# **OpenFF Evaluator Documentation**

openff-evaluator

# **GETTING STARTED**

1	Calcu	ulation Approaches	3
2	Supp	orted Physical Properties	5
	2.1	Installation	6
	2.2	Architecture	7
	2.3	Evaluator Client	7
	2.4	Evaluator Server	9
	2.5	Tutorial 01 - Loading Data Sets	11
	2.6	Tutorial 02 - Estimating Data Sets	15
	2.7	Tutorial 03 - Analysing Data Sets	20
	2.8	Tutorial 04 - Optimizing Force Fields	22
	2.9	Property Data Sets	29
	2.10	ThermoML Archive	31
	2.11	Taproom	33
	2.12	Data Set Curation	34
	2.13	Physical Properties	40
	2.14	Common Workflows	44
	2.15	Gradients	46
	2.16	Calculation Layers	47
	2.17	Workflow Layers	50
	2.18	The Direct Simulation Layer	52
	2.19	The MBAR Reweighting Layer	52
	2.20	Workflows	53
	2.21	Replicators	55
	2.22	Workflow Graphs	59
	2.23	Protocols	60
	2.24	Protocol Groups	63
	2.25	Observables	64
	2.26	Calculation Backends	66
	2.27	Dask Backends	67
	2.28	Storage Backends	69
	2.29	Data Classes and Queries	70
	2.30	Local File Storage	72
	2.31	Building the Docs	73
	2.32	API	73
	2.33	Release History	626
	2.34	Release Process	640
Bil	oliogra	aphy	643

Index 645

An automated and scalable framework for curating, manipulating, and computing data sets of physical properties from molecular simulation and simulation data.

The framework is built around four central ideas:

- **Flexibility:** New physical properties, data sources and calculation approaches are easily added via an extensible plug-in system and a flexible workflow engine.
- Automation: *Physical property measurements* are readily importable from open data sources (such as the NIST ThermoML Archive) through the data set APIs, and automatically calculated using either the built-in or user specified calculation schemas.
- **Scalability:** Calculations are readily scalable from single machines and laptops up to large HPC clusters and supercomputers through seamless integration with libraries such as dask.
- Efficiency: Properties are estimated using the fastest approach available to the framework, whether that be through evaluating a trained surrogate model, re-evaluating cached simulation data, or by running simulations directly.

GETTING STARTED 1

2 GETTING STARTED

**CHAPTER** 

ONE

# **CALCULATION APPROACHES**

The framework is designed around the idea of allowing multiple calculation approaches for estimating the same set of properties, in addition to estimation directly from molecular simulation, all using a uniform API.

The primary purpose of this is to take advantage of the many techniques exist which are able to leverage data from previous simulations to rapidly estimate sets of properties, such as reweighting cached simulation data, or evaluating surrogate models trained upon cached data. The most rapid approach which may accurately estimate a set of properties is automatically determined by the framework on the fly.

Each approach supported by the framework is implemented as a *calculation layer*. Two such layers are currently supported (although new calculation layers can be readily added via the plug-in system):

- evaluating physical properties directly from molecular simulation using the *SimulationLayer*.
- reprocessing cached simulation data with MBAR reweighting using the ReweightingLayer.

# SUPPORTED PHYSICAL PROPERTIES

The framework has built-in support for evaluating a number of *physical properties*, ranging from relatively 'cheap' to compute properties such as liquid densities, up to more computationally demanding properties such as solvation free energies and host-guest binding affinities.

Included for most of these properties is the ability to calculate their derivatives with respect to force field parameters, making the framework ideal for evaluating an objective function and it's gradient as part of a force field optimisation.

Table 1: The physical properties which are natively supported by the framework.

	Direct Simulation		MBAR Reweighting	
	Supported	Gradients	Supported	Gradients
Density	<b>√</b>	<b>√</b>	<b>√</b>	✓
Dielectric Constant	<b>√</b>	<b>/</b> *	<b>√</b>	<b>√</b> *
$H_{ ext{vaporization}}$	<b>√</b>	<b>√</b>	<b>√</b>	✓
$H_{ m mixing}$	<b>√</b>	<b>√</b>	<b>√</b> *	✓
V <sub>excess</sub>	<b>√</b>	<b>√</b>	<b>√</b>	<b>√</b>
G <sub>solvation</sub>	<b>√</b>	<b>√</b> *	×	×
G <sub>host-guest</sub> (beta)	<b>√</b> *	×	×	×

\* Entries marked with an asterisk are supported but have not yet been extensively tested and validated.

See the *physical properties overview page* for more details.

# 2.1 Installation

The OpenFF Evaluator is currently installable either through conda or directly from the source code. Whichever route is chosen, it is recommended to install the framework within a conda environment and allow the conda package manager to install the required and optional dependencies.

More information about conda and instructions to perform a lightweight miniconda installation can be found here. It will be assumed that these have been followed and conda is available on your machine.

#### 2.1.1 Installation from Conda

To install the openff-evaluator from the conda-forge channel simply run:

conda install -c conda-forge openff-evaluator

# 2.1.2 Recommended Dependencies

If you have access to the fantastic OpenEye toolkit we recommend installing this to enable (among many other things) the use of the BuildDockedCoordinates protocol and faster conformer generation / AM1BCC partial charge calculations:

```
conda install -c openeye openeye-toolkits
```

To parameterize systems with the Amber tleap tool using a TLeapForceFieldSource the ambertools package must be installed:

```
conda install -c conda-forge 'ambertools >=19.0'
```

#### 2.1.3 Installation from Source

To install the OpenFF Evaluator from source begin by cloning the repository from github:

```
\label{lem:git_com_open} git \ clone \ https://github.com/openforcefield/openff-evaluator.git \ cd \ openff-evaluator \\
```

Create a custom conda environment which contains the required dependencies and activate it:

```
conda env create --name openff-evaluator --file devtools/conda-envs/test_env.yaml
conda activate openff-evaluator
```

Finally, install the estimator itself:

```
python setup.py develop
```

# 2.2 Architecture

The openff-evaluator framework is constructed as a collection of modular components, each performing a specific role within the estimation of physical property data sets. These components are designed to be as extensible as possible, with support for user created plug-ins built into their core.

Fig. 1: An overview of the openff-evaluators modular design. The framework is split into a 'client-side' which handles the curation and preparation of data sets, and a 'server-side' which performs the estimation of the data sets.

The framework is implemented as a *client-server* architecture. This design allows users to spin up *Evaluator Server* instances on whichever compute resources they may have available (from a single machine up to a large HPC cluster), and to which *Evaluator Client* objects may connect to both request that data sets be estimated, and to query and retrieve the results of those requests.

The *client-side* of the framework is predominantly responsible for providing APIs and objects for:

- curating *data sets* of physical properties from open data sources.
- specifing custom calculation schemas which describe how individual properties should be computed.
- requesting that data sets be estimated by a running *Evaluator Server* instance.
- retrieving the results of estimation requests from a running *Evaluator Server* instance.

while the *server-side* is responsible for:

- receiving estimation requests from an Evaluator Client object.
- automatically determining which *calculation approach* to use for each property in the request.
- executing those requests across the available compute resources following the calculation schemas provided by the client
- caching data from any calculations which may be useful for future calculations.

All communication between servers and clients is handled through the TCP protocol.

# 2.3 Evaluator Client

The *EvaluatorClient* object is responsible for both submitting requests to estimate a data set of properties to a running *Evaluator Server* instance, and for pulling back the results of those requests when complete.

An *EvaluatorClient* object may optionally be created using a set of *ConnectionOptions* which specifies the network address of the running *Evaluator Server* instance to connect to:

```
# Specify the address of a server running on the local machine.
connection_options = ConnectionOptions(server_address="localhost", server_port=8000)
# Create the client object
evaluator_client = EvaluatorClient(connection_options)
```

2.2. Architecture 7

### 2.3.1 Requesting Estimates

The client can request the estimation of a data set of properties using the request\_estimate() function:

```
# Specify the data set.
data_set = PhysicalPropertyDataSet()
data_set.add_properties(...)
# Specify the force field source.
force_field = SmirnoffForceFieldSource.from_path("openff-1.0.0.offxml")
# Specify some estimation options (optional).
options = client.default_request_options(data_set, force_field)
# Specify the parameters to differentiate with respect to (optional).
gradient_keys = [
   ParameterGradientKey(tag="vdW", smirks="[#6X4:1]", attribute="epsilon")
# Request the estimation of the data set.
request, errors = evaluator_client.request_estimate(
   data_set.
    force_field,
   options,
    gradient_keys
)
```

A request must at minimum specify:

- the *data set* of physical properties to estimate.
- the force field parameters to estimate the data set using.

and may also optionally specify:

- the options to use when estimating the property set.
- the parameters to differentiate each physical property estimate with respect to.

**Note:** Gradients can currently only be computed for requests using a SMIRNOFF based force field.

The request\_estimate() function returns back two objects:

- a Request object which can be used to retrieve the results of the request and,
- an EvaluatorException object which will be populated if any errors occured while submitting the request.

The *Request* object is similar to a Future object, in that it is an object which can be used to query the current status of a request either asynchronously:

```
results = request.results(synchronous=False)
```

or synchronously:

```
results = request.results(synchronous=True)
```

The results (which may currently be incomplete) are returned back as a RequestResult object.

The *Request* object is fully JSON serializable:

```
# Save the request to JSON
request.json(file_path="request.json", format=True)
# Load the request from JSON
request = Request.from_json(file_path="request.json")
```

making it easy to keep track of any open requests.

# 2.3.2 Request Options

The RequestOptions object allows greater control over how properties are estimated by the server. It currently allows control over:

- calculation\_layers: The calculation layers which the server should attempt to use when estimating the data set. The order which the layers are specified in this list is the order which the server will attempt to use each layer.
- *calculation\_schemas*: The *calculation schemas* to use for each allowed calculation layer per class of property. These will be automatically populated in the cases where no user specified schema is provided, and where a default schema has been registered with the plugin system for the particular layer and property type.

If no options are passed to <code>request\_estimate()</code> a default set will be generated through a call to <code>default\_request\_options()</code>. For more information about how default calculation schemas are registered, see the <code>Default Schemas</code> section.

#### 2.3.3 Force Field Sources

Different force field representations (e.g. SMIRNOFF, TLeap, LigParGen) are defined within the framework as *ForceFieldSource* objects. A force field source should specify *all* of the options which would be required by a particular force field, such as the non-bonded cutoff or the charge scheme if not specified directly in the force field itself.

Currently the framework has built in support for force fields applied via:

- the OpenFF toolkit (SmirnoffForceFieldSource).
- the tleap program from the AmberTools suite (LigParGenForceFieldSource).
- an instance of the LigParGen server (LigParGenForceFieldSource).

The client will automatically adapt any of the built-in calculation schemas which are based off of the WorkflowCalculationSchema to use the correct workflow protocol (BuildSmirnoffSystem, BuildTLeapSystem or BuildLigParGenSystem) for the requested force field.

### 2.4 Evaluator Server

The *EvaluatorServer* object is responsible for coordinating the estimation of physical property data sets as requested by *evaluator clients*. Its primary responsibilities are to:

- recieve incoming requests from an *evaluator clients* to either estimate a dataset of properties, or to query the status of a previous request.
- request that each specified *calculation layers* attempt to estimate the data set of properties, cascading unestimated properties through the different layers.

2.4. Evaluator Server 9

An *EvaluatorServer* must be created with an accompanying *calculation backend* which will be responsible for distributing any calculations launched by the different calculation layers:

```
with DaskLocalCluster() as calculation_backend:
    evaluator_server = EvaluatorServer(calculation_backend)
    evaluator_server.start()
```

It may also be optionally created using a specific storage backend if the default LocalFileStorage is not sufficient:

```
with DaskLocalCluster() as calculation_backend:
    storage_backend = LocalFileStorage()
    evaluator_server = EvaluatorServer(calculation_backend, storage_backend)
    evaluator_server.start()
```

By default the server will run synchronously until it is killed, however it may also be run asynchronously such that it can be interacted with directly by a client in the same script:

```
with DaskLocalCluster() as calculation_backend:
    with EvaluatorServer(calculation_backend) as evaluator_server:

# Specify the data set.
    data_set = PhysicalPropertyDataSet()
    data_set.add_properties(...)

# Specify the force field source.
    force_field = SmirnoffForceFieldSource.from_path("openff-1.0.0.offxml")

# Request the estimation of the data set.
    request, errors = evaluator_client.request_estimate(data_set,force_field)
    # Wait for the results.
    results = request.results(synchronous=True)
```

### 2.4.1 Estimation Batches

When a server recieves a request from a client, it will attempt to split the requested set of properties into smaller batches, represented by the *Batch* object. The server is currently only able to mark entire batches of estimated properties as being completed, as opposed to individual properties.

Currently the server supports two ways of batching properties:

- SameComponents: All properties measured for the substance containing the *same* components will be batched together. As an example, the density of a 80:20 and a 20:80 mix of ethanol and water would be batched together, but the density of pure ethanol and the density of pure water would be placed into separate batches.
- SharedComponents: All properties measured for substances containing at least one common component will be batched together. As an example, the densities of 80:20 and 20:80 mixtures of ethanol and water, and the pure densities of ethanol and water would be batched together.

The mode of batching is set by the client using the *batch\_mode* attribute of the request options.

# 2.5 Tutorial 01 - Loading Data Sets

In this tutorial we will be exploring the frameworks utilities for loading and manipulating data sets of physical property measurements. The tutorial will cover

- Loading a data set of density measurements from NISTs ThermoML Archive
- Filtering the data set down using a range of criteria, including temperature pressure, and composition.
- Supplementing the data set with enthalpy of vaporization ( $\Delta H_v$ ) data sourced directly from the literature

If you haven't yet installed the OpenFF Evaluator framework on your machine, check out the *installation instructions* here!

Note: If you are running this tutorial in google colab you will need to run a setup script instead of following the installation instructions:

```
[]: # !wget https://raw.githubusercontent.com/openforcefield/openff-evaluator/main/docs/

→tutorials/colab_setup.ipynb

# %run colab_setup.ipynb
```

For the sake of clarity all warnings will be disabled in this tutorial:

```
[]: import warnings
warnings.filterwarnings("ignore")
import logging
logging.getLogger("openff.toolkit").setLevel(logging.ERROR)
```

# 2.5.1 Extracting Data from ThermoML

For anyone who is not familiar with the ThermoML archive - it is a fantastic database of physical property measurements which have been extracted from data published in the

- Journal of Chemical and Engineering Data
- Journal of Chemical Thermodynamics
- Fluid Phase Equilibria
- · Thermochimica Acta
- International Journal of Thermophysics

journals. It includes data for a wealth of different physical properties, from simple densities and melting points, to activity coefficients and osmotic coefficients, all of which is freely available. As such, it serves as a fantastic resource for benchmarking and optimising molecular force fields against.

The Evaluator framework has built-in support for extracting this wealth of data, storing the data in easy to manipulate python objects, and for automatically re-computing those properties using an array of calculation techniques, such as molecular simulations and, in future, from trained surrogate models.

This support is provided by the ThermoMLDataSet object:

```
[]: from openff.evaluator.datasets.thermoml import ThermoMLDataSet
```

The ThermoMLDataSet object offers two main routes for extracting data the the archive:

- extracting data directly from the NIST ThermoML web server
- extracting data from a local ThermoML XML archive file

Here we will be extracting data directly from the web server. To pull data from the web server we need to specify the digital object identifiers (DOIs) of the data we wish to extract - these correspond to the DOI of the publication that the data was initially sourced from.

For this tutorial we will be extracting data using the following DOIs:

```
[]: data_set = ThermoMLDataSet.from_doi(
        "10.1016/j.fluid.2013.10.034",
        "10.1021/je1013476",
)
```

We can inspect the data set to see how many properties were loaded:

```
[ ]: len(data_set)
```

and for how many different substances those properties were measured for:

```
[ ]: len(data_set.substances)
```

We can also easily check which types of properties were loaded in:

```
[ ]: print(data_set.property_types)
```

# 2.5.2 Filtering the Data Set

The data set object we just created contains many different functions which will allow us to filter the data down, retaining only those measurements which are of interest to us.

The first thing we will do is filter out all of the measurements which aren't density measurements:

```
[]: from openff.evaluator.datasets.curation.components.filtering import (
    FilterByPropertyTypes,
    FilterByPropertyTypesSchema,
)

data_set = FilterByPropertyTypes.apply(
    data_set, FilterByPropertyTypesSchema(property_types=["Density"])
)

print(data_set.property_types)
```

Next we will filter out all measurements which were made away from atmospheric conditions:

```
[]: from openff.evaluator.datasets.curation.components.filtering import (
    FilterByPressure,
    FilterByPressureSchema,
    FilterByTemperature,
    FilterByTemperatureSchema,
)

print(f"There were {len(data_set)} properties before filtering")
```

(continues on next page)

(continued from previous page)

```
# First filter by temperature.
data_set = FilterByTemperature.apply(
    data_set,
    FilterByTemperatureSchema(minimum_temperature=298.0, maximum_temperature=298.2),
)
# and then by pressure
data_set = FilterByPressure.apply(
    data_set, FilterByPressureSchema(minimum_pressure=101.224, maximum_pressure=101.426)
)
print(f"There are now {len(data_set)} properties after filtering")
```

Finally, we will filter out all measurements which were not measured for either ethanol (CCO) or isopropanol (CC(C)O):

```
[]: from openff.evaluator.datasets.curation.components.filtering import (
    FilterBySmiles,
    FilterBySmilesSchema,
)

data_set = FilterBySmiles.apply(
    data_set, FilterBySmilesSchema(smiles_to_include=["CCO", "CC(C)O"])
)

print(f"There are now {len(data_set)} properties after filtering")
```

We will convert the filtered data to a pandas DataFrame to more easily visualize the final data set:

Through filtering, we have now cut down from over 250 property measurements down to just 2. There are many more possible filters which can be applied. All of these and more information about the data set object can be found in the PhysicalPropertyDataSet (from which the ThermoMLDataSet class inherits) API documentation.

### 2.5.3 Adding Extra Data

For the final part of this tutorial, we will be supplementing our newly filtered data set with some enthalpy of vaporization  $(\Delta H_v)$  measurements sourced directly from the literature (as opposed to from the ThermoML archive).

We will be sourcing values of the  $\Delta H_v$  of ethanol and isopropanol, summarised in the table below, from the Enthalpies of vaporization of some aliphatic alcohols publication:

Compound	Temperature / $K$	$\Delta H_v / kJmol^{-1}$	$\delta \Delta H_v / kJmol^{-1}$
Ethanol	298.15	42.26	0.02
Isopropanol	298.15	45.34	0.02

In order to create a new  $\Delta H_v$  measurements, we will first define the state (namely temperature and pressure) that the measurements were recorded at:

```
[]: from openff.units import unit
  from openff.evaluator.thermodynamics import ThermodynamicState
  thermodynamic_state = ThermodynamicState(
      temperature=298.15 * unit.kelvin, pressure=1.0 * unit.atmosphere
)
```

Note: Here we have made use of the ``openff.units.unit`` module to attach units to the temperatures and pressures we are filtering by. This module simply exposes a ``UnitRegistry`` from the fantastic pintlibrary. Pint provides full support for attaching to units to values and is used extensively throughout this framework.

the substances that the measurements were recorded for:

```
[]: from openff.evaluator.substances import Substance

ethanol = Substance.from_components("CCO")
isopropanol = Substance.from_components("CC(C)O")
```

and the source of this measurement (defined as the DOI of the publication):

```
[]: from openff.evaluator.datasets import MeasurementSource
source = MeasurementSource(doi="10.1016/S0021-9614(71)80108-8")
```

We will combine this information with the values of the measurements to create an object which encodes each of the  $\Delta H_v$  measurements

```
[]: from openff.evaluator.datasets import PropertyPhase
    from openff.evaluator.properties import EnthalpyOfVaporization

ethanol_hvap = EnthalpyOfVaporization(
        thermodynamic_state=thermodynamic_state,
        phase=PropertyPhase.Liquid | PropertyPhase.Gas,
        substance=ethanol,
        value=42.26 * unit.kilojoule / unit.mole,
        uncertainty=0.02 * unit.kilojoule / unit.mole,
        source=source,
)
```

(continues on next page)

(continued from previous page)

```
isopropanol_hvap = EnthalpyOfVaporization(
    thermodynamic_state=thermodynamic_state,
    phase=PropertyPhase.Liquid | PropertyPhase.Gas,
    substance=isopropanol,
    value=45.34 * unit.kilojoule / unit.mole,
    uncertainty=0.02 * unit.kilojoule / unit.mole,
    source=source,
)
```

These properties can then be added to our data set:

```
[]: data_set.add_properties(ethanol_hvap, isopropanol_hvap)
```

If we print the data set again using pandas we should see that our new measurements have been added:

### 2.5.4 Conclusion

We will finish off this tutorial by saving the data set we have created as a JSON file for future use:

```
[ ]: data_set.json("filtered_data_set.json", format=True);
```

And that concludes the first tutorial. For more information about data sets in the Evaluator framework check out the *data set* and *ThermoML* documentation.

In the next tutorial we will be estimating the data set we have created here using molecular simulation.

If you have any questions and / or feedback, please open an issue on the GitHub issue tracker.

# 2.6 Tutorial 02 - Estimating Data Sets

In this tutorial we will be estimating the data set we created in the *first tutorial* using molecular simulation. The tutorial will cover:

- loading in the data set to estimate, and the force field parameters to use in the calculations.
- defining custom calculation schemas for the properties in our data set.
- estimating the data set of properties using an *Evaluator server* instance.
- retrieving the results from the server and storing them on disk.

Note: If you are running this tutorial in google colab you will need to run a setup script instead of following the installation instructions:

```
[]: #!wget https://raw.githubusercontent.com/openforcefield/openff-evaluator/main/docs/

→tutorials/colab_setup.ipynb

# %run colab_setup.ipynb
```

For this tutorial make sure that you are using a GPU accelerated runtime.

For the sake of clarity all warnings will be disabled in this tutorial:

```
[]: import warnings
warnings.filterwarnings("ignore")
import logging
logging.getLogger("openff.toolkit").setLevel(logging.ERROR)
```

We will also enable time-stamped logging to help track the progress of our calculations:

```
[]: from openff.evaluator.utils import setup_timestamp_logging
setup_timestamp_logging()
```

### 2.6.1 Loading the Data Set and Force Field Parameters

We will begin by loading in the data set which we created in the previous tutorial:

```
[]: import pathlib
from openff.evaluator.datasets import PhysicalPropertyDataSet
data_set_path = "filtered_data_set.json"

# If you have not yet completed that tutorial or do not have the data set file
# available, this tutorial will use a copies provided by the framework

if not pathlib.Path(data_set_path).exists():
    from openff.evaluator.utils import get_data_filename

    data_set_path = get_data_filename("tutorials/tutorial01/filtered_data_set.json")

data_set = PhysicalPropertyDataSet.from_json(data_set_path)
```

As a reminder, this data contains the experimentally measured density and  $H_{vap}$  measurements for ethanol and isopropanol at ambient conditions:

```
[]: data_set.to_pandas().head()
```

We will also define the set of force field parameters which we wish to use to estimate this data set of properties. The framework has support for estimating force field parameters from a range of sources, including those in the OpenFF SMIRNOFF format, those which can be applied by AmberTools, *and more*.

Each source of a force field has a corresponding source object in the framework. In this tutorial we will be using the OpenFF Parsley force field which is based off of the SMIRNOFF format:

```
[]: from openff.evaluator.forcefield import SmirnoffForceFieldSource

force_field_path = "openff-1.0.0.offxml"

force_field_source = SmirnoffForceFieldSource.from_path(force_field_path)
```

# 2.6.2 Defining the Calculation Schemas

The next step we will take will be to define a custom calculation schema for each type of property in our data set.

A calculation schema is the blueprint for how a type of property should be calculated using a particular *calculation approach*, such as directly by simulation, by reprocessing cached simulation data or, in future, a range of other options.

The framework has built-in schemas defining how densities and  $H_{vap}$  should be estimated from molecular simulation, covering all aspects from coordinate generation, force field assignment, energy minimisation, equilibration and finally the production simulation and data analysis. All of this functionality is implemented via the frameworks built-in, lightweight *workflow engine*, however we won't dive into the details of this until a later tutorial.

For the purpose of this tutorial, we will simply modify the default calculation schemas to reduce the number of molecules to include in our simulations to speed up the calculations. This step can be skipped entirely if the default options (which we recommend using for 'real-world' calculations) are to be used:

```
[]: from openff.evaluator.properties import Density, EnthalpyOfVaporization

density_schema = Density.default_simulation_schema(n_molecules=256)
h_vap_schema = EnthalpyOfVaporization.default_simulation_schema(n_molecules=256)
```

We could further use this method to set either the absolute or the relative uncertainty that the property should be estimated to within. If either of these are set, the simulations will automatically be extended until the target uncertainty in the property has been met.

For our purposes however we won't set any targets, leaving the simulations to run for the default 1 ns.

To use these custom schemas, we need to add them to the a request options object which defines all of the options for estimating our data set:

```
[]: from openff.evaluator.client import RequestOptions

# Create an options object which defines how the data set should be estimated.
estimation_options = RequestOptions()

# Specify that we only wish to use molecular simulation to estimate the data set.
estimation_options.calculation_layers = ["SimulationLayer"]

# Add our custom schemas, specifying that the should be used by the 'SimulationLayer'
estimation_options.add_schema("SimulationLayer", "Density", density_schema)
estimation_options.add_schema("SimulationLayer", "EnthalpyOfVaporization", h_vap_schema)
```

### 2.6.3 Launching the Server

The framework is split into two main applications - an EvaluatorServer and an EvaluatorClient.

The EvaluatorServer is the main object which will perform any and all calculations needed to estimate sets of properties. It is design to run on whichever compute resources you may have available (whether that be a single machine or a high performance cluster), wait until a user requests a set of properties be estimated, and then handle that request.

The EvaluatorClient is the object used by the user to send requests to estimate data sets to running server instances over a TCP connection. It is also used to query the server to see when that request has been fulfilled, and to pull back any results.

Let us begin by spawning a new server instance.

To launch a server, we need to define how this object is going to interact with the compute resource it is running on.

This is accomplished using a *calculation backend*. While there are several to choose from depending on your needs, well will go with a simple dask based one designed to run on a single machine:

```
from openff.evaluator.backends import ComputeResources
from openff.evaluator.backends.dask import DaskLocalCluster

os.environ["CUDA_VISIBLE_DEVICES"] = "0"

calculation_backend = DaskLocalCluster(
    number_of_workers=1,
    resources_per_worker=ComputeResources(
        number_of_threads=1,
        number_of_gpus=1,
        preferred_gpu_toolkit=ComputeResources.GPUToolkit.CUDA,
    ),
)
calculation_backend.start()
```

Here we have specified that we want to run our calculations on a single worker which has access to a single GPU.

With that defined, we can go ahead and spin up the server:

```
[]: from openff.evaluator.server import EvaluatorServer

evaluator_server = EvaluatorServer(calculation_backend=calculation_backend)
    evaluator_server.start(asynchronous=True)
```

The server will run asynchronously in the background waiting until a client connects and requests that a data set be estimated.

# 2.6.4 Estimating the Data Set

With the server spun up we can go ahead and connect to it using an EvaluatorClient and request that it estimate our data set using the custom options we defined earlier:

```
[]: from openff.evaluator.client import EvaluatorClient
    evaluator_client = EvaluatorClient()

request, exception = evaluator_client.request_estimate(
    property_set=data_set,
    force_field_source=force_field_source,
    options=estimation_options,
)

assert exception is None
```

The server will now receive the requests and begin whirring away fulfilling it. It should be noted that the request\_estimate() function returns two values - a request object, and an exception object. If all went well (as it should do here) the exception object will be None.

The request object represents the request which we just sent to the server. It stores the unique id which the server assigned to the request, as well as the address of the server that the request was sent to.

The request object is primarily used to query the current state of our request, and to pull down the results when it the request finishes. Here we will use it it synchronously query the server every 30 seconds until our request has completed.

```
[]: # Wait for the results.
results, exception = request.results(synchronous=True, polling_interval=30)
assert exception is None
```

Note: we could also asynchronously query for the results of the request. The resultant results object would then contain the partial results of any completed estimates, as well as any exceptions raised during the estimation.

# 2.6.5 Inspecting the Results

Now that the server has finished estimating our data set and returned the results to us, we can begin to inspect the results of the calculations:

```
[]: print(len(results.queued_properties))
    print(len(results.estimated_properties))
    print(len(results.unsuccessful_properties))
    print(len(results.exceptions))
```

We can (hopefully) see here that there were no exceptions raised during the calculation, and that all of our properties were successfully estimated.

We will extract the estimated data set and save this to disk:

```
[ ]: results.estimated_properties.json("estimated_data_set.json", format=True);
```

### 2.6.6 Conclusion

And that concludes the second tutorial. In the next tutorial we will be performing some basic analysis on our estimated results.

If you have any questions and / or feedback, please open an issue on the GitHub issue tracker.

# 2.7 Tutorial 03 - Analysing Data Sets

In this tutorial we will be analysing the results of the calculations which we performed in the *second tutorial*. The tutorial will cover:

- comparing the estimated data set with the experimental data set.
- plotting the two data sets.

Note: If you are running this tutorial in google colab you will need to run a setup script instead of following the installation instructions:

For the sake of clarity all warnings will be disabled in this tutorial:

```
[]: import warnings
warnings.filterwarnings("ignore")
import logging
logging.getLogger("openff.toolkit").setLevel(logging.ERROR)
```

### 2.7.1 Loading the Data Sets

We will begin by loading both the experimental data set and the estimated data set:

```
[]: import pathlib
from openff.evaluator.datasets import PhysicalPropertyDataSet

experimental_data_set_path = "filtered_data_set.json"
estimated_data_set_path = "estimated_data_set.json"

# If you have not yet completed the previous tutorials or do not have the data set files
# available, this tutorial will use copies provided by the framework

if not (
    pathlib.Path(experimental_data_set_path).exists()
    and pathlib.Path(estimated_data_set_path).exists()
):
    from openff.evaluator.utils import get_data_filename
```

(continues on next page)

(continued from previous page)

```
experimental_data_set_path = get_data_filename(
     "tutorials/tutorial01/filtered_data_set.json"
)
    estimated_data_set_path = get_data_filename(
        "tutorials/tutorial02/estimated_data_set.json"
)

experimental_data_set = PhysicalPropertyDataSet.from_json(experimental_data_set_path)
estimated_data_set = PhysicalPropertyDataSet.from_json(estimated_data_set_path)
```

if everything went well from the previous tutorials, these data sets will contain the density and  $H_{vap}$  of ethanol and isopropanol:

```
[]: experimental_data_set.to_pandas().head()

[]: estimated_data_set.to_pandas().head()
```

# 2.7.2 Extracting the Results

We will now compare how the value of each property estimated by simulation deviates from the experimental measurement.

To do this we will extract a list which contains pairs of experimental and evaluated properties. We can easily match properties based on the unique ids which were automatically assigned to them on their creation:

# 2.7.3 Plotting the Results

We will now compare the experimental results to the estimated ones by plotting them using matplotlib:

```
[]: from matplotlib import pyplot

# Create the figure we will plot to.
figure, axes = pyplot.subplots(nrows=1, ncols=2, figsize=(8.0, 4.0))

(continues on next page)
```

(continued from previous page)

```
# Set the axis titles
axes[0].set_xlabel("OpenFF 1.0.0")
axes[0].set_ylabel("Experimental")
axes[0].set_title("Density $kg m^{-3}$")
axes[1].set_xlabel("OpenFF 1.0.0")
axes[1].set_ylabel("Experimental")
axes[1].set_title("H_{vap} $kJ mol^{-1}$")
# Define the preferred units of the properties
from openff.units import unit
preferred_units = {
    "Density": unit.kilogram / unit.meter**3,
    "EnthalpyOfVaporization": unit.kilojoule / unit.mole,
}
for index, property_type in enumerate(properties_by_type):
    experimental_values = []
    estimated_values = []
   preferred_unit = preferred_units[property_type]
    # Convert the values of our properties to the preferred units.
    for experimental_property, estimated_property in properties_by_type[property_type]:
        experimental_values.append(
            experimental_property.value.to(preferred_unit).magnitude
        )
        estimated_values.append(estimated_property.value.to(preferred_unit).magnitude)
   axes[index].plot(
        estimated_values, experimental_values, marker="x", linestyle="None"
```

#### 2.7.4 Conclusion

And that concludes the third tutorial!

If you have any questions and / or feedback, please open an issue on the GitHub issue tracker.

# 2.8 Tutorial 04 - Optimizing Force Fields

In this tutorial we will be using the OpenFF Evaluator framework in combination with the fantastic ForceBalance software to optimize a molecular force field against the physical property data set we created in the *first tutorial*.

ForceBalance offers a suite of tools for optimizing molecular force fields against a set of target data. Perhaps one of the most fundamental targets to fit against is experimental physical property data. Physical property data has been used extensively for decades to inform the values of non-bonded Van der Waals (VdW) interaction parameters (often referred to as Lennard-Jones parameters).

*ForceBalance* is seamlessly integrated with the evaluator framework, using it to evaluate the deviations between target experimentally measured data points and those evaluated using the force field being optimized (as well as the gradient of those deviations with respect to the force field parameters being optimized).

The tutorial will cover:

- setting up the input files and directory structure required by ForceBalace.
- setting up an EvaluatorServer for *ForceBalance* to connect to.
- running ForceBalance using those input files.
- extracting and plotting a number of statistics output during the optimization.

Note: If you are running this tutorial in google colab you will need to run a setup script instead of following the installation instructions:

For this tutorial make sure that you are using a GPU accelerated runtime.

For the sake of clarity all warnings will be disabled in this tutorial:

```
[]: import warnings
warnings.filterwarnings("ignore")
import logging
logging.getLogger("openff.toolkit").setLevel(logging.ERROR)
```

We will also enable time-stamped logging to help track the progress of our calculations:

```
[]: from openff.evaluator.utils import setup_timestamp_logging
setup_timestamp_logging()
```

# 2.8.1 Setting up the ForceBalance Inputs

In this section we will be creating the directory structure required by *ForceBalance*, and populating it with the required input files.

### **Creating the Directory Structure**

To begin with, we will create a directory to store the starting force field parameters in:

```
[]:!mkdir forcefield
```

and one to store the input parameters for our 'fitting target' - in this case a data set of physical properties:

```
[]:!mkdir -p targets/pure_data
```

### **Defining the Training Data Set**

With the directories created, we will next specify the data set of physical properties which we will be training the force field against:

```
[]: # For convenience we will use the copy shipped with the framework
from openff.evaluator.utils import get_data_filename

data_set_path = get_data_filename("tutorials/tutorial01/filtered_data_set.json")

# Load the data set.
from openff.evaluator.datasets import PhysicalPropertyDataSet

data_set = PhysicalPropertyDataSet.from_json(data_set_path)

# Due to a small bug in ForceBalance we need to zero out any uncertainties
# which are undefined. This will be fixed in future versions.
from openff.evaluator.attributes import UNDEFINED

for physical_property in data_set:
    if physical_property.uncertainty != UNDEFINED:
        continue

    physical_property.uncertainty = 0.0 * physical_property.default_unit()
```

To speed up the runtime of this tutorial, we will only train the force field against measurements made for ethanol

```
[]: from openff.evaluator.datasets.curation.components.filtering import (
    FilterBySmiles,
    FilterBySmilesSchema,
)

data_set = FilterBySmiles.apply(
    data_set,
    FilterBySmilesSchema(smiles_to_include=["CCO"]),
)
```

in real optimizations however the data set should be **much** larger than two data points!

With those changes made, we can save the data set in our targets directory:

```
[]: # Store the data set in the `pure_data` targets folder:
   data_set.json("targets/pure_data/training_set.json");
```

#### **Defining the Starting Force Field Parameters**

We will use the OpenFF Parsley 1.0.0 force field as the starting parameters for the optimization. These can be loaded directly into an OpenFF ForceField object using the OpenFF toolkit:

```
[]: from openff.toolkit.typing.engines.smirnoff import ForceField

force_field = ForceField("openff-1.0.0.offxml")
```

In order to use these parameters in *ForceBalance*, we need to 'tag' the individual parameters in the force field that we wish to optimize. The toolkit easily enables us to add these tags using cosmetic attributes:

```
[]: # Extract the smiles of all unique components in our data set.
from openff.toolkit.topology import Molecule, Topology

all_smiles = {
    component.smiles
    for substance in data_set.substances
    for component in substance.components
}

for smiles in all_smiles:
    # Find those VdW parameters which would be applied to those components.
    molecule = Molecule.from_smiles(smiles)
    topology = Topology.from_molecules([molecule])

labels = force_field.label_molecules(topology)[0]

# Tag the exercised parameters as to be optimized.
for parameter in labels["vdW"].values():
    parameter.add_cosmetic_attribute("parameterize", "epsilon, rmin_half")
```

Here we have made use of the toolkit's handy label\_molecules function to see which VdW parameters will be assigned to the molecules in our data set, and tagged them to be parameterized.

With those tags added, we can save the parameters in the forcefield directory:

```
[]: # Save the annotated force field file.
force_field.to_file("forcefield/openff-1.0.0-tagged.offxml")
```

Note: The force field parameters are stored in the OpenFF SMIRNOFF XML format.

#### **Creating the Main Input File**

Next, we will create the main *ForceBalance* input file. For the sake of brevity a default input file which ships with this framework will be used:

```
[ ]: input_file_path = get_data_filename("tutorials/tutorial04/optimize.in")

# Copy the input file into our directory structure
import shutil

shutil.copyfile(input_file_path, "optimize.in")
```

While there are many options that can be set within this file, the main options of interest for our purposes appear at the bottom of the file:

```
[]:!tail -n 6 optimize.in
```

Here we have specified that we wish to create a new *ForceBalance* Evaluator\_SMIRNOFF target called pure\_data (corresponding to the name of the directory we created in the earlier step).

The main input to this target is the file path to an options. json file - it is this file which will specify all the options which should be used when *ForceBalance* requests that our target data set be estimated using the current sets of force

field parameters.

We will create this file in the targets/pure\_data directory later in this section.

The data set is the JSON serialized representation of the PhysicalPropertyDataSet we created during the *first* tutorial.

#### **Defining the Estimation Options**

The final step before we can start the optimization is to create the set of options which will govern how our data set is estimated using the Evaluator framework.

These options will be stored in an Evaluator\_SMIRNOFF object:

```
[]: from forcebalance.evaluator_io import Evaluator_SMIRNOFF

# Create the ForceBalance options object
target_options = Evaluator_SMIRNOFF.OptionsFile()
# Set the path to the data set
target_options.data_set_path = "training_set.json"
```

This object exposes both a set of ForceBalance specific options, as well as the set of Evaluator options.

The *ForceBalance* specific options allow us to define how each type of property will contribute to the optimization objective function (the value which we are trying to minimize):

$$\Delta(\theta) = \sum_{n=1}^{N} \frac{weight_n}{M_n} \sum_{m=1}^{M_n} \left( \frac{y_m^{ref} - y_m(\theta)}{denominator_n} \right)^2$$

where N is the number of types of properties (e.g. density, enthalpy of vaporization, etc.),  $M_n$  is the number of data points of type n,  $y_m^{ref}$  is the experimental value of data point m and  $y_m(\theta)$  is the estimated value of data point m using the current force field parameters

In particular, the options object allows us to specify both an amount to scale each type of properties contribution to the objective function by  $(weight_n)$ , and the amount to scale the difference between the experimental and estimated properties  $(denominator_n)$ :

```
[]: from openff.units import unit

target_options.weights = {"Density": 1.0, "EnthalpyOfVaporization": 1.0}
target_options.denominators = {
    "Density": 30.0 * unit.kilogram / unit.meter**3,
    "EnthalpyOfVaporization": 3.0 * unit.kilojoule / unit.mole,
}
```

where here we have chosen values that ensure that both types of properties contribute roughly equally to the total objective function.

The Evaluator specific options correspond to a standard RequestOptions object:

```
[]: from openff.evaluator.client import RequestOptions

# Create the options which evaluator should use.
evaluator_options = RequestOptions()
# Choose which calculation layers to make available.
evaluator_options.calculation_layers = ["SimulationLayer"]
```

(continues on next page)

(continued from previous page)

```
# Reduce the default number of molecules
from openff.evaluator.properties import Density, EnthalpyOfVaporization

density_schema = Density.default_simulation_schema(n_molecules=256)
h_vap_schema = EnthalpyOfVaporization.default_simulation_schema(n_molecules=256)

evaluator_options.add_schema("SimulationLayer", "Density", density_schema)
evaluator_options.add_schema("SimulationLayer", "EnthalpyOfVaporization", h_vap_schema)
target_options.estimation_options = evaluator_options
```

These options allow us to control exactly how each type of property should be estimated, which calculation approaches should be used and more. Here we use the same options are were used in the *second tutorial* 

Note: more information about the different estimation options can be found here

And that's the options created! We will finish off by serializing the options into our target directory:

```
[]: # Save the options to file.
with open("targets/pure_data/options.json", "w") as file:
    file.write(target_options.to_json())
```

# 2.8.2 Launching an Evaluator Server

With the *ForceBalance* options created, we can now move onto launching the EvaluatorServer which *ForceBalance* will call out to when it needs the data set to be evaluated:

```
[]: import os
    # Launch the calculation backend which will distribute any calculations.
    from openff.evaluator.backends import ComputeResources
    from openff.evaluator.backends.dask import DaskLocalCluster
    os.environ["CUDA_VISIBLE_DEVICES"] = "0"
    calculation_backend = DaskLocalCluster(
        number_of_workers=1,
        resources_per_worker=ComputeResources(
            number_of_threads=1,
            number_of_gpus=1,
            preferred_gpu_toolkit=ComputeResources.GPUToolkit.CUDA,
        ),
    calculation_backend.start()
    # Launch the server object which will listen for estimation requests and schedule any
    # required calculations.
    from openff.evaluator.server import EvaluatorServer
    evaluator_server = EvaluatorServer(calculation_backend=calculation_backend)
    evaluator_server.start(asynchronous=True)
```

We will not go into the details of this here as this was already covered in the second tutorial

# 2.8.3 Running ForceBalance

With the inputs created and an Evaluator server spun up, we are finally ready to run the optimization! This can be accomplished with a single command:

```
[ ]: !ForceBalance optimize.in
```

If everything went well *ForceBalance* should exit cleanly, and will have stored out newly optimized force field in the results directory.

```
[]: !ls result/optimize
```

# 2.8.4 Plotting the results

As a last step in this tutorial, we will extract the objective function at each iteration from the *ForceBalance* output files and plot this using matplotlib.

First, we will extract the objective function from the pickle serialized output files which can be found in the optimize.tmp/pure\_data/iter\_\*\*\*\*/ directories:

```
[]: # Determine how many iterations ForceBalance has completed.
from glob import glob

from forcebalance.nifty import lp_load

n_iterations = len(glob("optimize.tmp/pure_data/iter*"))

# Extract the objective function at each iteration.
objective_function = []

for iteration in range(n_iterations):
    folder_name = "iter_" + str(iteration).zfill(4)
    file_path = f"optimize.tmp/pure_data/{folder_name}/objective.p"

    statistics = lp_load(file_path)
    objective_function.append(statistics["X"])

print(objective_function)
```

The objective function is then easily plotted:

```
[]: from matplotlib import pyplot

figure, axis = pyplot.subplots(1, 1, figsize=(4, 4))

axis.set_xlabel("Iteration")
axis.set_ylabel("Objective Function")

axis.plot(range(n_iterations), objective_function, marker="o")

figure.tight_layout()
```

### 2.8.5 Conclusion

And that concludes the fourth tutorial!

If you have any questions and / or feedback, please open an issue on the GitHub issue tracker.

# 2.9 Property Data Sets

A *PhysicalPropertyDataSet* is a collection of measured physical properties encapsulated as *physical property* objects. They may be created from scratch:

```
# Define a density measurement
density = Density(
    substance=Substance.from_components("0"),
    thermodynamic_state=ThermodynamicState(
        pressure=1.0*unit.atmospheres, temperature=298.15*unit.kelvin
    ),
    phase=PropertyPhase.Liquid,
    value=1.0*unit.gram/unit.millilitre,
    uncertainty=0.0001*unit.gram/unit.millilitre
)

# Add the property to a data set
data_set = PhysicalPropertyDataset()
data_set.add_properties(density)
```

are readily JSON (de)serializable:

```
# Save the data set as a JSON file.
data_set.json(file_path="data_set.json", format=True)
# Load the data set from a JSON file
data_set = PhysicalPropertyDataset.from_json(file_path="data_set.json")
```

and may be converted to pandas DataFrame objects:

```
data_set.to_pandas()
```

The framework implements specific data set objects for extracting data measurements directly from a number of open data sources, such as the *ThermoMLDataSet* (see *ThermoML Archive*) which provides utilities for extracting the data from the NIST ThermoML Archive and converting it into the standard framework objects.

Data set objects are directly iterable:

```
for physical_property in data_set:
...
```

or can be iterated over for a specific substance:

```
for physical_property in data_set.properties_by_substance(substance):
    ...
```

or for a specific type of property:

```
for physical_property in data_set.properties_by_type("Density"):
    ...
```

# 2.9.1 Physical Properties

The *PhysicalProperty* object is a base class for any object which describes a measured property of substance, and is defined by a combination of:

- the observed value of the property.
- Substance specifying the substance that the measurement was collected for.
- PropertyPhase specifying the phase that the measurement was collected in.
- *ThermodynamicState* specifying the thermodynamic conditions under which the measurement was performed

as well as optionally

- the uncertainty in the value of the property.
- a list of *ParameterGradient* which defines the gradient of the property with respect to the model parameters if it was computationally estimated.
- a Source specifying the source (either experimental or computational) and provenance of the measurement.

Each type of property supported by the framework, such as a density of an enthalpy of vaporization, must have it's own class representation which inherits from *PhysicalProperty*:

```
# Define a density measurement
density = Density(
    substance=Substance.from_components("0"),
    thermodynamic_state=ThermodynamicState(
        pressure=1.0*unit.atmospheres, temperature=298.15*unit.kelvin
    ),
    phase=PropertyPhase.Liquid,
    value=1.0*unit.gram/unit.millilitre,
    uncertainty=0.0001*unit.gram/unit.millilitre
)
```

#### 2.9.2 Substances

A *Substance* is defined by a number of components (which may have specific roles assigned to them such as being solutes in the system) and the amount of each component in the substance.

To create a pure substance containing only water:

```
water_substance = Substance.from_components("0")
```

To create binary mixture of water and methanol in a 20:80 ratio:

```
binary_mixture = Substance()
binary_mixture.add_component(Component(smiles="0"), MoleFraction(value=0.2))
binary_mixture.add_component(Component(smiles="CO"), MoleFraction(value=0.8))
```

To create a substance of an infinitely dilute paracetamol solute dissolved in water:

# 2.9.3 Property Phases

The *PropertyPhase* enum describes the possible phases which a measurement was performed in.

While the enum only has three defined phases (Solid, Liquid and Gas), multiple phases can be formed by OR'ing (|) multiple phases together. As an example, to define a phase for a liquid and gas coexisting:

```
liquid_gas_phase = PropertyPhase.Liquid | PropertyPhase.Gas
```

# 2.9.4 Thermodynamic States

A *ThermodynamicState* specifies a combination of the temperature and (optionally) the pressure at which a measurement is performed:

```
thermodynamic_state = ThermodynamicState(
    temperature=298.15*unit.kelvin, pressure=1.0*unit.atmosphere
)
```

# 2.10 ThermoML Archive

The *ThermoMLDataSet* object offers an API for extracting physical properties from the NIST ThermoML Archive, both directly from the archive itself or from files stored in the IUPAC- standard ThermoML format.

The API only supports extracting those properties which have been *registered* with the frameworks plug-in system, and does not currently load the full set of metadata available in the archive files.

**Note:** If the metadata you require is not currently exposed, please open an issue on the GitHub issue tracker to request it.

Currently the framework has built-in support for extracting:

- Mass density, kg/m3 (Density)
- Excess molar volume, m3/mol (ExcessMolarVolume)
- Relative permittivity at zero frequency (DielectricConstant)
- Excess molar enthalpy (molar enthalpy of mixing), kJ/mol (EnthalpyOfMixing)
- Molar enthalpy of vaporization or sublimation, kJ/mol (EnthalpyOfVaporization)

where here both the ThermoML property name (as defined by the IUPAC XML schema) and the built-in framework class are listed.

# 2.10.1 Registering Properties

Properties to be extracted from ThermoML archives must have a corresponding class representation to be loading into. This class representation must both:

- inherit from the frameworks *PhysicalProperty* class and
- be registered with the frameworks plug-in system using either the thermoml\_property() decorator or the register\_thermoml\_property() method.

As an example, a class representation of the ThermoML 'Mass density, kg/m3' property could be defined and registered with the plug-in system using:

```
@thermoml_property("Mass density, kg/m3", supported_phases=PropertyPhase.Liquid)
class Density(PhysicalProperty):
    """A class representation of a mass density property"""
```

The thermoml\_property() decorator takes in the name of the ThermoML property (as defined by the IUPAC schema) as well as the phases which the framework will be able to estimate this property in.

Multiple ThermoML properties can be mapped onto a single class using the flexible <code>register\_thermoml\_property()</code> function. For example, the 'Specific volume, m3/kg' property (which is simply the reciprocal of mass density) may be mapped onto the <code>Density</code> object by providing a <code>conversion\_function</code>:

```
def specific_volume_to_mass_density(specific_volume):
    """Converts a specific volume measurement into a mass
    density.
    Parameters
    specific_volume: ThermoMLProperty
        The specific volume measurement to convert.
   mass_density = Density()
   mass_density.value = 1.0 / specific_volume.value
   if mass_density.uncertainty is not None:
        mass_density.uncertainty = 1.0 / mass_density.uncertainty
   mass_density.phase = specific_volume.phase
   mass_density.thermodynamic_state = specific_volume.thermodynamic_state
   mass_density.substance = specific_volume.substance
   return mass_density
# Register the ThermoML property using the conversion function.
register_thermoml_property(
    thermoml_string="Specific volume, m3/kg",
    supported_phases=PropertyPhase.Liquid,
   property_class=Density,
    conversion_function=specific_volume_to_mass_density
)
```

Converting the different density derivatives into a single density class removes the need to produce many very similar

class representations of density measurements, and allows a single calculation schema to be defined for all variants.

# 2.10.2 Loading Data Sets

Data sets are most easily loaded using their digital object identifiers (DOI). For example, to retrieve the ThermoML data set that accompanies this paper, we can simply use the DOI 10.1016/j.jct.2005.03.012:

```
data_set = ThermoMLDataset.from_doi('10.1016/j.jct.2005.03.012')
```

Data can be pulled from multiple sources at once by specifying multiple identifiers:

```
identifiers = ['10.1021/acs.jced.5b00365', '10.1021/acs.jced.5b00474']
dataset = ThermoMLDataset.from_doi(*identifiers)
```

Entire archives of properties can be downloaded directly from the ThermoML website and parsed by the framework. For example, to create a data set object containing all of the measurements recorded from the International Journal of Thermophysics:

```
# Download the archive of all properties from the IJT journal.
import requests
request = requests.get("https://trc.nist.gov/ThermoML/IJT.tgz", stream=True)
# Make sure the request went ok.
assert request
# Unzip the files into a new 'ijt_files' directory.
import io, tarfile
tar_file = tarfile.open(fileobj=io.BytesIO(request.content))
tar_file.extractall("ijt_files")
# Get the names of the extracted files
import glob
file_names = glob.glob("ijt_files/*.xml")
# Create the data set object
from openff.evaluator.datasets.thermoml import ThermoMLDataSet
data_set = ThermoMLDataSet.from_file(*file_names)
# Save the data set to a JSON object
data_set.json(file_path="ijt.json", format=True)
```

# 2.11 Taproom

The *TaproomDataSet* object offers an API for retrieving host-guest binding affinity measurements from the curated taproom repository.

Note: taproom may be installed by running conda install -c conda-forge taproom

This includes retrieving all of the data available:

2.11. Taproom 33

```
from openff.evaluator.datasets.taproom import TaproomDataSet
taproom_set = TaproomDataSet()
```

data measure for a single host molecule (e.g. alpha-cyclodextrin):

```
acd_taproom_set = TaproomDataSet(host_codes=["acd"])
```

or data for a particular host and guest pair:

```
acd_taproom_set = TaproomDataSet(host_codes=["acd"], guest_codes=["bam"])
```

All measurements in this data set have an associated *TaproomSource* as their source provenance. This tracks both the original source of the measurement as well as the taproom identifier.

**Note:** Currently the data set object will assume a default set of buffer conditions (either no buffer, or a buffer of a salt with a specified ionic strength) rather than reading the buffer from the taproom measurement directory. This is consistent with previous applications of the data set.

## 2.12 Data Set Curation

The framework offers a full suite of features to facilitate the curation of data sets of physical properties, including:

a significant amount of data filters, including to filter by state, substance composition and chemical functionalities.

and components to

- easily download and import the full NIST ThermoML and FreeSolv archives .
- select data points which were measured close to a set of target states, and which were measured for a diverse range of substances which contain specific functionalities.
- convert between different compatible property types (e.g. convert density <-> excess molar volume data).

These features are implemented as *CurationComponent* objects, which take as input an associated *CurationComponentSchema* which controls how the curation components should be applied to a particular data set (or a data set which is being stored as pandas DataFrame object).

An example of a curation component would be one that filters out data points which were measured outside of a particular temperature range:

```
# Filter data points measured at less than 290.0 K or greater than 320.0 K
filtered_frame = FilterByTemperature.apply(
    data_frame,
    FilterByTemperatureSchema(minimum_temperature=290.0, maximum_temperature=320.0),
)
```

Curation components can be conveniently chained together using a *CurationWorkflow* and an associated *CurationWorkflowSchema* so as to easily curated full training and testing data sets:

```
# Filter out any measurements made for systems with more than two components
        filtering.FilterByNComponentsSchema(n_components=[1, 2]),
        # Remove any duplicate data.
        filtering.FilterDuplicatesSchema(),
        # Filter out data points measured away from ambient
        # and biologically relevant temperatures.
        filtering.FilterByTemperatureSchema(
            minimum_temperature=298.0, maximum_temperature=320.0
        # Retain only density and enthalpy of mixing data points.
        filtering.FilterByPropertyTypesSchema(
            property_types=["Density", "EnthalpyOfMixing"],
        ),
        # Select data points measured for alcohols, esters or mixtures of both.
        selection.SelectSubstancesSchema(
            target_environments=[
                ChemicalEnvironment.Alcohol,
                ChemicalEnvironment.CarboxylicAcidEster,
            ],
            n_per_environment=10,
        ),
   ]
)
data_frame = Workflow.apply(pandas.DataFrame(), curation)
```

## 2.12.1 Examples

## **Data Extraction**

• *ImportFreeSolv*: A component which will download the latest, full FreeSolv data set from the GitHub repository:

```
from openff.evaluator.datasets.curation.components.freesolv import (
    ImportFreeSolv,
    ImportFreeSolvSchema,
)

# Import the full FreeSolv data set.
data_frame = ImportFreeSolv.apply(pandas.DataFrame(), ImportFreeSolvSchema())
```

 ImportThermoMLData: A component which will download all supported data from the NIST ThermoML Archive:

#### **Filtration**

• FilterDuplicates: A component to remove duplicate data points (within a specified precision) from a data set:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterDuplicates,
   FilterDuplicatesSchema,
)
filtered_frame = FilterDuplicates.apply(data_frame, FilterDuplicatesSchema())
```

• FilterByTemperature: A component which will filter out data points which were measured outside of a specified temperature range:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByTemperature,
   FilterByTemperatureSchema,
)

filtered_frame = FilterByTemperature.apply(
   data_frame,
   FilterByTemperatureSchema(minimum_temperature=290.0, maximum_temperature=320.0),
)
```

• *FilterByPressure*: A component which will filter out data points which were measured outside of a specified pressure range:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByPressure,
   FilterByPressureSchema,
)
filtered_frame = FilterByPressure.apply(
   data_frame,
   FilterByPressureSchema(minimum_pressure=100.0, maximum_pressure=140.0),
)
```

• FilterByMoleFraction: A component which will filter out data points which were measured outside of a specified mole fraction range:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByMoleFraction,
   FilterByMoleFractionSchema,
)

filtered_frame = FilterByMoleFraction.apply(
   data_frame, FilterByMoleFractionSchema(mole_fraction_ranges={2: [[(0.1, 0.3)]]})
)
```

• FilterByRacemic: A component which will filter out data points which were measured for racemic mixtures:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByRacemic,
   FilterByRacemicSchema,
)
filtered_frame = FilterByRacemic.apply(data_frame, FilterByRacemicSchema())
```

• *FilterByElements*: A component which will filter out data points which were measured for systems which contain specific elements:

```
from openff.evaluator.datasets.curation.components.filtering import (
    FilterByElements,
    FilterByElementsSchema,
)

filtered_frame = FilterByElements.apply(
    data_frame,
    FilterByElementsSchema(allowed_elements=["C", "O", "H"]),
)
```

• FilterByPropertyTypes: A component which will apply a filter which only retains properties of specified types:

```
from openff.evaluator.datasets.curation.components.filtering import (
    FilterByPropertyTypes,
    FilterByPropertyTypesSchema,
)

# Retain only density measurements made for either pure or binary systems.
filtered_frame = FilterByPropertyTypes.apply(
    data_frame,
    FilterByPropertyTypesSchema(
        property_types=["Density"],
        n_components={"Density": [1, 2]},
    ),
)
```

• *FilterByStereochemistry*: A component which filters out data points measured for systems whereby the stereochemistry of a number of components is undefined:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByStereochemistry,
   FilterByStereochemistrySchema,
)

filtered_frame = FilterByStereochemistry.apply(
   data_frame, FilterByStereochemistrySchema()
)
```

• FilterByCharged: A component which filters out data points measured for substance where any of the constituent components have a net non-zero charge.:

2.12. Data Set Curation 37

```
FilterByChargedSchema,
)
filtered_frame = FilterByCharged.apply(data_frame, FilterByChargedSchema())
```

• FilterByIonicLiquid: A component which filters out data points measured for substances which contain or are classed as an ionic liquids:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByIonicLiquid,
   FilterByIonicLiquidSchema,
)
filtered_frame = FilterByIonicLiquid.apply(data_frame, FilterByIonicLiquidSchema())
```

• *FilterBySmiles*: A component which filters the data set so that it only contains either a specific set of smiles, or does not contain any of a set of specifically excluded smiles:

```
from openff.evaluator.datasets.curation.components.filtering import (
    FilterBySmiles,
    FilterBySmilesSchema,
)

filtered_frame = FilterBySmiles.apply(
    data_frame, FilterBySmilesSchema(smiles_to_include=["CCCO"]),
)
```

• *FilterBySmirks*: A component which filters a data set so that it only contains measurements made for molecules which contain (or don't) a set of chemical environments represented by SMIRKS patterns:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterBySmirks,
   FilterBySmirksSchema,
)

filtered_frame = FilterBySmirks.apply(
   data_frame, FilterBySmirksSchema(smirks_to_include=["[#6a]"]),
)
```

• FilterByNComponents: A component which filters out data points measured for systems with specified number of components:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByNComponents,
   FilterByNComponentsSchema,
)

filtered_frame = FilterByNComponents.apply(
   data_frame, FilterByNComponentsSchema(n_components=[1, 2])
)
```

• *FilterBySubstances*: A component which filters the data set so that it only contains properties measured for particular substances:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterBySubstances,
   FilterBySubstancesSchema,
)

filtered_frame = FilterBySubstances.apply(
   data_frame, FilterBySubstancesSchema(substances_to_include=[("CO", "C")])
)
```

• FilterByEnvironments: A component which filters a data set so that it only contains measurements made for substances which contain specific chemical environments:

### **Data Selection**

• SelectSubstances: A component for selecting data points which were measured for specified number of maximally diverse systems containing a specified set of chemical functionalities:

```
# Select (if possible) data points which were measured for 10 different (and
# structurally diverse) alcohols.
schema = SelectSubstancesSchema(
    target_environments=[ChemicalEnvironment.Alcohol],
    n_per_environment=10,
)
data_frame = ConvertExcessDensityData.apply(data_frame, schema)
```

• SelectDataPoints: A component for selecting a set of data points which are close to a particular set of states:

(continues on next page)

### **Data Conversion**

• *ConvertExcessDensityData*: A component for converting binary mass density data to excess molar volume data and vice versa where pure density data measured for the components is available:

```
from openff.evaluator.datasets.curation.components.conversion import (
    ConvertExcessDensityData,
    ConvertExcessDensityDataSchema,
)

converted_data_frame = ConvertExcessDensityData.apply(
    data_frame, ConvertExcessDensityDataSchema()
)
```

# 2.13 Physical Properties

A core philosophy of this framework is that users should be able to seamlessly curate data sets of physical properties and then estimate that data set using computational methods without significant user intervention and using sensible, well validated workflows.

This page aims to provide an overview of which physical properties are supported by the framework and how they are computed using the different *calculation layers*.

In this document  $\langle X \rangle$  will be used to denote the ensemble average of an observable X.

## **2.13.1 Density**

The density  $(\rho)$  is computed according to

$$\rho = \left\langle \frac{M}{V} \right\rangle$$

where M and V are the total molar mass and volume the system respectively.

#### **Direct Simulation**

The density is estimated using the default *simulation workflow* without modification. The estimation of liquid densities is assumed.

### **MBAR** Reweighting

The density is estimated using the default *reweighting workflow* without modification. The estimation of liquid densities is assumed.

### 2.13.2 Dielectric Constant

The dielectric constant  $(\varepsilon)$  is computed from the fluctuations in a systems dipole moment (see Equation 7 of [1]) according to:

$$\varepsilon = 1 + \frac{\langle \vec{\mu}^2 \rangle - \langle \vec{\mu} \rangle^2}{3\varepsilon_0 \langle V \rangle k_b T}$$

where  $\vec{\mu}$ , V are the systems dipole moment and volume respectively,  $k_b$  the Boltzmann constant, T the temperature, and  $\varepsilon_0$  the permittivity of free space.

**Note:** In v0.2.2 and earlier of the framework the variance was computed as  $\langle (\vec{\mu} - \langle \vec{\mu} \rangle)^2 \rangle$  in order to match the mdtraj implementation which has been used in previous studies by the OpenFF Consortium (see for example [2]). The two approaches should be numerically indistinguishable however.

#### **Direct Simulation**

The dielectric is estimated using the default *simulation workflow* which has been modified to use the specialized *AverageDielectricConstant* protocol in place of the default *AverageObservable* protocol. The estimation of liquid dielectric constants is assumed.

## **MBAR Reweighting**

The dielectric is estimated using the default *reweighting workflow* which has been modified to use the specialized *ReweightDielectricConstant* protocol in place of the default *ReweightObservable* protocol. It should be noted that the *ReweightDielectricConstant* protocol employs bootstrapping to compute the uncertainty in the average dielectric constant, rather than attempting to propagate uncertainties in the average dipole moments and volumes. The estimation of liquid dielectric constants is assumed.

## 2.13.3 Enthalpy of Vaporization

The enthalpy of vaporization  $\Delta H_{vap}$  (see [3]) can be computed according to

$$\Delta H_{vap} = \langle H_{gas} \rangle - \langle H_{liquid} \rangle = \langle E_{gas} \rangle - \langle E_{liquid} \rangle + p \left( \langle V_{gas} \rangle - \langle V_{liquid} \rangle \right)$$

where H, E, and V are the enthalpy, total energy and volume respectively.

Under the assumption that  $V_{gas} >> V_{liquid}$  and that the gas is ideal the above expression can be simplified to

$$\Delta H_{vap} = \langle U_{gas} \rangle - \langle U_{liquid} \rangle + RT$$

where U is the potential energy, T the temperature and R the universal gas constant. This simplified expression is computed by default by this framework.

#### **Direct Simulation**

- **Liquid phase**: The potential energy of the liquid phase is estimated using the default *simulation workflow*, and divided by the number of molecules in the simulation box using the divisor input of the *AverageObservable* protocol.
- Gas phase: The potential energy of the gas phase is estimated using the default *simulation workflow*, which has been modified so that
  - the simulation box only contains a single molecule.
  - all periodic boundary conditions have been disabled.
  - all simulations are performed in the NVT ensemble.
  - the production simulation is run for 15000000 steps at a time (rather than 1000000 steps).
  - all simulations are run using the OpenMM reference platform (CPU only) regardless of whether a GPU is available. This is fastest platform to use when simulating a single molecule in vacuum with OpenMM.

The final enthalpy is then computed by subtracting the gas potential energy from the liquid potential energy (SubtractValues) and adding the RT term (AddValues). Uncertainties are propagated through the subtraction by the normal means using the uncertainties package.

### **MBAR Reweighting**

- **Liquid phase**: The potential energy of the liquid phase is estimated using the default *reweighting workflow*, and divided by the number of molecules in the simulation box using an extra *DivideValue* protocol.
- Gas phase: The potential energy of the gas phase is estimated using the default *reweighting workflow*, which has been modified so that all periodic boundary conditions have been disabled.

The final enthalpy is then computed by subtracting the gas potential energy from the liquid potential energy (SubtractValues) and adding the RT term (AddValues). Uncertainties are propagated through the subtraction by the normal means using the uncertainties package.

# 2.13.4 Enthalpy of Mixing

The enthalpy of mixing  $\Delta H_{mix}\left(x_{0},\cdots,x_{M-1}\right)$  for a system of M components is computed according to

$$\Delta H_{mix}\left(x_{0}, \cdots, x_{M-1}\right) = \frac{\langle H_{mix} \rangle}{N_{mix}} - \sum_{i}^{M} x_{i} \frac{\langle H_{i} \rangle}{N_{i}}$$

where  $H_{mix}$  is the enthalpy of the full mixture, and  $H_i$ ,  $x_i$  are the enthalpy and the mole fraction of component i respectively.  $N_{mix}$  and  $N_i$  are the total number of molecules used in the full mixture simulations and the simulations of each individual component respectively.

When re-weighting cached data to compute  $H_{mix}$  we make the approximation that the kinetic energy contributions cancel out between the mixture and each of the components, and hence can be computed by only re-weighting the NPT reduced potential:

$$\Delta H_{mix}\left(x_{0}, \cdots, x_{M-1}\right) \approx \frac{1}{\beta} \left(\frac{\langle u_{mix} \rangle}{N_{mix}} - \sum_{i}^{M} x_{i} \frac{\langle u_{i} \rangle}{N_{i}}\right)$$

where  $u \equiv \beta (U + pV)$  is the NPT reduced potential, U the potential energy, p the pressure and V the volume.

### **Direct Simulation**

- **Mixture**: The enthalpy of the full mixture is estimated using the default *simulation workflow* and divided by the number of molecules in the simulation box using the divisor input of the *AverageObservable* protocol.
- **Components**: The enthalpy of each of the components is estimated using the default *simulation workflow*, divided by the number of molecules in the simulation box using the divisor input of the *AverageObservable* protocol, and weighted by their mole fraction *in the mixture simulation box* using the *WeightByMoleFraction* protocol.

The final enthalpy is then computed by summing the component enthalpies (*AddValues*) and subtracting these from the mixture enthalpy (*SubtractValues*). Uncertainties are propagated through the summation and subtraction by the normal means using the uncertainties package.

## **MBAR Reweighting**

- **Mixture**: The reduced potential of the full mixture is estimated using the default *reweighting workflow* and divided by the number of molecules in the reweighting box using an extra *DivideValue* protocol.
- **Components**: The reduced potential of each of the components is estimated using the default *reweighting work-flow*, divided by the number of molecules in the reweighting box using an extra *DivideValue* protocol, and weighted by their mole fraction using the *WeightByMoleFraction* protocol.

The final enthalpy is then computed by summing the component enthalpies (AddValues), subtracting these from the mixture enthalpy (SubtractValues), and multiplying by  $1/\beta$  (MultiplyValue). Uncertainties are propagated by the normal means using the uncertainties package.

### 2.13.5 Excess Molar Volume

The excess molar volume  $\Delta V_{excess}(x_0, \cdots, x_{M-1})$  for a system of M components is computed according to

$$\Delta V_{excess}\left(x_{0},\cdots,x_{M-1}\right) = N_{A}\left(\frac{\langle V_{mix}\rangle}{N_{mix}} - \sum_{i}^{M} x_{i} \frac{\langle V_{i}\rangle}{N_{i}}\right)$$

where  $V_{mix}$  is the volume of the full mixture, and  $V_i$ ,  $x_i$  are the volume and the mole fraction of component i respectively.  $N_{mix}$  and  $N_i$  are the total number of molecules used in the full mixture simulations and the simulations of each individual component respectively, and  $N_A$  is the Avogadro constant.

#### **Direct Simulation**

- **Mixture**: The molar volume of the full mixture is estimated using the default *simulation workflow* and divided by the molar number of molecules in the simulation box using the divisor input of the *AverageObservable* protocol.
- **Components**: The molar volume of each of the components is estimated using the default *simulation work-flow*, divided by the molar number of molecules in the simulation box using the divisor input of the *AverageObservable* protocol, and weighted by their mole fraction *in the mixture simulation box* using the *WeightByMoleFraction* protocol.

The final excess molar volume is then computed by summing the component molar volumes (*AddValues*) and subtracting these from the mixture molar volume (*SubtractValues*). Uncertainties are propagated through the summation and subtraction by the normal means using the uncertainties package.

### **MBAR Reweighting**

- **Mixture**: The enthalpy of the full mixture is estimated using the default *reweighting workflow* and divided by the molar number of molecules in the reweighting box using an extra *DivideValue* protocol.
- **Components**: The enthalpy of each of the components is estimated using the default *reweighting workflow*, divided by the molar number of molecules in the reweighting box using an extra *DivideValue* protocol, and weighted by their mole fraction using the *WeightByMoleFraction* protocol.

The final enthalpy is then computed by summing the component enthalpies (*AddValues*) and subtracting these from the mixture enthalpy (*SubtractValues*). Uncertainties are propagated through the summation and subtraction by the normal means using the uncertainties package.

## 2.13.6 Solvation Free Energies

Solvation free energies are currently computed using the Yank free energy package using direct molecular simulations. By default the calculations attempt to use 2000 solvent molecules, and the alchemical lambda spacings are selected using the built-in 'trailblazing' algorithm.

See the Yank documentation for more details.

# 2.13.7 Host-Guest Binding Free Energy

**Warning:** The computation of this property is still in beta. Users are heavily recommended to validate any calculations involving this property.

Host-guest binding free energies are currently computed using the attach-pull-release (APR) method [4] through integration with the pAPRika framework.

### 2.13.8 References

## 2.14 Common Workflows

As may be expected, most of the workflows used to estimate the physical properties within the framework make use of very similar workflows. This page aims to document the built-in 'template' workflows from which the more complex physical property estimation workflows are constructed.

### 2.14.1 Direct Simulation

Properties being estimated using the *direct simulation* calculation layer typically base their workflows off of the *generate\_simulation\_protocols()* template.

**Note:** This template currently assumes that a liquid phase property is being computed.

The workflow produced by this template proceeds as follows:

1) 1000 molecules are inserted into a simulation box with an approximate density of 0.95 g / mL using packmol (BuildCoordinatesPackmol).

- 2) the system is parameterized using either the *OpenFF toolkit*, *TLeap* or *LigParGen* depending on the force field being employed (*BuildSmirnoffSystem*, *BuildTLeapSystem* or *BuildLigParGenSystem*).
- 3) an energy minimization is performed using the default OpenMM energy minimizer (OpenMMEnergyMinimisation).
- 4) the system is equilibrated by running a short NPT simulation for 100000 steps using a timestep of 2 fs and using the OpenMM simulation engine (*OpenMMSimulation*).
- 5) while the uncertainty in the average observable is greater than the requested tolerance (if specified):
  - 5a) a longer NPT production simulation is run for 1000000 steps with a timestep of 2 fs and using the OpenMM simulation protocol (*OpenMMSimulation*) with its default Langevin integrator and Monte Carlo barostat.
  - 5b) the correlated samples are removed from the simulation outputs and the average value of the observable of interest and its uncertainty are computed by bootstrapping with replacement for 250 iterations (*AverageObservable*). See [1] for details of the decorrelation procedure.
  - 5c) steps 5a) and 5b) are repeated until the uncertainty condition (if applicable) is met.

The decorrelated simulation outputs are then made available ready to be cached by a *storage backend* (*DecorrelateObservables*, *DecorrelateTrajectory*).

# 2.14.2 MBAR Reweighting

Properties being estimated using the *MBAR reweighting* calculation layer typically base their workflows off of the *generate\_reweighting\_protocols()* template.

The workflow produced by this template proceeds as follows:

- 1) for each stored simulation data:
  - 1a) the cached data is retrieved from disk (UnpackStoredSimulationData)
- 2) the cached data from is concatenated together to form a single trajectory of configurations and observables (*ConcatenateTrajectories*, ConcatenateStatistics).
- 3) for each stored simulation data:
  - 3a) the system is parameterized using the force field parameters which were used when originally generating the cached data i.e. one of the reference states (BuildSmirnoffSystem, BuildTLeapSystem or BuildLigParGenSystem).
  - 3b) the reduced potential of each configuration in the concatenated trajectory is evaluated using the parameterized system (*OpenMMEvaluateEnergies*).
- 4) the system is parameterized using the force field parameters with which the property of interest should be calculated using i.e. of the target state (BuildSmirnoffSystem, BuildTLeapSystem or BuildLigParGenSystem) and the reduced potential of each configuration in the concatenated trajectory is evaluated using the parameterized system (OpenMMEvaluateEnergies).
  - 4a) (optional) if the observable of interest is a function of the force field parameters it is recomputed using the target state parameters. These recomputed values then replace the original concatenated observables loaded from the cached data.
- 5) the reference potentials, target potentials and the joined observables are sub-sampled to only retain equilibrated, uncorrelated samples (*AverageObservable*, *DecorrelateObservables*, *DecorrelateTrajectory*). See [1] for details of the decorrelation procedure.

6) the MBAR method is employed to compute the average value of the observable of interest and its uncertainty at the target state, taking the reference state reduced potentials as input. See [2] for the theory behind this approach. An exception is raised if there are not enough effective samples to reweight (*ReweightObservable*).

In more specialised cases the *generate\_base\_reweighting\_protocols()* template (which *generate\_reweighting\_protocols()* is built off of) is instead used due to its greater flexibility.

### 2.14.3 References

## 2.15 Gradients

A most fundamental feature of this framework is its ability to rapidly compute the gradients of physical properties with respect to the force field parameters used to estimate them.

**Note:** Prior to v0.3.0 of this framework a combination of re-weighting and the central finite difference was employed to estimate the gradients of observables. From v0.3.0 onwards the fluctuation method [1] is instead used. The change was made to, in future, enable better integration with automatic differentiation libraries such as jax, and differentiable simulation engines such as timemachine.

## 2.15.1 Theory

The framework currently employs the fluctuation approach [1] to compute gradients of observables with respect to the force field parameters used to estimate them.

This approach may be derived by direct differentiation of the ensemble average an observable X:

$$\langle X(\theta) \rangle = \frac{1}{Q(\theta)} \int X(\theta) \exp\left[-\beta \left(U(\vec{r}, V; \theta) + pV\right)\right] d\vec{r} dV$$

where

$$Q(\theta) = \int \exp\left[-\beta \left(U\left(\vec{r}, V; \theta\right) + pV\right)\right] d\vec{r} dV$$

is the isothermal-isobaric partion function,  $\theta$  are the force field parameters being used to estimate the observable, U the systems potential energy,  $\beta \equiv k_b T$ ,  $k_b$  the Boltzmann constant, T the temperature, p the pressure and V the volume.

The derivative of the ensemble average defined above with respect to a particular force field parameter of interest  $\theta$  is given by:

$$\begin{split} \frac{\mathrm{d}\left\langle X\right\rangle }{\mathrm{d}\theta_{i}} &= \left\langle \frac{\mathrm{d}X}{\mathrm{d}\theta_{i}} \right\rangle - \beta \left[ \right. \\ \left\langle X \frac{\mathrm{d}U}{\mathrm{d}\theta_{i}} \right\rangle - \left\langle \frac{\mathrm{d}U}{\mathrm{d}\theta_{i}} \right\rangle \left\langle X \right\rangle \end{split}$$

# **2.15.2 Computing** $dU/d\theta_i$

While future integrations with differentiable simulation engines such as timemachine will allow  $dU/d\theta_i$  to be computed directly from molecular simulation runs, currently most common simulation engines do not directly support computing this quantity.

Until such an integration is complete, the framework currently employs a central finite difference approach, whereby

$$\frac{\mathrm{d}U}{\mathrm{d}\theta_{i}} \approx \frac{U(\theta_{i} + h) - U(\theta_{i} - h)}{2h}$$

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. This has been found to yield finite differences which do not suffer from precision issues, while being sufficiently small so as to yield an accurate estimate.

In practice the derivatives obtained by re-evaluating the energies of each configuration in a trajectory generated by a molecular simulation (either after a simulation or after loading one from disk) at each of the perturbed parameters.

While there is an expense associated with extra evaluations of the potential energy function for each configuration, this is mitigated by only computing those terms which depend upon (or may depend upon)  $\theta_i$ . As an example, when computing derivatives with respect to a bond length the electrostatic and van der Waal contributions are not computed. This significantly speeds up the computation of these derivatives.

The final derivatives are stored in *ObservableArray* objects for convenience and for easy propagation of gradients through workflows. See the *observables documentation* for more information.

### 2.15.3 References

# 2.16 Calculation Layers

A *CalculationLayer* is an implementation of one calculation approach for estimating a set of physical properties, such as via molecular simulation or evaluating some QSAR like model.

The framework stacks multiple layers together when estimating a data set of properties.

Fig. 2: A schematic of the layer system. A set of properties to estimate are fed into the first layer. Those which can be calculated are returned back. Those that can't are passed to the next layer until no layer are left.

Each layer will in turn attempt to evaluate the properties being estimated using the specific approach the layer represents, such as by running a set of simulations. If the layer is unable to estimate a given property, for example if a layer does not yet support a given property, or if the layer has insufficient data to reprocesses, the property will be passed to the next layer for it to try and evaluate.

In practice, this allows the framework to attempt to estimate a data set using the most rapid calculation layer first, before moving to successively slower yet more robust layers, and thus enabling as efficient as possible property estimation.

## 2.16.1 Defining a Calculation Layer

A calculation layer is defined by two objects - a *CalculationLayer* object which implements the main layer logic, and a *CalculationLayerSchema* which defines those settings and options exposed required by the layer.

One *CalculationLayerSchema* will be provided to the for each type of property that the layer is being asked to estimate. The base *CalculationLayerSchema* currently only exposes options for optionally defining either the relative or absolute uncertainty that the layer should attempt to estimate the associated property type to within, however custom schemas can be defined per layer.

The structure of a *CalculationLayer* is relatively simple and permissive:

```
@calculation_layer()
class MyCalculationLayer(CalculationLayer):

    @classmethod
    def required_schema_type(cls):
        return CalculationLayerSchema

    @classmethod
    def _schedule_calculation(
        cls,
        calculation_backend,
        storage_backend,
        layer_directory,
        batch
):
    ...
```

The first thing to note is the *calculation\_layer()* decorator which is being applied to the class. This registers the calculation layer with the frameworks plug-in system, allowing it to be used in future calculations.

The only other requirements is that the class implement a required\_schema\_type class method, which returns the type of *CalculationLayerSchema* that is associated with this layer, and a \_schedule\_calculation(). The \_schedule\_calculation() is responsible for performing the actual property calculations.

The form of the \_schedule\_calculation() function is very flexible:

(continues on next page)

```
return futures
```

It takes as arguments:

- a CalculationBackend which is used to asynchronously distribute any calculations across the available compute resources.
- a StorageBackend which may be used to store / cache any data generated by the calculations.
- the path to the directory within which all of the calculation working files should be stored.
- the *Batch* of properties which this layer should attempt to estimate. This object includes the properties to estimate, as well as the *CalculationLayerSchema* for each property type.

and must return a list of Future objects (which either must be or implement the same API as the asyncio Future object). The easiest way to generate the futures is to perform any calculations using the calculation\_backend which will automatically return the results of any functions as such.

The future objects returned by \_schedule\_calculation() must return a *CalculationLayerResult* object, which includes

- the estimated property if the calculation was successful (or *UNDEFINED* otherwise).
- a list of any exceptions (of type EvaluatorException) which were raised during the calculation.
- a list of any data to be stored by the storage backend.

As a minimal example of a method which returns one such object:

```
@classmethod
def process_property(cls, physical_property, **_):
    """Return a result as if the property had been successfully estimated.
"""

# TODO: Do some calculations

# Set the property provenance
physical_property.source = CalculationSource(fidelity=cls.__name__)

# Return the results object.
results = CalculationLayerResult()
results.physical_property = physical_property
return results
```

## 2.16.2 Default Schemas

Default schemas for each pair of a calculation layer and a type of physical property may be registered using the register\_calculation\_schema() function:

```
schema=Density.default_simulation_schema
```

where the schema object should either be an instance of a *CalculationLayerSchema*, or a function with no required arguments which returns a *CalculationLayerSchema*.

A list of the registered schemas is provided by the registered\_calculation\_schemas module attribute.

# 2.17 Workflow Layers

The WorkflowCalculationLayer and WorkflowCalculationSchema offer an abstract base implementation for any calculation layers (and their associated schemas) which will perform their calculations using the built-in workflow engine.

The WorkflowCalculationLayer takes as input from its calculation schema one WorkflowSchema object for each type of property to be estimated by this layer. These schemas must at a minimum provide both the schemas of the protocols in the workflow, and have the final\_value\_source attribute set to the value of the calculated observable. In addition, the layer fully supports schemas which provide gradient information (see the gradients\_sources attribute), as well as storing any generated dataclasses (see the outputs\_to\_store attribute) to the available storage backend.

This layer implements three key methods which are available to be overridden by any subclass implementations:

- \_get\_workflow\_metadata(): a method which returns the dictionary of *metadata* which will be made available to the workflow (see the *default metadata* section for details).
- \_build\_workflow\_graph(): the method which will construct the *workflow graph* to execute using the input workflow schemas and the metadata generated by the layer.
- workflow\_to\_layer\_result(): a method which will map any WorkflowResult objects generated by the workflow graph into the CalculationLayerResult objects which the layer requires.

The workflow layer will by default tag each property estimated using it (or one of its derivatives) with a *CalculationSource* with the *fidelity* attribute set to the name of the layer, and the *provenance* attribute set to the schema of the workflow used to generate the property.

# 2.17.1 Default Metadata

The metadata provided to the workflows generated by this layer is generated on a per property to estimate basis mainly using the <code>generate\_default\_metadata()</code> function. It includes:

Key	Туре	Description
thermodynamic_state	ThermodynamicState	The state at which the to perform any calculations.
substance	Substance	The substance to use in any calculations.
components	[Substance]	The components present in the main substance.
target_uncertainty	Quantity	The target uncertainty of any calculations defined by the calculation schema.
per_component_uncertainty	Quantity	The target_uncertainty divided by sqrt(substance.n_components + 1)
force_field_path	str	A file path to the force field parameters to use.
parameter_gradient_keys	[ParameterGradientKey]	The parameters to differentiate any observables with respect to (if any).

# 2.18 The Direct Simulation Layer

The *SimulationLayer* is a calculation layer which employs molecular simulation to estimate data sets of physical properties. It inherits the *WorkflowCalculationLayer* base layer, and primarily makes use of the built-in *workflow* engine to perform the required calculations.

The simulation layer is expected to *almost always* be able to estimate any properties requested of it (with exceptions being where a workflow schema has not yet been defined for a class of properties, or where an unexpected error occurs), and can be thought of as a safe 'fallback' layer when no other calculation approach are able to estimate particular properties.

It is expected that *workflow schemas* passed to the simulation layer should be able to estimate the gradients of the observable they aim to calculate, as well as specify a set of storage/dataclasses which contain the data generated by the molecular simulations.

### 2.18.1 Default Metadata

The simulation layer makes the same set of metadata available to its workflows as the parent workflow layer.

# 2.19 The MBAR Reweighting Layer

The ReweightingLayer is a calculation layer which employs the Multistate Bennett Acceptance Ratio (MBAR) method to calculate observables at states which have not been previously simulated, but for which simulations have been previously run at similar states and their data cached. It inherits the WorkflowCalculationLayer base layer, and primarily makes use of the built-in workflow engine to perform the required calculations.

Because MBAR is a technique which reprocesses exisiting simulation data rather than re-running new simulations, it is typically several fold faster than the *simulation layer* provided it has cached simulation data (made accessible via a *storage backend*) available. Any properties for which the required data (see *Calculation Schema*) is not available will be skipped.

# 2.19.1 Theory

The theory behind applying MBAR to reweighting observables from a simulated state to an unsimulated state is covered in detail in the publication Configuration-Sampling-Based Surrogate Models for Rapid Parameterization of Non-Bonded Interactions.

### 2.19.2 Calculation Schema

The reweighting layer will be provided with one *ReweightingSchema* per type of property that it is being requested to estimate. It builds off of the base *WorkflowCalculationSchema* schema providing an additional *storage\_queries* attribute.

The *storage\_queries* attribute will contain a dictionary of *SimulationDataQuery* which will be used by the layer to access the data required for each property from the storage backend. Each key in this dictionary will correspond to the key of a piece of metadata made available to the property workflows.

### 2.19.3 Default Metadata

The reweighting layer makes available the default metadata provided by the *parent workflow layer* in addition to any cached data retrieved via the schemas *storage\_queries*.

When building the metadata for each property, a copy of the query will be made and any of the supported attributes (currently only *substance*) whose values are set as *PlaceholderValue* objects will have their values updated using values directly from the property. This query will then be passed to the storage backend to retrieve any matching data.

The matching data will be stored as a list of tuples of the form:

```
(object_path, data_directory, force_field_path)
```

where object\_path is the file path to the stored dataclass, the data\_directory is the file path to the ancillary data directory and force\_field\_path is the file path to the force field parameters which were used to generate the data originally.

This list of tuples will be made available as metadata under the key that was associated with the query.

## 2.20 Workflows

The framework offers a lightweight workflow engine for executing graphs of tasks using the available *calculation backends*. While lightweight, it offers a large amount of extensibility and flexibility, and is currently used by both the *simulation* and *reweighting* layers to perform their required calculations.

A workflow is a wrapper around a collection of tasks that should be executed in succession, and whose outputs should be made available as the input to others.

Fig. 3: A an example workflow which combines a protocol which will build a set of coordinates for a particular system, assign parameters to that system, and then perform an energy minimisation.

The workflow engine offers a number of advanced features such as the *automatic reduction of redundant tasks*, and *looping over parts of a workflow* 

# 2.20.1 Building Workflows

At its core a workflow must define the tasks which need to be executed, and where the inputs to those tasks should be sourced from. Each task to be executed is represented by a *protocol object*, with each protocol requiring a specific set of user specified inputs:

```
# Define a protocol which will build some coordinates for a system.
build_coordinates = BuildCoordinatesPackmol("build_coordinates")
build_coordinates.max_molecules = 1000
build_coordinates.mass_density = 1.0 * unit.gram / unit.millilitre
build_coordinates.substance = Substance.from_components("0", "CO")

# Define a protocol which will assign force field parameters to the system.
assign_parameters = BuildSmirnoffSystem(f"assign_parameters")
assign_parameters.water_model = BuildSmirnoffSystem.WaterModel.TIP3P
assign_parameters.force_field_path = "openff-1.0.0.offxml"

# Set the `coordinate_file_path` input of the `assign_parameters` protocol
```

(continues on next page)

2.20. Workflows 53

```
# to the `coordinate_file_path` output of the `build_coordinates` protocol.
assign_parameters.coordinate_file_path = ProtocolPath(
    "coordinate_file_path", build_coordinates.id
)
```

The *ProtocolPath* object is used to reference the output of another protocol in the workflow, and will be replaced by the value of that output once that protocol has been executed by the workflow engine. It is constructed from two parts:

- the name of the output attribute to reference.
- the unique id of the protocol to take the output from.

To turn these tasks into a valid workflow which can be automatically executed, they must first be converted to a *workflow schema*:

```
# Create the schema object.
schema = WorkflowSchema()
# Add the individual protocol's schema representations to the workflow schema.
schema.protocol_schemas = [build_coordinates.schema, assign_parameters.schema]
# Create the executable workflow object from its schema.
workflow = Workflow.from_schema(schema, metadata=None)
```

A Workflow may either be synchronously executed in place yielding a WorkflowResult object directly:

```
workflow_result = workflow.execute()
```

or asynchronously using a calculation backend yielding a Future like object which will eventually return a <code>WorkflowResult</code>:

```
with DaskLocalCluster() as calculation_backend:
    result_future = workflow.execute(calculation_backend=calculation_backend)
```

In addition, a workflow may be add to, and executed as part as a larger workflow graphs.

### 2.20.2 Workflow Schemas

A *WorkflowSchema* is a blueprint from which all *Workflow* objects are constructed. It will predominantly define the tasks which compose the workflow, but may optionally define:

- final\_value\_source: A reference to the protocol output which corresponds to the value of the main observable calculated by the workflow.
- gradients\_sources: A list of references to the protocol outputs which correspond to the gradients of the main observable with respect to a set of force field parameters.
- outputs\_to\_store: A list of data classes whose values will be populated from protocol outputs.
- protocol\_replicators: A set of replicators which are used to flag parts of a workflow which should be replicated.

Each of these attributes will control whether the *value*, *gradients* and *data\_to\_store* attributes of the *WorkflowResult* results object will be populated respectively when executing a workflow.

#### Metadata

Because a schema is purely a blueprint for a general workflow, it need not define the exact values of all of the inputs of its constituent tasks. Consider the above example workflow for constructing a set of coordinates and assigning force field parameters to them. Ideally this one schema could be reused for multiple substances. This is made possible through a workflows *metadata*.

Each protocol within a workflow may access a dictionary of values unique to that workflow (termed here *metadata*) which is defined when the *Workflow* object is created from its schema.

This metadata may be accessed by protocols via a fictitious "global" protocol whose outputs map to the metadata dictionary:

```
build_coordinates = BuildCoordinatesPackmol("build_coordinates")
build_coordinates.substance = ProtocolPath("substance", "global")

# ...
substances = [
    Substance.from_components("CO"),
    Substance.from_components("CCO"),
    Substance.from_components("CCO"),
]

for substance in substances:

# Define the metadata to make available to the workflow protocols.
metadata = {"substance": substance}
# Create the executable workflow object from its schema.
workflow = Workflow.from_schema(schema, metadata=metadata)

# Execute the workflow ...
```

the created workflow will contain the build\_coordinates protocol but with its substance input set to the value from the metadata dictionary.

# 2.21 Replicators

A *ProtocolReplicator* is the workflow equivalent of a for loop. It is statically evaluated when a *Workflow* is created from its schema. This is useful when parts of a workflow should be run multiple times but using different values for certain protocol inputs.

**Note:** The syntax of replicators is still rather rough around the edges, and will be refined in future versions of the framework.

Each *ProtocolReplicator* requires both a unique id and the set of *template values* which the replicator will 'loop' over to be defined. These values must either be a list of constant values or a reference to a list of values provided as *metadata*.

The 'loop variable' is referenced by protocols in the workflow using the *ReplicatorValue* placeholder input, where the value is linked to the replicator through the replicators unique id.

As an example, consider the case where a set of coordinates should be built for each component in a substance:

2.21. Replicators 55

```
# Create the replicator object, and assign it a unique id.
replicator = ProtocolReplicator(replicator_id="component_replicator")
# Instruct the replicator to loop over all of the components of the substance
# made available by the global metadata
replicator.template_values = ProtocolPath("substance.components", "global")

# Define a protocol which will build some coordinates for a system.
build_coords = BuildCoordinatesPackmol("build_coords_" + replicator.placeholder_id})
# Instruct the protocol to use the value specified by the replicator.
build_coords.substance = ReplicatorValue(replicator.id)

# Build the schema containing the protocol and the replicator
schema = WorkflowSchema()
schema.protocol_schemas = [build_coords.schema]
schema.protocol_replicators = [replicator]
```

The requirement for a protocol to be replicated by a replicator is that its id *must* contain the replicators *placeholder\_id* - this is a simple string which the workflow engine looks for when applying the replicator. The contents of this schema can be easily inspected by printing its JSON representation:

```
{
    "@type": "openff.evaluator.workflow.schemas.WorkflowSchema",
    "protocol_replicators": [
        {
            "@type": "openff.evaluator.workflow.schemas.ProtocolReplicator",
            "id": "component_replicator",
            "template_values": {
                "@type": "openff.evaluator.workflow.utils.ProtocolPath",
                "full_path": "global.substance.components"
            }
        }
    ],
    "protocol_schemas": [
            "@type": "openff.evaluator.workflow.schemas.ProtocolSchema",
            "id": "build_coords_$(component_replicator)",
            "inputs": {
                ".substance": {
                    "@type": "openff.evaluator.workflow.utils.ReplicatorValue",
                    "replicator_id": "component_replicator"
                }
            "type": "BuildCoordinatesPackmol"
        }
   ]
}
```

It can be clearly seen that the schema only contains a single protocol entry, with the placeholder id present in its unique id. Once a workflow is created from this schema however:

```
# Define some metadata
metadata = {"substance": Substance.from_components("0", "CO")}

(continues on next page)
```

```
# Build the workflow from the schema.
workflow = Workflow.from_schema(schema, metadata)
# Output the contents of the workflow as JSON.
print(workflow.schema.json())
```

it can be seen that the replicator has been correctly been applied and the workflow now contains one protocol for each component in the substance passed as metadata:

```
{
    "@type": "openff.evaluator.workflow.schemas.WorkflowSchema",
    "protocol_schemas": [
        {
            "@type": "openff.evaluator.workflow.schemas.ProtocolSchema",
            "id": "build_coords_0",
            "inputs": {
                ".substance": {
                    "@type": "openff.evaluator.substances.components.Component",
                    "smiles": "0"
                }
            "type": "BuildCoordinatesPackmol"
        },
            "@type": "openff.evaluator.workflow.schemas.ProtocolSchema",
            "id": "build_coords_1".
            "inputs": {
                ".substance": {
                    "@type": "openff.evaluator.substances.components.Component",
                    "smiles": "CO"
                }
            },
            "type": "BuildCoordinatesPackmol"
        }
    ]
}
```

In both cases the replicators *placeholder\_id* has been replaced with the index of the value it was replicated for, and the substance input has been correctly set to the actual array value.

# 2.21.1 Nested Replicators

Replicators can be applied to other replicators to achieve a result similar to a set of nested for loops. For example the below loop:

```
components = [Component("0"), Component("CO")]
n_mols = [[1000], [500]]

for i, component in enumerate(components):
    for component_n_mols in n_mols[i]:
        ...
```

2.21. Replicators 57

can readily be reproduced using replicators:

```
# Define a replicator which will loop over all components in the substance.
component_replicator = ProtocolReplicator(replicator_id="components")
component_replicator.template_values = ProtocolPath("components", "global")
# Define a replicator to loop over the number of each component to add.
n_mols_replicator_id = f"n_mols_{component_replicator.placeholder_id}"
n_mols_replicator = ProtocolReplicator(replicator_id=n_mols_replicator_id)
n_mols_replicator.template_values = ProtocolPath(
    f"n_mols[{component_replicator.placeholder_id}]", "global"
)
# Define the suffix which must be applied to protocols to be replicated
id_suffix = f"{component_replicator.placeholder_id}"
# Define a protocol which will build some coordinates for a system.
build_coordinates = BuildCoordinatesPackmol(f"build_coordinates_{id_suffix}")
build_coordinates.substance = ReplicatorValue(component_replicator.id)
build_coordinates.max_molecules = ReplicatorValue(n_mols_replicator.id)
# Build the schema containing the protocol and the replicator
schema = WorkflowSchema()
schema.protocol_schemas = [build_coordinates.schema]
schema.protocol_replicators = [component_replicator, n_mols_replicator]
# Define some metadata
metadata = {
    "components": [Component("0"), Component("CO")],
    "n_mols": [[1000], [500]]
}
# Build the workflow from the created schema.
workflow = Workflow.from_schema(schema, metadata)
# Print the JSON representation of the workflow.
print(workflow.schema.json(format=True))
```

Here the component\_replicator placeholder id has been appended to the n\_mols\_replicator id to inform the workflow engine that the later is a child of the former. The component\_replicator placeholder id is then used as an index into the n\_mols array. This results in the following schema as desired:

(continues on next page)

```
},
    "type": "BuildCoordinatesPackmol"
},
{
    "@type": "openff.evaluator.workflow.schemas.ProtocolSchema",
    "id": "build_coordinates_1_0",
    "inputs": {
        ".max_molecules": 500,
        ".substance": {
            "@type": "openff.evaluator.substances.components.Component",
            "smiles": "CO"
        }
},
    "type": "BuildCoordinatesPackmol"
}
```

# 2.22 Workflow Graphs

A *WorkflowGraph* is a collection of *Workflow* objects which should be executed together. The primary advantage of executing workflows via the graph object is that the graph will automatically take advantage of the *protocols* built in redundancy / merging support to collapse duplicate tasks across multiple workflows.

As an example, consider the case of executing workflows to estimate the density and the dielectric constant at the same state point, for the same substance, and using the same force field parameters:

```
density_schema = Density.default_simulation_schema()
dielectric_schema = DielectricConstant.default_simulation_schema()

density_workflow = Workflow.from_schema(density_schema, metadata)
dielectric_workflow = Workflow.from_schema(dielectric_schema, metadata)

print(len(density_workflow.protocols), len(dielectric_workflow.protocols))

workflow_graph = WorkflowGraph()
workflow_graph.add_workflows(density_workflow, dielectric_workflow)

print(len(workflow_graph.protocols))
```

The final workflow graph has roughly half the total number of density and dielectric protocols to be executed. This is expected as both the density and dielectric workflows are almost identical, except for the final analysis steps.

Graphs can be executed either in place without using a calculation backend in the same way that workflows can.

## 2.23 Protocols

The *Protocol* class represents a single task to be executed, whether that be as a standalone task or as a task which is part of some larger workflow. The task encoded by a protocol may be as simple as adding two numbers together or even as complex as performing entire free energy simulations:

```
from openff.evaluator.protocols.miscellaneous import AddValues

# Create the protocol and assign it some unique name.
add_numbers = AddValues(protocol_id="add_values")

# Set the numbers to add together
add_numbers.values = [1, 2, 3, 4]

# Execute the protocol
add_numbers.execute()

# Retrieve the output
result = add_numbers.result
```

## 2.23.1 Inputs and Outputs

Each protocol exposes a set of the required inputs as well as the produced outputs. These inputs may either be set as a constant directly, or if used as part of a *workflow*, can take their value from one of the outputs of another protocol.

Fig. 4: A selection of the inputs and outputs of the *OpenMMSimulation* protocol.

A surprisingly rich spectrum of workflows can be constructed by chaining together many relatively simple protocols.

The inputs and outputs of a protocol are defined using the custom *InputAttribute* and *OutputAttribute* descriptors:

```
class AddValues(Protocol):

# Define the inputs that the protocol requires
values = InputAttribute(
    docstring="The values to add together.",
    type_hint=list, default_value=UNDEFINED
)

# Define the outputs that the protocol will produce
# once it is executed.
result = OutputAttribute(
    docstring="The sum of the values.",
    type_hint=typing.Union[int, float, unit.Measurement, unit.Quantity],
)

def _execute(self, directory, available_resources):
    ...

def validate(self, attribute_type=None):
    ...
```

Here we have defined a values input to the protocol and a result output. Both descriptors require a docstring and a type\_hint to be provided.

The type\_hint will be used by the workflow engine to ensure that a protocol which takes its input as the output of another protocol is receiving values of the correct type. Currently the type\_hint can be any type of python class, or a Union of multiple types should the protocol allow for that.

In addition, the input attribute must specify a default\_value for the attribute. This can either be a constant value, or a value set by some function such as a lambda statement:

```
some_input = InputAttribute(
   docstring="Takes it's default value from a function.",
   type_hint=int,
   default_value=lambda: return 1 + 1
)
```

In the above example we set the default value of values to *UNDEFINED* in order to specify that this input must be set by the user. The custom *UNDEFINED* class is used in place of None as None may be a valid input value for some attributes.

### 2.23.2 Task Execution

In addition to defining its inputs and outputs, a protocol must also implement an \_execute() function which handles the main logic of the task:

```
def _execute(self, directory, available_resources):
    self.result = self.values[0]
    for value in self.values[1:]:
        self.result += value
```

The function is passed the directory in which it should run and create any working files, as well as a *ComputeResources* object which describes which compute resources are available to run on. This function *must* set all of the output attributes of the protocol before returning.

The private <code>\_execute()</code> function which must be implemented should not be confused with the public <code>execute()</code> function. The public <code>execute()</code> function implements some common protocol logic (such as validating the inputs and creating the directory to run in if needed) before calling the private <code>\_execute()</code> function.

### 2.23.3 Protocol Validation

The protocols inputs will automatically be validated before <code>\_execute()</code> is called - this validation includes making sure that all of the non-optional inputs have been set, as well as ensuring they have been set to a value of the correct type. Protocols may implement additional validation logic by implementing a <code>execute()</code> function:

```
def validate(self, attribute_type=None):
    super(AddValues, self).validate(attribute_type)

if len(self.values) < 1:
    raise ValueError("There were no values to add together")</pre>
```

2.23. Protocols 61

### **2.23.4 Schemas**

Every protocol has a *ProtocolSchema* representation which uniquely describes the protocol, and from which the protocol can be exactly recreated. The schema stores not only the type of protocol which it represents, but also the values of each of the inputs. Protocol schemas are fully JSON serializable. The following is an example schema for the above add\_numbers protocol:

```
{
  "@type": "openff.evaluator.workflow.schemas.ProtocolSchema",
  "id": "add_values",
  "inputs": {
     ".allow_merging": true,
     ".values": [1, 2, 3, 4]
},
  "type": "AddValues"
}
```

A protocols schema can be accessed via it's *schema* attribute. A protocol can be directly created from its schema representation by calling the schema's *to\_protocol()* function.

## 2.23.5 Merging Protocols

When executing multiple workflows together (e.g. executing a workflow to estimate a substances density and potential energy) there is a large likelihood that some of tasks in those two workflows will be identical. Examples may include two workflows requiring protocols which build a set of coordinates, or assigning the same set of parameters to those coordinates.

Protocols have built-in support for comparing whether they are performing the same task / calculation as another protocol through the <code>can\_merge()</code> and <code>merge()</code> functions:

- The *can\_merge()* function checks to see whether two protocols are performing an identical task and hence whether they should be merged or not.
- The merge() function handles the actual merging of two protocols which can be merged.

The default <code>can\_merge()</code> function takes advantage of the <code>merge\_behvaiour</code> attribute of the different input descriptors. The <code>merge\_behvaiour</code> attribute describes how each input should be considered when checking to see if two protocols can be merged:

```
max_molecules = InputAttribute(
    docstring="The maximum number of molecules to be added to the system.",
    type_hint=int,
    default_value=1000,
    merge_behavior=MergeBehavior.ExactlyEqual
)
```

The most common behavior is to require that the inputs must be ExactlyEqual in order for two protocols two be considered to be identical. However, for some inputs such as the timestep of a simulation or the number of steps to simulate for, the exact values of the inputs don't necessarily need to be equal but rather, we may just wish to take the larger / smaller of the two inputs:

```
default_value=2.0 * unit.femtosecond,
)

total_number_of_iterations = InputAttribute(
    docstring="The number of times to propogate the system forward by.",
    type_hint=int,
    merge_behavior=InequalityMergeBehavior.LargestValue,
    default_value=1,
)
```

This can be accomplished using the InequalityMergeBehavior enum.

The default <code>merge()</code> function also relies upon the <code>merge\_behaviour</code> attributes to determine which values of the inputs should be retained when merging two protocols.

# 2.24 Protocol Groups

The *ProtocolGroup* class represents a collection of *protocols* which have been grouped together. All protocols within a group will be executed together on a single compute resources, i.e. there is currently no support for executing protocols within a group in parallel.

Protocol groups have a specialised *ProtocolGroupSchema* which is essentially a collection of *ProtocolSchema* objects.

# 2.24.1 Conditional Protocol Groups

A *ConditionalGroup* is a special class of *ProtocolGroup* which will execute all of the grouped protocols again and again until a set of conditions has been met or until a maximum number of iterations (see *max\_iterations*) has been performed. They can be thought of as being a protocol representation of a while statement.

Each condition to be met is represented by a *Condition* object:

```
condition = ConditionalGroup.Condition()

# Set the left and right hand values.
condition.left_hand_value = ...
condition.right_hand_value = ...

# Choose the type of condition
condition.type = ConditionalGroup.Condition.Type.LessThan
```

The left and right hand values can either be constants, or come from the output of another protocol (including grouped protocols) using a *ProtocolPath*. Currently a condition can either check that a value is less than or greater than another value.

Conditional groups expose a *current\_iteration* attribute which tracks how many times the grouped protocols have been executed. This can be used as input by any of the grouped protocols and is useful, for example, to run a simulation for longer and longer until the groups condition has been met:

```
conditional_group = ConditionalGroup("conditional_group")

# Set up protocols to run a simulation and then to extract the

(continues on next page)
```

```
# value of the density and its uncertainty.
simulation = OpenMMSimulation("simulation")
simulation.input_coordinate_file = "coords.pdb"
simulation.parameterized_system = ...
extract_density = AverageObservable("extract_density")
extract_density.observable = simulation.observables["Density"]
# Set the total number of iterations the simulation should perform to be equal
# to the current iteration of the group. I.e the simulation should perform a
# new iteration at each group iteration.
simulation.total_number_of_iterations = ProtocolPath(
    "current_iteration", conditional_group.id
# Add the protocols to the group.
conditional_group.add_protocols(production_simulation, analysis_protocol)
# Set up a condition which will check if the uncertainty is less than
# some threshold.
condition = ConditionalGroup.Condition()
condition.condition_type = groups.ConditionalGroup.Condition.Type.LessThan
condition.right_hand_value = 0.5 * unit.gram / unit.millilitre
condition.left_hand_value = ProtocolPath(
    "value.error", conditional_group.id, analysis_protocol.id
)
# Add the condition.
conditional_group.add_condition(condition)
```

It is this idea which is used to continue running a molecular simulations until an observable of interest (such as the density) has been calculated to within a specified uncertainty.

# 2.25 Observables

A key feature of this framework is its ability to compute the gradients of physical properties with respect to the force field parameters used to estimate them. This requires the framework be able to, internally, be able to not only track the gradients of all quantities which combine to yield the final observable of interest, but to also be able to propagate the gradients of those composite quantities through to the final value.

The framework offers three such objects to this end (*Observable*, *ObservableArray* and *ObservableFrame* objects) which will be covered in this document.

**Note:** In future versions of the framework the objects described here will likely be at least in part deprecated in favour of using full automatic differentiation libraries such as jax. Supporting these libraries will take a large re-write of the framework however, as well as full support between differentiable simulation engines like timemachine and the OpenFF toolkit. As such, these objects are implemented as stepping stones which can be gently phased out while working towards that larger, more modern goal.

## 2.25.1 Observable Objects

The base object used to track observables is the *Observable* object. It stores the average value, the standard error in the value and the gradient of the value with respect to force field parameters of interest.

Currently the value and error are internally stored in a composite Measurement object, which themselves wrap around the uncertainties package. This allows uncertainties to be automatically propagated through operations without the need for user intervention.

**Note:** Although uncertainties are automatically propagated, it is still up to property estimation workflow authors to ensure that such propagation (assuming a Gaussian error model) is appropriate. An alternative, which is employed throughout the framework is to make use of the bootstrapping technique.

Gradients are stored in a list as *ParameterGradient* gradient objects, which store both the floating value of the gradient alongside an identifying *ParameterGradientKey*.

## **Supported Operations**

- + and -: *Observable* objects can be summed with and subtracted from other *Observable* objects, Quantity objects, floats or integers. When two *Observable* objects are summed / subtracted, their gradients are combined by summing / subtracting also. When an *Observable* is summed / subtracted with a Quantity, float or int object it is assumed that these objects do not depend on any force field parameters.
- \*: Observable objects may be multiplied by other Observable objects, Quantity objects, and float or int objects. When two Observable objects are multiplied their gradients are propagated using the product rule. When an Observable is multiplied by a Quantity, float or int object it is assumed that these objects do not depend on any force field parameters.
- /: Observable objects may be divided by other Observable objects, Quantity objects, and float or int objects. Gradients are propagated through the division using the quotient rule. When an Observable is divided by a Quantity, float or int object (or when these objects are divided by an Observable object) it is assumed that these objects do not depend on any force field parameters.

In all cases two *Observable* objects can only be operated on provided the contain gradient information with respect to the same set of force field parameters.

## 2.25.2 Observable Arrays

An extension of the *Observable* object is the *ObservableArray* object. Unlike an *Observable*, an *ObservableArray* object does not contain error information, but rather the value it stores and the gradients of that value should be a numpy array with shape=(n\_data\_points, n\_dimensions). It is designed to store information such as the potential energy evaluated at each configuration sampled during a simulation, as well as the gradient of the potential, which can then be ensemble averaged using a fluctuation formula to propagate the gradients through to the average.

Like with *Observable* objects, gradients are stored in a list as *ParameterGradient* gradient objects. The length of the gradients is required to match the length of the value array.

ObservableArray objects may be concatenated together using their join() method or sub-sampled using their subset() method.

2.25. Observables 65

### **Supported Operations**

The *ObservableArray* object supports the same operations as the *Observable* object, whereby all operations are applied elementwise to the stored arrays.

### 2.25.3 Observable Frames

An *ObservableFrame* is a wrapper around a collection of *ObservableArray* which contain the types of observable specified by the *ObservableType* enum. It behaves as a dictionary which can take either an *ObservableType* or a string value of an *ObservableType* as an index.

Like an *ObservableArray*, observable frames may be concatenated together using their *join()* method or subsampled using their *subset()* method.

### **Supported Operations**

No operations are supported between observable frames.

## 2.26 Calculation Backends

A *CalculationBackend* is an object used to distribute calculation tasks across available compute resources. This is possible through specific backends which integrate with libraries such as multiprocessing, dask, parsl and cerlery.

Each backend is responsible for creating *compute workers*. A compute worker is an entity which has a set amount of dedicated compute resources available to it and which can execute python functions using those resources. Calculation backends may spawn multiple workers such that many tasks and calculations can be performed simultaneously.

A compute worker can be as simple as a new multiprocessing Process or something more complex like a dask worker. The resources available to a worker are described by the *ComputeResources* object.

CalculationBackend classes have a relatively simple structure:

```
class MyCalculationBackend(CalculationBackend):
    def __init__(self, number_of_workers, resources_per_worker):
        ...

def start(self):
        ...

def stop(self):
        ...

def submit_task(self, function, *args, **kwargs):
        ...
```

By default they implement a constructor which takes as input the number of workers that the backend should initially spawn as well as the compute resources which are available to each. They must further implement:

- a *start()* method which spawns the initial set of compute workers.
- a *stop()* method which should kill all workers spawned by the backend as well as cleanup any temporary worker files.

• a *submit\_task()* method which takes a function to be execute by a worker, and a set of args and kwargs to pass to that function.

The <code>submit\_task()</code> must run asynchronously and return an asyncio <code>Future</code> object (or an object which implements the same API) when called, which can then be queried for when the task has completed.

All calculation backends are implemented as context managers such that they can be used as:

```
with MyCalculationBackend(number_of_workers=..., resources_per_worker...) as backend:
   backend.submit_task(...)
```

where the *start()* and *stop()* methods will be called automatically.

## 2.27 Dask Backends

The framework implements a number of calculation backends which integrate with the dask distributed and job-queue libraries.

### 2.27.1 Dask Local Cluster

The DaskLocalCluster backend wraps around the dask LocalCluster class to distribute tasks on a single machine:

```
worker_resources = ComputeResources(
    number_of_threads=1,
    number_of_gpus=1,
    preferred_gpu_toolkit=ComputeResources.GPUToolkit.CUDA,
)

with DaskLocalCluster(number_of_workers=1, resources_per_worker=worker_resources) as_____
local_backend:
    local_backend.submit_task(logging.info, "Hello World")
    ...
```

Its main purpose is for use when debugging calculations locally, or when running calculations on machines with large numbers of CPUs or GPUs.

### 2.27.2 Dask HPC Cluster

The <code>DaskLSFBackend</code>, <code>DaskPBSBackend</code>, and <code>DaskSLURMBackend</code> backends wrap around the dask <code>LSFCluster</code>, <code>PBSCluster</code>, and <code>SLURMCluster</code> classes respectively, and both inherit the <code>BaseDaskJobQueueBackend</code> class which implements the core of their functionality. They predominantly run in an adaptive mode, whereby the backend will automatically scale up or down the number of workers based on the current number of tasks that the backend is trying to execute.

These backends integrate with the queueing systems which most HPC cluster use to manage task execution. They work by submitting jobs into the queueing system which themselves spawn dask workers, which in turn then execute tasks on the available compute nodes:

2.27. Dask Backends 67

```
number_of_gpus=1,
   preferred_gpu_toolkit=QueueWorkerResources.GPUToolkit.CUDA,
   per_thread_memory_limit=worker_memory,
   wallclock_time_limit="05:59",
)
# Create the backend object.
setup_script_commands = [
    f"conda activate evaluator",
    f"module load cuda/10.1",
]
calculation_backend = DaskLSFBackend(
   minimum_number_of_workers=1,
   maximum_number_of_workers=max_number_of_workers,
   resources_per_worker=queue_resources,
   queue_name="gpuqueue",
    setup_script_commands=setup_script_commands,
)
# Perform some tasks.
with calculation_backend:
    calculation_backend.submit_task(logging.info, "Hello World")
```

The setup\_script\_commands argument takes a list of commands which should be run by the queue job submission script before spawning the actual worker. This enables setting up custom environments, and setting any required environmental variables.

## Configuration

To ensure optimal behaviour we recommend changing / uncommenting the following settings in the dask distributed configuration file (this can be found at ~/.config/dask/distributed.yaml):

```
distributed:
    worker:
        daemon: False

comm:
        timeouts:
            connect: 10s
            tcp: 30s

deploy:
        lost-worker-timeout: 15s
```

See the dask documentation for more information about changing dask settings.

# 2.27.3 Selecting GPU Platform

The calculation backends alos allows the user to specify the GPU platform and precision level. Users can specify either auto, CUDA or OpenCL as the *preferred\_gpu\_toolkit* using the GPUToolkit enum class. The default precision level is set to mixed but can be overridden by specifying *preferred\_gpu\_precision* using the GPUPrecision enum class:

```
worker_resources = ComputeResources(
   number_of_threads=1,
   number_of_gpus=1,
   preferred_gpu_toolkit=ComputeResources.GPUToolkit.OpenCL,
   preferred_gpu_precision=ComputeResources.GPUPrecision.mixed,
)
```

With GPUToolkit.auto, the framework will determine the fastest available platform based on the precision level:

```
worker_resources = ComputeResources(
   number_of_threads=1,
   number_of_gpus=1,
   preferred_gpu_toolkit=ComputeResources.GPUToolkit.auto,
   preferred_gpu_precision=ComputeResources.GPUPrecision.mixed,
)
```

# 2.28 Storage Backends

A *StorageBackend* is an object used to store data generated as part of property calculations, and to retrieve that data for use in future calculations.

In general, most data stored in a storage backend is stored in two parts:

- A JSON serialized representation of this class (or a subclass), which contains lightweight information such as the state and composition of a system.
- A directory like structure (either directly a directory, or some NetCDF like compressed archive) of ancillary files
  which do not easily lend themselves to be serialized within a JSON object, such as simulation trajectories, whose
  files are referenced by their file name by the data object.

The ancillary directory-like structure is not required if the data may be suitably stored in the data object itself.

# 2.28.1 Data Storage / Retrieval

Each piece of data which is stored in a backend must inherit from the *BaseStoredData* class, will be assigned a unique key. This unique key is both useful for tracking provenance if this data is re-used in future calculations, and also can be used to retrieve the piece of data from the storage system.

In addition to retrieval using the data keys, each backend offers the ability to perform a 'query' to retrieve data which matches a set of given criteria. Data queries are implemented via <code>BaseDataQuery</code> objects, which expose different options for querying for specific types of data (such a simulation data, trained models, etc.).

A query may be used for example to match all simulation data that was generated for a given *Substance* in a particular phase:

```
# Look for all simulation data generated for liquid water
substance_query = SimulationDataQuery()

(continues on next page)
```

(continued from previous page)

```
substance_query.substance = Substance.from_components("0")
substance_query.property_phase = PropertyPhase.Liquid
found_data = backend.query(substance_query)
```

The returned found\_data will be a dictionary with keys of tuples and values as lists of tuples. Each key will be a tuple of the values which were matched, for example the matched thermodynamic state, or the matched substance. For each value tuple in the tuple list, the first item in the tuple is the unique key of the found data object, the second item is the data object itself, and the final object is the file path to the ancillary data directory (or None if none is present).

See the *Data Classes and Queries* page for more information about the available data classes, queries and their details.

# 2.28.2 Implementation

A *StorageBackend* must at minimum implement a structure of:

```
class MyStorageBackend(StorageBackend):
    def _store_object(self, object_to_store, storage_key=None, ancillary_data_path=None):
        ...
    def _retrieve_object(self, storage_key, expected_type=None):
        ...
    def _object_exists(self, storage_key):
        ...
```

where

- \_store\_object() must store a *BaseStoredData* object as well as optionally its ancillary data directory, and return a unique key assigned to that object.
- \_retrieve\_object() must return the <code>BaseStoredData</code> object which has been assigned a given key if the object exists in the system, as well as the file path to ancillary data directory if it exists.
- \_object\_exists() should return whether any object still exists in the storage system with a given key.

All of these methods will be called under a reentrant thread lock and may be considered as thread safe.

# 2.29 Data Classes and Queries

All data which is to be stored within a *StorageBackend* must inherit from the *BaseStoredData* class. More broadly there are typically two types of data which are expected to be stored:

- *HashableStoredData* data which is readily hashable and can be quickly queried for in a storage backend. The prime examples of such data are *ForceFieldData*, whose hash can be easily computed from the file representation of a force field.
- ReplaceableData data which should be replaced in a storage backend when new data of the same type, but which has a higher information content, is stored in the backend. An example of this is when storing a piece of StoredSimulationData in the backend which was generated for a particular Substance and at the same ThermodynamicState as an existing piece of data, but which stores many more uncorrelated configurations.

Every data class **must** be paired with a corresponding data query class which inherits from the <code>BaseDataQuery</code> class. In addition, each data object must implement a <code>to\_storage\_query()</code> function which returns the data query which would uniquely match that data object. The <code>to\_storage\_query()</code> is used heavily by storage backends when checking if a piece of data already exists within the backend.

## 2.29.1 Force Field Data

The ForceFieldData class is used to ForceFieldSource objects within the storage backend. It is a hashable storage object which allows for rapidly checking whether any calculations have been previously been performed for a particular force field source.

It has a corresponding *ForceFieldQuery* class which can be used to query for particular force field sources within a storage backend.

# 2.29.2 Cached Simulation Data

Classes derived from the <code>BaseSimulationData</code> class are used to store the data generated by molecular simulation. The data object primarily records the <code>Substance</code>, <code>PropertyPhase</code> and <code>ThermodynamicState</code> that the simulation was run at, as well as provenance about the calculation and the force field parameters used (as the key of the force field in the storage system).

It has a corresponding <code>BaseSimulationDataQuery</code> class which can be used to query for simulation data which matches a set of particular criteria within a storage backend, which in part includes querying for data collected:

- at a given thermodynamic\_state (i.e temperature and pressure).
- for a given property\_phase (e.g. gas, liquid, liquid+gas coexisting, ...).
- using a given set of force field parameters identified by their unique force\_field\_id assigned by the storage system

Additionally included is not only the ability to find data generated for a particular substance (e.g. only data for methanol), but also the ability to return data for each component of a given substance by setting the *substance\_query* attribute to a *SubstanceQuery* which has the *components\_only* attribute set to true:

```
# Load an existing storage backend
storage_backend = LocalFileStorage()

# Define a system of 50% water and 50% methanol.
full_substance = Substance.from_components("0", "CO")

# Look for all simulation data generated for the full substance
data_query = SimulationDataQuery()

data_query.substance = full_substance
data_query.property_phase = PropertyPhase.Liquid
full_substance_data = storage_backend.query(data_query)

# Now look for all of the pure data which has been stored for both pure
# water and pure methanol.
pure_substance_query = SubstanceQuery()
pure_substance_query.components_only = True
```

(continues on next page)

(continued from previous page)

```
data_query.substance_query = pure_substance_query
component_data = storage_backend.query(data_query)
```

This is particularly useful for when retrieving data for use in the calculation of excess properties (such as the enthalpy of mixing), where such calculations require information about both the full mixture as well as the pure components.

## **Single Simulation Data**

The *StoredSimulationData* class is used to store data generated by a *single* molecular simulation and can be queried for using its accompanying *SimulationDataQuery* query class. In addition to the data stored by the parent *BaseSimulationData* class, this class further stores:

- the number of molecules which were simulated.
- the topology of the simulated system (stored as ancillary data).
- and trajectory of configurations (stored as ancillary data) and observables generated by the simulation.
- the statistic inefficiency of the data.

Data of this kind is considered replaceable, whereby data which has the lowest statistical efficiency is preferred. The philosophy here is that we should store the maximum amount of samples (i.e the maximum number of uncorrelated samples for the property which has the shortest correlation time) which will be useful for future calculations, such that future calculations can simply discard the data which cannot be used (i.e. is likely correlated).

# **Free Energy Data**

The *StoredFreeEnergyData* class is used to store data generated by a free energy calculation which computes the free energy difference between an end and start state. It can be queried for using its accompanying *FreeEnergyDataQuery* query class.

In addition to the data stored by the parent BaseSimulationData class, this class further stores:

- the free energy difference between the end and starting state.
- the topology of the system (stored as ancillary data).
- and trajectory of configurations generated in the starting and end states (stored as ancillary data).

Although data of this kind inherits from the *ReplaceableData* base class, all data deposited in a storage backend will be retained. At this time no situation can be envisaged that the same free energy data from exactly the same calculation will be stored, with the exception of operator errors.

# 2.30 Local File Storage

The *LocalFileStorage* backend stores and retrieves all data objects to / from the local file system. The root directory in which all data is to be stored is defined when the object is created:

```
storage_backend = LocalFileStorage(root_directory="stored_data")
```

All data objects will be stored within this directory as JSON files, with file names of the storage key assigned to that object. If the data object has an associated ancillary data directory, this will be **moved** (not copied) into the root directory and renamed to the storage key when that object is stored into the system.

An example directory created by a local storage backend will look something similar to:

```
root_directory
```

- 1fe615c5cb48429ab77fd71125dec297
  - trajectory.dcd
  - statistics.csv
- 3e15d19e0e614d0491a1a0bc9a51534e
  - trajectory dcd
  - statistics.csv
- 1fe615c5cb48429ab77fd71125dec297.json
- 3e15d19e0e614d0491a1a0bc9a51534e.json
- 0f71f2b4a22042d89d6f0882406869b6.json

where here the backend contains two data objects with ancillary data directories, and one without.

When retrieving data which has an ancillary data directory from the backend, the returned directory path will be the full path to the directory in the root storage directory.

# 2.31 Building the Docs

Although documentation for the OpenFF Evaluator is readily available online, it is sometimes useful to build a local version such as when

- · developing new pages which you wish to preview without having to wait for ReadTheDocs to finish building.
- debugging errors which occur when building on ReadTheDocs.

In these cases, the docs can be built locally by doing the following:

```
git clone https://github.com/openforcefield/openff-evaluator.git
cd openff-evaluator
conda env create --name openff-evaluator-docs --file devtools/conda-envs/docs_env.yaml
conda activate openff-evaluator-docs
rm -rf docs/api docs/_build/html && sphinx-build -b html -j auto docs docs/_build/html
```

The above will yield a new directory named *docs/\_build* which will contain the built html files which can be viewed in your local browser.

# 2.32 API

Documentation for each of the classes contained within the *openff.evaluator* framework.

# 2.32.1 Client Side API

EvaluatorClient	The object responsible for connecting to, and submit-
	ting physical property estimation requests to an Evalua-
	torServer.
BatchMode	The different modes in which a server can batch together
	properties to estimate.
ConnectionOptions	The options to use when connecting to an Evalua-
	torServer
Request	An estimation request which has been sent to a Evalua-
	torServer instance.
RequestOptions	The options to use when requesting a set of physical
	properties be estimated by the server.
RequestResult	The current results of an estimation request - these re-
	sults may be partial if the server hasn't yet completed
	the request.

#### **EvaluatorClient**

class openff.evaluator.client.EvaluatorClient(connection\_options=None)

The object responsible for connecting to, and submitting physical property estimation requests to an *EvaluatorServer*.

## **Examples**

These examples assume that an *EvaluatorServer* has been set up and is running (either synchronously or asynchronously). This server can be connect to be creating an *EvaluatorClient*:

```
>>> from openff.evaluator.client import EvaluatorClient
>>> property_estimator = EvaluatorClient()
```

If the *EvaluatorServer* is not running on the local machine, you will need to specify its address and the port that it is listening on:

```
>>> from openff.evaluator.client import ConnectionOptions
>>>
>>> connection_options = ConnectionOptions(server_address='server_address',
>>> server_port=8000)
>>> property_estimator = EvaluatorClient(connection_options)
```

To asynchronously submit a request to the running server using the default estimation options:

```
>>> # Load in the data set of properties which will be used for comparisons
>>> from openff.evaluator.datasets.thermoml import ThermoMLDataSet
>>> data_set = ThermoMLDataSet.from_doi('10.1016/j.jct.2016.10.001')
>>>
>>> # Filter the dataset to only include densities measured between 130-260 K
>>> from openff.units import unit
>>> from openff.evaluator.properties import Density
>>>
>>> data_set.filter_by_property_types(Density)
(continues on next page)
```

Chapter 2. Supported Physical Properties

(continued from previous page)

The status of the request can be asynchronously queried by calling

```
>>> results = request.results()
```

or the main thread can be blocked until the results are available by calling

```
>>> results = request.results(synchronous=True)
```

How the property set will be estimated can easily be controlled by passing a *RequestOptions* object to the estimate commands.

The calculations layers which will be used to estimate the properties can be controlled for example like so:

Options for how properties should be estimated can be set on a per property, and per layer basis by providing a calculation schema to the options object.

(continued from previous page)

```
>>>
>>>
>>>
property_estimator.request_estimate(
>>> data_set,
>>> force_field_source,
>>> options,
>>> )
```

The gradients of the observables of interest with respect to a number of chosen parameters can be requested by passing a *parameter\_gradient\_keys* parameter. In the below example, gradients will be calculated with respect to both the bond length parameter for the [#6:1]-[#8:2] chemical environment, and the bond angle parameter for the [:1]-[#8:2]-[:3] chemical environment:

```
>>> from openff.evaluator.forcefield import ParameterGradientKey
>>>
>>> parameter_gradient_keys = [
>>>
        ParameterGradientKey('Bonds', '[#6:1]-[#8:2]', 'length')
        ParameterGradientKey('Angles', '[*:1]-[#8:2]-[*:3]', 'angle')
>>>
>>> ]
>>>
>>> property_estimator.request_estimate(
>>>
        data_set,
>>>
        force_field_source,
>>>
        options.
        parameter_gradient_keys
>>>
>>> )
```

\_\_init\_\_(connection\_options=None)

**Parameters connection\_options** (ConnectionOptions, optional) — The options used when connecting to the calculation server. If *None*, default options are used.

# **Methods**

init([connection_options])	param connection_options The options used when connecting to the calculation
default_request_options(data_set,)	Returns the default <i>RequestOptions</i> options used to estimate a set of properties if <i>None</i> are provided.
request_estimate(property_set,[,])	Submits a request for the <i>EvaluatorServer</i> to attempt to estimate the data set of physical properties using the specified force field parameters according to the provided options.
retrieve_results(request_id[, synchronous,])	Retrieves the current results of a request from the server.

### **Attributes**

server_address	The address of the server that this client is connected
	to.
server_port	The port of the server that this client is connected to.

# property server\_address

The address of the server that this client is connected to.

Type str

# property server\_port

The port of the server that this client is connected to.

Type int

#### static default\_request\_options(data set, force field source)

Returns the default RequestOptions options used to estimate a set of properties if None are provided.

#### **Parameters**

- data\_set (PhysicalPropertyDataSet) The data set which would be estimated.
- **force\_field\_source** (ForceFieldSource) The force field parameters which will be used by the request.

**Returns** The default options.

Return type RequestOptions

request\_estimate(property\_set, force\_field\_source, options=None, parameter\_gradient\_keys=None)

Submits a request for the *EvaluatorServer* to attempt to estimate the data set of physical properties using the specified force field parameters according to the provided options.

### **Parameters**

- **property\_set** (PhysicalPropertyDataSet) The set of properties to estimate.
- **force\_field\_source** (ForceFieldSource *or ForceField*) The force field parameters to estimate the properties using.
- **options** (RequestOptions, optional) A set of estimator options. If *None* default options will be used (see *default\_request\_options*).
- parameter\_gradient\_keys (list of ParameterGradientKey, optional) A list of the parameters that the physical properties should be differentiated with respect to.

#### **Returns**

- Request An object which will provide access to the results of this request.
- EvaluatorException, optional Any exceptions raised while attempting the submit the request.

**retrieve\_results**(request\_id, synchronous=False, polling\_interval=5)

Retrieves the current results of a request from the server.

#### **Parameters**

- request\_id (str) The server assigned id of the request.
- **synchronous** (*bool*) If true, this method will block the main thread until the server either returns a result or an error.

• **polling\_interval** (*float*) – If running synchronously, this is the time interval (seconds) between checking if the request has completed.

#### Returns

- RequestResult, optional Returns the current results of the request. This may be None if any unexpected exceptions occurred while retrieving the estimate.
- EvaluatorException, optional The exception raised will trying to retrieve the result, if any.

#### **BatchMode**

class openff.evaluator.client.BatchMode(value)

The different modes in which a server can batch together properties to estimate.

This enum may take values of

- SameComponents: All properties measured for substances containing exactly the same components will be placed into a single batch. E.g. The density of a 80:20 and a 20:80 mix of ethanol and water would be batched together, but the density of pure ethanol and the density of pure water would be placed into separate batches.
- SharedComponents: All properties measured for substances containing at least common component will be batched together. E.g.The densities of 80:20 and 20:80 mixtures of ethanol and water, and the pure densities of ethanol and water would be batched together.

Properties will only be marked as estimated by the server when all properties in a single batch are completed.

```
__init__()
```

# **Attributes**

 ${\tt Same Components}$ 

SharedComponents

# **ConnectionOptions**

**class** openff.evaluator.client.**ConnectionOptions**(server\_address=None, server\_port=None)

The options to use when connecting to an EvaluatorServer

```
__init__(server_address=None, server_port=None)
```

# **Parameters**

- **server\_address** (*str*) The address of the server to connect to.
- **server\_port** (*int*) The port of the server to connect to.

# **Methods**

init([server_address, server_port])	
	<b>param server_address</b> The address of the server to connect to.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

#### **Attributes**

server_address	The address of the server to connect to.
server_port	The port of the server to connect to.

#### server\_address

The address of the server to connect to. The default value of this attribute is localhost.

Type str

# server\_port

The port of the server to connect to. The default value of this attribute is 8000.

Type int

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

# json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# Request

class openff.evaluator.client.Request(client=None)

An estimation request which has been sent to a *EvaluatorServer* instance.

This object can be used to query and retrieve the results of the request when finished, or be stored to retrieve the request at some point in the future.

\_\_init\_\_(client=None)

**Parameters client** (EvaluatorClient, optional) — The client which submitted this request.

# **Methods**

init([client])	<b>param client</b> The client which submitted this request.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>results([synchronous, polling_interval])</pre>	Attempt to retrieve the results of the request from the
	server.
validate([attribute type])	Validate the values of the attributes.

### **Attributes**

connection_options	The options used to connect to the server handling
	the request.
id	The unique id assigned to this request by the server.

#### id

The unique id assigned to this request by the server. The default value of this attribute is not set and must be set by the user..

Type str

# connection\_options

The options used to connect to the server handling the request. The default value of this attribute is not set and must be set by the user..

Type ConnectionOptions

**results**(synchronous=False, polling\_interval=5)

Attempt to retrieve the results of the request from the server.

If the method is run synchronously it will block the main thread either all of the requested properties have been estimated, or an exception is returned.

#### **Parameters**

- **synchronous** (*bool*) If *True*, this method will block the main thread until the server either returns a result or an error.
- **polling\_interval** (*float*) If running synchronously, this is the time interval (seconds) between checking if the calculation has finished. This will be ignored if running asynchronously.

# Returns

- RequestResult, optional Returns the current results of the request. This may be None if any unexpected exceptions occurred while retrieving the estimate.
- EvaluatorException, optional The exception raised will trying to retrieve the result if any.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

**Return type** Any

validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### RequestOptions

# class openff.evaluator.client.RequestOptions

The options to use when requesting a set of physical properties be estimated by the server.

\_\_init\_\_()

# Methods

init()	
add_schema(layer_type, property_type, schema)	A convenience function for adding a calculation schema to the schema dictionary.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

batch_mode	The way in which the server should batch together
	properties to estimate.
calculation_layers	The calculation layers which may be used to estimate
	the set of physical properties.
calculation_schemas	The schemas that each calculation layer should use
	when estimating the set of physical properties.

### calculation\_layers

The calculation layers which may be used to estimate the set of physical properties. The order in which the layers appears in this list determines the order in which the layers will attempt to estimate the data set. The default value of this attribute is ['ReweightingLayer', 'SimulationLayer'].

Type *list* 

#### calculation\_schemas

The schemas that each calculation layer should use when estimating the set of physical properties. The dictionary should be of the form [property\_type][layer\_type]. The default value of this attribute is not set. This attribute is *optional*.

Type dict

### batch\_mode

The way in which the server should batch together properties to estimate. Properties will only be marked as finished when all properties in a single batch are completed. The default value of this attribute is BatchMode. SharedComponents. This attribute is *optional*.

Type BatchMode

### add\_schema(layer\_type, property\_type, schema)

A convenience function for adding a calculation schema to the schema dictionary.

#### Parameters

- layer\_type (str or type of CalculationLayer) The layer to associate the schema with.
- **property\_type** (*str or type of PhysicalProperty*) The class of property to associate the schema with.
- schema (CalculationSchema) The schema to add.

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

# RequestResult

### class openff.evaluator.client.RequestResult

The current results of an estimation request - these results may be partial if the server hasn't yet completed the request.

\_\_init\_\_()

### **Methods**

init()	
from icon(file path)	Create this object from a JSON file.
from_json(file_path)	<u> </u>
get_attributes([attribute_type])	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

estimated_properties	The set of properties which have been successfully
	estimated.
exceptions	The set of properties which have yet to be, or are cur-
	rently being estimated.
queued_properties	The set of properties which have yet to be, or are cur-
	rently being estimated.
unsuccessful_properties	The set of properties which could not be successfully
	estimated.

## queued\_properties

The set of properties which have yet to be, or are currently being estimated.

Type PhysicalPropertyDataSet

# estimated\_properties

The set of properties which have been successfully estimated.

Type PhysicalPropertyDataSet

# unsuccessful\_properties

The set of properties which could not be successfully estimated.

Type PhysicalPropertyDataSet

### exceptions

The set of properties which have yet to be, or are currently being estimated. The default value of this attribute is [7].

Type list

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### classmethod from\_json(file path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

#### **Exceptions**

**EvaluatorException** 

A serializable wrapper around an *Exception*.

# **EvaluatorException**

**exception** openff.evaluator.utils.exceptions.**EvaluatorException**(message=None)

A serializable wrapper around an Exception.

# classmethod from\_exception(exception)

Initialize this class from an existing exception.

Parameters exception (Exception) – The existing exception

**Returns** The initialized exception object.

Return type cls

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

```
classmethod parse_json(string_contents)
```

Parses a typed json string into the corresponding class structure.

```
Parameters string_contents (str or bytes) – The typed json string.
```

**Returns** The parsed class.

**Return type** Any

### with\_traceback()

Exception.with\_traceback(tb) - set self.\_\_traceback\_\_ to tb and return self.

# 2.32.2 Server Side API

EvaluatorServer	The object responsible for coordinating all properties estimations to be ran using the openff-evaluator frame-
	work.
Batch	Represents a batch of physical properties which are be-
	ing estimated by the server for a given set of force field
	parameters.

### **EvaluatorServer**

The object responsible for coordinating all properties estimations to be ran using the openff-evaluator framework.

This server is responsible for receiving estimation requests from the client, determining which calculation layer to use to launch the request, and distributing that estimation across the available compute resources.

#### **Notes**

Every client request is split into logical chunk batches. This enables batches of related properties (e.g. all properties for CO) to be estimated in one go (or one task graph in the case of workflow based layers) and returned when ready, rather than waiting for the full data set to complete.

### **Examples**

Setting up a general server instance using a dask based calculation backend, and a local file storage backend:

```
>>> # Create the backend which will be responsible for distributing the calculations
>>> from openff.evaluator.backends.dask import DaskLocalCluster
>>> calculation_backend = DaskLocalCluster()
>>> calculation_backend.start()
>>>
>>> # Create the server to which all estimation requests will be submitted
>>> from openff.evaluator.server import EvaluatorServer
>>> property_server = EvaluatorServer(calculation_backend)
>>>
>>> # Instruct the server to listen for incoming requests

(continues on next page)
```

(continued from previous page)

```
>>> # This command will run until killed.
>>> property_server.start()
```

\_\_init\_\_(calculation\_backend, storage\_backend=None, port=8000, working\_directory='working-data', enable\_data\_caching=True, delete\_working\_files=True)

Constructs a new EvaluatorServer object.

#### **Parameters**

- calculation\_backend (CalculationBackend) The backend to use for executing calculations.
- **storage\_backend** (StorageBackend, optional) The backend to use for storing information from any calculations. If *None*, a default *LocalFileStorage* backend will be used.
- port (int) The port on which to listen for incoming client requests.
- working\_directory (str) The local directory in which to store all local, temporary calculation data.
- **enable\_data\_caching** (*boo1*) Whether the server should attempt to cache any data, mainly the output of simulations, produced by estimation requests for future re-processing (e.g for reweighting).
- **delete\_working\_files** (*bool*) Whether to delete the working files produced while estimated a batch of properties using a specific calculation layer.

#### **Methods**

init(calculation_backend[,])	Constructs a new EvaluatorServer object.
start([asynchronous])	Instructs the server to begin listening for incoming requests from any <i>EvaluatorClients</i> .
stop()	Stops the property calculation server and it's provided backend.

### start(asynchronous=False)

Instructs the server to begin listening for incoming requests from any EvaluatorClients.

**Parameters asynchronous** (*boo1*) – If *True* the server will run on a separate thread in the background, returning control back to the main thread. Otherwise, this function will block the main thread until this server is killed.

# stop()

Stops the property calculation server and it's provided backend.

# **Batch**

### class openff.evaluator.server.Batch

Represents a batch of physical properties which are being estimated by the server for a given set of force field parameters.

The expectation is that this object will be passed between calculation layers, whereby each layer will attempt to estimate each of the *queued\_properties*. Those properties which can be estimated will be moved to the *estimated\_properties* set, while those that couldn't will remain in the *queued\_properties* set ready for the next layer.

\_\_init\_\_()

# **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

anabla data cachina	Whather the correspond estampt to each any data
enable_data_caching	Whether the server should attempt to cache any data,
	mainly the output of simulations, produced by this
	batch for future re-processing (e.g for reweighting).
estimated_properties	The set of properties which have been successfully
	estimated.
exceptions	The set of properties which have yet to be, or are cur-
	rently being estimated.
force_field_id	The id of the force field being used to estimatethis
	batch of properties.
id	The unique id of this batch.
options	The options being used to estimate this batch.
parameter_gradient_keys	The parameters that this batch of physical properties
	should be differentiated with respect to.
queued_properties	The set of properties which have yet to be estimated.
unsuccessful_properties	The set of properties which have been could not be
	estimated.

# id

The unique id of this batch.

Type str

# force\_field\_id

The id of the force field being used to estimate this batch of properties. The default value of this attribute is not set and must be set by the user..

```
Type str
```

#### options

The options being used to estimate this batch. The default value of this attribute is not set and must be set by the user..

Type RequestOptions

# parameter\_gradient\_keys

The parameters that this batch of physical properties should be differentiated with respect to. The default value of this attribute is not set and must be set by the user..

Type *list* 

# enable\_data\_caching

Whether the server should attempt to cache any data, mainly the output of simulations, produced by this batch for future re-processing (e.g for reweighting). The default value of this attribute is True.

Type bool

### queued\_properties

The set of properties which have yet to be estimated. The default value of this attribute is [].

Type list

### estimated\_properties

The set of properties which have been successfully estimated. The default value of this attribute is [].

Type list

### unsuccessful\_properties

The set of properties which have been could not be estimated. The default value of this attribute is [].

Type list

#### exceptions

The set of properties which have yet to be, or are currently being estimated. The default value of this attribute is [7].

Type list

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** *Any* 

# 2.32.3 Physical Property API

PhysicalProperty	Represents the value of any physical property and it's uncertainty if provided.
PropertyPhase	An enum describing the phase that a property was col-
	lected in.
Source	Container class for information about how a property
	was measured / calculated.
CalculationSource	Contains any metadata about how a physical property
	was calculated.
MeasurementSource	Contains any metadata about how a physical property
	was measured by experiment.

### **PhysicalProperty**

**class** openff.evaluator.datasets.**PhysicalProperty**(thermodynamic\_state=None,

phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Represents the value of any physical property and it's uncertainty if provided.

It additionally stores the thermodynamic state at which the property was collected, the phase it was collected in, information about the composition of the observed system, and metadata about how the property was collected.

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

#### **Parameters**

• **thermodynamic\_state** (ThermodynamicState) — The thermodynamic state that the property was measured in.

- phase (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.
- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

### **Methods**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

# abstract classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

id

A unique identifier string assigned to this property

Type str

# metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

### thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

#### phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

Type PropertyPhase

#### substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

#### uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

### gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type *list* 

#### source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

# **PropertyPhase**

# class openff.evaluator.datasets.PropertyPhase(value)

An enum describing the phase that a property was collected in.

# **Examples**

Properties measured in multiple phases (e.g. enthalpies of vaporization) can be defined be concatenating *PropertyPhase* enums:

```
>>> gas_liquid_phase = PropertyPhase.Gas | PropertyPhase.Liquid
```

\_\_init\_\_()

#### **Methods**

from\_string(enum\_string)

Parses a phase enum from its string representation.

# **Attributes**

Undefined			
Solid			
Liquid			
Gas			

## classmethod from\_string(enum\_string)

Parses a phase enum from its string representation.

**Parameters enum\_string** (str) – The str representation of a *PropertyPhase* 

Returns The created enum

Return type PropertyPhase

# **Examples**

To round-trip convert a phase enum: >>> phase = PropertyPhase.Liquid | PropertyPhase.Gas >>> phase\_str = str(phase) >>> parsed\_phase = PropertyPhase.from\_string(phase\_str)

### Source

# class openff.evaluator.datasets.Source

Container class for information about how a property was measured / calculated.

**Todo:** Swap this out with a more general provenance class.

\_\_init\_\_()

### **Methods**

Create this object from a JSON file.
Creates a JSON representation of this class.
Parses a typed json string into the corresponding class structure.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

#### **CalculationSource**

class openff.evaluator.datasets.CalculationSource(fidelity=None, provenance=None)

Contains any metadata about how a physical property was calculated.

This includes at which fidelity the property was calculated at (e.g Direct simulation, reweighting,  $\dots$ ) in addition to the parameters which were used as part of the calculations.

### fidelity

The fidelity at which the property was calculated

Type str

#### provenance

A dictionary containing information about how the property was calculated.

Type dict of str and Any

**\_\_init\_\_**(fidelity=None, provenance=None)

Constructs a new CalculationSource object.

#### **Parameters**

- **fidelity** (str) The fidelity at which the property was calculated
- **provenance** (*dict of str and Any*) A dictionary containing information about how the property was calculated.

# **Methods**

<pre>init([fidelity, provenance])</pre>	Constructs a new CalculationSource object.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

#### MeasurementSource

# class openff.evaluator.datasets.MeasurementSource(doi=", reference=")

Contains any metadata about how a physical property was measured by experiment.

This class contains either the DOI and/or the reference, but must contain at least one as the observable must have a source, even if it was measured in lab.

### doi

The DOI for the source, preferred way to identify for source

Type str or None, default None

#### reference

The long form description of the source if no DOI is available, or more information is needed or wanted.

```
Type str
```

```
__init__(doi=", reference=")
```

Constructs a new MeasurementSource object.

### **Parameters**

- doi(str or None, default None) The DOI for the source, preferred way to identify for source
- **reference** (*str*) The long form description of the source if no DOI is available, or more information is needed or wanted.

# **Methods**

init([doi, reference])	Constructs a new MeasurementSource object.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

 $\begin{tabular}{ll} \textbf{Parameters} & \textbf{string\_contents} & (str & or & bytes) - \textbf{The typed json string}. \end{tabular}$ 

Returns The parsed class.

**Return type** Any

# **Built-in Properties**

Density	A class representation of a density property
ExcessMolarVolume	A class representation of an excess molar volume prop-
	erty
DielectricConstant	A class representation of a dielectric property
EnthalpyOfMixing	A class representation of an enthalpy of mixing property
EnthalpyOfVaporization	A class representation of an enthalpy of vaporization
	property
SolvationFreeEnergy	A class representation of a solvation free energy prop-
	erty.
HostGuestBindingAffinity	A class representation of a host-guest binding affinity
	property

# **Density**

A class representation of a density property

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

#### **Parameters**

- **thermodynamic\_state** (ThermodynamicState) The thermodynamic state that the property was measured in.
- **phase** (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.
- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

#### **Methods**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
default_reweighting_schema([])	Returns the default calculation schema to use when
	estimating this property by reweighting existing data.
<pre>default_simulation_schema([])</pre>	Returns the default calculation schema to use when
	estimating this class of property from direct simula-
	tions.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

#### **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

### classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

static default\_simulation\_schema(absolute\_tolerance=UNDEFINED,

relative\_tolerance=UNDEFINED, n\_molecules=1000)  $\rightarrow$  SimulationSchema

Returns the default calculation schema to use when estimating this class of property from direct simulations.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_molecules** (*int*) The number of molecules to use in the simulation.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

static default\_reweighting\_schema(absolute tolerance=UNDEFINED,

relative\_tolerance=UNDEFINED, n\_effective\_samples=50)  $\rightarrow$  ReweightingSchema

Returns the default calculation schema to use when estimating this property by reweighting existing data.

### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_effective\_samples** (*int*) The minimum number of effective samples to require when reweighting the cached simulation data.

**Returns** The schema to follow when estimating this property.

Return type ReweightingSchema

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type list

id

A unique identifier string assigned to this property

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

### phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

Type PropertyPhase

#### source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

#### substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

# thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

### uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

#### **ExcessMolarVolume**

A class representation of an excess molar volume property

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

### **Parameters**

- **thermodynamic\_state** (ThermodynamicState) The thermodynamic state that the property was measured in.
- phase (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.

- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

### **Methods**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
<pre>default_reweighting_schema([])</pre>	Returns the default calculation schema to use when
	estimating this class of property by re-weighting
	cached simulation data.
default_simulation_schema([])	Returns the default calculation schema to use when
	estimating this class of property from direct simula-
	tions.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

# **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

# classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

 ${\bf classmethod\ default\_reweighting\_schema} (absolute\_tolerance:\ {\it \sim}pint.util.Quantity =$ 

<openff.evaluator.attributes.attributes.UndefinedAttribute object>, relative\_tolerance: float = <openff.evaluator.attributes.attributes.UndefinedAttribute object>, n\_effective\_samples: int = 50)  $\rightarrow$  ReweightingSchema

Returns the default calculation schema to use when estimating this class of property by re-weighting cached simulation data.

#### **Parameters**

- **absolute\_tolerance** The absolute tolerance to estimate the property to within.
- **relative\_tolerance** The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- n\_effective\_samples The minimum number of effective samples to require when reweighting the cached simulation data.

**Return type** The default re-weighting calculation schema.

 ${\bf classmethod\ default\_simulation\_schema} (absolute\_tolerance = < openff. evaluator. attributes. Undefined Attributes object>, rela-$ 

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, n\_molecules=1000)  $\rightarrow$  SimulationSchema

Returns the default calculation schema to use when estimating this class of property from direct simulations.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_molecules** (*int*) The number of molecules to use in the simulation.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type *list* 

id

A unique identifier string assigned to this property

Type str

#### json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

### phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

**Type** PropertyPhase

# source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

#### substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

#### thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

#### uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

#### **DielectricConstant**

A class representation of a dielectric property

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

#### **Parameters**

- **thermodynamic\_state** (ThermodynamicState) The thermodynamic state that the property was measured in.
- **phase** (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.
- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

# **Methods**

	Constructs and Dhanical Duaments alsiant
init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
<pre>default_reweighting_schema([])</pre>	Returns the default calculation schema to use when
	estimating this property by reweighting existing data.
<pre>default_simulation_schema([])</pre>	Returns the default calculation schema to use when
	estimating this class of property from direct simula-
	tions.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

### classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

static default\_simulation\_schema(absolute\_tolerance=UNDEFINED,

relative\_tolerance=UNDEFINED, n\_molecules=1000)

Returns the default calculation schema to use when estimating this class of property from direct simulations.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_molecules** (*int*) The number of molecules to use in the simulation.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

# ${\tt static \ default\_reweighting\_schema} (absolute\_tolerance = UNDEFINED,$

relative tolerance=UNDEFINED, n effective samples=50)

Returns the default calculation schema to use when estimating this property by reweighting existing data.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_effective\_samples** (*int*) The minimum number of effective samples to require when reweighting the cached simulation data.

**Returns** The schema to follow when estimating this property.

Return type ReweightingSchema

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type list

id

A unique identifier string assigned to this property

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

Type PropertyPhase

#### source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

#### substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

# thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

#### uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

# **EnthalpyOfMixing**

A class representation of an enthalpy of mixing property

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

## **Parameters**

- **thermodynamic\_state** (ThermodynamicState) The thermodynamic state that the property was measured in.
- **phase** (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.

- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

## **Methods**

Constructs a new PhysicalProperty object.
Returns the default calculation schema to use when
estimating this class of property by re-weighting
cached simulation data.
Returns the default calculation schema to use when
estimating this class of property from direct simula-
tions.
openff.evaluator.unit.Unit: The default unit (e.g.
Create this object from a JSON file.
Returns all attributes of a specific attribute_type.
Creates a JSON representation of this class.
Parses a typed json string into the corresponding class
structure.
Validate the values of the attributes.

# **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

# classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

 $\textbf{classmethod default\_reweighting\_schema} (absolute\_tolerance: ~pint.util.Quantity = \\$ 

<openff.evaluator.attributes.attributes.UndefinedAttribute
object>, relative\_tolerance: float =
<openff.evaluator.attributes.attributes.UndefinedAttribute
object>, n\_effective\_samples: int = 50)  $\rightarrow$ ReweightingSchema

Returns the default calculation schema to use when estimating this class of property by re-weighting cached simulation data.

#### **Parameters**

- **absolute\_tolerance** The absolute tolerance to estimate the property to within.
- **relative\_tolerance** The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_effective\_samples** The minimum number of effective samples to require when reweighting the cached simulation data.

**Return type** The default re-weighting calculation schema.

 ${\bf classmethod\ default\_simulation\_schema} (absolute\_tolerance = < openff. evaluator. attributes. Undefined Attributes object>, rela-$ 

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, n\_molecules=1000)  $\rightarrow$  SimulationSchema

Returns the default calculation schema to use when estimating this class of property from direct simulations.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_molecules** (*int*) The number of molecules to use in the simulation.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type list

id

A unique identifier string assigned to this property

Type str

#### json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

### phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

**Type** PropertyPhase

# source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

# substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

#### thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

#### uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

#### Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

# **EnthalpyOfVaporization**

A class representation of an enthalpy of vaporization property

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

#### **Parameters**

- **thermodynamic\_state** (ThermodynamicState) The thermodynamic state that the property was measured in.
- **phase** (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.
- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

# **Methods**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
<pre>default_reweighting_schema([])</pre>	Returns the default calculation schema to use when
	estimating this property by reweighting existing data.
<pre>default_simulation_schema([])</pre>	Returns the default calculation schema to use when
	estimating this class of property from direct simula-
	tions.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.
	·

#### **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

### classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

 ${\tt static \ default\_simulation\_schema} ({\it absolute\_tolerance} = {\it UNDEFINED},$ 

relative\_tolerance=UNDEFINED, n\_molecules=1000)

Returns the default calculation schema to use when estimating this class of property from direct simulations.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- **n\_molecules** (*int*) The number of molecules to use in the simulation.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

Returns the default calculation schema to use when estimating this property by reweighting existing data.

## **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.
- n\_effective\_samples (int) The minimum number of effective samples to require when reweighting the cached simulation data.

**Returns** The schema to follow when estimating this property.

Return type ReweightingSchema

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type list

id

A unique identifier string assigned to this property

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

Type PropertyPhase

#### source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

#### substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

# thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

#### uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

## SolvationFreeEnergy

A class representation of a solvation free energy property.

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

## **Parameters**

- **thermodynamic\_state** (ThermodynamicState) The thermodynamic state that the property was measured in.
- phase (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.

- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

#### **Methods**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
<pre>default_simulation_schema([])</pre>	Returns the default calculation schema to use when
	estimating this class of property from direct simula-
	tions.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

#### **Attributes**

gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.

# classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

static default\_simulation\_schema(absolute\_tolerance=UNDEFINED,

relative\_tolerance=UNDEFINED, n\_molecules=2000)

Returns the default calculation schema to use when estimating this class of property from direct simulations.

#### **Parameters**

- absolute\_tolerance (openff.evaluator.unit.Quantity, optional) The absolute tolerance to estimate the property to within.
- **relative\_tolerance** (*float*) The tolerance (as a fraction of the properties reported uncertainty) to estimate the property to within.

• **n\_molecules** (*int*) – The number of molecules to use in the simulation.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

#### gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type list

id

A unique identifier string assigned to this property

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

#### metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

### phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

```
Type PropertyPhase
```

#### source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

Type Source

#### substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

## thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

## uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

## HostGuestBindingAffinity

A class representation of a host-guest binding affinity property

\_\_init\_\_(thermodynamic\_state=None, phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

Constructs a new PhysicalProperty object.

#### **Parameters**

• **thermodynamic\_state** (ThermodynamicState) — The thermodynamic state that the property was measured in.

- **phase** (PropertyPhase) The phase that the property was measured in.
- **substance** (Substance) The composition of the substance that was measured.
- value (openff.evaluator.unit.Quantity) The value of the measured physical property.
- uncertainty (openff.evaluator.unit.Quantity) The uncertainty in the measured value.
- **source** (Source) The source of this property.

#### **Methods**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
<pre>default_paprika_schema([existing_schema,])</pre>	Returns the default calculation schema to use when
	estimating a host-guest binding affinity measurement
	with an APR calculation using the paprika package.
<pre>default_unit()</pre>	openff.evaluator.unit.Unit: The default unit (e.g.
<pre>default_yank_schema([existing_schema])</pre>	Returns the default calculation schema to use when
	estimating this class of property from direct simula-
	tions.
<pre>from_json(file_path)</pre>	
<pre>from_json(file_path) get_attributes([attribute_type])</pre>	tions.
	tions. Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	tions.  Create this object from a JSON file.  Returns all attributes of a specific attribute_type.
<pre>get_attributes([attribute_type]) json([file_path, format])</pre>	tions.  Create this object from a JSON file.  Returns all attributes of a specific <i>attribute_type</i> .  Creates a JSON representation of this class.

## **Attributes**

	The and it are after a second and a second to differ
gradients	The gradients of this property with respect to differ-
	ent force field parameters.
id	A unique identifier string assigned to this property
metadata	Additional metadata associated with this property.
phase	The phase / phases that this property was measured
	in.
source	The original source of this physical property.
substance	The substance that this property was measured esti-
	mated for.
thermodynamic_state	The thermodynamic state that this propertywas mea-
	sured / estimated at.
uncertainty	The uncertainty in measured / estimated value of this
	property.
value	The measured / estimated value of this property.
-	

# classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

# static default\_yank\_schema(existing\_schema=None)

Returns the default calculation schema to use when estimating this class of property from direct simulations.

**Parameters existing\_schema** (SimulationSchema, optional) — An existing schema whose settings to use. If set, the schema's workflow\_schema will be overwritten by this method.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

classmethod default\_paprika\_schema(existing\_schema: ~typ-

```
ing.Optional[~openff.evaluator.layers.simulation.SimulationSchema] = None, n_solvent_molecules: int = 2500, n_thermalization_steps: int = 50000, n_equilibration_steps: int = 200000, n_production_steps: int = 2500000, dt_thermalization: ~pint.util.Quantity = <Quantity(1.0, 'femtosecond')>, dt_equilibration: ~pint.util.Quantity = <Quantity(2.0, 'femtosecond')>, dt_production: ~pint.util.Quantity = <Quantity(2.0, 'femtosecond')>, debug: bool = False)
```

Returns the default calculation schema to use when estimating a host-guest binding affinity measurement with an APR calculation using the paprika package.

#### **Notes**

This schema requires additional metadata to be able to estimate each metadata. This metadata is automatically generated for properties loaded from the taproom package using the TaproomDataSet object.

### **Parameters**

- **existing\_schema** (SimulationSchema, *optional*) An existing schema whose settings to use. If set, the schema's *workflow\_schema* will be overwritten by this method.
- n\_solvent\_molecules The number of solvent molecules to add to the box.
- n\_thermalization\_steps The number of thermalization simulations steps to perform. Sample generated during this step will be discarded.
- n\_equilibration\_steps The number of equilibration simulations steps to perform. Sample generated during this step will be discarded.
- **n\_production\_steps** The number of production simulations steps to perform. Sample generated during this step will be used in the final free energy calculation.
- **dt\_thermalization** The integration timestep during thermalization
- **dt\_equilibration** The integration timestep during equilibration
- **dt\_production** The integration timestep during production
- **debug** Whether to return a debug schema. This is nearly identical to the default schema, albeit with significantly less solvent molecules (10), all simulations run in NVT and much shorter simulation runs (500 steps). If True, the other input arguments will be ignored.

**Returns** The schema to follow when estimating this property.

Return type SimulationSchema

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# gradients

The gradients of this property with respect to different force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type *list* 

id

A unique identifier string assigned to this property

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### metadata

Additional metadata associated with this property. All property metadata will be made accessible to estimation workflows. The default value of this attribute is not set. This attribute is *optional*.

Type dict

#### classmethod parse\_json(string contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

## phase

The phase / phases that this property was measured in. The default value of this attribute is not set and must be set by the user..

**Type** PropertyPhase

## source

The original source of this physical property. The default value of this attribute is not set. This attribute is *optional*.

# Type Source

## substance

The substance that this property was measured estimated for. The default value of this attribute is not set and must be set by the user..

Type Substance

# thermodynamic\_state

The thermodynamic state that this propertywas measured / estimated at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

# uncertainty

The uncertainty in measured / estimated value of this property. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

The measured / estimated value of this property. The default value of this attribute is not set and must be set by the user..

Type Quantity

## **Substance Definition**

Substance	Defines the components, their amounts, and their roles
	in a system.
Component	Defines a single component in a chemical system, as well
	as it's role within the system (if any).
Amount	A representation of the amount of a given component in
	a Substance.
ExactAmount	The exact number of instances of a Component in a Sub-
	stance.
MoleFraction	The mole fraction of a <i>Component</i> in a <i>Substance</i> .

# **Substance**

# class openff.evaluator.substances.Substance

Defines the components, their amounts, and their roles in a system.

# **Examples**

A neat liquid containing only a single component:

```
>>> from openff.evaluator.substances import Component, ExactAmount, MoleFraction
>>> liquid = Substance()
>>> liquid.add_component(Component(smiles='0'), MoleFraction(1.0))
```

A binary mixture containing two components, where the mole fractions are explicitly stated:

```
>>> binary_mixture = Substance()
>>> binary_mixture.add_component(Component(smiles='0'), MoleFraction(0.2))
>>> binary_mixture.add_component(Component(smiles='CO'), MoleFraction(0.8))
```

The infinite dilution of one molecule within a bulk solvent or mixture may also be specified by defining the exact number of copies of that molecule, rather than a mole fraction:

```
>>> benzene = Component(smiles='C1=CC=CC=C1', role=Component.Role.Solute)
>>> water = Component(smiles='0', role=Component.Role.Solvent)
>>>
>>> infinite_dilution = Substance()
>>> infinite_dilution.add_component(component=benzene, amount=ExactAmount(1)) #__
-Infinite_dilution.
>>> infinite_dilution.add_component(component=water, amount=MoleFraction(1.0))
```

In this example we explicitly flag benzene as being the solute and the water component the solvent. This enables workflow's to easily identify key molecules of interest, such as the molecule which should be 'grown' into solution during solvation free energy calculations.

```
__init__()
```

### Methods

init()	
add_component(component, amount)	Add a component to the Substance.
<pre>calculate_aqueous_ionic_mole_fraction()</pre>	Determines what mole fraction of ions is needed to yield
<pre>from_components(*components)</pre>	Creates a new <i>Substance</i> object from a list of components.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_amounts(component)</pre>	Returns the amounts of the component in this sub-
	stance.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
<pre>get_molecules_per_component(maximum_molecules_per_component)</pre>	uRe)urns the number of molecules for each component
	in this substance, given a maximum total number of
	molecules.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

amounts	the amounts of the component in this substance This
	attribute is <i>read-only</i> .
components	A list of all of the components in this substance.
identifier	A unique str representation of this substance, which
	encodes all components and their amounts in the sub-
	stance.
number_of_components	The number of different components in this sub-
	stance.

#### components

A list of all of the components in this substance. The default value of this attribute is (). This attribute is *read-only*.

Type tuple

## amounts

the amounts of the component in this substance This attribute is read-only.

Type dict

#### property identifier

A unique str representation of this substance, which encodes all components and their amounts in the substance.

Type str

# property number\_of\_components

The number of different components in this substance.

Type int

# classmethod from\_components(\*components)

Creates a new *Substance* object from a list of components. This method assumes that all components should be present with equal mole fractions.

**Parameters components** (Component *or str*) – The components to add to the substance. These may either be full *Component* objects or just the smiles representation of the component.

**Returns** The substance containing the requested components in equal amounts.

Return type Substance

### add\_component(component, amount)

Add a component to the Substance. If the component is already present in the substance, then the mole fraction will be added to the current mole fraction of that component.

# **Parameters**

- **component** (Component) The component to add to the system.
- amount (Amount) The amount of this component in the substance.

### get\_amounts(component)

Returns the amounts of the component in this substance.

**Parameters component** (*str or* Component) – The component (or it's identifier) to retrieve the amount of.

**Returns** The amounts of the component in this substance.

Return type tuple of Amount

Returns the number of molecules for each component in this substance, given a maximum total number of molecules.

#### **Parameters**

- maximum\_molecules (int) The maximum number of molecules.
- **tolerance** (*float*, *optional*) The tolerance within which this amount should be represented. As an example, when converting a mole fraction into a number of molecules, the total number of molecules may not be sufficiently large enough to reproduce this amount.
- **count\_exact\_amount** (*bool*) Whether components present in an exact amount (i.e. defined with an ExactAmount) should be considered when apply the maximum number
  - of molecules constraint. This may be set false, for example, when building a separate solvated protein (n = 1) and solvated protein + ligand complex (n = 2) system but wish for both systems to have the same number of solvent molecules.
- **truncate\_n\_molecules** (*bool*) Whether or not to attempt to truncate the number of molecules in the substance if the total number is over the specified maximum. If False, an exception will be raised in this case.

The truncation works by iteratively removing one molecule of the predominant component up to a limit of removing a total number of molecules equal to the number of components in the substance (e.g. for a binary substance a maximum of two molecules can be removed). An exception is raised if the number of molecules cannot be sensibly truncated.

**Returns** A dictionary of molecule counts per component, where each key is a component identifier.

**Return type** dict of str and int

static calculate\_aqueous\_ionic\_mole\_fraction(ionic\_strength)

Determines what mole fraction of ions is needed to yield an aqueous system of a given ionic strength.

**Parameters ionic\_strength** (openff.evaluator.unit.Quantity) – The ionic string in units of molar.

**Returns** The mole fraction of ions.

Return type float

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

classmethod from\_json(file path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

# Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

# Component

```
class openff.evaluator.substances.Component(smiles=UNDEFINED, role=Role.Solvent)
```

Defines a single component in a chemical system, as well as it's role within the system (if any).

```
__init__(smiles=UNDEFINED, role=Role.Solvent)
```

Constructs a new Component object with either a label or a smiles string, but not both.

# **Notes**

The *label* and *smiles* arguments are mutually exclusive, and only one can be passed while the other should be *None*.

#### **Parameters**

- **smiles** (*str*) A SMILES descriptor of the component
- **role** (Role) The role of this component in the system.

## **Methods**

init([smiles, role])	Constructs a new Component object with either a la-
	bel or a smiles string, but not both.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

#### **Attributes**

identifier	A unique identifier for this component.
role	The role of this component in the system.
smiles	The SMILES pattern which describes this compo-
	nent.

## class Role(value)

An enum which describes the role of a component in the system, such as whether the component is a solvent, a solute, a receptor etc.

These roles are mainly used by workflow to identify the correct species in a system, such as when doing docking or performing solvation free energy calculations.

## smiles

The SMILES pattern which describes this component. The default value of this attribute is not set and must be set by the user.. This attribute is *read-only*.

Type str

### role

The role of this component in the system. The default value of this attribute is Role.Solvent. This attribute is *read-only*.

Type Role

# property identifier

A unique identifier for this component.

Type str

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# **Amount**

class openff.evaluator.substances.Amount(value=UNDEFINED)

A representation of the amount of a given component in a Substance.

```
__init__(value=UNDEFINED)
```

**Parameters value** (*float or int*) – The value of this amount.

# **Methods**

init([value])	param value The value of this amount.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_number_of_molecules(total_substance_molecules	cule©onverts this amount to an exact number of
	molecules
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

identifier	A string identifier for this amount.
value	The value of this amount.

#### value

The value of this amount. The default value of this attribute is not set and must be set by the user.. This attribute is *read-only*.

Type Union

# property identifier

A string identifier for this amount.

abstract to\_number\_of\_molecules(total\_substance\_molecules, tolerance=None)

Converts this amount to an exact number of molecules

#### **Parameters**

- **total\_substance\_molecules** (*int*) The total number of molecules in the whole substance. This amount will contribute to a portion of this total number.
- **tolerance** (*float*, *optional*) The tolerance with which this amount should be in. As an example, when converting a mole fraction into a number of molecules, the total number of molecules may not be sufficiently large enough to reproduce this amount.

**Returns** The number of molecules which this amount represents, given the *to-tal\_substance\_molecules*.

Return type int

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** Any

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## **ExactAmount**

#### class openff.evaluator.substances.ExactAmount(value=UNDEFINED)

The exact number of instances of a Component in a Substance.

An assumption is made that this amount is for a component which is infinitely dilute (such as ligands in binding calculations), and hence do not contribute to the total mole fraction of a *Substance*.

```
__init__(value=UNDEFINED)
```

**Parameters value** (*float or int*) – The value of this amount.

### **Methods**

init([value])	param value The value of this amount.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_number_of_molecules(total_substance_molecu	le©onverts this amount to an exact number of
	molecules
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

# **Attributes**

identifier	A string identifier for this amount.
value	The value of this amount.

#### value

The value of this amount. The default value of this attribute is not set and must be set by the user..

Type int

#### property identifier

A string identifier for this amount.

# to\_number\_of\_molecules(total\_substance\_molecules, tolerance=None)

Converts this amount to an exact number of molecules

#### **Parameters**

- **total\_substance\_molecules** (*int*) The total number of molecules in the whole substance. This amount will contribute to a portion of this total number.
- **tolerance** (*float*, *optional*) The tolerance with which this amount should be in. As an example, when converting a mole fraction into a number of molecules, the total number of molecules may not be sufficiently large enough to reproduce this amount.

**Returns** The number of molecules which this amount represents, given the *to-tal\_substance\_molecules*.

Return type int

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### **MoleFraction**

class openff.evaluator.substances.MoleFraction(value=UNDEFINED)

The mole fraction of a *Component* in a *Substance*.

\_\_init\_\_(value=UNDEFINED)

**Parameters value** (*float or int*) – The value of this amount.

#### **Methods**

init([value])	param value The value of this amount.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_number_of_molecules(total_substance_molecules)	le©onverts this amount to an exact number of
	molecules
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

identifier	A string identifier for this amount.
value	The value of this amount.

# value

The value of this amount. The default value of this attribute is not set and must be set by the user..

Type float

# property identifier

A string identifier for this amount.

 $\textbf{to\_number\_of\_molecules}(\textit{total\_substance\_molecules}, \textit{tolerance} = None)$ 

Converts this amount to an exact number of molecules

#### **Parameters**

- $total\_substance\_molecules(int)$  The total number of molecules in the whole substance. This amount will contribute to a portion of this total number.
- **tolerance** (*float*, *optional*) The tolerance with which this amount should be in. As an example, when converting a mole fraction into a number of molecules, the total number of molecules may not be sufficiently large enough to reproduce this amount.

**Returns** The number of molecules which this amount represents, given the *to-tal\_substance\_molecules*.

Return type int

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

**Return type** Any

#### **State Definition**

ThermodynamicState	Data specifying a physical thermodynamic state obeying
	Boltzmann statistics.

# **ThermodynamicState**

**class** openff.evaluator.thermodynamics.**ThermodynamicState**(temperature=None, pressure=None)

Data specifying a physical thermodynamic state obeying Boltzmann statistics.

## **Notes**

Equality of two thermodynamic states is determined by comparing the temperature in kelvin to within 3 decimal places, and comparing the pressure (if defined) in pascals to within 3 decimal places.

# **Examples**

Specify an NPT state at 298 K and 1 atm pressure.

```
>>> state = ThermodynamicState(temperature=298.0*unit.kelvin, pressure=1.0*unit. 

atmospheres)
```

Note that the pressure is only relevant for periodic systems.

\_\_init\_\_(temperature=None, pressure=None)

Constructs a new ThermodynamicState object.

#### **Parameters**

- $\bullet \ \ \textbf{temperature} \ (\textit{openff.evaluator.unit.Quantity}) The \ \text{external temperature}$
- pressure (openff.evaluator.unit.Quantity) The external pressure

# **Methods**

init([temperature, pressure])	Constructs a new ThermodynamicState object.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

beta	Returns one divided by the temperature multiplied by
	the molar gas constant
inverse_beta	Returns the temperature multiplied by the molar gas
	constant
pressure	The external pressure.
temperature	The external temperature.

# property inverse\_beta

Returns the temperature multiplied by the molar gas constant

#### property beta

Returns one divided by the temperature multiplied by the molar gas constant

## temperature

The external temperature. The default value of this attribute is not set and must be set by the user..

**Type** Quantity

## pressure

The external pressure. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

# 2.32.4 Data Set API

PhysicalPropertyDataSet	An object for storing and curating data sets of both phys-
	ical property measurements and estimated.

# **PhysicalPropertyDataSet**

# class openff.evaluator.datasets.PhysicalPropertyDataSet

An object for storing and curating data sets of both physical property measurements and estimated. This class defines a number of convenience functions for filtering out unwanted properties, and for generating general statistics (such as the number of properties per substance) about the set.

# \_\_init\_\_()

Constructs a new PhysicalPropertyDataSet object.

# **Methods**

init()	Constructs a new PhysicalPropertyDataSet object.
<pre>add_properties(*physical_properties[, validate])</pre>	Adds a physical property to the data set.
from_json(file_path)	Create this object from a JSON file.
from_pandas(data_frame)	Constructs a data set object from a pandas
	DataFrame object.
json([file_path, format])	Creates a JSON representation of this class.
merge(data_set[, validate])	Merge another data set into the current one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
properties_by_substance(substance)	A generator which may be used to loop over all of
	the properties which were measured for a particular
	substance.
<pre>properties_by_type(property_type)</pre>	A generator which may be used to loop over all of
	properties of a particular type, e.g.
to_pandas()	Converts a PhysicalPropertyDataSet to a pan-
	das.DataFrame object with columns of
validate()	Checks to ensure that all properties within the set are
	valid physical property object.

# **Attributes**

properties	A list of all of the properties within this set.
property_types	The types of property within this data set.
sources	The sources from which the properties in this data set
	were gathered.
substances	The substances for which the properties in this data
	set were collected for.

# property properties

A list of all of the properties within this set.

**Type** tuple of PhysicalProperty

## property property\_types

The types of property within this data set.

Type set of str

#### property substances

The substances for which the properties in this data set were collected for.

**Type** set of Substance

# property sources

The sources from which the properties in this data set were gathered.

**Type** set of Source

# merge(data\_set, validate=True)

Merge another data set into the current one.

#### **Parameters**

- data\_set (PhysicalPropertyDataSet) The secondary data set to merge into this one.
- validate (bool) Whether to validate the other data set before merging.

# add\_properties(\*physical\_properties, validate=True)

Adds a physical property to the data set.

#### **Parameters**

- physical\_properties (PhysicalProperty) The physical property to add.
- validate (bool) Whether to validate the properties before adding them to the set.

## properties\_by\_substance(substance)

A generator which may be used to loop over all of the properties which were measured for a particular substance.

**Parameters** substance (Substance) – The substance of interest.

**Return type** generator of PhysicalProperty

#### properties\_by\_type(property\_type)

A generator which may be used to loop over all of properties of a particular type, e.g. all "Density" properties.

**Parameters property\_type**(*str or type of PhysicalProperty*) – The type of property of interest. This may either be the string class name of the property or the class type.

**Return type** generator of PhysicalProperty

## validate()

Checks to ensure that all properties within the set are valid physical property object.

# to\_pandas()

Converts a PhysicalPropertyDataSet to a pandas.DataFrame object with columns of

- 'Id'
- 'Temperature (K)'
- 'Pressure (kPa)'
- · 'Phase'
- · 'N Components'

- 'Component 1'
- 'Role 1'
- 'Mole Fraction 1'
- 'Exact Amount 1'
- ...
- · 'Component N'
- · 'Role N'
- 'Mole Fraction N'
- · 'Exact Amount N'
- '<Property 1> Value (<default unit>)'
- '<Property 1> Uncertainty / (<default unit>)'
- ...
- '<Property N> Value / (<default unit>)'
- '<Property N> Uncertainty / (<default unit>)'
- · 'Source'

where 'Component X' is a column containing the smiles representation of component X.

**Returns** The create data frame.

Return type DataFrame

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

classmethod from\_pandas( $data\_frame: DataFrame$ )  $\rightarrow PhysicalPropertyDataSet$ 

Constructs a data set object from a pandas DataFrame object.

# **Notes**

- All physical properties are assumed to be source from experimental measurements.
- Currently this method onlu supports data frames containing properties which are built-in to the framework (e.g. Density).
- This method assumes the data frame has a structure identical to that produced by the PhysicalPropertyDataSet.to\_pandas function.

**Parameters data\_frame** – The data frame to construct the data set from.

**Return type** The constructed data set.

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

**Return type** Any

#### **NIST ThermoML Archive**

ThermoMLDataSet	A dataset of physical property measurements created from a ThermoML dataset.
register_thermoml_property	A function used to map a property from the ThermoML archive to an internal <i>PhysicalProperty</i> object of the cor-
	rect type.
thermoml_property	A decorator which wraps around the regis-
	ter_thermoml_property method.

### **ThermoMLDataSet**

class openff.evaluator.datasets.thermoml.ThermoMLDataSet

A dataset of physical property measurements created from a ThermoML dataset.

# **Examples**

For example, we can use the DOI 10.1016/j.jct.2005.03.012 as a key for retrieving the dataset from the ThermoML Archive:

```
>>> dataset = ThermoMLDataSet.from_doi('10.1016/j.jct.2005.03.012')
```

You can also specify multiple ThermoML Archive keys to create a dataset from multiple ThermoML files:

```
>>> thermoml_keys = ['10.1021/acs.jced.5b00365', '10.1021/acs.jced.5b00474']
>>> dataset = ThermoMLDataSet.from_doi(*thermoml_keys)
```

```
__init__()
```

Constructs a new ThermoMLDataSet object.

init()	Constructs a new ThermoMLDataSet object.
<pre>add_properties(*physical_properties[, validate])</pre>	Adds a physical property to the data set.
<pre>from_doi(*doi_list)</pre>	Load a ThermoML data set from a list of DOIs
from_file(*file_list)	Load a ThermoML data set from a list of files
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>from_pandas(data_frame)</pre>	Constructs a data set object from a pandas
	DataFrame object.
<pre>from_url(*url_list)</pre>	Load a ThermoML data set from a list of URLs
<pre>from_xml(xml, default_source)</pre>	Load a ThermoML data set from an xml object.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
<pre>merge(data_set[, validate])</pre>	Merge another data set into the current one.
<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
	structure.
<pre>properties_by_substance(substance)</pre>	A generator which may be used to loop over all of
	the properties which were measured for a particular
	substance.
<pre>properties_by_type(property_type)</pre>	A generator which may be used to loop over all of
	properties of a particular type, e.g.
to_pandas()	Converts a PhysicalPropertyDataSet to a pan-
	das.DataFrame object with columns of
validate()	Checks to ensure that all properties within the set are
	valid physical property object.

## **Attributes**

properties	A list of all of the properties within this set.
property_types	The types of property within this data set.
registered_properties	
sources	The sources from which the properties in this data set
	were gathered.
substances	The substances for which the properties in this data
	set were collected for.

# classmethod from\_doi(\*doi\_list)

Load a ThermoML data set from a list of DOIs

**Parameters doi\_list** (*str*) – The list of DOIs to pull data from

**Returns** The loaded data set.

Return type ThermoMLDataSet

# classmethod from\_url(\*url\_list)

Load a ThermoML data set from a list of URLs

**Parameters url\_list** (*str*) – The list of URLs to pull data from

Returns The loaded data set.

Return type ThermoMLDataSet

## classmethod from\_file(\*file\_list)

Load a ThermoML data set from a list of files

**Parameters file\_list** (*str*) – The list of files to pull data from

**Returns** The loaded data set.

Return type ThermoMLDataSet

## add\_properties(\*physical\_properties, validate=True)

Adds a physical property to the data set.

#### **Parameters**

- physical\_properties (PhysicalProperty) The physical property to add.
- validate (bool) Whether to validate the properties before adding them to the set.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

classmethod from\_pandas( $data\_frame: DataFrame$ )  $\rightarrow PhysicalPropertyDataSet$ 

Constructs a data set object from a pandas DataFrame object.

## **Notes**

- All physical properties are assumed to be source from experimental measurements.
- Currently this method onlu supports data frames containing properties which are built-in to the framework (e.g. Density).
- This method assumes the data frame has a structure identical to that produced by the PhysicalPropertyDataSet.to\_pandas function.

**Parameters data\_frame** – The data frame to construct the data set from.

Return type The constructed data set.

#### classmethod from\_xml(xml, default\_source)

Load a ThermoML data set from an xml object.

## **Parameters**

- **xml** (*str*) The xml string to parse.
- default\_source (Source) The source to use if one cannot be parsed from the archive itself.

Returns The loaded ThermoML data set.

Return type ThermoMLDataSet

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

#### merge(data set, validate=True)

Merge another data set into the current one.

#### **Parameters**

- data\_set (PhysicalPropertyDataSet) The secondary data set to merge into this one.
- validate (bool) Whether to validate the other data set before merging.

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters** string\_contents (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** *Any* 

# property properties

A list of all of the properties within this set.

Type tuple of PhysicalProperty

# properties\_by\_substance(substance)

A generator which may be used to loop over all of the properties which were measured for a particular substance.

**Parameters substance** (Substance) – The substance of interest.

**Return type** generator of PhysicalProperty

## properties\_by\_type(property\_type)

A generator which may be used to loop over all of properties of a particular type, e.g. all "Density" properties.

**Parameters property\_type** (str or type of PhysicalProperty) – The type of property of interest. This may either be the string class name of the property or the class type.

**Return type** generator of PhysicalProperty

## property property\_types

The types of property within this data set.

**Type** set of str

# property sources

The sources from which the properties in this data set were gathered.

Type set of Source

## property substances

The substances for which the properties in this data set were collected for.

**Type** set of Substance

#### to\_pandas()

Converts a PhysicalPropertyDataSet to a pandas.DataFrame object with columns of

- 'Id'
- 'Temperature (K)'
- · 'Pressure (kPa)'
- · 'Phase'
- · 'N Components'
- 'Component 1'
- 'Role 1'
- · 'Mole Fraction 1'
- 'Exact Amount 1'
- ...
- · 'Component N'
- 'Role N'
- 'Mole Fraction N'
- 'Exact Amount N'
- '<Property 1> Value (<default unit>)'
- '<Property 1> Uncertainty / (<default unit>)'
- ...
- '<Property N> Value / (<default unit>)'
- '<Property N> Uncertainty / (<default unit>)'
- · 'Source'

where 'Component X' is a column containing the smiles representation of component X.

Returns The create data frame.

Return type DataFrame

## validate()

Checks to ensure that all properties within the set are valid physical property object.

# register\_thermoml\_property

```
openff.evaluator.datasets.thermoml.register_thermoml_property(thermoml_string, supported_phases, property_class=None, conversion_function=None)
```

A function used to map a property from the ThermoML archive to an internal *PhysicalProperty* object of the correct type.

This function takes either a specific class (e.g. *Density*) which maps directly to the specified *thermoml\_string*, or a a function which maps a *ThermoMLProperty* into a *PhysicalProperty* allowing fuller control.

#### **Parameters**

- **thermoml\_string** (*str*) The ThermoML string identifier (ePropName) for this property.
- **supported\_phases** (*PropertyPhase:*) An enum which encodes all of the phases for which this property supports being estimated in.
- property\_class (type of PhysicalProperty, optional) The class associated with this physical property. This argument is mutually exclusive with the *conversion\_function* argument.
- **conversion\_function** (*function*) A function which maps a *ThermoMLProperty* into a *PhysicalProperty*. This argument is mutually exclusive with the *property\_class* argument.

## thermoml property

openff.evaluator.datasets.thermoml.thermoml\_property(thermoml\_string, supported\_phases)
A decorator which wraps around the register\_thermoml\_property method.

#### **Parameters**

- **thermoml\_string** (*str*) The ThermoML string identifier (ePropName) for this property.
- **supported\_phases** (*PropertyPhase:*) An enum which encodes all of the phases for which this property supports being estimated in.

# **Taproom**

TaproomDataSet	A dataset of host-guest binding affinity measurements
	which sources its data from the taproom package.
TaproomSource	Contains metadata about the source of a host-guest bind-
	ing affinity measurement which was pulled from the
	taproom package.

## **TaproomDataSet**

```
class openff.evaluator.datasets.taproom.TaproomDataSet(host\_codes: Optional[List[str]] = None, guest\_codes: Optional[List[str]] = None, default\_ionic\_strength: Optional[Quantity] = 150 * unit.millimolar, negative\_buffer\_ion: str = '[Cl-]', positive\_buffer\_ion: str = '[Na+]', attach\_apr\_meta\_data: bool = True)
```

A dataset of host-guest binding affinity measurements which sources its data from the taproom package.

The loaded HostGuestBindingAffinity properties will also be optionally (enabled by default) initialized with the metadata required by the APR estimation workflow.

```
__init__(host_codes: Optional[List[str]] = None, guest_codes: Optional[List[str]] = None, default_ionic_strength: Optional[Quantity] = 150 * unit.millimolar, negative_buffer_ion: str = '[Cl-]', positive_buffer_ion: str = '[Na+]', attach_apr_meta_data: bool = True)
```

## **Parameters**

- **host\_codes** The three letter codes of the host molecules to load from taproom If no list is provided, all hosts will be loaded.
- **guest\_codes** The three letter codes of the guest molecules to load from taproom. If no list is provided, all guests will be loaded.

- **default\_ionic\_strength** The default ionic strength to use for measurements. The value specified in taproom will be ignored and this value used instead. If no value is provided, no buffer will be included.
- **negative\_buffer\_ion** The SMILES pattern of the negative buffer ion to use. The value specified in taproom will be ignored and this value used instead.
- **positive\_buffer\_ion** The SMILES pattern of the positive buffer ion to use. The value specified in taproom will be ignored and this value used instead.
- attach\_apr\_meta\_data Whether to add the metadata required for an APR based calculation using the paprika based workflow.

init([host_codes, guest_codes,])	
	<pre>param host_codes The three letter   codes of the host molecules to load   from taproom</pre>
<pre>add_properties(*physical_properties[, validate])</pre>	Adds a physical property to the data set.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_pandas(data_frame)	Constructs a data set object from a pandas
	DataFrame object.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
<pre>merge(data_set[, validate])</pre>	Merge another data set into the current one.
parse_json(string_contents)	Parses a typed json string into the corresponding class structure.
<pre>properties_by_substance(substance)</pre>	A generator which may be used to loop over all of the properties which were measured for a particular substance.
<pre>properties_by_type(property_type)</pre>	A generator which may be used to loop over all of properties of a particular type, e.g.
to_pandas()	Converts a <i>PhysicalPropertyDataSet</i> to a <i>pan-das.DataFrame</i> object with columns of
validate()	Checks to ensure that all properties within the set are valid physical property object.

# **Attributes**

properties	A list of all of the properties within this set.
property_types	The types of property within this data set.
sources	The sources from which the properties in this data set
	were gathered.
substances	The substances for which the properties in this data
	set were collected for.

add\_properties(\*physical\_properties, validate=True)

Adds a physical property to the data set.

# **Parameters**

- physical\_properties (PhysicalProperty) The physical property to add.
- validate (bool) Whether to validate the properties before adding them to the set.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

classmethod from\_pandas( $data\_frame: DataFrame$ )  $\rightarrow PhysicalPropertyDataSet$ 

Constructs a data set object from a pandas DataFrame object.

## **Notes**

- All physical properties are assumed to be source from experimental measurements.
- Currently this method onlu supports data frames containing properties which are built-in to the framework (e.g. Density).
- This method assumes the data frame has a structure identical to that produced by the PhysicalPropertyDataSet.to\_pandas function.

**Parameters data\_frame** – The data frame to construct the data set from.

**Return type** The constructed data set.

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

merge(data\_set, validate=True)

Merge another data set into the current one.

## **Parameters**

- data\_set (PhysicalPropertyDataSet) The secondary data set to merge into this one.
- validate (bool) Whether to validate the other data set before merging.

#### classmethod parse\_json(string contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

Return type Any

## property properties

A list of all of the properties within this set.

**Type** tuple of PhysicalProperty

## properties\_by\_substance(substance)

A generator which may be used to loop over all of the properties which were measured for a particular substance.

**Parameters substance** (Substance) – The substance of interest.

**Return type** generator of PhysicalProperty

# properties\_by\_type(property\_type)

A generator which may be used to loop over all of properties of a particular type, e.g. all "Density" properties.

**Parameters property\_type** (*str or type of PhysicalProperty*) – The type of property of interest. This may either be the string class name of the property or the class type.

Return type generator of PhysicalProperty

# property property\_types

The types of property within this data set.

Type set of str

## property sources

The sources from which the properties in this data set were gathered.

Type set of Source

# property substances

The substances for which the properties in this data set were collected for.

Type set of Substance

## to\_pandas()

Converts a PhysicalPropertyDataSet to a pandas.DataFrame object with columns of

- 'Id'
- 'Temperature (K)'
- 'Pressure (kPa)'
- · 'Phase'
- 'N Components'
- 'Component 1'
- 'Role 1'
- 'Mole Fraction 1'
- 'Exact Amount 1'
- ...
- · 'Component N'
- 'Role N'
- 'Mole Fraction N'

- · 'Exact Amount N'
- '<Property 1> Value (<default unit>)'
- '<Property 1> Uncertainty / (<default unit>)'
- ...
- '<Property N> Value / (<default unit>)'
- '<Property N> Uncertainty / (<default unit>)'
- 'Source'

where 'Component X' is a column containing the smiles representation of component X.

Returns The create data frame.

Return type DataFrame

## validate()

Checks to ensure that all properties within the set are valid physical property object.

# **TaproomSource**

Contains metadata about the source of a host-guest binding affinity measurement which was pulled from the taproom package.

```
__init__(doi=", comment=", technique=", host_identifier=", guest_identifier=")
```

Constructs a new MeasurementSource object.

## **Parameters**

- **doi** (*str*) The DOI for the source
- **comment** (*str*) A description of where the value came from in the source.
- **technique** (*str*) The technique used to measure this value.
- host\_identifier (str) The unique three letter host identifier
- **guest\_identifier** (*str*) The unique three letter guest identifier

# **Methods**

init([doi, comment, technique,])	Constructs a new MeasurementSource object.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

## Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** Any

#### **Data Set Curation**

CurationComponent	A base component for curation components which apply
	a particular operation (such as filtering or data conver-
	sion) to a data set.
CurationComponentSchema	A base class for schemas which specify how particular
	curation components should be applied to a data set.

# CurationComponent

# class openff.evaluator.datasets.curation.components.CurationComponent

A base component for curation components which apply a particular operation (such as filtering or data conversion) to a data set.

\_\_init\_\_()

#### **Methods**

init()	
apply()	Apply this curation component to a data set.

**classmethod apply**( $data\_set$ : PhysicalPropertyDataSet, schema: CurationComponentSchema,  $n\_processes$ : int = 1)  $\rightarrow PhysicalPropertyDataSet$ 

classmethod apply( $data\_set: DataFrame, schema: CurationComponentSchema, n\_processes: int = 1) <math>\rightarrow$  DataFrame

Apply this curation component to a data set.

#### **Parameters**

- data\_set The data frame to apply the component to.
- schema The schema which defines how this component should be applied.

n\_processes – The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **CurationComponentSchema**

# class openff.evaluator.datasets.curation.components.CurationComponentSchema

A base class for schemas which specify how particular curation components should be applied to a data set.

**\_\_init\_\_**(\*\*
$$data: Any$$
)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

#### **Methods**

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
<pre>copy(*[, include, exclude, update, deep])</pre>	Duplicate a model, optionally choose which fields to
	include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model,
	optionally specifying which fields to include or ex-
(1.)	clude.
<pre>from_orm(obj)</pre>	
json(*[, include, exclude, by_alias,])	Generate a JSON representation of the model, in-
JSon [, merude, exclude, by_anas,])	clude and exclude arguments as per dict().
parse_file(path, *[, content_type,])	crute and exerute arguments as per area().
parae_rrac(pain, [, content_t)pe, m <sub>1</sub> )	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
<pre>schema([by_alias, ref_template])</pre>	
<pre>schema_json(*[, by_alias, ref_template])</pre>	
1	T . 1. F . D C . C 11 1 1 . d :
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this
validata(valva)	Model, globalns and localns.
validate(value)	

 $\textbf{classmethod construct}(\_\textit{fields\_set: Optional[SetStr]} = \textit{None}, **values: Any}) \rightarrow \textbf{Model}$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

CurationWorkflow	A convenience class for applying a set of curation com-
	ponents sequentially to a data set.
CurationWorkflowSchema	A schemas which encodes how a set of curation compo-
	nents should be applied sequentially to a data set.

# CurationWorkflow

class openff.evaluator.datasets.curation.workflow.CurationWorkflow

A convenience class for applying a set of curation components sequentially to a data set.

\_\_init\_\_()

init()	
apply()	Apply each component of this curation workflow to an initial data set in sequence.

**classmethod apply**( $data\_set$ : PhysicalPropertyDataSet, schema: CurationWorkflowSchema,  $n\_processes$ :  $int = 1) \rightarrow PhysicalPropertyDataSet$ 

classmethod apply ( $data\_set: DataFrame, schema: CurationWorkflowSchema, n\_processes: int = 1) <math>\rightarrow$  DataFrame

Apply each component of this curation workflow to an initial data set in sequence.

## **Parameters**

- data\_set The data set to apply the workflow to. This may either be a data set object or it's pandas representation.
- **schema** The schema which defines the components to apply.
- **n\_processes** The number of processes that each component is allowed to parallelize across.

**Return type** The data set which has had the curation workflow applied to it.

## CurationWorkflowSchema

class openff.evaluator.datasets.curation.workflow.CurationWorkflowSchema(\*,

component\_schemas:

List[Union[ConvertExcessDensityDatas

FilterDuplicatesS-

chema,

FilterByTempera-

tureSchema,

FilterByPres-

sureSchema,

FilterByMoleFrac-

tionSchema,

FilterByRacemic-

Schema,

FilterByElementsS-

chema,

FilterByProperty-

TypesSchema,

FilterByStereochem-

istrySchema,

FilterByCharged-

Schema,

FilterByIonicLiquid-

Schema,

FilterBySmilesS-

chema,

FilterBySmirksS-

chema,

FilterByNCompo-

nentsSchema,

FilterBySubstancesS-

chema,

FilterByEnviron-

mentsSchema,

ImportFree-

SolvSchema,

SelectSubstancesS-

chema.

SelectDataPointsS-

chema,

ImportThermoML-

DataSchema]] =

None)

A schemas which encodes how a set of curation components should be applied sequentially to a data set.

**\_\_init\_\_(\*\*** $data: Any) \rightarrow None$ 

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
,,	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
<pre>schema_json(*[, by_alias, ref_template])</pre>	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

# **Attributes**

component\_schemas

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

## **Parameters**

- include fields to include in new model
- **exclude** fields to exclude from new model, as with values this takes precedence over include

- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### Returns new model instance

$$\label{eq:dict} \begin{split} \textbf{dict}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults: \\ Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none: \\ bool = False) \rightarrow \text{DictStrAny} \end{split}$$

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

Filtering

A component to remove duplicate data points (within a specified precision) from a data set.
A component which will filter out data points which were measured outside of a specified temperature range
2 2
A component which will filter out data points which were measured outside of a specified pressure range.
A component which will filter out data points which were measured outside of a specified mole fraction range.
A component which will filter out data points which were measured for racemic mixtures.
A component which will filter out data points which were measured for systems which contain specific elements.
A component which will apply a filter which only retains properties of specified types.
A component which filters out data points measured for systems whereby the stereochemistry of a number of components is undefined.
1
A component which filters out data points measured for substances where any of the constituent components have a net non-zero charge.
have a not non zoro enarge.
A component which filters out data points measured for substances which contain or are classed as an ionic liq- uids.
eras.
A component which filters the data set so that it only contains either a specific set of smiles, or does not contain any of a set of specifically excluded smiles.
any of a set of specifically excluded sinites.
A component which filters a data set so that it only contains measurements made for molecules which contain
(or don't) a set of chemical environments represented by-

A component which filters out data points measured for

FilterRyNComponents

# **FilterDuplicatesSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterDuplicatesSchema(\*,

```
type:
Lit-
eral['FilterDuplicates']
'Fil-
terDu-
pli-
cates',
tem-
pera-
ture_precision:
Con-
strained-
Int-
Value
= 2,
pres-
sure\_precision:
Con-
strained-
Int-
Value
=3,
mole_fraction_precision
Con-
strained-
Int-
Value
```

= 6)

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to
	include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model,
urer([, merude, exerude, by_anas,])	* *
	optionally specifying which fields to include or ex-
	clude.
<pre>from_orm(obj)</pre>	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
	clude and exclude arguments as per dict().
parse_file(path, *[, content_type,])	
•	
parse_obj(obj)	
F == 0 = 2 = 3 ( 0 = 3)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
parse_raw(o, [, content_type, encounig,])	
ashawa(flor alian ash tamalatal)	
<pre>schema([by_alias, ref_template])</pre>	
<pre>schema_json(*[, by_alias, ref_template])</pre>	
<pre>update_forward_refs(**localns)</pre>	Try to update ForwardRefs on fields based on this
	Model, globalns and localns.
validate(value)	-
` '	

## **Attributes**

type

temperature\_precision

pressure\_precision

mole\_fraction\_precision

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) \rightarrow None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterDuplicates**

```
class openff.evaluator.datasets.curation.components.filtering.FilterDuplicates
```

A component to remove duplicate data points (within a specified precision) from a data set.

```
__init__()
```

#### Methods

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterByTemperatureSchema**

 ${\bf class}\ open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Temperature Schema} (*, note that the component of th$ 

```
type:
Lit-
eral['FilterByTempe
'Fil-
ter-
ByTem-
per-
ture',
min-
i-
mum_temperature:
Op-
tional[PositiveFloat
max-
mum\_temperature:
Op-
tional[PositiveFloat
```

```
__init__(** data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
<pre>copy(*[, include, exclude, update, deep])</pre>	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .
<pre>parse_file(path, *[, content_type,])</pre>	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

## **Attributes**

type
minimum\_temperature
maximum\_temperature

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

# **Parameters**

• include – fields to include in new model

- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,
\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

## **FilterByTemperature**

class openff.evaluator.datasets.curation.components.filtering.FilterByTemperature

A component which will filter out data points which were measured outside of a specified temperature range \_\_init\_\_()

# Methods

\_\_init\_\_()

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- data\_set The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- n\_processes The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterByPressureSchema**

 ${\bf class} \ {\bf open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Pressure Schema} (*, {\bf class}) {\bf open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Pressure Schema} (*, {\bf class}) {\bf open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Pressure Schema} (*, {\bf class}) {\bf open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Pressure Schema} (*, {\bf class}) {\bf open ff. evaluator. datasets. curation. {\bf components. filtering. {\bf class} (*, {\bf class}) {\bf open ff. evaluator. datasets. {\bf class} ($ 

```
type:
Lit-
eral['FilterByPressure']
'Fil-
ter-
By-
Pres-
sure',
mini-
mum_pressure:
Op-
tional[PositiveFloat],
maxi-
mum_pressure:
Op-
tional[PositiveFloat])
```

**\_\_init\_\_(\*\*** $data: Any) \rightarrow None$ 

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

ini+ (**dota)	Create a new model by persing and validating input
init(**data)	Create a new model by parsing and validating input
(5.0.1.1	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to
	include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model,
, , , , , , , , , , , , , , , , , , , ,	optionally specifying which fields to include or ex-
	clude.
from_orm(obj)	ciude.
110m_01m(00j)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
• • • • • •	clude and exclude arguments as per dict().
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
Schema([by_anas, ref_template])	
schema_json(*[, by_alias, ref_template])	
<pre>update_forward_refs(**localns)</pre>	Try to update ForwardRefs on fields based on this
	Model, globalns and localns.
validate(value)	
Attributes	
THE INCHES	

type minimum\_pressure maximum\_pressure

 $\textbf{classmethod construct}(\_\textit{fields\_set: Optional[SetStr]} = \textit{None}, **values: Any}) \rightarrow \textbf{Model}$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if Config.extra = 'allow' was set since it adds all passed values

**copy** (\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:  $Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: \ Optional[DictStrAny] =$ *deep:*  $bool = False) \rightarrow Model$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

# **Parameters**

• include – fields to include in new model

- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

## **FilterByPressure**

class openff.evaluator.datasets.curation.components.filtering.FilterByPressure

A component which will filter out data points which were measured outside of a specified pressure range.

```
__init__()
```

# Methods

\_\_init\_\_()

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

classmethod apply(data set, schema, n processes=1)

Apply this curation component to a data set.

#### **Parameters**

- data\_set The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterByMoleFractionSchema**

 ${\bf class} \ {\bf open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Mole Fraction Schema} (*, {\bf class. components. filtering. {\bf Filter By Mole Fraction Schema} (*, {\bf class. components. filtering. {\bf class. components. {\bf class. class. {\bf class. {\bf class. class. {\bf cl$ 

```
type:
Lit-
eral['FilterByMole
'Fil-
ter-
By-
Mole-
Frac-
tion',
mole_fraction_ran
Dict[ConstrainedI
List[List[Tuple[Co
Con-
strained-
Float-
Value]]]])
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
,,	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

# **Attributes**

type
mole\_fraction\_ranges

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

## **Parameters**

• include – fields to include in new model

- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
 Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
 Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
 bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

# classmethod update\_forward\_refs(\*\*localns: Any) $\rightarrow$ None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

## **FilterByMoleFraction**

class openff.evaluator.datasets.curation.components.filtering.FilterByMoleFraction

A component which will filter out data points which were measured outside of a specified mole fraction range.

```
__init__()
```

# **Methods**

\_\_init\_\_()

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- data\_set The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterByRacemicSchema**

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

# Methods

init(**data)	Create a new model by parsing and validating input data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

#### **Attributes**

type

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- **exclude** fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterByRacemic**

class openff.evaluator.datasets.curation.components.filtering.FilterByRacemic

A component which will filter out data points which were measured for racemic mixtures.

```
__init__()
```

## **Methods**

```
__init__()
```

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- n\_processes The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterByElementsSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterByElementsSchema(\*,

```
a(*,
type:
Lit-
eral['FilterByElements']
=
'Fil-
ter-
ByEle-
ments',
al-
lowed_elements:
Op-
tional[List[Constrained.
```

None for-

jorbid-

den\_elements:

Op-

tional[List[Constrained

= None) **\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

# **Attributes**

type

allowed\_elements

forbidden\_elements

classmethod construct( $\_fields\_set$ : Optional[SetStr] = None, \*\*values: Any)  $\rightarrow$  Model

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

```
copy(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model
```

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) \rightarrow None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterByElements**

class openff.evaluator.datasets.curation.components.filtering.FilterByElements

A component which will filter out data points which were measured for systems which contain specific elements.

```
__init__()
```

# Methods

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

**Parameters** 

- data\_set The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterByPropertyTypesSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterByPropertyTypesSchema(\*,

```
type:
Lit-
eral['FilterByPro
'Fil-
ter-
ByProp-
er-
ty-
Types',
prop-
erty_types:
List[Constrained
n_components:
Dict[Constrained
List[PositiveInt]]
None,
strict:
bool
False)
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
<pre>copy(*[, include, exclude, update, deep])</pre>	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .
<pre>parse_file(path, *[, content_type,])</pre>	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

## **Attributes**

type
property\_types
n\_components
strict

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) \rightarrow None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterByPropertyTypes**

```
class openff.evaluator.datasets.curation.components.filtering.FilterByPropertyTypes
    A component which will apply a filter which only retains properties of specified types.
```

```
__init__()
```

#### Methods

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- **data\_set** The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- n\_processes The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# FilterByStereochemistrySchema

```
type:
Lit-
eral['FilterByS
=
'Fil-
ter-
ByStere-
o-
chem-
istry')
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating input
. ,	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
<pre>copy(*[, include, exclude, update, deep])</pre>	Duplicate a model, optionally choose which fields to
	include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model,
	optionally specifying which fields to include or ex-
	clude.
<pre>from_orm(obj)</pre>	
, (MF : 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
	clude and exclude arguments as per dict().
<pre>parse_file(path, *[, content_type,])</pre>	
names shi(shi)	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
parse_raw(o, [, content_type, encounig,])	
schema([by_alias, ref_template])	
Seriema([oy_unus, rer_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this
	Model, globalns and localns.
validate(value)	

#### **Attributes**

type

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
 Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
 Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
 bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterByStereochemistry**

class openff.evaluator.datasets.curation.components.filtering.FilterByStereochemistry

A component which filters out data points measured for systems whereby the stereochemistry of a number of components is undefined.

```
__init__()
```

#### **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- data\_set The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

Return type The data set which has had the component applied to it.

# **FilterByChargedSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterByChargedSchema(\*,

```
type:
Lit-
eral['FilterByCharged']
= 'Fil-
terBy-
Charged')
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
IIII ( · · data)	* * * * * * * * * * * * * * * * * * * *
	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to
•	include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model,
aree ([, merade, exerade, ey_ands,])	optionally specifying which fields to include or ex-
	clude.
<pre>from_orm(obj)</pre>	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
	clude and exclude arguments as per dict().
parse_file(path, *[, content_type,])	
F(F, [,	
parse_obj(obj)	
parse_obj(obj)	
(1 % (1 1 1)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
<pre>schema([by_alias, ref_template])</pre>	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this
apaate_rorwara_rers( localis)	
	Model, globalns and localns.
validate(value)	

## **Attributes**

type

 $\textbf{classmethod construct}(\_\textit{fields\_set: Optional[SetStr]} = \textit{None}, **values: Any}) \rightarrow \textbf{Model}$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

## **Parameters**

- include fields to include in new model
- **exclude** fields to exclude from new model, as with values this takes precedence over include

- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

Returns new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,
\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) $\rightarrow$ None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterByCharged**

## class openff.evaluator.datasets.curation.components.filtering.FilterByCharged

A component which filters out data points measured for substances where any of the constituent components have a net non-zero charge.

```
__init__()
```

#### **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# **FilterBylonicLiquidSchema**

 ${\bf class}\ open ff. evaluator. datasets. curation. components. filtering. {\bf FilterBy Ionic Liquid Schema} (*, note that the component of th$ 

```
type:
Lit-
eral['FilterByIonicL'
=
'Fil-
ter-
ByIon-
i-
cLiq-
uid')
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
<pre>update_forward_refs(**localns)</pre>	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

#### **Attributes**

type

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

classmethod update\_forward\_refs(\*\*localns: Any) → None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterBylonicLiquid**

class openff.evaluator.datasets.curation.components.filtering.FilterByIonicLiquid

A component which filters out data points measured for substances which contain or are classed as an ionic liquids.

```
__init__()
```

#### **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

Return type The data set which has had the component applied to it.

# **FilterBySmilesSchema**

```
class openff.evaluator.datasets.curation.components.filtering.FilterBySmilesSchema(*, type:
```

```
ema (*, type:
Lit-
eral ['FilterBySmiles']
=
'Filter-
BySmiles',
smiles_to_include:
Op-
tional [List[str]]
= None,
smiles_to_exclude:
Op-
tional [List[str]]
= None,
al-
low_partial_inclusion:
bool =
False)
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
,	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
<pre>schema_json(*[, by_alias, ref_template])</pre>	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

## **Attributes**

type
smiles\_to\_include
smiles\_to\_exclude
allow\_partial\_inclusion

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) → None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterBySmiles**

```
class openff.evaluator.datasets.curation.components.filtering.FilterBySmiles
```

A component which filters the data set so that it only contains either a specific set of smiles, or does not contain any of a set of specifically excluded smiles.

```
__init__()
```

#### **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- data\_set The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

Return type The data set which has had the component applied to it.

# **FilterBySmirksSchema**

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

False)

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

## **Attributes**

type
smirks\_to\_include
smirks\_to\_exclude
allow\_partial\_inclusion

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,

**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) → None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterBySmirks**

class openff.evaluator.datasets.curation.components.filtering.FilterBySmirks

A component which filters a data set so that it only contains measurements made for molecules which contain (or don't) a set of chemical environments represented by SMIRKS patterns.

```
__init__()
```

#### **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- data\_set The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- n\_processes The number of processes that this component is allowed to parallelize across.

Return type The data set which has had the component applied to it.

# **FilterByNComponentsSchema**

```
type:
Lit-
eral['FilterByNCom
=
'Fil-
ter-
ByN-
Com-
po-
nents',
n_components:
List[PositiveInt])
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

# Methods

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
<pre>copy(*[, include, exclude, update, deep])</pre>	Duplicate a model, optionally choose which fields to
	include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model,
	optionally specifying which fields to include or ex-
	clude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
	clude and exclude arguments as per dict().
<pre>parse_file(path, *[, content_type,])</pre>	
parse_obj(obj)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
<pre>schema([by_alias, ref_template])</pre>	
<pre>schema_json(*[, by_alias, ref_template])</pre>	
The Country of the Co	The state of the s
<pre>update_forward_refs(**localns)</pre>	Try to update ForwardRefs on fields based on this
	Model, globalns and localns.
validate(value)	

#### **Attributes**

type

n\_components

# classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any$ ) $\rightarrow$ Model

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:} \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \\ \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### Returns new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,
\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

# classmethod update\_forward\_refs(\*\*localns: Any) $\rightarrow$ None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterByNComponents**

class openff.evaluator.datasets.curation.components.filtering.FilterByNComponents

A component which filters out data points measured for systems with specified number of components.

```
__init__()
```

## **Methods**

```
__init__()
```

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# FilterBySubstancesSchema

 $\textbf{class} \ \ \textbf{open} \textbf{ff.evaluator.datasets.curation.components.filtering.} \textbf{FilterBySubstancesSchema} (*, \texttt{s.curation.components.filtering.} \textbf{FilterBySubstancesSchema} (*, \texttt{s.curation.components.filterBySubstancesSchema} (*, \texttt{s.curation.components.filterBySubstancesBotance$ 

```
type:
Lit-
eral['FilterBySubstan
'Fil-
ter-
By-
Sub-
stances'.
sub-
stances_to_include:
Op-
tional[List[Tuple[str,
... ]]]
None,
sub-
stances_to_exclude:
Op-
tional[List[Tuple[str,
```

...]]]

None)

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict and
(ME + 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	fields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

# **Attributes**

type
substances\_to\_include
substances\_to\_exclude

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

```
copy(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model
```

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) → None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FilterBySubstances**

## class openff.evaluator.datasets.curation.components.filtering.FilterBySubstances

A component which filters the data set so that it only contains properties measured for particular substances.

This method is similar to *filter\_by\_smiles*, however here we explicitly define the full substances compositions, rather than individual smiles which should either be included or excluded.

### **Examples**

To filter the data set to only include measurements for pure methanol, pure benzene or an aqueous ethanol mix:

```
>>> schema = FilterBySubstancesSchema(
>>> substances_to_include=[
>>> ('C0',),
>>> ('C1=CC=CC=C1',),
>>> ('CCO', 'O')
>>> ]
>>> )
```

To filter out measurements made for an aqueous mix of benzene:

```
>>> schema = FilterBySubstancesSchema(
>>> substances_to_exclude=[('0', 'C1=CC=CC=C1')]
>>> )
```

# Methods

\_\_init\_\_()

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

Return type The data set which has had the component applied to it.

# FilterByEnvironmentsSchema

```
class openff.evaluator.datasets.curation.components.filtering.FilterByEnvironmentsSchema(*,
```

```
type:
Lit-
eral['FilterByEnvi
'Fil-
ter-
ByEn-
vi-
ron-
ments',
per_component_er
Op-
tional[Dict[int,
List[List[Chemical
None,
en-
vi-
ron-
ments:
Op-
tional[List[Chemic
None,
at_least_one_envii
bool
True,
strictly_specified_e
bool
```

False)

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
,	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
<pre>schema_json(*[, by_alias, ref_template])</pre>	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

## **Attributes**

type

per\_component\_environments

environments

at\_least\_one\_environment

strictly\_specified\_environments

 $\textbf{classmethod construct}(\_\textit{fields\_set: Optional[SetStr]} = \textit{None}, **values: Any}) \rightarrow \textbf{Model}$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \\ \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) \rightarrow None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

## **FilterByEnvironments**

### class openff.evaluator.datasets.curation.components.filtering.FilterByEnvironments

A component which filters a data set so that it only contains measurements made for substances which contain specific chemical environments.

```
__init__()
```

# Methods

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- **schema** The schema which defines how this component should be applied.

• **n\_processes** – The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

## FreeSolv

ImportFreeSolvSchema	
ImportFreeSolv	A component which will import the latest version of the FreeSolv data set from the GitHub repository where it is stored.

# **ImportFreeSolvSchema**

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
IIII ( · · data)	* * * * * * * * * * * * * * * * * * * *
	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to
•	include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model,
aree ([, merade, exerade, ey_ands,])	optionally specifying which fields to include or ex-
	clude.
<pre>from_orm(obj)</pre>	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
	clude and exclude arguments as per dict().
parse_file(path, *[, content_type,])	
F(F, [,	
parse_obj(obj)	
parse_obj(obj)	
(1 % (1 1 1)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
<pre>schema([by_alias, ref_template])</pre>	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this
apaate_rorwara_rers( localis)	
	Model, globalns and localns.
validate(value)	

# **Attributes**

type

 $\textbf{classmethod construct}(\_\textit{fields\_set: Optional[SetStr]} = \textit{None}, **values: Any}) \rightarrow \textbf{Model}$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

## **Parameters**

- include fields to include in new model
- **exclude** fields to exclude from new model, as with values this takes precedence over include

- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

Returns new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
 Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
 Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
 bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,
\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) $\rightarrow$ None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

## **ImportFreeSolv**

## class openff.evaluator.datasets.curation.components.freesolv.ImportFreeSolv

A component which will import the latest version of the FreeSolv data set from the GitHub repository where it is stored.

\_\_init\_\_()

#### **Methods**

\_\_init\_\_()

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- n\_processes The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

**ThermoML** 

ImportThermoMLDataSchema	
ImportThermoMLData	A component which will import all supported data from the NIST ThermoML archive for (optionally) specified journals.

# **ImportThermoMLDataSchema**

 ${\bf class}\ open ff. evaluator. datasets. curation. components. thermoml. {\bf Import ThermoMLDataSchema} (*, note that the component of the co$ 

```
type:
Lit-
eral['ImportThermoM.
'Im-
port-
Ther-
moML-
Data',
tain_uncertainties:
bool
True,
cache_file_name:
Op-
tional[str]
None,
root_archive_url:
HttpUrl
'https://data.nist.gov/o
2422/ThermoML.v202
09-
```

30.tgz')

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	

## **Attributes**

type
retain\_uncertainties
cache\_file\_name
root\_archive\_url

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

**Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) → None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **ImportThermoMLData**

class openff.evaluator.datasets.curation.components.thermoml.ImportThermoMLData

A component which will import all supported data from the NIST ThermoML archive for (optionally) specified journals.

```
__init__()
```

#### **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

# **OpenFF Evaluator Documentation**

**Return type** The data set which has had the component applied to it.

# Data Point Selection

SelectSubstancesSchema	
SelectSubstances	A component for selecting a specified number data points which were measured for systems containing a specified set of chemical functionalities.
SelectDataPointsSchema	
SelectDataPoints	A component for selecting a set of data points which are measured as close as possible to a particular set of states.
State	
TargetState	
FingerPrintType	An enumeration.

# SelectSubstancesSchema

```
class openff.evaluator.datasets.curation.components.selection.SelectSubstancesSchema(*,
```

```
type:
Lit-
eral['SelectSubstances']
'Se-
lect-
Sub-
stances',
tar-
get_environments:
Con-
strainedList-
Value[ChemicalEnviron
n_per_environment:
Posi-
tiveInt,
sub-
stances\_to\_exclude:
List[Tuple[str,
...]]
None,
fin-
ger_print_type:
Fin-
ger-
Print-
Type
Fin-
ger-
Print-
Type.Tree,
per_property:
bool)
```

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
	data from keyword arguments.
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.
from_orm(obj)	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .
parse_file(path, *[, content_type,])	
parse_obj(obj)	
parse_raw(b, *[, content_type, encoding,])	
schema([by_alias, ref_template])	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.
validate(value)	-
Attributes	
type	

target\_environments

n\_per\_environment

substances\_to\_exclude

finger\_print\_type

per\_property

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

```
copy(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model
```

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

```
dict(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False) → DictStrAny
```

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:
Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,
**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) \rightarrow None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

#### **SelectSubstances**

## class openff.evaluator.datasets.curation.components.selection.SelectSubstances

A component for selecting a specified number data points which were measured for systems containing a specified set of chemical functionalities.

```
__init__()
```

## **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

#### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- **n\_processes** The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# SelectDataPointsSchema

class openff.evaluator.datasets.curation.components.selection.SelectDataPointsSchema(\*,

```
type:
Lit-
eral['SelectDataPoints']
=
'Se-
lect-
Data-
Points',
tar-
get_states:
List[TargetState])
```

```
__init__(**data: Any) \rightarrow None
```

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

init(**data)	Create a new model by parsing and validating input
IIII ( · · data)	* * * * * * * * * * * * * * * * * * * *
	data from keyword arguments.
<pre>construct([_fields_set])</pre>	Creates a new model settingdict and
	fields_set from trusted or pre-validated data.
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to
• • •	include, exclude and change.
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model,
aree ([, merade, exerade, ey_ands,])	optionally specifying which fields to include or ex-
	clude.
<pre>from_orm(obj)</pre>	
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-
	clude and exclude arguments as per dict().
parse_file(path, *[, content_type,])	
F(F, [,	
parse_obj(obj)	
parse_obj(obj)	
(1 4 5 ( ) ( ) ( ) ( )	
<pre>parse_raw(b, *[, content_type, encoding,])</pre>	
<pre>schema([by_alias, ref_template])</pre>	
schema_json(*[, by_alias, ref_template])	
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this
apaate_rorwara_rers( localis)	
	Model, globalns and localns.
validate(value)	

## **Attributes**

type
target\_states

 $\textbf{classmethod construct}(\_\textit{fields\_set: Optional[SetStr]} = \textit{None}, **values: Any}) \rightarrow \textbf{Model}$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

## **Parameters**

• include – fields to include in new model

- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

#### **Returns** new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
 Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
 Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
 bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,

**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) \rightarrow None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

## **SelectDataPoints**

#### class openff.evaluator.datasets.curation.components.selection.SelectDataPoints

A component for selecting a set of data points which are measured as close as possible to a particular set of states.

The points will be chosen so as to try and maximise the number of properties measured at the same condition (e.g. ideally we would have a data point for each property at T=298.15 and p=1atm) as this will maximise the chances that we can extract all properties from a single simulation.

```
__init__()
```

## **Methods**

\_\_init\_\_()

apply(data\_set, schema[, n\_processes])

Apply this curation component to a data set.

### **classmethod** apply(data set, schema, n processes=1)

Apply this curation component to a data set.

### **Parameters**

- data\_set The data frame to apply the component to.
- schema The schema which defines how this component should be applied.
- n\_processes The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# State

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating inpu				
	data from keyword arguments.				
<pre>construct([_fields_set])</pre>	Creates a new model settingdict and				
	fields_set from trusted or pre-validated data.				
<pre>copy(*[, include, exclude, update, deep])</pre>	Duplicate a model, optionally choose which fields to				
	include, exclude and change.				
<pre>dict(*[, include, exclude, by_alias,])</pre>	Generate a dictionary representation of the model,				
	optionally specifying which fields to include or ex-				
	clude.				
<pre>from_orm(obj)</pre>					
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, in-				
	clude and exclude arguments as per dict().				
<pre>parse_file(path, *[, content_type,])</pre>					
parse_obj(obj)					
<pre>parse_raw(b, *[, content_type, encoding,])</pre>					
schema([by_alias, ref_template])					
<pre>schema_json(*[, by_alias, ref_template])</pre>					
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this				
	Model, globalns and localns.				
validate(value)					

### **Attributes**

temperature

pressure

mole\_fractions

classmethod construct (fields set: Optional[SetStr] = None, \*\*values: Any)  $\rightarrow$  Model

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude: \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \\ \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

### Returns new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,

\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

## classmethod update\_forward\_refs(\*\*localns: Any) $\rightarrow$ None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **TargetState**

**\_\_init\_\_**(\*\*data: Any)  $\rightarrow$  None

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating input data from keyword arguments.				
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.				
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.				
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.				
from_orm(obj)					
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>in-clude</i> and <i>exclude</i> arguments as per <i>dict()</i> .				
parse_file(path, *[, content_type,])					
parse_obj(obj)					
parse_raw(b, *[, content_type, encoding,])					
property_types_validator(value)					
schema([by_alias, ref_template])					
<pre>schema_json(*[, by_alias, ref_template])</pre>					
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.				
validate(value)					

## **Attributes**

property\_types
states

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any$ )  $\rightarrow$  Model

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

 $\begin{array}{l} \textbf{copy}(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:} \\ Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, \\ deep: bool = False) \rightarrow \textbf{Model} \\ \end{array}$ 

Duplicate a model, optionally choose which fields to include, exclude and change.

#### **Parameters**

- include fields to include in new model
- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

Returns new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

json(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:
Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:
bool = False, encoder: Optional[Callable[[Any], Any]] = None, models\_as\_dict: bool = True,
\*\*dumps\_kwargs: Any) → unicode

Generate a JSON representation of the model, include and exclude arguments as per dict().

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

classmethod update\_forward\_refs(\*\*localns: Any)  $\rightarrow$  None

Try to update ForwardRefs on fields based on this Model, globalns and localns.

# **FingerPrintType**

${\bf class} \ {\tt openff.evaluator.datasets.curation.comp}$	oonents.selection. <b>FingerPrintType</b> (value)
An enumeration.	
init()	
Attributes	
Tree	
MACCS166	
Data Conversion	
ConvertExcessDensityDataSchema	
ConvertExcessDensityData	A component for converting binary mass density data to excess molar volume data and vice versa where pure density data measured for the components is available.

# ConvertExcessDensityDataSchema

class openff.evaluator.datasets.curation.components.conversion.ConvertExcessDensityDataSchema(\*,

```
type:
Lit-
eral['Conve
'Con-
ver-
tEx-
cess-
Den-
si-
ty-
DataScheme
tem-
per-
a-
ture_precisi
Con-
strained-
Int-
Value
2,
pres-
sure_precisi
Con-
strained-
Int-
Value
1)
```

**\_\_init\_\_(\*\***  $data: Any) \rightarrow None$ 

Create a new model by parsing and validating input data from keyword arguments.

Raises ValidationError if the input data cannot be parsed to form a valid model.

## **Methods**

init(**data)	Create a new model by parsing and validating input				
. ,	data from keyword arguments.				
construct([_fields_set])	Creates a new model settingdict andfields_set from trusted or pre-validated data.				
copy(*[, include, exclude, update, deep])	Duplicate a model, optionally choose which fields to include, exclude and change.				
dict(*[, include, exclude, by_alias,])	Generate a dictionary representation of the mode optionally specifying which fields to include or exclude.				
from_orm(obj)					
<pre>json(*[, include, exclude, by_alias,])</pre>	Generate a JSON representation of the model, <i>include</i> and <i>exclude</i> arguments as per <i>dict()</i> .				
<pre>parse_file(path, *[, content_type,])</pre>					
parse_obj(obj)					
parse_raw(b, *[, content_type, encoding,])					
schema([by_alias, ref_template])					
schema_json(*[, by_alias, ref_template])					
update_forward_refs(**localns)	Try to update ForwardRefs on fields based on this Model, globalns and localns.				
validate(value)					

## **Attributes**

type
temperature\_precision
pressure\_precision

classmethod construct( $\_fields\_set: Optional[SetStr] = None, **values: Any) \rightarrow Model$ 

Creates a new model setting \_\_dict\_\_ and \_\_fields\_set\_\_ from trusted or pre-validated data. Default values are respected, but no other validation is performed. Behaves as if *Config.extra* = 'allow' was set since it adds all passed values

**copy**(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:
Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, update: Optional[DictStrAny] = None, deep: bool = False) → Model

Duplicate a model, optionally choose which fields to include, exclude and change.

## **Parameters**

• include – fields to include in new model

- exclude fields to exclude from new model, as with values this takes precedence over include
- **update** values to change/add in the new model. Note: the data is not validated before creating the new model: you should trust this data
- **deep** set to *True* to make a deep copy of the model

### Returns new model instance

dict(\*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by\_alias: bool = False, skip\_defaults:

Optional[bool] = None, exclude\_unset: bool = False, exclude\_defaults: bool = False, exclude\_none:

bool = False) → DictStrAny

Generate a dictionary representation of the model, optionally specifying which fields to include or exclude.

```
json(*, include: Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, exclude:

Optional[Union[AbstractSetIntStr, MappingIntStrAny]] = None, by_alias: bool = False, skip_defaults:

Optional[bool] = None, exclude_unset: bool = False, exclude_defaults: bool = False, exclude_none:

bool = False, encoder: Optional[Callable[[Any], Any]] = None, models_as_dict: bool = True,

**dumps_kwargs: Any) → unicode
```

Generate a JSON representation of the model, *include* and *exclude* arguments as per *dict()*.

encoder is an optional function to supply as default to json.dumps(), other arguments as per json.dumps().

```
classmethod update_forward_refs(**localns: Any) → None
```

Try to update ForwardRefs on fields based on this Model, globalns and localns.

### ConvertExcessDensityData

### class openff.evaluator.datasets.curation.components.conversion.ConvertExcessDensityData

A component for converting binary mass density data to excess molar volume data and vice versa where pure density data measured for the components is available.

### **Notes**

This protocol may result in duplicate data points being generated. It is recommended to apply the de-duplication filter after this component has been applied.

```
__init__()
```

## **Methods**

```
__init__()

apply(data_set, schema[, n_processes])

Apply this curation component to a data set.
```

classmethod apply(data\_set, schema, n\_processes=1)

Apply this curation component to a data set.

### **Parameters**

- **data\_set** The data frame to apply the component to.
- schema The schema which defines how this component should be applied.

• **n\_processes** – The number of processes that this component is allowed to parallelize across.

**Return type** The data set which has had the component applied to it.

# 2.32.5 Force Field API

ForceFieldSource	A helper object to define the source of a force field and any associated meta data, such as version, file paths, or generation options.
SmirnoffForceFieldSource	A wrapper around force fields based on the SMIRks Native Open Force Field (SMIRNOFF) specification.
TLeapForceFieldSource	A wrapper around Amber force fields which may be applied via the <i>tleap</i> software package.
LigParGenForceFieldSource	A wrapper and the OPLSAAM force field which can be applied via the LigParGen server.

## **ForceFieldSource**

## class openff.evaluator.forcefield.ForceFieldSource

A helper object to define the source of a force field and any associated meta data, such as version, file paths, or generation options.

## **Notes**

It is likely that this class and classes based off of it will not be permanent fixtures of the framework, but rather will exist until the force fields can be stored in a uniform format / object model.

## **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

**Return type** Any

### **SmirnoffForceFieldSource**

## class openff.evaluator.forcefield.SmirnoffForceFieldSource(inner\_xml=None)

A wrapper around force fields based on the SMIRks Native Open Force Field (SMIRNOFF) specification.

**\_\_init\_\_**(inner\_xml=None)

Constructs a new SmirnoffForceFieldSource object

**Parameters inner\_xml** (str, optional) – A string containing the xml representation of the force field.

## **Methods**

init([inner_xml])	Constructs a new SmirnoffForceFieldSource object		
<pre>from_json(file_path)</pre>	Create this object from a JSON file.		
<pre>from_object(force_field)</pre>	Creates a new SmirnoffForceFieldSource from an ex-		
	isting ForceField object		
<pre>from_path(file_path)</pre>	Creates a new SmirnoffForceFieldSource from the		
	file path to a ForceField object.		
json([file_path, format])	Creates a JSON representation of this class.		
parse_json(string_contents)	Parses a typed json string into the corresponding class		
	structure.		
to_force_field()	Returns the SMIRNOFF force field created from this		
	source.		

## to\_force\_field()

Returns the SMIRNOFF force field created from this source.

**Returns** The created force field.

Return type ForceField

# classmethod from\_object(force\_field)

Creates a new SmirnoffForceFieldSource from an existing ForceField object

## **Notes**

All cosmetic attributes will be discarded.

Parameters force\_field (ForceField) - The existing force field.

**Returns** The created object.

Return type SmirnoffForceFieldSource

## classmethod from\_path(file\_path)

Creates a new SmirnoffForceFieldSource from the file path to a ForceField object.

### **Notes**

All cosmetic attributes will be discarded.

**Parameters file\_path** (*str*) – The file path to the force field object. This may also be the name of a file which can be loaded via an entry point.

**Returns** The created object.

Return type SmirnoffForceFieldSource

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

Return type Any

## **TLeapForceFieldSource**

A wrapper around Amber force fields which may be applied via the *tleap* software package.

## **Notes**

Currently this only supports force fields which are installed alongside *tleap*.

```
__init__(leap_source='leaprc.gaff2', cutoff=9.0 * unit.angstrom)
```

Constructs a new TLeapForceFieldSource object

### **Parameters**

- **leap\_source** (*str*) The parameter file which should be sourced by *leap* when applying the force field. Currently only '*leaprc.gaff*' and '*leaprc.gaff*2' are supported.
- **cutoff** (*openff.evaluator.unit.Quantity*) The non-bonded interaction cutoff.

## **Examples**

To create a source for the GAFF force field with tip3p water:

```
>>> amber_gaff_source = TLeapForceFieldSource('leaprc.gaff')
```

To create a source for the GAFF 2 force field with tip3p water:

```
>>> amber_gaff_2_source = TLeapForceFieldSource('leaprc.gaff2')
```

### **Methods**

init([leap_source, cutoff])	Constructs a new TLeapForceFieldSource object		
from_json(file_path)	Create this object from a JSON file.		
json([file_path, format])	Creates a JSON representation of this class.		
parse_json(string_contents)	Parses a typed json string into the corresponding class		
	structure.		

## **Attributes**

cutoff	The non-bonded interaction cutoff.			
leap_source	The parameter file which should be sourced by <i>leap</i>			
	when applying the force field.			

## property leap\_source

The parameter file which should be sourced by *leap* when applying the force field.

Type list of str

## property cutoff

The non-bonded interaction cutoff.

**Type** openff.evaluator.unit.Quantity

## classmethod from\_json(file path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

## LigParGenForceFieldSource

A wrapper and the OPLSAAM force field which can be applied via the LigParGen server.

## References

- [1] Potential energy functions for atomic-level simulations of water and organic and biomolecular systems. Jorgensen, W. L.; Tirado-Rives, J. Proc. Nat. Acad. Sci. USA 2005, 102, 6665-6670
- [2] 1.14\*CM1A-LBCC: Localized Bond-Charge Corrected CM1A Charges for Condensed-Phase Simulations. Dodda, L. S.; Vilseck, J. Z.; Tirado-Rives, J.; Jorgensen, W. L. J. Phys. Chem. B, 2017, 121 (15), pp 3864-3870
- [3] LigParGen web server: An automatic OPLS-AA parameter generator for organic ligands. Dodda, L. S.;Cabeza de Vaca, I.; Tirado-Rives, J.; Jorgensen, W. L. Nucleic Acids Research, Volume 45, Issue W1, 3 July 2017, Pages W331-W336

\_\_init\_\_(preferred\_charge\_model=ChargeModel.CM1A\_1\_14\_LBCC, cutoff=9.0 \* unit.angstrom, request\_url=", download\_url=")

Constructs a new LigParGenForceFieldSource object

#### **Parameters**

- **preferred\_charge\_model** (ChargeModel) The preferred charge model to apply. In some cases the preferred charge model may not be applicable (e.g. 1.14\*CM1A-LBCC may only be applied to neutral molecules) and so another model may be applied in its place.
- **cutoff** (openff.evaluator.unit.Quantity) The non-bonded interaction cutoff.
- **request\_url** (*str*) The URL of the LIGPARGEN server file to send the parametrization to request to.
- **download\_url** (*str*) The URL of the LIGPARGEN server file to download the results of a request from.

### **Methods**

init([preferred_charge_model, cutoff,])	Constructs a new LigParGenForceFieldSource object		
from_json(file_path)	Create this object from a JSON file.		
json([file_path, format])	Creates a JSON representation of this class.		
parse_json(string_contents)	Parses a typed json string into the corresponding class		
	structure.		

## **Attributes**

cutoff	The non-bonded interaction cutoff.
download_url	The URL of the LIGPARGEN server file to download
	the results of a request from.
preferred_charge_model	The preferred charge model to apply.
request_url	The URL of the LIGPARGEN server file to send the
	parametrization to request to.

## class ChargeModel(value)

An enumeration.

## property preferred\_charge\_model

The preferred charge model to apply. In some cases the preferred charge model may not be applicable (e.g. 1.14\*CM1A-LBCC may only be applied to neutral molecules) and so another model may be applied in its place.

Type ChargeModel

## property cutoff

The non-bonded interaction cutoff.

Type openff.evaluator.unit.Quantity

## property request\_url

The URL of the LIGPARGEN server file to send the parametrization to request to.

Type str

## property download\_url

The URL of the LIGPARGEN server file to download the results of a request from.

```
Type str
```

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

Returns The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

Return type Any

## **Gradient Estimation**

ParameterGradientKey

ParameterGradient

## **ParameterGradientKey**

**class** openff.evaluator.forcefield.**ParameterGradientKey**(tag=None, smirks=None, attribute=None)

```
__init__(tag=None, smirks=None, attribute=None)
```

## **Methods**

```
__init__([tag, smirks, attribute])
```

## **Attributes**

attribute			
smirks			
tag			

# **ParameterGradient**

class openff.evaluator.forcefield.ParameterGradient(key=None, value=None)
 \_\_init\_\_(key=None, value=None)

## **Methods**

\_\_init\_\_([key, value])

# **Attributes**

key value

# 2.32.6 Calculation Layers API

CalculationLayer	An abstract representation of a calculation layer whose
	goal is to estimate a set of physical properties using a sin-
	gle approach, such as a layer which employs direct sim-
	ulations to estimate properties, or one which reweights
	cached simulation data to the same end.
CalculationLayerResult	The result of attempting to estimate a property using a
	CalculationLayer.
CalculationLayerSchema	A schema which encodes the options that a Calculation-
	Layer should use when estimating a given class of phys-
	ical properties.
calculation_layer	A decorator which registers a class as being a calculation
	layer which may be used in property calculations.
register_calculation_layer	Registers a class as being a calculation layer which may
	be used in property calculations.
register_calculation_schema	Registers the default calculation schema to use when es-
	timating a class of properties (e.g.
register_calculation_schema	2

## CalculationLayer

## class openff.evaluator.layers.CalculationLayer

An abstract representation of a calculation layer whose goal is to estimate a set of physical properties using a single approach, such as a layer which employs direct simulations to estimate properties, or one which reweights cached simulation data to the same end.

## **Methods**

init()	
required_schema_type()	Returns the type of <i>CalculationLayerSchema</i> required by this layer.
schedule_calculation(calculation_backend,)	Submit the proposed calculation to the backend of
	choice.

## abstract classmethod required\_schema\_type()

Returns the type of CalculationLayerSchema required by this layer.

**Returns** The required schema type.

Return type type of CalculationLayerSchema

Submit the proposed calculation to the backend of choice.

### **Parameters**

- **calculation\_backend** (CalculationBackend) The backend to the submit the calculations to.
- **storage\_backend** (StorageBackend) The backend used to store / retrieve data from previous calculations.
- **layer\_directory** (*str*) The directory in which to store all temporary calculation data from this layer.
- batch (Batch) The batch of properties to estimate with the layer.
- **callback** (*function*) The function to call when the backend returns the results (or an error).
- **synchronous** (*bool*) If true, this function will block until the calculation has completed. This is mainly intended for debugging purposes.

## CalculationLayerResult

## class openff.evaluator.layers.CalculationLayerResult

The result of attempting to estimate a property using a *CalculationLayer*.

\_\_init\_\_()

### **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

data_to_store	Paths to the data objects to store.
exceptions	Any exceptions raised by the layer while estimating
	the property.
physical_property	The estimated property (if the layer was successful).

## physical\_property

The estimated property (if the layer was successful). The default value of this attribute is not set. This attribute is *optional*.

**Type** PhysicalProperty

## data\_to\_store

Paths to the data objects to store. The default value of this attribute is [].

Type list

## exceptions

Any exceptions raised by the layer while estimating the property. The default value of this attribute is [].

Type list

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to validate.

Raises ValueError or AssertionError -

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

## Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

# CalculationLayerSchema

## class openff.evaluator.layers.CalculationLayerSchema

A schema which encodes the options that a *CalculationLayer* should use when estimating a given class of physical properties.

\_\_init\_\_()

## **Methods**

Create this object from a JSON file.
Returns all attributes of a specific attribute_type.
Creates a JSON representation of this class.
Parses a typed json string into the corresponding class
structure.
Validate the values of the attributes.

### **Attributes**

absolute_tolerance	The absolute uncertainty that the property should be
	estimated to within.
relative_tolerance	The relative uncertainty that the property should be
	estimated to within, i.e relative_tolerance * mea-
	sured_property.uncertainty.

### absolute\_tolerance

The absolute uncertainty that the property should be estimated to within. This attribute is mutually exclusive with the *relative\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

### relative\_tolerance

The relative uncertainty that the property should be estimated to within, i.e *relative\_tolerance \* measured\_property.uncertainty*. This attribute is mutually exclusive with the *absolute\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type float

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

### calculation layer

```
openff.evaluator.layers.calculation_layer()
```

A decorator which registers a class as being a calculation layer which may be used in property calculations.

## register\_calculation\_layer

```
openff.evaluator.layers.register_calculation_layer(layer_class)
```

Registers a class as being a calculation layer which may be used in property calculations.

Parameters layer\_class (type of CalculationLayer) – The calculation layer to register.

## register\_calculation\_schema

```
openff.evaluator.layers.register_calculation_schema(property_class, layer_class, schema)
```

Registers the default calculation schema to use when estimating a class of properties (e.g. *Density*) with a specific calculation layer (e.g. the *SimulationLayer*).

#### **Parameters**

- **property\_class** (*type of PhysicalProperty*) The class of properties to associate with the specified *calculation\_layer* and *property\_class*.
- layer\_class (type of CalculationLayer) The calculation layer to associate the schema with.
- schema (CalculationLayerSchema or Callable[[CalculationLayerSchema], CalculationLayerSchema]) Either the calculation schema to use, or a function which will create the schema from an existing CalculationLayerSchema.

## **Built-in Calculation Layers**

WorkflowCalculationLayer	An calculation layer which uses the built-in workflow
	framework to estimate sets of physical properties.
WorkflowCalculationSchema	A schema which encodes the options and the workflow schema that a <i>CalculationLayer</i> should use when estimating a given class of physical properties using the built-in workflow framework.

## WorkflowCalculationLayer

## class openff.evaluator.layers.workflow.WorkflowCalculationLayer

An calculation layer which uses the built-in workflow framework to estimate sets of physical properties.

\_\_init\_\_()

### **Methods**

init()	
required_schema_type()	Returns the type of CalculationLayerSchema re-
	quired by this layer.
schedule_calculation(calculation_backend,)	Submit the proposed calculation to the backend of
	choice.
<pre>workflow_to_layer_result(queued_properties,</pre>	Converts a list of WorkflowResult to a list of Calcu-
)	lationLayerResult objects.

static workflow\_to\_layer\_result(queued\_properties, provenance, workflow\_results, \*\*\_)

Converts a list of WorkflowResult to a list of CalculationLayerResult objects.

#### **Parameters**

- **queued\_properties** (*list of PhysicalProperty*) The properties being estimated by this layer
- **provenance** (*dict of str and str*) The provenance of each property.
- workflow\_results (list of WorkflowResult) The results of each workflow.

**Returns** The calculation layer result objects.

**Return type** list of CalculationLayerResult

## abstract classmethod required\_schema\_type()

Returns the type of CalculationLayerSchema required by this layer.

**Returns** The required schema type.

**Return type** type of CalculationLayerSchema

Submit the proposed calculation to the backend of choice.

## **Parameters**

- calculation\_backend (CalculationBackend) The backend to the submit the calculations to.
- **storage\_backend** (StorageBackend) The backend used to store / retrieve data from previous calculations.
- **layer\_directory** (*str*) The directory in which to store all temporary calculation data from this layer.
- batch (Batch) The batch of properties to estimate with the layer.

- **callback** (*function*) The function to call when the backend returns the results (or an error).
- **synchronous** (*bool*) If true, this function will block until the calculation has completed. This is mainly intended for debugging purposes.

## WorkflowCalculationSchema

## class openff.evaluator.layers.workflow.WorkflowCalculationSchema

A schema which encodes the options and the workflow schema that a *CalculationLayer* should use when estimating a given class of physical properties using the built-in workflow framework.

#### **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

absolute_tolerance	The absolute uncertainty that the property should be
	estimated to within.
relative_tolerance	The relative uncertainty that the property should be
	estimated to within, i.e relative_tolerance * mea-
	sured_property.uncertainty.
workflow_schema	The workflow schema to use when estimating prop-
	erties.

## workflow\_schema

The workflow schema to use when estimating properties. The default value of this attribute is not set and must be set by the user..

Type WorkflowSchema

### validate(attribute type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## absolute\_tolerance

The absolute uncertainty that the property should be estimated to within. This attribute is mutually exclusive with the *relative\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## relative\_tolerance

The relative uncertainty that the property should be estimated to within, i.e *relative\_tolerance* \* *measured\_property.uncertainty*. This attribute is mutually exclusive with the *absolute\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type float

SimulationLayer	A calculation layer which employs molecular simulation
	to estimate sets of physical properties.
SimulationSchema	A schema which encodes the options and the workflow
	schema that the SimulationLayer should use when es-
	timating a given class of physical properties using the
	built-in workflow framework.

## **SimulationLayer**

## class openff.evaluator.layers.simulation.SimulationLayer

A calculation layer which employs molecular simulation to estimate sets of physical properties.

\_\_init\_\_()

### **Methods**

init()	
required_schema_type()	Returns the type of CalculationLayerSchema re-
	quired by this layer.
schedule_calculation(calculation_backend,)	Submit the proposed calculation to the backend of
	choice.
<pre>workflow_to_layer_result(queued_properties,</pre>	Converts a list of WorkflowResult to a list of Calcu-
)	lationLayerResult objects.

### classmethod required\_schema\_type()

Returns the type of CalculationLayerSchema required by this layer.

**Returns** The required schema type.

Return type type of CalculationLayerSchema

Submit the proposed calculation to the backend of choice.

#### **Parameters**

- calculation\_backend (CalculationBackend) The backend to the submit the calculations to.
- **storage\_backend** (StorageBackend) The backend used to store / retrieve data from previous calculations.
- **layer\_directory** (*str*) The directory in which to store all temporary calculation data from this layer.
- batch (Batch) The batch of properties to estimate with the layer.
- **callback** (*function*) The function to call when the backend returns the results (or an error).
- **synchronous** (*bool*) If true, this function will block until the calculation has completed. This is mainly intended for debugging purposes.

static workflow\_to\_layer\_result(queued\_properties, provenance, workflow\_results, \*\*\_)

Converts a list of WorkflowResult to a list of CalculationLayerResult objects.

## **Parameters**

- **queued\_properties** (list of PhysicalProperty) The properties being estimated by this layer
- **provenance** (*dict of str and str*) The provenance of each property.
- workflow\_results (list of WorkflowResult) The results of each workflow.

**Returns** The calculation layer result objects.

Return type list of CalculationLayerResult

## **SimulationSchema**

## class openff.evaluator.layers.simulation.SimulationSchema

A schema which encodes the options and the workflow schema that the *SimulationLayer* should use when estimating a given class of physical properties using the built-in workflow framework.

### **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
get_attributes([attribute_type])	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

absolute_tolerance	The absolute uncertainty that the property should be estimated to within.
relative_tolerance	The relative uncertainty that the property should be estimated to within, i.e <i>relative_tolerance</i> * <i>measured_property.uncertainty</i> .
workflow_schema	The workflow schema to use when estimating properties.

### absolute\_tolerance

The absolute uncertainty that the property should be estimated to within. This attribute is mutually exclusive with the *relative\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters** string\_contents (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** *Any* 

## relative\_tolerance

The relative uncertainty that the property should be estimated to within, i.e *relative\_tolerance* \* *measured\_property.uncertainty*. This attribute is mutually exclusive with the *absolute\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type float

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## workflow\_schema

The workflow schema to use when estimating properties. The default value of this attribute is not set and must be set by the user..

### Type WorkflowSchema

ReweightingLayer	A <i>CalculationLayer</i> which attempts to 'reweight' cached simulation data to evaluate the values of properties at states which have not previously been simulated directly, but where simulations at similar states have been run previously.
ReweightingSchema	A schema which encodes the options and the workflow schema that the <i>SimulationLayer</i> should use when estimating a given class of physical properties using the built-in workflow framework.
default_storage_query	Return the default query to use when retrieving cached simulation

## ReweightingLayer

## class openff.evaluator.layers.reweighting.ReweightingLayer

A *CalculationLayer* which attempts to 'reweight' cached simulation data to evaluate the values of properties at states which have not previously been simulated directly, but where simulations at similar states have been run previously.

\_\_init\_\_()

## **Methods**

init()	
required_schema_type()	Returns the type of <i>CalculationLayerSchema</i> required by this layer.
schedule_calculation(calculation_backend,)	Submit the proposed calculation to the backend of choice.
<pre>workflow_to_layer_result(queued_properties,)</pre>	Converts a list of WorkflowResult to a list of CalculationLayerResult objects.

## classmethod required\_schema\_type()

Returns the type of CalculationLayerSchema required by this layer.

**Returns** The required schema type.

Return type type of CalculationLayerSchema

**classmethod schedule\_calculation**(calculation\_backend, storage\_backend, layer\_directory, batch, callback, synchronous=False)

Submit the proposed calculation to the backend of choice.

### **Parameters**

- **calculation\_backend** (CalculationBackend) The backend to the submit the calculations to.
- **storage\_backend** (StorageBackend) The backend used to store / retrieve data from previous calculations.
- **layer\_directory** (*str*) The directory in which to store all temporary calculation data from this layer.
- batch (Batch) The batch of properties to estimate with the layer.
- **callback** (*function*) The function to call when the backend returns the results (or an error).
- **synchronous** (*bool*) If true, this function will block until the calculation has completed. This is mainly intended for debugging purposes.

**static workflow\_to\_layer\_result**(queued\_properties, provenance, workflow\_results, \*\*\_)

Converts a list of WorkflowResult to a list of CalculationLayerResult objects.

### **Parameters**

• **queued\_properties** (*list of PhysicalProperty*) – The properties being estimated by this layer

- **provenance** (*dict of str and str*) The provenance of each property.
- workflow\_results (list of WorkflowResult) The results of each workflow.

**Returns** The calculation layer result objects.

Return type list of CalculationLayerResult

# ReweightingSchema

## class openff.evaluator.layers.reweighting.ReweightingSchema

A schema which encodes the options and the workflow schema that the *SimulationLayer* should use when estimating a given class of physical properties using the built-in workflow framework.

\_\_init\_\_()

## **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

The absolute uncertainty that the property should be
estimated to within.
The maximum number of data points to include as
part of the multi-state reweighting calculations.
The relative uncertainty that the property should be
estimated to within, i.e relative_tolerance * mea-
sured_property.uncertainty.
The queries to perform when retrieving data for each
of the components in the system from the storage
backend.
The maximum difference between the target temper-
ature and the temperature at which cached data was
collected to.
The workflow schema to use when estimating prop-
erties.

## storage\_queries

The queries to perform when retrieving data for each of the components in the system from the storage backend. The keys of this dictionary will correspond to the metadata keys made available to the workflow system.

Type dict

### maximum\_data\_points

The maximum number of data points to include as part of the multi-state reweighting calculations. If zero, no cap will be applied. The default value of this attribute is 4.

Type int

### temperature\_cutoff

The maximum difference between the target temperature and the temperature at which cached data was collected to. Data collected for temperatures outside of this cutoff will be ignored. The default value of this attribute is 5.0 kelvin.

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## absolute\_tolerance

The absolute uncertainty that the property should be estimated to within. This attribute is mutually exclusive with the *relative\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type Quantity

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

## **Return type** Any

### relative\_tolerance

The relative uncertainty that the property should be estimated to within, i.e *relative\_tolerance* \* *measured\_property.uncertainty*. This attribute is mutually exclusive with the *absolute\_tolerance* attribute. The default value of this attribute is not set. This attribute is *optional*.

Type float

### workflow\_schema

The workflow schema to use when estimating properties. The default value of this attribute is not set and must be set by the user..

Type WorkflowSchema

## default\_storage\_query

openff.evaluator.layers.reweighting.default\_storage\_query()

Return the default query to use when retrieving cached simulation data from the storage backend.

Currently this query will search for data for the full substance of interest in the liquid phase.

**Returns** A single query with a key of "full\_system\_data".

Return type dict of str and SimulationDataQuery

## 2.32.7 Calculation Backends API

CalculationBackend	An abstract base representation of an openff-evaluator calculation backend.
ComputeResources	An object which stores how many of each type of computational resource (threads or gpu's) is available to a calculation worker.
QueueWorkerResources	An extended resource object with properties specific to calculations which will run on queue based resources, such as LSF, PBS or SLURM.

### **CalculationBackend**

An abstract base representation of an openff-evaluator calculation backend. A backend is responsible for coordinating, distributing and running calculations on the available hardware. This may range from a single machine to a multinode cluster, but *not* across multiple cluster or physical locations.

## **Notes**

All estimator backend classes must inherit from this class, and must implement the *start*, *stop*, and *submit\_task* method.

**\_\_init\_\_**(number\_of\_workers=1, resources\_per\_worker=None)

Constructs a new CalculationBackend object.

### **Parameters**

- **number\_of\_workers** (*int*) The number of works to run the calculations on. One worker can perform a single task (e.g run a simulation) at once.
- resources\_per\_worker (ComputeResources, optional) The number of resources to request per worker.

### **Methods**

init([number_of_workers,])	Constructs a new CalculationBackend object.
start()	Start the calculation backend.
stop()	Stop the calculation backend.
<pre>submit_task(function, *args, **kwargs)</pre>	Submit a task to the compute resources managed by
	this backend.

## **Attributes**

started	Returns whether this backend has been started yet.
---------	--

## property started

Returns whether this backend has been started yet.

Type bool

start()

Start the calculation backend.

abstract stop()

Stop the calculation backend.

abstract submit\_task(function, \*args, \*\*kwargs)

Submit a task to the compute resources managed by this backend.

**Parameters function** (function) – The function to run.

**Returns** Returns a future object which will eventually point to the results of the submitted task.

Return type Future

## **ComputeResources**

An object which stores how many of each type of computational resource (threads or gpu's) is available to a calculation worker.

\_\_init\_\_(number\_of\_threads=1, number\_of\_gpus=0, preferred\_gpu\_toolkit=GPUToolkit.auto, preferred\_gpu\_precision=GPUPrecision.mixed)

Constructs a new ComputeResources object.

#### **Parameters**

- **number\_of\_threads** (*int*) The number of threads available to a calculation worker.
- number\_of\_gpus (int) The number of GPUs available to a calculation worker.
- **preferred\_gpu\_toolkit** (GPUToolkit, *optional*) The preferred toolkit to use when running on GPUs.
- **preferred\_gpu\_precision** (GPUPrecision, *optional*) The preferred precision level to use when runnin on GPUs.

## **Methods**

## **Attributes**

gpu_device_indices	The indices of the GPUs to run on.
number_of_gpus	The number of GPUs available to a calculation
	worker.
number_of_threads	The number of threads available to a calculation
	worker.
preferred_gpu_precision	The preferred precision level to use when running on
	GPUs.
preferred_gpu_toolkit	The preferred toolkit to use when running on GPUs.

## class GPUToolkit(value)

An enumeration of the different GPU toolkits to make available to different calculations.

## class GPUPrecision(value)

An enumeration of the different precision for GPU calculations.

### property number\_of\_threads

The number of threads available to a calculation worker.

Type int

## property number\_of\_gpus

The number of GPUs available to a calculation worker.

Type int

### property preferred\_gpu\_toolkit

The preferred toolkit to use when running on GPUs.

```
Type GPUToolkit
```

### property preferred\_gpu\_precision

The preferred precision level to use when running on GPUs.

```
Type GPUPrecision
```

## property gpu\_device\_indices

The indices of the GPUs to run on. This is purely an internal implementation detail and should not be relied upon externally.

Type str

#### QueueWorkerResources

An extended resource object with properties specific to calculations which will run on queue based resources, such as LSF, PBS or SLURM.

```
__init__(number_of_threads=1, number_of_gpus=0, preferred_gpu_toolkit=None, preferred_gpu_precision=None, per_thread_memory_limit=1 * unit.gigabytes, wallclock_time_limit='01:00')
```

Constructs a new ComputeResources object.

### **Notes**

Both the requested *number\_of\_threads* and the *number\_of\_gpus* must be less than or equal to the number of threads (/cpus/cores) and GPUs available to each compute node in the cluster respectively, such that a single worker is able to be accommodated by a single compute node.

## **Parameters**

- per\_thread\_memory\_limit (openmm.unit.Quantity) The maximum amount of memory available to each thread.
- wallclock\_time\_limit (str) The maximum amount of wall clock time that a worker can run for. This should be a string of the form HH:MM where HH is the number of hours and MM the number of minutes

## **Methods**

init([number_of_threads,]) Constructs a new ComputeResources object.
--

### **Attributes**

gpu_device_indices	The indices of the GPUs to run on.
number_of_gpus	The number of GPUs available to a calculation
	worker.
number_of_threads	The number of threads available to a calculation
	worker.
per_thread_memory_limit	The maximum amount of memory available to each
	thread, such that the total memory limit will be
	<pre>per_cpu_memory_limit * number_of_threads.</pre>
preferred_gpu_precision	The preferred precision level to use when running on
	GPUs.
preferred_gpu_toolkit	The preferred toolkit to use when running on GPUs.
wallclock_time_limit	The maximum amount of wall clock time that a
	worker can run for.

## property per\_thread\_memory\_limit

The maximum amount of memory available to each thread, such that the total memory limit will be  $per\_cpu\_memory\_limit * number\_of\_threads$ .

Type openmm.unit.Quantity

# property wallclock\_time\_limit

The maximum amount of wall clock time that a worker can run for. This should be a string of the form *HH:MM* where HH is the number of hours and MM the number of minutes

Type str

## class GPUPrecision(value)

An enumeration of the different precision for GPU calculations.

## class GPUToolkit(value)

An enumeration of the different GPU toolkits to make available to different calculations.

## property gpu\_device\_indices

The indices of the GPUs to run on. This is purely an internal implementation detail and should not be relied upon externally.

Type str

# property number\_of\_gpus

The number of GPUs available to a calculation worker.

Type int

## property number\_of\_threads

The number of threads available to a calculation worker.

Type int

## property preferred\_gpu\_precision

The preferred precision level to use when running on GPUs.

Type GPUPrecision

## property preferred\_gpu\_toolkit

The preferred toolkit to use when running on GPUs.

Type GPUToolkit

### **Dask Backends**

BaseDaskBackend	A base dask backend class, which implements function-
	ality which is common to all other dask based backends.
BaseDaskJobQueueBackend	An openff-evaluator backend which uses a
	dask_jobqueue.JobQueueCluster object to run cal-
	culations within an existing HPC queuing system.
DaskLocalCluster	An openff-evaluator backend which uses a dask Local-
	Cluster object to run calculations on a single machine.
DaskLSFBackend	An openff-evaluator backend which uses a
	dask_jobqueue.LSFCluster object to run calcula-
	tions within an existing LSF queue.
DaskPBSBackend	An openff-evaluator backend which uses a
	dask_jobqueue.PBSCluster object to run calcula-
	tions within an existing PBS queue.

## **BaseDaskBackend**

A base *dask* backend class, which implements functionality which is common to all other *dask* based backends.

\_\_init\_\_(number\_of\_workers=1, resources\_per\_worker=ComputeResources())

Constructs a new BaseDaskBackend object.

## **Methods**

init([number_of_workers,])	Constructs a new BaseDaskBackend object.
start()	Start the calculation backend.
stop()	Stop the calculation backend.
<pre>submit_task(function, *args, **kwargs)</pre>	Submit a task to the compute resources managed by
	this backend.

## **Attributes**

started

Returns whether this backend has been started yet.

```
start()
```

Start the calculation backend.

stop()

Stop the calculation backend.

#### property started

Returns whether this backend has been started yet.

Type bool

```
abstract submit_task(function, *args, **kwargs)
```

Submit a task to the compute resources managed by this backend.

**Parameters function** (function) – The function to run.

**Returns** Returns a future object which will eventually point to the results of the submitted task.

Return type Future

## BaseDaskJobQueueBackend

 $\textbf{class} \ open ff. evaluator. backends. \textbf{dask.BaseDaskJobQueueBackend} (\textit{minimum\_number\_of\_workers} = 1, \\$ 

```
maximum_number_of_workers=1,
re-
sources_per_worker=QueueWorkerResources(),
queue_name='default',
setup_script_commands=None,
extra_script_options=None,
adaptive_interval='10000ms',
disable_nanny_process=False,
cluster_type=None,
adaptive_class=None)
```

An openff-evaluator backend which uses a *dask\_jobqueue.JobQueueCluster* object to run calculations within an existing HPC queuing system.

#### See also:

```
dask_jobqueue.JobQueueCluster
```

```
__init__(minimum_number_of_workers=1, maximum_number_of_workers=1, resources_per_worker=QueueWorkerResources(), queue_name='default', setup_script_commands=None, extra_script_options=None, adaptive_interval='10000ms', disable_nanny_process=False, cluster_type=None, adaptive_class=None)
```

Constructs a new BaseDaskJobQueueBackend object

## **Parameters**

- $minimum_number_of_workers(int)$  The minimum number of workers to request from the queue system.
- maximum\_number\_of\_workers (int) The maximum number of workers to request from the queue system.

- resources\_per\_worker (QueueWorkerResources) The resources to request per worker.
- $queue\_name (str)$  The name of the queue which the workers will be requested from.
- **setup\_script\_commands** (*list of str*)—A list of bash script commands to call within the queue submission script before the call to launch the dask worker.

This may include activating a python environment, or loading an environment module

- extra\_script\_options (1ist of str) A list of extra job specific options to include
  in the queue submission script. These will get added to the script header in the form

  #BSUB <extra\_script\_options[x]>
- adaptive\_interval (str) The interval between attempting to either scale up or down the cluster, of of the from 'XXXms'.
- **disable\_nanny\_process** (*bool*) If true, dask workers will be started in *-no-nanny* mode. This is required if using multiprocessing code within submitted tasks.

This has not been fully tested yet and my lead to stability issues with the workers.

• adaptive\_class (class of type distributed.deploy.AdaptiveCore, optional) – An optional class to pass to dask to use for its adaptive scaling handling. This is mainly exposed to allow easily working around certain dask bugs / quirks.

#### Methods

init([minimum_number_of_workers,])	Constructs a new BaseDaskJobQueueBackend object
<pre>job_script()</pre>	Returns the job script that dask will use to submit
	workers.
start()	Start the calculation backend.
stop()	Stop the calculation backend.
<pre>submit_task(function, *args, **kwargs)</pre>	Submit a task to the compute resources managed by
	this backend.

#### **Attributes**

started	Returns whether this backend has been started yet.

## job\_script()

Returns the job script that dask will use to submit workers. The backend must be started before calling this function.

## Return type str

# start()

Start the calculation backend.

# submit\_task(function, \*args, \*\*kwargs)

Submit a task to the compute resources managed by this backend.

**Parameters function** (function) – The function to run.

**Returns** Returns a future object which will eventually point to the results of the submitted task.

## Return type Future

# property started

Returns whether this backend has been started yet.

Type bool

stop()

Stop the calculation backend.

## **DaskLocalCluster**

class openff.evaluator.backends.dask.DaskLocalCluster(number\_of\_workers=1,

resources\_per\_worker=ComputeResources())

An openff-evaluator backend which uses a *dask LocalCluster* object to run calculations on a single machine.

## See also:

dask.LocalCluster

**\_\_init\_\_**(number\_of\_workers=1, resources\_per\_worker=ComputeResources())

Constructs a new DaskLocalCluster

## **Methods**

init([number_of_workers,])	Constructs a new DaskLocalCluster
start()	Start the calculation backend.
stop()	Stop the calculation backend.
<pre>submit_task(function, *args, **kwargs)</pre>	Submit a task to the compute resources managed by
	this backend.

#### **Attributes**

started	Returns whether this backend has been started yet.

## start()

Start the calculation backend.

submit\_task(function, \*args, \*\*kwargs)

Submit a task to the compute resources managed by this backend.

**Parameters function** (*function*) – The function to run.

**Returns** Returns a future object which will eventually point to the results of the submitted task.

Return type Future

# property started

Returns whether this backend has been started yet.

Type bool

## stop()

Stop the calculation backend.

#### **DaskLSFBackend**

An openff-evaluator backend which uses a *dask\_jobqueue.LSFCluster* object to run calculations within an existing LSF queue.

#### See also:

## **Examples**

To create an LSF queueing compute backend which will attempt to spin up workers which have access to a single GPU.

```
>>> # Create a resource object which will request a worker with
>>> # one gpu which will stay alive for five hours.
>>> from openff.evaluator.backends import QueueWorkerResources
>>>
>>> resources = QueueWorkerResources(number_of_threads=1,
                                      number_of_gpus=1,
>>>
>>>
                                      preferred_gpu_toolkit=QueueWorkerResources.
→GPUToolkit.CUDA,
                                      preferred_gpu_
→precision=QueueWorkerResources.GPUPrecision.mixed,
                                      wallclock_time_limit='05:00')
>>>
>>> # Define the set of commands which will set up the correct environment
>>> # for each of the workers.
>>> setup_script_commands = [
        'module load cuda/9.2',
>>>
>>> ]
>>>
>>> # Define extra options to only run on certain node groups
>>> extra_script_options = [
        '-m "ls-gpu lt-gpu"
>>>
>>> ]
>>>
>>>
```

(continues on next page)

(continued from previous page)

```
>>> # Create the backend which will adaptively try to spin up between one and
>>> # ten workers with the requested resources depending on the calculation.
\rightarrow load.
>>> from openff.evaluator.backends.dask import DaskLSFBackend
>>>
>>> lsf_backend = DaskLSFBackend(minimum_number_of_workers=1,
>>>
                                  maximum_number_of_workers=10,
>>>
                                  resources_per_worker=resources,
                                  queue_name='gpuqueue',
>>>
                                  setup_script_commands=setup_script_commands,
>>>
                                  extra_script_options=extra_script_options)
>>>
```

#### **Methods**

init([minimum_number_of_workers,])	Constructs a new DaskLSFBackend object
job_script()	Returns the job script that dask will use to submit
	workers.
start()	Start the calculation backend.
stop()	Stop the calculation backend.
<pre>submit_task(function, *args, **kwargs)</pre>	Submit a task to the compute resources managed by
	this backend.

## **Attributes**

started Returns whether this backend has been started yet.	started	Returns whether this backend has been started yet.
--	---------	--

# job\_script()

Returns the job script that dask will use to submit workers. The backend must be started before calling this function.

## Return type str

## start()

Start the calculation backend.

# property started

Returns whether this backend has been started yet.

Type bool

## stop()

Stop the calculation backend.

```
submit_task(function, *args, **kwargs)
```

Submit a task to the compute resources managed by this backend.

**Parameters function** (function) – The function to run.

**Returns** Returns a future object which will eventually point to the results of the submitted task.

**Return type** Future

## **DaskPBSBackend**

An openff-evaluator backend which uses a *dask\_jobqueue.PBSCluster* object to run calculations within an existing PBS queue.

#### See also:

**Parameters resource\_line** (*str*) – The string to pass to the #*PBS* -*l* line.

#### **Examples**

To create a PBS queueing compute backend which will attempt to spin up workers which have access to a single GPU.

```
>>> # Create a resource object which will request a worker with
>>> # one gpu which will stay alive for five hours.
>>> from openff.evaluator.backends import QueueWorkerResources
>>> resources = QueueWorkerResources(number_of_threads=1,
>>>
                                     number_of_gpus=1,
                                     preferred_gpu_toolkit=QueueWorkerResources.
>>>
→GPUToolkit.CUDA,
                                     preferred_qpu_
>>>
→precision=QueueWorkerResources.GPUPrecision.mixed,
                                     wallclock_time_limit='05:00')
>>>
>>>
>>> # Define the set of commands which will set up the correct environment
>>> # for each of the workers.
>>> setup_script_commands = [
        'module load cuda/9.2',
>>>
>>> ]
>>>
>>> # Create the backend which will adaptively try to spin up between one and
>>> # ten workers with the requested resources depending on the calculation.
>>> from openff.evaluator.backends.dask import DaskPBSBackend
```

(continues on next page)

(continued from previous page)

```
>>> pbs_backend = DaskPBSBackend(minimum_number_of_workers=1,
>>> maximum_number_of_workers=10,
>>> resources_per_worker=resources,
>>> queue_name='gpuqueue',
>>> setup_script_commands=setup_script_commands)
```

## **Methods**

init([minimum_number_of_workers,])	Constructs a new DaskLSFBackend object
<pre>job_script()</pre>	Returns the job script that dask will use to submit
	workers.
start()	Start the calculation backend.
stop()	Stop the calculation backend.
<pre>submit_task(function, *args, **kwargs)</pre>	Submit a task to the compute resources managed by
	this backend.

# **Attributes**

started	Returns whether this backend has been started yet.

# job\_script()

Returns the job script that dask will use to submit workers. The backend must be started before calling this function.

Return type str

# start()

Start the calculation backend.

#### property started

Returns whether this backend has been started yet.

Type bool

# stop()

Stop the calculation backend.

```
submit_task(function, *args, **kwargs)
```

Submit a task to the compute resources managed by this backend.

**Parameters function** (*function*) – The function to run.

**Returns** Returns a future object which will eventually point to the results of the submitted task.

Return type Future

# 2.32.8 Storage API

StorageBackend	An abstract base representation of how the openff-
	evaluator will interact with and store simulation data.

# StorageBackend

## class openff.evaluator.storage.StorageBackend

An abstract base representation of how the openff-evaluator will interact with and store simulation data.

## **Notes**

When implementing this class, only private methods should be overridden as the public methods only mainly implement thread locks, while their private version perform their actual function.

Constructs a new StorageBackend object.

#### **Methods**

init()	Constructs a new StorageBackend object.
has_force_field(force_field)	A convenience method for checking whether the
	specified ForceFieldSource object is stored in the
	backend.
has_object(storage_object)	Checks whether a given hashable object exists in the
	storage system.
query(data_query)	Query the storage backend for data matching the
	query criteria.
retrieve_force_field(storage_key)	A convenience method for retrieving ForceField-
	Source objects.
<pre>retrieve_object(storage_key[, expected_type])</pre>	Retrieves a stored object for the estimators storage
	system.
store_force_field(force_field)	A convenience method for storing ForceFieldSource
	objects.
store_object(object_to_store[,])	Store an object in the storage system, returning the
	key of the stored object.

# store\_object(object\_to\_store, ancillary\_data\_path=None)

Store an object in the storage system, returning the key of the stored object. This may be different to *storage\_key* depending on whether the same or a similar object was already present in the system.

#### **Parameters**

- **object\_to\_store** (BaseStoredData) The object to store.
- ancillary\_data\_path (str, optional) The data path to the ancillary directory-like data to store alongside the object if the data type requires one.

Returns The unique key assigned to the stored object.

Return type str

## store\_field(force\_field)

A convenience method for storing ForceFieldSource objects.

**Parameters force\_field** (ForceFieldSource) – The force field to store.

**Returns** The unique id of the stored force field.

Return type str

## retrieve\_object(storage\_key, expected\_type=None)

Retrieves a stored object for the estimators storage system.

#### **Parameters**

- **storage\_key** (*str*) A unique key that describes where the stored object can be found within the storage system.
- **expected\_type** (*type of BaseStoredData*, *optional*) The expected data type. An exception is raised if the retrieved data doesn't match the type.

## Returns

- BaseStoredData, optional The stored object if the object key is found, otherwise None.
- str, optional The path to the ancillary data if present.

#### retrieve\_force\_field(storage\_key)

A convenience method for retrieving ForceFieldSource objects.

**Parameters storage\_key** (str) – The key of the force field to retrieve.

**Returns** The retrieved force field source.

Return type ForceFieldSource

# has\_object(storage\_object)

Checks whether a given hashable object exists in the storage system.

**Parameters** storage\_object (BaseStoredData) – The object to check for.

**Returns** The unique key of the object if it is in the system, *None* otherwise.

**Return type** *str*, optional

## has\_force\_field(force field)

A convenience method for checking whether the specified ForceFieldSource object is stored in the backend.

Parameters force\_field (ForceFieldSource) - The force field to look for.

**Returns** The unique key of the object if it is in the system, *None* otherwise.

Return type str, optional

# query(data\_query)

Query the storage backend for data matching the query criteria.

**Parameters data\_query** (BaseDataQuery) – The query to perform.

**Returns** The data that matches the query partitioned by the matched values. The list values take the form (storage\_key, data\_object, data\_directory\_path).

Return type dict of tuple and list of tuple of str, BaseStoredData and str

## **Built-in Storage Backends**

LocalFileStorage	A storage backend which stores files in directories on the
	local disk.

## LocalFileStorage

class openff.evaluator.storage.LocalFileStorage(root\_directory='stored\_data')

A storage backend which stores files in directories on the local disk.

\_\_init\_\_(root\_directory='stored\_data')

Constructs a new StorageBackend object.

#### **Methods**

init([root_directory])	Constructs a new StorageBackend object.
has_force_field(force_field)	A convenience method for checking whether the
	specified ForceFieldSource object is stored in the
	backend.
has_object(storage_object)	Checks whether a given hashable object exists in the
	storage system.
query(data_query)	Query the storage backend for data matching the
	query criteria.
retrieve_force_field(storage_key)	A convenience method for retrieving ForceField-
	Source objects.
retrieve_object(storage_key[, expected_type])	Retrieves a stored object for the estimators storage
	system.
store_force_field(force_field)	A convenience method for storing ForceFieldSource
	objects.
<pre>store_object(object_to_store[,])</pre>	Store an object in the storage system, returning the
	key of the stored object.

# **Attributes**

root_directory	Returns the directory in which all stored objects are
	located.

# property root\_directory

Returns the directory in which all stored objects are located.

Type str

has\_force\_field(force\_field)

A convenience method for checking whether the specified ForceFieldSource object is stored in the backend.

Parameters force\_field (ForceFieldSource) - The force field to look for.

**Returns** The unique key of the object if it is in the system, *None* otherwise.

Return type str, optional

#### has\_object(storage\_object)

Checks whether a given hashable object exists in the storage system.

**Parameters** storage\_object (BaseStoredData) – The object to check for.

**Returns** The unique key of the object if it is in the system, *None* otherwise.

Return type str, optional

#### query(data\_query)

Query the storage backend for data matching the query criteria.

**Parameters data\_query** (BaseDataQuery) – The query to perform.

**Returns** The data that matches the query partitioned by the matched values. The list values take the form (storage\_key, data\_object, data\_directory\_path).

Return type dict of tuple and list of tuple of str, BaseStoredData and str

## retrieve\_force\_field(storage\_key)

A convenience method for retrieving ForceFieldSource objects.

**Parameters storage\_key** (*str*) – The key of the force field to retrieve.

**Returns** The retrieved force field source.

Return type ForceFieldSource

## retrieve\_object(storage\_key, expected\_type=None)

Retrieves a stored object for the estimators storage system.

#### **Parameters**

- **storage\_key** (*str*) A unique key that describes where the stored object can be found within the storage system.
- **expected\_type** (*type of BaseStoredData*, *optional*) The expected data type. An exception is raised if the retrieved data doesn't match the type.

#### **Returns**

- BaseStoredData, optional The stored object if the object key is found, otherwise None.
- str, optional The path to the ancillary data if present.

# store\_field(force\_field)

A convenience method for storing ForceFieldSource objects.

**Parameters force\_field** (ForceFieldSource) – The force field to store.

**Returns** The unique id of the stored force field.

Return type str

```
store_object(object_to_store, ancillary_data_path=None)
```

Store an object in the storage system, returning the key of the stored object. This may be different to *storage\_key* depending on whether the same or a similar object was already present in the system.

#### **Parameters**

- **object\_to\_store** (BaseStoredData) The object to store.
- **ancillary\_data\_path** (*str*, *optional*) The data path to the ancillary directory-like data to store alongside the object if the data type requires one.

**Returns** The unique key assigned to the stored object.

## Return type str

## **Data Classes**

BaseStoredData	A base representation of cached data to be stored by a storage backend.
HashableStoredData	Represents a class of data objects which can be rapidly compared / indexed by their hash values.
ForceFieldData	A data container for force field objects which will be saved to disk.
ReplaceableData	Represents a piece of stored data which can be replaced in a <i>StorageBackend</i> by another piece of data of the same type.
BaseSimulationData	A base class for classes which will store the outputs of a molecular simulation
StoredSimulationData	A representation of data which has been cached from a single previous simulation.
StoredFreeEnergyData	A representation of data which has been cached from an free energy calculation which computed the free energy difference between a start and end state.

## **BaseStoredData**

# class openff.evaluator.storage.data.BaseStoredData

A base representation of cached data to be stored by a storage backend.

The expectation is that stored data may exist in storage as two parts:

- A JSON serialized representation of this class (or a subclass), which contains lightweight information such
  as the state and composition of the system. Any larger pieces of data, such as coordinates or trajectories,
  should be referenced as a file name.
- 2) A directory like structure (either directly a directory, or some NetCDF like compressed archive) of ancillary files which do not easily lend themselves to be serialized within a JSON object, whose files are referenced by their file name by the data object.

The ancillary directory-like structure is not required if the data may be suitably stored in the data object itself.

# **Methods**

get_attributes([attribute_type])Returnshas_ancillary_data()Returnspanying	is object from a JSON file.
has_ancillary_data() Returns panying	is object from a soot, file.
panying	all attributes of a specific attribute_type.
	whether this data object requires an accom-
icon([flo math format])	lata directory-like structure.
json([file_path, format]) Creates a	JSON representation of this class.
parse_json(string_contents) Parses as	yped json string into the corresponding class
structure	
to_storage_query() Returns	the storage query which would match this
data obje	ct.
validate([attribute_type]) Validate	

## abstract classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

**Returns** True if this class requires an accompanying data directory-like structure.

Return type bool

#### to\_storage\_query()

Returns the storage query which would match this data object.

**Returns** The storage query which would match this data object.

Return type BaseDataQuery

#### classmethod from\_json(file path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

# json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### classmethod parse\_json(string contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## **HashableStoredData**

## class openff.evaluator.storage.data.HashableStoredData

Represents a class of data objects which can be rapidly compared / indexed by their hash values.

\_\_init\_\_()

## **Methods**

init()	
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
has_ancillary_data()	Returns whether this data object requires an accom-
	panying data directory-like structure.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_storage_query()	Returns the storage query which would match this
	data object.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## abstract classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

Returns True if this class requires an accompanying data directory-like structure.

Return type bool

#### json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

## Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

## to\_storage\_query()

Returns the storage query which would match this data object.

**Returns** The storage query which would match this data object.

Return type BaseDataQuery

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **ForceFieldData**

# class openff.evaluator.storage.data.ForceFieldData

A data container for force field objects which will be saved to disk.

\_\_init\_\_()

## **Methods**

init()	
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
has_ancillary_data()	Returns whether this data object requires an accom-
	panying data directory-like structure.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_storage_query()	
	returns The storage query which would
	match this
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

force field source

The force field source object.

#### force\_field\_source

The force field source object. The default value of this attribute is not set and must be set by the user..

Type ForceFieldSource

#### classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

**Returns** True if this class requires an accompanying data directory-like structure.

Return type bool

# to\_storage\_query()

**Returns** The storage query which would match this data object.

Return type SimulationDataQuery

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

#### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## ReplaceableData

## class openff.evaluator.storage.data.ReplaceableData

Represents a piece of stored data which can be replaced in a *StorageBackend* by another piece of data of the same type.

This may be the case for example when attempting to store a piece of *StoredSimulationData*, but another piece of data measured from the same calculation and for the same system already exists in the system, but stores less configurations.

\_\_init\_\_()

#### **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
has_ancillary_data()	Returns whether this data object requires an accom-
	panying data directory-like structure.
json([file_path, format])	Creates a JSON representation of this class.
<pre>most_information(stored_data_1, stored_data_2)</pre>	Returns the data object with the highest information
	content.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_storage_query()	Returns the storage query which would match this
	data object.
validate([attribute_type])	Validate the values of the attributes.

# abstract classmethod most\_information(stored\_data\_1, stored\_data\_2)

Returns the data object with the highest information content.

#### **Parameters**

- **stored\_data\_1** (ReplaceableData) The first piece of data to compare.
- **stored\_data\_2** (ReplaceableData) The second piece of data to compare.

**Returns** The data object with the highest information content, or *None* if the two pieces of information are incompatible with one another.

Return type ReplaceableData, optional

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

# abstract classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

**Returns** True if this class requires an accompanying data directory-like structure.

Return type bool

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## to\_storage\_query()

Returns the storage query which would match this data object.

Returns The storage query which would match this data object.

Return type BaseDataQuery

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate

Raises ValueError or AssertionError -

# **BaseSimulationData**

# class openff.evaluator.storage.data.BaseSimulationData

A base class for classes which will store the outputs of a molecular simulation

\_\_init\_\_()

## **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
has_ancillary_data()	Returns whether this data object requires an accom-
	panying data directory-like structure.
json([file_path, format])	Creates a JSON representation of this class.
<pre>most_information(stored_data_1, stored_data_2)</pre>	Returns the data object with the highest information
	content.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_storage_query()	Returns the storage query which would match this
	data object.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

force_field_id	The id of the force field parameters used to generate
	the data.
property_phase	The phase of the system (e.g.
source_calculation_id	The server id of the calculation which yielded this
	data.
substance	A description of the composition of the stored sys-
	tem.
thermodynamic_state	The state at which the data was collected.

# substance

A description of the composition of the stored system. The default value of this attribute is not set and must be set by the user..

Type Substance

# thermodynamic\_state

The state at which the data was collected. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

# property\_phase

The phase of the system (e.g. liquid, gas). The default value of this attribute is not set and must be set by the user..

Type PropertyPhase

#### source\_calculation\_id

The server id of the calculation which yielded this data. The default value of this attribute is not set and must be set by the user..

Type str

## force\_field\_id

The id of the force field parameters used to generate the data. The default value of this attribute is not set and must be set by the user..

Type str

# classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

**Returns** True if this class requires an accompanying data directory-like structure.

Return type bool

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## abstract classmethod most\_information(stored\_data\_1, stored\_data\_2)

Returns the data object with the highest information content.

#### **Parameters**

- **stored\_data\_1** (ReplaceableData) The first piece of data to compare.
- **stored\_data\_2** (ReplaceableData) The second piece of data to compare.

**Returns** The data object with the highest information content, or *None* if the two pieces of information are incompatible with one another.

Return type ReplaceableData, optional

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

```
Parameters string_contents (str or bytes) – The typed json string.
```

**Returns** The parsed class.

Return type Any

## to\_storage\_query()

Returns the storage query which would match this data object.

**Returns** The storage query which would match this data object.

Return type BaseDataQuery

#### validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### **StoredSimulationData**

## class openff.evaluator.storage.data.StoredSimulationData

A representation of data which has been cached from a single previous simulation.

#### **Notes**

The ancillary directory which stores larger information such as trajectories should be of the form:

```
|--- data_object.json
|--- data_directory
|--- coordinate_file_name.pdb
|--- trajectory_file_name.dcd
```

\_\_init\_\_()

# **Methods**

init()	
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
has_ancillary_data()	Returns whether this data object requires an accom-
	panying data directory-like structure.
json([file_path, format])	Creates a JSON representation of this class.
<pre>most_information(stored_data_1, stored_data_2)</pre>	Returns the data object with the lowest statisti-
	cal_inefficiency.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_storage_query()	
	returns The storage query which would
	match this
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

The name of a coordinate file which encodes the
topology information of the system.
The id of the force field parameters used to generate
the data.
The total number of molecules in the system.
A frame of observables collected over the duration of
the simulation.
The phase of the system (e.g.
The server id of the calculation which yielded this
data.
The statistical inefficiency of the collected data.
A description of the composition of the stored sys-
tem.
The state at which the data was collected.
The name of a .dcd trajectory file containing config-
urations generated by the simulation.

# coordinate\_file\_name

The name of a coordinate file which encodes the topology information of the system. The default value of this attribute is not set and must be set by the user..

Type FilePath

# trajectory\_file\_name

The name of a .dcd trajectory file containing configurations generated by the simulation. The default value of this attribute is not set and must be set by the user.

Type FilePath

#### observables

A frame of observables collected over the duration of the simulation. The default value of this attribute is not set and must be set by the user..

Type ObservableFrame

## statistical\_inefficiency

The statistical inefficiency of the collected data. The default value of this attribute is not set and must be set by the user..

Type float

# number\_of\_molecules

The total number of molecules in the system. The default value of this attribute is not set and must be set by the user..

Type int

## classmethod most\_information(stored\_data\_1, stored\_data\_2)

Returns the data object with the lowest *statistical\_inefficiency*.

#### **Parameters**

- **stored\_data\_1** (StoredSimulationData) The first piece of data to compare.
- stored\_data\_2 (StoredSimulationData) The second piece of data to compare.

Return type StoredSimulationData

#### to\_storage\_query()

**Returns** The storage query which would match this data object.

Return type SimulationDataQuery

# force\_field\_id

The id of the force field parameters used to generate the data. The default value of this attribute is not set and must be set by the user..

Type str

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

# classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

Returns True if this class requires an accompanying data directory-like structure.

#### Return type bool

#### json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** Any

# property\_phase

The phase of the system (e.g. liquid, gas). The default value of this attribute is not set and must be set by the user..

Type PropertyPhase

## source\_calculation\_id

The server id of the calculation which yielded this data. The default value of this attribute is not set and must be set by the user..

Type str

#### substance

A description of the composition of the stored system. The default value of this attribute is not set and must be set by the user..

Type Substance

# thermodynamic\_state

The state at which the data was collected. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

#### validate(attribute type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# StoredFreeEnergyData

# ${\bf class}\ {\bf open ff. evaluator. storage. data. {\bf Stored Free Energy Data}$

A representation of data which has been cached from an free energy calculation which computed the free energy difference between a start and end state.

## **Notes**

The ancillary directory which stores larger information such as trajectories should be of the form:

```
|--- data_object.json
|--- data_directory
|--- topology_file_name.pdb
|--- start_state_trajectory.dcd
|--- end_state_trajectory.dcd
```

```
__init__()
```

## **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
has_ancillary_data()	Returns whether this data object requires an accom-
	panying data directory-like structure.
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
<pre>most_information(stored_data_1, stored_data_2)</pre>	A comparison function which will always retain both
	pieces of free energy data.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_storage_query()	
	returns The storage query which would
	match this data object.
validate([attribute_type])	Validate the values of the attributes.

#### **Attributes**

and state trainstory	The name of a .dcd trajectory file containing config-
end_state_trajectory	, , , , , , , , , , , , , , , , , , ,
	urations generated by the simulation of the end state
	of the system.
force_field_id	The id of the force field parameters used to generate
	the data.
free_energy_difference	The free energy difference between the end state and
	the start state.
property_phase	The phase of the system (e.g.
source_calculation_id	The server id of the calculation which yielded this
	data.
start_state_trajectory	The name of a .dcd trajectory file containing config-
	urations generated by the simulation of the start state
	of the system.
substance	A description of the composition of the stored sys-
	tem.
thermodynamic_state	The state at which the data was collected.
topology_file_name	The name of a coordinate file which encodes the
	topology of the system.

# free\_energy\_difference

The free energy difference between the end state and the start state. The default value of this attribute is not set and must be set by the user..

Type Observable

#### topology\_file\_name

The name of a coordinate file which encodes the topology of the system. The default value of this attribute is not set and must be set by the user..

Type FilePath

#### start\_state\_trajectory

The name of a .dcd trajectory file containing configurations generated by the simulation of the start state of the system. The default value of this attribute is not set and must be set by the user..

Type FilePath

# end\_state\_trajectory

The name of a .dcd trajectory file containing configurations generated by the simulation of the end state of the system. The default value of this attribute is not set and must be set by the user..

Type FilePath

# **classmethod most\_information**(*stored\_data\_1:* StoredFreeEnergyData, *stored\_data\_2:* StoredFreeEnergyData) → Optional[*StoredFreeEnergyData*]

A comparison function which will always retain both pieces of free energy data. At this time no situation can be envisaged that the same free energy data from exactly the same calculation will be store.

## **Parameters**

- **stored\_data\_1** The first piece of data to compare.
- **stored\_data\_2** The second piece of data to compare.

#### to\_storage\_query()

Returns The storage query which would match this data object.

Return type FreeEnergyDataQuery

## force\_field\_id

The id of the force field parameters used to generate the data. The default value of this attribute is not set and must be set by the user..

Type str

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## classmethod has\_ancillary\_data()

Returns whether this data object requires an accompanying data directory-like structure.

**Returns** True if this class requires an accompanying data directory-like structure.

Return type bool

## json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters** string\_contents (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** Any

## property\_phase

The phase of the system (e.g. liquid, gas). The default value of this attribute is not set and must be set by the user..

**Type** PropertyPhase

#### source\_calculation\_id

The server id of the calculation which yielded this data. The default value of this attribute is not set and must be set by the user..

Type str

#### substance

A description of the composition of the stored system. The default value of this attribute is not set and must be set by the user..

Type Substance

## thermodynamic\_state

The state at which the data was collected. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

## validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **Data Queries**

BaseDataQuery	A base class for queries which can be made to a Storage-
-	Backend.
SubstanceQuery	A query which focuses on finding data which was col-
	lected for substances with specific traits, e.g which con-
	tains both a solute and solvent, or only a solvent etc.
ForceFieldQuery	A class used to query a StorageBackend for ForceField-
	Data which meet the specified criteria.
BaseSimulationDataQuery	The base class for queries which will retrieve
	BaseSimulationData derived data.
SimulationDataQuery	A class used to query a StorageBackend for
	StoredSimulationData objects which meet the
	specified set of criteria.
FreeEnergyDataQuery	A class used to query a StorageBackend for
	FreeEnergyData objects which meet the specified set
	of criteria.

# **BaseDataQuery**

## class openff.evaluator.storage.query.BaseDataQuery

A base class for queries which can be made to a *StorageBackend*.

\_\_init\_\_()

# **Methods**

init()	
apply(data_object)	Apply this query to a data object.
data_class()	The type of data class that this query can be applied
	to.
<pre>from_data_object(data_object)</pre>	Returns the query which would match this data ob-
	ject.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

# abstract classmethod data\_class()

The type of data class that this query can be applied to.

Return type type of BaseStoredData

#### apply(data\_object)

Apply this query to a data object.

**Parameters data\_object** (BaseStoredData) – The data object to apply the query to.

**Returns** The values of the matched parameters of the data object fully matched this query, otherwise *None*.

**Return type** tuple of Any, optional

## classmethod from\_data\_object(data\_object)

Returns the query which would match this data object.

**Parameters data\_object** (BaseStoredData) – The data object to construct the query for.

**Returns** The query which would match this data object.

Return type cls

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters** file\_path (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

**Return type** Any

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **SubstanceQuery**

# class openff.evaluator.storage.query.SubstanceQuery

A query which focuses on finding data which was collected for substances with specific traits, e.g which contains both a solute and solvent, or only a solvent etc.

\_\_init\_\_()

## **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

Components\_only

Only match pure data which was collected for one of the components in the query substance.

## components\_only

Only match pure data which was collected for one of the components in the query substance. The default value of this attribute is False.

Type bool

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## **ForceFieldQuery**

## class openff.evaluator.storage.query.ForceFieldQuery

A class used to query a StorageBackend for ForceFieldData which meet the specified criteria.

\_\_init\_\_()

#### **Methods**

init()	
ann I w(data object)	Apply this query to a data object.
apply(data_object)	
data_class()	The type of data class that this query can be applied
	to.
<pre>from_data_object(data_object)</pre>	Returns the query which would match this data ob-
	ject.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

#### **Attributes**

	force_field_source	The force field source to query for.
--	--------------------	--------------------------------------

## classmethod data\_class()

The type of data class that this query can be applied to.

Return type type of BaseStoredData

# force\_field\_source

The force field source to query for. The default value of this attribute is not set. This attribute is optional.

Type ForceFieldSource

# apply(data\_object)

Apply this query to a data object.

Parameters data\_object (BaseStoredData) - The data object to apply the query to.

**Returns** The values of the matched parameters of the data object fully matched this query, otherwise *None*.

**Return type** tuple of Any, optional

# classmethod from\_data\_object(data\_object)

Returns the query which would match this data object.

**Parameters data\_object** (BaseStoredData) – The data object to construct the query for.

**Returns** The query which would match this data object.

Return type cls

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **BaseSimulationDataQuery**

## class openff.evaluator.storage.query.BaseSimulationDataQuery

The base class for queries which will retrieve BaseSimulationData derived data.

```
__init__()
```

## **Methods**

init()	
app1y(data_object[, attributes_to_ignore])	Apply this query to a data object.
data_class()	The type of data class that this query can be applied
	to.
<pre>from_data_object(data_object)</pre>	Returns the query which would match this data ob-
	ject.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

force_field_id	The id of the force field parameters which used to generate the data.
property_phase	The phase of the substance (e.g.
source_calculation_id	The server id which should have generated this data.
substance	The substance which the data should have been col-
	lected for.
substance_query	The subset of the <i>substance</i> to query for.
thermodynamic_state	The state at which the data should have been col-
	lected.

## substance

The substance which the data should have been collected for. Data for a subset of this substance can be queried for by using the *substance\_query* attribute The default value of this attribute is not set. This attribute is *optional*.

Type Substance

## substance\_query

The subset of the *substance* to query for. This option can only be used when the *substance* attribute is set. The default value of this attribute is not set. This attribute is *optional*.

Type SubstanceQuery

# thermodynamic\_state

The state at which the data should have been collected. The default value of this attribute is not set. This attribute is *optional*.

Type ThermodynamicState

## property\_phase

The phase of the substance (e.g. liquid, gas). The default value of this attribute is not set. This attribute is *optional*.

Type PropertyPhase

#### source\_calculation\_id

The server id which should have generated this data. The default value of this attribute is not set. This attribute is *optional*.

Type str

## force\_field\_id

The id of the force field parameters which used to generate the data. The default value of this attribute is not set. This attribute is *optional*.

Type str

apply(data\_object, attributes\_to\_ignore=None)

Apply this query to a data object.

**Parameters data\_object** (BaseStoredData) – The data object to apply the query to.

**Returns** The values of the matched parameters of the data object fully matched this query, otherwise *None*.

Return type tuple of Any, optional

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## abstract classmethod data\_class()

The type of data class that this query can be applied to.

Return type type of BaseStoredData

# classmethod from\_data\_object(data\_object)

Returns the query which would match this data object.

Parameters data\_object (BaseStoredData) - The data object to construct the query for.

**Returns** The query which would match this data object.

Return type cls

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

Return type Any

# **SimulationDataQuery**

# class openff.evaluator.storage.query.SimulationDataQuery

A class used to query a StorageBackend for StoredSimulationData objects which meet the specified set of criteria.

\_\_init\_\_()

# **Methods**

init()	
apply(data_object[, attributes_to_ignore])	Apply this query to a data object.
data_class()	The type of data class that this query can be applied
	to.
<pre>from_data_object(data_object)</pre>	Returns the query which would match this data ob-
	ject.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

force_field_id	The id of the force field parameters which used to
	generate the data.
number_of_molecules	The total number of molecules in the system.
property_phase	The phase of the substance (e.g.
source_calculation_id	The server id which should have generated this data.
substance	The substance which the data should have been col-
	lected for.
substance_query	The subset of the <i>substance</i> to query for.
thermodynamic_state	The state at which the data should have been col-
	lected.

## classmethod data\_class()

The type of data class that this query can be applied to.

Return type type of BaseStoredData

## number\_of\_molecules

The total number of molecules in the system. The default value of this attribute is not set. This attribute is *optional*.

Type int

apply(data\_object, attributes\_to\_ignore=None)

Apply this query to a data object.

Parameters data\_object (BaseStoredData) - The data object to apply the query to.

**Returns** The values of the matched parameters of the data object fully matched this query, otherwise *None*.

**Return type** tuple of Any, optional

## force\_field\_id

The id of the force field parameters which used to generate the data. The default value of this attribute is not set. This attribute is *optional*.

Type str

# classmethod from\_data\_object(data\_object)

Returns the query which would match this data object.

**Parameters data\_object** (BaseStoredData) – The data object to construct the query for.

**Returns** The query which would match this data object.

Return type cls

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

Return type Any

# property\_phase

The phase of the substance (e.g. liquid, gas). The default value of this attribute is not set. This attribute is *optional*.

Type PropertyPhase

### source\_calculation\_id

The server id which should have generated this data. The default value of this attribute is not set. This attribute is *optional*.

Type str

### substance

The substance which the data should have been collected for. Data for a subset of this substance can be queried for by using the *substance\_query* attribute The default value of this attribute is not set. This attribute is *optional*.

Type Substance

# substance\_query

The subset of the *substance* to query for. This option can only be used when the *substance* attribute is set. The default value of this attribute is not set. This attribute is *optional*.

Type SubstanceQuery

## thermodynamic\_state

The state at which the data should have been collected. The default value of this attribute is not set. This attribute is *optional*.

Type ThermodynamicState

# validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

# FreeEnergyDataQuery

# class openff.evaluator.storage.query.FreeEnergyDataQuery

A class used to query a StorageBackend for FreeEnergyData objects which meet the specified set of criteria.

\_\_init\_\_()

## **Methods**

init()	
<pre>apply(data_object[, attributes_to_ignore])</pre>	Apply this query to a data object.
data_class()	The type of data class that this query can be applied
	to.
<pre>from_data_object(data_object)</pre>	Returns the query which would match this data ob-
	ject.
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

# **Attributes**

force_field_id	The id of the force field parameters which used to
	generate the data.
property_phase	The phase of the substance (e.g.
source_calculation_id	The server id which should have generated this data.
substance	The substance which the data should have been col-
	lected for.
substance_query	The subset of the <i>substance</i> to query for.
thermodynamic_state	The state at which the data should have been col-
	lected.

# classmethod data\_class()

The type of data class that this query can be applied to.

Return type type of BaseStoredData

```
apply(data_object, attributes_to_ignore=None)
```

Apply this query to a data object.

Parameters data\_object (BaseStoredData) - The data object to apply the query to.

**Returns** The values of the matched parameters of the data object fully matched this query, otherwise *None*.

Return type tuple of Any, optional

### force field id

The id of the force field parameters which used to generate the data. The default value of this attribute is not set. This attribute is *optional*.

Type str

# classmethod from\_data\_object(data\_object)

Returns the query which would match this data object.

Parameters data\_object (BaseStoredData) - The data object to construct the query for.

**Returns** The query which would match this data object.

Return type cls

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

# json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

# property\_phase

The phase of the substance (e.g. liquid, gas). The default value of this attribute is not set. This attribute is *optional*.

Type PropertyPhase

### source\_calculation\_id

The server id which should have generated this data. The default value of this attribute is not set. This attribute is *optional*.

Type str

### substance

The substance which the data should have been collected for. Data for a subset of this substance can be queried for by using the *substance\_query* attribute The default value of this attribute is not set. This attribute is *optional*.

Type Substance

## substance\_query

The subset of the *substance* to query for. This option can only be used when the *substance* attribute is set. The default value of this attribute is not set. This attribute is *optional*.

Type SubstanceQuery

# thermodynamic\_state

The state at which the data should have been collected. The default value of this attribute is not set. This attribute is *optional*.

Type ThermodynamicState

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

# **Attributes**

FilePath	Represents a string file path.
StorageAttribute	A descriptor used to mark attributes of a class as those which store information about a cached piece of data.
QueryAttribute	A descriptor used to add additional metadata to attributes of a storage query.

## **FilePath**

class openff.evaluator.storage.attributes.FilePath

Represents a string file path.

\_\_init\_\_()

# Methods

init()	
capitalize()	Return a capitalized version of the string.
casefold()	Return a version of the string suitable for caseless
cascioia()	comparisons.
center(width[, fillchar])	Return a centered string of length width.
count(sub[, start[, end]])	Return the number of non-overlapping occurrences
count(sub[, start[, chajj)	of substring sub in string S[start:end].
encode([encoding, errors])	Encode the string using the codec registered for encoding.
<pre>endswith(suffix[, start[, end]])</pre>	Return True if S ends with the specified suffix, False otherwise.
expandtabs([tabsize])	Return a copy where all tab characters are expanded using spaces.
find(sub[, start[, end]])	Return the lowest index in S where substring sub is
	found, such that sub is contained within S[start:end].
format(*args, **kwargs)	Return a formatted version of S, using substitutions
Tormat ( ango, " Rwango)	from args and kwargs.
<pre>format_map(mapping)</pre>	Return a formatted version of S, using substitutions
	from mapping.
<pre>index(sub[, start[, end]])</pre>	Return the lowest index in S where substring sub is
	found, such that sub is contained within S[start:end].
isalnum()	Return True if the string is an alpha-numeric string,
15d1ffdm()	False otherwise.
isalpha()	Return True if the string is an alphabetic string, False
15a1piia()	otherwise.
isascii()	Return True if all characters in the string are ASCII,
IBABCII()	False otherwise.
isdecimal()	Return True if the string is a decimal string, False
	otherwise.
isdigit()	Return True if the string is a digit string, False other-
	wise.
isidentifier()	Return True if the string is a valid Python identifier,
151ucirci111ci ()	False otherwise.
islower()	Return True if the string is a lowercase string, False
0	otherwise.
isnumeric()	Return True if the string is a numeric string, False
	otherwise.
isprintable()	Return True if the string is printable, False otherwise.
isspace()	Return True if the string is a whitespace string, False
	otherwise.
istitle()	Return True if the string is a title-cased string, False
	otherwise.
isupper()	Return True if the string is an uppercase string, False
Toubhet ()	otherwise.
join(iterable, /)	Concatenate any number of strings.
1just(width[, fillchar])	Return a left-justified string of length width.
lower()	Return a copy of the string converted to lowercase.
TOWET ()	continues on next page

Table 2 – continued from previous page

lstrip([chars])	Return a copy of the string with leading whitespace removed.
maketrans	Return a translation table usable for str.translate().
partition(sep,/)	Partition the string into three parts using the given separator.
removeprefix(prefix,/)	Return a str with the given prefix string removed if present.
removesuffix(suffix,/)	Return a str with the given suffix string removed if present.
replace(old, new[, count])	Return a copy with all occurrences of substring old replaced by new.
rfind(sub[, start[, end]])	Return the highest index in S where substring sub is found, such that sub is contained within S[start:end].
rindex(sub[, start[, end]])	Return the highest index in S where substring sub is found, such that sub is contained within S[start:end].
rjust(width[, fillchar])	Return a right-justified string of length width.
rpartition(sep,/)	Partition the string into three parts using the given separator.
rsplit([sep, maxsplit])	Return a list of the substrings in the string, using sep as the separator string.
rstrip([chars])	Return a copy of the string with trailing whitespace removed.
split([sep, maxsplit])	Return a list of the substrings in the string, using sep as the separator string.
splitlines([keepends])	Return a list of the lines in the string, breaking at line boundaries.
startswith(prefix[, start[, end]])	Return True if S starts with the specified prefix, False otherwise.
strip([chars])	Return a copy of the string with leading and trailing whitespace removed.
swapcase()	Convert uppercase characters to lowercase and lowercase characters to uppercase.
title()	Return a version of the string where each word is titlecased.
translate(table,/)	Replace each character in the string using the given translation table.
upper()	Return a copy of the string converted to uppercase.
zfill(width,/)	Pad a numeric string with zeros on the left, to fill a field of the given width.

# capitalize()

Return a capitalized version of the string.

More specifically, make the first character have upper case and the rest lower case.

# casefold()

Return a version of the string suitable for caseless comparisons.

# center(width, fillchar=' ',/)

Return a centered string of length width.

Padding is done using the specified fill character (default is a space).

$$count(sub[, start[, end]]) \rightarrow int$$

Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Encode the string using the codec registered for encoding.

encoding The encoding in which to encode the string.

errors The error handling scheme to use for encoding errors. The default is 'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible values are 'ignore', 'replace' and 'xmlcharrefreplace' as well as any other name registered with codecs.register\_error that can handle UnicodeEncodeErrors.

endswith(
$$suffix[, start[, end]]$$
)  $\rightarrow$  bool

Return True if S ends with the specified suffix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. suffix can also be a tuple of strings to try.

## expandtabs(tabsize=8)

Return a copy where all tab characters are expanded using spaces.

If tabsize is not given, a tab size of 8 characters is assumed.

$$find(sub[, start[, end]]) \rightarrow int$$

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

**format**(\*
$$args$$
, \*\* $kwargs$ )  $\rightarrow$  str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

# **format\_map**(mapping) $\rightarrow$ str

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{' and '}').

$$index(sub[, start[, end]]) \rightarrow int$$

Return the lowest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

#### isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

### isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

### isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

### isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

# isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

# isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Call keyword.iskeyword(s) to test whether string s is a reserved identifier, such as "def" or "class".

### islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

### isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

## isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

### isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

# istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

### isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

# join(iterable,/)

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

### ljust(width, fillchar='',/)

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

### lower()

Return a copy of the string converted to lowercase.

### lstrip(chars=None,/)

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

### static maketrans()

Return a translation table usable for str.translate().

If there is only one argument, it must be a dictionary mapping Unicode ordinals (integers) or characters to Unicode ordinals, strings or None. Character keys will be then converted to ordinals. If there are two arguments, they must be strings of equal length, and in the resulting dictionary, each character in x will be mapped to the character at the same position in y. If there is a third argument, it must be a string, whose characters will be mapped to None in the result.

# partition(sep,/)

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

### removeprefix(prefix,/)

Return a str with the given prefix string removed if present.

If the string starts with the prefix string, return string[len(prefix):]. Otherwise, return a copy of the original string.

## removesuffix(suffix,/)

Return a str with the given suffix string removed if present.

If the string ends with the suffix string and that suffix is not empty, return string[:-len(suffix)]. Otherwise, return a copy of the original string.

```
replace(old, new, count=- 1,/)
```

Return a copy with all occurrences of substring old replaced by new.

count Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are replaced.

```
rfind(sub[, start[, end]]) \rightarrow int
```

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Return -1 on failure.

$$rindex(sub[, start[, end]]) \rightarrow int$$

Return the highest index in S where substring sub is found, such that sub is contained within S[start:end]. Optional arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.

```
rjust(width, fillchar='',/)
```

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

# rpartition(sep,/)

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

```
rsplit(sep=None, maxsplit=-1)
```

Return a list of the substrings in the string, using sep as the separator string.

**sep** The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including \n \r \t \f and spaces) and will discard empty strings from the result.

**maxsplit** Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Splitting starts at the end of the string and works to the front.

```
rstrip(chars=None,/)
```

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

```
split(sep=None, maxsplit=- 1)
```

Return a list of the substrings in the string, using sep as the separator string.

sep The separator used to split the string.

When set to None (the default value), will split on any whitespace character (including \n \r \t \f and spaces) and will discard empty strings from the result.

**maxsplit** Maximum number of splits (starting from the left). -1 (the default value) means no limit.

Note, str.split() is mainly useful for data that has been intentionally delimited. With natural text that includes punctuation, consider using the regular expression module.

```
splitlines(keepends=False)
```

Return a list of the lines in the string, breaking at line boundaries.

Line breaks are not included in the resulting list unless keepends is given and true.

```
startswith(prefix[, start[, end]]) \rightarrow bool
```

Return True if S starts with the specified prefix, False otherwise. With optional start, test S beginning at that position. With optional end, stop comparing S at that position. prefix can also be a tuple of strings to try.

```
strip(chars=None,/)
```

Return a copy of the string with leading and trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

# swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

#### title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

### translate(table,/)

Replace each character in the string using the given translation table.

**table** Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via \_\_getitem\_\_, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

### upper()

Return a copy of the string converted to uppercase.

### **zfill**(width,/)

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

# **StorageAttribute**

class openff.evaluator.storage.attributes.StorageAttribute(docstring, type\_hint, optional=False)

A descriptor used to mark attributes of a class as those which store information about a cached piece of data.

```
__init__(docstring, type_hint, optional=False)
```

Initializes a new Attribute object.

#### **Parameters**

- **docstring** (*str*) A docstring describing the attributes purpose. This will automatically be decorated with additional information such as type hints, default values, etc.
- **type\_hint** (*type*, *Union*) The expected type of this attribute. This will be used to help the workflow engine ensure that expected input types match corresponding output values.
- **default\_value** (*Any*) The default value for this attribute.
- **optional** (*bool*) Defines whether this is an optional input of a class. If true, the *default\_value* should be set to *UNDEFINED*.
- **read\_only** (*bool*) Defines whether this attribute is read-only.

## **Methods**

init(docstring, type_hint[, optional])	Initializes a new Attribute object.
--	-------------------------------------

# QueryAttribute

A descriptor used to add additional metadata to attributes of a storage query.

\_\_init\_\_(docstring, type\_hint, optional=False, custom\_match=False)
Initializes self.

**Parameters custom\_match** (*boo1*) – Whether a custom behaviour will be implemented when matching this attribute against the matching data object attribute.

### **Methods**

init(docstring, type_hint[, optional,])	Initializes self.	

# 2.32.9 Workflow API

Workflow	Encapsulates and prepares a workflow which is able to
	estimate a physical property.
WorkflowException	An exception which was raised while executing a work-
	flow protocol.
WorkflowGraph	A hierarchical structure for storing and submitting the
	workflows which will estimate a set of physical proper-
	ties
WorkflowResult	The result of executing a Workflow as part of a Work-
	flow Graph.
Protocol	The base class for a protocol which would form one step
	of a larger property calculation workflow.
ProtocolGraph	A graph of connected protocols which may be executed
	together.
ProtocolGroup	A group of workflow protocols to be executed in one
	batch.
workflow_protocol	A decorator which registers a class as being a protocol
	which may be included in workflows.
register_workflow_protocol	Registers a class as being a protocol which may be in-
	cluded in workflows.

# Workflow

class openff.evaluator.workflow.Workflow(global\_metadata, unique\_id=None)

Encapsulates and prepares a workflow which is able to estimate a physical property.

\_\_init\_\_(global\_metadata, unique\_id=None)

Constructs a new Workflow object.

# **Parameters**

• **global\_metadata** (*dict of str and Any*) – A dictionary of the metadata which will be made available to each of the workflow protocols through the pseudo "global" scope.

• **unique\_id** (*str*, *optional*) – A unique identifier to assign to this workflow. This id will be appended to the ids of the protocols of this workflow. If none is provided, one will be chosen at random.

## **Methods**

init(global_metadata[, unique_id])	Constructs a new Workflow object.
execute([root_directory,])	Executes the workflow.
<pre>from_schema(schema, metadata[, unique_id])</pre>	Creates a workflow from its schema blueprint, and
	the associated metadata.
<pre>generate_default_metadata(physical_property,</pre>	Generates the default global metadata dictionary.
)	
label_molecules(force_field, topology)	
replace_protocol(old_protocol, new_protocol)	Replaces an existing protocol with a new one, while
	updating all input and local references to point to the
	new protocol.
to_graph()	Converts this workflow to an executable Workflow-
	Graph.

#### **Attributes**

final_value_source	The path to the protocol output which corresponds to
	the estimated value of the property being estimated.
outputs_to_store	A collection of data classes to populate ready to be
	stored by a StorageBackend.
protocols	The protocols in this workflow.
schema	

### property protocols

The protocols in this workflow.

Type tuple of Protocol

# property final\_value\_source

The path to the protocol output which corresponds to the estimated value of the property being estimated.

Type ProtocolPath

# property outputs\_to\_store

A collection of data classes to populate ready to be stored by a StorageBackend.

Type dict of str and StorageBackend

## replace\_protocol(old\_protocol, new\_protocol, update\_paths\_only=False)

Replaces an existing protocol with a new one, while updating all input and local references to point to the new protocol.

The main use of this method is when merging multiple protocols into one.

### **Parameters**

• old\_protocol (Protocol or ProtocolPath) – The protocol (or its id) to replace.

- new\_protocol (Protocol or ProtocolPath) The new protocol (or its id) to use.
- **update\_paths\_only** (*bool*) Whether only update the *final\_value\_source*, and *out-puts\_to\_store* attributes, or to also update all of the protocols in *protocols*.

Generates the default global metadata dictionary.

#### **Parameters**

- **physical\_property** (PhysicalProperty) The physical property whose arguments are available in the global scope.
- **force\_field\_path** (*str*) The path to the force field parameters to use in the workflow.
- parameter\_gradient\_keys (list of ParameterGradientKey) A list of references to all of the parameters which all observables should be differentiated with respect to.
- target\_uncertainty (openff.evaluator.unit.Quantity, optional) The uncertainty which the property should be estimated to within.

### **Returns**

The metadata dictionary, with the following keys / types:

- thermodynamic\_state: *ThermodynamicState* The state (T,p) at which the property is being computed
- substance: Substance The composition of the system of interest.
- **components: list of Substance - The components present in the system for** which the property is being estimated.
- target\_uncertainty: openff.evaluator.unit.Quantity The target uncertainty with which properties should be estimated.
- per\_component\_uncertainty: openff.evaluator.unit.Quantity The target uncertainty divided by the sqrt of the number of components in the system + 1
- force\_field\_path: str A path to the force field parameters with which the property should be evaluated with.
- parameter\_gradient\_keys: list of ParameterGradientKey A list of references to all of the parameters which all observables should be differentiated with respect to.

**Return type** dict of str, *Any* 

## to\_graph()

Converts this workflow to an executable WorkflowGraph.

**Returns** The graph representation of this workflow.

**Return type** WorkflowGraph

**classmethod from\_schema**(*schema*, *metadata*, *unique\_id=None*)

Creates a workflow from its schema blueprint, and the associated metadata.

#### **Parameters**

- **schema** (WorkflowSchema) The schema blueprint for this workflow.
- **metadata** (*dict of str and Any*) The metadata to make available to the workflow.

• **unique\_id** (*str*, *optional*) – A unique identifier to assign to this workflow. This id will be appended to the ids of the protocols of this workflow. If none is provided one will be chosen at random.

Returns The created workflow.

Return type cls

execute(root\_directory=", calculation\_backend=None, compute\_resources=None)

Executes the workflow.

#### **Parameters**

- **root\_directory** (*str*) The directory to execute the graph in.
- **calculation\_backend** (CalculationBackend, *optional.*) The backend to execute the graph on. This parameter is mutually exclusive with *compute\_resources*.
- **compute\_resources** (CalculationBackend, optional.) The compute resources to run using. If None and no *calculation\_backend* is specified, the workflow will be executed on a single CPU thread. This parameter is mutually exclusive with *calculation\_backend*.

**Returns** The result of executing this workflow. If executed on a *calculation\_backend*, the result will be wrapped in a *Future* object.

Return type WorkflowResult or Future of WorkflowResult

# WorkflowException

**exception** openff.evaluator.workflow.WorkflowException(message=None, protocol\_id=None)

An exception which was raised while executing a workflow protocol.

```
classmethod from_exception(exception)
```

Initialize this class from an existing exception.

**Parameters exception** (*Exception*) – The existing exception

Returns The initialized exception object.

Return type cls

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

json(file path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters** string\_contents (str or bytes) – The typed json string.

**Returns** The parsed class.

Return type Any

## with\_traceback()

Exception.with\_traceback(tb) - set self.\_\_traceback\_\_ to tb and return self.

# WorkflowGraph

# class openff.evaluator.workflow.WorkflowGraph

A hierarchical structure for storing and submitting the workflows which will estimate a set of physical properties..

\_\_init\_\_()

## **Methods**

ini+

111111()	
add_workflows(*workflows)	Insert a set of workflows into the workflow graph.
execute([root_directory,])	Executes the workflow graph.

## **Attributes**

protocols	The protocols in this graph.
root_protocols	The ids of the protocols in the group which do not
	take input from the other grouped protocols.

# property protocols

The protocols in this graph.

Type dict of str and Protocol

## property root\_protocols

The ids of the protocols in the group which do not take input from the other grouped protocols.

Type list of str

# add\_workflows(\*workflows)

Insert a set of workflows into the workflow graph.

**Parameters workflow** (Workflow) – The workflow to insert.

 $\textbf{execute}(root\_directory=", calculation\_backend=None, compute\_resources=None)$ 

Executes the workflow graph.

### **Parameters**

• **root\_directory** (*str*) – The directory to execute the graph in.

- **calculation\_backend** (CalculationBackend, *optional.*) The backend to execute the graph on. This parameter is mutually exclusive with *compute\_resources*.
- **compute\_resources** (CalculationBackend, optional.) The compute resources to run using. If None and no *calculation\_backend* is specified, the workflow will be executed on a single CPU thread. This parameter is mutually exclusive with *calculation\_backend*.

**Returns** The results of executing the graph. If a *calculation\_backend* is specified, these results will be wrapped in a *Future*.

Return type list of WorkflowResult or list of Future of WorkflowResult

### WorkflowResult

## class openff.evaluator.workflow.WorkflowResult

The result of executing a Workflow as part of a WorkflowGraph.

\_\_init\_\_()

## **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

data_to_store	Paths to the data objects to store.
exceptions	Any exceptions raised by the layer while estimating
	the property.
gradients	The gradients of the estimated value with respect to
	the specified force field parameters.
value	The estimated value of the property and the uncer-
	tainty in that value.
workflow_id	The id of the workflow associated with this result.

# workflow\_id

The id of the workflow associated with this result. The default value of this attribute is not set and must be set by the user..

Type str

#### value

The estimated value of the property and the uncertainty in that value. The default value of this attribute is not set. This attribute is *optional*.

### **Type** Measurement

## gradients

The gradients of the estimated value with respect to the specified force field parameters. The default value of this attribute is [].

```
Type list
```

## exceptions

Any exceptions raised by the layer while estimating the property. The default value of this attribute is [].

```
Type list
```

## data\_to\_store

Paths to the data objects to store. The default value of this attribute is [].

```
Type list
```

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# json(file\_path=None, format=False)

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** Any

## **Protocol**

## class openff.evaluator.workflow.Protocol(protocol\_id)

The base class for a protocol which would form one step of a larger property calculation workflow.

A protocol may for example:

- create the coordinates of a mixed simulation box
- set up a bound ligand-protein system
- build the simulation topology
- · perform an energy minimisation

An individual protocol may require a set of inputs, which may either be set as constants

```
>>> from openff.evaluator.protocols.openmm import OpenMMSimulation
>>>
>>> npt_equilibration = OpenMMSimulation('npt_equilibration')
>>> npt_equilibration.ensemble = OpenMMSimulation.Ensemble.NPT
```

or from the output of another protocol, pointed to by a ProtocolPath

```
>>> npt_production = OpenMMSimulation('npt_production')
>>> # Use the coordinate file output by the npt_equilibration protocol
>>> # as the input to the npt_production protocol
>>> npt_production.input_coordinate_file = ProtocolPath('output_coordinate_file',
>>> npt_equilibration.id)
```

In this way protocols may be chained together, thus defining a larger property calculation workflow from simple, reusable building blocks.

```
__init__(protocol_id)
```

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

allow_merging	Input - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

# property schema

A serializable schema for this object.

Type ProtocolSchema

# property required\_inputs

The inputs which must be set on this protocol.

**Type** list of ProtocolPath

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### Notes

This method is mainly intended to be used only when merging multiple protocols into one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- other(Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

**Return type** dict of ProtocolPath and ProtocolPath

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

# set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

# **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters** string\_contents (str or bytes) – The typed json string.

Returns The parsed class.

Return type Any

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

# **ProtocolGraph**

# class openff.evaluator.workflow.ProtocolGraph

A graph of connected protocols which may be executed together.

\_\_init\_\_()

### **Methods**

init()	
add_protocols(*protocols[,])	Adds a set of protocols to the graph.
execute([root_directory,])	Execute the protocol graph in the specified directory,
	and either using a CalculationBackend, or using a
	specified set of compute resources.

# **Attributes**

protocols	The protocols in this graph.
root_protocols	The ids of the protocols in the group which do not
	take input from the other grouped protocols.

# property protocols

The protocols in this graph.

Type dict of str and Protocol

## property root\_protocols

The ids of the protocols in the group which do not take input from the other grouped protocols.

Type list of str

add\_protocols(\*protocols, allow\_external\_dependencies=False)

Adds a set of protocols to the graph.

#### **Parameters**

- **protocols** (tuple of Protocol) The protocols to add.
- **allow\_external\_dependencies** (*bool*) If *False*, an exception will be raised if a protocol has a dependency outside of this graph.

**Returns** A mapping between the original protocols and protocols which were merged over the course of adding the new protocols.

Return type dict of str and str

**execute**(root\_directory=", calculation\_backend=None, compute\_resources=None, enable\_checkpointing=True, safe\_exceptions=True)

Execute the protocol graph in the specified directory, and either using a *CalculationBackend*, or using a specified set of compute resources.

#### **Parameters**

- **root\_directory** (*str*) The directory to execute the graph in.
- **calculation\_backend** (CalculationBackend, *optional.*) The backend to execute the graph on. This parameter is mutually exclusive with *compute\_resources*.
- **compute\_resources** (CalculationBackend, *optional.*) The compute resources to run using. This parameter is mutually exclusive with *calculation\_backend*.
- **enable\_checkpointing** (*boo1*) If enabled, protocols will not be executed more than once if the output from their previous execution is found.
- **safe\_exceptions** (*bool*) If true, exceptions will be serialized into the results file rather than directly raised, otherwise, the exception will be raised as normal.

**Returns** The paths to the JSON serialized outputs of the executed protocols. If executed using a calculation backend, these will be *Future* objects which will return the output paths on calling *future.result()*.

**Return type** dict of str and str or Future

## **ProtocolGroup**

class openff.evaluator.workflow.ProtocolGroup(protocol\_id)

A group of workflow protocols to be executed in one batch.

This may be used for example to cluster together multiple protocols that will execute in a linear chain so that multiple scheduler execution calls are reduced into a single one.

Additionally, a group may provide enhanced behaviour, for example running all protocols within the group self consistently until a given condition is met (e.g run a simulation until a given observable has converged).

\_\_init\_\_(protocol\_id)

Constructs a new ProtocolGroup.

# **Methods**

init(protocol_id)	Constructs a new ProtocolGroup.
add_protocols(*protocols)	Add protocols to this group.
<pre>apply_replicator(replicator, template_values)</pre>	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Store the uuid of the calculation this protocol belongs
	to
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
protocols	A dictionary of the protocols in this groups, where
	the dictionary key is the protocol id, and the value is
	the protocol itself.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### property protocols

A dictionary of the protocols in this groups, where the dictionary key is the protocol id, and the value is the protocol itself.

### **Notes**

This property should *not* be altered. Use *add\_protocols* to add new protocols to the group.

Type dict of str and Protocol

## add\_protocols(\*protocols)

Add protocols to this group.

**Parameters protocols** (Protocol) – The protocols to add.

## set\_uuid(value)

Store the uuid of the calculation this protocol belongs to

**Parameters value** (str) – The unid of the parent calculation.

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a different one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (*Protocol*) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

Returns A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

## **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters** reference\_path (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

```
execute(directory=", available_resources=None)
```

Execute the protocol.

### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

**Return type** cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

**Return type** Any

# property schema

A serializable schema for this object.

```
Type ProtocolSchema
```

# set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

```
validate(attribute_type=None)
```

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to validate.

Raises ValueError or AssertionError -

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

### workflow protocol

## openff.evaluator.workflow.workflow\_protocol()

A decorator which registers a class as being a protocol which may be included in workflows.

### register workflow protocol

## openff.evaluator.workflow.register\_workflow\_protocol(protocol\_class)

Registers a class as being a protocol which may be included in workflows.

## Schemas

ProtocolSchema	A json serializable representation of a workflow proto-
	col.
ProtocolGroupSchema	A json serializable representation of a workflow protocol
	group.
ProtocolReplicator	A protocol replicator contains the information necessary
	to replicate parts of a property estimation workflow.
WorkflowSchema	The schematic for a property estimation workflow.

# **ProtocolSchema**

A json serializable representation of a workflow protocol.

\_\_init\_\_(unique\_id=None, protocol\_type=None, inputs=None)

## **Methods**

init([unique_id, protocol_type, inputs])	
from_json(file_path)	Create this object from a JSON file.
get_attributes([attribute_type])	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_protocol()	Creates a new protocol object from this schema.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

id	The unique id associated with the protocol.
inputs	The inputs to the protocol.
type	The type of protocol associated with this schema.

# id

The unique id associated with the protocol. The default value of this attribute is not set and must be set by the user.

Type str

### type

The type of protocol associated with this schema. The default value of this attribute is not set and must be set by the user.. This attribute is *read-only*.

Type str

# inputs

The inputs to the protocol. The default value of this attribute is not set and must be set by the user.. This attribute is *read-only*.

Type dict

# to\_protocol()

Creates a new protocol object from this schema.

**Returns** The protocol created from this schema.

Return type Protocol

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### **ProtocolGroupSchema**

**class** openff.evaluator.workflow.schemas.**ProtocolGroupSchema**(unique\_id=None,

protocol\_type=None, inputs=None,
protocol\_schemas=None)

A json serializable representation of a workflow protocol group.

**\_\_init\_\_**(unique id=None, protocol type=None, inputs=None, protocol schemas=None)

## **Methods**

init([unique_id, protocol_type, inputs,])	
from_json(file_path)	Create this object from a JSON file.
get_attributes([attribute_type])	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
to_protocol()	Creates a new protocol object from this schema.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

id	The unique id associated with the protocol.
inputs	The inputs to the protocol.
protocol_schemas	The schemas of the protocols within this group.
type	The type of protocol associated with this schema.

# protocol\_schemas

The schemas of the protocols within this group. The default value of this attribute is not set and must be set by the user.. This attribute is *read-only*.

Type dict

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to validate.

Raises ValueError or AssertionError -

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

id

The unique id associated with the protocol. The default value of this attribute is not set and must be set by the user.

Type str

# inputs

The inputs to the protocol. The default value of this attribute is not set and must be set by the user. This attribute is *read-only*.

```
Type dict
```

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## to\_protocol()

Creates a new protocol object from this schema.

**Returns** The protocol created from this schema.

Return type Protocol

# type

The type of protocol associated with this schema. The default value of this attribute is not set and must be set by the user. This attribute is *read-only*.

Type str

# **ProtocolReplicator**

# class openff.evaluator.workflow.schemas.ProtocolReplicator(replicator\_id=")

A protocol replicator contains the information necessary to replicate parts of a property estimation workflow.

Any protocol whose id includes \$(replicator.id) (where replicator.id is the id of a replicator) will be cloned for each value present in template\_values. Protocols that are being replicated will also have any ReplicatorValue inputs replaced with the actual value taken from template\_values.

When the protocol is replicated, the \$(replicator.id) placeholder in the protocol id will be replaced an integer which corresponds to the index of a value in the template values array.

Any protocols which take input from a replicated protocol will be updated to instead take a list of value, populated by the outputs of the replicated protocols.

### **Notes**

- The *template\_values* property must be a list of either constant values, or *ProtocolPath* objects which take their value from the *global* scope.
- If children of replicated protocols are also flagged as to be replicated, they will only have their ids changed to match the index of the parent protocol, as opposed to being fully replicated.

```
__init__(replicator_id=")
```

Constructs a new ProtocolReplicator object.

**Parameters replicator\_id** (str) – The id of this replicator.

### **Methods**

init([replicator_id])	Constructs a new ProtocolReplicator object.
<pre>apply(protocols[, template_values,])</pre>	Applies this replicator to the provided set of protocols
	and any of their children.
from_json(file_path)	Create this object from a JSON file.
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
update_references(protocols,)	Redirects the input references of protocols to the
	replicated versions.

#### **Attributes**

placeholder_id	The string which protocols to be replicated should in-
	clude in their ids.

# property placeholder\_id

The string which protocols to be replicated should include in their ids.

**apply**(protocols, template\_values=None, template\_index=- 1, template\_value=None)

Applies this replicator to the provided set of protocols and any of their children.

This protocol should be followed by a call to *update\_references* to ensure that all protocols which take their input from a replicated protocol get correctly updated.

### **Parameters**

- **protocols** (*dict* of str and Protocol) The protocols to apply the replicator to.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with *template\_index* and *template\_value* 

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by this replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by this replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

#### Returns

- dict of str and Protocol The replicated protocols.
- dict of ProtocolPath and list of tuple of ProtocolPath and int A dictionary of references
  to all of the protocols which have been replicated, with keys of original protocol ids. Each
  value is comprised of a list of the replicated protocol ids, and their index into the template\_values array.

### update\_references(protocols, replication\_map, template\_values)

Redirects the input references of protocols to the replicated versions.

#### **Parameters**

- **protocols** (*dict of str and Protocol*) The protocols which have had this replicator applied to them.
- replication\_map (dict of ProtocolPath and list of tuple of ProtocolPath and int) A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the template\_values array.
- **template\_values** (*List of Any*) A list of the values which will be inserted into the newly replicated protocols.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

### **WorkflowSchema**

### class openff.evaluator.workflow.schemas.WorkflowSchema

The schematic for a property estimation workflow.

\_\_init\_\_()

### **Methods**

init()	
(C1,(1)	Controlling the Action of ICON Ch
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol_types(protocol_replacements)	Replaces protocols with given types with other pro-
	tocols of specified replacements.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

final_value_source	A reference to which protocol output corresponds to
	the estimated value of the property.
outputs_to_store	A collection of data classes to populate ready to be
	stored by a StorageBackend.
protocol_replicators	A set of replicators which will replicate parts of the
	workflow.
protocol_schemas	The schemas for the protocols which will make up the
	workflow.

# protocol\_schemas

The schemas for the protocols which will make up the workflow. The default value of this attribute is [].

Type list

# protocol\_replicators

A set of replicators which will replicate parts of the workflow. The default value of this attribute is not set. This attribute is *optional*.

Type list

# final\_value\_source

A reference to which protocol output corresponds to the estimated value of the property. The default value of this attribute is not set. This attribute is *optional*.

Type ProtocolPath

## outputs\_to\_store

A collection of data classes to populate ready to be stored by a *StorageBackend*. The default value of this attribute is not set. This attribute is *optional*.

Type dict

### replace\_protocol\_types(protocol\_replacements, protocol\_group\_schema=None)

Replaces protocols with given types with other protocols of specified replacements. This is useful when replacing the default protocols with custom ones, or swapping out base protocols with actual implementations

**Warning:** This method is NOT fully implemented and is likely to fail in all but a few specific cases. This method should be used with extreme caution.

#### **Parameters**

- **protocol\_replacements** (*dict of str and str, optional*) A dictionary with keys of the types of protocols which should be replaced with those protocols named by the values.
- **protocol\_group\_schema** (ProtocolGroupSchema) The protocol group to apply the replacements to. This is mainly used when applying this method recursively.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

**Return type** cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### json(file path=None, format=False)

Creates a JSON representation of this class.

### Parameters

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

**Return type** Any

### **Attributes**

A base class for enums which will describes how at-
tributes should be handled when attempting to merge
similar protocols.
A enum which describes how attributes should be han-
dled when attempting to merge similar protocols.
A enum which describes how attributes which can be
compared with inequalities should be merged.
A descriptor used to mark an attribute of an object as an
input to that object.
A descriptor used to mark an attribute of an as an output
of that object.

# BaseMergeBehavior

class openff.evaluator.workflow.attributes.BaseMergeBehavior(value)

A base class for enums which will describes how attributes should be handled when attempting to merge similar protocols.

\_\_init\_\_()

# MergeBehavior

 ${\bf class} \ {\bf openff.evaluator.workflow.attributes. {\bf MergeBehavior} (\it value) \\$ 

A enum which describes how attributes should be handled when attempting to merge similar protocols.

This enum may take values of

- Exactly Equal: This attribute must be exactly equal between two protocols for them to be able to merge.
- Custom: This attribute will be ignored by the built-in merging code such that user specified behavior can be implemented.

in	•	+	$\cap$

# **Attributes**

ExactlyEqual		
Custom		

## InequalityMergeBehavior

class openff.evaluator.workflow.attributes.InequalityMergeBehavior(value)

A enum which describes how attributes which can be compared with inequalities should be merged.

This enum may take values of

- SmallestValue: When two protocols are merged, the smallest value of this attribute from either protocol is retained.
- LargestValue: When two protocols are merged, the largest value of this attribute from either protocol is retained.

```
__init__()
```

## **Attributes**

```
SmallestValue

LargestValue
```

## **InputAttribute**

A descriptor used to mark an attribute of an object as an input to that object.

An attribute can either be set with a value directly, or it can also be set to a *ProtocolPath* to be set be the workflow manager.

## **Examples**

To mark an attribute as an input:

Initializes a new InputAttribute object.

**Parameters merge\_behavior** (BaseMergeBehavior) – An enum describing how this input should be handled when considering whether to, and actually merging two different objects.

## **Methods**

init(docstring, type_hint, default_value)	Initializes a new InputAttribute object.

## **OutputAttribute**

class openff.evaluator.workflow.attributes.OutputAttribute(docstring, type hint)

A descriptor used to mark an attribute of an as an output of that object. This attribute is expected to be populated by the object itself, rather than be set externally.

# **Examples**

To mark an attribute as an output:

```
>>> from openff.evaluator.attributes import AttributeClass
>>> from openff.evaluator.workflow.attributes import OutputAttribute
>>>
>>> class MyObject(AttributeClass):
>>>
>>> my_output = OutputAttribute(
>>> docstring='An output that will be filled.',
>>> type_hint=float
>>> )
```

\_\_init\_\_(docstring, type\_hint)

Initializes a new OutputAttribute object.

## **Methods**

init(docstring, type_hint) Initializes a new OutputAttribute object.
--

## Placeholder Values

ReplicatorValue	A placeholder value which will be set by a protocol replicator with the specified id.
ProtocolPath	Represents a pointer to the output of another protocol.

# ReplicatorValue

```
class openff.evaluator.workflow.utils.ReplicatorValue(replicator_id=")
```

A placeholder value which will be set by a protocol replicator with the specified id.

```
__init__(replicator_id=")
```

Constructs a new ReplicatorValue object

**Parameters replicator\_id** (*str*) – The id of the replicator which will set this value.

# Methods

init([replicator_id])	Constructs a new ReplicatorValue object

# **ProtocolPath**

class openff.evaluator.workflow.utils.ProtocolPath(property\_name="', \*protocol\_ids)

Represents a pointer to the output of another protocol.

\_\_init\_\_(property\_name=", \*protocol\_ids)

Constructs a new ProtocolPath object.

### **Parameters**

- $property\_name (str)$  The property name referenced by the path.
- **protocol\_ids** (*str*) An args list of protocol ids in the order in which they will appear in the path.

### **Methods**

init([property_name])	Constructs a new ProtocolPath object.
append_uuid(uuid)	Appends a uuid to each of the protocol id's in the path
copy()	Returns a copy of this path.
from_string(existing_path_string)	
pop_next_in_path()	Pops and then returns the leading protocol id from the
	path.
<pre>prepend_protocol_id(id_to_prepend)</pre>	Prepend a new protocol id onto the front of the path.

### **Attributes**

full_path	The full path referenced by this object.
is_global	
last western!	The and must east id of the moth
<pre>path_separator</pre>	The end protocol id of the path.
path_separator	
property_name	The property name pointed to by the path.
property_separator	
nmatacal ida	The ide of the must ends referenced by this chiest
protocol_ids	The ids of the protocols referenced by this object.
protocol_path	The full path referenced by this object excluding the
	property name.
start_protocol	The leading protocol id of the path.

## property property\_name

The property name pointed to by the path.

```
Type str
```

## property protocol\_ids

The ids of the protocols referenced by this object.

Type tuple of str

# property start\_protocol

The leading protocol id of the path.

Type str

## property last\_protocol

The end protocol id of the path.

Type str

## property protocol\_path

The full path referenced by this object excluding the property name.

Type str

# property full\_path

The full path referenced by this object.

Type str

## prepend\_protocol\_id(id\_to\_prepend)

Prepend a new protocol id onto the front of the path.

**Parameters id\_to\_prepend** (*str*) – The protocol id to prepend to the path

# pop\_next\_in\_path()

Pops and then returns the leading protocol id from the path.

Returns The previously leading protocol id.

Return type str

### append\_uuid(uuid)

Appends a uuid to each of the protocol id's in the path

**Parameters unid** (str) – The unid to append.

## replace\_protocol(old\_id, new\_id)

Redirect the input to point at a new protocol.

The main use of this method is when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the protocol to replace.
- **new\_id** (*str*) The id of the new protocol to use.

### copy()

Returns a copy of this path.

# 2.32.10 Built-in Workflow Protocols

# **Analysis**

BaseAverageObservable	An abstract base class for protocols which will calculate
	the average value of an observable and its uncertainty
	via bootstrapping.
AverageObservable	Computes the average value of an observable as well as
	bootstrapped uncertainties for the average.
AverageDielectricConstant	Computes the average value of the dielectric constant
	from a set of dipole moments (M) and volumes (V)
	sampled over the course of a molecular simulation
	such that eps = $1 + ( - ^2) / (3.0 *$
	eps_0 * <v> * kb * T) [1]</v>
AverageFreeEnergies	A protocol which computes the Boltzmann weighted av-
	erage $(G^{\circ} = -RT \times Log[_{n} \exp(-G^{\circ}_{n})])$ of a set of
	free energies which were measured at the same thermo-
	dynamic state.
ComputeDipoleMoments	A protocol which will compute the dipole moment for
	each configuration in a trajectory and for a given param-
	eterized system.
BaseDecorrelateProtocol	An abstract base class for protocols which will subsam-
	ple a set of data, yielding only equilibrated, uncorrelated
	data.
DecorrelateTrajectory	A protocol which will subsample frames from a trajec-
	tory, yielding only uncorrelated frames as determined
	from a provided statistical inefficiency and equilibration
	time.
DecorrelateObservables	A protocol which will subsample a trajectory of observ-
	ables, yielding only uncorrelated entries as determined
	from a provided statistical inefficiency and equilibration
	time.

# BaseAverageObservable

 $\textbf{class} \ open ff. evaluator. protocols. analysis. \textbf{BaseAverageObservable} (\textit{protocol\_id})$ 

An abstract base class for protocols which will calculate the average value of an observable and its uncertainty via bootstrapping.

\_\_init\_\_(protocol\_id)

# Methods

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
bootstrap_iterations	<b>Input</b> - The number of bootstrap iterations to per-
	form.
bootstrap_sample_size	<b>Input</b> - The relative sample size to use for bootstrap-
	ping.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
potential_energies	<b>Input</b> - The potential energies which were evaluated
	at the same configurations and using the same force
	field parameters as the observable to average.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The state at which the observables were com-
	puted.
time_series_statistics	Output - Statistics about the observables from which
	the average was computed. These include the statisti-
	cal inefficiency and the index after which the observ-
	ables have become stationary (i.e.
value	Output - The average value of the observable.

## bootstrap\_iterations

Input - The number of bootstrap iterations to perform. The default value of this attribute is 250.

Type int

### bootstrap\_sample\_size

**Input** - The relative sample size to use for bootstrapping. The default value of this attribute is 1.0.

Type float

### thermodynamic\_state

**Input** - The state at which the observables were computed. This is required to compute ensemble averages of the gradients of the observable with respect to force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type ThermodynamicState

### potential\_energies

**Input** - The potential energies which were evaluated at the same configurations and using the same force field parameters as the observable to average. This is required to compute ensemble averages of the gradients of the observable with respect to force field parameters. The default value of this attribute is not set. This attribute is *optional*.

Type ObservableArray

### value

**Output** - The average value of the observable. The default value of this attribute is not set and must be set by the user..

Type Observable

## time\_series\_statistics

**Output** - Statistics about the observables from which the average was computed. These include the statistical inefficiency and the index after which the observables have become stationary (i.e. equilibrated). The default value of this attribute is not set and must be set by the user..

Type TimeSeriesStatistics

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

# **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

Returns The class attribute.

Return type object

```
get_value(reference_path)
```

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

## Notes

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list | dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

**Return type** dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

# **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

## **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

 $\begin{tabular}{ll} \textbf{Parameters attribute\_type} & (\textit{type of Attribute, optional}) - \textbf{The type of attribute to} \\ & validate. \end{tabular}$ 

Raises ValueError or AssertionError -

# AverageObservable

class openff.evaluator.protocols.analysis.AverageObservable(protocol\_id)

Computes the average value of an observable as well as bootstrapped uncertainties for the average.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
bootstrap_iterations	Input - The number of bootstrap iterations to per-
	form.
bootstrap_sample_size	<b>Input</b> - The relative sample size to use for bootstrap-
	ping.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
divisor	<b>Input</b> - A value to divide the statistic by.
id	The unique id of this protocol.
observable	<b>Input</b> - The file path to the observable which should
	be averaged.
outputs	A dictionary of the outputs of this property.
potential_energies	<b>Input</b> - The potential energies which were evaluated
	at the same configurations and using the same force
	field parameters as the observable to average.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The state at which the observables were com-
	puted.
time_series_statistics	Output - Statistics about the observables from which
	the average was computed. These include the statisti-
	cal inefficiency and the index after which the observ-
	ables have become stationary (i.e.
value	Output - The average value of the observable.

### observable

**Input** - The file path to the observable which should be averaged. The default value of this attribute is not set and must be set by the user..

Type ObservableArray

## divisor

**Input** - A value to divide the statistic by. This is useful if a statistic (such as enthalpy) needs to be normalised by the number of molecules. The default value of this attribute is 1.0.

Type Union

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

## **Parameters**

• replicator (ProtocolReplicator) – The replicator to apply.

• **template\_values** (*list of Any*) – A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

# bootstrap\_iterations

**Input** - The number of bootstrap iterations to perform. The default value of this attribute is 250.

Type int

### bootstrap\_sample\_size

**Input** - The relative sample size to use for bootstrapping. The default value of this attribute is 1.0.

Type float

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type *object* 

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

# get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## potential\_energies

**Input** - The potential energies which were evaluated at the same configurations and using the same force field parameters as the observable to average. This is required to compute ensemble averages of the gradients of the observable with respect to force field parameters. The default value of this attribute is not set. This attribute is *optional*.

```
Type ObservableArray
```

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

# thermodynamic\_state

**Input** - The state at which the observables were computed. This is required to compute ensemble averages of the gradients of the observable with respect to force field parameters. The default value of this attribute is not set. This attribute is *optional*.

**Type** ThermodynamicState

### time\_series\_statistics

**Output** - Statistics about the observables from which the average was computed. These include the statistical inefficiency and the index after which the observables have become stationary (i.e. equilibrated). The default value of this attribute is not set and must be set by the user..

**Type** TimeSeriesStatistics

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## value

**Output** - The average value of the observable. The default value of this attribute is not set and must be set by the user..

Type Observable

# AverageDielectricConstant

# $\textbf{class} \ open ff. evaluator. protocols. analysis. \textbf{AverageDielectricConstant} (\textit{protocol\_id})$

Computes the average value of the dielectric constant from a set of dipole moments (M) and volumes (V) sampled over the course of a molecular simulation such that eps = 1 + (<M^2> - <M>^2) / (3.0 \* eps\_0 \* <V> \* kb \* T) [1]\_.

# References

[1] A. Glattli, X. Daura and W. F. van Gunsteren. Derivation of an improved simple point charge model for liquid water: SPC/A and SPC/L. J. Chem. Phys. 116(22): 9811-9828, 2002

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>set_value(reference_path, value) validate([attribute_type])</pre>	Sets the value of one of this protocols inputs.  Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
bootstrap_iterations	Input - The number of bootstrap iterations to per-
	form.
bootstrap_sample_size	<b>Input</b> - The relative sample size to use for bootstrap-
	ping.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
dipole_moments	<b>Input</b> - The dipole moments of each sampled config-
	uration.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
potential_energies	<b>Input</b> - The potential energies which were evaluated
	at the same configurations and using the same force
	field parameters as the observable to average.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The state at which the observables were com-
	puted.
time_series_statistics	Output - Statistics about the observables from which
	the average was computed. These include the statisti-
	cal inefficiency and the index after which the observ-
	ables have become stationary (i.e.
value	Output - The average value of the observable.
volumes	<b>Input</b> - The volume of each sampled configuration.

# dipole\_moments

**Input** - The dipole moments of each sampled configuration. The default value of this attribute is not set and must be set by the user..

**Type** ObservableArray

## volumes

**Input** - The volume of each sampled configuration. The default value of this attribute is not set and must be set by the user..

Type ObservableArray

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

## **Parameters**

• replicator (ProtocolReplicator) – The replicator to apply.

• **template\_values** (*list of Any*) – A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

# bootstrap\_iterations

**Input** - The number of bootstrap iterations to perform. The default value of this attribute is 250.

Type int

### bootstrap\_sample\_size

**Input** - The relative sample size to use for bootstrapping. The default value of this attribute is 1.0.

Type float

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

**Return type** cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

Returns The class attribute.

Return type *object* 

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

# get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## potential\_energies

**Input** - The potential energies which were evaluated at the same configurations and using the same force field parameters as the observable to average. This is required to compute ensemble averages of the gradients of the observable with respect to force field parameters. The default value of this attribute is not set. This attribute is *optional*.

```
Type ObservableArray
```

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## thermodynamic\_state

**Input** - The state at which the observables were computed. This is required to compute ensemble averages of the gradients of the observable with respect to force field parameters. The default value of this attribute is not set. This attribute is *optional*.

**Type** ThermodynamicState

### time\_series\_statistics

**Output** - Statistics about the observables from which the average was computed. These include the statistical inefficiency and the index after which the observables have become stationary (i.e. equilibrated). The default value of this attribute is not set and must be set by the user..

**Type** TimeSeriesStatistics

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# value

**Output** - The average value of the observable. The default value of this attribute is not set and must be set by the user..

Type Observable

# AverageFreeEnergies

# class openff.evaluator.protocols.analysis.AverageFreeEnergies(protocol\_id)

A protocol which computes the Boltzmann weighted average ( $G^{\circ} = -RT \times Log[\ _{n} \exp(-G^{\circ}_{n}) ]$ ) of a set of free energies which were measured at the same thermodynamic state. Confidence intervals are computed by bootstrapping with replacement.

\_\_init\_\_(protocol\_id)

# **Methods**

apply_replicator       Applies a ProtocolReplicator       to this protocol.         can_merge(other[, path_replacements])       Determines whether this protocol can be merged with another.         execute([directory, available_resources])       Execute the protocol.         from_json(file_path)       Create this object from a JSON file.         from_schema(schema)       Initializes a protocol from it's schema definition.         get_attributes([attribute_type])       Returns all attributes of a specific attribute_type.         get_class_attribute(reference_path)       Returns one of this protocols, or any of its children's, attributes directly (rather than its value).         get_value(reference_path)       Returns the value of one of this protocols inputs / outputs.         get_value_references(input_path)       Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.         json([file_path, format])       Create a JSON representation of this class.         merge(other)       Merges another Protocol with this one.         parse_json(string_contents)       Parses a typed json string into the corresponding class structure.         replace_protocol(old_id, new_id)       Finds each input which came from a given protocol set_uuid(value)         set_value(reference_path, value)       Sets the value of one of this protocols inputs.         validate([attribute_type])       Validate the values of the attributes.<	init(protocol_id)	
can_merge(other[, path_replacements])Determines whether this protocol can be merged with another.execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.	apply_replicator(replicator, template values)	Applies a <i>ProtocolReplicator</i> to this protocol.
another.  execute([directory, available_resources])  from_json(file_path)  Create this object from a JSON file.  from_schema(schema)  Initializes a protocol from it's schema definition.  get_attributes([attribute_type])  Returns all attributes of a specific attribute_type.  get_class_attribute(reference_path)  Returns one of this protocols, or any of its children's, attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / outputs.  get_value_references(input_path)  Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.  merge(other)  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.  replace_protocol(old_id, new_id)  Finds each input which came from a given protocol set_uuid(value)  Prepend a unique identifier to this protocols inputs.		
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.		•
from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.	<pre>execute([directory, available_resources])</pre>	Execute the protocol.
get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.	from_json(file_path)	Create this object from a JSON file.
get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.	from_schema(schema)	Initializes a protocol from it's schema definition.
attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / outputs.  get_value_references(input_path)  Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.  Creates a JSON representation of this class.  merge(other)  Merges another Protocol with this one.  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.  replace_protocol(old_id, new_id)  Finds each input which came from a given protocol set_uuid(value)  Set_value(reference_path, value)  Sets the value of one of this protocols inputs.	<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.	<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
puts.  get_value_references(input_path)  Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.  merge(other)  Merges another Protocol with this one.  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.  replace_protocol(old_id, new_id)  Finds each input which came from a given protocol set_uuid(value)  Prepend a unique identifier to this protocols id.  set_value(reference_path, value)  Sets the value of one of this protocols inputs.		attributes directly (rather than its value).
get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by in- put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocolset_uuid(value)Prepend a unique identifier to this protocols id.set_value(reference_path, value)Sets the value of one of this protocols inputs.	<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
which one of this protocols inputs (specified by input_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.  merge(other)  Merges another Protocol with this one.  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.  replace_protocol(old_id, new_id)  Finds each input which came from a given protocol set_uuid(value)  Prepend a unique identifier to this protocols id.  set_value(reference_path, value)  Sets the value of one of this protocols inputs.		puts.
put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Sets the value of one of this protocols inputs.	<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol set_uuid(value)set_value(reference_path, value)Prepend a unique identifier to this protocols inputs.		which one of this protocols inputs (specified by in-
merge(other)       Merges another Protocol with this one.         parse_json(string_contents)       Parses a typed json string into the corresponding class structure.         replace_protocol(old_id, new_id)       Finds each input which came from a given protocol set_uuid(value)         set_uuid(value)       Prepend a unique identifier to this protocols id.         set_value(reference_path, value)       Sets the value of one of this protocols inputs.		<pre>put_path) takes its value from.</pre>
parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocolset_uuid(value)Prepend a unique identifier to this protocols id.set_value(reference_path, value)Sets the value of one of this protocols inputs.	<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
structure.  replace_protocol(old_id, new_id)  Finds each input which came from a given protocol  set_uuid(value)  Prepend a unique identifier to this protocols id.  set_value(reference_path, value)  Sets the value of one of this protocols inputs.	merge(other)	Merges another Protocol with this one.
replace_protocol(old_id, new_id)Finds each input which came from a given protocolset_uuid(value)Prepend a unique identifier to this protocols id.set_value(reference_path, value)Sets the value of one of this protocols inputs.	parse_json(string_contents)	Parses a typed json string into the corresponding class
set_uuid(value)Prepend a unique identifier to this protocols id.set_value(reference_path, value)Sets the value of one of this protocols inputs.		structure.
set_value(reference_path, value) Sets the value of one of this protocols inputs.	replace_protocol(old_id, new_id)	
	set_uuid(value)	Prepend a unique identifier to this protocols id.
validate([attribute_type])Validate the values of the attributes.	set_value(reference_path, value)	
	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
bootstrap_cycles	<b>Input</b> - The number of bootstrap cycles to perform
	when estimating the uncertainty in the combined free
	energies.
confidence_intervals	Output - The 95% confidence intervals on the aver-
	age free energy.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
result	Output - The sum of the values.
schema	A serializable schema for this object.
thermodynamic_state	Input - The thermodynamic state at which the free
	energies were measured.
values	Input - The values to add together.

### values: List[Observable]

Input - The values to add together. The default value of this attribute is not set and must be set by the user..

Type list

# thermodynamic\_state

**Input** - The thermodynamic state at which the free energies were measured. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

### bootstrap\_cycles

**Input** - The number of bootstrap cycles to perform when estimating the uncertainty in the combined free energies. The default value of this attribute is 2000.

Type int

## result

Output - The sum of the values. The default value of this attribute is not set and must be set by the user...

Type Observable

# confidence\_intervals

**Output** - The 95% confidence intervals on the average free energy. The default value of this attribute is not set and must be set by the user..

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

## **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

```
Parameters string_contents (str or bytes) – The typed json string.
```

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

# ComputeDipoleMoments

**class** openff.evaluator.protocols.analysis.**ComputeDipoleMoments**(protocol id)

A protocol which will compute the dipole moment for each configuration in a trajectory and for a given parameterized system.

\_\_init\_\_(protocol\_id)

# Methods

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_uuid(value) set_value(reference_path, value)	Prepend a unique identifier to this protocols id.  Sets the value of one of this protocols inputs.
	<u> </u>

# **Attributes**

77 .	T 4 D C 1 d d: 4 1 1 1 1 4
allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
dipole_moments	Output - The computed dipole moments.
gradient_parameters	Input - An optional list of parameters to differentiate
	the dipole moments with respect to.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
parameterized_system	<b>Input</b> - The parameterized system which encodes the
	charge on each atom in the system.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
trajectory_path	Input - The file path to the trajectory of configura-
	tions.

# parameterized\_system

**Input** - The parameterized system which encodes the charge on each atom in the system. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

### trajectory\_path

**Input** - The file path to the trajectory of configurations. The default value of this attribute is not set and must be set by the user..

Type str

### gradient\_parameters

**Input** - An optional list of parameters to differentiate the dipole moments with respect to.

Type list

### dipole\_moments

**Output** - The computed dipole moments. The default value of this attribute is not set and must be set by the user

**Type** ObservableArray

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

### can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

```
get_value(reference_path)
```

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

json(file path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

### replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

```
set_value(reference_path, value)
```

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

```
validate(attribute_type=None)
```

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **BaseDecorrelateProtocol**

```
class openff.evaluator.protocols.analysis.BaseDecorrelateProtocol(protocol_id)
```

An abstract base class for protocols which will subsample a set of data, yielding only equilibrated, uncorrelated data.

```
__init__(protocol_id)
```

## **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
time_series_statistics	<b>Input</b> - Statistics about the data to decorrelate.

# time\_series\_statistics:

Union[openff.evaluator.utils.timeseries.TimeSeriesStatistics, List[openff.evaluator.utils.timeseries.TimeSeriesStatistics]]

**Input** - Statistics about the data to decorrelate. This should include the statistical inefficiency and the index after which the observables have become stationary (i.e. equilibrated). If a list of such statistics are provided it will be assumed that multiple time series which have been joined together are being decorrelated and hence will each be decorrelated separately. The default value of this attribute is not set and must be set by the user.

Type Union

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters** reference\_path (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

## property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

#### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

### **DecorrelateTrajectory**

## class openff.evaluator.protocols.analysis.DecorrelateTrajectory(protocol\_id)

A protocol which will subsample frames from a trajectory, yielding only uncorrelated frames as determined from a provided statistical inefficiency and equilibration time.

```
__init__(protocol_id)
```

# **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

# **Attributes**

<b>Input</b> - Defines whether this protocols is allowed to
merge with other protocols.
A list of pointers to the protocols which this protocol
takes input from.
The unique id of this protocol.
<b>Input</b> - The file path to the starting coordinates of a
trajectory.
<b>Input</b> - The file path to the trajectory to subsample.
Output - The file path to the subsampled trajectory.
A dictionary of the outputs of this property.
The inputs which must be set on this protocol.
A serializable schema for this object.
<b>Input</b> - Statistics about the data to decorrelate.

# input\_coordinate\_file

**Input** - The file path to the starting coordinates of a trajectory. The default value of this attribute is not set and must be set by the user..

Type str

#### input\_trajectory\_path

**Input** - The file path to the trajectory to subsample. The default value of this attribute is not set and must be set by the user..

Type str

### output\_trajectory\_path

**Output** - The file path to the subsampled trajectory. The default value of this attribute is not set and must be set by the user..

Type str

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

**Parameters** 

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

 $\label{parameters} \textbf{Parameters} \ \ \textbf{reference\_path} \ (\texttt{ProtocolPath}) - \textbf{The path pointing to the value to return.}$ 

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

Returns A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

#### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

#### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

#### time series statistics:

```
Union[openff.evaluator.utils.timeseries.TimeSeriesStatistics,
List[openff.evaluator.utils.timeseries.TimeSeriesStatistics]]
```

**Input** - Statistics about the data to decorrelate. This should include the statistical inefficiency and the index after which the observables have become stationary (i.e. equilibrated). If a list of such statistics are provided it will be assumed that multiple time series which have been joined together are being decorrelated and hence will each be decorrelated separately. The default value of this attribute is not set and must be set by the user..

Type Union

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **DecorrelateObservables**

# class openff.evaluator.protocols.analysis.DecorrelateObservables(protocol\_id)

A protocol which will subsample a trajectory of observables, yielding only uncorrelated entries as determined from a provided statistical inefficiency and equilibration time.

\_\_init\_\_(protocol\_id)

## **Methods**

init(protocol_id)	
<pre>apply_replicator(replicator, template_values)</pre>	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

allow_merging	Input - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
input_observables	Input - The observables to decorrelate.
output_observables	Output - The decorrelated observables.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
time_series_statistics	<b>Input</b> - Statistics about the data to decorrelate.

#### input\_observables

**Input** - The observables to decorrelate. The default value of this attribute is not set and must be set by the user..

Type Union

### output\_observables

**Output** - The decorrelated observables. The default value of this attribute is not set and must be set by the user..

Type Union

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

**Parameters** 

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

 $\label{parameters} \textbf{Parameters} \ \ \textbf{reference\_path} \ (\texttt{ProtocolPath}) - \textbf{The path pointing to the value to return.}$ 

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

Returns A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

#### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

**Type** list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## time\_series\_statistics:

Union[openff.evaluator.utils.timeseries.TimeSeriesStatistics, List[openff.evaluator.utils.timeseries.TimeSeriesStatistics]]

**Input** - Statistics about the data to decorrelate. This should include the statistical inefficiency and the index after which the observables have become stationary (i.e. equilibrated). If a list of such statistics are provided it will be assumed that multiple time series which have been joined together are being decorrelated and hence will each be decorrelated separately. The default value of this attribute is not set and must be set by the user..

Type Union

#### validate(attribute type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### **Coordinate Generation**

BuildCoordinatesPackmol	Creates a set of 3D coordinates with a specified compo-
	sition using the PACKMOL package.
SolvateExistingStructure	Solvates a set of 3D coordinates with a specified solvent
	using the PACKMOL package.
BuildDockedCoordinates	Creates a set of coordinates for a ligand bound to some
	receptor.

# BuildCoordinatesPackmol

 $\textbf{class} \ open ff. evaluator. protocols. coordinates. \textbf{\textit{BuildCoordinatesPackmol}} (\textit{protocol\_id})$ 

Creates a set of 3D coordinates with a specified composition using the PACKMOL package.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
assigned_residue_names	Output - The residue names which were assigned to
	each of the components.
box_aspect_ratio	<b>Input</b> - The aspect ratio of the simulation box.
coordinate_file_path	<b>Output</b> - The file path to the created PDB coordinate
	file.
count_exact_amount	Input - Whether components present in an exact
	amount (i.e.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
mass_density	<b>Input</b> - The target density of the created system.
max_molecules	Input - The maximum number of molecules to be
	added to the system.
output_number_of_molecules	Output - The number of molecules in the created sys-
	tem.
output_substance	Output - The substance which was built by packmol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
retain_packmol_files	Input - If True, packmol will not delete all of the tem-
	porary files it creates while building the coordinates.
schema	A serializable schema for this object.
substance	<b>Input</b> - The composition of the system to build.
tolerance	Input - The packmol distance tolerance in units com-
	patible with angstroms.
verbose_packmol	Input - If True, packmol will print verbose informa-
	tion to the logger The default value of this attribute is

#### max\_molecules

Input - The maximum number of molecules to be added to the system. The default value of this attribute is 1000.

Type int

### count\_exact\_amount

**Input** - Whether components present in an exact amount (i.e. defined with an ExactAmount) should be considered when apply the maximum number of molecules constraint. This may be set false, for example, when building a separate solvated protein (n = 1) and solvated protein + ligand complex (n = 2) system but wish for both systems to have the same number of solvent molecules. The default value of this attribute is True.

Type bool

## mass\_density

Input - The target density of the created system. The default value of this attribute is 0.95 gram /
milliliter.

Type Quantity

#### box\_aspect\_ratio

**Input** - The aspect ratio of the simulation box. The default value of this attribute is [1.0, 1.0, 1.0].

Type list

#### substance

**Input** - The composition of the system to build. The default value of this attribute is not set and must be set by the user..

Type Substance

#### tolerance

**Input** - The packmol distance tolerance in units compatible with angstroms. The default value of this attribute is 2.0 angstrom.

Type Quantity

### verbose\_packmol

**Input** - If True, packmol will print verbose information to the logger The default value of this attribute is False.

Type bool

### retain\_packmol\_files

**Input** - If True, packmol will not delete all of the temporary files it creates while building the coordinates. The default value of this attribute is False.

Type bool

#### output\_number\_of\_molecules

**Output** - The number of molecules in the created system. This may be less than maximum requested due to rounding of mole fractions The default value of this attribute is not set and must be set by the user..

Type int

## output\_substance

**Output** - The substance which was built by packmol. This may differ from the input substance for system containing two or more components due to rounding of mole fractions. The mole fractions provided by this output should always be used when weighting values by a mole fraction. The default value of this attribute is not set and must be set by the user.

Type Substance

## assigned\_residue\_names

**Output** - The residue names which were assigned to each of the components. Each key corresponds to a component identifier. The default value of this attribute is not set and must be set by the user..

Type dict

## coordinate\_file\_path

**Output** - The file path to the created PDB coordinate file. The default value of this attribute is not set and must be set by the user..

Type str

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

```
apply_replicator(replicator, template_values, template_index=- 1, template_value=None, update input references=False)
```

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

## **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

**execute**(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

#### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

## property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

#### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### SolvateExistingStructure

```
class openff.evaluator.protocols.coordinates.SolvateExistingStructure(protocol_id)
```

Solvates a set of 3D coordinates with a specified solvent using the PACKMOL package.

```
__init__(protocol_id)
```

# Methods

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific attribute_type.
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
assigned_residue_names	<b>Output</b> - The residue names which were assigned to
	each of the components.
box_aspect_ratio	<b>Input</b> - The aspect ratio of the simulation box.
center_solute_in_box	<b>Input</b> - If <i>True</i> , the solute to solvate will be centered
	in the simulation box.
coordinate_file_path	Output - The file path to the created PDB coordinate
	file.
count_exact_amount	Input - Whether components present in an exact
	amount (i.e.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
mass_density	<b>Input</b> - The target density of the created system.
max_molecules	<b>Input</b> - The maximum number of molecules to be
	added to the system.
output_number_of_molecules	Output - The number of molecules in the created sys-
	tem.
output_substance	Output - The substance which was built by packmol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
retain_packmol_files	Input - If True, packmol will not delete all of the tem-
	porary files it creates while building the coordinates.
schema	A serializable schema for this object.
solute_coordinate_file	<b>Input</b> - A file path to the solute to solvate.
substance	<b>Input</b> - The composition of the system to build.
tolerance	Input - The packmol distance tolerance in units com-
	patible with angstroms.
verbose_packmol	Input - If True, packmol will print verbose informa-
•	tion to the logger The default value of this attribute is
	False.

## solute\_coordinate\_file

**Input** - A file path to the solute to solvate. The default value of this attribute is not set and must be set by the user..

Type str

# center\_solute\_in\_box

**Input** - If *True*, the solute to solvate will be centered in the simulation box. The default value of this attribute is True.

Type bool

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is **True**.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

#### assigned\_residue\_names

**Output** - The residue names which were assigned to each of the components. Each key corresponds to a component identifier. The default value of this attribute is not set and must be set by the user.

Type dict

#### box\_aspect\_ratio

**Input** - The aspect ratio of the simulation box. The default value of this attribute is [1.0, 1.0, 1.0].

Type list

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## coordinate\_file\_path

**Output** - The file path to the created PDB coordinate file. The default value of this attribute is not set and must be set by the user..

Type str

#### count\_exact\_amount

**Input** - Whether components present in an exact amount (i.e. defined with an ExactAmount) should be considered when apply the maximum number of molecules constraint. This may be set false, for example, when building a separate solvated protein (n = 1) and solvated protein + ligand complex (n = 2) system but wish for both systems to have the same number of solvent molecules. The default value of this attribute is True.

Type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

Returns The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

#### Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

Returns The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### mass\_density

Input - The target density of the created system. The default value of this attribute is 0.95 gram /
milliliter.

Type Quantity

#### max\_molecules

**Input** - The maximum number of molecules to be added to the system. The default value of this attribute is 1000.

Type int

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## output\_number\_of\_molecules

**Output** - The number of molecules in the created system. This may be less than maximum requested due to rounding of mole fractions The default value of this attribute is not set and must be set by the user..

Type int

## output\_substance

**Output** - The substance which was built by packmol. This may differ from the input substance for system containing two or more components due to rounding of mole fractions. The mole fractions provided by this output should always be used when weighting values by a mole fraction. The default value of this attribute is not set and must be set by the user.

Type Substance

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# retain\_packmol\_files

**Input** - If True, packmol will not delete all of the temporary files it creates while building the coordinates. The default value of this attribute is False.

Type bool

## property schema

A serializable schema for this object.

Type ProtocolSchema

#### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

#### substance

**Input** - The composition of the system to build. The default value of this attribute is not set and must be set by the user..

Type Substance

#### tolerance

**Input** - The packmol distance tolerance in units compatible with angstroms. The default value of this attribute is 2.0 angstrom.

Type Quantity

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### verbose\_packmol

**Input** - If True, packmol will print verbose information to the logger The default value of this attribute is False.

Type bool

## **BuildDockedCoordinates**

class openff.evaluator.protocols.coordinates.BuildDockedCoordinates(protocol\_id)

Creates a set of coordinates for a ligand bound to some receptor.

#### **Notes**

This protocol currently only supports docking with the OpenEye OEDocking framework.

\_\_init\_\_(protocol\_id)

# Methods

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
get_value(reference_path)	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

activate_site_location	<b>Input</b> - Defines the method by which the activate site
	is identified.
allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
docked_complex_coordinate_path	Output - The file path to the docked ligand-receptor
	complex.
docked_ligand_coordinate_path	Output - The file path to the coordinates of the lig-
	and in it's docked pose, aligned with the initial recep-
	tor_coordinate_file.
id	The unique id of this protocol.
ligand_residue_name	Output - The residue name assigned to the docked
	ligand.
ligand_substance	Input - A substance containing only the ligand to
	dock.
number_of_ligand_conformers	<b>Input</b> - The number of conformers to try and dock
	into the receptor structure.
outputs	A dictionary of the outputs of this property.
receptor_coordinate_file	<b>Input</b> - The file path to the MOL2 coordinates of the
	receptor molecule.
receptor_residue_name	Output - The residue name assigned to the receptor.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

## class ActivateSiteLocation(value)

An enum which describes the methods by which a receptors activate site(s) is located.

### ligand\_substance

**Input** - A substance containing only the ligand to dock. The default value of this attribute is not set and must be set by the user..

Type Substance

## number\_of\_ligand\_conformers

**Input** - The number of conformers to try and dock into the receptor structure. The default value of this attribute is 100.

Type int

### receptor\_coordinate\_file

Input - The file path to the MOL2 coordinates of the receptor molecule. The default value of this attribute is not set and must be set by the user..

Type str

### activate\_site\_location

**Input** - Defines the method by which the activate site is identified. The default value of this attribute is ActivateSiteLocation.ReceptorCenterOfMass.

Type ActivateSiteLocation

#### docked\_ligand\_coordinate\_path

**Output** - The file path to the coordinates of the ligand in it's docked pose, aligned with the initial *receptor\_coordinate\_file*. The default value of this attribute is not set and must be set by the user..

Type str

#### docked\_complex\_coordinate\_path

**Output** - The file path to the docked ligand-receptor complex. The default value of this attribute is not set and must be set by the user..

Type str

## ligand\_residue\_name

**Output** - The residue name assigned to the docked ligand. The default value of this attribute is not set and must be set by the user..

Type str

#### receptor\_residue\_name

**Output** - The residue name assigned to the receptor. The default value of this attribute is not set and must be set by the user.

Type str

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

**Type** list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **Force Field Assignment**

BaseBuildSystem	The base class for any protocol whose role is to apply a
	set of force field parameters to a given system.
BuildSmirnoffSystem	Parametrise a set of molecules with a given smirnoff
	force field using the OpenFF toolkit.
BuildLigParGenSystem	Parametrise a set of molecules with the OPLS-AA/M
	force field.
BuildTLeapSystem	Parametrise a set of molecules with an Amber based
	force field.

# BaseBuildSystem

class openff.evaluator.protocols.forcefield.BaseBuildSystem(protocol\_id)

The base class for any protocol whose role is to apply a set of force field parameters to a given system.

\_\_init\_\_(protocol\_id)

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

#### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
coordinate_file_path	<b>Input</b> - The file path to the PDB coordinate file which
	defines the topology of the system to which the force
	field parameters will be assigned.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
force_field_path	<b>Input</b> - The file path to the force field parameters to
	assign to the system.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
parameterized_system	Output - The parameterized system object.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
substance	<b>Input</b> - The composition of the system.

### force\_field\_path

**Input** - The file path to the force field parameters to assign to the system. The default value of this attribute is not set and must be set by the user.

Type str

## coordinate\_file\_path

**Input** - The file path to the PDB coordinate file which defines the topology of the system to which the force field parameters will be assigned. The default value of this attribute is not set and must be set by the user..

Type str

## substance

**Input** - The composition of the system. The default value of this attribute is not set and must be set by the user..

Type Substance

#### parameterized\_system

**Output** - The parameterized system object. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

• replicator (ProtocolReplicator) – The replicator to apply.

• **template\_values** (*list of Any*) – A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

**execute**(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

```
set_uuid(value)
```

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

```
set_value(reference_path, value)
```

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

```
validate(attribute_type=None)
```

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to validate.

Raises ValueError or AssertionError -

## BuildSmirnoffSystem

class openff.evaluator.protocols.forcefield.BuildSmirnoffSystem(protocol\_id)

Parametrise a set of molecules with a given smirnoff force field using the OpenFF toolkit.

\_\_init\_\_(protocol\_id)

## Methods

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
allow_merging	<u>.</u>
	merge with other protocols.
coordinate_file_path	<b>Input</b> - The file path to the PDB coordinate file which
	defines the topology of the system to which the force
	field parameters will be assigned.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
force_field_path	<b>Input</b> - The file path to the force field parameters to
	assign to the system.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
parameterized_system	Output - The parameterized system object.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
substance	<b>Input</b> - The composition of the system.

# allow\_merging

Input - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

## **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## coordinate\_file\_path

**Input** - The file path to the PDB coordinate file which defines the topology of the system to which the force field parameters will be assigned. The default value of this attribute is not set and must be set by the user..

Type str

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### force\_field\_path

**Input** - The file path to the force field parameters to assign to the system. The default value of this attribute is not set and must be set by the user.

```
Type str
```

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** *Any* 

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### parameterized\_system

**Output** - The parameterized system object. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

#### classmethod parse\_json(string contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

Returns The parsed class.

Return type Any

## replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## substance

**Input** - The composition of the system. The default value of this attribute is not set and must be set by the user..

Type Substance

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## BuildLigParGenSystem

class openff.evaluator.protocols.forcefield.BuildLigParGenSystem(protocol\_id)

Parametrise a set of molecules with the OPLS-AA/M force field. using a LigParGen server.

### **Notes**

This protocol is currently a work in progress and as such has limited functionality compared to the more established *BuildSmirnoffSystem* protocol.

#### References

- [1] Potential energy functions for atomic-level simulations of water and organic and biomolecular systems. Jorgensen, W. L.; Tirado-Rives, J. Proc. Nat. Acad. Sci. USA 2005, 102, 6665-6670
- [2] 1.14\*CM1A-LBCC: Localized Bond-Charge Corrected CM1A Charges for Condensed-Phase Simulations. Dodda, L. S.; Vilseck, J. Z.; Tirado-Rives, J.; Jorgensen, W. L. J. Phys. Chem. B, 2017, 121 (15), pp 3864-3870
- [3] LigParGen web server: An automatic OPLS-AA parameter generator for organic ligands. Dodda, L. S.;Cabeza de Vaca, I.; Tirado-Rives, J.; Jorgensen, W. L. Nucleic Acids Research, Volume 45, Issue W1, 3 July 2017, Pages W331-W336

\_\_init\_\_(protocol\_id)

### **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
coordinate_file_path	<b>Input</b> - The file path to the PDB coordinate file which
	defines the topology of the system to which the force
	field parameters will be assigned.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
force_field_path	<b>Input</b> - The file path to the force field parameters to
	assign to the system.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
parameterized_system	Output - The parameterized system object.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
substance	<b>Input</b> - The composition of the system.
water_model	Input - The water model to apply, if any water
	molecules are present.
·	

### class WaterModel(value)

An enum which describes which water model is being used, so that correct charges can be applied.

**Warning:** This is only a temporary addition until full water model support is introduced.

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

## **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## coordinate\_file\_path

**Input** - The file path to the PDB coordinate file which defines the topology of the system to which the force field parameters will be assigned. The default value of this attribute is not set and must be set by the user..

Type str

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

#### force\_field\_path

**Input** - The file path to the force field parameters to assign to the system. The default value of this attribute is not set and must be set by the user.

Type str

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

**Return type** cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

Returns The value of the input / output

**Return type** Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

## json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

• **file\_path** (str, optional) – The (optional) file path to save the JSON file to.

• **format** (*bool*) – Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

Returns A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## parameterized\_system

**Output** - The parameterized system object. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

#### property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

#### substance

**Input** - The composition of the system. The default value of this attribute is not set and must be set by the user..

Type Substance

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### water\_model

**Input** - The water model to apply, if any water molecules are present. The default value of this attribute is WaterModel.TIP3P.

Type WaterModel

### BuildTLeapSystem

class openff.evaluator.protocols.forcefield.BuildTLeapSystem(protocol\_id)

Parametrise a set of molecules with an Amber based force field. using the tleap package.

## **Notes**

- This protocol is currently a work in progress and as such has limited functionality compared to the more established *BuildSmirnoffSystem* protocol.
- This protocol requires the optional *ambertools* >=19.0 dependency to be installed.

\_\_init\_\_(protocol\_id)

# Methods

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

•
work to use to assign par-
PDB coordinate file which
system to which the force
igned.
tocols which this protocol
force field parameters to
col.
of this property.
ed system object.
set on this protocol.
his object.
f the system.
l to apply, if any water

class ChargeBackend(value)

The framework to use to assign partial charges.

### charge\_backend

**Input** - The backend framework to use to assign partial charges.

Type ChargeBackend

### class WaterModel(value)

An enum which describes which water model is being used, so that correct charges can be applied.

**Warning:** This is only a temporary addition until full water model support is introduced.

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

**Return type** dict of ProtocolPath and list of tuple of ProtocolPath and int

#### can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## coordinate\_file\_path

**Input** - The file path to the PDB coordinate file which defines the topology of the system to which the force field parameters will be assigned. The default value of this attribute is not set and must be set by the user..

Type str

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

**execute**(directory=", available resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## force\_field\_path

**Input** - The file path to the force field parameters to assign to the system. The default value of this attribute is not set and must be set by the user.

Type str

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list | dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

#### property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### parameterized\_system

**Output** - The parameterized system object. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

Return type Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- $\mathbf{new\_id}(str)$  The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

## **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

#### substance

**Input** - The composition of the system. The default value of this attribute is not set and must be set by the user..

Type Substance

validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### water\_model

**Input** - The water model to apply, if any water molecules are present. The default value of this attribute is WaterModel.TIP3P.

Type WaterModel

### Gradients

ZeroGradients	Zeros the gradients of an observable with respect to a
	specified set of force field parameters.

## ZeroGradients

class openff.evaluator.protocols.gradients.ZeroGradients(protocol\_id)

Zeros the gradients of an observable with respect to a specified set of force field parameters.

\_\_init\_\_(protocol\_id)

## Methods

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
force_field_path	<b>Input</b> - The path to the force field which contains
	the parameters to differentiate the observable with re-
	spect to.
gradient_parameters	<b>Input</b> - The parameters to zero the gradient with re-
	spect to.
id	The unique id of this protocol.
input_observables	<b>Input</b> - The observable to set the gradients of.
output_observables	Output - The observable with zeroed gradients.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
-	<u> </u>

# input\_observables

**Input** - The observable to set the gradients of. The default value of this attribute is not set and must be set by the user.

Type Union

## force\_field\_path

**Input** - The path to the force field which contains the parameters to differentiate the observable with respect to. This is many used to get the correct units for the parameters. The default value of this attribute is not set and must be set by the user.

Type str

### gradient\_parameters

**Input** - The parameters to zero the gradient with respect to.

Type *list* 

## output\_observables

**Output** - The observable with zeroed gradients. The default value of this attribute is not set and must be set by the user..

Type Union

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

#### classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

## Return type object

```
get_value(reference_path)
```

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

### property schema

A serializable schema for this object.

Type ProtocolSchema

#### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

## **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## Groups

ConditionalGroup	A collection of protocols which are to execute until a
	given condition is met.

# ConditionalGroup

 $\textbf{class} \ \texttt{openff.evaluator.protocols.groups.} \textbf{ConditionalGroup} (\textit{protocol\_id})$ 

A collection of protocols which are to execute until a given condition is met.

\_\_init\_\_(protocol\_id)

Constructs a new ProtocolGroup.

## **Methods**

Constructs a new ProtocolGroup.
Adds a condition to this groups list of conditions if it
not already in the condition list.
Add protocols to this group.
Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another ProtocolGroup with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Store the uuid of the calculation this protocol belongs
to
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
conditions	<b>Input</b> - The conditions which must be satisfied be-
	forethe group will cleanly exit.
current_iteration	Output - The current number of iterations this group
	has performed while attempting to satisfy the speci-
	fied conditions.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
max_iterations	<b>Input</b> - The maximum number of iterations to run for
	to try and satisfy the groups conditions.
outputs	A dictionary of the outputs of this property.
protocols	A dictionary of the protocols in this groups, where
	the dictionary key is the protocol id, and the value is
	the protocol itself.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

#### class Condition

Defines a specific condition which must be met of the form *left\_hand\_value* [TYPE] *right\_hand\_value*, where [TYPE] may be less than or greater than.

## class Type(value)

The available condition types.

### left\_hand\_value

The left-hand value to compare. The default value of this attribute is not set and must be set by the user..

Type Union

## right\_hand\_value

The right-hand value to compare. The default value of this attribute is not set and must be set by the user..

Type Union

## type

The right-hand value to compare. The default value of this attribute is Type.LessThan.

Type Type

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### classmethod parse\_json(string contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### conditions

**Input** - The conditions which must be satisfied beforethe group will cleanly exit. The default value of this attribute is [].

Type list

#### current\_iteration

**Output** - The current number of iterations this group has performed while attempting to satisfy the specified conditions. This value starts from one. The default value of this attribute is not set and must be set by the user..

Type int

#### max\_iterations

**Input** - The maximum number of iterations to run for to try and satisfy the groups conditions. The default value of this attribute is 100.

Type int

### merge(other)

Merges another ProtocolGroup with this one. The id of this protocol will remain unchanged.

It is assumed that can\_merge has already returned that these protocol groups are compatible to be merged together.

**Parameters other** (ConditionalGroup) – The protocol to merge into this one.

# add\_condition(condition\_to\_add)

Adds a condition to this groups list of conditions if it not already in the condition list.

Parameters condition\_to\_add (Conditional Group. Condition) - The condition to add.

#### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

## add\_protocols(\*protocols)

Add protocols to this group.

**Parameters protocols** (Protocol) – The protocols to add.

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

#### can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

## **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

#### classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

```
get_value(reference_path)
```

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

#### property protocols

A dictionary of the protocols in this groups, where the dictionary key is the protocol id, and the value is the protocol itself.

## **Notes**

This property should *not* be altered. Use *add\_protocols* to add new protocols to the group.

Type dict of str and Protocol

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a different one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

### property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Store the uuid of the calculation this protocol belongs to

**Parameters value** (*str*) – The uuid of the parent calculation.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## Miscellaneous

AddValues	A protocol to add together a list of values.
SubtractValues	A protocol to subtract one value from another such that:
MultiplyValue	A protocol which multiplies a value by a specified scalar
DivideValue	A protocol which divides a value by a specified scalar
WeightByMoleFraction	Multiplies a value by the mole fraction of a component
	in a Substance.
FilterSubstanceByRole	A protocol which takes a substance as input, and returns
	a substance which only contains components whose role
	match a given criteria.
DummyProtocol	A protocol whose only purpose is to return an input value
	as an output value.

#### **AddValues**

 $\textbf{class} \ \ \textbf{open} \textbf{ff.evaluator.protocols.miscellaneous.} \textbf{AddValues} (\textit{protocol\_id})$ 

A protocol to add together a list of values.

## **Notes**

The *values* input must either be a list of openff.evaluator.unit.Quantity, a ProtocolPath to a list of openff.evaluator.unit.Quantity, or a list of ProtocolPath which each point to a openff.evaluator.unit.Quantity.

\_\_init\_\_(protocol\_id)

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
get_value(reference_path)	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols which one of this protocols inputs (specified by <i>in-put_path</i> ) takes its value from.
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
result	Output - The sum of the values.
schema	A serializable schema for this object.
values	Input - The values to add together.

## values

Input - The values to add together. The default value of this attribute is not set and must be set by the user..

Type list

#### result

Output - The sum of the values. The default value of this attribute is not set and must be set by the user...

Type Union

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with *template\_index* and *template\_value* 

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

## Parameters

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

Returns The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### **SubtractValues**

class openff.evaluator.protocols.miscellaneous.SubtractValues(protocol\_id)

A protocol to subtract one value from another such that:

```
result = value_b - value_a
__init__(protocol_id)
```

## **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
result	Output - The results of value_b - value_a.
schema	A serializable schema for this object.
value_a	<b>Input</b> - $value\_a$ in the formula $result = value\_b$ -
	value_a.
value_b	<b>Input</b> - $value_b$ in the formula $result = value_b$ -
	value_a.

## value\_a

**Input** -  $value_a$  in the formula  $result = value_b - value_a$ . The default value of this attribute is not set and must be set by the user..

Type Union

#### value\_b

**Input** -  $value_b$  in the formula  $result = value_b - value_a$ . The default value of this attribute is not set and must be set by the user..

**Type** Union

#### result

**Output** - The results of *value\_b* - *value\_a*. The default value of this attribute is not set and must be set by the user..

Type Union

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template index and template value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

**Parameters** 

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

 $\label{parameters} \textbf{Parameters} \ \ \textbf{reference\_path} \ (\texttt{ProtocolPath}) - \textbf{The path pointing to the value to return.}$ 

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

```
validate(attribute_type=None)
```

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **MultiplyValue**

```
class openff.evaluator.protocols.miscellaneous.MultiplyValue(protocol_id)
```

A protocol which multiplies a value by a specified scalar

```
__init__(protocol_id)
```

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
multiplier	Input - The scalar to multiply by.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
result	Output - The result of the multiplication.
schema	A serializable schema for this object.
value	Input - The value to multiply.

## value

Input - The value to multiply. The default value of this attribute is not set and must be set by the user..

Type Union

## multiplier

Input - The scalar to multiply by. The default value of this attribute is not set and must be set by the user..

Type Union

#### result

**Output** - The result of the multiplication. The default value of this attribute is not set and must be set by the user..

Type Union

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

Returns The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### **DivideValue**

class openff.evaluator.protocols.miscellaneous.DivideValue(protocol\_id)

A protocol which divides a value by a specified scalar

```
__init__(protocol_id)
```

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
divisor	Input - The scalar to divide by.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
result	Output - The result of the division.
schema	A serializable schema for this object.
value	<b>Input</b> - The value to divide.

## value

Input - The value to divide. The default value of this attribute is not set and must be set by the user..

Type Union

### divisor

Input - The scalar to divide by. The default value of this attribute is not set and must be set by the user..

Type Union

#### result

**Output** - The result of the division. The default value of this attribute is not set and must be set by the user..

Type Union

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

Returns The initialized protocol.

Return type cls

### classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

#### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### WeightByMoleFraction

```
class openff.evaluator.protocols.miscellaneous.WeightByMoleFraction(protocol_id)
```

Multiplies a value by the mole fraction of a component in a Substance.

```
__init__(protocol_id)
```

## **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
allow_merging	
	merge with other protocols.
component	<b>Input</b> - The component whose mole fraction to
	weight by.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
full_substance	<b>Input</b> - The full substance which describes the mole
	fraction of the component.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
value	Input - The value to be weighted.
weighted_value	Output - The value weighted by the <i>component's</i>
	mole fraction as determined from the `full_substance.

# value

Input - The value to be weighted. The default value of this attribute is not set and must be set by the user..

Type Union

#### component

**Input** - The component whose mole fraction to weight by. The default value of this attribute is not set and must be set by the user..

Type Substance

### full\_substance

**Input** - The full substance which describes the mole fraction of the component. The default value of this attribute is not set and must be set by the user..

Type Substance

## weighted\_value

**Output** - The value weighted by the *component's mole fraction as determined from the 'full\_substance*. The default value of this attribute is not set and must be set by the user.

Type Union

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

**Return type** dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

#### classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

### Return type object

```
get_value(reference_path)
```

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) - The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** Dict[str, str]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

### property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

### **FilterSubstanceByRole**

### class openff.evaluator.protocols.miscellaneous.FilterSubstanceByRole(protocol\_id)

A protocol which takes a substance as input, and returns a substance which only contains components whose role match a given criteria.

```
__init__(protocol id)
```

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
<pre>set_value(reference_path, value)</pre>	Sets the value of one of this protocols inputs.
<pre>set_value(reference_path, value) validate([attribute_type])</pre>	Sets the value of one of this protocols inputs.  Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
component_roles	<b>Input</b> - The roles to filter substance components
	against.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
expected_components	<b>Input</b> - The number of components expected to re-
	main after filtering.
filtered_substance	Output - The filtered substance.
id	The unique id of this protocol.
input_substance	Input - The substance to filter.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

## input\_substance

Input - The substance to filter. The default value of this attribute is not set and must be set by the user..

Type Substance

#### component\_roles

**Input** - The roles to filter substance components against. The default value of this attribute is not set and must be set by the user..

Type *list* 

### expected\_components

**Input** - The number of components expected to remain after filtering. An exception is raised if this number is not matched. The default value of this attribute is not set. This attribute is *optional*.

Type int

#### filtered\_substance

Output - The filtered substance. The default value of this attribute is not set and must be set by the user..

Type Substance

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

## can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

**Return type** cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

```
get_class_attribute(reference_path)
```

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

**Type** list of ProtocolPath

### property schema

A serializable schema for this object.

```
Type ProtocolSchema
```

```
set_uuid(value)
```

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

```
set_value(reference_path, value)
```

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

### **DummyProtocol**

```
class openff.evaluator.protocols.miscellaneous.DummyProtocol(protocol_id)
```

A protocol whose only purpose is to return an input value as an output value.

```
__init__(protocol_id)
```

## **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
input_value	Input - A dummy input.
output_value	Output - A dummy output.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

## input\_value

Input - A dummy input. The default value of this attribute is not set and must be set by the user..

Type Union

## output\_value

Output - A dummy output. The default value of this attribute is not set and must be set by the user..

Type Union

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

## property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

```
Parameters string_contents (str or bytes) – The typed json string.
```

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## **OpenMM**

OpenMMEnergyMinimisation	A protocol to minimise the potential energy of a system
	using OpenMM.
OpenMMSimulation	Performs a molecular dynamics simulation in a given en-
	semble using an OpenMM backend.
OpenMMEvaluateEnergies	Re-evaluates the energy of a series of configurations for
	a given set of force field parameters using OpenMM.

# **OpenMMEnergyMinimisation**

 $\textbf{class} \ open \texttt{ff.evaluator.protocols.open \texttt{mm.Open MMEnergy Minimisation}} (\textit{protocol\_id})$ 

A protocol to minimise the potential energy of a system using OpenMM.

\_\_init\_\_(protocol\_id)

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

<b>Input</b> - Defines whether this protocols is allowed to
merge with other protocols.
A list of pointers to the protocols which this protocol
takes input from.
<b>Input</b> - If true, periodic boundary conditions will be
enabled.
The unique id of this protocol.
Input - The coordinates to minimise.
<b>Input</b> - The maximum number of iterations to per-
form.
<b>Output</b> - The file path to the minimised coordinates.
A dictionary of the outputs of this property.
Input - The parameterized system object which en-
codes the systems potential energy function.
The inputs which must be set on this protocol.
A serializable schema for this object.
Input - The energy tolerance to which the system
should be minimized.

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

- **replicator** (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• **update\_input\_references** (*bool*) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually

replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

**Return type** dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

## enable\_pbc

Input - If true, periodic boundary conditions will be enabled. The default value of this attribute is True.

Type bool

execute(directory=", available\_resources=None)

Execute the protocol.

### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters** reference\_path (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

### input\_coordinate\_file

**Input** - The coordinates to minimise. The default value of this attribute is not set and must be set by the user.

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

### Return type str

### max\_iterations

**Input** - The maximum number of iterations to perform. If this is 0, minimization is continued until the results converge without regard to how many iterations it takes. The default value of this attribute is **0**.

```
Type int
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## output\_coordinate\_file

**Output** - The file path to the minimised coordinates. The default value of this attribute is not set and must be set by the user..

Type str

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## parameterized\_system

**Input** - The parameterized system object which encodes the systems potential energy function. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

Returns The parsed class.

Return type Any

## replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

### property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

### tolerance

**Input** - The energy tolerance to which the system should be minimized. The default value of this attribute is 10.0 kilojoule / mole.

Type Quantity

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **OpenMMSimulation**

class openff.evaluator.protocols.openmm.OpenMMSimulation(protocol\_id)

Performs a molecular dynamics simulation in a given ensemble using an OpenMM backend.

This protocol employs the Langevin integrator implemented in the openmmtools package to propagate the state of the system using the default BAOAB splitting [1]. Further, simulations which are run in the NPT simulation will have a Monte Carlo barostat (openmm.MonteCarloBarostat) applied every 25 steps (the OpenMM default).

### References

[1] Leimkuhler, Ben, and Charles Matthews. "Numerical methods for stochastic molecular dynamics." Molecular Dynamics. Springer, Cham, 2015. 261-328.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_gpu_platforms	<b>Input</b> - If true, the simulation will be performed
arron_gpa_pracrorms	using a GPU if available, otherwise it will be con-
	strained to only using CPUs.
allow_merging	Input - Defines whether this protocols is allowed to
u==0.1u= y=y	merge with other protocols.
checkpoint_frequency	Input - The frequency (in multiples of out-
	<pre>put_frequency) with which to write to a checkpoint</pre>
	file, e.g.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
enable_pbc	<b>Input</b> - If true, periodic boundary conditions will be
	enabled.
ensemble	<b>Input</b> - The thermodynamic ensemble to simulate in.
gradient_parameters	<b>Input</b> - An optional list of parameters to differentiate
	the evaluated energies with respect to.
high_precision	Input - If true, the simulation will be run using dou-
	ble precision.
id	The unique id of this protocol.
input_coordinate_file	<b>Input</b> - The file path to the starting coordinates.
observables	Output - The observables collected during the sim-
	ulation.
output_coordinate_file	Output - The file path to the coordinates of the final
	system configuration.
output_frequency	Input - The frequency (in number of steps) with
	which to write to the output statistics and trajectory
	files.
outputs	A dictionary of the outputs of this property.
parameterized_system	Input - The parameterized system object which en-
	codes the systems potential energy function.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
steps_per_iteration	<b>Input</b> - The number of steps to propogate the system
	by at each iteration.
thermodynamic_state	<b>Input</b> - The thermodynamic conditions to simulate
	under The default value of this attribute is not set and
	must be set by the user
thermostat_friction	Input - The thermostat friction coefficient.
timestep	<b>Input</b> - The timestep to evolve the system by at each
	step.
total_number_of_iterations	<b>Input</b> - The number of times to propogate the system
	forward by the <i>steps_per_iteration</i> number of steps.
trajectory_file_path	Output - The file path to the trajectory sampled dur-
	ing the simulation.

# allow\_gpu\_platforms

**Input** - If true, the simulation will be performed using a GPU if available, otherwise it will be constrained to only using CPUs. The default value of this attribute is True.

Type bool

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## checkpoint\_frequency

**Input** - The frequency (in multiples of *output\_frequency*) with which to write to a checkpoint file, e.g. if *output\_frequency=100* and *checkpoint\_frequency==2*, a checkpoint file would be saved every 200 steps. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 10. This attribute is *optional*.

Type int

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

## enable\_pbc

Input - If true, periodic boundary conditions will be enabled. The default value of this attribute is True.

Type bool

#### ensemble

Input - The thermodynamic ensemble to simulate in. The default value of this attribute is Ensemble.NPT.

**Type** Ensemble

execute(directory=", available\_resources=None)

Execute the protocol.

### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters** reference\_path (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

## gradient\_parameters

**Input** - An optional list of parameters to differentiate the evaluated energies with respect to.

Type *list* 

### high\_precision

**Input** - If true, the simulation will be run using double precision. The default value of this attribute is False.

Type bool

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

## input\_coordinate\_file

**Input** - The file path to the starting coordinates. The default value of this attribute is not set and must be set by the user..

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

#### observables

**Output** - The observables collected during the simulation. The default value of this attribute is not set and must be set by the user..

Type ObservableFrame

## output\_coordinate\_file

**Output** - The file path to the coordinates of the final system configuration. The default value of this attribute is not set and must be set by the user..

Type str

### output\_frequency

**Input** - The frequency (in number of steps) with which to write to the output statistics and trajectory files. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 3000.

Type int

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## parameterized\_system

**Input** - The parameterized system object which encodes the systems potential energy function. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (str) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

## steps\_per\_iteration

**Input** - The number of steps to propogate the system by at each iteration. The total number of steps performed by this protocol will be *total\_number\_of\_iterations* \* *steps\_per\_iteration*. The default value of this attribute is 1000000.

Type int

## thermodynamic\_state

**Input** - The thermodynamic conditions to simulate under The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

# thermostat\_friction

**Input** - The thermostat friction coefficient. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 1.0 / picosecond.

Type Quantity

## timestep

**Input** - The timestep to evolve the system by at each step. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 2.0 femtosecond.

Type Quantity

## total\_number\_of\_iterations

**Input** - The number of times to propogate the system forward by the *steps\_per\_iteration* number of steps. The total number of steps performed by this protocol will be *total\_number\_of\_iterations* \* *steps\_per\_iteration*. The default value of this attribute is 1.

Type int

## trajectory\_file\_path

**Output** - The file path to the trajectory sampled during the simulation. The default value of this attribute is not set and must be set by the user..

Type str

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

# **OpenMMEvaluateEnergies**

# ${\bf class} \ open {\bf ff.evaluator.protocols.open {\bf mm.Open MMEvaluate Energies} ({\it protocol\_id})}$

Re-evaluates the energy of a series of configurations for a given set of force field parameters using OpenMM.

\_\_init\_\_(protocol\_id)

## **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
1 1

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
enable_pbc	<b>Input</b> - If true, periodic boundary conditions will be
	enabled.
gradient_parameters	Input - An optional list of parameters to differentiate
	the evaluated energies with respect to.
id	The unique id of this protocol.
output_observables	Output - An observable array which stores the re-
	duced potentials potential energies evaluated at the
	specified state and using the specified system object
	for each configuration in the trajectory.
outputs	A dictionary of the outputs of this property.
parameterized_system	Input - The parameterized system object which en-
	codes the systems potential energy function.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The state to calculate the reduced potentials
	at.
trajectory_file_path	<b>Input</b> - The path to the trajectory file which contains
	the configurations to calculate the energies of.
The state of the s	

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with *template\_index* and *template\_value* 

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- other (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

### enable\_pbc

Input - If true, periodic boundary conditions will be enabled. The default value of this attribute is True.

Type bool

**execute**(directory=", available\_resources=None)

Execute the protocol.

### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

## Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

Returns The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

## **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

Returns A dictionary of the protocol paths that the input targeted by input\_path depends upon.

Return type dict of ProtocolPath and ProtocolPath

#### gradient\_parameters

**Input** - An optional list of parameters to differentiate the evaluated energies with respect to.

Type list

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

## json(file\_path=None, format=False)

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

Returns A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

# output\_observables

**Output** - An observable array which stores the reduced potentials potential energies evaluated at the specified state and using the specified system object for each configuration in the trajectory. The default value of this attribute is not set and must be set by the user..

Type ObservableFrame

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

### parameterized\_system

**Input** - The parameterized system object which encodes the systems potential energy function. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

Return type Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## thermodynamic\_state

**Input** - The state to calculate the reduced potentials at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

## trajectory\_file\_path

**Input** - The path to the trajectory file which contains the configurations to calculate the energies of. The default value of this attribute is not set and must be set by the user..

Type str

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **Paprika**

PreparePullCoordinates	A protocol which will align a host-guest complex to the z-axis and position the guest molecule at a specified point along the pull axis.
PrepareReleaseCoordinates	A protocol which will extract the host molecule from a file containing both the host and guest molecules and produce a coordinate file containing only the host which has been correctly aligned to the z-axis.
AddDummyAtoms	A protocol which will add the reference 'dummy' atoms to a parameterised system.

# **PreparePullCoordinates**

 $\textbf{class} \ \ \textbf{openff.evaluator.protocols.paprika.coordinates.} \\ \textbf{PreparePullCoordinates}(\textit{protocol\_id})$ 

A protocol which will align a host-guest complex to the z-axis and position the guest molecule at a specified point along the pull axis.

\_\_init\_\_(protocol\_id)

## **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
complex_file_path	<b>Input</b> - The path to the file which the coordinates of
	the guest moleculebound to the host molecule.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
guest_orientation_mask	Input - The string mask which describes which guest
	atoms will be restrained relative to the dummy atoms
	to keep the molecule aligned to the z-axis.
id	The unique id of this protocol.
n_pull_windows	<b>Input</b> - The total number of the pull windows in the
	calculation.
output_coordinate_path	Output - The file path to the system which has been
	correctly aligned to the z-axis.
outputs	A dictionary of the outputs of this property.
pull_distance	<b>Input</b> - The total distance that the guest will be pulled
	along the z-axis during the pull phase.
pull_window_index	<b>Input</b> - The index of the pull window to generate co-
	ordinates for.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
substance	<b>Input</b> - The substance which defines the host, guest
	and solvent.

# guest\_orientation\_mask

**Input** - The string mask which describes which guest atoms will be restrained relative to the dummy atoms to keep the molecule aligned to the z-axis. This should be of the form 'X Y' where X Y are ParmEd selectors for the first and second guest atoms. The default value of this attribute is not set and must be set by the user..

Type str

# pull\_distance

**Input** - The total distance that the guest will be pulled along the z-axis during the pull phase. The default value of this attribute is not set and must be set by the user..

Type Quantity

## pull\_window\_index

**Input** - The index of the pull window to generate coordinates for. The default value of this attribute is not set and must be set by the user..

Type int

## n\_pull\_windows

**Input** - The total number of the pull windows in the calculation. The default value of this attribute is not set and must be set by the user..

Type int

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

### complex\_file\_path

**Input** - The path to the file which the coordinates of the guest moleculebound to the host molecule. The default value of this attribute is not set and must be set by the user..

Type str

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

Returns The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** Dict[str, str]

### output\_coordinate\_path

**Output** - The file path to the system which has been correctly aligned to the z-axis. The default value of this attribute is not set and must be set by the user.

```
Type str
```

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

### substance

**Input** - The substance which defines the host, guest and solvent. The default value of this attribute is not set and must be set by the user..

```
Type Substance
```

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## **PrepareReleaseCoordinates**

class openff.evaluator.protocols.paprika.coordinates.PrepareReleaseCoordinates(protocol\_id)

A protocol which will extract the host molecule from a file containing both the host and guest molecules and produce a coordinate file containing only the host which has been correctly aligned to the z-axis.

```
__init__(protocol_id)
```

# **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
complex_file_path	<b>Input</b> - The path to the file which the coordinates of
	the guest moleculebound to the host molecule.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
output_coordinate_path	Output - The file path to the system which has been
	correctly aligned to the z-axis.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
substance	<b>Input</b> - The substance which defines the host, guest
	and solvent.
· · · · · · · · · · · · · · · · · · ·	

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is **True**.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

## **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## complex\_file\_path

**Input** - The path to the file which the coordinates of the guest moleculebound to the host molecule. The default value of this attribute is not set and must be set by the user..

Type str

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** *Any* 

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### output\_coordinate\_path

**Output** - The file path to the system which has been correctly aligned to the z-axis. The default value of this attribute is not set and must be set by the user.

```
Type str
```

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

### substance

**Input** - The substance which defines the host, guest and solvent. The default value of this attribute is not set and must be set by the user..

```
Type Substance
```

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## **AddDummyAtoms**

# class openff.evaluator.protocols.paprika.coordinates.AddDummyAtoms(protocol\_id)

A protocol which will add the reference 'dummy' atoms to a parameterised system. This protocol assumes the host / complex has already been correctly aligned to the z-axis and has been placed at the origin.

```
__init__(protocol_id)
```

# **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

# **Attributes**

allow_merging	Input - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
input_coordinate_path	<b>Input</b> - The file path to the coordinates which the
	dummy atoms should be added to.
input_system	<b>Input</b> - The parameterized system which the dummy
	atoms should be added to.
offset	<b>Input</b> - The distance to offset the dummy atoms from
	the origin $(0, 0, 0)$ backwards along the z-axis.
output_coordinate_path	Output - The file path to the coordinates which in-
	clude the added dummy atoms.
output_system	Output - The parameterized system which include
	the added dummy atoms.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
substance	<b>Input</b> - The substance which defines the host, guest
	and solvent.

#### substance

**Input** - The substance which defines the host, guest and solvent. The default value of this attribute is not set and must be set by the user..

Type Substance

### offset

**Input** - The distance to offset the dummy atoms from the origin (0, 0, 0) backwards along the z-axis. The default value of this attribute is not set and must be set by the user..

Type Quantity

# input\_coordinate\_path

**Input** - The file path to the coordinates which the dummy atoms should be added to. The default value of this attribute is not set and must be set by the user..

Type str

## input\_system

**Input** - The parameterized system which the dummy atoms should be added to. The default value of this attribute is not set and must be set by the user.

Type ParameterizedSystem

### output\_coordinate\_path

**Output** - The file path to the coordinates which include the added dummy atoms. The default value of this attribute is not set and must be set by the user..

Type str

#### output\_system

**Output** - The parameterized system which include the added dummy atoms. The default value of this attribute is not set and must be set by the user..

**Type** ParameterizedSystem

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

# property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

### **Return type** cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

## **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

Returns A dictionary of the protocol paths that the input targeted by input\_path depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

## **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

GenerateAttachRestraints	Generates the restraint values to apply during the 'at-
	tach' phase from a set of restraint schema definitions and
	makes them easily accessible for the protocols which
	will apply them to the parameterized system.
GeneratePullRestraints	Generates the restraint values to apply during the 'pull'
	phase from a set of restraint schema definitions and
	makes them easily accessible for the protocols which
	will apply them to the parameterized system.
GenerateReleaseRestraints	Generates the restraint values to apply during the 're-
	lease' phase from a set of restraint schema definitions
	and makes them easily accessible for the protocols which
	will apply them to the parameterized system.
ApplyRestraints	A protocol which will apply the restraints defined in a
	restraints JSON file to a specified system.

### **GenerateAttachRestraints**

class openff.evaluator.protocols.paprika.restraints.GenerateAttachRestraints(protocol\_id)

Generates the restraint values to apply during the 'attach' phase from a set of restraint schema definitions and makes them easily accessible for the protocols which will apply them to the parameterized system.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_uuid(value) set_value(reference_path, value)	Prepend a unique identifier to this protocols id.  Sets the value of one of this protocols inputs.
	<u> </u>

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
attach_lambdas	<b>Input</b> - The values of lambda to use for the attach
	phase.
complex_coordinate_path	<b>Input</b> - The file path to a coordinate file which con-
	tains the solvatedhost-guest complex and has the an-
	chor dummy atoms added.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
restraint_schemas	Input - The full set of restraint schemas.
restraints_path	Output - The file path to the paprika generated re-
	straints JSON file.
schema	A serializable schema for this object.

# complex\_coordinate\_path

**Input** - The file path to a coordinate file which contains the solvatedhost-guest complex and has the anchor dummy atoms added. The default value of this attribute is not set and must be set by the user..

Type str

#### attach\_lambdas

**Input** - The values of lambda to use for the attach phase. These must start from 0.0 and increase monotonically to and include 1.0. The default value of this attribute is not set and must be set by the user..

Type *list* 

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with *template\_index* and *template\_value* 

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

# property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

# merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# restraint\_schemas

**Input** - The full set of restraint schemas. The default value of this attribute is not set and must be set by the user.

Type dict

#### restraints\_path

**Output** - The file path to the *paprika* generated restraints JSON file. The default value of this attribute is not set and must be set by the user..

Type str

## property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# **GeneratePullRestraints**

# class openff.evaluator.protocols.paprika.restraints.GeneratePullRestraints(protocol\_id)

Generates the restraint values to apply during the 'pull' phase from a set of restraint schema definitions and makes them easily accessible for the protocols which will apply them to the parameterized system.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
<pre>apply_replicator(replicator, template_values)</pre>	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allan manaina	Innut Defines whether this must seels is allowed to
allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
attach_lambdas	<b>Input</b> - The values of lambda to use for the attach
	phase.
complex_coordinate_path	<b>Input</b> - The file path to a coordinate file which con-
	tains the solvatedhost-guest complex and has the an-
	chor dummy atoms added.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
n_pull_windows	Input - The number of lambda to use for the pull
	phase.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
restraint_schemas	<b>Input</b> - The full set of restraint schemas.
restraints_path	Output - The file path to the paprika generated re-
	straints JSON file.
schema	A serializable schema for this object.

# n\_pull\_windows

**Input** - The number of lambda to use for the pull phase. The default value of this attribute is not set and must be set by the user..

Type int

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

#### attach\_lambdas

**Input** - The values of lambda to use for the attach phase. These must start from 0.0 and increase monotonically to and include 1.0. The default value of this attribute is not set and must be set by the user..

```
Type list
```

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### complex\_coordinate\_path

**Input** - The file path to a coordinate file which contains the solvatedhost-guest complex and has the anchor dummy atoms added. The default value of this attribute is not set and must be set by the user..

```
Type str
```

### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

**Return type** cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

Returns The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

# json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

• **file\_path** (str, optional) – The (optional) file path to save the JSON file to.

• **format** (*bool*) – Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

# property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

# **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## restraint\_schemas

**Input** - The full set of restraint schemas. The default value of this attribute is not set and must be set by the user..

Type dict

#### restraints\_path

**Output** - The file path to the *paprika* generated restraints JSON file. The default value of this attribute is not set and must be set by the user..

Type str

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

# set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### **GenerateReleaseRestraints**

class openff.evaluator.protocols.paprika.restraints.GenerateReleaseRestraints(protocol\_id)

Generates the restraint values to apply during the 'release' phase from a set of restraint schema definitions and makes them easily accessible for the protocols which will apply them to the parameterized system.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
<pre>set_value(reference_path, value)</pre>	Sets the value of one of this protocols inputs.
<pre>set_value(reference_path, value) validate([attribute_type])</pre>	Sets the value of one of this protocols inputs.  Validate the values of the attributes.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
host_coordinate_path	<b>Input</b> - The file path to a coordinate file which con-
	tains the solvatedhost molecule and has the anchor
	dummy atoms added.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
release_lambdas	<b>Input</b> - The values of lambda to use for the release
	phase.
required_inputs	The inputs which must be set on this protocol.
restraint_schemas	Input - The full set of restraint schemas.
restraints_path	Output - The file path to the paprika generated re-
	straints JSON file.
schema	A serializable schema for this object.

# host\_coordinate\_path

**Input** - The file path to a coordinate file which contains the solvatedhost molecule and has the anchor dummy atoms added. The default value of this attribute is not set and must be set by the user..

Type str

### release\_lambdas

**Input** - The values of lambda to use for the release phase. These must start from 1.0 and decrease monotonically to and include 0.0. The default value of this attribute is not set and must be set by the user..

Type *list* 

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

# property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

#### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

# property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# restraint\_schemas

**Input** - The full set of restraint schemas. The default value of this attribute is not set and must be set by the user.

Type dict

#### restraints\_path

**Output** - The file path to the *paprika* generated restraints JSON file. The default value of this attribute is not set and must be set by the user..

Type str

## property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# **ApplyRestraints**

 $\textbf{class} \ open ff. evaluator. protocols. paprika. restraints. \textbf{ApplyRestraints} (\textit{protocol\_id})$ 

A protocol which will apply the restraints defined in a restraints JSON file to a specified system.

\_\_init\_\_(protocol\_id)

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
	Creates a JSON representation of this class.
load_restraints(file_path)	Loads a set of <i>paprika</i> restraint objects from a JSON
	file.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
input_system	Input - The parameterized system which the re-
	straints should be added to.
output_system	Output - The parameterized system which now in-
	cludes the added restraints.
outputs	A dictionary of the outputs of this property.
phase	<b>Input</b> - The APR phase to take the restraints from.
required_inputs	The inputs which must be set on this protocol.
restraints_path	<b>Input</b> - The file path to the JSON file which contains
	the restraint definitions.
schema	A serializable schema for this object.
window_index	<b>Input</b> - The index of the window to take the restraints
	from.

### restraints\_path

**Input** - The file path to the JSON file which contains the restraint definitions. This will usually have been generated by a *GenerateXXXRestraints* protocol. The default value of this attribute is not set and must be set by the user..

Type str

#### phase

**Input** - The APR phase to take the restraints from. The default value of this attribute is not set and must be set by the user..

Type str

## window\_index

**Input** - The index of the window to take the restraints from. The default value of this attribute is not set and must be set by the user..

Type int

# input\_system

**Input** - The parameterized system which the restraints should be added to. The default value of this attribute is not set and must be set by the user.

Type ParameterizedSystem

## output\_system

**Output** - The parameterized system which now includes the added restraints. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

# classmethod load\_restraints(file\_path: str)

Loads a set of *paprika* restraint objects from a JSON file.

**Parameters file\_path** – The path to the JSON serialized restraints.

**Return type** The loaded *paprika* restraint objects.

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

## property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

```
Parameters string_contents (str or bytes) – The typed json string.
```

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

## Raises ValueError or AssertionError -

AnalyzeAPRPhase	A protocol which will analyze the outputs of the attach, pull or release phases of an APR calculation and return the change in free energy for that phase of the calcula-
ComputeSymmetryCorrection	tion.  Computes the symmetry correction for an APR calcula-
	tion which involves a guest with symmetry.
ComputeReferenceWork	Computes the reference state work.

# AnalyzeAPRPhase

# $\textbf{class} \ open ff. evaluator. protocols. paprika. analysis. \textbf{AnalyzeAPRPhase} (\textit{protocol\_id})$

A protocol which will analyze the outputs of the attach, pull or release phases of an APR calculation and return the change in free energy for that phase of the calculation.

\_\_init\_\_(protocol\_id)

# **Methods**

another.  execute([directory, available_resources])	init(protocol_id)	
another.  execute([directory, available_resources])	apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol		another.
from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	<pre>execute([directory, available_resources])</pre>	Execute the protocol.
get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	<pre>from_json(file_path)</pre>	Create this object from a JSON file.
get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	from_schema(schema)	Initializes a protocol from it's schema definition.
attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / out puts.  get_value_references(input_path)  Returns a dictionary of references to the protocol	<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
puts.  get_value_references(input_path)  Returns a dictionary of references to the protocol		attributes directly (rather than its value).
get_value_references(input_path)  Returns a dictionary of references to the protocol	get_value(reference_path)	Returns the value of one of this protocols inputs / out-
		puts.
which one of this protocols inputs (specified by <i>in</i>	<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
		which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>		<pre>put_path) takes its value from.</pre>
<i>json</i> ([file_path, format]) Creates a JSON representation of this class.	<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
<i>merge</i> (other) Merges another Protocol with this one.	merge(other)	Merges another Protocol with this one.
parse_json(string_contents) Parses a typed json string into the corresponding class	parse_json(string_contents)	Parses a typed json string into the corresponding class
structure.		structure.
	replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value) Prepend a unique identifier to this protocols id.		Prepend a unique identifier to this protocols id.
set_value(reference_path, value) Sets the value of one of this protocols inputs.	set_value(reference_path, value)	<u> </u>
validate([attribute_type])Validate the values of the attributes.	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
phase	<b>Input</b> - The phase of the calculation being analyzed.
required_inputs	The inputs which must be set on this protocol.
restraints_path	<b>Input</b> - The file path to the JSON file which contains
	the restraint definitions.
result	Output - The analysed free energy.
schema	A serializable schema for this object.
topology_path	<b>Input</b> - The file path to a coordinate file which con-
	tains topological information about the system.
trajectory_paths	<b>Input</b> - A list of paths to the trajectories (in the cor-
	rect order) generated during the phase being ana-
	lyzed.

## topology\_path

**Input** - The file path to a coordinate file which contains topological information about the system. The default value of this attribute is not set and must be set by the user..

Type str

## trajectory\_paths

**Input** - A list of paths to the trajectories (in the correct order) generated during the phase being analyzed. The default value of this attribute is not set and must be set by the user..

Type list

## phase

**Input** - The phase of the calculation being analyzed. The default value of this attribute is not set and must be set by the user..

Type str

# restraints\_path

**Input** - The file path to the JSON file which contains the restraint definitions. This will usually have been generated by a *GenerateXXXRestraints* protocol. The default value of this attribute is not set and must be set by the user..

Type str

### result

Output - The analysed free energy. The default value of this attribute is not set and must be set by the user...

Type Observable

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

# **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

#### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## ComputeSymmetryCorrection

```
class openff.evaluator.protocols.paprika.analysis.ComputeSymmetryCorrection(protocol_id)

Computes the symmetry correction for an APR calculation which involves a guest with symmetry.
```

```
__init__(protocol_id)
```

# Methods

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.
Validate the values of the attributes.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
n_microstates	<b>Input</b> - The number of symmetry microstates of the
	guest molecule.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
result	Output - The symmetry correction.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The thermodynamic state that the calculation
	was performed at.

# n\_microstates

Input - The number of symmetry microstates of the guest molecule. The default value of this attribute is not set and must be set by the user..

Type int

#### thermodynamic\_state

**Input** - The thermodynamic state that the calculation was performed at. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

#### result

**Output** - The symmetry correction. The default value of this attribute is not set and must be set by the user..

Type Observable

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

**Parameters** 

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

 $\label{parameters} \textbf{Parameters} \ \ \textbf{reference\_path} \ (\texttt{ProtocolPath}) - \textbf{The path pointing to the value to return.}$ 

**Returns** The value of the input / output

**Return type** Any

# get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

```
validate(attribute_type=None)
```

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## ComputeReferenceWork

```
\begin{cal}{class open} ff. evaluator. protocols. paprika. analysis. {\it ComputeReferenceWork} (protocol\_id) \\ Computes the reference state work. \end{cal}
```

```
__init__(protocol_id)
```

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
<pre>set_value(reference_path, value)</pre>	Sets the value of one of this protocols inputs.
<pre>set_value(reference_path, value) validate([attribute_type])</pre>	Sets the value of one of this protocols inputs.  Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
restraints_path	<b>Input</b> - The file path to the JSON file which contains
	the restraint definitions.
result	Output - The reference state work.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The thermodynamic state that the calculation
	was performed at.

# thermodynamic\_state

**Input** - The thermodynamic state that the calculation was performed at. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

# restraints\_path

Input - The file path to the JSON file which contains the restraint definitions. This will usually have been

generated by a *GenerateXXXRestraints* protocol. The default value of this attribute is not set and must be set by the user..

Type str

#### result

Output - The reference state work. The default value of this attribute is not set and must be set by the user..

Type Observable

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• **update\_input\_references** (*bool*) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

### **Parameters**

• **other** (Protocol) – The protocol to compare against.

• path\_replacements (list of tuple of str, optional) - Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters** reference\_path (ProtocolPath) – The path pointing to the value to return.

Returns The value of the input / output

#### **Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list | dict* which contains at least one ProtocolPath.

**Parameters** input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

**Return type** dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

Returns A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

# set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

# set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# Reweighting

ConcatenateTrajectories	A protocol which concatenates multiple trajectories into
	a single one.
ConcatenateObservables	A protocol which concatenates multiple
	ObservableFrame objects into a single
	ObservableFrame object.
BaseEvaluateEnergies	A base class for protocols which will re-evaluate the en-
	ergy of a series of configurations for a given set of force
	field parameters.
BaseMBARProtocol	Re-weights a set of observables using MBAR to calcu-
	late the average value of the observables at a different
	state than they were originally measured.
ReweightObservable	Reweight an array of observables to a new state using
	MBAR.
ReweightDielectricConstant	Computes the avergage value of the dielectric constant
	be re-weighting a set a set of dipole moments and vol-
	umes using MBAR.

# ConcatenateTrajectories

 $\textbf{class} \ open ff. evaluator. protocols. reweighting. \textbf{ConcatenateTrajectories} (\textit{protocol\_id})$ 

A protocol which concatenates multiple trajectories into a single one.

\_\_init\_\_(protocol\_id)

# **Methods**

another.  execute([directory, available_resources])  from_json(file_path)  from_schema(schema)  get_attributes([attribute_type])  get_class_attribute(reference_path)  get_value(reference_path)  get_value_references(input_path)  another.  Execute the protocol.  Execute the protocol.  Execute the protocol.  Execute the protocol.  Initializes a protocol from it's schema definition.  Returns all attributes of a specific attribute_type  get_class_attribute(reference_path)  Returns one of this protocols, or any of its childrent attributes directly (rather than its value).  Returns the value of one of this protocols inputs / puts.  get_value_references(input_path)  Returns a dictionary of references to the protocols.	init(protocol_id)	
can_merge(other[, path_replacements])Determines whether this protocol can be merged another.execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childred attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.		another.
from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>execute([directory, available_resources])</pre>	Execute the protocol.
get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	from_json(file_path)	Create this object from a JSON file.
get_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	from_schema(schema)	Initializes a protocol from it's schema definition.
attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / puts.  get_value_references(input_path)  Returns a dictionary of references to the proto which one of this protocols inputs (specified by put_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.	<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
get_value(reference_path)       Returns the value of one of this protocols inputs / puts.         get_value_references(input_path)       Returns a dictionary of references to the protocol which one of this protocols inputs (specified by put_path) takes its value from.         json([file_path, format])       Creates a JSON representation of this class.	<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
puts.  get_value_references(input_path)  Returns a dictionary of references to the proto which one of this protocols inputs (specified by put_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.		attributes directly (rather than its value).
get_value_references(input_path)Returns a dictionary of references to the proto which one of this protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
which one of this protocols inputs (specified by <a href="mailto:put_path">put_path</a> ) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.		1
put_path)takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
json([file_path, format]) Creates a JSON representation of this class.		which one of this protocols inputs (specified by in-
		<pre>put_path) takes its value from.</pre>
merge(other) Merges another Protocol with this one.	<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
	merge(other)	Merges another Protocol with this one.
parse_json(string_contents) Parses a typed json string into the corresponding of	<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
structure.		structure.
<u> </u>		Finds each input which came from a given protocol
set_uuid(value) Prepend a unique identifier to this protocols id.	set_uuid(value)	<u> </u>
set_value(reference_path, value) Sets the value of one of this protocols inputs.		
<i>validate</i> ([attribute_type]) Validate the values of the attributes.	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
input_coordinate_paths	Input - A list of paths to the starting PDB coordinates
	for each of the trajectories.
input_trajectory_paths	<b>Input</b> - A list of paths to the trajectories to concate-
	nate.
output_coordinate_path	Output - The path the PDB coordinate file which
	contains the topology of the concatenated trajectory.
output_trajectory_path	Output - The path to the concatenated trajectory.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.

#### input\_coordinate\_paths

**Input** - A list of paths to the starting PDB coordinates for each of the trajectories. The default value of this attribute is not set and must be set by the user..

Type *list* 

# input\_trajectory\_paths

**Input** - A list of paths to the trajectories to concatenate. The default value of this attribute is not set and must be set by the user..

Type list

## output\_coordinate\_path

**Output** - The path the PDB coordinate file which contains the topology of the concatenated trajectory. The default value of this attribute is not set and must be set by the user..

Type str

#### output\_trajectory\_path

**Output** - The path to the concatenated trajectory. The default value of this attribute is not set and must be set by the user..

Type str

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

## **Parameters**

• replicator (ProtocolReplicator) – The replicator to apply.

• **template\_values** (*list of Any*) – A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

**execute**(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

#### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

```
set_value(reference_path, value)
```

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

```
validate(attribute_type=None)
```

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### ConcatenateObservables

class openff.evaluator.protocols.reweighting.ConcatenateObservables(protocol\_id)

A protocol which concatenates multiple ObservableFrame objects into a single ObservableFrame object.

```
__init__(protocol_id)
```

# **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

<b>Input</b> - Defines whether this protocols is allowed to
merge with other protocols.
A list of pointers to the protocols which this protocol
takes input from.
The unique id of this protocol.
<b>Input</b> - A list of observable arrays to concatenate.
Output - The concatenated observable array.
A dictionary of the outputs of this property.
The inputs which must be set on this protocol.
A serializable schema for this object.

# input\_observables

**Input** - A list of observable arrays to concatenate. The default value of this attribute is not set and must be set by the user..

Type list

# output\_observables

**Output** - The concatenated observable array. The default value of this attribute is not set and must be set by the user..

Type Union

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template index or template value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

# classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

# get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters** reference\_path (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

Returns The JSON representation of this class.

```
Return type str
```

# merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** Dict[str, str]

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

## **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

## **BaseEvaluateEnergies**

# class openff.evaluator.protocols.reweighting.BaseEvaluateEnergies(protocol\_id)

A base class for protocols which will re-evaluate the energy of a series of configurations for a given set of force field parameters.

\_\_init\_\_(protocol\_id)

# Methods

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
json([file_path, format])	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
set_varue(reference_path, value)	sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
enable_pbc	<b>Input</b> - If true, periodic boundary conditions will be
	enabled.
gradient_parameters	Input - An optional list of parameters to differentiate
	the evaluated energies with respect to.
id	The unique id of this protocol.
output_observables	Output - An observable array which stores the re-
	duced potentials potential energies evaluated at the
	specified state and using the specified system object
	for each configuration in the trajectory.
outputs	A dictionary of the outputs of this property.
parameterized_system	Input - The parameterized system object which en-
	codes the systems potential energy function.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
thermodynamic_state	<b>Input</b> - The state to calculate the reduced potentials
	at.
trajectory_file_path	<b>Input</b> - The path to the trajectory file which contains
	the configurations to calculate the energies of.

#### thermodynamic\_state

**Input** - The state to calculate the reduced potentials at. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

#### parameterized\_system

**Input** - The parameterized system object which encodes the systems potential energy function. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

# enable\_pbc

Input - If true, periodic boundary conditions will be enabled. The default value of this attribute is True.

Type bool

# trajectory\_file\_path

**Input** - The path to the trajectory file which contains the configurations to calculate the energies of. The default value of this attribute is not set and must be set by the user..

Type str

#### gradient\_parameters

**Input** - An optional list of parameters to differentiate the evaluated energies with respect to.

Type *list* 

## output\_observables

**Output** - An observable array which stores the reduced potentials potential energies evaluated at the specified state and using the specified system object for each configuration in the trajectory. The default value of this attribute is not set and must be set by the user..

Type ObservableFrame

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

# property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

#### Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters** reference\_path (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

Returns The value of the input / output

**Return type** Any

# get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

## **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

Returns A dictionary of the protocol paths that the input targeted by input\_path depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

# **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

#### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

#### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

## **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# **BaseMBARProtocol**

# class openff.evaluator.protocols.reweighting.BaseMBARProtocol(protocol\_id)

Re-weights a set of observables using MBAR to calculate the average value of the observables at a different state than they were originally measured.

\_\_init\_\_(protocol\_id)

## **Methods**

init(protocol_id)	
<pre>apply_replicator(replicator, template_values)</pre>	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
get_value(reference_path)	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
validate([attribute_type])	Validate the values of the attributes.

#### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
bootstrap_iterations	<b>Input</b> - The number of bootstrap iterations to perform
	if bootstraped uncertainties have been requested The
	default value of this attribute is 250.
bootstrap_uncertainties	Input - If true, bootstrapping will be used to esti-
	mated the total uncertainty in the reweighted value.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
effective_samples	Output - The number of effective samples which
	were re-weighted.
frame_counts	<b>Input</b> - The number of configurations per reference
	state.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
reference_reduced_potentials	<b>Input</b> - The reduced potentials of each configuration
	evaluated at each of the reference states.
required_effective_samples	<b>Input</b> - The minimum number of effective samples
	required to be able to reweight the observable.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
target_reduced_potentials	<b>Input</b> - The reduced potentials of each configuration
	evaluated at the target state.
value	Output - The re-weighted average value of the ob-
	servable at the target state.

#### reference\_reduced\_potentials: List[ObservableArray]

**Input** - The reduced potentials of each configuration evaluated at each of the reference states. The default value of this attribute is not set and must be set by the user..

Type list

## target\_reduced\_potentials

**Input** - The reduced potentials of each configuration evaluated at the target state. The default value of this attribute is not set and must be set by the user..

Type ObservableArray

#### frame\_counts

**Input** - The number of configurations per reference state. The sum of theseshould equal the length of the reference\_reduced\_potentials and target\_reduced\_potentials input arrays as well any input observable arrays. The default value of this attribute is not set and must be set by the user..

Type list

#### bootstrap\_uncertainties

**Input** - If true, bootstrapping will be used to estimated the total uncertainty in the reweighted value. The default value of this attribute is False.

Type bool

## bootstrap\_iterations

**Input** - The number of bootstrap iterations to perform if bootstraped uncertainties have been requested The default value of this attribute is 250.

Type int

#### required\_effective\_samples

**Input** - The minimum number of effective samples required to be able to reweight the observable. If the effective samples is less than this minimum an exception will be raised. The default value of this attribute is 50.

Type int

#### value

**Output** - The re-weighted average value of the observable at the target state. The default value of this attribute is not set and must be set by the user..

Type Observable

#### effective\_samples

**Output** - The number of effective samples which were re-weighted. The default value of this attribute is not set and must be set by the user..

Type float

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update input references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

```
can_merge(other, path_replacements=None)
```

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

# property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

Returns The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

# merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

**Type** list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

# set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

# validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

# ReweightObservable

 $\textbf{class} \ \ \textbf{openff.evaluator.protocols.reweighting.} \textbf{Reweight0bservable} (\textit{protocol\_id})$ 

Reweight an array of observables to a new state using MBAR.

\_\_init\_\_(protocol\_id)

# **Methods**

another.  execute([directory, available_resources])  from_json(file_path)  from_schema(schema)  get_attributes([attribute_type])  get_class_attribute(reference_path)  get_value(reference_path)  get_value_references(input_path)  another.  Execute the protocol.  Execute the protocol.  Execute the protocol.  Execute the protocol.  Initializes a protocol from it's schema definition.  Returns all attributes of a specific attribute_type  get_class_attribute(reference_path)  Returns one of this protocols, or any of its childrent attributes directly (rather than its value).  Returns the value of one of this protocols inputs / puts.  get_value_references(input_path)  Returns a dictionary of references to the protocols.	init(protocol_id)	
can_merge(other[, path_replacements])Determines whether this protocol can be merged another.execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childred attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.		another.
from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>execute([directory, available_resources])</pre>	Execute the protocol.
get_attributes([attribute_type])Returns all attributes of a specific attribute_typeget_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	from_json(file_path)	Create this object from a JSON file.
get_class_attribute(reference_path)Returns one of this protocols, or any of its childrent attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / puts.get_value_references(input_path)Returns a dictionary of references to the protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	from_schema(schema)	Initializes a protocol from it's schema definition.
attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / puts.  get_value_references(input_path)  Returns a dictionary of references to the proto which one of this protocols inputs (specified by put_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.	<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
get_value(reference_path)       Returns the value of one of this protocols inputs / puts.         get_value_references(input_path)       Returns a dictionary of references to the protocol which one of this protocols inputs (specified by put_path) takes its value from.         json([file_path, format])       Creates a JSON representation of this class.	<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
puts.  get_value_references(input_path)  Returns a dictionary of references to the proto which one of this protocols inputs (specified by put_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.		attributes directly (rather than its value).
get_value_references(input_path)Returns a dictionary of references to the proto which one of this protocols inputs (specified by put_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
which one of this protocols inputs (specified by <a href="mailto:put_path">put_path</a> ) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.		1
put_path)takes its value from.json([file_path, format])Creates a JSON representation of this class.	<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
json([file_path, format]) Creates a JSON representation of this class.		which one of this protocols inputs (specified by in-
		<pre>put_path) takes its value from.</pre>
merge(other) Merges another Protocol with this one.	<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
	merge(other)	Merges another Protocol with this one.
parse_json(string_contents) Parses a typed json string into the corresponding of	<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
structure.		structure.
<u> </u>		Finds each input which came from a given protocol
set_uuid(value) Prepend a unique identifier to this protocols id.	set_uuid(value)	<u> </u>
set_value(reference_path, value) Sets the value of one of this protocols inputs.		
<i>validate</i> ([attribute_type]) Validate the values of the attributes.	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

- 7.7	T D. C 1
allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
bootstrap_iterations	<b>Input</b> - The number of bootstrap iterations to perform
	if bootstraped uncertainties have been requested The
	default value of this attribute is 250.
bootstrap_uncertainties	Input - If true, bootstrapping will be used to esti-
	mated the total uncertainty in the reweighted value.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
effective_samples	Output - The number of effective samples which
	were re-weighted.
frame_counts	<b>Input</b> - The number of configurations per reference
	state.
id	The unique id of this protocol.
observable	Input - The observables to reweight.
outputs	A dictionary of the outputs of this property.
reference_reduced_potentials	<b>Input</b> - The reduced potentials of each configuration
	evaluated at each of the reference states.
required_effective_samples	<b>Input</b> - The minimum number of effective samples
	required to be able to reweight the observable.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
target_reduced_potentials	<b>Input</b> - The reduced potentials of each configuration
	evaluated at the target state.
value	Output - The re-weighted average value of the ob-
	servable at the target state.

#### observable

**Input** - The observables to reweight. The array should contain the values of the observable evaluated for of each configuration at the target state. The default value of this attribute is not set and must be set by the user..

**Type** ObservableArray

# allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

## **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with *template\_index* and *template\_value* 

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is provided.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

#### bootstrap\_iterations

**Input** - The number of bootstrap iterations to perform if bootstraped uncertainties have been requested The default value of this attribute is 250.

Type int

## bootstrap\_uncertainties

**Input** - If true, bootstrapping will be used to estimated the total uncertainty in the reweighted value. The default value of this attribute is False.

Type bool

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

# effective\_samples

**Output** - The number of effective samples which were re-weighted. The default value of this attribute is not set and must be set by the user..

Type float

```
execute(directory=", available_resources=None)
```

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

#### frame\_counts

**Input** - The number of configurations per reference state. The sum of theseshould equal the length of the reference\_reduced\_potentials and target\_reduced\_potentials input arrays as well any input observable arrays. The default value of this attribute is not set and must be set by the user..

```
Type list
```

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (str) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

**Return type** list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters** reference\_path (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

```
Return type str
```

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

```
Return type Dict[str, str]
```

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

# classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

```
Parameters string_contents (str or bytes) – The typed json string.
```

**Returns** The parsed class.

**Return type** Any

## reference\_reduced\_potentials: List[ObservableArray]

**Input** - The reduced potentials of each configuration evaluated at each of the reference states. The default value of this attribute is not set and must be set by the user..

```
Type list
```

```
replace_protocol(old_id, new_id)
```

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# required\_effective\_samples

**Input** - The minimum number of effective samples required to be able to reweight the observable. If the effective samples is less than this minimum an exception will be raised. The default value of this attribute is 50.

Type int

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

#### property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## target\_reduced\_potentials

**Input** - The reduced potentials of each configuration evaluated at the target state. The default value of this attribute is not set and must be set by the user..

**Type** ObservableArray

#### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

**Output** - The re-weighted average value of the observable at the target state. The default value of this attribute is not set and must be set by the user..

Type Observable

# ReweightDielectricConstant

# $\textbf{class} \ \ \textbf{openff.evaluator.protocols.reweighting.} \textbf{ReweightDielectricConstant} (\textit{protocol\_id})$

Computes the avergage value of the dielectric constant be re-weighting a set a set of dipole moments and volumes using MBAR.

\_\_init\_\_(protocol\_id)

# **Methods**

another.  execute([directory, available_resources])	init(protocol_id)	
another.  execute([directory, available_resources])	apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol		another.
from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	<pre>execute([directory, available_resources])</pre>	Execute the protocol.
get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	<pre>from_json(file_path)</pre>	Create this object from a JSON file.
get_class_attribute(reference_path)Returns one of this protocols, or any of its children's attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	from_schema(schema)	Initializes a protocol from it's schema definition.
attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / out puts.  get_value_references(input_path)  Returns a dictionary of references to the protocol	<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
get_value(reference_path)Returns the value of one of this protocols inputs / out puts.get_value_references(input_path)Returns a dictionary of references to the protocol	<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
puts.  get_value_references(input_path)  Returns a dictionary of references to the protocol		attributes directly (rather than its value).
get_value_references(input_path)  Returns a dictionary of references to the protocol	get_value(reference_path)	Returns the value of one of this protocols inputs / out-
		puts.
which one of this protocols inputs (specified by <i>in</i>	<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
		which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>		<pre>put_path) takes its value from.</pre>
<i>json</i> ([file_path, format]) Creates a JSON representation of this class.	<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
<i>merge</i> (other) Merges another Protocol with this one.	merge(other)	Merges another Protocol with this one.
parse_json(string_contents) Parses a typed json string into the corresponding class	parse_json(string_contents)	Parses a typed json string into the corresponding class
structure.		structure.
	replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value) Prepend a unique identifier to this protocols id.		Prepend a unique identifier to this protocols id.
set_value(reference_path, value) Sets the value of one of this protocols inputs.	set_value(reference_path, value)	<u> </u>
validate([attribute_type])Validate the values of the attributes.	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to merge with other protocols.
bootstrap_iterations	<b>Input</b> - The number of bootstrap iterations to perform if bootstraped uncertainties have been requested The
	default value of this attribute is 250.
bootstrap_uncertainties	<b>Input</b> - If true, bootstrapping will be used to estimated the total uncertainty in the reweighted value.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
dipole_moments	<b>Input</b> - The dipole moments evaluated at reference
	state's configurations using the force field of the target
	state.
effective_samples	Output - The number of effective samples which
	were re-weighted.
frame_counts	<b>Input</b> - The number of configurations per reference
	state.
id	The unique id of this protocol.
outputs	A dictionary of the outputs of this property.
reference_reduced_potentials	<b>Input</b> - The reduced potentials of each configuration
	evaluated at each of the reference states.
required_effective_samples	<b>Input</b> - The minimum number of effective samples
	required to be able to reweight the observable.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
target_reduced_potentials	<b>Input</b> - The reduced potentials of each configuration
	evaluated at the target state.
thermodynamic_state	<b>Input</b> - The thermodynamic state to re-weight to.
value	Output - The re-weighted average value of the ob-
	servable at the target state.
volumes	Input - The dipole moments evaluated at reference
	state's configurations using the force field of the target
	state.

# dipole\_moments

**Input** - The dipole moments evaluated at reference state's configurations using the force field of the target state. The default value of this attribute is not set and must be set by the user..

Type Union

## volumes

**Input** - The dipole moments evaluated at reference state's configurations using the force field of the target state. The default value of this attribute is not set and must be set by the user..

Type Union

# thermodynamic\_state

**Input** - The thermodynamic state to re-weight to. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

#### bootstrap\_uncertainties

**Input** - If true, bootstrapping will be used to estimated the total uncertainty in the reweighted value. The default value of this attribute is False.

Type bool

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

# bootstrap\_iterations

**Input** - The number of bootstrap iterations to perform if bootstraped uncertainties have been requested The default value of this attribute is 250.

Type int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

**Parameters** 

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

**Type** list of ProtocolPath

# effective\_samples

**Output** - The number of effective samples which were re-weighted. The default value of this attribute is not set and must be set by the user..

Type float

**execute**(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

#### frame\_counts

**Input** - The number of configurations per reference state. The sum of theseshould equal the length of the reference\_reduced\_potentials and target\_reduced\_potentials input arrays as well any input observable arrays. The default value of this attribute is not set and must be set by the user..

Type list

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

Parameters attribute\_type (type of Attribute, optional) - The type of attribute to search for

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

**Parameters reference\_path** (ProtocolPath) – The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

## **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

## merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

# property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

#### classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

Return type Any

## reference\_reduced\_potentials: List[ObservableArray]

**Input** - The reduced potentials of each configuration evaluated at each of the reference states. The default value of this attribute is not set and must be set by the user..

Type list

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## required\_effective\_samples

**Input** - The minimum number of effective samples required to be able to reweight the observable. If the effective samples is less than this minimum an exception will be raised. The default value of this attribute is 50.

Type int

#### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

# set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

#### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

## **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## target\_reduced\_potentials

**Input** - The reduced potentials of each configuration evaluated at the target state. The default value of this attribute is not set and must be set by the user..

**Type** ObservableArray

validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

#### value

**Output** - The re-weighted average value of the observable at the target state. The default value of this attribute is not set and must be set by the user..

Type Observable

#### **Simulation**

BaseEnergyMinimisation	A base class for protocols which will minimise the po-
	tential energy of a given system.
BaseSimulation	A base class for protocols which will perform a molec-
	ular simulation in a given ensemble and at a specified
	state.

# BaseEnergyMinimisation

 ${\bf class} \ open ff. evaluator. protocols. simulation. {\bf Base Energy Minimisation} ({\it protocol\_id})$ 

A base class for protocols which will minimise the potential energy of a given system.

\_\_init\_\_(protocol\_id)

# **Methods**

Applies a <i>ProtocolReplicator</i> to this protocol.
Determines whether this protocol can be merged with
another.
Execute the protocol.
Create this object from a JSON file.
Initializes a protocol from it's schema definition.
Returns all attributes of a specific <i>attribute_type</i> .
Returns one of this protocols, or any of its children's,
attributes directly (rather than its value).
Returns the value of one of this protocols inputs / out-
puts.
Returns a dictionary of references to the protocols
which one of this protocols inputs (specified by in-
<pre>put_path) takes its value from.</pre>
Creates a JSON representation of this class.
Merges another Protocol with this one.
Parses a typed json string into the corresponding class
structure.
Finds each input which came from a given protocol
Prepend a unique identifier to this protocols id.
Sets the value of one of this protocols inputs.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
enable_pbc	Input - If true, periodic boundary conditions will be
	enabled.
id	The unique id of this protocol.
input_coordinate_file	Input - The coordinates to minimise.
max_iterations	<b>Input</b> - The maximum number of iterations to per-
	form.
	ioriii.
output_coordinate_file	Output - The file path to the minimised coordinates.
output_coordinate_file outputs	
<del>-</del>	Output - The file path to the minimised coordinates.
outputs	Output - The file path to the minimised coordinates.  A dictionary of the outputs of this property.
outputs	Output - The file path to the minimised coordinates.  A dictionary of the outputs of this property.  Input - The parameterized system object which en-
outputs parameterized_system	Output - The file path to the minimised coordinates.  A dictionary of the outputs of this property.  Input - The parameterized system object which encodes the systems potential energy function.
outputs parameterized_system required_inputs	Output - The file path to the minimised coordinates.  A dictionary of the outputs of this property.  Input - The parameterized system object which encodes the systems potential energy function.  The inputs which must be set on this protocol.
outputs parameterized_system  required_inputs schema	Output - The file path to the minimised coordinates.  A dictionary of the outputs of this property.  Input - The parameterized system object which encodes the systems potential energy function.  The inputs which must be set on this protocol.  A serializable schema for this object.

# input\_coordinate\_file

Input - The coordinates to minimise. The default value of this attribute is not set and must be set by the

user..

Type str

## parameterized\_system

**Input** - The parameterized system object which encodes the systems potential energy function. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

#### tolerance

**Input** - The energy tolerance to which the system should be minimized. The default value of this attribute is 10.0 kilojoule / mole.

Type Quantity

#### max\_iterations

**Input** - The maximum number of iterations to perform. If this is 0, minimization is continued until the results converge without regard to how many iterations it takes. The default value of this attribute is **0**.

Type int

### enable\_pbc

Input - If true, periodic boundary conditions will be enabled. The default value of this attribute is True.

Type bool

#### output\_coordinate\_file

**Output** - The file path to the minimised coordinates. The default value of this attribute is not set and must be set by the user.

Type str

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- other (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

#### classmethod from\_json(file path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

## get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

#### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

#### property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended uuid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

#### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

## **BaseSimulation**

## class openff.evaluator.protocols.simulation.BaseSimulation(protocol\_id)

A base class for protocols which will perform a molecular simulation in a given ensemble and at a specified state.

\_\_init\_\_(protocol\_id)

## **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
replace_protocol(old_id, new_id)	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
<pre>set_value(reference_path, value)</pre>	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_gpu_platforms	<b>Input</b> - If true, the simulation will be performed
diion_gpu_pidcioims	using a GPU if available, otherwise it will be con-
	strained to only using CPUs.
allow_merging	Input - Defines whether this protocols is allowed to
	merge with other protocols.
checkpoint_frequency	Input - The frequency (in multiples of out-
	<pre>put_frequency) with which to write to a checkpoint</pre>
	file, e.g.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
enable_pbc	Input - If true, periodic boundary conditions will be
	enabled.
ensemble	<b>Input</b> - The thermodynamic ensemble to simulate in.
gradient_parameters	<b>Input</b> - An optional list of parameters to differentiate
	the evaluated energies with respect to.
high_precision	Input - If true, the simulation will be run using dou-
	ble precision.
id	The unique id of this protocol.
<pre>input_coordinate_file</pre>	<b>Input</b> - The file path to the starting coordinates.
observables	Output - The observables collected during the sim-
	ulation.
output_coordinate_file	<b>Output</b> - The file path to the coordinates of the final
	system configuration.
output_frequency	Input - The frequency (in number of steps) with
	which to write to the output statistics and trajectory
	files.
outputs	A dictionary of the outputs of this property.
parameterized_system	Input - The parameterized system object which en-
	codes the systems potential energy function.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
steps_per_iteration	<b>Input</b> - The number of steps to propogate the system
	by at each iteration.
thermodynamic_state	<b>Input</b> - The thermodynamic conditions to simulate
	under The default value of this attribute is not set and
	must be set by the user
thermostat_friction	Input - The thermostat friction coefficient.
timestep	<b>Input</b> - The timestep to evolve the system by at each
	step.
total_number_of_iterations	<b>Input</b> - The number of times to propogate the system
	forward by the <i>steps_per_iteration</i> number of steps.
trajectory_file_path	Output - The file path to the trajectory sampled dur-
	ing the simulation.

## steps\_per\_iteration

**Input** - The number of steps to propogate the system by at each iteration. The total number of steps performed by this protocol will be *total\_number\_of\_iterations* \* *steps\_per\_iteration*. The default value of this attribute is 1000000.

Type int

### total\_number\_of\_iterations

**Input** - The number of times to propogate the system forward by the *steps\_per\_iteration* number of steps. The total number of steps performed by this protocol will be *total\_number\_of\_iterations* \* *steps\_per\_iteration*. The default value of this attribute is 1.

Type int

## output\_frequency

**Input** - The frequency (in number of steps) with which to write to the output statistics and trajectory files. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 3000.

Type int

## checkpoint\_frequency

**Input** - The frequency (in multiples of *output\_frequency*) with which to write to a checkpoint file, e.g. if *output\_frequency=100* and *checkpoint\_frequency==2*, a checkpoint file would be saved every 200 steps. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 10. This attribute is *optional*.

Type int

## timestep

**Input** - The timestep to evolve the system by at each step. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 2.0 femtosecond.

Type Quantity

#### thermodynamic\_state

**Input** - The thermodynamic conditions to simulate under The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

## ensemble

Input - The thermodynamic ensemble to simulate in. The default value of this attribute is Ensemble.NPT.

Type Ensemble

#### thermostat\_friction

**Input** - The thermostat friction coefficient. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 1.0 / picosecond.

Type Quantity

#### input\_coordinate\_file

**Input** - The file path to the starting coordinates. The default value of this attribute is not set and must be set by the user..

Type str

## parameterized\_system

**Input** - The parameterized system object which encodes the systems potential energy function. The default value of this attribute is not set and must be set by the user.

Type ParameterizedSystem

## enable\_pbc

Input - If true, periodic boundary conditions will be enabled. The default value of this attribute is True.

Type bool

#### allow\_gpu\_platforms

**Input** - If true, the simulation will be performed using a GPU if available, otherwise it will be constrained to only using CPUs. The default value of this attribute is True.

Type bool

## high\_precision

**Input** - If true, the simulation will be run using double precision. The default value of this attribute is False.

Type bool

#### gradient\_parameters

**Input** - An optional list of parameters to differentiate the evaluated energies with respect to.

Type list

#### output\_coordinate\_file

**Output** - The file path to the coordinates of the final system configuration. The default value of this attribute is not set and must be set by the user..

Type str

## trajectory\_file\_path

**Output** - The file path to the trajectory sampled during the simulation. The default value of this attribute is not set and must be set by the user.

Type str

#### observables

**Output** - The observables collected during the simulation. The default value of this attribute is not set and must be set by the user..

Type ObservableFrame

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with *template\_index* and *template\_value* 

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

#### **Return type** cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

Returns A dictionary of the protocol paths that the input targeted by input\_path depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

#### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

#### property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

#### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

validate(attribute\_type=None)

Validate the values of the attributes. If *attribute\_type* is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

# Storage

UnpackStoredSimulationData	Loads a StoredSimulationData object from disk, and
	makes its attributes easily accessible to other protocols.

## UnpackStoredSimulationData

class openff.evaluator.protocols.storage.UnpackStoredSimulationData(protocol\_id)

Loads a StoredSimulationData object from disk, and makes its attributes easily accessible to other protocols.

\_\_init\_\_(protocol\_id)

### **Methods**

init(protocol_id)	
apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
<pre>from_json(file_path)</pre>	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
	structure.
<pre>replace_protocol(old_id, new_id)</pre>	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
<pre>set_value(reference_path, value)</pre>	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
coordinate_file_path	Output - A path to the stored simulation output co-
	ordinates.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
force_field_path	Output - A path to the force field parameters used to
	generate the stored data.
id	The unique id of this protocol.
observables	Output - The stored observables frame.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
simulation_data_path	<b>Input</b> - A list / tuple which contains both the path
	to the simulation data object, it's ancillary data direc-
	tory, and the force field which was used to generate
	the stored data.
substance	Output - The substance which was stored.
thermodynamic_state	Output - The thermodynamic state which was stored.
total_number_of_molecules	Output - The total number of molecules in the stored
	system.
trajectory_file_path	Output - A path to the stored simulation trajectory.

## simulation\_data\_path

**Input** - A list / tuple which contains both the path to the simulation data object, it's ancillary data directory, and the force field which was used to generate the stored data. The default value of this attribute is not set and must be set by the user..

Type Union

#### substance

**Output** - The substance which was stored. The default value of this attribute is not set and must be set by the user..

Type Substance

## total\_number\_of\_molecules

**Output** - The total number of molecules in the stored system. The default value of this attribute is not set and must be set by the user..

Type int

## thermodynamic\_state

**Output** - The thermodynamic state which was stored. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

## observables

**Output** - The stored observables frame. The default value of this attribute is not set and must be set by the user..

Type ObservableFrame

#### coordinate\_file\_path

**Output** - A path to the stored simulation output coordinates. The default value of this attribute is not set and must be set by the user..

Type str

## trajectory\_file\_path

**Output** - A path to the stored simulation trajectory. The default value of this attribute is not set and must be set by the user..

Type str

## force\_field\_path

**Output** - A path to the force field parameters used to generate the stored data. The default value of this attribute is not set and must be set by the user..

Type str

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

**apply\_replicator**(replicator, template\_values, template\_index=- 1, template\_value=None, update\_input\_references=False)

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format \$(replicator.id)).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific *template\_index* or *template\_value* is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

**Return type** dict of ProtocolPath and list of tuple of ProtocolPath and int

```
can_merge(other, path_replacements=None)
```

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

#### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters** schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

#### classmethod get\_attributes(attribute type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

#### Return type object

```
get_value(reference_path)
```

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

#### get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

```
Type str
```

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** Dict[str, str]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

### **Return type** Any

## replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

## set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

Parameters attribute\_type (type of Attribute, optional) — The type of attribute to validate.

Raises ValueError or AssertionError -

### **YANK Free Energies**

BaseYankProtocol	An abstract base class for protocols which will performs
	a set of alchemical free energy simulations using the
	YANK framework.
LigandReceptorYankProtocol	A protocol for performing ligand-receptor alchemical
	free energy calculations using the YANK framework.
SolvationYankProtocol	A protocol for estimating the change in free energy upon
	transferring a solute into a solvent (referred to as solvent
	1) from a second solvent (referred to as solvent 2) by
	performing an alchemical free energy calculation using
	the YANK framework.

## **BaseYankProtocol**

# $\textbf{class} \ \texttt{openff.evaluator.protocols.yank.BaseYankProtocol}(\textit{protocol\_id})$

An abstract base class for protocols which will performs a set of alchemical free energy simulations using the YANK framework.

\_\_init\_\_(protocol\_id)

## **Methods**

apply_replicator       Applies a ProtocolReplicator       to this protocol.         can_merge       Determines whether this protocol can be merged another.         execute       Execute the protocol.         from_json       Create this object from a JSON file.         from_schema       Initializes a protocol from it's schema definition	
another.  execute([directory, available_resources])  from_json(file_path)  from_schema(schema)  Execute the protocol.  Create this object from a JSON file.  Initializes a protocol from it's schema definition	
execute([directory, available_resources])       Execute the protocol.         from_json(file_path)       Create this object from a JSON file.         from_schema(schema)       Initializes a protocol from it's schema definition	
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition	
from_schema(schema)  Initializes a protocol from it's schema definition	
get_attributes([attribute_type]) Returns all attributes of a specific attribute_typ	•
get_class_attribute(reference_path)  Returns one of this protocols, or any of its child	en's,
attributes directly (rather than its value).	
get_value(reference_path)  Returns the value of one of this protocols inputs	out-
puts.	
get_value_references(input_path)  Returns a dictionary of references to the prot	
which one of this protocols inputs (specified by	y in-
<i>put_path</i> ) takes its value from.	
json([file_path, format]) Creates a JSON representation of this class.	
merge(other) Merges another Protocol with this one.	
parse_json(string_contents) Parses a typed json string into the corresponding	class
structure.	
replace_protocol(old_id, new_id) Finds each input which came from a given prot	col
set_uuid(value) Prepend a unique identifier to this protocols id.	
set_value(reference_path, value) Sets the value of one of this protocols inputs.	
validate([attribute_type])Validate the values of the attributes.	

### **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
checkpoint_interval	<b>Input</b> - The number of iterations between saving
	YANK checkpoint files.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
free_energy_difference	Output - The estimated free energy difference be-
	tween the two phases ofinterest.
<pre>gradient_parameters</pre>	<b>Input</b> - An optional list of parameters to differentiate
	the estimated free energy with respect to.
id	The unique id of this protocol.
number_of_equilibration_iterations	<b>Input</b> - The number of iterations used for equilibra-
	tion before production run.
number_of_iterations	<b>Input</b> - The number of YANK iterations to perform.
outputs	A dictionary of the outputs of this property.
required_inputs	The inputs which must be set on this protocol.
schema	A serializable schema for this object.
setup_only	Input - If true, YANK will only create and validate
	the setup files, but not actually run any simulations.
steps_per_iteration	<b>Input</b> - The number of steps per YANK iteration to
	perform.
thermodynamic_state	<b>Input</b> - The state at which to run the calculations.
timestep	<b>Input</b> - The length of the timestep to take.
verbose	<b>Input</b> - Controls whether or not to run YANK at high
	•

## thermodynamic\_state

**Input** - The state at which to run the calculations. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

## number\_of\_equilibration\_iterations

**Input** - The number of iterations used for equilibration before production run. Only post-equilibration iterations are written to file. The default value of this attribute is 1.

Type int

## number\_of\_iterations

**Input** - The number of YANK iterations to perform. The default value of this attribute is 5000.

Type int

## steps\_per\_iteration

Input - The number of steps per YANK iteration to perform. The default value of this attribute is 500.

Type int

## checkpoint\_interval

**Input** - The number of iterations between saving YANK checkpoint files. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 1.

Type int

#### timestep

**Input** - The length of the timestep to take. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 2 femtosecond.

```
Type Quantity
```

#### verbose

Input - Controls whether or not to run YANK at high verbosity. The default value of this attribute is False.

Type bool

#### setup\_only

**Input** - If true, YANK will only create and validate the setup files, but not actually run any simulations. This argument is mainly only to be used for testing purposes. The default value of this attribute is False.

Type bool

## gradient\_parameters

Input - An optional list of parameters to differentiate the estimated free energy with respect to.

Type *list* 

### free\_energy\_difference

**Output** - The estimated free energy difference between the two phases ofinterest. The default value of this attribute is not set and must be set by the user.

Type Observable

## validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

#### **Return type** cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

#### get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

**Parameters reference\_path** (ProtocolPath) – The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** Any

## get\_value\_references(input\_path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

Returns A dictionary of the protocol paths that the input targeted by input\_path depends upon.

Return type dict of ProtocolPath and ProtocolPath

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

## **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

#### property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

## set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The uuid to prepend.

#### set\_value(reference path, value)

Sets the value of one of this protocols inputs.

### **Parameters**

- reference\_path (ProtocolPath) The path pointing to the value to return.
- value (Any) The value to set.

# LigandReceptorYankProtocol

 $\textbf{class} \ open ff. evaluator. protocols. yank. \textbf{LigandReceptorYankProtocol}(\textit{protocol\_id})$ 

 $A\ protocol\ for\ performing\ ligand-receptor\ alchemical\ free\ energy\ calculations\ using\ the\ YANK\ framework.$ 

\_\_init\_\_(protocol\_id)

Constructs a new LigandReceptorYankProtocol object.

## **Methods**

init(protocol_id)	Constructs a new LigandReceptorYankProtocol ob-
/	ject.
<pre>apply_replicator(replicator, template_values)</pre>	Applies a <i>ProtocolReplicator</i> to this protocol.
can_merge(other[, path_replacements])	Determines whether this protocol can be merged with
	another.
<pre>execute([directory, available_resources])</pre>	Execute the protocol.
from_json(file_path)	Create this object from a JSON file.
from_schema(schema)	Initializes a protocol from it's schema definition.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
	attributes directly (rather than its value).
<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
	puts.
<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
	which one of this protocols inputs (specified by in-
	<pre>put_path) takes its value from.</pre>
<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
merge(other)	Merges another Protocol with this one.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
<pre>replace_protocol(old_id, new_id)</pre>	Finds each input which came from a given protocol
set_uuid(value)	Prepend a unique identifier to this protocols id.
set_value(reference_path, value)	Sets the value of one of this protocols inputs.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

## **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
apply_restraints	<b>Input</b> - Determines whether the ligand should be ex-
	plicitly restrained to the receptor in order to stop the
	ligand from temporarily unbinding.
checkpoint_interval	<b>Input</b> - The number of iterations between saving
	YANK checkpoint files.
complex_electrostatic_lambdas	Input - The list of electrostatic alchemical states that
	YANK should sample at when calculating the free
	energy of the ligand in complex with the receptor.
	continues on next page

	ued from previous page
complex_steric_lambdas	Input - The list of steric alchemical states that YANK
	should sample at when calculating the free energy of
	the ligand in complex with the receptor.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
force_field_path	<b>Input</b> - The path to the force field which defines the
	charge method to use for the calculation.
free_energy_difference	Output - The estimated free energy difference be-
	tween the two phases ofinterest.
gradient_parameters	<b>Input</b> - An optional list of parameters to differentiate
	the estimated free energy with respect to.
id	The unique id of this protocol.
ligand_electrostatic_lambdas	<b>Input</b> - The list of electrostatic alchemical states that
	YANK should sample at when calculating the free
	energy of the solvated ligand.
ligand_residue_name	Input - The residue name of the ligand.
ligand_steric_lambdas	Input - The list of steric alchemical states that YANK
	should sample at when calculating the free energy of
	the solvated ligand.
number_of_equilibration_iterations	<b>Input</b> - The number of iterations used for equilibra-
	tion before production run.
number_of_iterations	<b>Input</b> - The number of YANK iterations to perform.
outputs	A dictionary of the outputs of this property.
receptor_residue_name	<b>Input</b> - The residue name of the receptor.
required_inputs	The inputs which must be set on this protocol.
restraint_type	<b>Input</b> - The type of ligand restraint applied, provided
	that <i>apply_restraints</i> is <i>True</i> The default value of this
	attribute is RestraintType.Harmonic.
schema	A serializable schema for this object.
setup_only	<b>Input</b> - If true, YANK will only create and validate
Sccup_only	the setup files, but not actually run any simulations.
solvated_complex_coordinates	Input - The file path to the solvated complex coordi-
SOLVATCA_COMPTEX_COOLATHATES	nates.
solvated_complex_system	Input - The parameterized solvated complex system
301vacca_comprex_3y3ccm	object.
solvated_complex_trajectory_path	Output - The file path to the generated ligand trajec-
Solvated_complex_trajectory_path	
solvated_ligand_coordinates	Input - The file path to the solvated ligand coordi-
Solvated_ligand_coordinates	
solvated_ligand_system	nates.
SUIVALEU_IIYAIIU_SYSLEIII	<b>Input</b> - The parameterized solvated ligand system object
colvered lineard two-in-the	ject.
solvated_ligand_trajectory_path	Output - The file path to the generated ligand trajectory
atana man itanatian	tory.
steps_per_iteration	<b>Input</b> - The number of steps per YANK iteration to
the same demands a state	perform.
thermodynamic_state	Input - The state at which to run the calculations.
timestep	Input - The length of the timestep to take.
verbose	Input - Controls whether or not to run YANK at high
	verhocity

# class RestraintType(value)

The types of ligand restraints available within yank.

2.32. API 597

verbosity.

#### ligand\_residue\_name

**Input** - The residue name of the ligand. The default value of this attribute is not set and must be set by the user..

Type str

### receptor\_residue\_name

**Input** - The residue name of the receptor. The default value of this attribute is not set and must be set by the user..

Type str

## solvated\_ligand\_coordinates

**Input** - The file path to the solvated ligand coordinates. The default value of this attribute is not set and must be set by the user..

Type str

## solvated\_ligand\_system

**Input** - The parameterized solvated ligand system object. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

#### solvated\_complex\_coordinates

**Input** - The file path to the solvated complex coordinates. The default value of this attribute is not set and must be set by the user..

Type str

### solvated\_complex\_system

**Input** - The parameterized solvated complex system object. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

### force\_field\_path

**Input** - The path to the force field which defines the charge method to use for the calculation. The default value of this attribute is not set and must be set by the user.

Type str

### apply\_restraints

**Input** - Determines whether the ligand should be explicitly restrained to the receptor in order to stop the ligand from temporarily unbinding. The default value of this attribute is **True**.

Type bool

### restraint\_type

**Input** - The type of ligand restraint applied, provided that *apply\_restraints* is *True* The default value of this attribute is RestraintType.Harmonic.

Type RestraintType

### ligand\_electrostatic\_lambdas

**Input** - The list of electrostatic alchemical states that YANK should sample at when calculating the free energy of the solvated ligand. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

## ligand\_steric\_lambdas

**Input** - The list of steric alchemical states that YANK should sample at when calculating the free energy of the solvated ligand. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

## complex\_electrostatic\_lambdas

**Input** - The list of electrostatic alchemical states that YANK should sample at when calculating the free energy of the ligand in complex with the receptor. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

## complex\_steric\_lambdas

**Input** - The list of steric alchemical states that YANK should sample at when calculating the free energy of the ligand in complex with the receptor. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

## solvated\_ligand\_trajectory\_path

**Output** - The file path to the generated ligand trajectory. The default value of this attribute is not set and must be set by the user..

Type str

#### solvated\_complex\_trajectory\_path

**Output** - The file path to the generated ligand trajectory. The default value of this attribute is not set and must be set by the user..

Type str

## allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

can\_merge(other, path replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

## checkpoint\_interval

**Input** - The number of iterations between saving YANK checkpoint files. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 1.

Type int

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

**execute**(directory=", available resources=None)

Execute the protocol.

#### **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

#### free\_energy\_difference

**Output** - The estimated free energy difference between the two phases ofinterest. The default value of this attribute is not set and must be set by the user..

Type Observable

## classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

## classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

Parameters schema (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific *attribute\_type*.

**Parameters attribute\_type** (*type of Attribute, optional*) — The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) – The path pointing to the attribute to return.

Returns The class attribute.

Return type object

### get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

**Return type** *Any* 

#### get\_value\_references(input path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *input path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list | dict* which contains at least one ProtocolPath.

Parameters input\_path (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

**Return type** dict of ProtocolPath and ProtocolPath

#### gradient\_parameters

**Input** - An optional list of parameters to differentiate the estimated free energy with respect to.

Type list

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### number\_of\_equilibration\_iterations

**Input** - The number of iterations used for equilibration before production run. Only post-equilibration iterations are written to file. The default value of this attribute is 1.

Type int

#### number\_of\_iterations

Input - The number of YANK iterations to perform. The default value of this attribute is 5000.

Type int

## property outputs

A dictionary of the outputs of this property.

**Type** dict of ProtocolPath and Any

#### classmethod parse\_json(string contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

## property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

## property schema

A serializable schema for this object.

Type ProtocolSchema

### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (str) – The unid to prepend.

#### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

## setup\_only

**Input** - If true, YANK will only create and validate the setup files, but not actually run any simulations. This argument is mainly only to be used for testing purposes. The default value of this attribute is False.

Type bool

### steps\_per\_iteration

**Input** - The number of steps per YANK iteration to perform. The default value of this attribute is 500.

Type int

### thermodynamic\_state

**Input** - The state at which to run the calculations. The default value of this attribute is not set and must be set by the user..

**Type** ThermodynamicState

### timestep

**Input** - The length of the timestep to take. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 2 femtosecond.

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

### verbose

Input - Controls whether or not to run YANK at high verbosity. The default value of this attribute is False.

Type bool

### SolvationYankProtocol

## class openff.evaluator.protocols.yank.SolvationYankProtocol(protocol\_id)

A protocol for estimating the change in free energy upon transferring a solute into a solvent (referred to as solvent 1) from a second solvent (referred to as solvent 2) by performing an alchemical free energy calculation using the YANK framework.

This protocol can be used for box solvation free energies (setting the *solvent\_1* input to the solvent of interest and setting *solvent\_2* as an empty *Substance*) or transfer free energies (setting both the *solvent\_1* and *solvent\_2* inputs to different solvents).

\_\_init\_\_(protocol\_id)

### **Methods**

apply_replicator(replicator, template_values)Applies a ProtocolReplicator to this protocol.can_merge(other[, path_replacements])Determines whether this protocol can be merged with another.execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.replace_protocol(old_id, new_id)Finds each input which came from a given protocol	init(protocol_id)	
another.  execute([directory, available_resources])	apply_replicator(replicator, template_values)	Applies a <i>ProtocolReplicator</i> to this protocol.
execute([directory, available_resources])Execute the protocol.from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.	<pre>can_merge(other[, path_replacements])</pre>	Determines whether this protocol can be merged with
from_json(file_path)Create this object from a JSON file.from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.		another.
from_schema(schema)Initializes a protocol from it's schema definition.get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.	<pre>execute([directory, available_resources])</pre>	Execute the protocol.
get_attributes([attribute_type])Returns all attributes of a specific attribute_type.get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.	<pre>from_json(file_path)</pre>	Create this object from a JSON file.
get_class_attribute(reference_path)Returns one of this protocols, or any of its children's, attributes directly (rather than its value).get_value(reference_path)Returns the value of one of this protocols inputs / outputs.get_value_references(input_path)Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.	from_schema(schema)	Initializes a protocol from it's schema definition.
attributes directly (rather than its value).  get_value(reference_path)  Returns the value of one of this protocols inputs / outputs.  get_value_references(input_path)  Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.  merge(other)  Merges another Protocol with this one.  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.	<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific attribute_type.
get_value(reference_path)       Returns the value of one of this protocols inputs / outputs.         get_value_references(input_path)       Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.         json([file_path, format])       Creates a JSON representation of this class.         merge(other)       Merges another Protocol with this one.         parse_json(string_contents)       Parses a typed json string into the corresponding class structure.	<pre>get_class_attribute(reference_path)</pre>	Returns one of this protocols, or any of its children's,
puts.  get_value_references(input_path)  Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.  merge(other)  Merges another Protocol with this one.  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.		attributes directly (rather than its value).
get_value_references(input_path)       Returns a dictionary of references to the protocols which one of this protocols inputs (specified by input_path) takes its value from.         json([file_path, format])       Creates a JSON representation of this class.         merge(other)       Merges another Protocol with this one.         parse_json(string_contents)       Parses a typed json string into the corresponding class structure.	<pre>get_value(reference_path)</pre>	Returns the value of one of this protocols inputs / out-
which one of this protocols inputs (specified by input_path) takes its value from.  json([file_path, format])  Creates a JSON representation of this class.  merge(other)  Merges another Protocol with this one.  parse_json(string_contents)  Parses a typed json string into the corresponding class structure.		puts.
put_path)takes its value from.json([file_path, format])Creates a JSON representation of this class.merge(other)Merges another Protocol with this one.parse_json(string_contents)Parses a typed json string into the corresponding class structure.	<pre>get_value_references(input_path)</pre>	Returns a dictionary of references to the protocols
json([file_path, format])       Creates a JSON representation of this class.         merge(other)       Merges another Protocol with this one.         parse_json(string_contents)       Parses a typed json string into the corresponding class structure.		1 1 1
merge(other)         Merges another Protocol with this one.           parse_json(string_contents)         Parses a typed json string into the corresponding class structure.		<pre>put_path) takes its value from.</pre>
parse_json(string_contents)  Parses a typed json string into the corresponding class structure.	<pre>json([file_path, format])</pre>	Creates a JSON representation of this class.
structure.	merge(other)	Merges another Protocol with this one.
	<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
replace_protocol(old_id, new_id) Finds each input which came from a given protocol		structure.
	replace_protocol(old_id, new_id)	
set_uuid(value) Prepend a unique identifier to this protocols id.	set_uuid(value)	
set_value(reference_path, value) Sets the value of one of this protocols inputs.	set_value(reference_path, value)	Sets the value of one of this protocols inputs.
validate([attribute_type])Validate the values of the attributes.	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

# **Attributes**

	III of other it.
solvent_1_coordinate_path	first solvent.
column 1 coordinate math	the first solvent.  Output - The file path to the coordinates of only the
solvent_1	<b>Input</b> - The substance describing the composition of
	solute in the second solvent.
solution_2_trajectory_path	Output - The file path to the trajectory containing the
	embedded in the second solvent.
solution_2_system	<b>Input</b> - The parameterized system object of the solute
1	molecule in the second solvent.
	the an ideal solute molecule into a fully interacting
solution_2_free_energy	Output - The free energy change of transforming
and the same of th	embedded in the second solvent.
SOIUTION_2_COORGINATES	<b>Input</b> - The file path to the coordinates of the solute
solution 2 coordinates	
solution_1_trajectory_path	<b>Output</b> - The file path to the trajectory containing the solute in the first solvent.
colution 1 traingtons with	
solution_1_system	Input - The parameterized system object of the solute embedded in the first solvent.
colution 1 custom	molecule in the first solvent.
	the an ideal solute molecule into a fully interacting
solution_1_free_energy	Output - The free energy change of transforming
and the same of th	embedded in the first solvent.
solution_1_coordinates	<b>Input</b> - The file path to the coordinates of the solute
colution 1 coordinates	the solute.
solute	<b>Input</b> - The substance describing the composition of the solute
coluto	
setup_only	the setup files, but not actually run any simulations.
	Input - If true, YANK will only create and validate
schema	A serializable schema for this object.
outputs required_inputs	A dictionary of the outputs of this property.  The inputs which must be set on this protocol.
number_of_iterations	<b>Input</b> - The number of YANK iterations to perform.
number_or_equilibration_iterations	tion before production run.
number_of_equilibration_iterations	<b>Input</b> - The number of iterations used for equilibra-
id	The unique id of this protocol.
grautent_parameters	the estimated free energy with respect to.
gradient_parameters	<b>Input</b> - An optional list of parameters to differentiate
	tween the solute in the first solvent and the second solvent (i.e.
free_energy_difference	Output - The estimated free energy difference be-
6 11.66	YANK should sample at.
electrostatic_lambdas_2	Input - The list of electrostatic alchemical states that
	YANK should sample at.
electrostatic_lambdas_1	<b>Input</b> - The list of electrostatic alchemical states that
	takes input from.
dependencies	A list of pointers to the protocols which this protocol
	YANK checkpoint files.
checkpoint_interval	Input - The number of iterations between saving
allow_merging	merge with other protocols.

Table 4 – continued from previous page

solvent_1_trajectory_path	Output - The file path to the trajectory containing
	only the first solvent.
solvent_2	<b>Input</b> - The substance describing the composition of
	the second solvent.
solvent_2_coordinate_path	Output - The file path to the coordinates of only the
	second solvent.
solvent_2_trajectory_path	Output - The file path to the trajectory containing
	only the second solvent.
steps_per_iteration	<b>Input</b> - The number of steps per YANK iteration to
	perform.
steric_lambdas_1	Input - The list of steric alchemical states that YANK
	should sample at.
steric_lambdas_2	Input - The list of steric alchemical states that YANK
	should sample at.
thermodynamic_state	<b>Input</b> - The state at which to run the calculations.
timestep	<b>Input</b> - The length of the timestep to take.
verbose	Input - Controls whether or not to run YANK at high
	verbosity.

#### solute

**Input** - The substance describing the composition of the solute. This should include the solute molecule as well as any counter ions. The default value of this attribute is not set and must be set by the user..

Type Substance

#### solvent\_1

**Input** - The substance describing the composition of the first solvent. The default value of this attribute is not set and must be set by the user.

Type Substance

#### solvent\_2

**Input** - The substance describing the composition of the second solvent. The default value of this attribute is not set and must be set by the user..

Type Substance

## solution\_1\_coordinates

**Input** - The file path to the coordinates of the solute embedded in the first solvent. The default value of this attribute is not set and must be set by the user..

Type str

#### solution\_1\_system

**Input** - The parameterized system object of the solute embedded in the first solvent. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

## solution\_2\_coordinates

**Input** - The file path to the coordinates of the solute embedded in the second solvent. The default value of this attribute is not set and must be set by the user..

Type str

#### solution\_2\_system

**Input** - The parameterized system object of the solute embedded in the second solvent. The default value of this attribute is not set and must be set by the user..

Type ParameterizedSystem

#### electrostatic\_lambdas\_1

**Input** - The list of electrostatic alchemical states that YANK should sample at. These values will be passed to the YANK *lambda\_electrostatics* option. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

#### steric\_lambdas\_1

**Input** - The list of steric alchemical states that YANK should sample at. These values will be passed to the YANK *lambda\_sterics* option. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

## electrostatic\_lambdas\_2

**Input** - The list of electrostatic alchemical states that YANK should sample at. These values will be passed to the YANK *lambda\_electrostatics* option. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type *list* 

## steric\_lambdas\_2

**Input** - The list of steric alchemical states that YANK should sample at. These values will be passed to the YANK *lambda\_sterics* option. If no option is set, YANK will use *trailblaze* algorithm to determine this option automatically. The default value of this attribute is not set. This attribute is *optional*.

Type list

### solution\_1\_free\_energy

**Output** - The free energy change of transforming the an ideal solute molecule into a fully interacting molecule in the first solvent. The default value of this attribute is not set and must be set by the user..

Type Observable

## solvent\_1\_coordinate\_path

**Output** - The file path to the coordinates of only the first solvent. The default value of this attribute is not set and must be set by the user..

Type str

### solvent\_1\_trajectory\_path

**Output** - The file path to the trajectory containing only the first solvent. The default value of this attribute is not set and must be set by the user..

Type str

#### solution\_1\_trajectory\_path

**Output** - The file path to the trajectory containing the solute in the first solvent. The default value of this attribute is not set and must be set by the user..

Type str

#### solution\_2\_free\_energy

**Output** - The free energy change of transforming the an ideal solute molecule into a fully interacting molecule in the second solvent. The default value of this attribute is not set and must be set by the user..

**Type** Observable

#### solvent\_2\_coordinate\_path

**Output** - The file path to the coordinates of only the second solvent. The default value of this attribute is not set and must be set by the user..

Type str

#### solvent\_2\_trajectory\_path

**Output** - The file path to the trajectory containing only the second solvent. The default value of this attribute is not set and must be set by the user..

Type str

### solution\_2\_trajectory\_path

**Output** - The file path to the trajectory containing the solute in the second solvent. The default value of this attribute is not set and must be set by the user.

Type str

#### free\_energy\_difference

**Output** - The estimated free energy difference between the solute in the first solvent and the second solvent (i.e.  $G = G_1 - G_2$ ). The default value of this attribute is not set and must be set by the user.

Type Observable

#### allow\_merging

**Input** - Defines whether this protocols is allowed to merge with other protocols. The default value of this attribute is True.

Type bool

Applies a *ProtocolReplicator* to this protocol. This method should clone any protocols whose id contains the id of the replicator (in the format *\$(replicator.id)*).

#### **Parameters**

- replicator (ProtocolReplicator) The replicator to apply.
- **template\_values** (*list of Any*) A list of the values which will be inserted into the newly replicated protocols.

This parameter is mutually exclusive with template\_index and template\_value

• **template\_index** (*int*, *optional*) – A specific value which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_value*.

• **template\_value** (*Any*, *optional*) – A specific index which should be used for any protocols flagged as to be replicated by the replicator. This option is mainly used when replicating children of an already replicated protocol.

This parameter is mutually exclusive with *template\_values* and must be set along with a *template\_index*.

• update\_input\_references (bool) – If true, any protocols which take their input from a protocol which was flagged for replication will be updated to take input from the actually replicated protocol. This should only be set to true if this protocol is not nested within a workflow or a protocol group.

This option cannot be used when a specific template\_index or template\_value is providied.

**Returns** A dictionary of references to all of the protocols which have been replicated, with keys of original protocol ids. Each value is comprised of a list of the replicated protocol ids, and their index into the *template\_values* array.

Return type dict of ProtocolPath and list of tuple of ProtocolPath and int

#### can\_merge(other, path\_replacements=None)

Determines whether this protocol can be merged with another.

#### **Parameters**

- **other** (Protocol) The protocol to compare against.
- path\_replacements (list of tuple of str, optional) Replacements to make in any value reference protocol paths before comparing for equality.

**Returns** True if the two protocols are safe to merge.

Return type bool

#### checkpoint\_interval

**Input** - The number of iterations between saving YANK checkpoint files. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 1.

Type int

#### property dependencies

A list of pointers to the protocols which this protocol takes input from.

Type list of ProtocolPath

execute(directory=", available\_resources=None)

Execute the protocol.

## **Parameters**

- **directory** (*str*) The directory to store output data in.
- available\_resources (ComputeResources) The resources available to execute on. If *None*, the protocol will be executed on a single CPU.

### classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

#### classmethod from\_schema(schema)

Initializes a protocol from it's schema definition.

**Parameters schema** (ProtocolSchema) – The schema to initialize the protocol using.

**Returns** The initialized protocol.

Return type cls

# classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## get\_class\_attribute(reference\_path)

Returns one of this protocols, or any of its children's, attributes directly (rather than its value).

Parameters reference\_path (ProtocolPath) - The path pointing to the attribute to return.

**Returns** The class attribute.

Return type object

# get\_value(reference\_path)

Returns the value of one of this protocols inputs / outputs.

Parameters reference\_path (ProtocolPath) - The path pointing to the value to return.

**Returns** The value of the input / output

Return type Any

## get\_value\_references(input path)

Returns a dictionary of references to the protocols which one of this protocols inputs (specified by *in-put\_path*) takes its value from.

#### **Notes**

Currently this method only functions correctly for an input value which is either currently a ProtocolPath, or a *list / dict* which contains at least one ProtocolPath.

**Parameters input\_path** (ProtocolPath) – The input value to check.

**Returns** A dictionary of the protocol paths that the input targeted by *input\_path* depends upon.

Return type dict of ProtocolPath and ProtocolPath

### gradient\_parameters

**Input** - An optional list of parameters to differentiate the estimated free energy with respect to.

Type list

id

The unique id of this protocol. The default value of this attribute is not set and must be set by the user..

Type str

```
json(file_path=None, format=False)
```

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (*bool*) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

#### Return type str

#### merge(other)

Merges another Protocol with this one. The id of this protocol will remain unchanged.

**Parameters other** (Protocol) – The protocol to merge into this one.

**Returns** A map between any original protocol ids and their new merged values.

**Return type** *Dict*[*str*, *str*]

### number\_of\_equilibration\_iterations

**Input** - The number of iterations used for equilibration before production run. Only post-equilibration iterations are written to file. The default value of this attribute is 1.

Type int

## number\_of\_iterations

**Input** - The number of YANK iterations to perform. The default value of this attribute is 5000.

Type int

## property outputs

A dictionary of the outputs of this property.

Type dict of ProtocolPath and Any

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (*str or bytes*) – The typed json string.

**Returns** The parsed class.

**Return type** Any

# replace\_protocol(old\_id, new\_id)

**Finds each input which came from a given protocol** and redirects it to instead take input from a new one.

#### **Notes**

This method is mainly intended to be used only when merging multiple protocols into one.

#### **Parameters**

- **old\_id** (*str*) The id of the old input protocol.
- **new\_id** (*str*) The id of the new input protocol.

# property required\_inputs

The inputs which must be set on this protocol.

Type list of ProtocolPath

# property schema

A serializable schema for this object.

Type ProtocolSchema

#### set\_uuid(value)

Prepend a unique identifier to this protocols id. If the id already has a prepended unid, it will be overwritten by this value.

**Parameters value** (*str*) – The uuid to prepend.

### set\_value(reference\_path, value)

Sets the value of one of this protocols inputs.

#### **Parameters**

- **reference\_path** (ProtocolPath) The path pointing to the value to return.
- **value** (*Any*) The value to set.

### setup\_only

**Input** - If true, YANK will only create and validate the setup files, but not actually run any simulations. This argument is mainly only to be used for testing purposes. The default value of this attribute is False.

Type bool

## steps\_per\_iteration

**Input** - The number of steps per YANK iteration to perform. The default value of this attribute is 500.

Type int

### thermodynamic\_state

**Input** - The state at which to run the calculations. The default value of this attribute is not set and must be set by the user..

Type ThermodynamicState

## timestep

**Input** - The length of the timestep to take. When two protocols are merged, the largest value of this attribute from either protocol is retained. The default value of this attribute is 2 femtosecond.

Type Quantity

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) – The type of attribute to validate.

Raises ValueError or AssertionError -

#### verbose

Input - Controls whether or not to run YANK at high verbosity. The default value of this attribute is False.

Type bool

# 2.32.11 Workflow Construction Utilities

SimulationProtocols	The common set of protocols which would be required to estimate an observable by running a new molecule simulation.
ReweightingProtocols	The common set of protocols which would be required to re-weight an observable from cached simulation data.
<pre>generate_base_reweighting_protocols</pre>	Constructs a set of protocols which, when combined in a workflow schema, may be executed to reweight a set of cached simulation data to estimate the average value of an observable.
generate_reweighting_protocols	
generate_simulation_protocols	Constructs a set of protocols which, when combined in a workflow schema, may be executed to run a single simulation to estimate the average value of an observable.

## **SimulationProtocols**

 $\textbf{class} \ open ff. evaluator. protocols. utils. \textbf{SimulationProtocols} (\textit{build\_coordinates:}$ 

BuildCoordinatesPackmol, assign\_parameters: BaseBuildSystem, energy\_minimisation:
OpenMMEnergyMinimisation, equilibration\_simulation:
OpenMMSimulation, production\_simulation:
OpenMMSimulation,
analysis\_protocol: S,
converge\_uncertainty: ProtocolGroup, decorrelate\_trajectory:
DecorrelateTrajectory, decorrelate\_observables:
DecorrelateObservables)

The common set of protocols which would be required to estimate an observable by running a new molecule simulation.

\_\_init\_\_(build\_coordinates: BuildCoordinatesPackmol, assign\_parameters: BaseBuildSystem, energy\_minimisation: OpenMMEnergyMinimisation, equilibration\_simulation: OpenMMSimulation, production\_simulation: OpenMMSimulation, analysis\_protocol: S, converge\_uncertainty: ProtocolGroup, decorrelate\_trajectory: DecorrelateTrajectory, decorrelate\_observables: DecorrelateObservables) 

None

# **OpenFF Evaluator Documentation**

## **Methods**

\_\_init\_\_(build\_coordinates, ...)

## **Attributes**

build\_coordinates

assign\_parameters

energy\_minimisation

equilibration\_simulation

production\_simulation

analysis\_protocol

converge\_uncertainty

decorrelate\_trajectory

decorrelate\_observables

# ReweightingProtocols

class openff.evaluator.protocols.utils.ReweightingProtocols(unpack\_stored\_data:

UnpackStoredSimulationData, join trajectories: ConcatenateTrajectories, join observables: ConcatenateObservables, build reference system: BaseBuildSystem, evaluate\_reference\_potential: BaseEvaluateEnergies, build\_target\_system: BaseBuildSystem, evaluate\_target\_potential: BaseEvaluateEnergies, statistical\_inefficiency: S, replicate\_statistics: DummyProtocol, decorrelate\_reference\_potential: DecorrelateObservables. decorrelate\_target\_potential: DecorrelateObservables, decorrelate\_observable: DecorrelateObservables, zero\_gradients: Optional[ZeroGradients], reweight\_observable: T)

The common set of protocols which would be required to re-weight an observable from cached simulation data.

\_\_init\_\_(unpack\_stored\_data: UnpackStoredSimulationData, join\_trajectories: ConcatenateTrajectories, join\_observables: ConcatenateObservables, build\_reference\_system: BaseBuildSystem, evaluate\_reference\_potential: BaseEvaluateEnergies, build\_target\_system: BaseBuildSystem, evaluate\_target\_potential: BaseEvaluateEnergies, statistical\_inefficiency: S, replicate\_statistics: DummyProtocol, decorrelate\_reference\_potential: DecorrelateObservables, decorrelate\_target\_potential: DecorrelateObservables, decorrelate\_observable: DecorrelateObservables, zero\_gradients: Optional[ZeroGradients], reweight\_observable: T) → None

#### **Methods**

\_\_init\_\_(unpack\_stored\_data, ...)

#### **Attributes**

```
unpack_stored_data

join_trajectories

join_observables

build_reference_system

evaluate_reference_potential

build_target_system

evaluate_target_potential

statistical_inefficiency

replicate_statistics

decorrelate_reference_potential

decorrelate_target_potential

decorrelate_target_potential

decorrelate_target_potential

reweight_observable
```

## generate\_base\_reweighting\_protocols

```
openff.evaluator.protocols.utils.generate_base_reweighting_protocols(statistical_inefficiency: S, reweight_observable: T, replicator_id: str = 'data\_replicator', id_suffix: str = ") \rightarrow Tu-ple[ReweightingProtocols[S, T], ProtocolReplicator]
```

Constructs a set of protocols which, when combined in a workflow schema, may be executed to reweight a set of cached simulation data to estimate the average value of an observable.

#### **Parameters**

- **statistical\_inefficiency** The protocol which will be used to compute the statistical inefficiency and equilibration time of the observable of interest. This information will be used to decorrelate the cached data prior to reweighting.
- reweight\_observable The MBAR reweighting protocol to use to reweight the observable to the target state. This method will automatically set the reduced potentials on the object.

- **replicator\_id** (*str*) The id to use for the cached data replicator.
- id\_suffix (str) A string suffix to append to each of the protocol ids.

#### Returns

- The protocols to add to the workflow, a reference to the average value of the
- estimated observable (an Observable object), and the replicator which will
- clone the workflow for each piece of cached simulation data.

# generate reweighting\_protocols

openff.evaluator.protocols.utils.generate\_reweighting\_protocols(observable\_type:

```
ObservableType, replicator_id:

str = 'data\_replicator', id\_suffix:

str = ") \rightarrow Tu-

ple[ReweightingProtocols[AverageObservable, ReweightObservable],

ProtocolReplicator]
```

## generate\_simulation\_protocols

```
openff.evaluator.protocols.utils.generate_simulation_protocols (analysis_protocol: S, use_target_uncertainty: bool, id_suffix: str = ", conditional_group: Optional[ConditionalGroup] = None, n_molecules: int = 1000) \rightarrow Tuple[SimulationProtocols[S], ProtocolPath, StoredSimulationData]
```

Constructs a set of protocols which, when combined in a workflow schema, may be executed to run a single simulation to estimate the average value of an observable.

The protocols returned will:

- 1) Build a set of liquid coordinates for the property substance using packmol.
- 2) Assign a set of smirnoff force field parameters to the system.
- 3) Perform an energy minimisation on the system.
- 4) Run a short NPT equilibration simulation for 100000 steps using a timestep of 2fs.
- 5) Within a conditional group (up to a maximum of 100 times):
  - 5a) Run a longer NPT production simulation for 1000000 steps using a timestep of 2fs
  - 5b) Extract the average value of an observable and it's uncertainty.
  - **5c)** If a convergence mode is set by the options, check if the target uncertainty has been met. If not, repeat steps 5a), 5b) and 5c).
- 6) Extract uncorrelated configurations from a generated production simulation.
- 7) Extract uncorrelated statistics from a generated production simulation.

#### **Parameters**

- **analysis\_protocol** The protocol which will extract the observable of interest from the generated simulation data.
- **use\_target\_uncertainty** Whether to run the simulation until the observable is estimated to within the target uncertainty.
- id\_suffix (str) A string suffix to append to each of the protocol ids.
- **conditional\_group** (ProtocolGroup, *optional*) A custom group to wrap the main simulation / extraction protocols within. It is up to the caller of this method to manually add the convergence conditions to this group. If *None*, a default group with uncertainty convergence conditions is automatically constructed.
- **n\_molecules** (*int*) The number of molecules to use in the workflow.

#### Returns

- The protocols to add to the workflow, a reference to the average value of the
- estimated observable (an Observable object), and an object which describes
- the default data from a simulation to store, such as the uncorrelated statistics
- and configurations.

## 2.32.12 Attribute Utilities

Attribute	A custom descriptor used to add useful metadata to class
	attributes.
AttributeClass	A base class for objects which require well defined at-
	tributes with additional metadata.
UNDEFINED	A custom type used to differentiate between None val-
	ues, and an undeclared optional value.
PlaceholderValue	A class to act as a place holder for an attribute whose
	value is not known a priori, but will be set later by some
	specialised code.

#### **Attribute**

A custom descriptor used to add useful metadata to class attributes.

This decorator expects the object to have a matching private field in addition to the public attribute. For example if an object has an attribute *substance*, the object must also have a *\_substance* field.

### **Notes**

The attribute class will automatically create this private attribute on the object and populate it with the default value.

\_\_init\_\_(docstring, type\_hint, default\_value=UNDEFINED, optional=False, read\_only=False)
Initializes a new Attribute object.

#### **Parameters**

- **docstring** (*str*) A docstring describing the attributes purpose. This will automatically be decorated with additional information such as type hints, default values, etc.
- **type\_hint** (*type*, *Union*) The expected type of this attribute. This will be used to help the workflow engine ensure that expected input types match corresponding output values.
- **default\_value** (*Any*) The default value for this attribute.
- **optional** (*bool*) Defines whether this is an optional input of a class. If true, the *default\_value* should be set to *UNDEFINED*.
- **read\_only** (*bool*) Defines whether this attribute is read-only.

### **Methods**

init(docstring, type_hint[,])	Initializes a new Attribute object.

## **AttributeClass**

## class openff.evaluator.attributes.AttributeClass

A base class for objects which require well defined attributes with additional metadata.

\_\_init\_\_()

#### **Methods**

init()	
from_json(file_path)	Create this object from a JSON file.
<pre>get_attributes([attribute_type])</pre>	Returns all attributes of a specific <i>attribute_type</i> .
json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents)	Parses a typed json string into the corresponding class
	structure.
validate([attribute_type])	Validate the values of the attributes.

### validate(attribute\_type=None)

Validate the values of the attributes. If attribute\_type is set, only attributes of that type will be validated.

**Parameters attribute\_type** (*type of Attribute*, *optional*) — The type of attribute to validate.

Raises ValueError or AssertionError -

## classmethod get\_attributes(attribute\_type=None)

Returns all attributes of a specific attribute\_type.

**Parameters attribute\_type** (*type of Attribute, optional*) – The type of attribute to search for.

**Returns** The names of the attributes of the specified type.

Return type list of str

## classmethod parse\_json(string\_contents)

Parses a typed json string into the corresponding class structure.

**Parameters string\_contents** (str or bytes) – The typed json string.

**Returns** The parsed class.

Return type Any

# classmethod from\_json(file\_path)

Create this object from a JSON file.

**Parameters file\_path** (*str*) – The path to load the JSON from.

**Returns** The parsed class.

Return type cls

json(file\_path=None, format=False)

Creates a JSON representation of this class.

#### **Parameters**

- **file\_path** (*str*, *optional*) The (optional) file path to save the JSON file to.
- **format** (bool) Whether to format the JSON or not.

**Returns** The JSON representation of this class.

Return type str

#### **UNDEFINED**

```
openff.evaluator.attributes.UNDEFINED =
<openff.evaluator.attributes.attributes.UndefinedAttribute object>
```

A custom type used to differentiate between None values, and an undeclared optional value.

#### **PlaceholderValue**

## class openff.evaluator.attributes.PlaceholderValue

A class to act as a place holder for an attribute whose value is not known a priori, but will be set later by some specialised code. This may include the input to a protocol which will be set by a workflow as the output of an executed protocol.

```
__init__()
```

## **Methods**

\_\_init\_\_()

# 2.32.13 Observable Utilities

Observable	A class which stores the mean value of an observable as
	well as the standard error in the mean.
ObservableArray	A class which stores the value(s) of an observable ob-
	tained via molecule simulation (or simulation data) as
	well as optionally the derivatives of the value with re-
	spect to certain force field parameters.
ObservableType	An enumeration of the common observables which may
	be extracted from molecular simulations (or simulation
	data) and stored in an ObservableFrame.
ObservableFrame	A data object for storing and retrieving frames of
	the thermodynamic observables enumerated by the
	ObservableType enum.
bootstrap	Bootstrapping a set of observables to compute the aver-
	age value of the observables as well as the the standard
	error in the average.

## **Observable**

class openff.evaluator.utils.observables.Observable(value: Optional[Union[Measurement, Quantity]] = None, gradients: Optional[List[ParameterGradient]] = None)

A class which stores the mean value of an observable as well as the standard error in the mean. Optionally, the derivatives of the mean with respect to certain force field parameters may also be stored.

\_\_init\_\_(value: Optional[Union[Measurement, Quantity]] = None, gradients: Optional[List[ParameterGradient]] = None)

## **Methods**

\_\_init\_\_([value, gradients])

clear\_gradients()

Clears all gradient information.

### **Attributes**

error		
gradients		
value		

### clear\_gradients()

Clears all gradient information.

# **ObservableArray**

A class which stores the value(s) of an observable obtained via molecule simulation (or simulation data) as well as optionally the derivatives of the value with respect to certain force field parameters.

\_\_init\_\_(value: Optional[Quantity] = None, gradients: Optional[List[ParameterGradient]] = None)

### **Methods**

init([value, gradients])	
clear_gradients()	Clears all gradient information.
join(*observables)	Concatenates multiple observables together in the or-
	der that they appear in the args list.
subset(indices)	Extracts the subset of the values stored for this ob-
	servable at the specified indices.

## **Attributes**

gradients		
value	The value(s) of the observable.	

## property value: Quantity

The value(s) of the observable.

 $subset(indices: Iterable[int]) \rightarrow ObservableArray$ 

Extracts the subset of the values stored for this observable at the specified indices.

**Parameters indices** – The indices of the entries to extract.

**Return type** The subset of the observable values.

```
classmethod join(*observables: ObservableArray) → ObservableArray
```

Concatenates multiple observables together in the order that they appear in the args list.

**Parameters observables** – The observables to join.

**Return type** The concatenated observable object.

### clear\_gradients()

Clears all gradient information.

# ObservableType

## class openff.evaluator.utils.observables.ObservableType(value)

An enumeration of the common observables which may be extracted from molecular simulations (or simulation data) and stored in an ObservableFrame.

\_\_init\_\_()

#### **Attributes**

PotentialEnergy
KineticEnergy
TotalEnergy
Temperature
Volume
Density
Enthalpy
ReducedPotential

#### **ObservableFrame**

```
\begin{tabular}{ll} \textbf{class} & \textbf{openff.evaluator.utils.observables.ObservableFrame} (observables: Optional[Dict[Union[str, ObservableType], ObservableArray]] = \\ & None) \end{tabular}
```

A data object for storing and retrieving frames of the thermodynamic observables enumerated by the ObservableType enum.

\_\_init\_\_(observables: Optional[Dict[Union[str, ObservableType], ObservableArray]] = None)

### **Methods**

init([observables])	
clear()	
clear_gradients()	Clears all gradient information for each observable in the frame.
<pre>from_openmm(file_path[, pressure])</pre>	Creates an observable frame from the CSV output of an OpenMM simulation.
<pre>get(k[,d])</pre>	
items()	
join(*observable_frames)	Joins multiple observable frames together in the order that they appear in the args list.
keys()	7 11
pop(k[,d])	If key is not found, d is returned if given, otherwise KeyError is raised.
popitem()	as a 2-tuple; but raise KeyError if D is empty.
setdefault(k[,d])	
subset(indices)	Extracts the subset of the the array which is located at the specified indices.
update([E, ]**F)	If E present and has a .keys() method, does: for k in E: D[k] = E[k] If E present and lacks .keys() method, does: for (k, v) in E: D[k] = v In either case, this is followed by: for k, v in F.items(): D[k] = v
values()	-

**classmethod from\_openum** ( $file\_path$ : str, pressure: Optional[Quantity] = None)  $\rightarrow ObservableFrame$  Creates an observable frame from the CSV output of an OpenMM simulation.

### **Parameters**

- **file\_path** The file path to the CSV file.
- **pressure** The pressure at which the observables in the csv file were collected.

**Return type** The imported observables.

 $subset(indices: Iterable[int]) \rightarrow ObservableFrame$ 

Extracts the subset of the the array which is located at the specified indices.

**Parameters indices** – The indices of the entries to extract.

Return type The subset of data.

classmethod join(\*observable\_frames: ObservableFrame)  $\rightarrow$  ObservableFrame

Joins multiple observable frames together in the order that they appear in the args list.

**Parameters observable\_frames** – The observable frames to join.

Return type The joined observable frame.

```
clear_gradients()
```

Clears all gradient information for each observable in the frame.

**clear()**  $\rightarrow$  None. Remove all items from D.

 $get(k[,d]) \rightarrow D[k]$  if k in D, else d. d defaults to None.

**items()**  $\rightarrow$  a set-like object providing a view on D's items

**keys()**  $\rightarrow$  a set-like object providing a view on D's keys

 $pop(k[,d]) \rightarrow v$ , remove specified key and return the corresponding value.

If key is not found, d is returned if given, otherwise KeyError is raised.

**popitem()**  $\rightarrow$  (k, v), remove and return some (key, value) pair

as a 2-tuple; but raise KeyError if D is empty.

**setdefault**(k[, d])  $\rightarrow$  D.get(k,d), also set D[k]=d if k not in D

**update**([E], \*\*F)  $\rightarrow$  None. Update D from mapping/iterable E and F.

If E present and has a .keys() method, does: for k in E: D[k] = E[k] If E present and lacks .keys() method, does: for (k, v) in E: D[k] = v In either case, this is followed by: for k, v in F.items(): D[k] = v

**values()**  $\rightarrow$  an object providing a view on D's values

## bootstrap

```
openff.evaluator.utils.observables.bootstrap(bootstrap_function: Callable, iterations: int = 200, relative_sample_size: float = 1.0, sub_counts: Optional[Iterable[int]] = None, **observables: ObservableArray) \rightarrow Observable
```

Bootstrapping a set of observables to compute the average value of the observables as well as the the standard error in the average.

#### **Parameters**

- **bootstrap\_function** The function to evaluate at each bootstrap iteration.
- **iterations** The number of bootstrap iterations to perform.
- **relative\_sample\_size** The percentage sample size to bootstrap over, relative to the size of the full data set.
- **sub\_counts** If the data being bootstrapped contains arrays of concatenated sub data (such as when reweighting), this variable can be used to specify the number of items which belong to each subset. Data is then sampled with replacement so that the bootstrap sample contains the correct proportion of data from each subset.

If the data to bootstrap is of the form [x0, x1, x2, y0, y1] for example, then  $data\_sub\_counts=[3, 2]$  and a possible sample may look like [x0, x0, x2, y0, y0], but never [x0, x1, y0, y1, y1].

The sub-counts must sum up to the total length of the data provided to observables.

observables – The observables which will be passed to the bootstrap function. All observables must have the same length.

**Return type** The average of the data and the uncertainty in the average.

# 2.32.14 Plug-in Utilities

## **Plug-ins**

register_default_plugins	Registers the built-in workflow protocols, calculation
	layers and physical properties with the plugin system.
register_external_plugins	Registers any supported plugins found in external pack-
	ages with the plugin system.

## register\_default\_plugins

openff.evaluator.plugins.register\_default\_plugins()

Registers the built-in workflow protocols, calculation layers and physical properties with the plugin system.

## register\_external\_plugins

openff.evaluator.plugins.register\_external\_plugins()

Registers any supported plugins found in external packages with the plugin system.

# 2.33 Release History

Releases follow the major.minor.micro scheme recommended by PEP440, where

- major increments denote a change that may break API compatibility with previous major releases
- minor increments add features but do not break API compatibility
- micro increments represent bugfix releases or improvements in documentation

# 2.33.1 0.4.3

This release adds support for OpenMM 8. Support for OpenMM 7 will be removed in a future release.

## **Bugfixes**

- PR #496: Adds support for OpenMM 8.
- PR #503: Updates FilterBySmirks to no longer ignore isotopes.

#### **Documentation**

• PR #499: Updates tutorials for upstream changes.

# 2.33.2 0.4.2

### **New Features**

• PR #479: Improves compatibility with *pymbar* 3 and 4.

## **Bugfixes**

• PR #489: Properly creates Amber-style systems involving multiple molecules.

## **Behavior Changes**

• PR #490: Deprecates Yank protocols.

### 2.33.3 0.4.1

#### **New Features**

• PR #475: Tests compatibility with pymbar 3 and 4.

## 2.33.4 0.4.0

## **Behavior Changes**

This release is intended to be compatible with OpenFF Toolkit version 0.11.0 and newer and is not compatible with older versions. To use OpenFF Evaluator in environments with older versions of the toolkit (0.10.x and older) please use the 0.3.x release line.

The simtk namespace is no longer supported. It is recommended to use OpenMM 7.6 or newer.

Two other version constraints have been added: pymbar >=4.0.0 and mdtraj =1.9.3,1.9.4. A future release should fix compatibility with newer versions of MDTraj.

Unit-bearing quantities are now handled by openff-units instead of the openmm.units units module. See the openff-units Documentation for more information, including OpenMM interoperability.

The use of CMILES has been replaced with equivalent behavior using the OpenFF Toolkit as CMILES is no longer actively maintained.

#### **New Features**

• PR #465: Adds support for AMD GPUs via OpenCL.

### **Documentation**

- PR #409: Replaces some uses of *Behaviour* with the American spelling *Behaviour*.
- PR #413: Adds a brief citation recommendation.

# 2.33.5 0.3.11

# **Bugfixes**

• PR #445: Fix OpenMM unit utils API regression

### 2.33.6 0.3.10

## **Bugfixes**

• PR #444: Fix labelling molecules with virtual sites

## 2.33.7 0.3.9

## **Bugfixes**

• PR #402: Fix importing full ThermoML archive

## **Behavior Changes**

The way that ThermoML archive files are served was changed in 2021 so that individual journal archives are no longer made available. Instead, now only the full ThermoML archive can be downloaded. Because of this, the ImportThermoMLDataSchema schema no longer allows users to select which journal to pull data from.

## 2.33.8 0.3.8

# **Bugfixes**

• PR #390: Fix excluding v-sites from OpenMM positions

# 2.33.9 0.3.7

## **Bugfixes**

• PR #389: Fix v-site positions not set by OpenMM

# 2.33.10 0.3.6

# **Bugfixes**

- PR #375: Fix #374 import from collections.abc
- PR #379: Fix #378 'FilterDuplicates' unintentionally selects values without uncertainty if multiple are present
- PR #384: Fix #382 Default keyword arguments result in error
- PR #387: Fix #380 Recursion error in local file storage

#### **New Features**

- PR #385: Support custom OpenMM nonbonded forces
- PR #386: Migrate to new OpenMM namespace

## 2.33.11 0.3.5

## **Bugfixes**

- PR #367: Fix #365 to/from\_pandas does not roundtrip.
- PR #368: Fix #364 Parsing an invalid IUPAC name raises an exception rather than a warning.
- PR #371: Fix gradients of non-Quantity parameters.

### **New Features**

- PR #362: Support dask-jobqueue Slurm backend.
- PR #366: Support gradients of handler attributes.

## 2.33.12 0.3.4

A patch release which adds the option (and enables it by default) to remove working files, such as simulated trajectories, when they are no longer needed.

## **Behavior Changes**

• PR #349: Working files are deleted by default after an estimation batch completes.

## 2.33.13 0.3.3

This release facilitates the migration of the *openff-evaluator* package from *omnia* to *conda-forge*. This mainly involves changes which update the package to use the new namespaces introduced in the *openff-tookit* package, rather than the old and now deprecated *openforcefield* namespaces.

## **Bugfixes**

• PR #346: Remove the unsupported *encoding* json kwarg.

#### **New Features**

- PR #341: Replace usages of dynamic Pint classes with internal static variants.
- PR #343: Migrate to the new OpenFF Toolkit namespace.
- PR #345: Migrate all reference from omnia to conda-forge.

## 2.33.14 0.3.2

This release exposes the option to disable caching of simulation data by an evaluator server. The performance of the local storage backend is currently poor when dealing with large amounts of cached data and hence it may be preferable to disable caching in such cases.

#### **New Features**

• PR #337: Expose server option to dis/enable data caching.

## 2.33.15 0.3.1

This release fixes a bug introduced in version 0.3.0 of this framework, whereby the default workflows for computing excess properties could in rare cases be incorrectly merged leading to downstream protocols taking their inputs from the wrong upstream protocol outputs.

While this bug should not affect most calculations, it is recommended that any production calculations performed using version 0.3.0 of this framework be repeated using version 0.3.1.

### **Bugfixes**

• PR #331: Fixes merging excess properties.

## 2.33.16 0.3.0

The main feature of this release is the overhauling of how the framework computes the gradients of observables with respect to force field parameters.

In particular, from this release onwards all gradients will be computed using the fluctuation formula (also referred to as the thermodynamic gradient), rather than calculation be the re-weighted finite difference approach (PR #280). In general the two methods produce gradients which are numerically indistinguishable, and so this should not markedly change any scientific output of this framework.

The change was made to, in future, enable better integration with automatic differentiation libraries such as jax, and differentiable simulation engines such as timemachine which readily and rapidly give access to  $dU/d\theta_i$ .

**Additionally**, as of version 0.3.0 'known' charges (i.e. those assigned to TIP3P water and ions) are no longer automatically applied when using a SMIRNOFF based force field. This feature was originally included in the framework as the OpenFF toolkit did not support defining charges on specific molecules in the force field itself. This is now fully supported through the LibraryCharges section of a SMIRNOFF force field and hence this workaround is no longer required. From now on all ion and water charges **must** be specified in the SMIRNOFF force field.

Finally, this release includes **beta** support for computing host-guest binding affinities using the attach-pull-release (APR) method through integration with the pAPRika and taproom packages. This support was largely facilitated by the efforts of the paprika authors - David R. Slochower and Jeffry Setiadi.

## **Bugfixes**

- PR #285: Use merged protocols in workflow provenance.
- PR #287: Fix merging of nested protocol inputs

#### **New Features**

- PR #262: Initial host-guest binding affinity support via paprika and taproom.
- PR #280: Switch to computing thermodynamic gradients.
- PR #309: Add a date to the timestamp logging output.
- PR #311: Initial solvation free energy gradient support.
- PR #312: Support caching free energy data.
- PR #324: Adds new miscellaneous DummyProtocol protocol.

### **Behavior Changes**

- PR #280: Migrate to thermodynamic gradients.
- PR #310: The SMIRNOFF protocol no longer applies 'known' charges (i.e. water and ions).
- PR #316: Add library charges to the TIP3P test data file.
- PR #328: Store workflow provenance as serialized string.

## **Breaking Changes**

- The StatisticsArray array has been completely removed and replaced with a new set of observable (Observable, ObservableArray, ObservableFrame objects (#279, #286).
- The following protocol inputs / outputs have been renamed:
  - SolvationYankProtocol.solvent\_X\_system -> SolvationYankProtocol.solution\_X\_system
  - SolvationYankProtocol.solvent\_X\_coordinates -> SolvationYankProtocol. solution\_X\_coordinates
  - SolvationYankProtocol.estimated\_free\_energy -> SolvationYankProtocol.
    free\_energy\_difference
- The following classes have been renamed:
  - OpenMMReducedPotentials -> OpenMMEvaluateEnergies.
  - AveragePropertyProtocol -> BaseAverageObservable, ExtractAverageStatistic
     -> AverageObservable, ExtractUncorrelatedData -> BaseDecorrelateProtocol,
     ExtractUncorrelatedTrajectoryData -> DecorrelateTrajectory,
     ExtractUncorrelatedStatisticsData -> DecorrelateObservables
  - ConcatenateStatistics -> ConcatenateObservables, BaseReducedPotentials -> BaseEvaluateEnergies, ReweightStatistics -> ReweightObservable
- The following classes have been removed:
  - OpenMMGradientPotentials, BaseGradientPotentials, CentralDifferenceGradient
- The final value estimated by a workflow must now be an Observable object which contains any gradient information to return. (#296).

# 2.33.17 0.2.2

This release adds documentation for how physical properties are computed within the framework (both for this, and for previous releases.

## **Documentation**

• PR #281: Initial pass at physical property documentation.

### 2.33.18 0.2.1

A patch release offering minor bug fixes and quality of life improvements.

# **Bugfixes**

- PR #259: Adds is\_file\_and\_not\_empty and addresses OpenMM failure modes.
- PR #275: Workaround for N substance molecules > user specified maximum.

#### **New Features**

- PR #267: Adds workflow protocol to Boltzmann average free energies.
- PR #269: Expose exclude exact amount from max molecule cap.

## 2.33.19 0.2.0

This release overhauls the frameworks data curation abilities. In particular, it adds

a significant amount of data filters, including to filter by state, substance composition and chemical functionalities.

and components to

- easily import all of the ThermoML and FreeSolv archives.
- convert between property types (currently density <-> excess molar volume).
- select data points close to a set of target states, and substances which contain specific functionalities (i.e. select only data points measured for ketones, alcohols or alkanes).

More information about the new curation abilities can be found in the documentation here.

### **New Features**

- PR #260: Data set curation overhaul.
- PR #261: Adds PhysicalPropertyDataSet.from\_pandas.

## **Breaking Changes**

• All of the PhysicalPropertyDataSet.filter\_by\_XXX functions have now been removed in favor of the new curation components. See the *documentation* for information about the newly available filters and more.

## 2.33.20 0.1.2

A patch release offering minor bug fixes and quality of life improvements.

## **Bugfixes**

- PR #254: Fix incompatible protocols being merged due to an id replacement bug.
- PR #255: Fix recursive ThermodynamicState string representation.
- PR #256: Fix incorrect version when installing from tarballs.

## 2.33.21 0.1.1

A patch release offering minor bug fixes and quality of life improvements.

## **Bugfixes**

- PR #249: Fix replacing protocols of non-existent workflow schema.
- PR #253: Fix antechamber truncating charge file.

### **Documentation**

• PR #252: Use *conda-forge* for *ambertools* installation.

# 2.33.22 0.1.0 - OpenFF Evaluator

Introducing the OpenFF Evaluator! The release marks a significant milestone in the development of this project, and constitutes an almost full redesign of the framework with a focus on stability and ease of use.

**Note:** because of the extensive changes made throughout the entire framework, this release should almost be considered as an entirely new package. No files produced by previous versions of this will work with this new release.

## **Clearer Branding**

First and foremost, this release marks the complete rebranding from the previously named *propertyestimator* to the new *openff-evaluator* package. This change is accompanied by the introduction of a new openff namespace for the package, signifying it's position in the larger Open Force Field infrastructure and piplelines.

What was previously:

```
import propertyestimator
```

now becomes:

```
import openff.evaluator
```

The rebranded package is now shipped on conda under the new name of openff-evaluator:

```
conda install -c conda-forge -c omnia openff-evaluator
```

## **Markedly Improved Documentation**

In addition, the release includes for the first time a significant amount of documentation for using the `framework and it's features` as well as a collection of user focused tutorials which can be ran directly in the browser.

## **Support for RDKit**

This release almost entirely removes the dependence on OpenEye thanks to support for RDKit almost universally across the framework.

The only remaining instance where OpenEye is still required is for host-guest binding affinity calculations where it is used to perform docking.

#### **Model Validation**

Starting with this release almost all models, range from PhysicalProperty entries to ProtocolSchema objects, are now heavily validated to help catch any typos or errors early on.

# **Batching of Similar Properties**

The EvaluatorServer now more intelligently attempts to batch properties which may be computed using the same simulations into a single batch to be estimated. While the behaviour was already supported for pure properties in previous, this has now been significantly expanded to work well with mixture properties.

# 2.33.23 0.0.9 - Multi-state Reweighting Fix

This release implements a fix for calculating the gradients of properties being estimated by reweighting data cached from multiple independant simulations.

## **Bugfixes**

• PR #143: Fix for multi-state gradient calculations.

# 2.33.24 0.0.8 - ThermoML Improvements

This release is centered around cleaning up the ThermoML data set utilities. The main change is that ThermoML archive files can now be loaded even if they don't contain measurement uncertainties.

## **New Features**

• PR #142: ThermoML archives without uncertainties can now be loaded.

## **Breaking Changes**

• PR #142: All *ThermoMLXXX* classes other than *ThermoMLDataSet* are now private.

# 2.33.25 0.0.7 - Bug Quick Fixes

This release aims to fix a number of minor bugs.

## **Bugfixes**

- PR #136: Fix for comparing thermodynamic states with unset pressures.
- PR #138: Fix for a typo in the maximum number of minimization iterations.

# 2.33.26 0.0.6 - Solvation Free Energies

This release centers around two key changes -

- i) a general refactoring of the protocol classes to be much cleaner and extensible through the removal of the old stub functions and the addition of cleaner descriptors.
- ii) the addition of workflows to estimate solvation free energies via the new SolvationYankProtocol and SolvationFreeEnergy classes.

The implemented free energy workflow is still rather basic, and does not yet support calculating parameter gradients or estimation from cached simulation data through reweighting.

A new table has been added to the documentation to make clear which built-in properties support which features.

#### **New Features**

- PR #110: Cleanup and refactor of protocol classes.
- PR #125: Support for PBS based HPC clusters.
- PR #127: Adds a basic workflow for estimating solvation free energies with YANK.
- PR #130: Adds a cleaner mechanism for restarting simulations from checkpoints.
- PR #134: Update to a more stable dask version.

## **Bugfixes**

- PR #128: Removed the defunct dask backend processes kwarg.
- PR #133: Fix for tests failing on MacOS due to travis issues.

## **Breaking Changes**

• PR #130: The RunOpenMMSimulation.steps input has now been split into the steps\_per\_iteration and total\_number\_of\_iterations inputs.

## **Migration Guide**

This release contained several public API breaking changes. For the most part, these can be remedied by the follow steps:

• Replace all instances of run\_openmm\_simulation\_protocol.steps to run\_openmm\_simulation\_protocol.steps\_per\_iteration

# 2.33.27 0.0.5 - Fix For Merging of Estimation Requests

This release implements a fix for a major bug which caused incorrect results to be returned when submitting multiple estimation requests at the same time - namely, the returned results became jumbled between the different requests. As an example, if a request was made to estimate a data set using the *smirnoff99frosst* force field, and then straight after with the *gaff 1.81* force field, the results of the *smirnoff99frosst* request may contain some properties estimated with *gaff 1.81* and vice versa.

This issue does not affect cases where only a single request was made and completed at a time (i.e the results of the previous request completed before the next estimation request was made).

## **Bugfixes**

- PR #119: Fixes gather task merging.
- PR #121: Update to distributed 2.5.1.

# 2.33.28 0.0.4 - Initial Support for Non-SMIRNOFF FFs

This release adds initial support for estimating property data sets using force fields not based on the SMIRNOFF specification. In particular, initial AMBER force field support has been added, along with a protocol which applies said force fields using tleap.

#### **New Features**

- PR #96: Adds a mechanism for specifying force fields not in the SMIRNOFF spec.
- PR #99: Adds support for applying AMBER force field parameters through tleap
- PR #111: Protocols now stream trajectories from disk, rather than pre-load the whole thing.
- PR #112: Specific types of protocols can now be easily be replaced using WorkflowOptions.
- PR #117: Adds support for converting PhysicalPropertyDataSet objects to pandas.DataFrame.

## **Bugfixes**

- PR #115: Fixes caching data for substances whose smiles contain forward slashes.
- PR #116: Fixes inconsistent mole fraction rounding.

## **Breaking Changes**

• PR #96: The PropertyEstimatorClient.request\_estimate(force\_field=... argument has been renamed to force\_field\_source.

## **Migration Guide**

This release contained several public API breaking changes. For the most part, these can be remedied by the follow steps:

• Change all instances of PropertyEstimatorClient.request\_estimate(force\_field=...) to PropertyEstimatorClient.request\_estimate(force\_field\_source=...)

# 2.33.29 0.0.3 - ExcessMolarVolume and Typing Improvements

This release implements a number of bug fixes and adds two key new features, namely built in support for estimating excess molar volume measurements, and improved type checking for protocol inputs and outputs.

#### **New Features**

- PR #98: Substance objects may now have components with multiple amount types.
- PR #101: Added support for estimating ExcessMolarVolume measurements from simulations.
- PR #104: typing. Union is now a valid type arguemt to protocol\_output and protocol\_input.

## **Bugfixes**

- PR #94: Fixes exception when testing equality of ProtocolPath objects.
- PR #100: Fixes precision issues when ensuring mole fractions are  $\leq 1.0$ .
- PR #102: Fixes replicated input for children of replicated protocols.
- PR #105: Fixes excess properties weighting by the wrong mole fractions.
- PR #107: Fixes excess properties being converged to the wrong uncertainty.
- PR #108: Fixes calculating MBAR gradients of reweighted properties.

## **Breaking Changes**

- PR #98: Substance.get\_amount renamed to Substance.get\_amounts and now returns an immutable frozenset of Amount objects, rather than a single Amount.
- PR #104: The DivideGradientByScalar, MultiplyGradientByScalar, AddGradients, SubtractGradients and WeightGradientByMoleFraction protocols have been removed. The WeightQuantityByMoleFraction protocol has been renamed to WeightByMoleFraction.

## **Migration Guide**

This release contained several public API breaking changes. For the most part, these can be remedied by the follow steps:

- Change all instances of Substance.get\_amount to Substance.get\_amounts and handle the newly returned frozenset of amounts, rather than the previously returned single amount.
- Replace the now removed protocols as follows:
  - DivideGradientByScalar -> DivideValue
  - MultiplyGradientByScalar -> MultiplyValue
  - AddGradients -> AddValues
  - SubtractGradients -> SubtractValues
  - WeightGradientByMoleFraction -> WeightByMoleFraction
  - WeightQuantityByMoleFraction -> WeightByMoleFraction

# 2.33.30 0.0.2 - Replicator Quick Fixes

A minor release to fix a number of minor bugs related to replicating protocols.

## **Bugfixes**

- PR #90: Fixes merging gradient protocols with the same id.
- PR #92: Fixes replicating protocols for more than 10 template values.
- PR #93: Fixes ConditionalGroup objects losing their conditions input.

## 2.33.31 0.0.1 - Initial Release

The initial pre-alpha release of the framework.

# 2.34 Release Process

This document aims to outline the steps needed to release the openff-evaluator on conda-forge. This should only be done with the approval of the core maintainers.

# 2.34.1 1. Update the Release History

If no PR has been submitted, create a new one to keep track of changes to the release notes *only*. Only the releasehistory.rst file may be edited in this PR.

Ensure that the release history file is up to date, and conforms to the below template:

# 2.34.2 2: Cut the Release on GitHub

To cut a new release on GitHub:

- 1) Go to the Releases tab on the front page of the repo and choose Create a new release.
- 2) Set the release tag using the form: X.Y.Z
- 3) Added a descriptive title using the form: X.Y.Z [Descriptive Title]
- 4) Ensure the This is a pre-release checkbox is ticked.

- 5) Reformat the release notes from part 1) into markdown and paste into the description box.
- a) Append the following extra message above the New Features title:

```
A richer version of these release notes with live links to API documentation is available on [our ReadTheDocs page](https://property-estimator.readthedocs.io/en/latest/
--releasehistory.html)

See our [installation instructions](https://property-estimator.readthedocs.io/en/latest/
--install.html).

Please report bugs, request features, or ask questions through our [issue tracker](https://github.com/openforcefield/openff-evaluator/issues).

**Please note that this is a pre-alpha release and there will still be major changes to the API prior to a stable 1.0.0 release.**
```

Note - You do not need to upload any files. The source code will automatically be added as a `.tar.gz` file.

# 2.34.3 3: Trigger a New Build on Conda Forge

To trigger the build on conda-forge:

- 1) Create a fork of the openff-evaluator-feedstock and make the following changes to the recipe/meta.yaml file:
  - a) Update the version to match the release.
  - b) Set build to 0
  - c) Update any dependencies in the requirements section
  - d) Update the sha256 hash to the output of curl -sL https://github.com/openforcefield/ openff-evaluator/archive/{{ version }}.tar.gz | openssl sha256
  - 2) Open PR to merge the fork into the main feedstock:
  - a) The PR title should have the format Release X.Y.Z
  - b) No PR body text is needed
  - c) The CI will run on this PR (~30 minutes) and attempt to build the package.
  - d) If the build is successful the PR should be reviewed and merged by the feedstock maintainers.
  - e) **Once merged** the package is built again on and uploaded to anaconda.
  - 3) Test the conda-forge package:
  - a) conda install -c conda-forge openff-evaluator

2.34. Release Process 641

# 2.34.4 4: Update the ReadTheDocs Build Versions

To ensure that the read the docs pages are updated:

- 1) Trigger a RTD build of latest.
- 2) Under the Versions tab add the new release version to the list of built versions and save.
- 3) Verify the new version docs have been built and pushed correctly
- 4) Under Admin | Advanced Settings: Set the new release version as Default version to display and save.

# **BIBLIOGRAPHY**

- [1] Alice Glättli, Xavier Daura, and Wilfred F van Gunsteren. Derivation of an improved simple point charge model for liquid water: spc/a and spc/l. *The Journal of chemical physics*, 116(22):9811–9828, 2002.
- [2] Kyle A Beauchamp, Julie M Behr, Ariën S Rustenburg, Christopher I Bayly, Kenneth Kroenlein, and John D Chodera. Toward automated benchmarking of atomistic force fields: neat liquid densities and static dielectric constants from the thermoml data archive. *The Journal of Physical Chemistry B*, 119(40):12912–12920, 2015.
- [3] Junmei Wang and Tingjun Hou. Application of molecular dynamics simulations in molecular property prediction. 1. density and heat of vaporization. *Journal of chemical theory and computation*, 7(7):2151–2165, 2011.
- [4] David R Slochower, Niel M Henriksen, Lee-Ping Wang, John D Chodera, David L Mobley, and Michael K Gilson. Binding thermodynamics of host–guest systems with smirnoff99frosst 1.0. 5 from the open force field initiative. *Journal of Chemical Theory and Computation*, 15(11):6225–6242, 2019.
- [1] John D Chodera. A simple method for automated equilibration detection in molecular simulations. *Journal of chemical theory and computation*, 12(4):1799–1805, 2016.
- [2] Richard A Messerly, S Mostafa Razavi, and Michael R Shirts. Configuration-sampling-based surrogate models for rapid parameterization of non-bonded interactions. *Journal of Chemical Theory and Computation*, 14(6):3144–3162, 2018.
- [1] Lee-Ping Wang, Teresa Head-Gordon, Jay W Ponder, Pengyu Ren, John D Chodera, Peter K Eastman, Todd J Martinez, and Vijay S Pande. Systematic improvement of a classical molecular model of water. *The Journal of Physical Chemistry B*, 117(34):9956–9972, 2013.

Bibliography

## **INDEX**

 $\verb|\__init\__()| (openff.evaluator.datasets.PropertyPhase|$ 

,			* * *
init_	1 00		method), 94
	method), 619	1n1t_	_() (openff.evaluator.datasets.Source method),
init_	_() (openff.evaluator.attributes.AttributeClass		95
	method), 619	1n1t_	_() (openff.evaluator.datasets.curation.components.CurationCom
init_	_() (openff.evaluator.attributes.PlaceholderValue		method), 150
	method), 620 -	1n1t_	_() (openff.evaluator.datasets.curation.components.CurationCom
init_	_() (openff.evaluator.backends.CalculationBacken	d	method), 151
	method), 244	init_	() (openff.evaluator.datasets.curation.components.conversion.Co
init_	_() (openff.evaluator.backends.ComputeResources		method), 220
	method), 245	init_	() (openff.evaluator.datasets.curation.components.conversion.Co
init_	_() (openff.evaluator.backends.QueueWorkerResou	ırces	method), 218
	method), 246	init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init_	_() (openff.evaluator.backends.dask.BaseDaskBack	kend	method), 182
	method), 248	init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering. $Filter$
init_	_() (openff.evaluator.backends.dask.BaseDaskJob(	<u> ДиеиеВа</u>	ckendod), 180
	method), 249	init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init	_() (openff.evaluator.backends.dask.DaskLSFBack	end	method), 174
	method), 252	init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init	_() (openff.evaluator.backends.dask.DaskLocalClu	ıster	method), 172
	method), 251	init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init	_() (openff.evaluator.backends.dask.DaskPBSBack	end	method), 199
	method), 254	init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init_			method), 197
	method), 78	init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
init	_() (openff.evaluator.client.ConnectionOptions		method), 185
	method), 78	init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
init_			method), 183
	method), 76	init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
ini+	_() (openff.evaluator.client.Request method), 80		method), 169
init_		init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
	method), 82		method), 167
init_		init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
	method), 84		method), 193
ini+	() (openff.evaluator.datasets.CalculationSource -	init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
1111	() (openy.evaluator.aataseis.Catcutationsource - method), 96		method), 191
ini+	() (openff avaluator datasets Measurement Course	init	_() (openff.evaluator.datasets.curation.components.filtering.Filter
1111			method), 166
ini+	method), 97	init	_() (openff.evaluator.datasets.curation.components.filtering.Filter
	() (openff.evaluator.datasets.PhysicalProperty -		method), 164

\_\_init\_\_() (openff.evaluator.datasets.PhysicalPropertyDatasets() (openff.evaluator.datasets.curation.components.filtering.Filt

method), 177

Symbols

method), 137

	ng <b>Ki) tea Bent fapeut in Trypals Sichsens</b> auration.workflow.CurationWorkflo
method), 175	method), 154
init() (openff.evaluator.datasets.curation.components.filmin method), 172	ng. <b>n.y.eopsyg.eomu</b> ator.aatasets.taproom.1aproomDataset method), 145
init() (openff.evaluator.datasets.curation.components.filmin	
method), 170	method), 149
init() (openff.evaluator.datasets.curation.components.filmin	· · · · · · · · · · · · · · · · · · ·
method), 187	method), 140
init() (openff.evaluator.datasets.curation.components.filmin	
method), 185	method), 221
init() (openff.evaluator.datasets.curation.components.filmin	ng Kilter Bersffrerkeluator.forcefield.LigParGenForceFieldSource
method), 190	method), 225
init() (openff.evaluator.datasets.curation.components.filmin	ng Kilte <b>v BerSffrerks/Sichonsf</b> orcefield.ParameterGradient
method), 188	method), 228
init() (openff.evaluator.datasets.curation.components.filmin	
method), 180	method), 227
init() (openff.evaluator.datasets.curation.components.filmin	
method), 178	method), 222
init() (openff.evaluator.datasets.curation.components.filmin	
method), 196	method), 224
init() (openff.evaluator.datasets.curation.components.filmin	
method), 193	method), 229
init() (openff.evaluator.datasets.curation.components.filmin	
method), 163	method), 230
init() (openff.evaluator.datasets.curation.components.filmin	
method), 161	method), 231
init() (openff.evaluator.datasets.curation.components.filmin	
method), 160	method), 240
init() (openff.evaluator.datasets.curation.components.filmin	
method), 158	method), 241
init() (openff.evaluator.datasets.curation.components.frenia	
method), 202	method), 237
init() (openff.evaluator.datasets.curation.components.fienia	
method), 200	
	method), 238
	ion. (F)ingpenffict Tylpator. layers. workflow. Workflow Calculation Layer
method), 217	method), 234
	ion.SeleptenfaePalutstor.layers.workflow.WorkflowCalculationScheme
	method), 235
init() (openff.evaluator.datasets.curation.components.sidait	
method), 210	method), 99
init() (openff.evaluator.datasets.curation.components.salarit	1 1
method), 209	method), 106
init() (openff.evaluator.datasets.curation.components.salarit	
method), 207	method), 109
init() (openff.evaluator.datasets.curation.components.salarit	
method), 213	method), 113
init() (openff.evaluator.datasets.curation.components.salarit	
method), 215	method), 102
init() (openff.evaluator.datasets.curation.components.thann	
method), 205	method), 119
init() (openff.evaluator.datasets.curation.components.tlianin	
method), 203	method), 116
	${f Wo}$ (A) flow penff. evaluator. protocols. analysis. Average Dielectric Constants
method), 152	method), 341

```
__init__() (openff.evaluator.protocols.analysis.AverageFreeEnirgies() (openff.evaluator.protocols.paprika.analysis.ComputeReference
         method), 347
                                                                   method), 531
__init__() (openff.evaluator.protocols.analysis.AverageObseiwaltle_() (openff.evaluator.protocols.paprika.analysis.ComputeSymmetri
         method), 335
                                                                   method), 526
__init__() (openff.evaluator.protocols.analysis.BaseAverage this trya Dicopenff.evaluator.protocols.paprika.coordinates.AddDummyA
         method), 329
                                                                   method), 492
__init__() (openff.evaluator.protocols.analysis.BaseDeco<u>rrelairProt</u>6c6bpenff.evaluator.protocols.paprika.coordinates.PreparePullC
         method), 357
                                                                    method), 482
__init___() (openff.evaluator.protocols.analysis.Compute<u>Dipoh:Mom@</u>Mopenff.evaluator.protocols.paprika.coordinates.PrepareRelea
         method), 352
                                                                   method), 487
__init__() (openff.evaluator.protocols.analysis.Decorrelate@hsietyakle.copenff.evaluator.protocols.paprika.restraints.ApplyRestraints
         method), 368
                                                                    method), 515
__init__() (openff.evaluator.protocols.analysis.DecorrelateTinjettor()) (openff.evaluator.protocols.paprika.restraints.GenerateAttach.
         method), 362
                                                                    method), 498
_init__() (openff.evaluator.protocols.coordinates.BuildC<u>oordintates</u>Palahpenlff.evaluator.protocols.paprika.restraints.GeneratePullRe
         method), 373
                                                                    method), 504
__init__() (openff.evaluator.protocols.coordinates.BuildDockailCookJihapesnff.evaluator.protocols.paprika.restraints.GenerateReleas
         method), 386
                                                                   method), 509
_init__() (openff.evaluator.protocols.coordinates.SolvateExixixittgStQdtopenff.evaluator.protocols.reweighting.BaseEvaluateEnergies
         method), 379
                                                                   method), 547
__init__() (openff.evaluator.protocols.forcefield.BaseBuildSixtart__() (openff.evaluator.protocols.reweighting.BaseMBARProtocol
                                                                   method), 553
         method), 393
__init__() (openff.evaluator.protocols.forcefield.BuildLigPaiGirtSys@n(openff.evaluator.protocols.reweighting.ConcatenateObserval
         method), 404
                                                                   method), 542
__init__() (openff.evaluator.protocols.forcefield.BuildSmi<u>rnoffSysterfi</u>) (openff.evaluator.protocols.reweighting.ConcatenateTrajector.
         method), 398
                                                                   method), 537
__init__() (openff.evaluator.protocols.forcefield.BuildTLeap\square_n\struc() (openff.evaluator.protocols.reweighting.ReweightDielectricCo
         method), 409
                                                                   method), 565
__init__() (openff.evaluator.protocols.gradients.ZeroGradieitsit__() (openff.evaluator.protocols.reweighting.ReweightObservable
         method), 415
                                                                    method), 559
__init__() (openff.evaluator.protocols.groups.ConditionalGranit__() (openff.evaluator.protocols.simulation.BaseEnergyMinimisati
         method), 421
                                                                    method), 571
__init__() (openff.evaluator.protocols.miscellaneous.Add<u>Va</u>luxit__() (openff.evaluator.protocols.simulation.BaseSimulation
         method), 428
                                                                   method), 577
__init__() (openff.evaluator.protocols.miscellaneous.Divide\nhte__() (openff.evaluator.protocols.storage.UnpackStoredSimulationD
                                                                   method), 584
         method), 442
__init__() (openff.evaluator.protocols.miscellaneous.Dumm\dagge\textbf{RictionCol}) (openff.evaluator.protocols.utils.ReweightingProtocols
         method), 457
                                                                    method), 615
__init__() (openff.evaluator.protocols.miscellaneous.FilterSithstanceByRpenff.evaluator.protocols.utils.SimulationProtocols
         method), 452
                                                                   method), 613
__init__() (openff.evaluator.protocols.miscellaneous.Mult<u>ipl</u>yXat<u>ue (</u>) (openff.evaluator.protocols.yank.BaseYankProtocol
         method), 437
                                                                   method), 590
__init___() (openff.evaluator.protocols.miscellaneous.Subt<u>raċtVatues</u>() (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
         method), 432
                                                                   method), 596
__init__() (openff.evaluator.protocols.miscellaneous.WeightByMoleFydotpenff.evaluator.protocols.yank.SolvationYankProtocol
         method), 447
                                                                    method), 604
__init__() (openff.evaluator.protocols.openmm.OpenMMEnargyMix()) (septemff.evaluator.server.Batch method), 89
                                                                          (openff.evaluator.server.EvaluatorServer
                                                          __init__()
__init__() (openff.evaluator.protocols.openmm.OpenMMEvaluateEmentyied), 88
         method), 476
                                                          __init__() (openff.evaluator.storage.LocalFileStorage
__init__() (openff.evaluator.protocols.openmm.OpenMMSimulationmethod), 258
                                                          __init__() (openff.evaluator.storage.StorageBackend
         method), 468
__init__() (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRiPithoel), 256
         method), 521
                                                          __init__() (openff.evaluator.storage.attributes.FilePath
```

	method), 289		method), 309
init_	$\_()$ (openff.evaluator.storage.attributes. $Q$ uery $A$ ttr	<i>i<u>bu</u>i</i> enit_	_() (openff.evaluator.workflow.ProtocolGroup
	method), 297		method), 310
init_	_() (openff.evaluator.storage.attributes.StorageAtt	<i>r<u>ib</u>ine</i> it_	* * **
	method), 296		method), 297
init_	_() (openff.evaluator.storage.data.BaseSimulation	<i>Da</i> ianit_	· · · · · · · · · · · · · · · · · · ·
	method), 267		method), 301
init_	_() (openff.evaluator.storage.data.BaseStoredData	ainit_	
	method), 260		method), 302
1n1t_		<i>i</i> 1n1t_	_() (openff.evaluator.workflow.attributes.BaseMergeBehavior
	method), 263	ID 33.	method), 324
1n1t_		i <u>Da</u> nait_	_() (openff.evaluator.workflow.attributes.InequalityMergeBehavid
: m : +	method), 262	4a ini+	method), 325
1n1t_		<i>ι<u>α_</u>1n1τ_</i>	_() (openff.evaluator.workflow.attributes.InputAttribute
: m : ±	method), 265		method), 325
1N1 (_		<u>гуугчикас_</u>	_() (openff.evaluator.workflow.attributes.MergeBehavior method), 324
ini+	method), 273	na Divaria +	_() (openff.evaluator.workflow.attributes.OutputAttribute
1111	_() (openy.evaluator.storage.aata.storeastmutatic method), 269	/// // // // // // // // // // // // //	method), 326
ini+		rv init	() (openff.evaluator.workflow.schemas.ProtocolGroupSchema
	method), 276	<i>,</i>	method), 317
init		nD <del>intoit</del> u	(openff.evaluator.workflow.schemas.ProtocolReplicator
	method), 281	<u> </u>	method), 320
init		ervinit	_() (openff.evaluator.workflow.schemas.ProtocolSchema
	method), 280	·24	method), 316
init		ıta <b>Onė</b> rtv	() (openff.evaluator.workflow.schemas.WorkflowSchema
	method), 287	- <b>-</b>	method), 322
init_	_() (openff.evaluator.storage.query.SimulationDa	taQimeiyt_	() (openff.evaluator.workflow.utils.ProtocolPath
	method), 284		method), 327
init_	_() (openff.evaluator.storage.query.SubstanceQue	<u>ry</u> init_	_() (openff.evaluator.workflow.utils.ReplicatorValue
	method), 278		method), 326
init_	(openff.evaluator.substances.Amount	۸	
	method), 129	Α	
init_	1	absolut	e_tolerance(openff.evaluator.layers.CalculationLayerSchema
	method), 127		attribute), 232
init_	() (openff.evaluator.substances.ExactAmount	absolut	$\verb"e_tolerance" (open ff. evaluator. layers. reweighting. Reweighting Scale and the property of the property $
	method), 131		attribute), 242
1n1t_	_() (openff.evaluator.substances.MoleFraction	absolut	$\verb"e_tolerance" (open ff. evaluator. layers. simulation. Simulation Scheme for the property of the property o$
	method), 133		attribute), 238
init_		absolut	${\tt e\_tolerance} (open \textit{ff.evaluator.layers.workflow.WorkflowCalcula})$
: m : ±	method), 124	and aCtat	attribute), 235
1N1 (_	() (openff.evaluator.thermodynamics.Thermodynamics), 135	activat	
ini+	() (openff.evaluator.utils.observables.Observabl	a	(openff.evaluator.protocols.coordinates.BuildDockedCoordinates
	method), 621		attribute), 388
ini+	_() (openff.evaluator.utils.observables.Observable	aaa_com Array	ponent() (openff.evaluator.substances.Substance
	method), 622		method), 125
init	_() (openff.evaluator.utils.observables.Observable	auu_con •Frame	dition() (openff.evaluator.protocols.groups.ConditionalGroup method), 423
	method), 623		methoa), 425 perties() (openff.evaluator.datasets.PhysicalPropertyDataSet
init	_() (openff.evaluator.utils.observables.Observable	auu_pro eTvpe	method), 138
	method), 623		perties() (openff.evaluator.datasets.taproom.TaproomDataSet
init_		aaa_pr0	method), 146
_	method), 304	add nro	perties() (openff.evaluator.datasets.thermoml.ThermoMLDataS
init_	_() (openff.evaluator.workflow.ProtocolGraph	pro	method), 142

method), 142

add_protocols() (openff.evaluator.protocols.groups.Conditional	lGr <b>auw</b> ibute), 424
	_merging(openff.evaluator.protocols.miscellaneous.AddValues
$\verb add_protocols()  (open ff. evaluator. work flow. Protocol Graph $	attribute), 429
	_merging(openff.evaluator.protocols.miscellaneous.DivideValue
$\verb"add_protocols()" (\textit{openff.evaluator.workflow.ProtocolGroup"}$	attribute), 444
	_merging(openff.evaluator.protocols.miscellaneous.DummyProtoco
add_schema() (openff.evaluator.client.RequestOptions	attribute), 458
	_merging (openff.evaluator.protocols.miscellaneous.FilterSubstance
add_workflows() (openff.evaluator.workflow.WorkflowGraph method), 301 allow	attribute), 454 _merging(openff.evaluator.protocols.miscellaneous.MultiplyValue
AddDummyAtoms (class in	attribute), 439
·	_merging (openff.evaluator.protocols.miscellaneous.SubtractValues
492	attribute), 434
AddValues (class in openff.evaluator.protocols.miscellaneous)].ow	$\verb _merging  (open ff. evaluator. protocols. miscellaneous. Weight By Mole Foundation for the protocol states and the protocol states are also as a finite protocol states and the protocol states are also as a finite protocol state are also as a finite protocol states a$
427	attribute), 449
	$\verb _merging  (open ff. evaluator. protocols. open mm. Open MME nergy Miniscopen for the property of the prope$
(open ff. evaluator. protocols. open mm. Open MMS imulation the substitution of the protocol	
	_merging(openff.evaluator.protocols.openmm.OpenMMEvaluateEn
allow_gpu_platforms	attribute), 477
(openyr.evaluator.protocots.stmutation.baseStmut <b>ation</b> attribute), 579	_merging (openff.evaluator.protocols.openmm.OpenMMSimulation attribute), 470
	annome), 470 Pa <b>neir Giongt(opt</b> enff.evaluator.protocols.paprika.analysis.AnalyzeAPRI
attribute), 342	attribute), 522
	E <b>nengiang</b> (openff.evaluator.protocols.paprika.analysis.ComputeRefe
attribute), 348	attribute), 533
allow_merging(openff.evaluator.protocols.analysis.Average\Olowe	a <b>rweblg</b> ing (openff.evaluator.protocols.paprika.analysis.ComputeSym
attribute), 336	attribute), 528
	<b>Quergiald</b> openff.evaluator.protocols.paprika.coordinates.AddDumn
attribute), 331	attribute), 494
attribute), 358	Lune Pyong Obpenff. evaluator. protocols. paprika. coordinates. Prepare Pattribute), 483
	aurtome), 465 o <b>h&amp;fgingf</b> &penff.evaluator.protocols.paprika.coordinates.PrepareR
attribute), 354	attribute), 488
	<b>Lineargably (</b> openff.evaluator.protocols.paprika.restraints.ApplyRestra
attribute), 369	attribute), 516
allow_merging(openff.evaluator.protocols.analysis.Decoradlated	E <b>mjerginyg</b> (openff.evaluator.protocols.paprika.restraints.GenerateAtt
attribute), 364	attribute), 500
	o <b>rdiargisRy(dpe</b> olff.evaluator.protocols.paprika.restraints.GeneratePu
attribute), 375	attribute), 505
	kmdCgirrdj.hapesnff.evaluator.protocols.paprika.restraints.GenerateRe
attribute), 389	attribute), 511 <b>cinvengStng</b> d <b>wre</b> nff.evaluator.protocols.reweighting.BaseEvaluateEner
attribute), 381	attribute), 549
	y <b>xteng</b> ing (openff.evaluator.protocols.reweighting.BaseMBARProtoc
attribute), 394	attribute), 555
	r <b>(Men:Spisng)</b> (openff.evaluator.protocols.reweighting.ConcatenateObser
attribute), 405	attribute), 543
allow_merging(openff.evaluator.protocols.forcefield.Buildashiom	o <b>fflexogang</b> (openff.evaluator.protocols.reweighting.ConcatenateTrajec
attribute), 399	attribute), 538
	Smerteging (openff.evaluator.protocols.reweighting.ReweightDielectri
attribute), 411	attribute), 567
allow_merging(openff.evaluator.protocols.gradients.Zero@hddie	<u>и</u> шы циц (openп.evaluator.protocols.reweighting.Keweight <b>(</b> )bserval

attribute), 560

 ${\tt allow\_merging} \ (open \textit{ff.evaluator.protocols.groups.} Conditi \textbf{\textit{allabouromerging}} \ (open \textit{ff.evaluator.protocols.simulation.} Base \textit{EnergyMinimized} \ (open \textit{ff.evaluator.protocols.simulation.} Base \textit{EnergyMinimized} \ (open \textit{ff.evaluator.protocols.simulation.} \ (open \textit{ff.evalua$ 

 $attribute),\,417$ 

attribute), 573

attribute), 580

attribute), 586

attribute), 592

attribute), 599

attribute), 608

tribute), 305

attribute), 313

Amount (class in openff.evaluator.substances), 129

amounts (openff.evaluator.substances.Substance at- method), 283
tribute), 125 apply() (openff.evaluator.storage.query.ForceFieldQuery
AnalyzeAPRPhase (class in method), 280
$openff. evaluator. protocols. paprika. analysis), \qquad \texttt{apply()} \ (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query. Free Energy Data Query apply (openff. evaluator. storage. query apply (openff. evaluato$
521 <i>method</i> ), 287
append_uuid() (openff.evaluator.workflow.utils.ProtocolPatply() (openff.evaluator.storage.query.SimulationDataQuery
method), 328 method), 285
apply() (openff.evaluator.datasets.curation.components.coapprisien) CoppenfftExachesiDensoitylflataschemas.ProtocolReplicator
class method), 220 method), 320
apply() (openff.evaluator.datasets.curation.components.Cuapply_Cropplaineattor() (openff.evaluator.protocols.analysis.AverageDielecture)
class method), 150 method), 342
apply() (openff.evaluator.datasets.curation.components.filtapinkyEithepByChttoged) (openff.evaluator.protocols.analysis.AverageFreeE
class method), 182 method), 349
apply() (openff.evaluator.datasets.curation.components.filtapplyEittepByEittep
class method), 174 method), 336
apply() (openff.evaluator.datasets.curation.components.filtapinkyEithepByEarophineopenff.evaluator.protocols.analysis.BaseAverageC
class method), 199 method), 331
apply() (openff.evaluator.datasets.curation.components.filtapinkyEithepByhaniokiQu(dpenff.evaluator.protocols.analysis.BaseDecorrel
class method), 185 method), 359
apply() (openff.evaluator.datasets.curation.components.filtapinkyEithepByMatloFf@c(apenff.evaluator.protocols.analysis.ComputeDipo
class method), 169 method), 354
apply() (openff.evaluator.datasets.curation.components.filtapinkyEittepByNCompOnlantsnff.evaluator.protocols.analysis.DecorrelateO
class method), 193 method), 369
apply() (openff.evaluator.datasets.curation.components.filtapinkyEittepRyPactssnf@ (openff.evaluator.protocols.analysis.DecorrelateTri
class method), 166 method), 364
apply() (openff.evaluator.datasets.curation.components.filtapinkyFittepByPartpar()Typesnff.evaluator.protocols.coordinates.BuildCoordinates.Bu
class method), 177 method), 375
apply() (openff.evaluator.datasets.curation.components.filtapinkyEitepByRatem(c) (openff.evaluator.protocols.coordinates.BuildDock
class method), 172 method), 389
apply() (openff.evaluator.datasets.curation.components.filtapingyEitepBisarilers() (openff.evaluator.protocols.coordinates.SolvateEx
class method), 187 method), 381
apply() (openff.evaluator.datasets.curation.components.filtapingyEittepBySavixks() (openff.evaluator.protocols.forcefield.BaseBuildSy
class method), 190 method), 394
apply() (openff.evaluator.datasets.curation.components.filtapinkyFittepByStartexx(he/noisternff.evaluator.protocols.forcefield.BuildLigPar
class method), 180 method), 405
apply() (openff.evaluator.datasets.curation.components.filtapingyEittepBySaltstatide(openff.evaluator.protocols.forcefield.BuildSmirno
class method), 196 method), 399
apply() (openff.evaluator.datasets.curation.components.filtapingyEittepByTertoper(dytopenff.evaluator.protocols.forcefield.BuildTLeapS

class method), 163

class method), 160

class method), 202

class method), 212

class method), 209

class method), 205

class method), 153

apply() (openff.evaluator.storage.query.BaseSimulationDataQuery

method), 277

allow\_merging (openff.evaluator.protocols.simulation.Basaspiployl@picoppenff.evaluator.datasets.curation.components.filtering.FilterDup

allow\_merging(openff.evaluator.protocols.storage.Unpacksponeyl6)numbariffreDahator.datasets.curation.components.freesolv.ImportFr

allow\_merging (openff.evaluator.protocols.yank.BaseYank\polog\partition (openff.evaluator.datasets.curation.components.selection.SelectDo

allow\_merging (openff.evaluator.protocols.yank.LigandReapptbyYQnldprotff@daluator.datasets.curation.components.selection.SelectSu

allow\_merging (openff.evaluator.protocols.yank.Solvation YappkPyO) (explenff.evaluator.datasets.curation.components.thermoml.Import

allow\_merging (openff.evaluator.workflow.Protocol at- apply() (openff.evaluator.datasets.curation.workflow.CurationWorkflow

allow\_merging(openff.evaluator.workflow.ProtocolGroupapply()(openff.evaluator.storage.query.BaseDataQuery

	nethod), 567
apply_replicator() (openff.evaluator.protocols.gradientapplyGreet	hkintsator() (openff.evaluator.protocols.reweighting.ReweightO
method), 417	nethod), 560
${\tt apply\_replicator()} \ (open \textit{ff.evaluator.protocols.groups.} \textit{Capplitionel} \\ \textbf{protocols.groups.} \ \textbf{Capplitionel} \\ \textbf{protocols.groups.} \ Capplitio$	<b>Graptor()</b> (openff.evaluator.protocols.simulation.BaseEnergy
method), 424	nethod), 573
apply_replicator() (openff.evaluator.protocols.miscellaapplsyAdd	Mikeator() (openff.evaluator.protocols.simulation.BaseSimulat
method), 429	nethod), 580
apply_replicator() (openff.evaluator.protocols.miscellaapplsyDren	hWahror() (openff.evaluator.protocols.storage.UnpackStoredSi
	nethod), 586
apply_replicator() (openff.evaluator.protocols.miscellaapplsyDnop	
	nethod), 592
apply_replicator() (openff.evaluator.protocols.miscellaapplsyEidep	
	nethod), 599
apply_replicator() (openff.evaluator.protocols.miscellaapplsyMndp	
	nethod), 608
apply_replicator() (openff.evaluator.protocols.miscellaapplsySrlen	
	nethod), 307
apply_replicator() (openff.evaluator.protocols.miscellaapplsy\(\frac{Weeg}{e}\)	
	nethod), 314
apply_replicator() (openff.evaluator.protocols.openmma@paryMMeE	
	ttribute), 598
$\verb apply_replicator()  (open \textit{ff.evaluator.protocols.open mm. \textit{Mp} \textit{arg} \texttt{Mess} \textit{fine} \textit$	
	penff.evaluator.protocols.paprika.restraints),
$\verb"apply_replicator"()" (open \textit{ff.evaluator.protocols.open mm.Open MMS}") apply_replicator()" (open ff.evaluator.protocols.open mm.open mm.Open mm.Open mm.Open mm.Open mm.Open mm.Open mm.Open mm.o$	
method), 471 assigned_	_residue_names
$apply\_replicator()$ (openff.evaluator.protocols.paprika.analysis.Applicator())	n <b>qlspfAePRIPlatase</b> protocols.coordinates.BuildCoordinatesPackmo
method), 522	ttribute), 375
apply_replicator() (openff.evaluator.protocols.paprika.asshginedo	proprietrefe name \$Vork
method), 533	openff.evaluator.protocols.coordinates.SolvateExistingStructure
apply_replicator() (openff.evaluator.protocols.paprika.analysis.Com	
	ambdas (openff.evaluator.protocols.paprika.restraints.GenerateA
apply_replicator() (openff.evaluator.protocols.paprika.coordinates	
	ambdas ( <i>openff.evaluator.protocols.paprika.restraints.GenerateP</i>
apply_replicator() (openff.evaluator.protocols.paprika.coordinates	
	c(class in openff.evaluator.attributes), 618
apply_replicator() (openff.evaluator.protocols.paprika.Advantibutes	
	19
apply_replicator() (openff.evaluator.protocols.paprika. Assemaigs DA	
•	penff.evaluator.protocols.analysis), 341
apply_replicator() (openff.evaluator.protocols.paprika. Assemages FG	
	penff.evaluator.protocols.analysis), 347
apply_replicator() (openff.evaluator.protocols.paprika. Assemages Of	
**	penff.evaluator.protocols.analysis), 335
apply_replicator() (openff.evaluator.protocols.paprika.restraints.G	GenerateReleaseRestraints
<i>method</i> ), 511	
apply_replicator() (openff.evaluator.protocols.reweighting BayeFy	deleobservabre (class in
method), 549	penff.evaluator.protocols.analysis), 329
apply_replicator() (openff.evaluator.protocols.reweighting Bright Medicator)	RARRENTOCOL (class in
.7 7 555	penff.evaluator.protocols.forcefield), 393
apply_replicator() (openff.evaluator.protocols.reweightings Course	enogeAbservables (class in
T T # 4.4	penff.evaluator.backends.dask), 248
apply_replicator() (openff.evaluator.protocols.reweight	rong:evanaao:backenas.aask), 248 rongaeraenas.aask), 248 in
7 70 700	penff.evaluator.backends.dask), 249
apply_replicator() (openff.evaluator.protocols.reweighting.Reweigh	pengj.evananor.vackenas.aask), 249 htDielectricConstant

BaseDataQuery	(class	in	$(open {\it ff. evaluator. protocols. analysis.} A verage Observable$
	torage.query), 276		attribute), 337
BaseDecorrelateProtoc	*	in	bootstrap_sample_size
	protocols.analysis), 357		$(open {\it ff. evaluator. protocols. analysis. Base Average Observable}$
BaseEnergyMinimisation	n (class	in	attribute), 331
openff.evaluator.p	protocols.simulation), 571		bootstrap_uncertainties
BaseEvaluateEnergies	(class	in	(openff. evaluator. protocols. reweighting. Base MBAR Protocol
openff.evaluator.p	protocols.reweighting), 547	7	attribute), 554
BaseMBARProtocol	(class	in	bootstrap_uncertainties
openff.evaluator.p	protocols.reweighting), 553	3	(openff.evaluator.protocols.reweighting.ReweightDielectricConst
BaseMergeBehavior	(class	in	attribute), 566
openff.evaluator.w	vorkflow.attributes), 324		bootstrap_uncertainties
BaseSimulation	(class	in	(openff.evaluator.protocols.reweighting.ReweightObservable
	protocols.simulation), 577		attribute), 561
BaseSimulationData	(class	in	$\verb box_aspect_ratio   (open \textit{ff.evaluator.protocols.coordinates.BuildCoordinates.}  (open ff.evaluator.protocols.coordinates.BuildCoordina$
openff.evaluator.s	,		attribute), 374
BaseSimulationDataQue:		in	box_aspect_ratio(openff.evaluator.protocols.coordinates.SolvateExistin
	torage.query), 281	.,,	attribute), 382
BaseStoredData	(class	in	BuildCoordinatesPackmol (class in
openff.evaluator.s	`	.,,	openff.evaluator.protocols.coordinates), 373
BaseYankProtocol	(class	in	BuildDockedCoordinates (class in
	protocols.yank), 590	iri	openff.evaluator.protocols.coordinates), 386
Batch (class in openff.evali			BuildDockedCoordinates.ActivateSiteLocation
=		at	
batch_mode (openff.evalue tribute), 83	uor.cueni.kequesiOpiions	ai-	(class in openff.evaluator.protocols.coordinates), 388
BatchMode (class in openff.			BuildLigParGenSystem (class in
beta (openff.evaluator.thern	nodynamics.Thermodynai	micSte	ate openff.evaluator.protocols.forcefield), 404
property), 135			BuildLigParGenSystem.WaterModel (class in
<pre>bootstrap()</pre>	(in mod	dule	openff.evaluator.protocols.forcefield), 405
openff.evaluator.u	tils.observables), 625		BuildSmirnoffSystem (class in
<pre>bootstrap_cycles(openf)</pre>	f.evaluator.protocols.anal	ysis.A	werageFree <b>&amp;pænffies</b> aluator.protocols.forcefield), 398
attribute), 348			BuildTLeapSystem (class in
bootstrap_iterations			openff.evaluator.protocols.forcefield), 409
(openff.evaluator.)	protocols.analysis.Average	eDiele	e <b>Bruic</b> Yd <b>Fisearp</b> System.ChargeBackend (class in
attribute), 343			openff.evaluator.protocols.forcefield), 410
bootstrap_iterations			BuildTLeapSystem.WaterModel (class in
(openff.evaluator.)	protocols.analysis.Average	eObse	ervable openff.evaluator.protocols.forcefield), 411
attribute), 337			
bootstrap_iterations			C
	protocols.analysis.BaseAv	erage	Carrigate_aqueous_ionic_mole_fraction()
attribute), 331	V	0	(openff.evaluator.substances.Substance static
bootstrap_iterations			method), 126
	protocols reweighting Ras	eMR	ARRICHTETion_layer() (in module
attribute), 554	57010coisiremeighting.Bus	(C1)11D1	
bootstrap_iterations			openff.evaluator.layers), 233
	protocols.reweighting.Rew	veioht	calculation_layers (openff.evaluator.client.RequestOptions
attribute), 567	orotocois.reweighting.rem	reigni	<i>''</i>
bootstrap_iterations			calculation_schemas
	protocols.reweighting.Rew	veiaht	(openff.evaluator.client.RequestOptions at-
attribute), 561	orocous.reweighting.Rev	veigni	<i>"</i>
bootstrap_sample_size			CalculationBackend (class in
	nrotocols analysis Averag	aDial.	openff.evaluator.backends), 243
attribute), 343	motocois.unui ysis.Averuge	ואונונ	ectif Grade Class in openff.evaluator.layers),
bootstrap_sample_size			229
noorstrah sambre 2176			

- CalculationLayerResult (class in can\_merge() (openff.evaluator.protocols.openmm.OpenMMEnergyMinimi method), 465

  CalculationLayerSchema (class in can\_merge() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergyMinimi method), 465
  - openff.evaluator.layers), 231 method), 478
- CalculationSource (class in can\_merge() (openff.evaluator.protocols.openmm.OpenMMSimulation openff.evaluator.datasets), 96 method), 471
- can\_merge() (openff.evaluator.protocols.analysis.AverageDixlectneirGe(i)st(openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPho method), 343 method), 523
- can\_merge() (openff.evaluator.protocols.analysis.AverageFranEmergie() (openff.evaluator.protocols.paprika.analysis.ComputeReferent method), 349 method), 533
- can\_merge() (openff.evaluator.protocols.analysis.Average@lasermeblge() (openff.evaluator.protocols.paprika.analysis.ComputeSymmemethod), 337 method), 528
- can\_merge() (openff.evaluator.protocols.analysis.BaseAverage() (openff.evaluator.protocols.paprika.coordinates.AddDummyamethod), 332 method), 495
- can\_merge() (openff.evaluator.protocols.analysis.BaseDeconnelmePgenOcoppenff.evaluator.protocols.paprika.coordinates.PreparePull method), 359 method), 484
- can\_merge() (openff.evaluator.protocols.analysis.ComputeDip\_oheMge()\_1(openff.evaluator.protocols.paprika.coordinates.PrepareRele method), 355 method), 489
- can\_merge() (openff.evaluator.protocols.analysis.DecorrelateOlmergel@eopenff.evaluator.protocols.paprika.restraints.ApplyRestraint method), 369 method), 517
- can\_merge() (openff.evaluator.protocols.analysis.DecorrelateTimjerge(y) (openff.evaluator.protocols.paprika.restraints.GenerateAttac method), 364 method), 500
- can\_merge() (openff.evaluator.protocols.coordinates.BuildCam\_rdinxgesPa(uppenff.evaluator.protocols.paprika.restraints.GeneratePullR method), 376 method), 506
- can\_merge() (openff.evaluator.protocols.coordinates.Buildencemetre () (openff.evaluator.protocols.paprika.restraints.GenerateReleanuethod), 390 method), 511
- can\_merge() (openff.evaluator.protocols.coordinates.Solva@Evimerg@f() (toprenff.evaluator.protocols.reweighting.BaseEvaluateEnergimethod), 382 method), 550
- can\_merge() (openff.evaluator.protocols.forcefield.BaseBuildfsymerage() (openff.evaluator.protocols.reweighting.BaseMBARProtocol method), 395 method), 556
- can\_merge() (openff.evaluator.protocols.forcefield.BuildLigEnr@er@er@er@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr@er.forcefield.BuildLigEnr.forcef
- can\_merge() (openff.evaluator.protocols.forcefield.BuildSmcianoff@rygen() (openff.evaluator.protocols.reweighting.ConcatenateTrajecto method), 400 method), 539
- can\_merge() (openff.evaluator.protocols.forcefield.BuildTl@apSmetege() (openff.evaluator.protocols.reweighting.ReweightDielectricC method), 411 method), 567
- can\_merge() (openff.evaluator.protocols.gradients.ZeroGradianmerge() (openff.evaluator.protocols.reweighting.ReweightObservable method), 418 method), 561
  can\_merge() (openff.evaluator.protocols.groups.ConditionalGramprge() (openff.evaluator.protocols.simulation.BaseEnergyMinimisation.BaseEnergyMin
- method), 424 method), 574

  can\_merge() (openff.evaluator.protocols.miscellaneous.Add Yalpwerge() (openff.evaluator.protocols.simulation.BaseSimulation
- can\_merge() (openff.evaluator.protocols.miscellaneous.Add malimerge() (openff.evaluator.protocols.simulation.BaseSimulation
  method), 429
  method), 581
  can\_merge() (openff.evaluator.protocols.miscellaneous.Dividh\_mbeege() (openff.evaluator.protocols.storage.UnpackStoredSimulation.
- method), 444 method), 587 can\_merge() (openff.evaluator.protocols.miscellaneous.DuoanyRearge()) (openff.evaluator.protocols.yank.BaseYankProtocol
- can\_merge() (openff.evaluator.protocols.miscellaneous.Duvanytherige(l) (openff.evaluator.protocols.yank.BaseYankProtocol method), 459 method), 593
- can\_merge() (openff.evaluator.protocols.miscellaneous.FilterSubstance(B)(Rplenff.evaluator.protocols.yank.LigandReceptorYankProtocols.method), 455

  method), 600

  Method), 600
- can\_merge() (openff.evaluator.protocols.miscellaneous.MudtiplyMertige() (openff.evaluator.protocols.yank.SolvationYankProtocol method), 439 method), 609
- can\_merge() (openff.evaluator.protocols.miscellaneous.Sukranctheriges() (openff.evaluator.workflow.Protocol method), 434 (openff.evaluator.workflow.Protocol
- can\_merge() (openff.evaluator.protocols.miscellaneous.WeightByAfoleF)a(apunff.evaluator.workflow.ProtocolGroup method), 450 method), 312

capitalize() (openff.evaluator.storage.attributes.FilePa	
method), 291	${\tt component\_roles} \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. miscellaneous. Filter Substantial Component\_roles) \ (open ff. evaluator. protocols. proto$
${\tt casefold()} \ (open ff. evaluator. storage. attributes. File Path$	attribute), 453
method), 291	components (openff.evaluator.substances.Substance at-
<pre>center() (openff.evaluator.storage.attributes.FilePath</pre>	tribute), 125
method), 291	$\verb components_only  (open ff. evaluator. storage. query. Substance Query$
center_solute_in_box	attribute), 279
(openff. evaluator. protocols. coordinates. Solvate E	ExisimpsineDipoleMoments (class in
attribute), 381	openff.evaluator.protocols.analysis), 352
${\tt charge\_backend} \ (\textit{openff.evaluator.protocols.forcefield.But})$	ui <b>lcompresser</b> enceWork (class in
attribute), 411	openff.evaluator.protocols.paprika.analysis),
<pre>checkpoint_frequency</pre>	531
(open ff. evaluator. protocols. open mm. Open MMS in the contract of the con	m <b>Chmpu</b> teResources (class in
attribute), 471	openff.evaluator.backends), 245
checkpoint_frequency	ComputeResources.GPUPrecision (class in
(openff.evaluator.protocols.simulation.BaseSimu	ulation openff.evaluator.backends), 245
attribute), 579	ComputeResources.GPUToolkit (class in
checkpoint_interval	openff.evaluator.backends), 245
(openff.evaluator.protocols.yank.BaseYankProtoc	colomputeSymmetryCorrection (class in
attribute), 591	openff.evaluator.protocols.paprika.analysis),
checkpoint_interval	526
(openff.evaluator.protocols.yank.LigandReceptor	r <b>YanhPattenal</b> teObservables (class in
attribute), 600	openff.evaluator.protocols.reweighting), 542
checkpoint_interval	ConcatenateTrajectories (class in
(openff.evaluator.protocols.yank.SolvationYankP	·
attribute), 609	Conditional Group (class in
clear() (openff.evaluator.utils.observables.ObservableFr	
method), 625	ConditionalGroup.Condition (class in
clear_gradients() (openff.evaluator.utils.observables.(	
method), 622	ConditionalGroup.Condition.Type (class in
clear_gradients() (openff.evaluator.utils.observables.(	
method), 623	conditions (openff.evaluator.protocols.groups), 422
clear_gradients() (openff.evaluator.utils.observables.0	
method), 624	confidence_intervals
	_
complex_coordinate_path	(openff.evaluator.protocols.analysis.AverageFreeEnergies
(openff.evaluator.protocols.paprika.restraints.Ge	
attribute), 499	connection_options (openff.evaluator.client.Request
complex_coordinate_path	attribute), 81
	en <b>comeatiRoxOptrio</b> ns (class in openff.evaluator.client),
attribute), 506	/8
complex_electrostatic_lambdas	construct() (openff.evaluator.datasets.curation.components.conversion.C
(openff.evaluator.protocols.yank.LigandReceptor	
attribute), 599	construct() (openff.evaluator.datasets.curation.components.CurationCon
complex_file_path(openff.evaluator.protocols.paprika.	
attribute), 484	construct() (openff.evaluator.datasets.curation.components.filtering.Filtering.
$\verb complex_file_path  (open \textit{ff.evaluator.protocols.paprika.} $	
attribute), 489	$\verb construct()  (open ff. evaluator. datasets. curation. components. filtering. Filteri$
complex_steric_lambdas	class method), 173
	rY and $R$ $tau$ call ( ) (openff. evaluator. datasets. curation. components. filtering. Filtering and $R$ is a component of the $R$ $t$
attribute), 599	class method), 198
Component (class in openff.evaluator.substances), 127	$\verb construct()  (open ff. evaluator. datasets. curation. components. filtering. Filteri$
component (openff.evaluator.protocols.miscellaneous.Wei	
attribute), 448	<pre>construct() (openff.evaluator.datasets.curation.components.filtering.Filte</pre>
Component.Role (class in openff.evaluator.substances),	class method), 168

copy() (openff.evaluator.datasets.curation.components.selection.TargetSta

```
construct() (openff.evaluator.datasets.curation.components.filtering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiiltering.fiilteri
                                                                                                                                                                 coordinate_file_path
                          class method), 192
construct() (openff.evaluator.datasets.curation.components.filteringdfilterffleyBhessarqSchoods.forcefield.BuildSmirnoffSystem
                          class method), 165
                                                                                                                                                                                            attribute), 400
construct() (openff.evaluator.datasets.curation.component.oftheimetEilfeiBePpartartyTypesSchema
                                                                                                                                                                                            (openff.evaluator.protocols.forcefield.BuildTLeapSystem
                          class method), 176
construct() (openff.evaluator.datasets.curation.components.filteringaffilterila)RucemicSchema
                                                                                                                                                                 coordinate_file_path
                           class method), 171
construct() (openff.evaluator.datasets.curation.components.filteringafieneffæy&hnides&chromowols.storage.UnpackStoredSimulationData
                          class method), 186
                                                                                                                                                                                             attribute), 585
construct() (openff.evaluator.datasets.curation.components.ft/te)i(repFiifferBasmiokxsadusets.curation.components.conversion.Conversion.
                                                                                                                                                                                             method), 219
                          class method), 189
construct() (openff.evaluator.datasets.curation.components.ft/t/e)i(nepFiiffeeBeyStanondhuensists:xSucheina.components.CurationComponents.
                          class method), 179
                                                                                                                                                                                             method), 151
construct() (openff.evaluator.datasets.curation.components.filterByCl
                          class method), 194
                                                                                                                                                                                             method), 181
construct() (openff.evaluator.datasets.curation.components/filterByEllattpredatase&chemation.components.filterByEllattpredatase&chemation.components.filterByEllattpredatase
                                                                                                                                                                                            method), 173
                          class method), 162
construct() (openff.evaluator.datasets.curation.componenta.filterByEn
                          class method), 159
                                                                                                                                                                                             method), 198
construct() (openff.evaluator.datasets.curation.components.filterBylor
                          class method), 201
                                                                                                                                                                                            method), 184
construct() (openff.evaluator.datasets.curation.components.pylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylentopesylento
                                                                                                                                                                                             method), 168
                          class method), 211
construct() (openff.evaluator.datasets.curation.components.filterByNC
                          class method), 208
                                                                                                                                                                                             method), 192
construct() (openff.evaluator.datasets.curation.components.filterByPr
                          class method), 214
                                                                                                                                                                                             method), 165
construct() (openff.evaluator.datasets.curation.components.filtering.FilterByPr
                          class method), 216
                                                                                                                                                                                             method), 176
construct() (openff.evaluator.datasets.curation.components.ph/e()n/openff.pval/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Itherman/Ither
                          class method), 204
                                                                                                                                                                                             method), 171
construct() (openff.evaluator.datasets.curation.workflow.Copyt()n(Wporkflow&bluenn.datasets.curation.components.filtering.FilterBySn
                          class method), 155
                                                                                                                                                                                             method), 186
ConvertExcessDensityData
                                                                                                                                                     in copy() (openff.evaluator.datasets.curation.components.filtering.FilterBySn
                                                                                                             (class
                          openff.evaluator.datasets.curation.components.conversion), method), 189
                           220
                                                                                                                                                                  copy() (openff.evaluator.datasets.curation.components.filtering.FilterBySte
ConvertExcessDensityDataSchema
                                                                                                                        (class
                                                                                                                                                      in
                                                                                                                                                                                            method), 179
                          openff.evaluator.datasets.curation.components.components.components.filterBySu
                           217
                                                                                                                                                                                            method), 194
coordinate_file_name
                                                                                                                                                                  copy() (openff.evaluator.datasets.curation.components.filtering.FilterByTe
                           (openff.evaluator.storage.data.StoredSimulationData
                                                                                                                                                                                             method), 162
                          attribute), 270
                                                                                                                                                                  copy() (openff.evaluator.datasets.curation.components.filtering.FilterDupl
coordinate_file_path
                                                                                                                                                                                             method), 159
                          (openff.evaluator.protocols.coordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.BuildCoordinates.
                          attribute), 375
                                                                                                                                                                                             method), 201
coordinate_file_path
                                                                                                                                                                 copy() (openff.evaluator.datasets.curation.components.selection.SelectDat
                          (openff.evaluator.protocols.coordinates.SolvateExistingStruotnetbod), 211
                          attribute), 382
                                                                                                                                                                 copy() (openff.evaluator.datasets.curation.components.selection.SelectSub
coordinate_file_path
                                                                                                                                                                                             method), 208
                          (openff.evaluator.protocols.forcefield.BaseBuildSystepty() (openff.evaluator.datasets.curation.components.selection.State
                          attribute), 394
                                                                                                                                                                                            method), 214
```

Index 655

(openff.evaluator.protocols.forcefield.BuildLigParGenSystemmethod), 216

coordinate\_file\_path

copy() (openff.evaluator.datasets.curation.components method), 204	s.ther <b>D</b>	<b>eaok hepertaThe wjooMbby</b> taSchema (class in openff.evaluator.protocols.analysis), 362
copy() (openff.evaluator.datasets.curation.workflow.CurationeWardflowsSignerika_schema()		
method), 155 copy() (openff.evaluator.workflow.utils.ProtocolPa	ath	(openff.evaluator.properties.HostGuestBindingAffinity class method), 121
method), 328		efault_request_options()
count() (openff.evaluator.storage.attributes.FilePamethod), 291		(openff.evaluator.client.EvaluatorClient static method), 77
count_exact_amount (openff.evaluator.protocols.coo	ordina	<i>"</i>
attribute), 374	) i cui i cua	(openff.evaluator.properties.Density static
<pre>count_exact_amount(openff.evaluator.protocols.coo</pre>	ordinat	es.Solvat <b>eleximatin</b> gStructure
attribute), 383	d	efault_reweighting_schema()
CurationComponent (class	in	(openff.evaluator.properties.DielectricConstant
openff.evaluator.datasets.curation.component	ets),	static method), 107
150	d	efault_reweighting_schema()
CurationComponentSchema (class	in	(openff.evaluator.properties.EnthalpyOfMixing
openff.evaluator.datasets.curation.componen		class method), 110
151	d	efault_reweighting_schema()
CurationWorkflow (class	in	$(open {\it ff.evaluator.properties.} Enthalpy Of Vaporization$
openff. evaluator. datasets. curation. work flow),		class method), 114
152	d	efault_reweighting_schema()
`	in	$(open {\it ff. evaluator. properties.} Excess Molar Volume$
openff.evaluator.datasets.curation.workflow),		class method), 103
153		efault_simulation_schema()
current_iteration(openff.evaluator.protocols.group attribute), 423	ps.Cor	nditional <b>(inpenf</b> f.evaluator.properties.Density static method), 100
cutoff(openff.evaluator.forcefield.LigParGenForceFi	ald Sad	
property), 226	ciasom	(openff.evaluator.properties.DielectricConstant
cutoff (openff.evaluator.forcefield.TLeapForceFieldSo	ource	static method), 107
property), 224		efault_simulation_schema()
1 1 2//		(openff.evaluator.properties.EnthalpyOfMixing
D		class method), 111
DaskLocalCluster (class	in d	efault_simulation_schema()
openff.evaluator.backends.dask), 251		(openff.evaluator.properties.EnthalpyOfVaporization
DaskLSFBackend (class	in	static method), 114
openff.evaluator.backends.dask), 252	d	efault_simulation_schema()
DaskPBSBackend (class	in	$(open {\it ff.} evaluator. properties. Excess Molar Volume$
openff.evaluator.backends.dask), 254		class method), 104
data_class() (openff.evaluator.storage.query.BaseDo	ataQtd	efault_simulation_schema()
class method), 277		(openff.evaluator.properties.SolvationFreeEnergy
data_class() (openff.evaluator.storage.query.BaseSid	mulati	onDataOtlefic method), 117 efault_storage_query() (in module
class method), 283		
data_class() (openff.evaluator.storage.query.ForceF class method), 280	retaQt d	efault_unit() (openff.evaluator.datasets.PhysicalProperty
data_class() (openff.evaluator.storage.query.FreeEn		
class method), 287	d	efault_unit() (openff.evaluator.properties.Density
data_class() (openff.evaluator.storage.query.Simula.	tionDo	ntaQueryclass method), 100
class method), 285	d	efault_unit() (openff.evaluator.properties.DielectricConstant
data_to_store (openff.evaluator.layers.CalculationL	averRe	esult class method), 107
attribute), 230	d	efault_unit() (openff.evaluator.properties.EnthalpyOfMixing
data_to_store(openff.evaluator.workflow.WorkflowR	Result	class method), 110
attribute), 303	d	$\verb efault_unit()  (open ff. evaluator. properties. Enthalpy Of Vaporization   \\$
DecorrelateObservables (class	in	class method), 114
openff.evaluator.protocols.analysis), 368	Ч	efault_unit()(openff.evaluator.properties.ExcessMolarVolume

class method), 103	property), 435
	Affinity es (openff.evaluator.protocols.miscellaneous.WeightByMoleFr
class method), 120	property), 450
	gdencies (openff.evaluator.protocols.openmm.OpenMMEnergyMinin
class method), 117	property), 465
	ndencies (openff. evaluator. protocols. openmm. OpenMME valuate Energy $a$
(openff.evaluator.properties.HostGuestBindingAffinity	property), 478
	ndencies (openff.evaluator.protocols.openmm.OpenMMSimulation
Density (class in openff.evaluator.properties), 99	property), 472
property), 343	adeinCioast(uptenff.evaluator.protocols.paprika.analysis.AnalyzeAPRPi property), 523
	dergiaes (openff.evaluator.protocols.paprika.analysis.ComputeRefer
property), 349	property), 534
dependencies (openff.evaluator.protocols.analysis.Averaged Dependencies)	rdebleies (openff.evaluator.protocols.paprika.analysis.ComputeSymm
property), 337	property), 529
dependencies (openff.evaluator.protocols.analysis.BaseAvener	<b>ประก<i>ะ</i>เวเปร</b> (openff.evaluator.protocols.paprika.coordinates.AddDumm
property), 332	property), 495
dependencies (openff.evaluator.protocols.analysis.BaseDedeper	<b>ldæRcùæs (o</b> bpenff.evaluator.protocols.paprika.coordinates.PreparePu
property), 359	property), 484
	n <b>lleMaines</b> (copenff.evaluator.protocols.paprika.coordinates.PrepareRe
property), 355	property), 489
	Ndenvales (openff.evaluator.protocols.paprika.restraints.ApplyRestrai
property), 370	property), 517
	ndemaios (openff.evaluator.protocols.paprika.restraints.GenerateAtta
property), 365	property), 501
	ndianstisses (chapeolff.evaluator.protocols.paprika.restraints.GeneratePull
property), 376	property), 506
	ked6oivesi(happenff.evaluator.protocols.paprika.restraints.GenerateRele
property), 390	property), 512
	iskeing:Siesa(wprenff.evaluator.protocols.reweighting.BaseEvaluateEnerg
property), 383	property), 550
	ndemcies (openff.evaluator.protocols.reweighting.BaseMBARProtoco
property), 395  dependencies (openflowalyater protocols forcefield BuildideBor	property), 556  (Gens is to penff. evaluator. protocols. reweighting. Concatenate Observ
property), 406	property), 544
	property), 544 <b>MEnstains</b> (openff.evaluator.protocols.reweighting.ConcatenateTraject
property), 400	property), 539
	Sd <b>ena</b> i es (openff.evaluator.protocols.reweighting.ReweightDielectric
property), 412	property), 568
	ndencies (openff.evaluator.protocols.reweighting.ReweightObservab
property), 418	property), 561
	udencies (openff.evaluator.protocols.simulation.BaseEnergyMinimis
property), 425	property), 574
dependencies (openff.evaluator.protocols.miscellaneous.Add\delta	
property), 430	property), 581
	idlancies (openff.evaluator.protocols.storage.UnpackStoredSimulation
property), 445	property), 587
dependencies (openff.evaluator.protocols.miscellaneous.Dieper	
property), 459	property), 593
	ndsmoi:eByRpknff.evaluator.protocols.yank.LigandReceptorYankProto
property), 455	property), 600
donondoncias (onerff evaluator protocols miscellaneous Midriot	ndenta as (onenff evaluator protocols vank Solvation Yank Protocol

property), 609

(openff.evaluator.workflow.Protocol

property), 440

dependencies (openff.evaluator.protocols.miscellaneous.Salependenaixes

property), 306	attribute), 342
dependencies (openff.evaluator.workflow.ProtocolGroup	${\tt dipole\_moments} (open {\it ff. evaluator. protocols. analysis. Compute Dipole Months}) and {\it fine to the protocols of the$
property), 311	attribute), 354
dict() (openff.evaluator.datasets.curation.components.com	dipole Comment Es (epsil) flusity Dattou Sphotna ols. reweighting. Reweight Dielect
method), 220	attribute), 566
dict() (openff.evaluator.datasets.curation.components.Cur	Diivii GeViqloue nt Schema (class in
method), 152	openff.evaluator.protocols.miscellaneous),
dict() (openff.evaluator.datasets.curation.components.filte	
	divisor (openff.evaluator.protocols.analysis.AverageObservable
dict() (openff.evaluator.datasets.curation.components.filte	
	divisor (openff.evaluator.protocols.miscellaneous.DivideValue
dict() (openff.evaluator.datasets.curation.components.filte	
	docked_complex_coordinate_path
	ring.Filten <b>BpknffalvafunalSchennto</b> cols.coordinates.BuildDockedCoordinates
method), 184	attribute), 389
dict() (openff.evaluator.datasets.curation.components.filte	
method), 169	(openff.evaluator.protocols.coordinates.BuildDockedCoordinates
dict() (openff.evaluator.datasets.curation.components.filte	
	doi (openff.evaluator.datasets.MeasurementSource at-
dict() (openff.evaluator.datasets.curation.components.filte	. 1 00
	download_url(openff.evaluator.forcefield.LigParGenForceFieldSource
dict() (openff.evaluator.datasets.curation.components.filte	
	DummyProtocol (class in
	· · · · · · · · · · · · · · · · · · ·
dict() (openff.evaluator.datasets.curation.components.filte	ring.Finen <b>p<sub>y</sub>anggeenuusunerpu</b> otocois.misceuaneous), 457
method), 171	
dict() (openff.evaluator.datasets.curation.components.filte method), 187	ring.FilierBySmilesSchema F
	onige Eilten Dr. Canindia Coltona C. I
	viffeCitivBySmirk1861(openff.evaluator.protocols.reweighting.BaseMBARF
method), 190	attribute), 555
	vifigoCitievByStemp1dve(vipenyfielventuator.protocols.reweighting.ReweightDie
method), 179	attribute), 568
	vifigoCitievByStalmpluese(sSpeleoffic)valuator.protocols.reweighting.ReweightOb.
method), 195	attribute), 561
dict() (openff.evaluator.datasets.curation.components.filte	
method), 163	(openff.evaluator.protocols.yank.SolvationYankProtocol
dict() (openff.evaluator.datasets.curation.components.filte	- //
	electrostatic_lambdas_2
	esolv.ImpottFreeffelwMahamaprotocols.yank.SolvationYankProtocol
method), 202	attribute), 607
$\verb"dict()" (open ff. evaluator. datasets. curation. components. selection of the components of the co$	
method), 212	attribute), 90
	e <b>etiabSe<u>l</u>ottSubstemsfæsEdhema</b> .protocols.openmm.OpenMMEnergyMinimis
method), 209	attribute), 465
	e <b>ctiab.Seate</b> bc (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergi
method), 214	attribute), 478
$\verb"dict()" (open ff. evaluator. datasets. curation. components. selection for the component of the componen$	etiab Terpto (apenff.evaluator.protocols.openmm.OpenMMSimulation
method), 216	attribute), 472
$\verb"dict()" (open ff. evaluator. datasets. curation. components. the large states and the large states are also also also also also also also also$	r <b>onable<u>n</u>porThornfiMhDatoSprono</b> cols.reweighting.BaseEvaluateEnergie.
method), 205	attribute), 549
$\verb"dict()" (open ff. evaluator. datasets. curation. work flow. Curati$	enWolles@ph&dupraff.evaluator.protocols.simulation.BaseEnergyMinimisati
method), 156	attribute), 573
DielectricConstant (class in	enable_pbc(openff.evaluator.protocols.simulation.BaseSimulation
openff.evaluator.properties), 106	attribute), 579
dipole_moments (openff.evaluator.protocols.analysis.Aver	

encode() (openff.evaluator.storage.attributes.FilePath	method), 383
method), 292	${\tt execute()} \ (open {\it ff.evaluator.protocols.force field.Base Build System}$
end_state_trajectory	method), 395
(openff.evaluator.storage.data.StoredFreeEnergy.attribute), 274	Detaecute() (openff.evaluator.protocols.forcefield.BuildLigParGenSystem method), 406
	execute() (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
method), 292	method), 401
<i>"</i>	in <b>ex/acione()</b> (openff.evaluator.protocols.forcefield.BuildTLeapSystem
attribute), 472	method), 412
	ulaxiooute() (openff.evaluator.protocols.gradients.ZeroGradients
attribute), 579	method), 418
	execute() (openff.evaluator.protocols.groups.ConditionalGroup
openff.evaluator.properties), 109	method), 425
EnthalpyOfVaporization (class in	execute() (openff.evaluator.protocols.miscellaneous.AddValues
openff.evaluator.properties), 113	method), 430
estimated_properties	$\verb"execute()" (open ff. evaluator. protocols. miscellane ous. Divide Value$
(openff.evaluator.client.RequestResult at-	method), 445
tribute), 85	${\tt execute()} \ (open {\it ff.evaluator.protocols.miscellaneous.DummyProtocol}$
<pre>estimated_properties (openff.evaluator.server.Batch</pre>	method), 460
attribute), 90	${\tt execute()} \ (open {\it ff.evaluator.protocols.miscellaneous.Filter Substance By Rosellaneous}) \ (open {\it ff.evaluator.protocols.miscellaneous.Filter Substance By Rosellaneous.Filter By Rosellane$
EvaluatorClient (class in openff.evaluator.client), 74	method), 455
EvaluatorException, 86	${\tt execute()} \ (open {\it ff.evaluator.protocols.miscellaneous.MultiplyValue}$
EvaluatorServer (class in openff.evaluator.server), 87	method), 440
ExactAmount (class in openff.evaluator.substances), 131	${\tt execute()} \ (open ff. evaluator. protocols. miscellaneous. Subtract Values$
exceptions (openff.evaluator.client.RequestResult at-	method), 435
tribute), 85	${\tt execute()} \ (open ff. evaluator. protocols. miscellaneous. Weight By Mole Fraction of the control of the $
${\tt exceptions} \ (open {\it ff.evaluator.layers.CalculationLayerRes} \\$	
attribute), 230	$\verb execute()  (open ff. evaluator. protocols. open mm. Open MMEnergy Minimis and the substitution of the $
exceptions (openff.evaluator.server.Batch attribute), 90	method), 465
exceptions (openff.evaluator.workflow.WorkflowResult	execute() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergie
attribute), 303	method), 478
	execute() (openff.evaluator.protocols.openmm.OpenMMSimulation
openff.evaluator.properties), 102	method), 472
	ie <b>lexeriaCo())</b> t( <b>mp</b> enff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase
method), 343	method), 523
method), 350	eeEnergies() (openff.evaluator.protocols.paprika.analysis.ComputeReference method), 534
<pre>execute() (openff.evaluator.protocols.analysis.AverageOb</pre>	bs <b>exeablee()</b> (openff.evaluator.protocols.paprika.analysis.ComputeSymmetry
method), 337	method), 529
execute() (openff.evaluator.protocols.analysis.BaseAvera	g <b>eDisate(D)</b> (openff.evaluator.protocols.paprika.coordinates.AddDummyAto
method), 332	method), 495
execute() (openff.evaluator.protocols.analysis.BaseDeco	r <b>ækeæPtel@</b> c6bpenff.evaluator.protocols.paprika.coordinates.PreparePullCo

execute() (openff.evaluator.protocols.analysis.BaseDecorrected (b) (b) penff.evaluator.protocols.paprika.coordinates.PreparePullC method), 360 method), 485

execute() (openff.evaluator.protocols.analysis.ComputeDipmeddata@()) (openff.evaluator.protocols.paprika.coordinates.PrepareRelease method), 355 method), 490

- execute() (openff.evaluator.protocols.analysis.Decorrelate(X)exerta(X)) (openff.evaluator.protocols.paprika.restraints.ApplyRestraints method), 370 method), 518
- execute() (openff.evaluator.protocols.analysis.Decorrelate (Frajeutor()) (openff.evaluator.protocols.paprika.restraints.GenerateAttachR method), 365 method), 501
- execute() (openff.evaluator.protocols.coordinates.BuildCoexdinates.Pa(hpenff.evaluator.protocols.paprika.restraints.GeneratePullRes. method), 376 method), 506
- execute() (openff.evaluator.protocols.coordinates.BuildDo**ekedGoo**(d)inapenff.evaluator.protocols.paprika.restraints.GenerateReleased method), 390 method), 512
- $\textbf{execute()} \ (open \textit{ff.evaluator.protocols.coordinates.Solvate} \\ \textbf{\textit{Existing.W()}} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \\ \textit{EvaluateEnergies} \\ \textbf{\textit{execute()}} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \\ \textbf{\textit{EvaluateEnergies}} \\ \textbf{\textit{execute()}} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \\ \textbf{\textit{EvaluateEnergies}} \\ \textbf{\textit{execute()}} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \\ \textbf{\textit{EvaluateEnergies}} \\ \textbf{\textit{execute()}} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \ (\textbf{\textit{topren}} \textit{ff.evaluator.protocols.reweighting.Base} \ (\textbf{\textit{topren}} \textbf{\textit{ff.evaluator.protocols.reweighting.Base} \ (\textbf{\textit{topren}} \textbf{\textit{ff.evaluator.protocols.reweighting.Base} \ (\textbf{\textit{topren}} \textbf{\textit{ff.evaluator.protocols.reweighting.Base} \ (\textbf{\textit{fo.evaluator.protocols.prot$

method), 550	openff.evaluator.datasets.curation.components.filtering),
execute() (openff.evaluator.protocols.reweighting.BaseMBAR.	
method), 556 Fil execute() (openff.evaluator.protocols.reweighting.Concatenate	terByEnvironments (class in
method), 545	199
execute() (openff.evaluator.protocols.reweighting.Concaterials	t <b>EnRyEawizr</b> onmentsSchema (class in
method), 539	$open {\it ff. evaluator. datasets. curation. components. filtering)},$
$\verb execute()  (open ff. evaluator. protocols. reweighting. Reweight Directions and the protocol of the protoc$	
	terByIonicLiquid (class in
execute() (openff.evaluator.protocols.reweighting.ReweightOb method), 561	185
execute() (openff.evaluator.protocols.simulation.BaseEneFgiyll	
method), 574 execute() (openff.evaluator.protocols.simulation.BaseSimulati	openff.evaluator.datasets.curation.components.filtering),
	terByMoleFraction (class in
execute() (openff.evaluator.protocols.storage.UnpackStoredSi	
method), 587	169
$\verb"execute()" (open ff. evaluator. protocols. yank. Base Yank Proto \textit{Exil}$	terByMoleFractionSchema (class in
method), 593	$open {\it ff. evaluator. data sets. curation. components. filtering)},$
execute() (openff.evaluator.protocols.yank.LigandReceptorYan	
	terByNComponents (class in
<pre>execute() (openff.evaluator.protocols.yank.SolvationYankProto</pre>	193
<pre>execute() (openff.evaluator.workflow.Protocol method), Fil</pre>	
308	openff.evaluator.datasets.curation.components.filtering),
execute() (openff.evaluator.workflow.ProtocolGraph method), 310 Fil	terByPressure (class in
execute() (openff.evaluator.workflow.ProtocolGroup	openff.evaluator.datasets.curation.components.filtering),
method), 313	166
execute() (openff.evaluator.workflow.Workflow Fil	terByPressureSchema (class in
method), 300	openff.evaluator.datasets.curation.components.filtering),
execute() (openff.evaluator.workflow.WorkflowGraph	164
	terByPropertyTypes (class in openff.evaluator.datasets.curation.components.filtering),
expandtabs() (openff.evaluator.storage.attributes.FilePath method), 292	177
	terByPropertyTypesSchema (class in
attribute), 454	anceBy <b>Rele</b> ff.evaluator.datasets.curation.components.filtering), 175
F	terByRacemic (class in
	openff.evaluator.datasets.curation.components.filtering),
fidelity (openff.evaluator.datasets.CalculationSource	172 terByRacemicSchema (class in
<i>ann tottle)</i> , > 0	openff.evaluator.datasets.curation.components.filtering),
FilePath (class in openff.evaluator.storage.attributes), 289	170
	terBySmiles (class in penff.evaluator.datasets.curation.components.filtering),
openff.evaluator.datasets.curation.components.filterin 182	187
	+ and last that I act also and I also a see
	terBySmilesSchema (class in
openff.evaluator.datasets.curation.components.filterin	g), openff.evaluator.datasets.curation.components.filtering), 185
openff.evaluator.datasets.curation.components.filtering 180 FilterByElements (class in Fil	g), openff.evaluator.datasets.curation.components.filtering), 185 terBySmirks (class in
openff.evaluator.datasets.curation.components.filterin 180	g), openff.evaluator.datasets.curation.components.filtering), 185 terBySmirks (class in

```
openff.evaluator.datasets.curation.components.filt&bing),_field_path(openff.evaluator.protocols.forcefield.BaseBuildSysten
                             188
                                                                                                                                                                                                              attribute), 394
FilterByStereochemistry
                                                                                                                                                                   in \quad {\tt force\_field\_path} \ (openff. evaluator. protocols. force field. Build LigPar General Control of the first protocols of the first protocols of the first protocols of the first protocol of the first protocols of the first protocols of the first protocols of the first protocol o
                                                                                                                     (class
                             openff.evaluator.datasets.curation.components.filtering),
                                                                                                                                                                                                            attribute), 406
                                                                                                                                                                                force_field_path(openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
FilterByStereochemistrySchema
                                                                                                                                (class
                                                                                                                                                                                                             attribute), 401
                                                                                                                                                                   in
                             openff.evaluator.datasets.curation.components.filtatinge_field_path (openff.evaluator.protocols.forcefield.BuildTLeapSyste
                                                                                                                                                                                                              attribute), 412
FilterBySubstances
                                                                                                           (class
                                                                                                                                                                   in force_field_path(openff.evaluator.protocols.gradients.ZeroGradients
                             openff.evaluator.datasets.curation.components.filtering),
                                                                                                                                                                                                             attribute), 416
                                                                                                                                                                                 {\tt force\_field\_path} (\textit{openff.evaluator.protocols.storage.UnpackStoredSimilar between the property of the p
                                                                                                                                                                                                             attribute), 586
FilterBySubstancesSchema
                                                                                                                       (class
                                                                                                                                                                   in
                             openff.evaluator.datasets.curation.components.filtefoirg):_field_path(openff.evaluator.protocols.yank.LigandReceptorYank.
                             193
                                                                                                                                                                                                             attribute), 598
FilterByTemperature
                                                                                                              (class
                                                                                                                                                                   in \quad force\_field\_source \ (openff.evaluator.storage.data.ForceFieldData) \ force\_field\_source \ (openff.evaluator.storage.data.Force\_field\_source \ (openff.evaluator.storage.data) \ force\_field\_source \ (openff.evaluator.sto
                             openff.evaluator.datasets.curation.components.filtering),
                                                                                                                                                                                                             attribute), 264
                                                                                                                                                                                force_field_source(openff.evaluator.storage.query.ForceFieldQuery
FilterByTemperatureSchema
                                                                                                                         (class
                                                                                                                                                                                                             attribute), 280
                             openff.evaluator.datasets.curation.components.filtErongeFieldData
                                                                                                                                                                                                                                                                                    (class
                                                                                                                                                                                                                                                                                                                                                   in
                                                                                                                                                                                                              openff.evaluator.storage.data), 263
FilterDuplicates
                                                                                                        (class
                                                                                                                                                                   in ForceFieldQuery
                                                                                                                                                                                                                                                                                      (class
                                                                                                                                                                                                                                                                                                                                                   in
                             openff.evaluator.datasets.curation.components.filtering),
                                                                                                                                                                                                             openff.evaluator.storage.query), 280
                             160
                                                                                                                                                                                ForceFieldSource
                                                                                                                                                                                                                                                                                        (class
                                                                                                                                                                                                                                                                                                                                                   in
FilterDuplicatesSchema
                                                                                                                   (class
                                                                                                                                                                                                              openff.evaluator.forcefield), 221
                             openff.evaluator.datasets.curation.components.filteformat()
                                                                                                                                                                                                                       (openff.evaluator.storage.attributes.FilePath
                                                                                                                                                                                                             method), 292
filtered_substance(openff.evaluator.protocols.miscellaficorumaEilman Gub.tampaneff.eRalluator.storage.attributes.FilePath
                             attribute), 454
                                                                                                                                                                                                             method), 292
FilterSubstanceByRole
                                                                                                                                                                   in frame_counts(openff.evaluator.protocols.reweighting.BaseMBARProtoco
                                                                                                                 (class
                             openff.evaluator.protocols.miscellaneous),
                                                                                                                                                                                                             attribute), 554
                                                                                                                                                                                 frame_counts(openff.evaluator.protocols.reweighting.ReweightDielectric
final_value_source (openff.evaluator.workflow.schemas.Workflow.Sahabute), 568
                                                                                                                                                                                 frame_counts (openff.evaluator.protocols.reweighting.ReweightObservable)
                             attribute), 322
final_value_source(openff.evaluator.workflow.Workflow
                                                                                                                                                                                                             attribute), 562
                                                                                                                                                                                 free_energy_difference
                            property), 298
find()
                                       (openff.evaluator.storage.attributes.FilePath
                                                                                                                                                                                                             (openff.evaluator.protocols.yank.BaseYankProtocol
                            method), 292
                                                                                                                                                                                                              attribute), 592
FingerPrintType
                                                                                                      (class
                                                                                                                                                                   in free_energy_difference
                             openff.evaluator.datasets.curation.components.selection), (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
                             217
                                                                                                                                                                                                             attribute), 600
force_field_id (openff.evaluator.server.Batch
                                                                                                                                                                                free_energy_difference
                                                                                                                                                                at-
                             tribute), 89
                                                                                                                                                                                                              (openff.evaluator.protocols.yank.SolvationYankProtocol
force_field_id(openff.evaluator.storage.data.BaseSimulationDataattribute), 608
                             attribute), 268
                                                                                                                                                                                free_energy_difference
force\_field\_id (\textit{openff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt@penff.evaluator.storage.data.StoredFreeEnergyDdt.gata.StoredFreeEnergyDdt.gata.StoredFreeDenergyDdt.gata.StoredFre
                                                                                                                                                                                                              attribute), 274
                             attribute), 275
force_field_id(openff.evaluator.storage.data.StoredSimuFlneioEnDewrgyDataQuery
                                                                                                                                                                                                                                                                                              (class
                                                                                                                                                                                                                                                                                                                                                   in
                             attribute), 271
                                                                                                                                                                                                             openff.evaluator.storage.query), 287
force_field_id(openff.evaluator.storage.query.BaseSimulationnDoutp@neryts()(openff.evaluator.substances.Substance
                                                                                                                                                                                                              class method), 125
                             attribute), 283
force_field_id(openff.evaluator.storage.query.FreeEner fr.Dmtalater_object()(openff.evaluator.storage.query.BaseDataQuery
                             attribute), 288
                                                                                                                                                                                                              class method), 277
force_field_id(openff.evaluator.storage.query.SimulatiofaDamaQarexyobject()(openff.evaluator.storage.query.BaseSimulationDat
                                                                                                                                                                                                             class method), 283
                             attribute), 285
```

class method), 96

- from\_data\_object() (openff.evaluator.storage.query.ForceFond(sam()) (openff.evaluator.layers.simulation.SimulationSchema class method), 280 class method), 280
- from\_data\_object() (openff.evaluator.storage.query.FreeFramgjDataQuopenff.evaluator.layers.workflow.WorkflowCalculationSchen class method), 288 class method), 236
- from\_data\_object() (openff.evaluator.storage.query.SimufinoinnjkanQuenopenff.evaluator.properties.Density class class method), 285 method), 100
- from\_doi() (openff.evaluator.datasets.thermoml.ThermoMEDamaSeton() (openff.evaluator.properties.DielectricConstant class method), 141 class method), 107
- from\_exception() (openff.evaluator.utils.exceptions.Evaluator.Eyacution(openff.evaluator.properties.EnthalpyOfMixing class method), 86 class method), 111
- from\_exception() (openff.evaluator.workflow.WorkflowExtreptionson() (openff.evaluator.properties.EnthalpyOfVaporization class method), 300 class method), 114
- from\_file() (openff.evaluator.datasets.thermoml.ThermoMtDmutgSem() (openff.evaluator.properties.ExcessMolarVolume class method), 141 class method), 104
- from\_json() (openff.evaluator.attributes.AttributeClass from\_json() (openff.evaluator.properties.HostGuestBindingAffinity class method), 620 class method), 121
- class method), 79 class method), 118
  from\_json() (openff.evaluator.client.Request class from\_json() (openff.evaluator.protocols.analysis.AverageDielectricConstant method), 81 class method), 344

- class method), 85 class method), 338
  from\_json() (openff.evaluator.datasets.CalculationSourcefrom\_json() (openff.evaluator.protocols.analysis.BaseAverageObservable
- from\_json() (openff.evaluator.datasets.MeasurementSourcErom\_json() (openff.evaluator.protocols.analysis.BaseDecorrelateProtocol class method), 98 class method), 360

class method), 332

- from\_json() (openff.evaluator.datasets.PhysicalPropertyDfrometjson() (openff.evaluator.protocols.analysis.DecorrelateObservables class method), 139 class method), 370
- from\_json() (openff.evaluator.datasets.Source class from\_json() (openff.evaluator.protocols.analysis.DecorrelateTrajectory method), 95 class method), 365
- from\_json() (openff.evaluator.datasets.taproom.TaproomPartonSetjson() (openff.evaluator.protocols.coordinates.BuildCoordinatesPactors method), 147

  class method), 377
- from\_json() (openff.evaluator.datasets.taproom.TaproomSforome\_json() (openff.evaluator.protocols.coordinates.BuildDockedCoordinates.Build
- from\_json() (openff.evaluator.datasets.thermoml.ThermoMLtdmtigSetn() (openff.evaluator.protocols.coordinates.SolvateExistingStrucclass method), 142

  class method), 383
- from\_json() (openff.evaluator.forcefield.LigParGenForceFirbitSojuscen() (openff.evaluator.protocols.forcefield.BuildLigParGenSystem class method), 227 class method), 406
- from\_json() (openff.evaluator.forcefield.TLeapForceField\( \) (openff.evaluator.protocols.forcefield.BuildTLeapSystem class method), 225 class method), 412
- from\_json() (openff.evaluator.layers.CalculationLayerResfittom\_json() (openff.evaluator.protocols.gradients.ZeroGradients class method), 230 class method), 418
- from\_json() (openff.evaluator.layers.CalculationLayerSch**£rom**\_json() (openff.evaluator.protocols.groups.ConditionalGroup class method), 232 class method), 425
- from\_json() (openff.evaluator.layers.reweighting.Reweightingthcljson() (openff.evaluator.protocols.groups.ConditionalGroup.Co

- from\_json() (openff.evaluator.protocols.miscellaneous.AddYotuejson() (openff.evaluator.protocols.simulation.BaseSimulation class method), 430 class method), 581
- from\_json() (openff.evaluator.protocols.miscellaneous.Div**ideWaljæ**on() (openff.evaluator.protocols.storage.UnpackStoredSimulation.class method), 445 class method), 587
- from\_json() (openff.evaluator.protocols.miscellaneous.DufnomPjson()) (openff.evaluator.protocols.yank.BaseYankProtocol class method), 460 class method), 593
- from\_json() (openff.evaluator.protocols.miscellaneous.FilferSmb.jtsom&y(Rplenff.evaluator.protocols.yank.LigandReceptorYankProtocols.yank.ligandReceptorYankProtocols
- from\_json() (openff.evaluator.protocols.miscellaneous.Multiphy\_Ystom() (openff.evaluator.protocols.yank.SolvationYankProtocol class method), 440 class method), 609
- from\_json() (openff.evaluator.protocols.miscellaneous.Sulfirom(vjstors()) (openff.evaluator.server.Batch class method), 435 method), 90
- from\_json() (openff.evaluator.protocols.miscellaneous.WefglatByMontFractionff.evaluator.storage.data.BaseSimulationData class method), 450 class method), 268
- from\_json() (openff.evaluator.protocols.openmm.OpenMMEonrgjsdiw()n(sqteaff.evaluator.storage.data.BaseStoredData class method), 465 class method), 261
- from\_json() (openff.evaluator.protocols.openmm.OpenMMEomlujscom() (openff.evaluator.storage.data.ForceFieldData class method), 478 class method), 264
- from\_json() (openff.evaluator.protocols.openmm.OpenMMEsiom() (openff.evaluator.storage.data.HashableStoredData class method), 472 class method), 262
- from\_json() (openff.evaluator.protocols.paprika.analysis. Arrobyzės BRP) (uspenff.evaluator.storage.data.ReplaceableData class method), 524 class method), 265
- from\_json() (openff.evaluator.protocols.paprika.analysis. **EvorpujesCafk)e(npeMffrk**valuator.storage.data.StoredFreeEnergyData class method), 534 class method), 275
- from\_json() (openff.evaluator.protocols.paprika.analysis. **EvonpujeSom(i) (trps:Colficevedlinon**tor.storage.data.StoredSimulationData class method), 529

  class method), 271
- from\_json() (openff.evaluator.protocols.paprika.coordinat**ExchaldJoon(n)**)(Appenff.evaluator.storage.query.BaseDataQuery class method), 495 class method), 277
- from\_json() (openff.evaluator.protocols.paprika.coordinaleschrejpson() (apenff.evaluator.storage.query.ForceFieldQuery class method), 490 class method), 280
- from\_json() (openff.evaluator.protocols.paprika.restraints from ly Restraints from ly Res
- from\_json() (openff.evaluator.protocols.paprika.restraintsf6omerjson() (openff.evaluator.storage.query.SimulationDataQuery class method), 501 class method), 285
- from\_json() (openff.evaluator.protocols.paprika.restraintsformerjseRv()Repartfresvaluator.storage.query.SubstanceQuery class method), 506 class method), 279
- from\_json() (openff.evaluator.protocols.paprika.restraintsformerjssekt@asqResffrenialnator.substances.Amount class class method), 512 method), 130
- from\_json() (openff.evaluator.protocols.reweighting.BaseErodnajesTore() ies(openff.evaluator.substances.Component class method), 550 class method), 128
- from\_json() (openff.evaluator.protocols.reweighting.BaseMBARFsom() (openff.evaluator.substances.ExactAmount class method), 556 class method), 132
- from\_json() (openff.evaluator.protocols.reweighting.Conclinomtej@brefy.lubpenff.evaluator.substances.MoleFraction class method), 545

  class method), 134
- from\_json() (openff.evaluator.protocols.reweighting.ConchromtejBogicCoriekopenff.evaluator.substances.Substance class method), 539 class method), 126
- from\_json() (openff.evaluator.protocols.reweighting.RewefglatDigksotr()Copsuffievaluator.thermodynamics.ThermodynamicState class method), 568 class method), 136
- from\_json() (openff.evaluator.protocols.reweighting.Rewefglandbjsordbl&openff.evaluator.utils.exceptions.EvaluatorException class method), 562 class method), 86
- from\_json() (openff.evaluator.protocols.simulation.BaseEftregyMisom(satiopenff.evaluator.workflow.Protocol class class method), 308

- from\_json() (openff.evaluator.workflow.ProtocolGroup from\_schema() (openff.evaluator.protocols.forcefield.BuildTLeapSystem class method), 313 class method), 412
- from\_json() (openff.evaluator.workflow.schemas.Protocol&romp.Schema () (openff.evaluator.protocols.gradients.ZeroGradients class method), 318 class method), 418
- from\_json() (openff.evaluator.workflow.schemas.Protocol#appincaschema() (openff.evaluator.protocols.groups.ConditionalGroup class method), 321 class method), 425
- from\_json() (openff.evaluator.workflow.schemas.Protocol\( \) from\_oschema() (openff.evaluator.protocols.miscellaneous.AddValues class method), 316 class method), 430
- from\_json() (openff.evaluator.workflow.schemas.Workflow**Sche**mas.Workflow**Sche**ma() (openff.evaluator.protocols.miscellaneous.DivideValue class method), 323 class method), 445
- from\_json() (openff.evaluator.workflow.WorkflowExceptiofirom\_schema() (openff.evaluator.protocols.miscellaneous.DummyProtococlass method), 300 class method), 460
- from\_json() (openff.evaluator.workflow.WorkflowResult class method), 303 from\_schema() (openff.evaluator.protocols.miscellaneous.FilterSubstance class method), 455
- from\_object() (openff.evaluator.forcefield.SmirnoffForceHirbitCosachema() (openff.evaluator.protocols.miscellaneous.MultiplyValue class method), 222 class method), 440
- from\_openmm() (openff.evaluator.utils.observables.Observables.Observablema() (openff.evaluator.protocols.miscellaneous.SubtractValues class method), 624 class method), 435
- from\_pandas() (openff.evaluator.datasets.PhysicalPropert\$PomaSchema() (openff.evaluator.protocols.miscellaneous.WeightByMoleR class method), 139 class method), 450
- from\_pandas() (openff.evaluator.datasets.taproom.TaproofnDomaSehema() (openff.evaluator.protocols.openmm.OpenMMEnergyMinicclass method), 147 class method), 465
- from\_pandas() (openff.evaluator.datasets.thermoml.Thermfr/Min\_pschSma() (openff.evaluator.protocols.openmm.OpenMMEvaluateEn class method), 142 class method), 478
- from\_schema() (openff.evaluator.protocols.analysis.AverageDinleschie@a(i))(openff.evaluator.protocols.paprika.analysis.AnalyzeAPRI class method), 344 class method), 524
- from\_schema() (openff.evaluator.protocols.analysis.Averagefome\_Enhemia() (openff.evaluator.protocols.paprika.analysis.ComputeRefe class method), 350 class method), 534
- from\_schema() (openff.evaluator.protocols.analysis.AverageObuseschleha() (openff.evaluator.protocols.paprika.analysis.ComputeSymclass method), 338 class method), 529
- from\_schema() (openff.evaluator.protocols.analysis.BaseAfarongeOthsemacDle(openff.evaluator.protocols.paprika.coordinates.AddDumnclass method), 332 class method), 495
- from\_schema() (openff.evaluator.protocols.analysis.BaseDecromceschema(); (openff.evaluator.protocols.paprika.coordinates.PrepareP class method), 360 class method), 485
- from\_schema() (openff.evaluator.protocols.analysis.Compr**fredhipskMema()** (openff.evaluator.protocols.paprika.coordinates.PrepareR class method), 355 class method), 490
- from\_schema() (openff.evaluator.protocols.analysis.Decorfadorae(Stokemall)e (openff.evaluator.protocols.paprika.restraints.ApplyRestractions method), 370 class method), 518
- from\_schema() (openff.evaluator.protocols.analysis.Decorfadom:Tsajvenavý) (openff.evaluator.protocols.paprika.restraints.GenerateAtt class method), 365 class method), 501
- from\_schema() (openff.evaluator.protocols.coordinates.Bufld@wosdhemesPa(depenff.evaluator.protocols.paprika.restraints.GeneratePuclass method), 377 class method), 507
- from\_schema() (openff.evaluator.protocols.coordinates.Buffdonockell@nacdinapesnff.evaluator.protocols.paprika.restraints.GenerateRe class method), 390 class method), 512
- from\_schema() (openff.evaluator.protocols.coordinates.SolfmonExistingust()) (noprenff.evaluator.protocols.reweighting.BaseEvaluateEnergy class method), 383 class method), 550
- from\_schema() (openff.evaluator.protocols.forcefield.Base BriddSschema() (openff.evaluator.protocols.reweighting.BaseMBARProtocols.governethod), 396 class method), 556
- from\_schema() (openff.evaluator.protocols.forcefield.Build\*\*Lightas@heefiga@n(openff.evaluator.protocols.reweighting.ConcatenateObseclass method), 407 class method), 545
- from\_schema() (openff.evaluator.protocols.forcefield.BuildSnoimn.ssfSignam() (openff.evaluator.protocols.reweighting.ConcatenateTrajectics method), 401 class method), 540

```
from_schema() (openff.evaluator.protocols.reweighting.Regetglatibitekets()C(openff.evaluator.substances.Substance class method), 568 method), 125
from_schema() (openff.evaluator.protocols.reweighting.RegetglatOlssibutids() (openff.evaluator.attributes.AttributeClass class method), 562 class method), 619
```

- from\_schema() (openff.evaluator.protocols.simulation.BasgEnergyWfithwiestOn(openff.evaluator.client.ConnectionOptions class method), 574 class method), 79
- from\_schema() (openff.evaluator.protocols.simulation.Basg&invalutioiibutes() (openff.evaluator.client.Request class method), 581 class method), 81
- from\_schema() (openff.evaluator.protocols.storage.Unpack@atrealSimibut@stQt(openff.evaluator.client.RequestOptions class method), 587 class method), 83
- from\_schema() (openff.evaluator.protocols.yank.BaseYank@entoentributes() (openff.evaluator.client.RequestResult class method), 593 class method), 85
- from\_schema() (openff.evaluator.protocols.yank.LigandRegeptoaYankiPowtocs()) (openff.evaluator.datasets.PhysicalProperty class method), 601 class method), 93
- from\_schema() (openff.evaluator.protocols.yank.Solvation book Partaciditutes() (openff.evaluator.layers.CalculationLayerResult class method), 609 class method), 231
- class method), 306

  class method), 232

  from\_schema() (openff.evaluator.workflow.ProtocolGroup get\_attributes() (openff.evaluator.layers.reweighting.ReweightingSchemateless method), 313

  class method), 242
- from\_schema() (openff.evaluator.workflow.Workflow get\_attributes()(openff.evaluator.layers.simulation.SimulationSchema class method), 299 class method), 238

from\_string() (openff.evaluator.datasets.PropertyPhase get\_attributes() (openff.evaluator.layers.workflow.WorkflowCalculation

- class method), 95 class method), 236 from\_url() (openff.evaluator.datasets.thermoml.ThermoM&DataStaributes() (openff.evaluator.properties.Density
- from\_url() (openff.evaluator.datasets.thermoml.ThermoM**gentalia**tributes() (openff.evaluator.properties.Density class method), 141 class method), 101
- from\_xml() (openff.evaluator.datasets.thermoml.ThermoM**bD**at**aSa**tributes() (openff.evaluator.properties.DielectricConstant class method), 142 class method), 108
- full\_path(openff.evaluator.workflow.utils.ProtocolPath get\_attributes()(openff.evaluator.properties.EnthalpyOfMixing property), 328
  class method), 111
- full\_substance (openff.evaluator.protocols.miscellaneous Weight By Whale Esc () t (oppenff.evaluator.properties. Enthalpy Of Vaporization attribute), 449 class method), 115

## G class meth generate\_base\_reweighting\_protocols() (in mod-get\_attributes(

generate\_default\_metadata()
 (openff.evaluator.workflow.Workflow static
 method), 299

ule openff.evaluator.protocols.utils), 616

- GenerateAttachRestraints (class in openff.evaluator.protocols.paprika.restraints), 498
- GeneratePullRestraints (class in openff.evaluator.protocols.paprika.restraints), 504
- GenerateReleaseRestraints (class in openff.evaluator.protocols.paprika.restraints), 509

- class method), 122

- get\_attributes() (openff.evaluator.protocols.analysis.AverageFreeEnergelass method), 350

- get\_attributes() (openff.evaluator.protocols.analysis.BaseDecorrelatePass method), 360

get() (openff.evaluator.utils.observables.ObservableFrameget\_attributes() (openff.evaluator.protocols.analysis.DecorrelateTrajecture), 625
class method), 365

- get\_attributes() (openff.evaluator.protocols.coordinate.gBuilatCuridiutessPackpeolff.evaluator.protocols.paprika.restraints.Generat class method), 377 class method), 501
- get\_attributes() (openff.evaluator.protocols.coordinates@BtilatDarkbulCos@inapesnff.evaluator.protocols.paprika.restraints.Generat class method), 390 class method), 507
- get\_attributes() (openff.evaluator.protocols.coordinate.getlvataFribuitgSt() (toppenff.evaluator.protocols.paprika.restraints.General class method), 383 class method), 512
- get\_attributes() (openff.evaluator.protocols.forcefield.BystBattlfixtures() (openff.evaluator.protocols.reweighting.BaseEvaluater.class method), 396

  class method), 551
- get\_attributes() (openff.evaluator.protocols.forcefield.ByeltdlagRariGertEps(en(openff.evaluator.protocols.reweighting.BaseMBARPr class method), 407 class method), 556
- get\_attributes() (openff.evaluator.protocols.forcefield.ByeldSwtituriffSystem() (openff.evaluator.protocols.reweighting.ConcatenateC class method), 401 class method), 545
- get\_attributes() (openff.evaluator.protocols.forcefield.ByeldTdvetpSbytens() (openff.evaluator.protocols.reweighting.ConcatenateToclass method), 412 class method), 540
- get\_attributes() (openff.evaluator.protocols.gradients.ZgetGatdieibutes() (openff.evaluator.protocols.reweighting.ReweightDielectus method), 418 class method), 568
- get\_attributes() (openff.evaluator.protocols.groups.Cor**gliti**oattlCricbuptes() (openff.evaluator.protocols.reweighting.ReweightObseclass method), 425 class method), 562
- get\_attributes() (openff.evaluator.protocols.groups.Cor**gliti**oathCricbutEsh()i(ippenff.evaluator.protocols.simulation.BaseEnergyMin class method), 422 class method), 574
- get\_attributes() (openff.evaluator.protocols.miscellane@est\_Adtl\rubbastes() (openff.evaluator.protocols.simulation.BaseSimulation.class method), 430 class method), 582
- get\_attributes() (openff.evaluator.protocols.miscellane@yestDixtitleVolumees() (openff.evaluator.protocols.storage.UnpackStoredSimu class method), 445
- get\_attributes() (openff.evaluator.protocols.miscellane@est\_Dattrip\_Buttes() (openff.evaluator.protocols.yank.BaseYankProtocol class method), 460 class method), 594
- get\_attributes() (openff.evaluator.protocols.miscellane@estFiltev&ibstues&ByRplenff.evaluator.protocols.yank.LigandReceptorYank.

  class method), 455

  class method), 601

- get\_attributes() (openff.evaluator.protocols.miscellane@ext\_WarightByMels\_F)(ctponff.evaluator.storage.data.BaseSimulationData class method), 450 class method), 268
- get\_attributes() (openff.evaluator.protocols.openmm.OpertMalEnakyyveisi()n(squienff.evaluator.storage.data.BaseStoredData class method), 465 class method), 261
- get\_attributes() (openff.evaluator.protocols.openmm.OpertMtSimilhatios() (openff.evaluator.storage.data.HashableStoredData class method), 472 class method), 262
- get\_attributes() (openff.evaluator.protocols.paprika.an**g**k<u>xis</u>a**AnatisheAPRPh(usp**enff.evaluator.storage.data.ReplaceableData class method), 524 class method), 266
- get\_attributes() (openff.evaluator.protocols.paprika.an**glyxisaCturiphoteResf@e(ngeMiffik**valuator.storage.data.StoredFreeEnergyData class method), 534 class method), 275
- get\_attributes() (openff.evaluator.protocols.paprika.an**gl**stisaCturiphovetSestinktry&rifficerctlioontor.storage.data.StoredSimulationData class method), 529 class method), 271
- get\_attributes() (openff.evaluator.protocols.paprika.co**get**in**ates.riddfless(n)**/(**topenf**f.evaluator.storage.query.BaseDataQuery class method), 496 class method), 277
- get\_attributes() (openff.evaluator.protocols.paprika.co**grdinates.fl)roptiesf()|(Openffinates**ator.storage.query.BaseSimulationDataQclass method), 485

  class method), 283
- get\_attributes() (openff.evaluator.protocols.paprika.co**getinates:FibrotiesRel.cape**Offordihattor.storage.query.ForceFieldQuery class method), 490 class method), 281
- get\_attributes() (openff.evaluator.protocols.paprika.res**get**in**tstApplyReess(i)n(o**penff.evaluator.storage.query.FreeEnergyDataQuer\_class method), 518 class method), 288

```
get_attributes() (openff.evaluator.storage.query.SimulationDataQmethod), 377
         class method), 285
                                                       get_class_attribute()
get_attributes() (openff.evaluator.storage.query.SubstanceQuery(openff.evaluator.protocols.coordinates.BuildDockedCoordinates
                                                                method), 390
         class method), 279
get_attributes() (openff.evaluator.substances.Amount get_class_attribute()
         class method), 130
                                                                (openff.evaluator.protocols.coordinates.SolvateExistingStructure
get_attributes() (openff.evaluator.substances.Component
                                                                method), 383
                                                       get_class_attribute()
         class method), 128
get_attributes() (openff.evaluator.substances.ExactAmount
                                                                (open ff. evaluator. protocols. force field. Base Build System
                                                                method), 396
         class method), 132
get_attributes() (openff.evaluator.substances.MoleFrac@et_class_attribute()
         class method), 134
                                                                (openff.evaluator.protocols.forcefield.BuildLigParGenSystem
                                                                method), 407
get_attributes() (openff.evaluator.substances.Substance
         class method), 127
                                                       get_class_attribute()
get_attributes() (openff.evaluator.thermodynamics.Thermodynam(inSenff.evaluator.protocols.forcefield.BuildSmirnoffSystem
         class method), 136
                                                                method), 401
get_attributes() (openff.evaluator.workflow.Protocol get_class_attribute()
         class method), 308
                                                                (openff.evaluator.protocols.forcefield.BuildTLeapSystem
get_attributes() (openff.evaluator.workflow.ProtocolGroup
                                                                method), 413
         class method), 313
                                                       get_class_attribute()
get_attributes() (openff.evaluator.workflow.schemas.ProtocolGroupEnlfrevaluator.protocols.gradients.ZeroGradients
         class method), 318
                                                                method), 418
get_attributes() (openff.evaluator.workflow.schemas.Prgencotsdammattribute()
         class method), 317
                                                                (openff.evaluator.protocols.groups.ConditionalGroup
get_attributes() (openff.evaluator.workflow.schemas.WorkflowSchmathod), 425
         class method), 323
                                                       get_class_attribute()
get_attributes() (openff.evaluator.workflow.WorkflowResult
                                                                (openff.evaluator.protocols.miscellaneous.AddValues
         class method), 303
                                                                method), 430
get_class_attribute()
                                                       get_class_attribute()
         (openff.evaluator.protocols.analysis.AverageDielectricConstantenff.evaluator.protocols.miscellaneous.DivideValue
         method), 344
                                                                method), 445
get_class_attribute()
                                                       get_class_attribute()
         (openff.evaluator.protocols.analysis.AverageFreeEnergies (openff.evaluator.protocols.miscellaneous.DummyProtocol
         method), 350
                                                                method), 460
get_class_attribute()
                                                       get_class_attribute()
         (open ff. evaluator. protocols. analysis. Average Observable
                                                                (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRole
         method), 338
                                                                method), 455
get_class_attribute()
                                                       get_class_attribute()
         (openff.evaluator.protocols.analysis.BaseAverageObservableopenff.evaluator.protocols.miscellaneous.MultiplyValue
         method), 333
                                                                method), 440
get_class_attribute()
                                                       get_class_attribute()
         (open ff. evaluator. protocols. analysis. Base Decorrelate Protocol benefi. evaluator. protocols. miscellaneous. Subtract Values
         method), 360
                                                                method), 435
get_class_attribute()
                                                       get_class_attribute()
         (openff.evaluator.protocols.analysis.ComputeDipoleMoment@openff.evaluator.protocols.miscellaneous.WeightByMoleFraction
         method), 355
                                                                method), 450
get_class_attribute()
                                                       get_class_attribute()
         (openff.evaluator.protocols.analysis.DecorrelateObservable(openff.evaluator.protocols.openmm.OpenMMEnergyMinimisatio
         method), 370
                                                                method), 466
                                                       get_class_attribute()
get_class_attribute()
         (openff.evaluator.protocols.analysis.DecorrelateTrajectory (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies
                                                                method), 479
         method), 365
get_class_attribute()
                                                       get_class_attribute()
```

(openff.evaluator.protocols.coordinates.BuildCoordinatesPachpeolff.evaluator.protocols.openmm.OpenMMSimulation

```
method), 472
                                                                                                       method), 582
get_class_attribute()
                                                                                        get_class_attribute()
              (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPh(xxpenff.evaluator.protocols.storage.UnpackStoredSimulationData
              method), 524
                                                                                                       method), 587
get_class_attribute()
                                                                                        get_class_attribute()
              (open \textit{ff.} evaluator. protocols. paprika. analysis. Compute \textit{Refere} (\textit{tage} \& \textit{M} \textit{ffr} \textit{k} \textit{valuator}. protocols. yank. \textit{Base} \textit{Yank} \textit{Protocol}
              method), 534
                                                                                                       method), 594
get_class_attribute()
                                                                                        get_class_attribute()
              (openff.evaluator.protocols.paprika.analysis.ComputeSymm&try&tryff.evalliaoutor.protocols.yank.LigandReceptorYankProtocol
              method), 529
                                                                                                       method), 601
get_class_attribute()
                                                                                        get_class_attribute()
               (openff.evaluator.protocols.paprika.coordinates.AddDummy(Astpemff.evaluator.protocols.yank.SolvationYankProtocol
                                                                                                       method), 610
              method), 496
get_class_attribute()
                                                                                         get_class_attribute()
              (openff.evaluator.protocols.paprika.coordinates.PreparePullGpomffinuates.ator.workflow.Protocol method),
              method), 485
get_class_attribute()
                                                                                         get_class_attribute()
              (openff.evaluator.protocols.paprika.coordinates.PrepareRel@ape@ff.avdihatesr.workflow.ProtocolGroup
              method), 490
                                                                                                       method), 313
                                                                                        get_molecules_per_component()
get_class_attribute()
              (openff.evaluator.protocols.paprika.restraints.ApplyRestraintopenff.evaluator.substances.Substance
              method), 518
                                                                                                       method), 126
get_class_attribute()
                                                                                         get_value() (openff.evaluator.protocols.analysis.AverageDielectricConsta
              (openff.evaluator.protocols.paprika.restraints.GenerateAttaohRtesort)jints44
              method), 501
                                                                                        get_value() (openff.evaluator.protocols.analysis.AverageFreeEnergies
get_class_attribute()
                                                                                                       method), 350
               (openff.evaluator.protocols.paprika.restraints.GengeatePalh&&i)-úiptsnff.evaluator.protocols.analysis.AverageObservable
              method), 507
                                                                                                       method), 338
get_class_attribute()
                                                                                        get_value() (openff.evaluator.protocols.analysis.BaseAverageObservable
              (openff.evaluator.protocols.paprika.restraints.GenerateReleauseRecal)วุลชิงใ
              method), 512
                                                                                         get_value() (openff.evaluator.protocols.analysis.BaseDecorrelateProtocol
get_class_attribute()
                                                                                                       method), 360
              (openff.evaluator.protocols.reweighting.BaseEvalugutEnergies() (openff.evaluator.protocols.analysis.ComputeDipoleMoment.
              method), 551
                                                                                                       method), 355
                                                                                        get_value() (openff.evaluator.protocols.analysis.DecorrelateObservables
get_class_attribute()
              (openff.evaluator.protocols.reweighting.BaseMBARProtocolmethod), 370
              method), 556
                                                                                        get_value() (openff.evaluator.protocols.analysis.DecorrelateTrajectory
get_class_attribute()
                                                                                                       method), 365
              (openff.evaluator.protocols.reweighting.Concatenage Obsahuel Os(openff.evaluator.protocols.coordinates.BuildCoordinatesPac
              method), 545
                                                                                                       method), 377
get_class_attribute()
                                                                                        get_value() (openff.evaluator.protocols.coordinates.BuildDockedCoordin
               (openff.evaluator.protocols.reweighting.ConcatenateTrajectoniathod), 391
              method), 540
                                                                                        get_value() (openff.evaluator.protocols.coordinates.SolvateExistingStruc
get_class_attribute()
                                                                                                       method), 384
              (openff.evaluator.protocols.reweighting.ReweightDietectel.tom) (openff.evaluator.protocols.forcefield.BaseBuildSystem
              method), 568
                                                                                                       method), 396
get_class_attribute()
                                                                                         \verb"get_value"()" (open ff. evaluator. protocols. force field. Build Lig Par Gen System) and the substitution of the substitut
              (openff.evaluator.protocols.reweighting.ReweightObservablenethod), 407
              method), 562
                                                                                        get_value() (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
get_class_attribute()
                                                                                                        method), 401
              (openff.evaluator.protocols.simulation.BaseEnerggleftnimalsuuch) (openff.evaluator.protocols.forcefield.BuildTLeapSystem
              method), 575
                                                                                                       method), 413
get_class_attribute()
                                                                                        get_value() (openff.evaluator.protocols.gradients.ZeroGradients
```

method), 419

(openff.evaluator.protocols.simulation.BaseSimulation

```
\verb|get_value()| (open ff. evaluator. protocols. groups. Condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition \verb|gktrowplue()| (open ff. evaluator. protocols. simulation. Base Energy Minimis and the condition of the condition 
                       method), 425
                                                                                                                                                                     method), 575
get_value() (openff.evaluator.protocols.miscellaneous.Addetuealue() (openff.evaluator.protocols.simulation.BaseSimulation
                                                                                                                                                                     method), 582
                       method), 430
get_value() (openff.evaluator.protocols.miscellaneous.Divgetvalue() (openff.evaluator.protocols.storage.UnpackStoredSimulation.
                       method), 445
                                                                                                                                                                     method), 588
get_value() (openff.evaluator.protocols.miscellaneous.DuganyRadroed) (openff.evaluator.protocols.yank.BaseYankProtocol
                                                                                                                                                                     method), 594
                       method), 460
get_value() (openff.evaluator.protocols.miscellaneous.FilgerSubstitue&ByRplenff.evaluator.protocols.yank.LigandReceptorYankProtocols.
                                                                                                                                                                     method), 601
                       method), 456
get_value() (openff.evaluator.protocols.miscellaneous.Mudtiplywallue() (openff.evaluator.protocols.yank.SolvationYankProtocol
                       method), 440
                                                                                                                                                                     method), 610
get_value() (openff.evaluator.protocols.miscellaneous.SulgeticNalues()
                                                                                                                                                                                                 (openff.evaluator.workflow.Protocol
                       method), 435
                                                                                                                                                                     method), 307
get_value() (openff.evaluator.protocols.miscellaneous.WeigehtByAlweF)a(ajpanff.evaluator.workflow.ProtocolGroup
                        method), 451
                                                                                                                                                                     method), 313
get_value() (openff.evaluator.protocols.openmm.OpenMMEnergalMinismessationces()
                       method), 466
                                                                                                                                                                     (openff.evaluator.protocols.analysis.AverageDielectricConstant
get_value() (openff.evaluator.protocols.openmm.OpenMMEvaluate Encellagie), 344
                       method), 479
                                                                                                                                             get_value_references()
get_value() (openff.evaluator.protocols.openmm.OpenMMSimulationpenff.evaluator.protocols.analysis.AverageFreeEnergies
                       method), 473
                                                                                                                                                                     method), 350
get_value() (openff.evaluator.protocols.paprika.analysis.ApatlyvaABRPhaserences()
                       method), 524
                                                                                                                                                                     (openff.evaluator.protocols.analysis.AverageObservable
get_value() (openff.evaluator.protocols.paprika.analysis.ComputeRaferlande,Work
                       method), 534
                                                                                                                                             get_value_references()
get_value() (openff.evaluator.protocols.paprika.analysis.ComputeSyopmaff.yQahnataiqprotocols.analysis.BaseAverageObservable
                                                                                                                                                                     method), 333
                       method), 529
get_value() (openff.evaluator.protocols.paprika.coordinatex.td.dtiDumenneMornsnces()
                                                                                                                                                                     (openff.evaluator.protocols.analysis.BaseDecorrelateProtocol
                       method), 496
get_value() (openff.evaluator.protocols.paprika.coordinates.Preparalethbaoordinates
                       method), 485
                                                                                                                                             get_value_references()
get_value() (openff.evaluator.protocols.paprika.coordinates.PrepareMednffse Cabandionaptestocols.analysis.ComputeDipoleMoments
                                                                                                                                                                     method), 356
                       method), 490
get_value() (openff.evaluator.protocols.paprika.restraintsq&pplw&Pruka;iretferences()
                       method), 518
                                                                                                                                                                     (openff.evaluator.protocols.analysis.DecorrelateObservables
get_value() (openff.evaluator.protocols.paprika.restraints.Generater that the handle of the control of the cont
                       method), 501
                                                                                                                                             get_value_references()
get_value() (openff.evaluator.protocols.paprika.restraints.Generate@pleRffsexailutestor.protocols.analysis.DecorrelateTrajectory
                       method), 507
                                                                                                                                                                     method), 366
get_value() (openff.evaluator.protocols.paprika.restraintsg6tn.mailtalecleasceRestaciss(s)
                       method), 512
                                                                                                                                                                     (openff.evaluator.protocols.coordinates.BuildCoordinatesPackmo
get_value() (openff.evaluator.protocols.reweighting.BaseEvaluateEnerthied), 377
                       method), 551
                                                                                                                                             get_value_references()
\verb|get_value()| (open ff. evaluator. protocols. reweighting. Base MBARP rotocols. coordinates. Build Docked Coordinates and the protocols of 
                                                                                                                                                                     method), 391
                        method), 557
get_value() (openff.evaluator.protocols.reweighting.Concgreva.acatheences()
                       method), 545
                                                                                                                                                                     (openff.evaluator.protocols.coordinates.SolvateExistingStructure
get_value() (openff.evaluator.protocols.reweighting.ConcatenateTrajecthord), 384
                                                                                                                                             get_value_references()
                       method), 540
get_value() (openff.evaluator.protocols.reweighting.ReweightDieledtrpediffnestaluator.protocols.forcefield.BaseBuildSystem
                                                                                                                                                                     method), 396
                       method), 569
get_value() (openff.evaluator.protocols.reweighting.ReweightiObakumalhieferences()
                                                                                                                                                                     (openff.evaluator.protocols.forcefield.BuildLigParGenSystem
                       method), 562
```

T 10 10 =		1 P 40 C
method), 407	aa+1	method), 496
<u>-</u>	-	ue_references() (openff.evaluator.protocols.paprika.coordinates.PreparePullCoor
method), 402	ŋjsysiem	method), 485
	get val	ue_references()
(openff.evaluator.protocols.forcefield.BuildTLeapS	-	(openff.evaluator.protocols.paprika.coordinates.PrepareReleaseC
method), 413	- ,	method), 490
	get_val	ue_references()
(openff.evaluator.protocols.gradients.ZeroGradien	nts	(openff.evaluator.protocols.paprika.restraints.ApplyRestraints
method), 419		method), 518
	-	ue_references()
(openff.evaluator.protocols.groups.ConditionalGramethod), 423	оир	(openff.evaluator.protocols.paprika.restraints.GenerateAttachRes.method), 501
<pre>get_value_references()</pre>	get_val	ue_references()
(openff.evaluator.protocols.miscellaneous.AddValumethod), 430	ues	(openff.evaluator.protocols.paprika.restraints.GeneratePullRestramethod), 507
<pre>get_value_references()</pre>	get_val	ue_references()
(openff.evaluator.protocols.miscellaneous.Dividel method), 445	/alue	(openff.evaluator.protocols.paprika.restraints.GenerateReleaseRemethod), 512
<pre>get_value_references()</pre>	get_val	ue_references()
(openff.evaluator.protocols.miscellaneous.Dummy method), 460	Protocol	(openff.evaluator.protocols.reweighting.BaseEvaluateEnergies method), 551
<pre>get_value_references()</pre>	get_val	ue_references()
(openff.evaluator.protocols.miscellaneous.FilterSimethod), 456	ubstanceB	RyRpenff.evaluator.protocols.reweighting.BaseMBARProtocol method), 557
<pre>get_value_references()</pre>	get_val	ue_references()
(openff.evaluator.protocols.miscellaneous.Multipl	yValue	(open ff. evaluator. protocols. reweighting. Concatenate Observables
method), 440	_	method), 545
-	-	ue_references()
method), 436		$(open \textit{ff.evaluator.protocols.reweighting.} Concate nate \textit{Trajectories} \\ \textit{method}), 540$
<u>-</u>	-	ue_references()
method), 451		d <b>ctpon</b> ff.evaluator.protocols.reweighting.ReweightDielectricConstant method), 569
		ue_references()
	ergyMinin	n(sptenff.evaluator.protocols.reweighting.ReweightObservable
method), 466		method), 562
	-	ue_references() rgopenff.evaluator.protocols.simulation.BaseEnergyMinimisation
method), 479		method), 575
<u>-</u>	-	ue_references()
method), 473		(openff.evaluator.protocols.simulation.BaseSimulation method), 582
=	_	ue_references()
method), 524		h <b>(xsp</b> enff.evaluator.protocols.storage.UnpackStoredSimulationData method), 588
=	_	ue_references()
(openff.evaluator.protocols.paprika.analysis.Comp method), 535	puteRefer	e(wpeMffrkvaluator.protocols.yank.BaseYankProtocol method), 594
=	_	ue_references()
method), 530		Actipación le executivantor.protocols.yank.LigandReceptorYankProtocol method), 601
=	_	ue_references()
(openff evaluator protocols paprika coordinates A	ddDumm	vActoremsf evaluator protocols vank SolvationYankProtocol

method), 610	gradients (openff.evaluator.workflow.WorkflowResult
<pre>get_value_references()</pre>	attribute), 303
(openff.evaluator.workflow.Protocol method),	
307	(openff.evaluator.protocols.paprika.coordinates.PreparePullCoor
<pre>get_value_references()</pre>	attribute), 483
(openff.evaluator.workflow.ProtocolGroup method), 312	Н
${\tt gpu\_device\_indices} \ (\textit{openff.evaluator.backends.Computations}) \ \ \textit{computations} \ \ \ \textit{computations} \ \ \textit{computations} \ \ \textit{computations} \ \ \ \ \textit{computations} \ \ \ \textit{computations} \ \ \ \textit{computations} \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \$	uteResouteCtllary_data()
property), 246	(openff.evaluator.storage.data.Base Simulation Data
${\tt gpu\_device\_indices} \ (open {\it ff.evaluator.backends.QueueVelocity}) \\$	WorkerResoนุเลยูร method), 268
property), 247	<pre>has_ancillary_data()</pre>
<pre>gradient_parameters</pre>	(openff. evaluator. storage. data. Base Stored Data
(openff. evaluator. protocols. analysis. Compute Dip	poleMoments <sub>lass</sub> method), 261
attribute), 354	<pre>has_ancillary_data()</pre>
<pre>gradient_parameters</pre>	(openff.evaluator.storage.data.ForceFieldData
(open ff. evaluator. protocols. gradients. Zero Gradients and the protocol of the protocol o	ents class method), 264
attribute), 417	has_ancillary_data()
<pre>gradient_parameters</pre>	(openff.evaluator.storage.data.HashableStoredData
(openff. evaluator. protocols. openmm. Open MME valuation of the protocol states of the p	valuateEnergiess method), 262
attribute), 479	has_ancillary_data()
<pre>gradient_parameters</pre>	(openff.evaluator.storage.data.ReplaceableData
(openff.evaluator.protocols.openmm.OpenMMSin	
attribute), 473	has_ancillary_data()
<pre>gradient_parameters</pre>	(openff.evaluator.storage.data.StoredFreeEnergyData
(openff. evaluat or. protocols. reweighting. Base Evaluation and the protocol stress of t	uluateEnergieguss method), 275
attribute), 549	has_ancillary_data()
gradient_parameters	(openff.evaluator.storage.data.StoredSimulationData
(openff.evaluator.protocols.simulation.BaseSimul	
attribute), 580	has_force_field() (openff.evaluator.storage.LocalFileStorage
gradient_parameters	method), 258
(openff.evaluator.protocols.yank.BaseYankProtoc	colas_force_field() (openff.evaluator.storage.StorageBackend
attribute), 592	method), 257
gradient_parameters	has_object() (openff.evaluator.storage.LocalFileStorage
(openff.evaluator.protocols.yank.LigandReceptor	rYankProtocolethod) 258
attribute), 601	has_object() (openff.evaluator.storage.StorageBackend
gradient_parameters	method), 257
(openff.evaluator.protocols.yank.SolvationYankPa	PrHasMableStoredData (class in
attribute), 610	openff.evaluator.storage.data), 262
<pre>gradients (openff.evaluator.datasets.PhysicalProperty</pre>	high_precision(openff.evaluator.protocols.openmm.OpenMMSimulation
attribute), 93	attribute), 473
gradients (openff.evaluator.properties.Density at-	high_precision(openff.evaluator.protocols.simulation.BaseSimulation
tribute), 101	attribute), 580
${\tt gradients}. {\it Open ff. evaluator. properties. Die lectric Constant of the properties of the prop$	nthost coordinate nath
attribute), 108	(openff.evaluator.protocols.paprika.restraints.GenerateReleaseRe
gradients (openff.evaluator.properties.EnthalpyOfMixing	
attribute), 111	HostGuestBindingAffinity (class in
gradients (openff.evaluator.properties.EnthalpyOfVapori	rization openff.evaluator.properties), 119
attribute), 115	openy.evanacionproperties), 119
gradients (openff.evaluator.properties.ExcessMolarVolum	mk
attribute), 104	· 1 /
gradients (openff.evaluator.properties.HostGuestBinding	id (openff.evaluator.client.Request attribute), 81 gAffinity ad (openff.evaluator.datasets.PhysicalProperty attribute),
gradients (openff.evaluator.properties.SolvationFreeEner	ergyd (onguff gyglygton proportion Dougity, gytributs) 101
attribute) 118	Tu (openy).evaluator.properties.Density attribute), 101

id	(openff.evaluator.properties.DielectricConstant attribute), 108		f.evaluator.protocols.miscellaneous.MultiplyValue attribute), 441
id	(openff.evaluator.properties.EnthalpyOfMixing attribute), 111	id(openf)	f.evaluator.protocols.miscellaneous.SubtractValues attribute), 436
id	(openff.evaluator.properties.EnthalpyOfVaporization attribute), 115	id(openf)	f.evaluator.protocols.miscellaneous.WeightByMoleFraction attribute), 451
id	(openff.evaluator.properties.ExcessMolarVolume attribute), 104	id(openf)	f.evaluator.protocols.openmm.OpenMMEnergyMinimisation attribute), 466
id	(openff.evaluator.properties.HostGuestBindingAffinity attribute), 122	id(openf)	f.evaluator.protocols.openmm.OpenMMEvaluateEnergies attribute), 479
id	(openff.evaluator.properties.SolvationFreeEnergy attribute), 118	id(openf)	f.evaluator.protocols.openmm.OpenMMSimulation attribute), 473
id		o <b>iod</b> t( <b>op</b> tenf)	f.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase attribute), 525
id	(openff.evaluator.protocols.analysis.AverageFreeEnergie attribute), 351		f.evaluator.protocols.paprika.analysis.ComputeReferenceWork attribute), 535
id	(openff.evaluator.protocols.analysis.AverageObservable attribute), 339		f.evaluator.protocols.paprika.analysis.ComputeSymmetryCorrection attribute), 530
id	(openff.evaluator.protocols.analysis.BaseAverageObserv attribute), 333		f.evaluator.protocols.paprika.coordinates.AddDummyAtoms attribute), 496
id	(openff.evaluator.protocols.analysis.BaseDecorrelatePro attribute), 361		f.evaluator.protocols.paprika.coordinates.PreparePullCoordinates attribute), 486
id	(openff.evaluator.protocols.analysis.ComputeDipoleMonattribute), 356		f.evaluator.protocols.paprika.coordinates.PrepareReleaseCoordin attribute), 491
	(openff.evaluator.protocols.analysis.DecorrelateObserva attribute), 371		attribute), 519
	attribute), 366		F.evaluator.protocols.paprika.restraints.GenerateAttachRestraints attribute), 502
	attribute), 378		F.evaluator.protocols.paprika.restraints.GeneratePullRestraints attribute), 507
	attribute), 391		f.evaluator.protocols.paprika.restraints.GenerateReleaseRestraint. attribute), 513
id	(openff.evaluator.protocols.coordinates.SolvateExistingS attribute), 384		f.evaluator.protocols.reweighting.BaseEvaluateEnergies attribute), 551
	(openff.evaluator.protocols.forcefield.BaseBuildSystem attribute), 396		attribute), 557
	attribute), 407	. 1 00	f.evaluator.protocols.reweighting.ConcatenateObservables attribute), 546
	attribute), 402		f.evaluator.protocols.reweighting.ConcatenateTrajectories attribute), 540
id	(openff.evaluator.protocols.forcefield.BuildTLeapSystem attribute), 413		f.evaluator.protocols.reweighting.ReweightDielectricConstant attribute), 569
id	(openff.evaluator.protocols.gradients.ZeroGradients attribute), 419		f.evaluator.protocols.reweighting.ReweightObservable attribute), 563
	(openff.evaluator.protocols.groups.ConditionalGroup attribute), 426		f.evaluator.protocols.simulation.BaseEnergyMinimisation attribute), 575
	(openff.evaluator.protocols.miscellaneous.AddValues attribute), 431		f.evaluator.protocols.simulation.BaseSimulation attribute), 582
	(openff.evaluator.protocols.miscellaneous.DivideValue attribute), 446		f.evaluator.protocols.storage.UnpackStoredSimulationData attribute), 588
	(openff.evaluator.protocols.miscellaneous.DummyProtoc attribute), 461		tribute), 594
id	(openff.evaluator.protocols.miscellaneous.FilterSubstance attribute) 456		f.evaluator.protocols.yank.LigandReceptorYankProtocol

${\tt id}$ (openff.evaluator.protocols.yank.Solvation Yank Protocol	input_coordinate_paths
attribute), 610	$(open {\it ff. evaluator. protocols. reweighting. Concatenate Trajectories}$
id (openff.evaluator.server.Batch attribute), 89	attribute), 538
id (openff.evaluator.workflow.Protocol attribute), 306	input_observables(openff.evaluator.protocols.analysis.DecorrelateObservables)
id (openff.evaluator.workflow.ProtocolGroup attribute),	attribute), 368
314	<pre>input_observables(openff.evaluator.protocols.gradients.ZeroGradients</pre>
${\tt id}\ (open ff. evaluator. work flow. schemas. Protocol Group Sche$	rma attribute), 416
attribute), 318 id (openff.evaluator.workflow.schemas.ProtocolSchema	<pre>input_observables(openff.evaluator.protocols.reweighting.Concatenate</pre>
attribute), 316	input_substance(openff.evaluator.protocols.miscellaneous.FilterSubstan
identifier (openff.evaluator.substances.Amount prop-	attribute), 453
erty), 130	input_system(openff.evaluator.protocols.paprika.coordinates.AddDumm
identifier (openff.evaluator.substances.Component property), 128	attribute), 494 input_system(openff.evaluator.protocols.paprika.restraints.ApplyRestrain
identifier (openff.evaluator.substances.ExactAmount	attribute), 516
property), 131	input_trajectory_path
identifier (openff.evaluator.substances.MoleFraction	(openff.evaluator.protocols.analysis.DecorrelateTrajectory
property), 133	attribute), 363
identifier (openff.evaluator.substances.Substance	input_trajectory_paths
property), 125	(openff.evaluator.protocols.reweighting.ConcatenateTrajectories
ImportFreeSolv (class in	attribute), 538
	e and ut, value (openff.evaluator.protocols.miscellaneous.DummyProtocol
202	attribute), 458
ImportFreeSolvSchema (class in	InputAttribute (class in
openff.evaluator.datasets.curation.components.fr	reesolv), openff.evaluator.workflow.attributes), 325
200	$\verb"inputs" (open ff. evaluator. work flow. schemas. Protocol Group Schema")$
ImportThermoMLData (class in	attribute), 319
openff.evaluator.datasets.curation.components.th	ne <b>ˈːmpwt/3,</b> (openff.evaluator.workflow.schemas.ProtocolSchema
205	attribute), 316
	$\verb"inverse_beta" (open ff. evaluator. the rmodynamics. The rmodynamic State")$
openff.evaluator.datasets.curation.components.th	nermoml), property), 135
203	isalnum() (openff.evaluator.storage.attributes.FilePath
index() (openff.evaluator.storage.attributes.FilePath	method), 292
method), 292	isalpha() (openff.evaluator.storage.attributes.FilePath
InequalityMergeBehavior (class in	method), 292
openff.evaluator.workflow.attributes), 325	isascii() (openff.evaluator.storage.attributes.FilePath
input_coordinate_file	method), 292
	Trisedeoimal() (openff.evaluator.storage.attributes.FilePath
attribute), 363	method), 292
input_coordinate_file	isdigit() (openff.evaluator.storage.attributes.FilePath
(openff.evaluator.protocols.openmm.OpenMMEn	••
attribute), 466	isidentifier() (openff.evaluator.storage.attributes.FilePath
input_coordinate_file	method), 293
	niukstriower() (openff.evaluator.storage.attributes.FilePath
attribute), 473	method), 293
input_coordinate_file	isnumeric() (openff.evaluator.storage.attributes.FilePath
(openff.evaluator.protocols.simulation.BaseEnerg	
attribute), 572	isprintable() (openff.evaluator.storage.attributes.FilePath
input_coordinate_file	method), 293
attribute), 579	laisspace() (openff.evaluator.storage.attributes.FilePath method), 293
input_coordinate_path	istitle() (openff.evaluator.storage.attributes.FilePath
nput_coorumate_patn (openff.evaluator.protocols.paprika.coordinates./	
attribute) 494	isunner() (openff evaluator storage attributes FilePath

	method), 293		(open ff.evaluator. datasets. curation. components. filterByState filterByStat
items()	(openff.evaluator.utils.observables.ObservableFra		method), 179
	method), 625		(openff.evaluator.datasets.curation.components.filtering.FilterBySumethod), 195
J			(open ff.evaluator. datasets. curation. components. filterByTe
job_scr	ript() (openff.evaluator.backends.dask.BaseDask. method), 250	JobQueu json()	re <b>Bucklerd</b> ), <sup>163</sup> (openff.evaluator.datasets.curation.components.filtering.FilterDupl
job_scr	ript() (openff.evaluator.backends.dask.DaskLSFI	Backend	method), 160
	method), 253		(openff.evaluator.datasets.curation.components.freesolv.ImportFree
job_scr	ript() (openff.evaluator.backends.dask.DaskPBSI method), 255	Backend json()	method), 202 (openff.evaluator.datasets.curation.components.selection.SelectData
join()	(openff.evaluator.storage.attributes.FilePath method), 293	json()	method), 212 (openff.evaluator.datasets.curation.components.selection.SelectSub
join()(	openff.evaluator.utils.observables.ObservableArra	ay	method), 209
	class method), 622		$(open {\it ff. evaluator.} datasets. curation. components. selection. State$
join()(	openff.evaluator.utils.observables.ObservableFran		method), 214
	class method), 624	json()	(open ff.evaluator. datasets. curation. components. selection. Target Statistical Statistical Components) and target Statistical Components an
json()	$(open f\!f. evaluator. attributes. Attribute Class$		method), 216
	method), 620	json()	(openff.evaluator.datasets.curation.components.thermoml.ImportTl
json()	(openff.evaluator.client.ConnectionOptions	icon()	method), 205
	method), 79	JSOII()	(openff.evaluator.datasets.curation.workflow.CurationWorkflowSch method), 156
	(openff.evaluator.client.Request method), 81	json()	· · · · · · · · · · · · · · · · · · ·
json()(	(openff.evaluator.client.RequestOptions method), 84	JSOII()	method), 98
json()	(openff.evaluator.client.RequestResult method),	json()	
J 5011()	85	3()	method), 94
json()	(openff.evaluator.datasets.CalculationSource method), 97	json()	(openff.evaluator.datasets.PhysicalPropertyDataSet method), 139
ison()(	openff.evaluator.datasets.curation.components.com	กรัสเรา	
J = === () (	method), 220	<pre>json()</pre>	(openff.evaluator.datasets.taproom.TaproomDataSet
json()(	openff.evaluator.datasets.curation.components.Cu	rationCo	omportened chema
	<i>method</i> ), 152	json()	(openff.evaluator.datasets.taproom.TaproomSource
json()(	openff.evaluator.datasets.curation.components.filt method), 182	ering.Fili json()	ter <b>BSCAd)</b> gedSchema (openff.evaluator.datasets.thermoml.ThermoMLDataSet
json()(	openff.evaluator.datasets.curation.components.filt method), 174	ering.Fili json()	ter <b>B&amp;E</b> RAhenASSchema (openff.evaluator.forcefield.ForceFieldSource
json()(	openff.evaluator.datasets.curation.components.filt method), 199	ering.Fili json()	terB\$InAironmentsSchema (openff.evaluator.forcefield.LigParGenForceFieldSource
json()(	openff.evaluator.datasets.curation.components.filt method), 184	ering.Fili json()	te <b>rB\$\$&amp;A</b> dLiquidSchema (openff.evaluator.forcefield.SmirnoffForceFieldSource
json()(	(openff.evaluator.datasets.curation.components.filt method), 169	ering.Fili	ter <b>B\$MFA</b> FF <del>A&amp;</del> tionSchema (openff.evaluator.forcefield.TLeapForceFieldSource
json()(	openff.evaluator.datasets.curation.components.filt method), 192	ering.Fili	terB\$NCOmponentsSchema (openff.evaluator.layers.CalculationLayerResult
json()(	openff.evaluator.datasets.curation.components.filt method), 166	ering.Fili	ter <b>ByPress</b> ure\$chema (openff.evaluator.layers.CalculationLayerSchema
json()(	openff.evaluator.datasets.curation.components.filt method), 177	ering.Fili	. 1 00
json()(	openff.evaluator.datasets.curation.components.filt	ering.Fili	
json()(	method), 171 (openff.evaluator.datasets.curation.components.filt	ering.Fili	* * **
icor() (	method), 187 openff.evaluator.datasets.curation.components.filt	-	* * **
Json()(	openff.evaluator.aatasets.curation.components.fitt method). 190	ering.Fili json()	ter <b>вузти</b> кsэспета (openff.evaluator.properties.Density method),

101							n	reth	od)	, 46	ıΙ

- json() (openff.evaluator.properties.DielectricConstant json() (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRole method), 108 method), 456
- json() (openff.evaluator.properties.EnthalpyOfMixing json() (openff.evaluator.protocols.miscellaneous.MultiplyValue method), 112 method), 441
- json() (openff.evaluator.properties.EnthalpyOfVaporizationjson() (openff.evaluator.protocols.miscellaneous.SubtractValues method), 115 method), 436
- json() (openff.evaluator.properties.HostGuestBindingAffinijson() (openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation method), 122 method), 466
- json() (openff.evaluator.protocols.analysis.AverageDielectris6hh) (openff.evaluator.protocols.openmm.OpenMMSimulation method), 345 method), 473
- json() (openff.evaluator.protocols.analysis.AverageFreeEngsgons() (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase method), 351 method), 525
- method), 351 method), 525 json() (openff.evaluator.protocols.analysis.AverageObserv**ijkke**n() (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWorld (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWorld (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWorld (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWorld (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWorld (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWorld (openff.evaluator.protocols.analysis.ComputeReferenceWorld (openff.evaluato
- method), 339
  method), 535
  json() (openff.evaluator.protocols.analysis.BaseAverageObjsordDlcopenff.evaluator.protocols.paprika.analysis.ComputeSymmetryConmethod), 333
  method), 530
- json() (openff.evaluator.protocols.analysis.BaseDecorrelat**jeProt**(); (bpenff.evaluator.protocols.paprika.coordinates.AddDummyAtoms method), 361 method), 496
- json() (openff.evaluator.protocols.analysis.ComputeDipole**jsom@n(**popenff.evaluator.protocols.paprika.coordinates.PreparePullCoordinates.preparePullCoordinates.
- json() (openff.evaluator.protocols.analysis.DecorrelateObs**jesonk**()e(openff.evaluator.protocols.paprika.coordinates.PrepareReleaseComethod), 371 method), 491
- json() (openff.evaluator.protocols.analysis.DecorrelateTrajeson() (openff.evaluator.protocols.paprika.restraints.ApplyRestraints method), 366 method), 519
- json() (openff.evaluator.protocols.coordinates.BuildCoord
- json() (openff.evaluator.protocols.coordinates.BuildDocke**dCont**(dispenff.evaluator.protocols.paprika.restraints.GeneratePullRestrain method), 391 method), 507

json() (openff.evaluator.protocols.coordinates.SolvateExistjssest()) (toppenff.evaluator.protocols.paprika.restraints.GenerateReleaseRest

- method), 384 method), 513
  json() (openff.evaluator.protocols.forcefield.BaseBuildSystignson() (openff.evaluator.protocols.reweighting.BaseEvaluateEnergies
- method), 396 method), 551
- json() (openff.evaluator.protocols.forcefield.BuildLigParGinsbys@n(openff.evaluator.protocols.reweighting.BaseMBARProtocol method), 407 method), 557
- json() (openff.evaluator.protocols.forcefield.BuildSmirnoff**System**) (openff.evaluator.protocols.reweighting.ConcatenateObservables method), 402 method), 546
- json() (openff.evaluator.protocols.forcefield.BuildTLeapSysystem() (openff.evaluator.protocols.reweighting.ConcatenateTrajectories method), 413 method), 540
- json() (openff.evaluator.protocols.gradients.ZeroGradientsjson() (openff.evaluator.protocols.reweighting.ReweightDielectricConstant method), 419 method), 569
- json() (openff.evaluator.protocols.groups.ConditionalGroupson() (openff.evaluator.protocols.reweighting.ReweightObservable method), 426 method), 563
- json() (openff.evaluator.protocols.groups.ConditionalGrou**js6nhdi(ipe**nff.evaluator.protocols.simulation.BaseEnergyMinimisation method), 422 method), 575
- json() (openff.evaluator.protocols.miscellaneous.AddValuejson() (openff.evaluator.protocols.simulation.BaseSimulation method), 431
  method), 582
- json() (openff.evaluator.protocols.miscellaneous.DivideValjæon() (openff.evaluator.protocols.storage.UnpackStoredSimulationData method), 446 method), 588
- json() (openff.evaluator.protocols.miscellaneous.DummyPjason(l) (openff.evaluator.protocols.yank.BaseYankProtocol

ison()	method), 594 (openff.evaluator.protocols.yank.LigandReceptorYa	_	openff.evaluator.workflow.schemas.ProtocolSchema
J30II()	method), 602		openff.evaluator.workflow.schemas.WorkflowSchema
<pre>json()</pre>	(openff.evaluator.protocols.yank.SolvationYankPromethod), 610		method), 323 (openff.evaluator.workflow.WorkflowException
-	(openff.evaluator.server.Batch method), 91		method), 300
json()	(openff.evaluator.storage.data.BaseSimulationData	ijson()	(openff.evaluator.workflow.WorkflowResult
json()	method), 268 (openff.evaluator.storage.data.BaseStoredData		method), 303
J30II()	method), 261	K	
json()	(openff.evaluator.storage.data.ForceFieldData method), 264	keys()(	openff.evaluator.utils.observables.ObservableFrame method), 625
<pre>json()</pre>	(openff.evaluator.storage.data.HashableStoredData	a	<i>"</i>
·()	method), 262	L	
json()	(openff.evaluator.storage.data.ReplaceableData method), 266		rotocol (openff.evaluator.workflow.utils.ProtocolPath
<pre>json()</pre>	(openff.evaluator.storage.data.StoredFreeEnergyDa	ata so	property), 328 purce (openff.evaluator.forcefield.TLeapForceFieldSource
json()	(openff.evaluator.storage.data.StoredSimulationDa	<sup>tq</sup> left_ha	property), 224 nd_value(openff.evaluator.protocols.groups.ConditionalGroup.C
json()	method), 272 (openff.evaluator.storage.query.BaseDataQuery		attribute), 422
J3011()	method), 277	ligand_	electrostatic_lambdas
<pre>json()</pre>	(openff.evaluator.storage.query.BaseSimulationDa	taQuery	(openff.evaluator.protocols.yank.LigandReceptorYankProtocol attribute), 598
	method), 283	ligand_	residue_name
json()	(openff.evaluator.storage.query.ForceFieldQuery	_	$(open {\it ff. evaluator. protocols. coordinates. Build Docked Coordinates}$
ison()	method), 281 (openff.evaluator.storage.query.FreeEnergyDataQt	uerv .	attribute), 389
10011()	method), 288	"I'i'gand_	residue_name (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
<pre>json()</pre>	(openff.evaluator.storage.query.Simulation Data Querting and the property of	ery	attribute), 598
	method), 286	ligand_	steric_lambdas
json()	method), 279		(openff.evaluator.protocols.yank.LigandReceptorYankProtocol attribute), 598
json()	(openff.evaluator.substances.Amount method), 130	ligand_	substance (openff.evaluator.protocols.coordinates.BuildDockedCattribute), 388
json()	(openff.evaluator.substances.Component method), 129	LigandR	eceptorYankProtocol (class in
json()	(openff.evaluator.substances.ExactAmount	1 ID	openff.evaluator.protocols.yank), 596
5 - 0	method), 132	Liganak	eceptorYankProtocol.RestraintType (class in openff.evaluator.protocols.yank), 597
<pre>json()</pre>	(openff.evaluator.substances.MoleFraction	LigParG	enForceFieldSource (class in
icon()	method), 134		openff.evaluator.forcefield), 225
json()	(openff.evaluator.substances.Substance method), 127	LigParG	enForceFieldSource.ChargeModel (class in
json()	(openff.evaluator.thermodynamics.Thermodynamic method), 136	:Statest()	
json()	(openff.evaluator.utils.exceptions.EvaluatorExceptimethod), 86	<i>ion</i> load_re	method), 293 straints() (openff.evaluator.protocols.paprika.restraints.Applyl
json()	(openff.evaluator.workflow.Protocol method), 308	LocalFi	class method), 516 leStorage (class in openff.evaluator.storage),
json()	(openff.evaluator.workflow.ProtocolGroup method), 314	lower()	matha d. 204
<pre>json()</pre>	(openff.evaluator.work flow.schemas.Protocol Group flow) for the protocol of	Schema Istrip(	method), 294 () (openff.evaluator.storage.attributes.FilePath
ison()	method), 319 (openff.evaluator.workflow.schemas.ProtocolReplic	rator	method), 294
5011()	method), 321		

M merg	e() (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
$\verb maketrans()  (open fi.evaluator.storage.attributes. File Path $	method), 402
state method), 274	(e() (openff.evaluator.protocols.forcefield.BuildTLeapSystem
mass_density (openff.evaluator.protocols.coordinates.BuildCoo attribute), 374 merg	ordintltesPttckmot  e() (openff.evaluator.protocols.gradients.ZeroGradients
mass_density(openff.evaluator.protocols.coordinates.SolvateE	xistingSthactore <sup>419</sup>
attribute), 384	(openjj.evatuator.protocots.groups.ConditionatGroup
max_iterations (openff.evaluator.protocols.groups.Conditiona attribute), 423 merg	lGrollf (100a), 425
<pre>max_iterations (openff.evaluator.protocols.openmm.OpenMM</pre>	
<pre>max_iterations (openff.evaluator.protocols.simulation.BaseEn</pre>	
an touch, 57	e() (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRolo
max_molecules (openff.evaluator.protocols.coordinates.Solvatelattribute), 384 merg	e() (openy.evaluator.protocois.miscettaneous.muttipty value
maximum_data_points	method), 441
(openff.evaluator.layers.reweighting.Reweighting Sthe	(c) (openff.evaluator.protocols.miscellaneous.SubtractValues method), 436
attribute), 241 MeasurementSource (class in merg	e() (openff.evaluator.protocols.miscellaneous.WeightByMoleFraction
open of evaluator datasets) 07	method), 451
merge() (openff.evaluator.datasets.PhysicalPropertyDataSeterg method), 138	(e() (openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation method), 467
merge() (openff.evaluator.datasets.taproom.TaproomDataSeerg	(e() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies
mathod) 147	metnoa), 480
merge() (openff.evaluator.datasets.thermoml.ThermoMLDdtaset	() (openff.evaluator.protocols.openmm.OpenMMSimulation method), 474
method), 143	(apenff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase
method) 345	meinoa), 525
merge() (openff.evaluator.protocols.analysis.AverageFreeEmerg	g() (openff.evaluator.protocols.paprika.analysis.ComputeReferenceV
method) 351	metnoa), 555
mathod) 330	(e() (openff.evaluator.protocols.paprika.analysis.ComputeSymmetryC method), 530
merge() (openff.evaluator.protocols.analysis.BaseAverage Obset)	(e) (openff.evaluator.protocols.paprika.coordinates.AddDummyAton
mathod) 333	meinoa), 490
merge() (openff.evaluator.protocols.analysis.BaseDecorrelater	(gpenff.evaluator.protocols.paprika.coordinates.PreparePullCoo method), 486
method), 361	memod,, 100  [ments] (openff.evaluator.protocols.paprika.coordinates.PrepareRelease
mathod) 356	metnoa), 491
merge() (openff.evaluator.protocols.analysis.DecorrelateObserv	(c) (c) (c) (c) (d) (d) (e) (e) (e) (e) (e) (e) (e) (e) (e) (e
merge() (openff.evaluator.protocols.analysis.DecorrelateTrufer	(openff.evaluator.protocols.paprika.restraints.GenerateAttachRemethod), 502
merge() (openff.evaluator.protocols.coordinates.BuildCoordinates	e(Q) (apenff.evaluator.protocols.paprika.restraints.GeneratePullRestr method), 508
merge() (openff.evaluator.protocols.coordinates.BuildDockedE9 method), 391	Scal (openff.evaluator.protocols.paprika.restraints.GenerateReleaseR method), 513

 $\verb|merge()| (open ff. evaluator. protocols. for cefield. Build Lig Par Gen. System (open ff. evaluator. protocols. reweighting. Concatenate Observables) | Concatenate Observables | Concatenate Observ$ method), 546method), 408

 $\verb|merge()| (open ff. evaluator. protocols. coordinates. Solvate Existing Structure ff. evaluator. protocols. reweighting. Base Evaluate Energies for the protocols of the prot$ 

 $\verb|merge()| (open ff. evaluator. protocols. for cefield. Base Build System \texttt{ge}()| (open ff. evaluator. protocols. reweighting. Base MBAR Protocols)| (open ff. evaluator. protocols. protocols.$ 

*method*), 551

method), 557

Index 677

method), 384

method), 397

```
merge() (openff.evaluator.protocols.reweighting.ConcatenateTrajecto#les
         method), 541
merge() (openff.evaluator.protocols.reweighting.ReweightDielectricConstant
         method), 569
                                                          n\_microstates (openff. evaluator. protocols. paprika. analysis. Compute Symmetry n\_microstates)
merge() (openff.evaluator.protocols.reweighting.ReweightObservableattribute), 527
         method), 563
                                                          n_pull_windows (openff.evaluator.protocols.paprika.coordinates.Prepare.
merge() (openff.evaluator.protocols.simulation.BaseEnergyMinimisatiumibute), 483
         method), 575
                                                          n_pull_windows (openff.evaluator.protocols.paprika.restraints.GenerateP
merge() (openff.evaluator.protocols.simulation.BaseSimulation
                                                                    attribute), 505
         method), 582
                                                          number_of_components
merge() (openff.evaluator.protocols.storage.UnpackStoredSimulationDatenff.evaluator.substances.Substance prop-
         method), 588
                                                                    erty), 125
merge() (openff.evaluator.protocols.yank.BaseYankProtocohumber_of_equilibration_iterations
         method), 594
                                                                    (openff.evaluator.protocols.yank.BaseYankProtocol
merge() (openff.evaluator.protocols.yank.LigandReceptorYankProtocaltribute), 591
         method), 602
                                                          number_of_equilibration_iterations
merge() (openff.evaluator.protocols.yank.SolvationYankProtocol
                                                                    (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
         method), 611
                                                                    attribute), 602
merge() (openff.evaluator.workflow.Protocol method),
                                                         number_of_equilibration_iterations
                                                                    (open {\it ff. evaluator. protocols. yank. Solvation Yank Protocol}
merge()
               (openff.evaluator.workflow.ProtocolGroup
                                                                    attribute), 611
         method), 312
                                                          number\_of\_gpus (openff.evaluator.backends.ComputeResources
MergeBehavior
                                (class
                                                     in
                                                                    property), 245
         openff.evaluator.workflow.attributes), 324
                                                          number\_of\_gpus (\textit{openff.evaluator.backends.QueueWorkerResources})
             (openff.evaluator.datasets.PhysicalProperty
                                                                   property), 247
         attribute), 92
                                                          number_of_iterations
metadata (openff.evaluator.properties.Density attribute),
                                                                    (openff.evaluator.protocols.yank.BaseYankProtocol
                                                                    attribute), 591
metadata(openff.evaluator.properties.DielectricConstant number_of_iterations
         attribute), 108
                                                                    (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
metadata (openff.evaluator.properties.EnthalpyOfMixing
                                                                    attribute), 602
         attribute), 112
                                                          number_of_iterations
metadata (openff.evaluator.properties.EnthalpyOfVaporization
                                                                    (open {\it ff. evaluator. protocols. yank. Solvation Yank Protocol}
         attribute), 115
                                                                    attribute), 611
metadata(openff.evaluator.properties.ExcessMolarVolumenumber_of_ligand_conformers
         attribute), 105
                                                                    (openff.evaluator.protocols.coordinates.BuildDockedCoordinates
metadata (openff.evaluator.properties.HostGuestBindingAffinity
                                                                    attribute), 388
         attribute), 122
                                                          number_of_molecules
metadata (openff.evaluator.properties.SolvationFreeEnergy
                                                                    (openff.evaluator.storage.data.StoredSimulationData
         attribute), 118
                                                                    attribute), 271
MoleFraction (class in openff.evaluator.substances), number_of_molecules
                                                                    (openff.evaluator.storage.query.SimulationDataQuery
most_information() (openff.evaluator.storage.data.BaseSimulationDatabute), 285
         class method), 268
                                                          \verb|number_of_threads|| (open \textit{ff.evaluator.backends.} Compute \textit{Resources}||
most_information() (openff.evaluator.storage.data.ReplaceableDataroperty), 245
         class method), 265
                                                          \verb|number_of_threads|| (open \textit{ff.evaluator.backends.QueueWorkerResources}||
most_information() (openff.evaluator.storage.data.StoredFreeEnergyDetay), 247
         class method), 274
most_information() (openff.evaluator.storage.data.Store dimulationData
         class method), 271
                                                          Observable (class in openff.evaluator.utils.observables),
multiplier (openff.evaluator.protocols.miscellaneous.MultiplyValue621
         attribute), 438
                                                          {\tt observable} \ (open {\it ff. evaluator. protocols. analysis. Average Observable}
MultiplyValue
                                (class
                                                      in
                                                                    attribute), 336
         openff.evaluator.protocols.miscellaneous),
```

outputs (openff.evaluator.protocols.coordinates.SolvateExistingStructure

```
observable (openff.evaluator.protocols.reweighting.ReweightObservable (openff.evaluator.protocols.coordinates.BuildCoordinatesPackmo
                                          attribute), 560
                                                                                                                                                                                                                                                                                                         attribute), 375
ObservableArray
                                                                                                                                                                                                                                                             output_number_of_molecules
                                          openff.evaluator.utils.observables), 622
                                                                                                                                                                                                                                                                                                         (openff.evaluator.protocols.coordinates.SolvateExistingStructure
ObservableFrame
                                                                                                                                                   (class
                                                                                                                                                                                                                                                                                                         attribute), 385
                                          openff.evaluator.utils.observables), 623
                                                                                                                                                                                                                                                             output_observables(openff.evaluator.protocols.analysis.DecorrelateOb
observables (openff.evaluator.protocols.openmm.OpenMMSimulatiattribute), 369
                                          attribute), 474
                                                                                                                                                                                                                                                              \verb"output_observables" (open ff. evaluator. protocols. gradients. Zero Gradients) and the substitution of the substitution of
observables (openff.evaluator.protocols.simulation.BaseSimulation attribute), 417
                                          attribute), 580
                                                                                                                                                                                                                                                              \verb"output_observables" (open ff. evaluator. protocols. open mm. Open MME valuator. protocols. Open MME valuator. Protoc
observables (openff.evaluator.protocols.storage.UnpackStoredSimulativibDret), 480
                                                                                                                                                                                                                                                              output_observables(openff.evaluator.protocols.reweighting.BaseEvaluation)
                                          attribute), 585
observables (openff.evaluator.storage.data.StoredSimulationData attribute), 549
                                         attribute), 270
                                                                                                                                                                                                                                                              output_observables (openff.evaluator.protocols.reweighting.Concatenat
ObservableType
                                                                                                                                                                                                                                          in
                                                                                                                                                                                                                                                                                                         attribute), 543
                                                                                                                                               (class
                                           openff.evaluator.utils.observables), 623
                                                                                                                                                                                                                                                              \verb"output_substance" (open ff. evaluator. protocols. coordinates. Build Coordinates (open ff. evaluator. protocols. coordinates) and the substance of the coordinates of the coordinate
offset (openff.evaluator.protocols.paprika.coordinates.AddDummyAtatmisbute), 375
                                         attribute), 494
                                                                                                                                                                                                                                                              \verb"output_substance" (open ff. evaluator. protocols. coordinates. Solvate Existing the contraction of the c
OpenMMEnergyMinimisation
                                                                                                                                                                                                                                          in
                                                                                                                                                                                                                                                                                                         attribute), 385
                                                                                                                                                                           (class
                                                                                                                                                                                                                                                              \verb"output_system" (open ff. evaluator. protocols. paprika. coordinates. Add Dummark (open ff. evaluator. protocols. paprika. 
                                          openff.evaluator.protocols.openmm), 463
                                                                                                                                                                                                                                                                                                         attribute), 494
OpenMMEvaluateEnergies
                                                                                                                                                                      (class
                                                                                                                                                                                                                                          in
                                          openff.evaluator.protocols.openmm), 476
                                                                                                                                                                                                                                                             \verb"output_system" (open ff. evaluator. protocols. paprika. restraints. Apply Restra
OpenMMSimulation
                                                                                                                                                      (class
                                                                                                                                                                                                                                                                                                         attribute), 516
                                                                                                                                                                                                                                          in
                                           openff.evaluator.protocols.openmm), 468
                                                                                                                                                                                                                                                             output_trajectory_path
options (openff.evaluator.server.Batch attribute), 90
                                                                                                                                                                                                                                                                                                         (openff.evaluator.protocols.analysis.DecorrelateTrajectory
output_coordinate_file
                                                                                                                                                                                                                                                                                                         attribute), 364
                                           (openff.evaluator.protocols.openmm.OpenMMEneovyt/divintinatiextory_path
                                         attribute), 467
                                                                                                                                                                                                                                                                                                         (open ff. evaluator. protocols. reweighting. Concatenate Trajectories
output_coordinate_file
                                                                                                                                                                                                                                                                                                         attribute), 538
                                          (openff.evaluator.protocols.openmm.OpenMMSimolutiput_value (openff.evaluator.protocols.miscellaneous.DummyProtocol
                                         attribute), 474
                                                                                                                                                                                                                                                                                                         attribute), 458
output_coordinate_file
                                                                                                                                                                                                                                                             OutputAttribute
                                                                                                                                                                                                                                                                                                                                                                                                                 (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          in
                                          (openff.evaluator.protocols.simulation.BaseEnergyMinimisaoponff.evaluator.workflow.attributes), 326
                                         attribute), 573
                                                                                                                                                                                                                                                             \verb"outputs" (open ff. evaluator. protocols. analysis. Average Dielectric Constant
                                                                                                                                                                                                                                                                                                        property), 345
output_coordinate_file
                                          (openff.evaluator.protocols.simulation.BaseSimulativputs (openff.evaluator.protocols.analysis.AverageFreeEnergies
                                         attribute), 580
                                                                                                                                                                                                                                                                                                        property), 351
output_coordinate_path
                                                                                                                                                                                                                                                             \verb"outputs" (open ff. evaluator. protocols. analysis. Average Observable
                                           (openff.evaluator.protocols.paprika.coordinates.AddDummyAstopesty), 339
                                                                                                                                                                                                                                                             \verb"outputs" (open ff. evaluator. protocols. analysis. Base Average Observable
                                         attribute), 494
output_coordinate_path
                                                                                                                                                                                                                                                                                                        property), 334
                                          (openff.evaluator.protocols.paprika.coordinates.PraptapeRs)(openff.evalustor.protocols.analysis.BaseDecorrelateProtocol
                                         attribute), 486
                                                                                                                                                                                                                                                                                                        property), 361
output_coordinate_path
                                                                                                                                                                                                                                                             \verb"outputs" (open ff. evaluator. protocols. analysis. Compute Dipole Moments
                                          (openff.evaluator.protocols.paprika.coordinates.PrepareRelearsqueotyn)diffutes
                                         attribute), 491
                                                                                                                                                                                                                                                              outputs (openff.evaluator.protocols.analysis.DecorrelateObservables
output_coordinate_path
                                                                                                                                                                                                                                                                                                        property), 371
                                          (openff.evaluator.protocols.reweighting.ConcatenateTrajectory
                                         attribute), 538
                                                                                                                                                                                                                                                                                                        property), 366
output_frequency (openff.evaluator.protocols.openmm.OpentMMSi(mykutffievaluator.protocols.coordinates.BuildCoordinatesPackmol
                                         attribute), 474
                                                                                                                                                                                                                                                                                                        property), 378
output_frequency (openff.evaluator.protocols.simulation.\(\mathbb{O}\) are solved large in the first of the f
                                         attribute), 579
                                                                                                                                                                                                                                                                                                        property), 391
```

Index 679

output\_number\_of\_molecules

	property), 385	property), 552			
outputs	$(open \textit{ff.evaluator.protocols.force field.} Base \textit{BuildSy.} \textbf{\textit{stertuput}}$	s (openff.evaluator.protocol	s.reweighti	ing.BaseMBARProtocol	
	property), 397	property), 557			
outputs	(openff.evaluator.protocols.forcefield.BuildLigParGert.putte	<b>s</b> n(openff.evaluator.protocol	s.reweighti	ing.ConcatenateObservab	les
	property), 408	property), 546			
outputs	(openff.evaluator.protocols.forcefield.BuildSmirnoffSixtsperts	s (openff.evaluator.protocol	s.reweighti	ing.ConcatenateTrajectori	ies
	property), 402	property), 541		•	
outputs	(openff.evaluator.protocols.forcefield.BuildTLeapSoureput	s (openff.evaluator.protocol	s.reweighti	ing.ReweightDielectricCo	nstc
•	property), 413	property), 569	O		
outputs	(openff.evaluator.protocols.gradients.ZeroGradientsutput	1 1 2	s.reweighti	ing.ReweightObservable	
•	property), 419	property), 563			
outnuts	(openff.evaluator.protocols.groups.ConditionalGrautput		s simulatio	on RaseEnerovMinimisatio	on
o ar ep ar es	property), 426	property), 576	<b>5151111111111</b>		
outnuts	(openff.evaluator.protocols.miscellaneous.AddValuartput		s simulatio	on RaseSimulation	
oucpucs	property), 431	property), 583	s.simmano	n.Basesimmanon	
011+111+0	(openff.evaluator.protocols.miscellaneous.DivideV <b>olu</b> teput		s storage I	Innack Stored Simulation D	lata
outputs	property), 446	property), 588	s.siorage.c	прискыотеазинишиюны	ши
++	property), 440 (openff.evaluator.protocols.miscellaneous.Dummy <b>Вимри</b> ф.		a naul Daa	a Van la Duata a a l	
outputs	1 00 1	. 1 00	s.yank.bas	e tankProtocot	
	property), 461	property), 595	1.7.	ID , V ID ,	1
outputs	(openff.evaluator.protocols.miscellaneous.FilterSubsttepat.		s.yank.Ligo	апакесеріог тапкР гогосог	,
	property), 456	property), 602	1.0.1		
outputs	(openff.evaluator.protocols.miscellaneous.Multiplydanlpert		s.yank.Solv	vationYankProtocol	
	property), 441	property), 611			
outputs	(openff.evaluator.protocols.miscellaneous.Subtractivatipest		w.Protocol	property),	
	property), 436	306			
outputs	(open ff. evaluator. protocols. miscellaneous. Weight ByM-phe-f	$oldsymbol{arepsilon}$ action(openff.evaluator.wo	rkflow.Pro	tocolGroup	
	property), 451	property), 311			
outputs	(openff.evaluator.protocols.openmm.OpenMMEnergythiuti	<u>nistatias</u> tore ( <i>openff.evalua</i>	tor.workflo	w.schemas.WorkflowSche	та
	property), 467	attribute), 322			
outputs	(open ff. evaluator. protocols. open mm. Open MME valuator. protocols. open mm. Open MME valua	sgto_store(openff.evalua	tor.workflo	w.Workflow	
	property), 480	property), 298			
outputs	(openff.evaluator.protocols.openmm.OpenMMSimulation				
	property), 474				
outputs	(openff.evaluator.protocols.paprika.analysis.AnalyseAPRI	hase gradient keys			
-	property), 525	(openff.evaluator.server.B	Ratch	attribute),	
outputs	(openff.evaluator.protocols.paprika.analysis.ComputeRefe	enceWork	cucri	annonie),	
•		terGradient	(class	in	
outputs	(openff.evaluator.protocols.paprika.analysis.ComputeSym	netryCorrection or forcefield	(ciuss 1) 228	iti	
ou cp u co		terGradientKey		*	
outnuts	(openff.evaluator.protocols.paprika.coordinates.AddDumn	vAtoms:l	(class	in	
oucpucs			1), 227		
011+111+0		terized_system	, , ,	C D' LM	
outputs	(openff.evaluator.protocols.paprika.coordinates.PreparePi	Mopenff!evanuator.protoco	ls.analysis.	ComputeDipoleMoments	
011+211+0	property), 486	attribute), 353			
outputs	(openff.evaluator.protocols.paprika.coordinates.Preparelle	•			
	property), 491	. (openff.evaluator.protoco	ls.forcefield	d.BaseBuildSystem	
outputs	(openff.evaluator.protocols.paprika.restraints.ApplyRestra	,,			
	property), 519 parame	terized_system			
outputs	(openff.evaluator.protocols.paprika.restraints.GenerateAtt	<sup>ach</sup> Beriff!eValuator.protoco	ls.forcefield	d.BuildLigParGenSystem	
	property), 502	attribute), 408			
outputs	(openff.evaluator.protocols.paprika.restraints.GeneratePut	ter <i>itzeat</i> system			
	property), 508	(openff.evaluator.protoco	ls.forcefield	d.BuildSmirnoffSystem	
outputs	(open ff.evaluator.protocols.paprika.restraints. GenerateRelation) and figure for the protocols.paprika.restraints. GenerateRelation and figure for the protocols.paprika.papr	easaRestrajn <b>4</b> 02			
	property), 513 parame	terized_system			
outputs	(openff.evaluator.protocols.reweighting.BaseEvaluateEner	gi <b>es</b> nanff avaluator protoco	ls forcefield	d RuildTI aan System	

 $\verb"parse_json()" (open ff. evaluator. layers. Calculation Layer Schema"$ 

class method), 231

(openff.evaluator.protocols.openmm.OpenMMEnergyMinimislatiomethod), 232

attribute), 414

parameterized\_system

attribute), 467	<pre>parse_json() (openff.evaluator.layers.reweighting.ReweightingSchema</pre>
parameterized_system	class method), 242
(openff.evaluator.protocols.openmm.OpenMMEv	a <b>իամե</b> Enġ <b>gon(</b> ) (openff.evaluator.layers.simulation.SimulationSchema
attribute), 480	class method), 239
parameterized_system	<pre>parse_json() (openff.evaluator.layers.workflow.WorkflowCalculationSch</pre>
(openff.evaluator.protocols.openmm.OpenMMSin	
attribute), 474	parse_json() (openff.evaluator.properties.Density
parameterized_system	class method), 101
	upar Energisiem () (openff. evaluator. properties. Dielectric Constant
attribute), 549	class method), 108
parameterized_system	parse_json() (openff.evaluator.properties.EnthalpyOfMixing
(openff.evaluator.protocols.simulation.BaseEnerg	
attribute), 573	parse_json() (openff.evaluator.properties.EnthalpyOfVaporization
parameterized_system	class method), 115
	aparse_json() (openff.evaluator.properties.ExcessMolarVolume
attribute), 579	class method), 105
	parse_json() (openff.evaluator.properties.HostGuestBindingAffinity
class method), 620	class method), 122
	sparse_json() (openff.evaluator.properties.SolvationFreeEnergy
class method), 80	class method), 118
	parse_json() (openff.evaluator.protocols.analysis.AverageDielectricCon
method), 82	class method), 345
	parse_json() (openff.evaluator.protocols.analysis.AverageFreeEnergies
class method), 84	class method), 351
	parse_json() (openff.evaluator.protocols.analysis.AverageObservable
class method), 86	class method), 339
	ctass method), 339 cparse_json() (openff.evaluator.protocols.analysis.BaseAverageObserval
class method), 97	class method), 334
	<pre>uparse_json() (openff.evaluator.protocols.analysis.BaseDecorrelateProto class method), 361</pre>
class method), 98	
	yparse_json() (openff.evaluator.protocols.analysis.ComputeDipoleMome
class method), 94	class method), 356
class method), 140	y <b>Paras S</b> et j son() (openff.evaluator.protocols.analysis.DecorrelateObservable
**	class method), 371
	parse_json() (openff.evaluator.protocols.analysis.DecorrelateTrajectory
method), 96	class method), 366
- · · · · · · · · · · · · · · · · · · ·	matara6etj son() (openff.evaluator.protocols.coordinates.BuildCoordinatesF
class method), 147	class method), 378
	mansee_json() (openff.evaluator.protocols.coordinates.BuildDockedCoo
class method), 150	class method), 391
	p <b>ALButgSem()</b> (openff.evaluator.protocols.coordinates.SolvateExistingStr
class method), 143	class method), 385
	caparse_json() (openff.evaluator.protocols.forcefield.BaseBuildSystem
class method), 222	class method), 397
	edarstojisca () (openff.evaluator.protocols.forcefield.BuildLigParGenSyste
class method), 227	class method), 408
	TiphtSpargeon() (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
class method), 223	class method), 402
	lasanseejson() (openff.evaluator.protocols.forcefield.BuildTLeapSystem
class method), 225	class method), 414
<pre>parse_json() (openff.evaluator.layers.CalculationLayerF</pre>	Rapadrse_json() (openff.evaluator.protocols.gradients.ZeroGradients

class method), 419	class method), 569
parse_json() (openff.evaluator.protocols.groups.Conditiopal@en_i	${\tt json()}$ (openff.evaluator.protocols.reweighting.ReweightObservable)
class method), 426	class method), 563
parse_json() (openff.evaluator.protocols.groups.Conditiopal@eo.j	ps6nhditapenff.evaluator.protocols.simulation.BaseEnergyMinimis
class method), 423	class method), 576
parse_json() (openff.evaluator.protocols.miscellaneous.ApdYsbee	son() (openff.evaluator.protocols.simulation.BaseSimulation
class method), 431	class method), 583
parse_json() (openff.evaluator.protocols.miscellaneous.Dpaids&alj	won() (openff.evaluator.protocols.storage.UnpackStoredSimulatio
class method), 446	class method), 588
parse_json() (openff.evaluator.protocols.miscellaneous.DparsePf	pson(l) (openff.evaluator.protocols.yank.BaseYankProtocol
class method), 461	class method), 595
parse_json() (openff.evaluator.protocols.miscellaneous.FilarSals	usameByRpeaff.evaluator.protocols.yank.LigandReceptorYankProto
class method), 456	class method), 602
parse_json() (openff.evaluator.protocols.miscellaneous.MpdriphyY	(spenff.evaluator.protocols.yank.SolvationYankProtocol
class method), 441	class method), 611
parse_json() (openff.evaluator.protocols.miscellaneous.SpharsetY	
class method), 436	method), 91
parse_json() (openff.evaluator.protocols.miscellaneous.Waright&y	
class method), 451	class method), 268
parse_json() (openff.evaluator.protocols.openmm.OpenMp4Eserg	
class method), 467	class method), 261
parse_json() (openff.evaluator.protocols.openmm.OpenMp4Eselug	
class method), 480	class method), 264
parse_json() (openff.evaluator.protocols.openmm.OpenMp4.Sisseuli	
class method), 474	class method), 263
parse_json() (openff.evaluator.protocols.paprika.analysispanskyze	
class method), 525	class method), 266
parse_json() (openff.evaluator.protocols.paprika.analysisparse_uj	
class method), 535	class method), 275
parse_json() (openff.evaluator.protocols.paprika.analysispanse_uj	
class method), 530	class method), 272
parse_json() (openff.evaluator.protocols.paprika.coordinparssed_d	
class method), 497	class method), 278
parse_json() (openff.evaluator.protocols.paprika.coordinpars&rej	
	class method), 284
class method), 486 parse_json() (openff.evaluator.protocols.paprika.coordin <b>parsSe</b> rgi	· · · · · · · · · · · · · · · · · · ·
class method), 491	class method), 281
parse_json() (openff.evaluator.protocols.paprika.restrain <b>patrsel</b> y	· · · · · · · · · · · · · · · · · · ·
class method), 519	class method), 288
parse_json() (openff.evaluator.protocols.paprika.restrainpa@seerj	
class method), 502	class method), 286
parse_json() (openff.evaluator.protocols.paprika.restrainpa@seerj	
class method), 508	class method), 279
parse_json() (openff.evaluator.protocols.paprika.restrain <b>pa@sm</b> er	
class method), 513	class method), 130
parse_json() (openff.evaluator.protocols.reweighting.BaspArslead	
class method), 552	class method), 129
parse_json() (openff.evaluator.protocols.reweighting.BaspMB+RI	
class method), 557	class method), 132
parse_json() (openff.evaluator.protocols.reweighting.Compartsments	
class method), 546	class method), 134
parse_json() (openff.evaluator.protocols.reweighting.Comparesmented)	
class method), 541	class method), 127
parse_json() (openff.evaluator.protocols.reweighting.RewpiglstDig	<b>Bott()Copstiffte</b> valuator.thermodynamics.ThermodynamicState

class method), 136	pop() (openff.evaluator.utils.observables.ObservableFrame
parse_json() (openff.evaluator.utils.exceptions.Evaluato class method), 86	pop_next_in_path() (openff.evaluator.workflow.utils.ProtocolPath
parse_json() (openff.evaluator.workflow.Protocol class	method), 328
method), 309	popitem() (openff.evaluator.utils.observables.ObservableFrame
parse_json() (openff.evaluator.workflow.ProtocolGroup	
class method), 314	potential_energies (openff.evaluator.protocols.analysis.AverageDielect
parse_json() (openff.evaluator.workflow.schemas.Protoc	
class method), 319	potential_energies (openff.evaluator.protocols.analysis.AverageObserv
parse_json() (openff.evaluator.workflow.schemas.Protoc	
class method), 321	potential_energies (openff.evaluator.protocols.analysis.BaseAverageO
parse_json() (openff.evaluator.workflow.schemas.Protoc	
class method), 317	preferred_charge_model
	lowSchema (openff.evaluator.forcefield.LigParGenForceFieldSource
class method), 323	property), 226
<pre>parse_json() (openff.evaluator.workflow.WorkflowExcep</pre>	
class method), 300	(openff.evaluator.backends.ComputeResources
parse_json() (openff.evaluator.workflow.WorkflowResult	
class method), 303	preferred_gpu_precision
partition() (openff.evaluator.storage.attributes.FilePath	
method), 294	property), 247
per_thread_memory_limit	preferred_gpu_toolkit
(openff.evaluator.backends.QueueWorkerResourc	
property), 247	property), 245
phase (openff.evaluator.datasets.PhysicalProperty	
attribute), 93	(openff.evaluator.backends.QueueWorkerResources
phase (openff.evaluator.properties.Density attribute),	property), 248
101	PreparePullCoordinates (class in
phase (openff.evaluator.properties.DielectricConstant	openff.evaluator.protocols.paprika.coordinates),
attribute), 108	482
phase (openff.evaluator.properties.EnthalpyOfMixing at-	PrepareReleaseCoordinates (class in
tribute), 112	openff.evaluator.protocols.paprika.coordinates),
${\tt phase}(open {\it ff.evaluator.properties.} Enthalpy Of Vaporization and {\it finite open filteral properties}) and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties} and {\it finite open filteral properties}. The {\it finite open filteral properties} and {\it finite open filteral properties}$	n 487
attribute), 115	<pre>prepend_protocol_id()</pre>
$phase \ \ (open \textit{ff.evaluator.properties.ExcessMolarVolume}$	(openff. evaluator. work flow. utils. Protocol Path
attribute), 105	method), 328
	itpressure(openff.evaluator.thermodynamics.ThermodynamicState
attribute), 122	attribute), 136
	${\tt properties} \ (open \textit{ff.evaluator.datasets.PhysicalPropertyDataSet}$
attribute), 118	property), 137
	eAPMPensies (openff.evaluator.datasets.taproom.TaproomDataSet
attribute), 522	property), 147
	Raroparties (openff.evaluator.datasets.thermoml.ThermoMLDataSet
attribute), 516	property), 143
$\verb"physical_property" (open \textit{ff.evaluator.layers.Calculation}) \\$	
attribute), 230	(openff.evaluator.datasets.PhysicalPropertyDataSet
PhysicalProperty (class in openff.evaluator.datasets),	method), 138
91	<pre>properties_by_substance()</pre>
PhysicalPropertyDataSet (class in	(openff.evaluator.datasets.taproom.TaproomDataSet
openff.evaluator.datasets), 137	method), 148
placeholder_id(openff.evaluator.workflow.schemas.Pro	
property), 320	(openff.evaluator.datasets.thermoml.ThermoMLDataSet
PlaceholderValue (class in	method), 143
openff.evaluator.attributes), 620	<pre>properties_by_type()</pre>

```
(openff.evaluator.datasets.PhysicalPropertyDataSet
                                                                                                      property), 309
              method), 138
                                                                                                             (openff.evaluator.workflow.ProtocolGroup
                                                                                       protocols
properties_by_type()
                                                                                                      property), 312
               (openff.evaluator.datasets.taproom.TaproomDataSprotocols (openff.evaluator.workflow.Workflow prop-
              method), 148
                                                                                                      erty), 298
properties_by_type()
                                                                                       protocols (openff.evaluator.workflow.WorkflowGraph
              (openff.evaluator.datasets.thermoml.ThermoMLDataSet
                                                                                                      property), 301
              method), 143
                                                                                       ProtocolSchema
                                                                                                                                         (class
                                                                                                                                                                        in
property\_name (openff.evaluator.workflow.utils.ProtocolPath
                                                                                                      openff.evaluator.workflow.schemas), 316
                                                                                       {\tt provenance}\,(open \textit{ff.evaluator.datasets.CalculationSource}
              property), 327
\verb"property_phase" (open ff. evaluator. storage. data. Base Simulation Data attribute), 96
                                                                                       pull\_distance (openff.evaluator.protocols.paprika.coordinates.PrepareP
              attribute), 267
property_phase (openff.evaluator.storage.data.StoredFreeEnergyDattribute), 483
              attribute), 275
                                                                                       pull_window_index (openff.evaluator.protocols.paprika.coordinates.Prep
property_phase (openff.evaluator.storage.data.StoredSimulationDatattribute), 483
              attribute), 272
property_phase(openff.evaluator.storage.query.BaseSimulationDataQuery
              attribute), 282
                                                                                                             (open {\it ff. evaluator. storage.} Local File {\it Storage}
                                                                                       query()
property_phase (openff.evaluator.storage.query.FreeEnergyDataQuerethod). 259
              attribute), 288
                                                                                       query()
                                                                                                              (openff.evaluator.storage.StorageBackend
property_phase (openff.evaluator.storage.query.SimulationDataQuantithod), 257
              attribute), 286
                                                                                       QueryAttribute
                                                                                                                                                                        in
property\_types (\textit{openff.evaluator.datasets.PhysicalPropertyDataSebpenff.evaluator.storage.attributes}), 297 in the property of the property
              property), 138
                                                                                       queued_properties(openff.evaluator.client.RequestResult
property_types (openff.evaluator.datasets.taproom.TaproomDataSetttribute), 85
              property), 148
                                                                                       queued_properties (openff.evaluator.server.Batch at-
property_types (openff.evaluator.datasets.thermoml.ThermoMLDatasets), 90
              property), 143
                                                                                       QueueWorkerResources
                                                                                                                                               (class
                                                                                                                                                                        in
PropertyPhase (class in openff.evaluator.datasets), 94
                                                                                                      openff.evaluator.backends), 246
Protocol (class in openff.evaluator.workflow), 304
                                                                                       QueueWorkerResources.GPUPrecision
                                                                                                                                                          (class
                                                                                                                                                                        in
protocol_ids (openff.evaluator.workflow.utils.ProtocolPath
                                                                                                      openff.evaluator.backends), 247
              property), 328
                                                                                       QueueWorkerResources.GPUToolkit
                                                                                                                                                         (class
                                                                                                                                                                        in
protocol_path(openff.evaluator.workflow.utils.ProtocolPath
                                                                                                      openff.evaluator.backends), 247
              property), 328
                                                                                       R
protocol_replicators
              (openff.evaluator.workflow.schemas.WorkflowSchema
receptor_coordinate_file
              attribute), 322
                                                                                                      (openff.evaluator.protocols.coordinates.BuildDockedCoordinates
protocol_schemas (openff.evaluator.workflow.schemas.ProtocolGroupSglema 388
               attribute), 318
attribute), 318 receptor_residue_name protocol_schemas (openff.evaluator.workflow.schemas.WorkflowSchema (openff.evaluator.protocols.coordinates.BuildDockedCoordinates
              attribute), 322
                                                                                                      attribute), 389
ProtocolGraph (class in openff.evaluator.workflow),
                                                                                       receptor_residue_name
                                                                                                      (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
ProtocolGroup (class in openff.evaluator.workflow),
                                                                                                      attribute), 598
               310
                                                                                       reference (openff.evaluator.datasets.MeasurementSource
ProtocolGroupSchema
                                                      (class
                                                                                 in
                                                                                                      attribute), 97
               openff.evaluator.workflow.schemas), 317
                                                                                       reference_reduced_potentials
ProtocolPath (class in openff.evaluator.workflow.utils),
                                                                                                      (openff.evaluator.protocols.reweighting.BaseMBARProtocol
               327
                                                                                                      attribute), 554
ProtocolReplicator
                                                     (class
                                                                                       reference_reduced_potentials
              openff.evaluator.workflow.schemas), 319
                                                                                                      (openff.evaluator.protocols.reweighting.ReweightDielectricConsta
protocols(openff.evaluator.protocols.groups.ConditionalGroup
                                                                                                      attribute), 570
              property), 426
                                                                                       reference_reduced_potentials
protocols (openff.evaluator.workflow.ProtocolGraph
```

(openff.evaluator.protocols.reweighting.ReweightObservable

replace\_protocol() (openff.evaluator.protocols.forcefield.BuildTLeapSy

replace\_protocol() (openff.evaluator.protocols.gradients.ZeroGradients.

replace\_protocol() (openff.evaluator.protocols.groups.ConditionalGroups.

 $replace\_protocol()$  (openff.evaluator.protocols.miscellaneous.AddValuator.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.protocols.miscellaneous.protocols.protocols.protocols.protocols.protocols.protocols.prot

replace\_protocol() (openff.evaluator.protocols.miscellaneous.DivideVa

module replace\_protocol() (openff.evaluator.protocols.forcefield.BuildSmirnoff

method), 408

method), 402

method), 414

method), 420

method), 426

*method*), 431

openff.evaluator.workflow), 315 method), 446	
relative_tolerance(openff.evaluator.layers.Calculationteplasce_epreotocol()(openff.evaluator.layers.Calculationteplasce_epreotocol()	enff.evaluator.protocols.miscellaneous.DummyP
attribute), 232 method), 461	
relative_tolerance(openff.evaluator.layers.reweightingnenteighting	enff.evaluator.protocols.miscellaneous.Filter Sub
attribute), 243 method), 457	
relative_tolerance(openff.evaluator.layers.simulation.Siepllatien_Siderocol()(openff.evaluator.layers.simulation.Siepllatien_Siderocol()	enff.evaluator.protocols.miscellaneous.Multiply
attribute), 239 method), 441	
relative_tolerance(openff.evaluator.layers.workflow.Workflow.CalpubitionCollerance)	enff.evaluator.protocols.miscellaneous.Subtract V
attribute), 236 method), 436	
release_lambdas(openff.evaluator.protocols.paprika.restmeiplsaGen.protokelola.@Resp	<b>eaffix</b> valuator.protocols.miscellaneous.WeightBy
attribute), 511 method), 452	
removeprefix() (openff.evaluator.storage.attributes.FileParteplace_protocol() (openff.evaluator.storage.attributes.FileParteplace_protocol()	enff.evaluator.protocols.openmm.OpenMMEnerg
method), 294 method), 467	
$remove suffix () \ (open ff. evaluator. storage. attributes. File {\it Parter} place\_protocol() \ (open ff. evaluator. storage) \ (open ff. evaluator. storage$	enff.evaluator.protocols.openmm.OpenMMEvaluupungungungungungungungungungungungungungu
method), 294 method), 480	
replace() (openff.evaluator.storage.attributes.FilePath replace_protocol()(openff.evaluator.storage.attributes.FilePath replace_protocol()	enff.evaluator.protocols.openmm.OpenMMSimu
method), 294 method), 474	
replace_protocol() (openff.evaluator.protocols.analysisx*ephagad_iphotcic6bf)*t(mphagad_iph	enff.evaluator.protocols.paprika.analysis.Analyz
method), 345 method), 525	
replace_protocol() (openff.evaluator.protocols.analysisx4ephaga4rprotocods() (openff.evaluator.protocols.analysisx4ephaga4rprotocods()	enff.evaluator.protocols.paprika.analysis.Compi
method), 351 method), 535	
replace_protocol() (openff.evaluator.protocols.analysisx4eplagaeQlpscortoblel() (openff.evaluator.protocols.analysisx4eplagaeQlpscortoblel()	enff.evaluator.protocols.paprika.analysis.Compi
method), 339 method), 530	
replace_protocol() (openff.evaluator.protocols.analysis_perkacerogrotocold) (openff.evaluator.protocols.analysis_perkacerogrotocold) (openff.evaluator.protocols.analysis_perkacerogrotocold)	enff.evaluator.protocols.paprika.coordinates.Ad
method), 334 method), 497	
replace_protocol() (openff.evaluator.protocols.analysis_perkperopredatePolifor(b)p	enff.evaluator.protocols.paprika.coordinates.Pre
method), 361 method), 486	
replace_protocol() (openff.evaluator.protocols.analysisxCephpateDipotedColh()) (openff.evaluator.protocol) (openff	enff.evaluator.protocols.paprika.coordinates.Pre
method), 356 method), 491	
replace_protocol() (openff.evaluator.protocols.analysis.peplacelapatohoarolklestop	enff.evaluator.protocols.paprika.restraints.Apply
method), 371 method), 519	
replace_protocol() (openff.evaluator.protocols.analysis.peqclaredapafotipectalry) (openff.evaluator.protocols.analysis.peqclaredapafotipectalry)	enff.evaluator.protocols.paprika.restraints.Gene
method), 366 method), 502	
replace_protocol() (openff.evaluator.protocols.coordinatepBadelCprotibactsPadelprot	wolff.evaluator.protocols.paprika.restraints.Gene
method), 378 method), 508	
replace_protocol() (openff.evaluator.protocols.coordinatepBadelDmdtdColfdinape	esnff.evaluator.protocols.paprika.restraints.Gene
method) 392 method) 513	

replace\_protocol() (openff.evaluator.protocols.coordinatep\substantial and interplace and interp

replace\_protocol() (openff.evaluator.protocols.forcefield:@plkaBailplfsostermol() (openff.evaluator.protocols.reweighting.BaseMBAR

replace\_protocol() (openff.evaluator.protocols.forcefield: Bpilakling Bardients)s(@n(openff.evaluator.protocols.reweighting.Concatenat

method), 552

method), 558

attribute), 563

register\_default\_plugins()

register\_external\_plugins()

register\_thermoml\_property()

register\_workflow\_protocol()

*method*), 385

method), 397

register\_calculation\_layer()

register\_calculation\_schema()

openff.evaluator.layers), 233

openff.evaluator.layers), 233

openff.evaluator.plugins), 626

openff.evaluator.plugins), 626

openff.evaluator.datasets.thermoml), 144

(in

(in

(in

(in

(in

(in

module

module

module

module

module

replace\_protocol() (openff.evaluator.protocols.reweighting.ConcapenperTryujeAtories

replace\_protocol() (openff.evaluator.protocols.reweighting.Reweight@Constant

method), 546

method), 541

property), 352

```
method), 570
                                                                                                                                                                                                                                                required_inputs (openff.evaluator.protocols.analysis.BaseDecorrelatePr
replace_protocol() (openff.evaluator.protocols.reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweighting.Reweigh
                                       method), 563
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.analysis.ComputeDipoleMo
replace_protocol() (openff.evaluator.protocols.simulation.BaseEnergyMin)iniisation
                                       method), 576
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.analysis.DecorrelateObserv
replace_protocol() (openff.evaluator.protocols.simulation.BaseSimulation), 372
                                       method), 583
                                                                                                                                                                                                                                               \verb"required_inputs" (open ff. evaluator. protocols. analysis. Decorrelate Traject to the control of the contro
replace_protocol() (openff.evaluator.protocols.storage.UnpackStoppedSemtwlati6nData
                                       method), 589
                                                                                                                                                                                                                                               required_inputs(openff.evaluator.protocols.coordinates.BuildCoordinates)
replace_protocol() (openff.evaluator.protocols.yank.BaseYankPropoperty), 379
                                       method), 595
                                                                                                                                                                                                                                                {\tt required\_inputs} \ (open \textit{ff.evaluator.protocols.coordinates.BuildDockedCollege}) \ a \textit{topological policy} \ 
replace_protocol() (openff.evaluator.protocols.yank.LigandReceptors\\mathrace\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\r
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.coordinates.SolvateExisting
                                       method), 602
replace_protocol() (openff.evaluator.protocols.yank.SolvationYankPopenty), 385
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.forcefield.BaseBuildSystem
                                       method), 611
replace_protocol() (openff.evaluator.workflow.Protocol
                                                                                                                                                                                                                                                                                       property), 397
                                       method), 306
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.forcefield.BuildLigParGenS
replace_protocol() (openff.evaluator.workflow.ProtocolGroup
                                                                                                                                                                                                                                                                                      property), 408
                                       method), 312
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.forcefield.BuildSmirnoffSyst
replace_protocol() (openff.evaluator.workflow.utils.ProtocolPath property), 403
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.forcefield.BuildTLeapSystem
                                       method), 328
replace_protocol() (openff.evaluator.workflow.Workflow
                                                                                                                                                                                                                                                                                       property), 414
                                       method), 298
                                                                                                                                                                                                                                                {\tt required\_inputs} \ (open \textit{ff.evaluator.protocols.gradients.ZeroGradients}
replace_protocol_types()
                                                                                                                                                                                                                                                                                       property), 420
                                       (openff.evaluator.workflow.schemas.WorkflowSchemauired_inputs(openff.evaluator.protocols.groups.ConditionalGroup
                                       method), 322
                                                                                                                                                                                                                                                                                        property), 426
ReplaceableData
                                                                                                                                                                                                                              in required_inputs(openff.evaluator.protocols.miscellaneous.AddValues
                                       openff.evaluator.storage.data), 265
                                                                                                                                                                                                                                                                                       property), 432
ReplicatorValue
                                                                                                                                          (class
                                                                                                                                                                                                                                             required_inputs (openff.evaluator.protocols.miscellaneous.DivideValue
                                        openff.evaluator.workflow.utils), 326
                                                                                                                                                                                                                                                                                       property), 447
Request (class in openff.evaluator.client), 80
                                                                                                                                                                                                                                               required_inputs(openff.evaluator.protocols.miscellaneous.DummyProto
request_estimate() (openff.evaluator.client.EvaluatorClient
                                                                                                                                                                                                                                                                                       property), 462
                                       method), 77
                                                                                                                                                                                                                                                required_inputs (openff.evaluator.protocols.miscellaneous.FilterSubstar
request_url (openff.evaluator.forcefield.LigParGenForceFieldSourcproperty), 457
                                       property), 226
                                                                                                                                                                                                                                                required_inputs (openff.evaluator.protocols.miscellaneous.MultiplyValu
RequestOptions (class in openff.evaluator.client), 82
                                                                                                                                                                                                                                                                                       property), 442
RequestResult (class in openff.evaluator.client), 84
                                                                                                                                                                                                                                               required\_inputs (openff.evaluator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.SubtractValuator.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.
required_effective_samples
                                                                                                                                                                                                                                                                                       property), 437
                                       (openff.evaluator.protocols.reweighting.BaseMBARPquixced_inputs (openff.evaluator.protocols.miscellaneous.WeightByMo
                                       attribute), 555
                                                                                                                                                                                                                                                                                       property), 452
required_effective_samples
                                                                                                                                                                                                                                               required_inputs (openff.evaluator.protocols.openmm.OpenMMEnergyM
                                       (openff.evaluator.protocols.reweighting.ReweightDielectric Comptents), 467
                                                                                                                                                                                                                                               \verb"required_inputs" (open \textit{ff.evaluator.protocols.open mm.Open MME valuate inputs) (open \textit{ff.evaluator.protocols.open mm.open mm.op
                                       attribute), 570
required_effective_samples
                                                                                                                                                                                                                                                                                       property), 480
                                       (openff.evaluator.protocols.reweighting.Reweight Obequivable_inputs (openff.evaluator.protocols.openmm.OpenMMSimulation
                                       attribute), 564
                                                                                                                                                                                                                                                                                       property), 475
required_inputs (openff.evaluator.protocols.analysis.AvenegaDiredathinfwastorpenff.evaluator.protocols.paprika.analysis.AnalyzeAF
                                       property), 346
                                                                                                                                                                                                                                                                                       property), 526
```

required\_inputs(openff.evaluator.protocols.analysis.AverageObservable

required\_inputs(openff.evaluator.protocols.analysis.BaseAverageObser

686 Index

required\_inputs (openff.evaluator.protocols.analysis.AvenequFireEnirquats (openff.evaluator.protocols.paprika.analysis.ComputeR

property), 536

```
required_inputs (openff.evaluator.protocols.paprika.analysis.CompateSoftianalreConhaveiosuworkflow.WorkflowCalculationLayer
         property), 531
                                                                  class method), 234
required_inputs (openff.evaluator.protocols.paprika.coom/istaneaiAndd/Isalmenn/Ast@ppenff.evaluator.protocols.paprika.restraints.Genera
         property), 497
                                                                  attribute), 503
required_inputs (openff.evaluator.protocols.paprika.coombisturaiPrepscHendsCoopdinffinexaluator.protocols.paprika.restraints.Genera
                                                                  attribute), 508
         property), 487
required_inputs (openff.evaluator.protocols.paprika.coombistaneaiPrepschPedass/aGonfflevalexator.protocols.paprika.restraints.Genero
         property), 492
                                                                  attribute), 514
required_inputs (openff.evaluator.protocols.paprika.restmestschipplyReypa@upenff.evaluator.protocols.yank.LigandReceptorYankPro
         property), 520
                                                                  attribute), 598
required_inputs (openff.evaluator.protocols.paprika.restmaistxx@intersupAtthd@pestff.eintdsuator.protocols.paprika.analysis.AnalyzeAF
                                                                  attribute), 522
         property), 503
required_inputs (openff.evaluator.protocols.paprika.restmastx@intrsupAttMRopartfftevaluator.protocols.paprika.analysis.ComputeR
                                                                  attribute), 532
         property), 508
required_inputs (openff.evaluator.protocols.paprika.restmints.ApplyRes
         property), 514
                                                                  attribute), 516
required_inputs (openff.evaluator.protocols.reweighting. Berst Evaluator.protocols.paprika.restraints. Generate.
         property), 552
                                                                  attribute), 503
required_inputs (openff.evaluator.protocols.reweighting.BerstMHARB_uptath(openff.evaluator.protocols.paprika.restraints.Generate
         property), 558
                                                                  attribute), 508
required_inputs (openff.evaluator.protocols.reweighting. Contractive Openff.evaluator.protocols.paprika.restraints. Generate
                                                                  attribute), 514
         property), 547
required_inputs (openff.evaluator.protocols.reweighting. Cossalue inputs (openff.evaluator.protocols.analysis. Average Free Energies
                                                                  attribute), 348
         property), 541
required_inputs (openff.evaluator.protocols.reweighting.Resvalghthehftevialhontstraptotocols.miscellaneous.AddValues
         property), 570
                                                                  attribute), 429
required_inputs (openff.evaluator.protocols.reweighting.Resulstin@hendf.walalleator.protocols.miscellaneous.DivideValue
         property), 564
                                                                  attribute), 443
required_inputs (openff.evaluator.protocols.simulation.BreasEliter(gyMinthibenisdutiator.protocols.miscellaneous.MultiplyValue
         property), 576
                                                                  attribute), 438
required_inputs (openff.evaluator.protocols.simulation.Brasslitutlapionff.evaluator.protocols.miscellaneous.SubtractValues
         property), 583
                                                                  attribute), 434
required_inputs (openff.evaluator.protocols.storage.UnpacelsState(tSjinnffaevahl)awa.protocols.paprika.analysis.AnalyzeAPRPhase
         property), 589
                                                                  attribute), 522
required_inputs (openff.evaluator.protocols.yank.BaseYankBudto(openff.evaluator.protocols.paprika.analysis.ComputeReferenceWor
         property), 595
                                                                  attribute), 533
required_inputs (openff.evaluator.protocols.yank.LigandReswittor.brotocols.paprika.analysis.ComputeSymmetryCon
         property), 603
                                                                  attribute), 528
required_inputs (openff.evaluator.protocols.yank.SolvativnEvalktSiQ)o(aplenff.evaluator.client.Request method), 81
                                                        retain_packmol_files
         property), 611
required_inputs (openff.evaluator.workflow.Protocol
                                                                  (openff.evaluator.protocols.coordinates.BuildCoordinatesPackmo
         property), 305
                                                                  attribute), 375
required_inputs(openff.evaluator.workflow.ProtocolGrowpetain_packmol_files
                                                                  (openff.evaluator.protocols.coordinates.SolvateExistingStructure
         property), 311
required_schema_type()
                                                                  attribute), 385
         (openff.evaluator.layers.CalculationLayer
                                                        retrieve_force_field()
                                                                  (openff.evaluator.storage.LocalFileStorage
         class method), 229
                                                                  method), 259
required_schema_type()
         (openff.evaluator.layers.reweighting.ReweightingLretrieve_force_field()
         class method), 240
                                                                  (openff.evaluator.storage.StorageBackend
required_schema_type()
                                                                  method), 257
         (openff.evaluator.layers.simulation.SimulationLayeretrieve_object() (openff.evaluator.storage.LocalFileStorage
         class method), 237
                                                                  method), 259
required_schema_type()
                                                        retrieve_object() (openff.evaluator.storage.StorageBackend
```

method), 257	${\tt schema}(open \textit{ff.evaluator.protocols.analysis.} A \textit{verageObservable}$
retrieve_results() (openff.evaluator.client.EvaluatorC	lient property), 340
method), 77	${\tt schema}(open \textit{ff.evaluator.protocols.analysis.Base Average Observable}$
ReweightDielectricConstant (class in	property), 334
openff.evaluator.protocols.reweighting), 565	${\tt schema}(open \textit{ff.evaluator.protocols.analysis.BaseDecorrelateProtocol}$
ReweightingLayer (class in	property), 362
openff.evaluator.layers.reweighting), 240	${\tt schema}(open ff. evaluator. protocols. analysis. Compute Dipole Moments$
ReweightingProtocols (class in	property), 357
openff.evaluator.protocols.utils), 614	${\tt schema}(open \textit{ff.evaluator.protocols.analysis.DecorrelateObservables}$
ReweightingSchema (class in	property), 372
openff.evaluator.layers.reweighting), 241	${\tt schema}(open \textit{ff.evaluator.protocols.analysis.} Decorrelate \textit{Trajectory}$
ReweightObservable (class in	property), 367
openff.evaluator.protocols.reweighting), 559	${\tt schema}(open ff. evaluator. protocols. coordinates. Build Coordinates Packmonth and Coordi$
rfind() (openff.evaluator.storage.attributes.FilePath	property), 379
method), 294	${\tt schema}(open \textit{ff.evaluator.protocols.coordinates.BuildDockedCoordinates})$
$\verb"right_hand_value" (open ff. evaluator. protocols. groups. College of the control of the cont$	nditionalG <b>poupeCty)</b> ,ditidn
attribute), 422	${\tt schema}(open \textit{ff.evaluator.protocols.coordinates.Solvate Existing Structure}$
<pre>rindex() (openff.evaluator.storage.attributes.FilePath</pre>	property), 385
method), 294	${\tt schema}(open ff. evaluator. protocols. force field. Base Build System$
rjust() (openff.evaluator.storage.attributes.FilePath	property), 397
method), 294	${\tt schema}(open \textit{ff.evaluator.protocols.force field.BuildLigParGenSystem}$
role (openff.evaluator.substances.Component attribute),	property), 408
128	${\tt schema}(open \textit{ff.evaluator.protocols.force field.BuildSmirno \textit{ffSystem}}$
$\verb"root_directory" (open \textit{ff.evaluator.storage.Local File Storage}. Local File Storage (open \textit{ff.evaluator.storage}) and the property of th$	ge property), 403
property), 258	${\tt schema}(open ff. evaluator. protocols. force field. Build TLeap System$
$\verb"root_protocols" (open ff. evaluator. work flow. Protocol Graphics) and the protocol of the$	ph property), 414
property), 309	${\tt schema}(open \textit{ff.evaluator.protocols.gradients.Zero Gradients}$
$\verb"root_protocols" (open ff.evaluator.workflow.WorkflowGrades) and the protocols of the $	ph property), 420
property), 301	${\tt schema}(open \textit{ff.evaluator.protocols.groups.} Conditional Group$
${\tt rpartition()}\ (open {\it ff.evaluator.storage.attributes.} File {\it Patallicians}) and {\it file} {\it patallicians}) and {\it file} {\it file}$	
method), 295	${\tt schema}(open ff. evaluator. protocols. miscellaneous. Add Values$
rsplit() (openff.evaluator.storage.attributes.FilePath	property), 432
method), 295	${\tt schema}(open \textit{ff.evaluator.protocols.miscellaneous.DivideValue}$
<pre>rstrip() (openff.evaluator.storage.attributes.FilePath</pre>	property), 447
method), 295	${\tt schema}(open \textit{ff.evaluator.protocols.miscellaneous.DummyProtocol}$
0	property), 462
S	${\tt schema}(open ff. evaluator. protocols. miscellaneous. Filter Substance By Role, and the protocol for t$
<pre>schedule_calculation()</pre>	property), 457
(openff.evaluator.layers.CalculationLayer	${\tt schema}(open ff. evaluator. protocols. miscellaneous. Multiply Value$
class method), 229	property), 442
<pre>schedule_calculation()</pre>	${\tt schema}(open\!f\!f\!.evaluator.protocols.miscellaneous.SubtractValues$
(openff.evaluator.layers.reweighting.Reweighting	
class method), 240	${\tt schema} \ (open ff. evaluator. protocols. miscellaneous. Weight By Mole Fraction for the protocol for t$
<pre>schedule_calculation()</pre>	property), 452
(openff.evaluator.layers.simulation.SimulationLa	$y$ s $oldsymbol{arepsilon}$ hema ( $openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation$
class method), 237	property), 467
<pre>schedule_calculation()</pre>	${\tt schema}(open \textit{ff.evaluator.protocols.openmm.OpenMME} valuate \textit{Energies}$
(openff.evaluator.layers.workflow.WorkflowCalcu	lationLayeproperty), 480
class method), 234	schema (openff.evaluator.protocols.openmm.OpenMMSimulation
${\tt schema}(open \textit{ff.evaluator.protocols.analysis.Average Dielec}$	tricConstaPtroperty), 475
property), 346	schema (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase
${\tt schema}(open \textit{ff.evaluator.protocols.analysis.AverageFreeEndown})$	nergies property), 526
property), 352	schema (openff.evaluator.protocols.paprika.analysis.ComputeReferenceW property), 536

```
schema (openff.evaluator.protocols.paprika.analysis.ComputeSymmetryenffrevtibuator.datasets.curation.components.selection), property), 531 206
```

- schema (openff.evaluator.protocols.paprika.coordinates.Add**Hermeny**Atddusess (openff.evaluator.client.ConnectionOptions property), 497 attribute), 79
- schema (openff.evaluator.protocols.paprika.coordinates.Preparededladarinstaspenff.evaluator.client.EvaluatorClient property), 487 property), 77
- schema (openff.evaluator.protocols.paprika.coordinates.PreparaRedepsacta@phinfftesaluator.client.ConnectionOptions property), 492 attribute), 79
- schema (openff.evaluator.protocols.paprika.restraints.Apply Rextraintport (openff.evaluator.client.EvaluatorClient property), 520 property), 77
- schema (openff.evaluator.protocols.paprika.restraints.GenersetAtuicleRes(repienff.evaluator.protocols.analysis.AverageDielectricConstant property), 503 method), 346
- schema (openff.evaluator.protocols.paprika.restraints.Gene**rsat**PullRestpalmenff.evaluator.protocols.analysis.AverageFreeEnergies property), 508 method), 352
- schema (openff.evaluator.protocols.paprika.restraints.GenersætRehais&Rek
- schema (openff.evaluator.protocols.reweighting.BaseEvalua**seFnergics**() (openff.evaluator.protocols.analysis.BaseAverageObservable property), 552 method), 334
- schema (openff.evaluator.protocols.reweighting.BaseMBARBettoutld() (openff.evaluator.protocols.analysis.BaseDecorrelateProtocol property), 558 method), 362
- schema (openff.evaluator.protocols.reweighting.Concatenate@bservibles (openff.evaluator.protocols.analysis.ComputeDipoleMoments property), 547 method), 357
- schema (openff.evaluator.protocols.reweighting.Concatenate Frijania de) (openff.evaluator.protocols.analysis.DecorrelateObservables property), 541 method), 372
- schema (openff.evaluator.protocols.reweighting.ReweightDi**skettnixGdi**) tunpenff.evaluator.protocols.analysis.DecorrelateTrajectory property), 570 method), 367
- schema (openff.evaluator.protocols.reweighting.ReweightObsetyahlæd() (openff.evaluator.protocols.coordinates.BuildCoordinatesPack property), 564 method), 379
- schema (openff.evaluator.protocols.simulation.BaseEnergyMatrimixxititi) (openff.evaluator.protocols.coordinates.BuildDockedCoordinates.property), 576 method), 392
  schema (openff.evaluator.protocols.simulation.BaseSimulationates.SolvateExistingStructure)
- property), 583 method), 385 schema (openff.evaluator.protocols.storage.UnpackStoredSinsertarioxilda(i)a(openff.evaluator.protocols.forcefield.BaseBuildSystem
- schema (openff.evaluator.protocols.storage.UnpackStoredStreetStoredStreetatwandata(openff.evaluator.protocols.forcefield.BaseBuildSystem property), 589 method), 397
- schema (openff.evaluator.protocols.yank.BaseYankProtocol set\_uuid() (openff.evaluator.protocols.forcefield.BuildLigParGenSystem property), 595 method), 408
- schema (openff.evaluator.protocols.yank.LigandReceptorYan**slePtrotwicd()** (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem property), 603 method), 403
- $schema (\textit{openff.evaluator.protocols.yank.SolvationYankProtocols.yank.SolvationYankProtocols.yank.SolvationYankProtocols.yank.org. (\textit{openff.evaluator.protocols.forcefield.BuildTLeapSystem property)}, 611 \\ \textit{method}), 414$
- schema (openff.evaluator.workflow.Protocol property), set\_uuid() (openff.evaluator.protocols.gradients.ZeroGradients 305 method), 420
- schema (openff.evaluator.workflow.ProtocolGroup property), 314 set\_uuid() (openff.evaluator.protocols.groups.ConditionalGroup method), 427
- SelectDataPoints (class in set\_uuid() (openff.evaluator.protocols.miscellaneous.AddValues openff.evaluator.datasets.curation.components.selection), method), 432
  212 set\_uuid() (openff.evaluator.protocols.miscellaneous.DivideValue
- SelectDataPointsSchema (class in method), 447
  openff.evaluator.datasets.curation.components.selection), id() (openff.evaluator.protocols.miscellaneous.DummyProtocol
- openff.evaluator.datasets.curation.components.selsettompid() (openff.evaluator.protocols.miscellaneous.DummyProtocol
  210
  method), 462
  SelectSubstances (class in set\_uuid() (openff.evaluator.protocols.miscellaneous.FilterSubstanceByR
- openff.evaluator.datasets.curation.components.selection), method), 457
  209 set\_uuid() (openff.evaluator.protocols.miscellaneous.MultiplyValue

SelectSubstancesSchema (class in method), 442

- set\_uuid() (openff.evaluator.protocols.miscellaneous.Subt**sect\_Valuies**() (openff.evaluator.workflow.Protocol method), 437 method), 306
- set\_uuid() (openff.evaluator.protocols.miscellaneous.WeiglerByWhileF)rac(inpenff.evaluator.workflow.ProtocolGroup method), 452 method), 312
- set\_uuid() (openff.evaluator.protocols.openmm.OpenMMEntrgyMinin)s(apemff.evaluator.protocols.analysis.AverageDielectricConstant method), 468 method), 346
- set\_uuid() (openff.evaluator.protocols.openmm.OpenMMEetluntelFree(releaning), 481 method), 352
- set\_uuid() (openff.evaluator.protocols.openmm.OpenMMSiertulnathue() (openff.evaluator.protocols.analysis.AverageObservable method), 475 method), 340
- set\_uuid() (openff.evaluator.protocols.paprika.analysis.AselyzwAPReVintopenff.evaluator.protocols.analysis.BaseAverageObservable method), 526

  method), 334
- set\_uuid() (openff.evaluator.protocols.paprika.analysis.ComputeReferen(aprenfkevaluator.protocols.analysis.BaseDecorrelateProtocols.BaseDecorrelateProtocols.BaseDecorrel
- set\_uuid() (openff.evaluator.protocols.paprika.analysis.ComputeDywer();twy/Geonff.extinbuator.protocols.analysis.ComputeDipoleMoment. method), 357
- set\_uuid() (openff.evaluator.protocols.paprika.coordinate**set**dd**lalum**(y**A**(topenff.evaluator.protocols.analysis.DecorrelateObservables method), 497 method), 372
- set\_uuid() (openff.evaluator.protocols.paprika.coordinate**ssPrepalare**(**I)Copedifiates**luator.protocols.analysis.DecorrelateTrajectory method), 487 method), 367
- set\_uuid() (openff.evaluator.protocols.paprika.coordinatessetrepakel@dsq@nffrelinhtasor.protocols.coordinates.BuildCoordinatesPadmethod), 492

  method), 379
- set\_uuid() (openff.evaluator.protocols.paprika.restraints. ApplyRaluativ. (openff.evaluator.protocols.coordinates. BuildDockedCoordinates. BuildDocked

set\_uuid() (openff.evaluator.protocols.paprika.restraints. General Ann () (Openff.evaluator.protocols.coordinates. Solvate Existing Struc

- method), 503

  method), 386

  set\_wid() (openff evaluator protocols paprika restraints General Poll Restraints evaluator protocols forcefield Base Build System
- set\_uuid() (openff.evaluator.protocols.paprika.restraints.GeneralBellRestpainff.evaluator.protocols.forcefield.BaseBuildSystem method), 509 method), 398
- set\_uuid() (openff.evaluator.protocols.paprika.restraints. **General Releas esperificinus** uator.protocols.forcefield.BuildLigParGenSystem method), 514 method), 409
- set\_uuid() (openff.evaluator.protocols.reweighting.BaseEscetuareEncer(s)e(copenff.evaluator.protocols.forcefield.BuildSmirnoffSystem method), 552 method), 403
- set\_uuid() (openff.evaluator.protocols.reweighting.BaseMSPARPedrae&) (openff.evaluator.protocols.forcefield.BuildTLeapSystem method), 558 method), 414

  set\_uuid() (openff.evaluator.protocols.reweighting.ConcapatyteAbyeA) (openff.evaluator.protocols.gradients.ZeroGradients
- set\_uuid() (openff.evaluator.protocols.reweighting.Conca**setus@bse**(subspenff.evaluator.protocols.gradients.ZeroGradients method), 547 method), 420
- set\_uuid() (openff.evaluator.protocols.reweighting.Conca**sentate/Frlajec()**/liopenff.evaluator.protocols.groups.ConditionalGroup method), 541 method), 427
- set\_uuid() (openff.evaluator.protocols.reweighting.Reweigsted) whether the complete of the control of the contr
- set\_uuid() (openff.evaluator.protocols.reweighting.Rew
- set\_uuid() (openff.evaluator.protocols.simulation.BaseEn**s**ey\_Windiwie(a) (openff.evaluator.protocols.miscellaneous.DummyProtocol method), 576 method), 462
- set\_uuid() (openff.evaluator.protocols.simulation.BaseSinsultivalue() (openff.evaluator.protocols.miscellaneous.FilterSubstanceBy method), 583 method), 457
- set\_uuid() (openff.evaluator.protocols.storage.UnpackStosselSimalatic()Dapenff.evaluator.protocols.miscellaneous.MultiplyValue method), 589 method), 442
- set\_uuid() (openff.evaluator.protocols.yank.BaseYankProsetoLvalue() (openff.evaluator.protocols.miscellaneous.SubtractValues method), 595 method), 437
- method), 437
  set\_uuid() (openff.evaluator.protocols.yank.LigandReceptseYankHwesGolopenff.evaluator.protocols.miscellaneous.WeightByMoleFra
  method), 603
  method), 452
- set\_uuid() (openff.evaluator.protocols.yank.SolvationYankPerotocols.openff.evaluator.protocols.openmm.OpenMMEnergyMinimi.method), 468

set_value() (openff.evaluator.protocols.openmm.OpenMMEt		ocols.yank.BaseYan	kProtocol
method), 481 set_value() (openff.evaluator.protocols.openmm.OpenMM⁢	attribute), 592	aala wank LiaandD	aaantauVankDuotaaa
method), 475	<b>ир<sub>а</sub>ахир</b> (openjj.evanaaior.proid attribute), 603	cois.yank.Ligana <b>k</b>	есеріот і апкі тоюсої
set_value() (openff.evaluator.protocols.paprika.analysis.Aset	ypeARP/wysenff.evaluator.proto	ocols.yank.Solvatioi	nYankProtocol
method), 526	attribute), 612		
set_value() (openff.evaluator.protocols.paprika.analysis.&m	pl <i>atRefo<u>re</u>dataWopa</i> th		
method), 536	(openff.evaluator.protocol	s.storage.UnpackSt	oredSimulationData
<pre>set_value() (openff.evaluator.protocols.paprika.analysis.Com</pre>		0 1	
	ulationDataQuery	(class	in
set_value() (openff.evaluator.protocols.paprika.coordinates.A			
			in
set_value() (openff.evaluator.protocols.paprika.coordinates.l	•		
	ulationProtocols		in
set_value() (openff.evaluator.protocols.paprika.coordinates.l		`	
			in
set_value() (openff.evaluator.protocols.paprika.restraints.Ap			li l
	les (openff.evaluator.substan		at-
set_value() (openff.evaluator.protocols.paprika.restraints.Ge		ces.Component t	ui-
		(alaaa	:
	rnoffForceFieldSource	`	in
set_value() (openff.evaluator.protocols.paprika.restraints.Ge			D . 1
	ute (openff.evaluator.protocols.	yank.SolvationYank	<i>Protocol</i>
<pre>set_value() (openff.evaluator.protocols.paprika.restraints.Ge</pre>			
	ute_coordinate_file		
$\verb set_value()  (open ff.evaluator.protocols.reweighting.Base Evaluations and the proposed proposed$		s.coordinates.Solva	teExistingStructure
method), 552	attribute), 381		
<pre>set_value() (openff.evaluator.protocols.reweighting.BaseModel</pre>			
method), 558	(openff.evaluator.protocol	s.yank.SolvationYai	nkProtocol
$\verb set_value()  (open ff. evaluator. protocols. reweighting. Concater)  \\$			
method), 547 sol	ution_1_free_energy		
<pre>set_value() (openff.evaluator.protocols.reweighting.Concater method), 542</pre>	ateTr <b>(jepenff.es</b> valuator.protocol attribute), 607	s.yank.SolvationYai	nkProtocol
set_value() (openff.evaluator.protocols.reweighting.Reweighting	**	ator.protocols.vank	.SolvationYankProto
method), 570	attribute), 606	4	
set_value() (openff.evaluator.protocols.reweighting.Re		101 1	1D . 1
method), 564	(openff.evaluator.protocol	s.yank.SolvationYai	nkProtocol
set_value() (openff.evaluator.protocols.simulation.BaseEnerg			
	ution_2_coordinates		
set_value() (openff.evaluator.protocols.simulation.BaseSimulation), 583	ation (openff.evaluator.protocol attribute), 606	s.yank.SolvationYai	nkProtocol
set_value() (openff.evaluator.protocols.storage.UnpackStsm2	Strumhat2onff2ete_energy		
method), 589	(openff.evaluator.protocol	s.yank.SolvationYa	nkProtocol
<pre>set_value() (openff.evaluator.protocols.yank.BaseYankProtocols.</pre>			
	ution_2_system( <i>openff.evalua</i>	ator.protocols.vank	.SolvationYankProto
set_value() (openff.evaluator.protocols.yank.LigandReceptor		aro aprovo construir	
	ution_2_trajectory_path		
set_value() (openff.evaluator.protocols.yank.SolvationYankPi		s vank SolvationVa	nkProtocol
method), 612	attribute), 608		iki rotocot
<pre>set_value() (openff.evaluator.workflow.Protocol sol</pre>	_		
method), 307	(openff.evaluator.protocol	s.yank.LigandRecej	ptorYankProtocol
<pre>set_value() (openff.evaluator.workflow.ProtocolGroup</pre>	attribute), 598		
	vated_complex_system		
$\verb setdefault()  (open ff. evaluator. utils. observables. Observable For the property of the $	rame (openff.evaluator.protocol	s.yank.LigandRecej	ptorYankProtocol
method), 625	attribute), 598		

```
solvated_complex_trajectory_path
                                                         source_calculation_id
         (openff.evaluator.protocols.yank.LigandReceptorYankProtocolpenff.evaluator.storage.data.StoredFreeEnergyData
         attribute), 599
                                                                  attribute), 275
solvated_ligand_coordinates
                                                         source_calculation_id
         (openff.evaluator.protocols.yank.LigandReceptorYankProtocolpenff.evaluator.storage.data.StoredSimulationData
                                                                  attribute), 272
         attribute), 598
solvated_ligand_system
                                                         source calculation id
         (openff.evaluator.protocols.yank.LigandReceptorYankProtoc@penff.evaluator.storage.query.BaseSimulationDataQuery
         attribute), 598
                                                                  attribute), 282
solvated_ligand_trajectory_path
                                                         source_calculation_id
         (openff.evaluator.protocols.yank.LigandReceptorYankProtocolpenff.evaluator.storage.query.FreeEnergyDataQuery
         attribute), 599
                                                                  attribute), 289
SolvateExistingStructure
                                                         source_calculation_id
                                      (class
                                                    in
         openff.evaluator.protocols.coordinates), 379
                                                                  (openff.evaluator.storage.query.SimulationDataQuery
SolvationFreeEnergy
                                   (class
                                                    in
                                                                  attribute), 286
         openff.evaluator.properties), 116
                                                         sources (openff.evaluator.datasets.PhysicalPropertyDataSet
SolvationYankProtocol
                                                                  property), 138
                                    (class
                                                    in
         openff.evaluator.protocols.vank), 604
                                                         sources (openff.evaluator.datasets.taproom.TaproomDataSet
solvent_1(openff.evaluator.protocols.yank.SolvationYankProtocol property), 148
         attribute), 606
                                                         sources (openff.evaluator.datasets.thermoml.ThermoMLDataSet
solvent_1_coordinate_path
                                                                  property), 143
         (openff.evaluator.protocols.yank.SolvationYankProsphalt ()
                                                                     (openff.evaluator.storage.attributes.FilePath
         attribute), 607
                                                                  method), 295
solvent_1_trajectory_path
                                                         splitlines() (openff.evaluator.storage.attributes.FilePath
         (openff.evaluator.protocols.yank.SolvationYankProtocol
                                                                  method), 295
         attribute), 607
                                                         start() (openff.evaluator.backends.CalculationBackend
solvent_2(openff.evaluator.protocols.yank.SolvationYankProtocol method), 244
         attribute), 606
                                                         start() (openff.evaluator.backends.dask.BaseDaskBackend
solvent_2_coordinate_path
                                                                  method), 249
         (openff.evaluator.protocols.yank.SolvationYankProstbeath () (openff.evaluator.backends.dask.BaseDaskJobQueueBackend
         attribute), 608
                                                                  method), 250
solvent_2_trajectory_path
                                                         start() (openff.evaluator.backends.dask.DaskLocalCluster
         (openff.evaluator.protocols.yank.SolvationYankProtocol
                                                                  method), 251
         attribute), 608
                                                         {\tt start()}\ (openff.evaluator.backends.dask.DaskLSFBackend
Source (class in openff.evaluator.datasets), 95
                                                                  method), 253
source (openff.evaluator.datasets.PhysicalProperty at-
                                                        start() (openff.evaluator.backends.dask.DaskPBSBackend
         tribute), 93
                                                                  method), 255
source (openff.evaluator.properties.Density attribute),
                                                        start()
                                                                         (openff.evaluator.server.EvaluatorServer
                                                                  method), 88
source (openff.evaluator.properties.DielectricConstant
                                                        {\tt start\_protocol}\ (open ff. evaluator. work flow. utils. Protocol Path
         attribute), 108
                                                                  property), 328
         (openff.evaluator.properties.EnthalpyOfMixing
                                                        start_state_trajectory
source
         attribute), 112
                                                                  (openff.evaluator.storage.data.StoredFreeEnergyData
\verb+source+ (open ff. evaluator. properties. Enthalpy Of Vaporization
                                                                  attribute), 274
         attribute), 115
                                                         started (openff.evaluator.backends.CalculationBackend
source (openff.evaluator.properties.ExcessMolarVolume
                                                                  property), 244
                                                        \verb|started| (open ff. evaluator. backends. dask. Base Dask Backend|
         attribute), 105
source (openff.evaluator.properties.HostGuestBindingAffinity
                                                                  property), 249
         attribute), 122
                                                         started(openff.evaluator.backends.dask.BaseDaskJobQueueBackend
source (openff.evaluator.properties.SolvationFreeEnergy
                                                                  property), 251
         attribute), 119
                                                         \verb|started|| (open ff. evaluator. backends. dask. Dask Local Cluster||
                                                                  property), 251
source_calculation_id
         (openff.evaluator.storage.data.BaseSimulationDatstarted(openff.evaluator.backends.dask.DaskLSFBackend
         attribute), 267
                                                                  property), 253
```

$\verb started   (open ff. evaluator. backends. dask. Dask PBSB ackends. dask. $	
property), 255	<pre>store_object() (openff.evaluator.storage.StorageBackend</pre>
${\tt startswith()}\ (open \emph{ff.evaluator.storage.attributes.File Path)}$	
method), 295	StoredFreeEnergyData (class in
State (class in openff.evaluator.datasets.curation.compone	17 00
213	StoredSimulationData (class in
statistical_inefficiency	openff.evaluator.storage.data), 269
(openff.evaluator.storage.data.StoredSimulationDatribute), 271	method), 295
steps_per_iteration	${\tt submit\_task()}\ (open ff. evaluator. backends. Calculation Backend$
$(open {\it ff.evaluator.protocols.open mm. Open MMS in}$	
attribute), 475	submit_task() (openff.evaluator.backends.dask.BaseDaskBackend
steps_per_iteration	method), 249
(openff.evaluator.protocols.simulation.BaseSimulatribute), 578	<pre>lasiubmit_task() (openff.evaluator.backends.dask.BaseDaskJobQueueBack</pre>
steps_per_iteration	$\verb submit_task()  (open ff. evaluator. backends. dask. Dask Local Cluster $
(openff.evaluator.protocols.yank.BaseYankProtoc	
attribute), 591	<pre>submit_task() (openff.evaluator.backends.dask.DaskLSFBackend</pre>
steps_per_iteration	method), 253
	Yauhniatotakk() (openff.evaluator.backends.dask.DaskPBSBackend
attribute), 603	method), 255
steps_per_iteration	subset() (openff.evaluator.utils.observables.ObservableArray
(openff.evaluator.protocols.yank.SolvationYankPr attribute), 612	rotocol method), 622 subset() (openff.evaluator.utils.observables.ObservableFrame
steric_lambdas_1(openff.evaluator.protocols.yank.Solva	
attribute), 607	Substance (class in openff.evaluator.substances), 123
steric_lambdas_2 (openff.evaluator.protocols.yank.Solva	
attribute), 607	attribute), 93
stop() (openff.evaluator.backends.CalculationBackend	
method), 244	tribute), 102
stop() (openff.evaluator.backends.dask.BaseDaskBackend	substance (openff.evaluator.properties.DielectricConstant
method), 249	attribute), 109
	usBbskanke (openff.evaluator.properties.EnthalpyOfMixing
method), 251	attribute), 112 substance (openff.evaluator.properties.EnthalpyOfVaporization
method), 251	attribute), 116
	substance (openff.evaluator.properties.ExcessMolarVolume
	attribute), 105
stop() (openff.evaluator.backends.dask.DaskPBSBackend method), 255	substance (openff.evaluator.properties.HostGuestBindingAffinity attribute), 123
stop() (openff.evaluator.server.EvaluatorServer	substance (openff.evaluator.properties.SolvationFreeEnergy
method), 88	attribute), 119
storage_queries (openff.evaluator.layers.reweighting.Re attribute), 241	www.gstiagse/@penff.evaluator.protocols.coordinates.BuildCoordinatesPackr attribute), 375
	substance (openff.evaluator.protocols.coordinates.SolvateExistingStructur
openff.evaluator.storage.attributes), 296	attribute), 386
StorageBackend (class in openff.evaluator.storage), 256	substance (openff.evaluator.protocols.forcefield.BaseBuildSystem
store_force_field()	attribute), 394
(openff.evaluator.storage.LocalFileStorage	substance (openff.evaluator.protocols.forcefield.BuildLigParGenSystem
method), 259	attribute), 409
store_force_field()	$\verb+substance+ (open \textit{ff.evaluator.protocols.force} field. \textit{BuildSmirnoffSystem}$
(openff. evaluator. storage. Storage Backend	attribute), 403
method), 256	$\verb+substance+ (open ff. evaluator. protocols. force field. Build TLeap System+$
<pre>store_object() (openff.evaluator.storage.LocalFileStorage</pre>	ge attribute), 414

attribute), 564

```
substance (openff.evaluator.protocols.paprika.coordinates.PackHeitSchackToms
                                                                                                                                         (class
              attribute), 493
                                                                                                        openff.evaluator.datasets.curation.components.selection),
substance (openff.evaluator.protocols.paprika.coordinates.PreparePulCoordinates
                                                                                         {\tt temperature} \ (open \textit{ff.evaluator.thermodynamics.} Thermodynamic \textit{State}
              attribute), 487
substance (openff.evaluator.protocols.paprika.coordinates.PrepareRatexibeCa) on diffrates
              attribute), 492
                                                                                         temperature_cutoff(openff.evaluator.layers.reweighting.ReweightingSc
substance (openff.evaluator.protocols.storage.UnpackStoredSimulatiattPhtae), 242
                                                                                         thermodynamic_state
              attribute), 585
substance (openff.evaluator.storage.data.BaseSimulationData
                                                                                                        (openff.evaluator.datasets.PhysicalProperty
                                                                                                        attribute), 92
              attribute), 267
substance(openff.evaluator.storage.data.StoredFreeEnergthetmodynamic_state
                                                                                                        (openff.evaluator.properties.Density attribute),
              attribute), 276
substance\ (open ff. evaluator. storage. data. Stored Simulation Data
                                                                                         thermodynamic_state
              attribute), 272
\verb|substance|| (openff.evaluator.storage.query. Base Simulation Data Query. openff.evaluator.properties. Dielectric Constant (openff.evaluator.properties. Dielectric Constant (openff.evaluator.properties.) (openff.evaluator.properties. Dielectric Constant (openff.evaluator.properties. Dielectric Constant (openff.evaluator.properties.) (openff.evaluator.properties. Dielectric Constant (openff.evaluator.properties.) (openff.evaluator.properties. Dielectric Constant (openff.evaluator.properties. Dielectric Constant (openff.evaluator.propertie
              attribute), 282
                                                                                                        attribute), 109
substance (openff.evaluator.storage.query.FreeEnergyDatathermodynamic_state
              attribute), 289
                                                                                                        (openff.evaluator.properties.EnthalpyOfMixing
substance\ (open ff. evaluator. storage. query. Simulation Data Query
                                                                                                        attribute), 112
                                                                                         thermodynamic_state
              attribute), 286
substance_query (openff.evaluator.storage.query.BaseSimulationDataQuffrgvaluator.properties.EnthalpyOfVaporization
              attribute), 282
                                                                                                        attribute), 116
substance_query (openff.evaluator.storage.query.FreeEnetheDnadQnamic_state
              attribute), 289
                                                                                                        (openff.evaluator.properties.ExcessMolarVolume
substance_query (openff.evaluator.storage.query.SimulationDataQuatryibute), 105
              attribute), 286
                                                                                         thermodynamic_state
SubstanceQuery
                                                  (class
                                                                                  in
                                                                                                        (openff.evaluator.properties.HostGuestBindingAffinity
              openff.evaluator.storage.query), 278
                                                                                                        attribute), 123
substances(openff.evaluator.datasets.PhysicalPropertyDathSeamodynamic_state
              property), 138
                                                                                                        (openff.evaluator.properties.SolvationFreeEnergy
substances(openff.evaluator.datasets.taproom.TaproomDataSet
                                                                                                        attribute), 119
              property), 148
                                                                                         thermodynamic_state
substances (openff.evaluator.datasets.thermoml.ThermoMLDataSet (openff.evaluator.protocols.analysis.AverageDielectricConstant
                                                                                                        attribute), 346
              property), 143
SubtractValues
                                                  (class
                                                                                         thermodynamic_state
              openff.evaluator.protocols.miscellaneous),
                                                                                                        (openff.evaluator.protocols.analysis.AverageFreeEnergies
                                                                                                        attribute), 348
swapcase() (openff.evaluator.storage.attributes.FilePath thermodynamic_state
              method), 295
                                                                                                        (openff.evaluator.protocols.analysis.AverageObservable
                                                                                                        attribute), 340
                                                                                         thermodynamic_state
                                                                                                        (openff. evaluator. protocols. analysis. Base Average Observable\\
TaproomDataSet
                                                  (class
                                                                                  in
                                                                                                        attribute), 331
              openff.evaluator.datasets.taproom), 145
                                                                                         thermodynamic_state
TaproomSource
                                                  (class
                                                                                   in
                                                                                                        (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies
               openff.evaluator.datasets.taproom), 149
                                                                                                        attribute), 481
target_reduced_potentials
              (\textit{openff.evaluator.protocols.reweighting.BaseMBAR \texttt{PASTMOdy} namic\_state) \\
                                                                                                        (openff.evaluator.protocols.openmm.OpenMMSimulation
              attribute), 554
                                                                                                        attribute), 475
target_reduced_potentials
              (openff.evaluator.protocols.reweighting.ReweightDheemodynanismstate
                                                                                                        (openff.evaluator.protocols.paprika.analysis.ComputeReferenceW
              attribute), 570
                                                                                                        attribute), 532
target_reduced_potentials
               (openff.evaluator.protocols.reweighting.ReweightObermodynamic_state
```

694 Index

(openff.evaluator.protocols.paprika.analysis.ComputeSymmetryC

attribute), 527			attribute), 346		
thermodynamic_state		time s	eries_statistics		
	rotocols.reweighting.BaseE			ols.analysis.Averag	eObservable
attribute), 549	0 0		attribute), 340	, 0	
thermodynamic_state		time_s	eries_statistics		
(openff.evaluator.p	rotocols.reweighting.Rewei	ightDielectri	c <b>Copsuffix</b> valuator.protoc	ols.analysis.BaseAv	verageObservable
attribute), 566			attribute), 331		
thermodynamic_state		time_s	eries_statistics		
(openff.evaluator.p	rotocols.simulation.BaseSi	mulation	(openff.evaluator.protoc	ols.analysis.BaseD	ecorrelateProtocol
attribute), 579			attribute), 358		
thermodynamic_state			eries_statistics		
	rotocols.storage.UnpackSto	oredSimulatio	=	ols.analysis.Decorr	elateObservables
attribute), 585			attribute), 372		
thermodynamic_state			eries_statistics		
attribute), 591	rotocols.yank.BaseYankPro	otocol	(openff.evaluator.protoc attribute), 367	ols.analysis.Decorr	elateTrajectory
thermodynamic_state		timest	ep (openff.evaluator.proto	cols.openmm.Open.	MMSimulation
	rotocols.yank.LigandRecep				
attribute), 603		timest	ep(openff.evaluator.proto	cols.simulation.Bas	eSimulation
thermodynamic_state			attribute), 579		
	rotocols.yank.SolvationYan	<i>kProti</i> ome <i>l</i> st		cols.yank.BaseYank	Protocol
attribute), 612			attribute), 591		
thermodynamic_state			ep (openff.evaluator.proto	cols.yank.LigandRe	ceptorYankProtocol
	torage.data.BaseSimulation		attribute), 603	1 101	W 10 . 1
attribute), 267		timest	ep (openff.evaluator.proto	cols.yank.Solvation	YankProtocol
thermodynamic_state	tougos data Stoudellus Euro	Db-#+1 o C	attribute), 612	naaa attuihutaa Eila	Dath
attribute), 276	torage.data.StoredFreeEner	rgyDatatie(	) (openff.evaluator.sto method), 295	rage.auribuies.rue	rain
thermodynamic_state		TIoonE	orceFieldSource	(class	in
-	torage.data.StoredSimulatio	-	openff.evaluator.forcefie	`	iri
attribute), 272	orage.aaia.sioreasimiiaii		ce_field() (openff.evalı		rnoffForceFieldSource
thermodynamic_state		00_101	method), 222	anor.joreejieia.siiii	nojji oreei tetasouree
-	torage.query.BaseSimulatio	nDatt <b>o</b> Ower		ator.workflow.Work	flow
attribute), 282		25	method), 299	, , , , , , , , , , , , , , , , , , ,	,
thermodynamic_state		to_num	ber_of_molecules()		
	torage.query.FreeEnergyDo	ataQuery	(openff.evaluator.substa 130	nces.Amount meth	od),
thermodynamic_state		to num	ber_of_molecules()		
-	torage.query.SimulationDa		(openff.evaluator.substa	nces FractAmount	
attribute), 286	orage.quer y.summanonDa	iagiery	method), 132	nees.Exacti intouni	
ThermodynamicState	(class	in to numi	ber_of_molecules()		
_	ermodynamics), 135		(openff.evaluator.substa	nces.MoleFraction	
thermoml_property()	(in modu	le	method), 133		
	utasets.thermoml), 145		das() (openff.evaluator.d	atasets.PhysicalPro	pertyDataSet
ThermoMLDataSet		in	method), 138	•	
openff.evaluator.da	tasets.thermoml), 140	to_pan	das() (openff.evaluator.d	atasets.taproom.Taj	proomDataSet
thermostat_friction			method), 148		
(openff.evaluator.p	rotocols.openmm.OpenMM	<i>Simtubut i</i> pan	das() (openff.evaluator.d	atasets.thermoml.Ti	hermoMLDataSet
attribute), 475			method), 143		
thermostat_friction			tocol()(openff.evaluato	r.workflow.schemas.	ProtocolGroupSchem
(openff.evaluator.p	rotocols.simulation.BaseSis		method), 319		
attribute), 579		to_pro	tocol()(openff.evaluato	r.workflow.schemas.	ProtocolSchema
time_series_statistics			method), 316	_	
(openff.evaluator.p	rotocols.analysis.AverageD	ielectroicsCtcon	<b>sæge_query()</b> (openff.ev	aluator.storage.data	a.BaseSimulationData

attribute), 353

```
method), 269
                                                                                                                                                                                        trajectory_paths(openff.evaluator.protocols.paprika.analysis.AnalyzeA
to_storage_query() (openff.evaluator.storage.data.BaseStoredDatattribute), 522
                                                                                                                                                                                       translate() (openff.evaluator.storage.attributes.FilePath
                              method), 261
to_storage_query() (openff.evaluator.storage.data.ForceFieldDatamethod), 296
                              method), 264
                                                                                                                                                                                        {\tt type} \ (open {\it ff.evaluator.protocols.groups.} Conditional {\it Group.Condition}
to_storage_query() (openff.evaluator.storage.data.HashableStoredItatbute), 422
                                                                                                                                                                                        type (openff.evaluator.workflow.schemas.ProtocolGroupSchema
                              method), 263
to_storage_query() (openff.evaluator.storage.data.ReplaceableDatatribute), 319
                              method), 266
                                                                                                                                                                                        type (openff.evaluator.workflow.schemas.ProtocolSchema
to_storage_query() (openff.evaluator.storage.data.StoredFreeEnergy Elanae), 316
                              method), 274
to_storage_query() (openff.evaluator.storage.data.Stored imulationData
                              method), 271
                                                                                                                                                                                        uncertainty (openff.evaluator.datasets.PhysicalProperty
tolerance (openff.evaluator.protocols.coordinates.BuildCoordinatesParkmak), 93
                              attribute), 375
                                                                                                                                                                                        uncertainty
                                                                                                                                                                                                                                                         (openff.evaluator.properties.Density
tolerance (openff.evaluator.protocols.coordinates.SolvateExistingStrugfweate), 102
                              attribute), 386
                                                                                                                                                                                       uncertainty (openff.evaluator.properties.DielectricConstant
tolerance (openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation), 109
                              attribute), 468
                                                                                                                                                                                       uncertainty (openff.evaluator.properties.EnthalpyOfMixing
tolerance (openff.evaluator.protocols.simulation.BaseEnergyMinimisatiphite), 112
                              attribute), 573
                                                                                                                                                                                       uncertainty (openff.evaluator.properties.EnthalpyOfVaporization
topology_file_name (openff.evaluator.storage.data.StoredFreeEnergyPiptde), 116
                               attribute), 274
                                                                                                                                                                                       uncertainty (open ff. evaluator. properties. Excess Molar Volume
attribute), 522
                                                                                                                                                                                       uncertainty (openff.evaluator.properties.HostGuestBindingAffinity
total_number_of_iterations
                                                                                                                                                                                                                       attribute), 123
                               (open \textit{ff.evaluator.protocols.openmm.} Open \textit{MMSimulation} \textbf{tainty} (open \textit{ff.evaluator.properties.Solvation} Free \textit{Energy}) and \textit{tainty} (open \textit{ff.evaluator.properties.properties.properties.Solvation} Free \textit{Energy}) and \textit{tainty} (open \textit{ff.evaluator.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.properties.pr
                              attribute), 475
                                                                                                                                                                                                                       attribute), 119
total_number_of_iterations
                                                                                                                                                                                       UNDEFINED (in module openff.evaluator.attributes), 620
                              (open \emph{ff.} evaluator. protocols. simulation. Base Simulation \textit{Pack} Stored Simulation \textit{Data}) and \textit{Constitution} and \textit{
                                                                                                                                                                                                                                                                                                                                                                  in
                              attribute), 578
                                                                                                                                                                                                                       openff.evaluator.protocols.storage), 584
total_number_of_molecules
                                                                                                                                                                                       unsuccessful_properties
                              (open \emph{ff}. evaluator. protocols. storage. Unpack Stored Simulation \textbf{Q} \textit{gta} \textit{ff}. evaluator. client. Request Result \textit{fig.} \textit{
                                                                                                                                                                                                                                                                                                                                                                at-
                              attribute), 585
                                                                                                                                                                                                                       tribute), 85
trajectory_file_name
                                                                                                                                                                                        unsuccessful_properties
                              (open ff. evaluator. storage. data. Stored Simulation Data
                                                                                                                                                                                                                       (openff.evaluator.server.Batch
                                                                                                                                                                                                                                                                                                                                          attribute),
                              attribute), 270
trajectory_file_path
                                                                                                                                                                                       update() (openff.evaluator.utils.observables.ObservableFrame
                               (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergieshod), 625
                              attribute), 481
                                                                                                                                                                                       update_forward_refs()
trajectory_file_path
                                                                                                                                                                                                                       (openff.evaluator.datasets.curation.components.conversion.Conve
                               (openff.evaluator.protocols.openmm.OpenMMSimulation
                                                                                                                                                                                                                       class method), 220
                              attribute), 476
                                                                                                                                                                                        update_forward_refs()
trajectory_file_path
                                                                                                                                                                                                                       (openff.evaluator.datasets.curation.components.CurationCompon
                              (openff.evaluator.protocols.reweighting.BaseEvaluateEnergietass method), 152
                              attribute), 549
                                                                                                                                                                                       update_forward_refs()
trajectory_file_path
                                                                                                                                                                                                                       (openff.evaluator.datasets.curation.components.filtering.FilterBy
                              (openff.evaluator.protocols.simulation.BaseSimulation
                                                                                                                                                                                                                       class method), 182
                              attribute), 580
                                                                                                                                                                                       update_forward_refs()
trajectory_file_path
                                                                                                                                                                                                                       (openff.evaluator.datasets.curation.components.filtering.FilterByl
                              (openff.evaluator.protocols.storage.UnpackStoredSimulationPlats method), 174
                              attribute), 586
                                                                                                                                                                                        update_forward_refs()
```

696 Index

 $\verb|trajectory_path| (open ff. evaluator. protocols. analysis. Compute Dip of Specific New Management New$ 

class method), 199

update_forward_refs()	<pre>update_forward_refs()</pre>
(openff.evaluator.datasets.curation.compo	onents.filtering.Filt <b>@ByMfinivMiqaidSdheasa</b> ts.curation.workflow.CurationWorkflowSc
class method), 184	class method), 156
update_forward_refs()	<pre>update_references()</pre>
(openff.evaluator.datasets.curation.compo	onents.filtering.Filte <b>oBeMfotelihuationnSorleflow</b> .schemas.ProtocolReplicator
class method), 169	method), 321
update_forward_refs()	upper() (openff.evaluator.storage.attributes.FilePath
	onents.filtering.Filt <b>erBhNC</b> 9011900nentsSchema
class method), 192	J
update_forward_refs()	V
	onents.filterivenEilterByRrpssygeSchwarbr.attributes.AttributeClass
class method), 166	
update_forward_refs()	method), 619
<del>-</del>	validate() (openff.evaluator.client.ConnectionOptions onents.filtering.FiltgrBnBrgpggtyTypesSchema
class method), 177	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
· · · · · · · · · · · · · · · · · · ·	validate() (openff.evaluator.client.Request method), 82
update_forward_refs()	validate() (openff.evaluator.client.RequestOptions
(openff.evaluator.datasets.curation.compo	,,,
class method), 171	validate() (openff.evaluator.client.RequestResult
update_forward_refs()	method), 85
(openff.evaluator.datasets.curation.compo	onents.filtexiveaEilecPxSpilfg.Schumgor.datasets.PhysicalProperty
class method), 187	method), 93
update_forward_refs()	${\tt validate()} \ (open \textit{ff.evaluator.datasets.PhysicalPropertyDataSet}$
(openff.evaluator.datasets.curation.compo	onents.filtering.FilterBnSynjrksSchema
class method), 190	validate() (openff.evaluator.datasets.taproom.TaproomDataSet
update_forward_refs()	method), 149
(openff.evaluator.datasets.curation.compo	onents.filtexivaaEilecBrSperegychemiatorStanessets.thermoml.ThermoMLDataSet
class method), 179	method), 144
update_forward_refs()	validate() (openff.evaluator.layers.CalculationLayerResult
(openff.evaluator.datasets.curation.compo	onents.filtering.FilterBySylystancesSchema
class method), 195	validate() (openff.evaluator.layers.CalculationLayerSchema
update_forward_refs()	method), 232
	onents.filterivaEilecBrIsenpreงเนเผืองในเงษาร.reweighting.ReweightingSchema
class method), 163	method), 242
update_forward_refs()	validate() (openff.evaluator.layers.simulation.SimulationSchema
(openff.evaluator.datasets.curation.compo	valiuate() (openy.evanaaor.iayers.simuianon.simuianonschema ments filtering FilterDunbicateoSchema
class method), 160	<i>"</i>
update_forward_refs()	validate() (openff.evaluator.layers.workflow.WorkflowCalculationScheme
	method), 235
olana mathad) 202	onents.fraexplodatearyFreeSolvSchafftevaluator.properties.Density
class method), 202	method), 102
update_forward_refs()	validate() (openff.evaluator.properties.DielectricConstant
(openff.evaluator.datasets.curation.compo	,,,
class method), 212	${\tt validate()}\ (open {\it ff.evaluator.properties.EnthalpyOfMixing}$
update_forward_refs()	method), 112
(openff.evaluator.datasets.curation.compo	onents.seleaTipaaSelest&upetyff:e&&Sultemproperties.EnthalpyOfVaporization
class method), 209	method), 116
update_forward_refs()	validate() (openff.evaluator.properties.ExcessMolarVolume
(openff.evaluator.datasets.curation.compo	onents.selection.Staffethod), 105
class method), 214	validate() (openff.evaluator.properties.HostGuestBindingAffinity
update_forward_refs()	method), 123
	onents.seleaTipaakarggi(Siptenff.evaluator.properties.SolvationFreeEnergy
class method), 216	method), 119
update_forward_refs()	validate() (openff.evaluator.protocols.analysis.AverageDielectricConstant
	variuate() (openjj.evatuator.protocois.anatysts.AverageDtetectricConstar onents.thermoml.ImportIliermoMLDataSchema
class method), 205	memou), 570

- validate() (openff.evaluator.protocols.analysis.AverageFrvaEiner() (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhas *method*), 348 method), 526
- $\verb|validate()| (open \textit{ff.evaluator.protocols.analysis.} A \textit{verageObsse} \textbf{ridble} \textbf{e}() (open \textit{ff.evaluator.protocols.paprika.analysis.} Compute \textit{Reference} \textbf{e}() \textbf{e}()$ method), 340 *method*), 536
- validate() (openff.evaluator.protocols.analysis.BaseAveragaDidatad) (openff.evaluator.protocols.paprika.analysis.ComputeSymmetri *method*), 334 *method*), 531
- validate() (openff.evaluator.protocols.analysis.BaseDecomælidate() (openff.evaluator.protocols.paprika.coordinates.AddDummyA method), 362 method), 497
- validate() (openff.evaluator.protocols.analysis.ComputeDiphitadatanentes.PreparePullC method), 357 method), 487
- validate() (openff.evaluator.protocols.analysis.DecorrelateaOhstavakip (openff.evaluator.protocols.paprika.coordinates.PrepareRelea method), 372 method), 492
- validate() (openff.evaluator.protocols.analysis.Decorrelated raiptator()) (openff.evaluator.protocols.paprika.restraints.ApplyRestraints method), 520 method), 367
- validate() (openff.evaluator.protocols.coordinates.BuildCravrilitattesPachpronff.evaluator.protocols.paprika.restraints.GenerateAttach. method), 379 method), 503
- validate() (openff.evaluator.protocols.coordinates.BuildDuckidGootGinapesnff.evaluator.protocols.paprika.restraints.GeneratePullRe method), 392 method), 509
- *method*), 386 method), 514
- validate() (openff.evaluator.protocols.forcefield.BaseBuild&\sitemate() (openff.evaluator.protocols.reweighting.BaseEvaluateEnergies method), 398 method), 552
- validate() (openff.evaluator.protocols.forcefield.BuildLig RadGeastys (entopenff.evaluator.protocols.reweighting.BaseMBARProtocol method), 409 method), 558
- validate() (openff.evaluator.protocols.forcefield.BuildSmirradffSharter() (openff.evaluator.protocols.reweighting.ConcatenateObserval *method*), 403 method), 547
- validate() (openff.evaluator.protocols.forcefield.BuildTLemp\( \) \( \) \( \) (openff.evaluator.protocols.reweighting. \( \) ConcatenateTrajector. method), 542 *method*), 415
- validate() (openff.evaluator.protocols.gradients.ZeroGradiathisdate() (openff.evaluator.protocols.reweighting.ReweightDielectricCo method), 420 method), 571
- validate() (openff.evaluator.protocols.groups.Conditional@highate() (openff.evaluator.protocols.reweighting.ReweightObservable method), 427 method), 564 validate() (openff.evaluator.protocols.groups.ConditionalGulzidate() (openff.evaluator.protocols.simulation.BaseEnergyMinimisati
- *method*), 423 method), 576 validate() (openff.evaluator.protocols.miscellaneous.AddValvicate() (openff.evaluator.protocols.simulation.BaseSimulation
- *method*), 432 method), 583 validate() (openff.evaluator.protocols.miscellaneous.DividaVablacte() (openff.evaluator.protocols.storage.UnpackStoredSimulationD
- method), 447 method), 589 validate() (openff.evaluator.protocols.miscellaneous.Dumnalfdatbe()) (openff.evaluator.protocols.yank.BaseYankProtocol
- method), 462 method), 592 validate() (openff.evaluator.protocols.miscellaneous.Filter Substance (By Replenff.evaluator.protocols.yank.Ligand Receptor Yank Protocol
- *method*), 454 *method*), 603 validate() (openff.evaluator.protocols.miscellaneous.MultiplyMatte() (openff.evaluator.protocols.yank.SolvationYankProtocol
- *method*), 442 method), 612
- validate() (openff.evaluator.protocols.miscellaneous.Subtradtibilidates() (openff.evaluator.server.Batch method), 90 validate() (openff.evaluator.storage.data.BaseSimulationData method), 437
- validate() (openff.evaluator.protocols.miscellaneous.WeightByMolafiethatibn 269 validate() (openff.evaluator.storage.data.BaseStoredData method), 452
- validate() (openff.evaluator.protocols.openmm.OpenMMEnergyMimintlxadi)qn261
- *method*), 468
  - validate() (openff.evaluator.storage.data.ForceFieldData
- validate() (openff.evaluator.protocols.openmm.OpenMMEvaluateEmentgied), 264
  - validate() (openff.evaluator.storage.data.HashableStoredData method), 481
- validate() (openff.evaluator.protocols.openmm.OpenMMSimulationmethod), 263 *method*), 476 validate() (openff.evaluator.storage.data.ReplaceableData

	bute), 123
validate() (openff.evaluator.storage.data.StoredFreeEneryaDuta (openff method), 276 attrib	veratuator.properties.sotvationFreeEnergy bute), 119
validate() (openff.evaluator.storage.data.StoredSimulationDhue (openff.	
	bute), 346
validate() (openff.evaluator.storage.query.BaseDataQueryalue(openff.	
	bute), 340
validate() (openff.evaluator.storage.query.BaseSimulationDhraeQupenff. method), 283 attrib	evaluator.protocois.analysis.BaseAverageObservable bute), 331
validate() (openff.evaluator.storage.query.ForceFieldQueralue (openff.	
	bute), 443
${\tt validate()} \ (open \textit{ff.evaluator.storage.query.} Free \textit{EnergyDataQuery} open \textit{ff.evaluator.storage.query}. \\$	* * *
	bute), 438
	bute), 448
	bute), 555
	evaluator.protocols.reweighting.Reweight Dielectric Constant bute), 571
validate() (openff.evaluator.substances.Component value(openff. method), 129 attrib	evaluator.protocols.reweighting.ReweightObservable bute), 564
validate() (openff.evaluator.substances.ExactAmount value (openf method), 132 130	ff.evaluator.substances.Amount attribute),
validate() (openff.evaluator.substances.MoleFraction value (open	off.evaluator.substances.ExactAmount at- te), 131
validate() (openff.evaluator.substances.Substance value (open	ff.evaluator.substances.MoleFraction at- te), 133
validate() (openff.evaluator.thermodynamics.Thermodynamalosed(topenff.	
	erty), 622
validate() (openff.evaluator.workflow.Protocol value (open	
	te), 302
<pre>validate() (openff.evaluator.workflow.ProtocolGroup value_a (open</pre>	iff.evaluator.protocols.miscellaneous.SubtractValues bute), 433
validate() (openff.evaluator.workflow.schemas.ProtocolGvalpschehopen	
	bute), 433
$\verb validate()  (open \textit{ff.evaluator.workflow.schemas.Protocol Scheduces})  (open ff.evaluator.workflow.go.)  (open \textit{ff.evaluator.workflow.go.)  (open \textit{ff.evaluator.workflow.go.)  (o$	
	bute), 348
validate() (openff.evaluator.workflow.schemas.Workflow.Schalmes (openfj.method), 323 attrib	f.evaluator.protocols.miscellaneous.AddValues bute), 428
validate() (openff.evaluator.workflow.WorkflowResult values() (openff.evaluetor.workflow.WorkflowResult valuetor.workflow.WorkflowResult valuetor.workflow.workflow.Workflow.WorkflowResult valuetor.workflow.workflow.workflow.workflow.workflow.workflow.workflow.workflow.workflow.workflow.workflow.worf	
The state of the s	od), 625
	nff.evaluator.protocols.yank.BaseYankProtocol
	bute), 592
	aff.evaluator.protocols.yank.LigandReceptorYankProtocol bute), 604
value (openff.evaluator.properties.DielectricConstant verbose(open	nff.evaluator.protocols.yank.SolvationYankProtocol
	bute), 612
- 1 10 11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	kmol (openff.evaluator.protocols.coordinates.BuildCoordinat bute), 375
value (openff.evaluator.properties.EnthalpyOfVaporizationverbose_pacl	
attribute), 116 attrib	bute), 386

attribute), 106

 ${\tt value} \ (open \textit{ff.evaluator.properties.HostGuestBindingAffinit} \textbf{yolumes} \ (open \textit{ff.evaluator.protocols.reweighting.ReweightDielectricConstants}) \ and \ an instants \ (open \textit{ff.evaluator.protocols.reweighting.ReweightDielectricConstants}) \ and \ an instants \ (open \textit{ff.evaluator.protocols.reweighting.ReweightDielectricConstants}) \ and \ an instants \ an instants \ and \ an instants \ an instants \ and \ an instants \ and \ an instants \ an instants \ and \ an instants \ an instants \ and \ an instants \ an instan$ 

attribute), 342

```
Ζ
         attribute), 566
                                                         ZeroGradients
                                                                                         (class
                                                                                                              in
W
                                                                   openff.evaluator.protocols.gradients), 415
wallclock_time_limit
                                                                      (openff.evaluator.storage.attributes.FilePath
                                                         zfill()
         (openff.evaluator.backends.QueueWorkerResources
                                                                  method), 296
         property), 247
water_model(openff.evaluator.protocols.forcefield.BuildLigParGenSystem
         attribute), 409
water\_model (openff.evaluator.protocols.forcefield.BuildTLeapSystem)
         attribute), 415
WeightByMoleFraction
                                    (class
                                                     in
         openff.evaluator.protocols.miscellaneous),
{\tt weighted\_value} \ (open \textit{ff.evaluator.protocols.miscellaneous.WeightByMoleFraction} \ )
         attribute), 449
window_index(openff.evaluator.protocols.paprika.restraints.ApplyRestraints
         attribute), 516
with_traceback() (openff.evaluator.utils.exceptions.EvaluatorException
         method), 87
with_traceback() (openff.evaluator.workflow.WorkflowException
         method), 301
Workflow (class in openff.evaluator.workflow), 297
workflow\_id(openff.evaluator.workflow.WorkflowResult
         attribute), 302
workflow_protocol()
                                               module
                                  (in
         openff.evaluator.workflow), 315
workflow_schema (openff.evaluator.layers.reweighting.ReweightingSchema
         attribute), 243
workflow_schema (openff.evaluator.layers.simulation.SimulationSchema
         attribute), 239
{\tt workflow\_schema} \ (open \textit{ff.evaluator.layers.workflow.WorkflowCalculationSchema} \ )
         attribute), 235
workflow_to_layer_result()
         (openff.evaluator.layers.reweighting.ReweightingLayer
         static method), 240
workflow_to_layer_result()
         (openff.evaluator.layers.simulation.SimulationLayer
         static method), 237
workflow_to_layer_result()
         (openff.evaluator.layers.workflow.WorkflowCalculationLayer
         static method), 234
WorkflowCalculationLayer
                                      (class
                                                     in
         openff.evaluator.layers.workflow), 234
WorkflowCalculationSchema
                                       (class
                                                     in
         openff.evaluator.layers.workflow), 235
WorkflowException, 300
WorkflowGraph (class in openff.evaluator.workflow),
WorkflowResult (class in openff.evaluator.workflow),
         302
WorkflowSchema
                                (class
                                                     in
         openff.evaluator.workflow.schemas), 322
```