Figure 7 that the objective function has already greatly decreased by the first few iterations before there is even enough data to be able to employ reweighting. It should be noted however that this optimization was performed on a relatively small training set. For large training sets it is likely that the optimization would take longer to converge to a minimum, and hence in these cases it is likely that reducing the step size so that reweighting is employed would be beneficial.

Finally, it should be noted that the physical properties included in the training set (densities and enthalpies) are themselves relatively 'cheap' to simulate, requiring only short simulations (on the order of a nanosecond) to converge their ensemble averages. The real advantage of reweighting will likely come when applied to more expensive physical properties, including solvation free energies and binding free energies which take on the order of hours to simulate, but would take only minutes to reweight. The framework is set up to, in the future, be able to support reweighting such properties through the robust workflow engine and flexible plugin architecture.

4 Obtaining the Framework

The framework is fully open source and available under the MIT license on GitHub [36]. It is readily installable with the conda command `conda install -c conda-forge openff-evaluator`. See the documentation [19] for full installation instructions.

To provide feedback on performance of the OpenFF force fields, we highly recommend using the issue tracker at http://github.com/openforcefield/openff-evaluator. Alternatively, inquiries may be e-mailed to support@openforcefield.org, though responses to e-mails sent to this address may be delayed and GitHub issues receive higher priority. For information on getting started with OpenFF, please see the documentation linked at https://openff-evaluator.readthedocs.io/en/stable/, and note the availability of several introductory examples.

5 Conclusion

The OpenFF Evaluator framework is a flexible, scalable and highly extensible framework for curating data sets from large, open data sources and estimating those data sets of physical property measurements and their derivatives with respect to force field parameters for optimization. The framework can use a range of common force fields, as well as an expandable range of estimation techniques. Through integration with optimization engines such as ForceBalance, the framework readily facilitates the training of new force fields directly against physical property data, as well as assessing such force fields against even larger data sets. In this work, we lay out how this framework can be used to optimize force fields, and discovered that for parameter optimization of simple physical properties of liquids such as densities and heats of vaporization, reweighting using cached data from previous iterations of optimization may not be efficient compared to direct physical simulation.

6 Acknowledgements and Funding

SB acknowledges support from a Joint OpenFF-XtalPi Distinguished Postdoctoral Fellowship and an Open Force Field Consortium Fellowship. JDC acknowledges support from NSF grant CHE-1738979, NIH grant P30CA008748, NIH grant R01GM132386, and the Sloan Kettering Institute. Research reported in this publication was in part supported by the National Institute of General Medical Sciences of the National Institutes of Health under award number R01GM132386. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

We thank the Open Force Field Consortium and Initiative for scientific and financial support, and Molecular Sciences Software Institute (MolSSI) for its support of the Open Force Field Initiative. We also thank David Slochower (ORCID: 0000-0003-3928-5050) and Jeff Wagner (ORCID: 0000-0001-6448-0873) for useful discussions on the API of the framework and Josh Fass (ORCID: 0000-0003-3719-266X) and Owen Madin (ORCID: 0000-0002-6736-3442) for useful discussions on reweighting and gradient calculations.

7 Author Contributions

Contributions based on CRediT taxonomy:
S.B.: Conceptualization, Writing – Original Draft, Writing – Review & Editing, Methodology, Investigation
L.P.W.: Writing – Review & Editing, Funding Acquisition
D.L.M.: Conceptualization, Writing – Review & Editing, Funding Acquisition
J.D.C.: Conceptualization, Writing – Review & Editing, Supervision, Resources, Funding Acquisition
M.R.S.: Conceptualization, Writing – Review & Editing, Supervision, Funding Acquisition

8 Disclosures

MRS is an Open Science Fellow at and consults for Silicon Therapeutics. DLM is an Open Science Fellow with Silicon Therapeutics and serves on the Scientific Advisory Board for OpenEye Scientific Software. SB is the director of Boothroyd Scientific Consulting Ltd. JDC is a current member of the Scientific Advisory Board of OpenEye Scientific Software, Redesign Science, and Interline Therapeutics, and has equity interests in Redesign Science and Interline Therapeutics. The Chodera laboratory receives or has received funding from multiple sources, including the National Institutes of Health, the National Science Foundation, the Parker Institute for Cancer Immunotherapy, Relay Therapeutics, Entasis Therapeutics, Silicon Therapeutics, EMD Serono (Merck KGaA), AstraZeneca, Vir Biotechnology, Bayer, XtalPi, Interline Therapeutics, the Molecular Sciences Software Institute, the Starr Cancer Consortium, the Open Force Field Consortium, Cycle for Survival, a Louis V. Gerstner Young Investigator Award, and the Sloan Kettering Institute. A complete funding history for the Chodera lab can be found at http://choderalab.org/funding.

9 Supporting Information

The inputs and scripts used to produce and analyse the results presented in this publication are provided at https://github.com/SimonBoothroyd/openff-evaluator-publication as a tagged release (1.0.0)

References

[1] **Boulanger E**, Huang L, Rupakheti C, MacKerell Jr AD, Roux B. Optimized Lennard-Jones parameters for druglike small molecules. Journal of chemical theory and computation. 2018; 14(6):3121–3131.

[2] **Stroet M**, Koziara KB, Malde AK, Mark AE. Optimization of empirical force fields by parameter space mapping: A single-step perturbation approach. Journal of chemical theory and computation. 2017; 13(12):6201–6212.

[3] **Horn HW**, Swope WC, Pitera JW, Madura JD, Dick TJ, Hura GL, Head-Gordon T. Development of an improved four-site water model for biomolecular simulations: TIP4P-Ew. The Journal of chemical physics. 2004; 120(20):9665–9678.

[4] **Horn HW**, Swope WC, Pitera JW. Characterization of the TIP4P-Ew water model: Vapor pressure and boiling point. The Journal of chemical physics. 2005; 123(19):194504.

[5] **Jorgensen WL**, Maxwell DS, Tirado-Rives 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(45):11225–11236. https://doi.org/10.1021/ja9621760, doi: 10.1021/ja9621760.

[6] **Wang J**, Wolf RM, Caldwell JW, Kollman PA, Case DA. Development and testing of a general amber force field. Journal of Computational Chemistry. 2004; 25(9):1157–1174. https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.20035, doi: 10.1002/jcc.20035.

[7] **Dauber-Osguthorpe P**, Hagler AT. Biomolecular force fields: where have we been, where are we now, where do we need to go and how do we get there? Journal of Computer-Aided Molecular Design. 2018 Nov; 33(2):133–203. https://doi.org/10.1007/s10822-018-0111-4, doi: 10.1007/s10822-018-0111-4.

[8] **Frenkel M**, Chirico RD, Diky VV, Dong Q, Frenkel S, Franchois PR, Embry DL, Teague TL, Marsh KN, Wilhoit RC. ThermoMLAn XML-Based Approach for Storage and Exchange of Experimental and Critically Evaluated Thermophysical and Thermochemical Property Data. 1. Experimental Data. Journal of Chemical & Engineering Data. 2003 Jan; 48(1):2–13. https://doi.org/10.1021/je025645o, doi: 10.1021/je025645o.

[9] **Chirico RD**, Frenkel M, Diky VV, Marsh KN, Wilhoit RC. ThermoMLAn XML-Based Approach for Storage and Exchange of Experimental and Critically Evaluated Thermophysical and Thermochemical Property Data. 2. Uncertainties. Journal of Chemical & Engineering Data. 2003 Sep; 48(5):1344–1359. https://doi.org/10.1021/je034088i, doi: 10.1021/je034088i.

[10] **Chirico RD**, Frenkel M, Diky V, Goldberg RN, Heerklotz H, Ladbury JE, Remeta DP, Dymond JH, Goodwin ARH, Marsh KN, Wakeham WA. ThermoML†—An XML-Based Approach for Storage and Exchange of Experimental and Critically Evaluated Thermophysical and Thermochemical Property Data. 4. Biomaterials. Journal of Chemical & Engineering Data. 2010 Apr; 55(4):1564–1572. https://doi.org/10.1021/je900685d, doi: 10.1021/je900685d.

[11] **Frenkel M**, Diky V, Chirico RD, Goldberg RN, Heerklotz H, Ladbury JE, Remeta DP, Dymond JH, Goodwin ARH, Marsh KN, Wakeham WA, Stein SE, Brown PL, Königsberger E, Williams PA. ThermoML: an XML-Based Approach for Storage and Exchange of Experimental and Critically Evaluated Thermophysical and Thermochemical Property Data. 5. Speciation and Complex Equilibria. Journal of Chemical & Engineering Data. 2011 Feb; 56(2):307–316. https://doi.org/10.1021/je100999j, doi: 10.1021/je100999j.

[12] An XML-Based IUPAC Standard for Storage and Exchange of Experimental Thermophysical and Thermochemical Property Data;. https://trc.nist.gov/ThermoML.html.

[13] **Mobley DL**, Guthrie JP. FreeSolv: a database of experimental and calculated hydration free energies, with input files. Journal of Computer-Aided Molecular Design. 2014 Jun; 28(7):711–720. https://doi.org/10.1007/s10822-014-9747-x, doi: 10.1007/s10822-014-9747-x.

[14] **Mobley DL**, Chodera J, Beauchamp K, Lee-Ping, Mobleylab/Freesolv: Version 0.320. Zenodo; 2016. https://zenodo.org/record/596537, doi: 10.5281/ZENODO.596537.

[15] **Liu T**, Lin Y, Wen X, Jorissen RN, Gilson MK. BindingDB: a web-accessible database of experimentally determined protein-ligand binding affinities. Nucleic Acids Research. 2007 Jan; 35(Database):D198–D201. https://doi.org/10.1093/nar/gkl999, doi: 10.1093/nar/gkl999.

[16] **Pontolillo J**, Eganhouse RP. The search for reliable aqueous solubility (Sw) and octanol-water partition coefficient (Kow) data for hydrophobic organic compounds: DDT and DDE as a case study, vol. 1. US Department of the Interior, US Geological Survey; 2001.

[17] **Qiu Y**, Smith D, Boothroyd S, Jang H, Wagner J, Bannan CC, Gokey T, Lim VT, Stern C, Rizzi A, et al. Development and Benchmarking of Open Force Field v1.0.0, the Parsley Small Molecule Force Field. ChemRxiv. 2021; doi: 10.33774/chemrxiv-2021-l070l-v3.

[18] **Wang J**, Development of the Second Generation of the General AMBER Force Field; 2017. https://www.researchgate.net/project/Development-of-the-Second-Generation-of-the-General-AMBER-Force-Field.

[19] OpenFF Evaluator Documentation; 2020. https://openff-evaluator.readthedocs.io/en/stable/.

[20] **Wang LP**, Chen J, Voorhis TV. Systematic Parametrization of Polarizable Force Fields from Quantum Chemistry Data. Journal of Chemical Theory and Computation. 2012 Nov; 9(1):452–460. https://doi.org/10.1021/ct300826t, doi: 10.1021/ct300826t.

[21] **Wang LP**, Martinez TJ, Pande VS. Building Force Fields: An Automatic, Systematic, and Reproducible Approach. The Journal of Physical Chemistry Letters. 2014 May; 5(11):1885–1891. https://doi.org/10.1021/jz500737m, doi: 10.1021/jz500737m.

[22] **Qiu Y**, Nerenberg PS, Head-Gordon T, Wang LP. Systematic Optimization of Water Models Using Liquid/Vapor Surface Tension Data. The Journal of Physical Chemistry B. 2019 Jul; 123(32):7061–7073. https://doi.org/10.1021/acs.jpcb.9b05455, doi: 10.1021/acs.jpcb.9b05455.

[23] Deploy Dask on Job Queueing systems;. https://github.com/dask/dask-jobqueue.

[24] **Shirts MR**, Chodera JD. Statistically optimal analysis of samples from multiple equilibrium states. The Journal of Chemical Physics. 2008; 129(12):124105. https://doi.org/10.1063/1.2978177, doi: 10.1063/1.2978177.

[25] **Messerly RA**, Razavi SM, Shirts MR. Configuration-Sampling-Based Surrogate Models for Rapid Parameterization of Non-Bonded Interactions. Journal of Chemical Theory and Computation. 2018; 14(6):3144–3162. https://doi.org/10.1021/acs.jctc.8b00223, doi: 10.1021/acs.jctc.8b00223, pMID: 29727563.

[26] A distributed task scheduler for Dask;. https://github.com/dask/distributed/.

[27] **Rizzi A**, Chodera J, Naden L, Beauchamp K, Albanese S, Grinaway P, Prada-Gracia D, Rustenburg B, ajsilveira, Saladi S, Boehm K, Gmach J, Rodríguez-Guerra J, choderalab/yank: 0.25.2 - Bugfix release. Zenodo; 2019. https://doi.org/10.5281/zenodo.3534289, doi: 10.5281/zenodo.3534289.

[28] **Eastman P**, Swails J, Chodera JD, McGibbon RT, Zhao Y, Beauchamp KA, Wang LP, Simmonett AC, Harrigan MP, Stern CD, et al. OpenMM 7: Rapid development of high performance algorithms for molecular dynamics. PLoS computational biology. 2017; 13(7):e1005659.

[29] **Wang LP**, Head-Gordon T, Ponder JW, Ren P, Chodera JD, Eastman PK, Martinez TJ, Pande VS. Systematic improvement of a classical molecular model of water. The Journal of Physical Chemistry B. 2013; 117(34):9956–9972.

[30] Time Machine - A high-performance differentiable molecular dynamics and optimization engine.; 2020. https://github.com/proteneer/timemachine.

[31] **Wagner J**, Mobley DL, Chodera J, Bannan C, Rizzi A, Thompson M, Horton J, Dotson D, , Camila, Rodríguez-Guerra J, Bayly C, JoshHorton, Lim NM, Trevorgokey, Lim V, SimonBoothroyd, Sukanya Sasmal, Smith D, Lee-Ping, Yutong Zhao, openforcefield/openforcefield: 0.7.1 OETK2020 Compatibility and Minor Update. Zenodo; 2020. https://zenodo.org/record/597754, doi: 10.5281/ZENODO.597754.

[32] **Case D**, Belfon K, Ben-Shalom I, Brozell S, Cerutti D, Cheatham III T, Cruzeiro V, Darden T, Duke R, Giambasu G, et al., AMBER 2020. University of California San Francisco; 2020.

[33] **Dodda LS**, de Vaca IC, Tirado-Rives J, Jorgensen WL. LigParGen web server: an automatic OPLS-AA parameter generator for organic ligands. Nucleic Acids Research. 2017 Apr; 45(W1):W331–W336. https://doi.org/10.1093/nar/gkx312, doi: 10.1093/nar/gkx312.

[34] **Dodda LS**, Vilseck JZ, Tirado-Rives J, Jorgensen WL. 1.14*CM1A-LBCC: Localized Bond-Charge Corrected CM1A Charges for Condensed-Phase Simulations. The Journal of Physical Chemistry B. 2017 Mar; 121(15):3864–3870. https://doi.org/10.1021/acs.jpcb.7b00272, doi: 10.1021/acs.jpcb.7b00272.

[35] **Messerly RA**, Razavi SM, Shirts MR. Configuration-sampling-based surrogate models for rapid parameterization of non-bonded interactions. Journal of Chemical Theory and Computation. 2018; 14(6):3144–3162.

[36] OpenFF Evaluator - A physical property evaluation toolkit from the Open Forcefield Consortium.; 2020. https://github.com/openforcefield/openff-evaluator.