# **OpenFF Evaluator Documentation**

openff-evaluator

# **GETTING STARTED**

1	Calcu	ulation Approaches	3
2	Supp	oorted 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	16
	2.7	Tutorial 03 - Analysing Data Sets	20
	2.8	Tutorial 04 - Optimizing Force Fields	23
	2.9	Property Data Sets	30
	2.10	ThermoML Archive	33
	2.11	Taproom	35
	2.12	Data Set Curation	36
	2.13	Physical Properties	42
	2.14	Common Workflows	46
	2.15	Gradients	48
	2.16	Calculation Layers	49
	2.17	Workflow Layers	52
	2.18	The Direct Simulation Layer	54
	2.19	The MBAR Reweighting Layer	54
	2.20	Workflows	55
	2.21	Replicators	57
	2.22	Workflow Graphs	61
	2.23	Protocols	62
	2.24	Protocol Groups	65
	2.25	Observables	66
	2.26	Calculation Backends	68
	2.27	Dask Backends	69
	2.28	Storage Backends	71
	2.29	Data Classes and Queries	72
	2.30	Local File Storage	74
	2.31	Building the Docs	75
	2.32	API	75
	2.33	Release History	572
	2.34	Release Process	584
Bil	oliogra	aphy	587

Index 589

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 Reweighting Layer.

# 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:

```
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:

```
{\tt 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:

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

```
[2]: 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:

```
[3]: 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:

```
[4]: 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:

```
[5]: len(data_set)
[5]: 275
```

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

```
[6]: len(data_set.substances)
[6]: 254
```

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

```
[7]: print(data_set.property_types)
   {'EnthalpyOfMixing', 'Density'}
```

#### 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:

```
[8]: 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)
{'Density'}
```

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

(continued from previous page)

```
FilterByTemperature,
   FilterByTemperatureSchema,
)
print(f"There were {len(data_set)} properties before filtering")
# 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")
There were 213 properties before filtering
There are now 9 properties after filtering
```

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

```
[10]: 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")

There are now 2 properties after filtering
```

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

```
[11]: pandas_data_set = data_set.to_pandas()
     pandas_data_set[
          ["Temperature (K)", "Pressure (kPa)", "Component 1", "Density Value (g / ml)",
      →"Source"]
     ].head()
[11]:
         Temperature (K) Pressure (kPa) Component 1 Density Value (g / ml)
                  298.15
                                 101.325
                                                                     0.78270
     0
                                              CC(C)0
                  298.15
                                 101.325
                                                 CCO
                                                                     0.78507
        10.1016/j.fluid.2013.10.034
                   10.1021/je1013476
```

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:

```
[12]: from openff.evaluator 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.evaluator.unit`` module to attach units to the temperatures and pressures we are filtering by. This module simply exposes a ``UnitRegistry`` from the fantasticpintlibrary. 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:

```
[13]: 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):

```
[14]: 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

```
[15]: 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,
```

(continues on next page)

(continued from previous page)

```
uncertainty=0.02*unit.kilojoule / unit.mole,
    source=source
)
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:

```
[16]: 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:

```
[17]: pandas_data_set = data_set.to_pandas()
      pandas_data_set[
          ["Temperature (K)",
           "Pressure (kPa)",
           "Component 1",
           "Density Value (g / ml)",
           "EnthalpyOfVaporization Value (kJ / mol)",
           "Source"
           1
      ].head()
         Temperature (K) Pressure (kPa) Component 1 Density Value (g / ml)
[17]:
                  298.15
                                  101.325
                                               CC(C)0
                                                                       0.78270
                  298.15
                                  101.325
                                                  CCO
                                                                       0.78507
      1
      2
                  298.15
                                  101.325
                                                  CCO
                                                                           NaN
      3
                  298.15
                                  101.325
                                               CC(C)0
                                                                           NaN
         EnthalpyOfVaporization Value (kJ / mol)
                                                                           Source
      0
                                              NaN
                                                     10.1016/j.fluid.2013.10.034
                                                                10.1021/je1013476
      1
                                              NaN
      2
                                            42.26
                                                   10.1016/S0021-9614(71)80108-8
      3
                                            45.34
                                                   10.1016/S0021-9614(71)80108-8
```

#### 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:

```
[18]: 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:

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

```
[2]: import warnings
warnings.filterwarnings('ignore')
import logging
logging.getLogger("openforcefield").setLevel(logging.ERROR)
```

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

```
[3]: 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:

```
[4]: 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, a copy is provided by the framework:

# 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:

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

[5]: Temperature (K) ... Source
0 298.15 ... 10.1016/j.fluid.2013.10.034
1 298.15 ... 10.1021/je1013476
2 298.15 ... 10.1016/S0021-9614(71)80108-8
3 298.15 ... 10.1016/S0021-9614(71)80108-8
[4 rows x 13 columns]
```

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:

```
[6]: 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:

```
[7]: 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:

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

# Create an options object which defines how the data set should be estimated.

(continues on next page)
```

(continued from previous page)

```
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:

```
[9]: from openff.evaluator.backends import ComputeResources
    from openff.evaluator.backends.dask import DaskLocalCluster

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:

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:

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.

```
[12]: # 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:

```
[13]: print(len(results.queued_properties))
    print(len(results.estimated_properties))
    print(len(results.unsuccessful_properties))
    print(len(results.exceptions))
    0
    4
    0
    0
    0
```

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:

```
[14]: 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:

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

→tutorials/colab_setup.ipynb

# %run colab_setup.ipynb
```

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

```
[2]: import warnings
warnings.filterwarnings('ignore')
import logging
logging.getLogger("openforcefield").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:

(continues on next page)

(continued from previous page)

```
# "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:

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

[4]: Temperature (K) ... Source
0 298.15 ... 10.1016/j.fluid.2013.10.034
1 298.15 ... 10.1021/je1013476
2 298.15 ... 10.1016/S0021-9614(71)80108-8
3 298.15 ... 10.1016/S0021-9614(71)80108-8

[4 rows x 13 columns]
```

# 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:

```
[6]: properties_by_type = {
    "Density": [],
    "EnthalpyOfVaporization": []
}

for experimental_property in experimental_data_set:

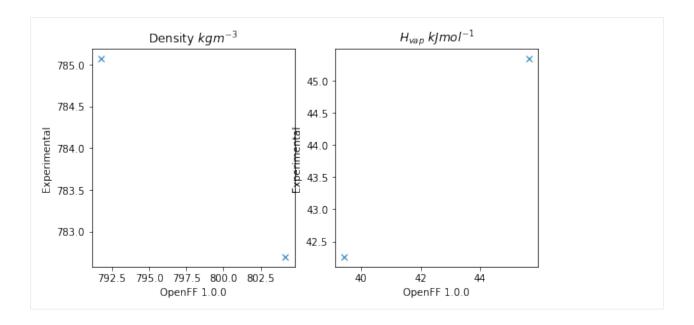
# Find the estimated property which has the same id as the
# experimental property.
estimated_property = next(
    x for x in estimated_data_set if x.id == experimental_property.id
)

# Add this pair of properties to the list of pairs
property_type = experimental_property.__class__.__name__
properties_by_type[property_type].append((experimental_property, estimated_property))
```

#### 2.7.3 Plotting the Results

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

```
[7]: from matplotlib import pyplot
    # Create the figure we will plot to.
    figure, axes = pyplot.subplots(nrows=1, ncols=2, figsize=(8.0, 4.0))
    # 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.evaluator 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:

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

```
[2]: import warnings
warnings.filterwarnings('ignore')
import logging
logging.getLogger("openforcefield").setLevel(logging.ERROR)
```

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

```
[3]: 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:

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

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

```
[5]: !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:

```
[6]: # 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:
```

(continues on next page)

(continued from previous page)

```
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

```
[7]: data_set.filter_by_smiles("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:

```
[8]: # 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:

```
[9]: from openforcefield.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:

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

all_smiles = set(
    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:

```
[11]: # 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:

```
[12]: 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")

[12]: '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

$target
name pure_data
type Evaluator_SMIRNOFF
weight 1.0
openff.evaluator_input options.json
$end
```

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:

```
[14]: 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):

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

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  $(denominator_n)$ :

```
[15]: from openff.evaluator 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:

```
[16]: 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"]

# Reduce the default number of molecules
from 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:

```
[17]: # 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:

```
[18]: # Launch the calculation backend which will distribute any calculations.
     from openff.evaluator.backends import ComputeResources
     from openff.evaluator.backends.dask import DaskLocalCluster
     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)
     01:30:20.505 INFO
                            Server listening at port 8000
```

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:

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

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

```
[20]: !ls result/optimize openff-1.0.0-tagged_1.offxml openff-1.0.0-tagged.offxml
```

### 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:

```
[21]: from forcebalance.nifty import lp_load

# Determine how many iterations ForceBalance has completed.
from glob import glob
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)

[0.9270359101845124, 0.011497456194198362]
```

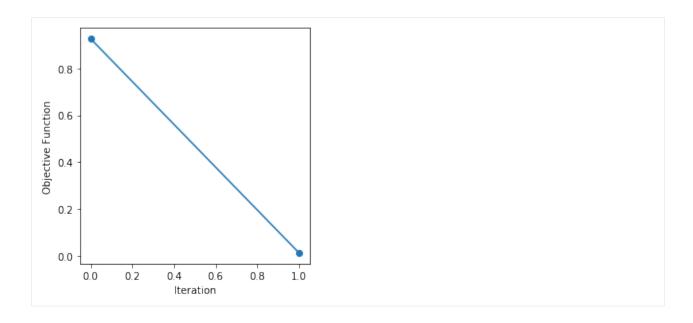
The objective function is then easily plotted:

```
[22]: 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 register\_thermoml\_property() function. For example, the 'Specific volume, m3/kg' property (which is simply the reciprocal of mass density) may be mapped onto the Density object by providing a conversion\_function:

```
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:

```
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.

2.11. Taproom 35

## 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:

```
curation_schema = WorkflowSchema(
   component_schemas=[
        # Import the ThermoML archive.
        thermoml.ImportThermoMLDataSchema()
        # 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:

```
from openff.evaluator.datasets.curation.components.thermoml import (
    ImportThermoMLData,
    ImportThermoMLDataSchema,
)

# Import all data collected from the IJT journal.
data_frame = ImportThermoMLData.apply(pandas.DataFrame(),
    ImportThermoMLDataSchema())
```

### **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.:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByCharged,
   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:

```
from openff.evaluator.datasets.curation.components.filtering import (
   FilterByEnvironments,
   FilterByEnvironmentsSchema,
)
```

#### **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:

#### **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  $\Delta H_{mix}(x_0, \dots, x_{M-1})$  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.

## 2.14 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:

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

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:

)

#### 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 <code>CalculationLayerResult</code> 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:

```
# Register the default schema to use for density measurements being estimated
# by the direct simulation calculation layer.
register_calculation_schema(
   property_class=Density,
   layer_class=SimulationLayer,
```

```
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 :doc:` storage/dataclasses <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 55

```
# 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 57

```
# 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", "C0")}
```

```
# 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 59

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}_{n_mols_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:

# 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 63

#### **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 *can\_merge()* and *merge()* 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=MergeBehaviour.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:

```
timestep = InputAttribute(
   docstring="The timestep to evolve the system by at each step.",
   type_hint=unit.Quantity,
   merge_behavior=InequalityMergeBehaviour.SmallestValue,
```

```
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=InequalityMergeBehaviour.LargestValue,
    default_value=1,
)
```

This can be accomplished using the InequalityMergeBehaviour 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 67

## **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. submit\_task

## 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 *submit\_task()* must run asynchronously and return an asyncio Future 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:

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> and <code>DaskPBSBackend</code> backends wrap around the dask LSFCluster and PBSCluster 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:

# Create the object which describes the compute resources each worker should request from # the queueing system.

(continues on next page)

2.27. Dask Backends 69

(continued from previous page)

```
worker_resources = QueueWorkerResources(
   number_of_threads=1,
   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  $\sim$ /.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.

# 2.28 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()

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):
        ...
```

(continues on next page)

(continued from previous page)

```
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 *BaseStoredData* 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 = SubstanceQuery()
pure_substance_query.components_only = True

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

(continues on next page)

(continued from previous page)

- 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/docs
conda env create --name openff-evaluator-docs --file environment.yaml
conda activate openff-evaluator-docs
rm -rf api && make clean && make html
```

The above will yield a new directory named \_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.
	continues on next page

### Table 2 – continued from previous page

	· · · · · · · · · · · · · · · · · · ·
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.evaluator import unit
>>> from openff.evaluator.properties import Density
>>>
>>> data_set.filter_by_property_types(Density)
>>> data_set.filter_by_temperature(
       min_temperature=130*unit.kelvin,
>>>
>>>
       max_temperature=260*unit.kelvin
>>> )
>>>
>>> # Load in the force field parameters
>>> from openff.evaluator.forcefield import SmirnoffForceFieldSource
>>> force_field_source = SmirnoffForceFieldSource.from_path('smirnoff99Frosst-1.1.0.
→offxml')
>>> # Submit the estimation request to a running server.
>>> request = property_estimator.request_estimate(data_set, force_field_source)
```

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:

```
>>> from openff.evaluator.layers.reweighting import ReweightingLayer
>>> from openff.evaluator.layers.simulation import SimulationLayer
>>>
>>> options = RequestOptions(calculation_layers=[
        "ReweightingLayer",
>>>
        "SimulationLayer"
>>>
>>> 1)
>>>
>>> request = property_estimator.request_estimate(data_set, force_field_source,_
→options)
```

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.

```
>>> from openff.evaluator.properties import DielectricConstant
>>>
>>> # Generate a schema to use when estimating densities directly
>>> # from simulations.
>>> density_simulation_schema = Density.default_simulation_schema()
>>> # Generate a schema to use when estimating dielectric constants
>>> # from cached simulation data.
>>> dielectric_reweighting_schema = DielectricConstant.default_reweighting_schema()
>>>
>>> options.workflow_options = {
        'Density': {'SimulationLayer': density_simulation_schema},
>>>
>>>
        'Dielectric': {'SimulationLayer': dielectric_reweighting_schema}
>>> }
>>>
>>> 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')
>>>
```

2.32. API 77

(continues on next page)

(continued from previous page)

```
>>> 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.
<pre>retrieve_results(request_id[, synchronous,])</pre>	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 openff.toolkit.typing. engines.smirnoff.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**

# **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])	param server_address 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 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**

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 st

### 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])	param client 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> .
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.
results([synchronous, polling_interval])	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.
<pre>validate([attribute_type])</pre>	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()	
<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**

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 [].

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**

enable\_data\_caching=True, delete\_working\_files=True)

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
>>> # 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 EvaluatorClients.
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.

#### **Methods**

from_json(file_path)  Create this object from a JSON file.	1n1t()	
	from_json(file_path)	Create this object from a JSON file.
get_attributes([attribute_type]) Returns all attributes of a specific attribute_type.	<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.	json([file_path, format])	Creates a JSON representation of this class.
parse_json(string_contents) Parses a typed json string into the corresponding cl	<pre>parse_json(string_contents)</pre>	Parses a typed json string into the corresponding class
structure.		structure.
validate([attribute_type])Validate the values of the attributes.	<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

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 estimate his
	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 [].

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.
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.

### 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.

 $\textbf{Parameters} \ \ \textbf{enum\_string} \ (\textit{str}) - \text{The str representation of a } \textit{PropertyPhase}$ 

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**

				1
7	n	7	+	(
 _	11	_	L.	 ١.

<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

#### **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**

init([fidelity, provenance])	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.

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

**Returns** The parsed class.

Return type Any

# **Built-in Properties**

A class representation of a density property
A class representation of an excess molar volume prop-
erty
A class representation of a dielectric property
A class representation of an enthalpy of mixing property
A class representation of an enthalpy of vaporization
property
A class representation of a solvation free energy prop-
erty.
A class representation of a host-guest binding affinity
property

# **Density**

class openff.evaluator.properties.Density(thermodynamic\_state=None,

phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

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.
<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.
validate([attribute_type])	Validate the values of the attributes.

### **Attributes**

gradients	The gradients of this property with respect to differ-
gradients	
	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.

 $\textbf{static default\_simulation\_schema} (absolute\_tolerance = < open \textit{ff.evaluator.attributes.} Unde \textit{finedAttributes} object >, rela-$ 

 $tive\_tolerance = < openff.evaluator.attributes.attributes.Undefined Attribute object>, n\_molecules = 1000) \rightarrow$ 

openff.evaluator.layers.simulation.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=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, rela-

 $tive\_tolerance = < openff.evaluator.attributes. Undefined Attribute \\ object>, n\_effective\_samples = 50) \rightarrow$ 

openff.evaluator.layers.reweighting.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**

- 1 1 (F.1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	O
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 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.

classmethod default\_reweighting\_schema (absolute\_tolerance: openff.evaluator.utils.units.Quantity = < openff.evaluator.attributes.attributes.UndefinedAttribute object>, relative\_tolerance: float = < openff.evaluator.attributes.attributes.UndefinedAttribute object>, n\_effective\_samples: int = 50)  $\rightarrow$  openff.evaluator.layers.reweighting.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.

# Returns

**Return type** The default re-weighting calculation schema.

classmethod default\_simulation\_schema(absolute\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, relative\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>,  $n_m$ olecules=1000)  $\rightarrow$ 

openff.evaluator.layers.simulation.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**

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.
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.
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.

 $\textbf{static default\_simulation\_schema} (absolute\_tolerance = < openff. evaluator. attributes. Undefined Attributes object >, rela-$ 

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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

**static default\_reweighting\_schema**(absolute\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, rela-

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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**

class openff.evaluator.properties.EnthalpyOfMixing(thermodynamic\_state=None,

phase=PropertyPhase.Undefined, substance=None, value=None, uncertainty=None, source=None)

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**

init([thermodynamic_state, phase,])	Constructs a new PhysicalProperty object.
default_reweighting_schema([])	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 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.	

## classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

classmethod default\_reweighting\_schema (absolute\_tolerance: openff.evaluator.utils.units.Quantity = < openff.evaluator.attributes.attributes.UndefinedAttribute object>, relative\_tolerance: float = < openff.evaluator.attributes.attributes.UndefinedAttribute object>, n\_effective\_samples: int = 50)  $\rightarrow$  openff.evaluator.layers.reweighting.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.

# Returns

**Return type** The default re-weighting calculation schema.

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, n\_molecules=1000)  $\rightarrow$ 

 $open {\it ff. evaluator. layers. simulation. Simulation Schema}$ 

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 Physical Property 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.
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.
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.
	1 1 7

#### classmethod default\_unit()

openff.evaluator.unit.Unit: The default unit (e.g. g / mol) associated with this class of property.

 $\textbf{static default\_simulation\_schema} (absolute\_tolerance = < openff. evaluator. attributes. Undefined Attributes object >, rela-$ 

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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

 $\textbf{classmethod default\_reweighting\_schema} (absolute\_tolerance = < open \textit{ff.evaluator.attributes.} Unde \textit{finedAttributes.} Object > , rela-$ 

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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

# 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.
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 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.

continues on next page

Table 39 – Continued from Drevious Dat	<ul><li>– continued from previous</li></ul>	s page
--	---	--------

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.

 $\textbf{static default\_simulation\_schema} (absolute\_tolerance = < openff. evaluator. attributes. Undefined Attributes object >, rela-$ 

tive\_tolerance=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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**

Constructs a new PhysicalProperty object.
Returns the default calculation schema to use when
estimating a host-guest binding affinity measurement
with an APR calculation using the paprika package.
openff.evaluator.unit.Unit: The default unit (e.g.
Returns the default calculation schema to use when
estimating this class of property from direct simula-
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.
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.

## 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:

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: openff.evaluator.utils.units.Quantity = <Quantity(1.0, 'femtosecond')>, dt\_equilibration: openff.evaluator.utils.units.Quantity = <Quantity(2.0, 'femtosecond')>, dt\_production: openff.evaluator.utils.units.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)) #__
\[ \sum_{Infinite} \] 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 Substance object from a list of compo-
	nents.
<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 attribute_type.
<pre>get_molecules_per_component(maximum_molecules)</pre>	cu <b>Rs</b> ) 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.
validate([attribute_type])	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 substance.
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

Defines a single component in a chemical system, as well as it's role within the system (if any).

\_\_init\_\_(smiles=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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 (Component . 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.
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**

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 Component.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**

A representation of the amount of a given component in a Substance.

\_\_init\_\_(value=<openff.evaluator.attributes.attributes.UndefinedAttribute object>)

**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)	lest)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.. This attribute is *read-only*.

**Type** typing.Union[float, int]

# 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**

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=<openff.evaluator.attributes.UndefinedAttribute object>)

**Parameters value** (*float or int*) – The value of this amount.

# **Methods**

init([value])	
	<b>param value</b> The value of this amount.
	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	lest)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 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 total 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

The mole fraction of a Component in a Substance.

\_\_init\_\_(value=<openff.evaluator.attributes.attributes.UndefinedAttribute object>)

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 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.
to_number_of_molecules(total_substance_molec	ule©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 float

### 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

### 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.
```

Note that the pressure is only relevant for periodic systems.

```
__init__(temperature=None, pressure=None)
Constructs a new ThermodynamicState object.
```

#### **Parameters**

- temperature (openff.evaluator.unit.Quantity) The 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 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**

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.
	continues on next page

Table 57 – continued from previous page

<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.

#### Returns

**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.

#### **Returns**

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 pandas.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: pandas.core.frame.DataFrame) → openff.evaluator.datasets.datasets.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.

**Returns** 

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.

## **Methods**

init()	Constructs a new ThermoMLDataSet object.	
<pre>add_properties(*physical_properties[, validate])</pre>	e]) Adds a physical property to the data set.	
from_doi(*doi_list)  Load a ThermoML data set from a list of DO		
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.	
from_pandas(data_frame)	Constructs a data set object from a pandas	
	DataFrame object.	
from_url(*url_list)	Load a ThermoML data set from a list of URLs	
from_xml(xml, default_source)	Load a ThermoML data set from an xml 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.	
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: pandas.core.frame.DataFrame*) →

open ff. evaluator. datas ets. datas ets. Physical Property Data Set

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.

## Returns

**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.

Returns

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.

**Returns** 

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 pandas.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[openff.evaluator.utils.units.Quantity] = < Quantity(150, 'millimolar')>, negative\_buffer\_ion: <math>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[openff.evaluator.utils.units.Quantity] = <Quantity(150, '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.

# Methods

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.
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.
<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.
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

 $\begin{tabular}{ll} {\bf classmethod\ from\_pandas}({\it data\_frame}: {\it pandas.core.frame.DataFrame}) \to \\ {\it openff.evaluator.datasets.datasets.PhysicalPropertyDataSet} \end{tabular}$ 

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.

Returns

**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 ison 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.

Returns

**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.

#### Returns

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 pandas.DataFrame

## validate()

Checks to ensure that all properties within the set are valid physical property object.

# **TaproomSource**

```
 \textbf{class} \  \, \text{openff.evaluator.datasets.taproom.} \\ \textbf{TaproomSource}(\textit{doi="}, \textit{comment="}, \textit{technique="}, \textit{host\_identifier="}, \textit{guest\_identifier="}) \\
```

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.
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.

# 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 conversion) to a data set
CurationComponentSchema	sion) to a data set.  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: openff.evaluator.datasets.PhysicalPropertyDataSet, schema:

openff.evaluator.datasets.curation.components.curationComponentSchema,

 $n\_processes$ :  $int = 1) \rightarrow openff.evaluator.datasets.datasets.PhysicalPropertyDataSet$ 

**classmethod apply**(*data\_set: pandas.core.frame.DataFrame, schema:* 

openff.evaluator.datasets.curation.components.curationComponentSchema,

 $n\_processes: int = 1) \rightarrow pandas.core.frame.DataFrame$ 

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.

#### **Returns**

Return type The data set which has had the component applied to it.

# **CurationComponentSchema**

 $\textbf{class} \ \, \textbf{openff.evaluator.datasets.curation.components.} \\ \textbf{CurationComponentSchema(*} \textit{args: Any,} \\ **kwargs: Any)$ 

A base class for schemas which specify how particular curation components should be applied to a data set.

```
__init__(*args: Any, **kwargs: Any) \rightarrow None
```

## **Methods**

init(*args, **kwargs)			

CurationWorkflow	A convenience class for applying a set of curation components sequentially to a data set.
CurationWorkflowSchema	A schemas which encodes how a set of curation components 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.

## **Methods**

init()	
apply()	Apply each component of this curation workflow to an initial data set in sequence.

**classmethod apply**(data set: openff.evaluator.datasets.datasets.PhysicalPropertyDataSet, schema:

openff.evaluator.datasets.curation.workflow.CurationWorkflowSchema, n\_processes:

 $int = 1) \rightarrow openff.evaluator.datasets.datasets.PhysicalPropertyDataSet$ 

**classmethod apply**(data\_set: pandas.core.frame.DataFrame, schema:

openff.evaluator.datasets.curation.workflow.CurationWorkflowSchema,  $n\_processes$ :  $int = 1) \rightarrow pandas.core.frame.DataFrame$ 

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.

# Returns

**Return type** The data set which has had the curation workflow applied to it.

# CurationWorkflowSchema

A schemas which encodes how a set of curation components should be applied sequentially to a data set.

**\_\_init\_\_**(\*args: Any, \*\*kwargs: Any) 
$$\rightarrow$$
 None

## **Methods**

init(*args, **kwargs)		

# **Attributes**

component\_schemas

# Filtering

FilterDuplicatesSchema	
FilterDuplicates	A component to remove duplicate data points (within a specified precision) from a data set.
FilterByTemperatureSchema	
FilterByTemperature	A component which will filter out data points which were measured outside of a specified temperature range
FilterByPressureSchema	
FilterByPressure	A component which will filter out data points which were measured outside of a specified pressure range.
FilterByMoleFractionSchema	
FilterByMoleFraction	A component which will filter out data points which were measured outside of a specified mole fraction range.
FilterByRacemicSchema	
FilterByRacemic	A component which will filter out data points which were measured for racemic mixtures.
FilterByElementsSchema	
FilterByElements	A component which will filter out data points which were measured for systems which contain specific elements.
	continues on next page

Table	70		f		
rabie	73 –	continued	mom	previous	page

Table 75 – C	continued from previous page
FilterByPropertyTypesSchema	
FilterByPropertyTypes	A component which will apply a filter which only retains properties of specified types.
FilterByStereochemistrySchema	
FilterByStereochemistry	A component which filters out data points measured for systems whereby the stereochemistry of a number of components is undefined.
FilterByChargedSchema	
FilterByCharged	A component which filters out data points measured for substances where any of the constituent components have a net non-zero charge.
FilterByIonicLiquidSchema	
FilterByIonicLiquid	A component which filters out data points measured for substances which contain or are classed as an ionic liquids.
FilterBySmilesSchema	
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.
FilterBySmirksSchema	, , , , , , , , , , , , , , , , , , ,
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.
FilterByNComponentsSchema	•
FilterByNComponents	A component which filters out data points measured for systems with specified number of components.
FilterBySubstancesSchema	
FilterBySubstances	A component which filters the data set so that it only contains properties measured for particular substances.
FilterByEnvironmentsSchema	
FilterByEnvironments	A component which filters a data set so that it only contains measurements made for substances which contain specific chemical environments.

# **FilterDuplicatesSchema**

# **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.

 ${\tt classmethod\ apply}({\it data\_set}, {\it schema}, {\it n\_processes}{=}{\it l})$ 

Apply this curation component to a data set.

- **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.

#### Returns

Return type The data set which has had the component applied to it.

# **FilterByTemperatureSchema**

 $\textbf{class} \ \, \text{openff.evaluator.datasets.curation.components.filtering.} \\ \textbf{FilterByTemperatureSchema} (*args: Any, **kwargs: Any)$ 

```
__init__(*args: Any, **kwargs: Any) \rightarrow None
```

## **Methods**

```
__init__(*args, **kwargs)
```

# **Attributes**

maximum\_temperature
minimum\_temperature
type

# **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.

**Returns** 

**Return type** The data set which has had the component applied to it.

# **FilterByPressureSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterByPressureSchema(\*args:

Any,

\*\*kwargs:

Any)

\_\_init\_\_(\*args: Any, \*\*kwargs: Any)  $\rightarrow$  None

Methods

```
__init__(*args, **kwargs)
```

#### **Attributes**

maximum\_pressure
minimum\_pressure
type

# **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.

- **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.

#### Returns

Return type The data set which has had the component applied to it.

# **FilterByMoleFractionSchema**

# **FilterByMoleFraction**

type

mole\_fraction\_ranges

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.

#### Returns

Return type The data set which has had the component applied to it.

# **FilterByRacemicSchema**

# **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.

- 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.

#### Returns

Return type The data set which has had the component applied to it.

# **FilterByElementsSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterByElementsSchema(\*args: Any, \*\*kwargs: Any)

\_\_init\_\_(\*args: Any, \*\*kwargs: Any)  $\rightarrow$  None

```
Methods
```

```
__init__(*args, **kwargs)
```

## **Attributes**

```
allowed_elements
forbidden_elements
type
```

# **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.

Returns

Return type The data set which has had the component applied to it.

# **FilterByPropertyTypesSchema**

class openff.evaluator.datasets.curation.components.filtering.FilterByPropertyTypesSchema(\*args: Any, \*\*kwargs: Any) **\_\_init\_\_**(\*args: Any, \*\*kwargs: Any)  $\rightarrow$  None **Methods** \_\_init\_\_(\*args, \*\*kwargs) **Attributes** n\_components property\_types strict type **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.

- 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.

#### Returns

**Return type** The data set which has had the component applied to it.

# **FilterByStereochemistrySchema**

# **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.

#### **Returns**

**Return type** The data set which has had the component applied to it.

# **FilterByChargedSchema**

# **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.

- **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.

#### Returns

Return type The data set which has had the component applied to it.

# FilterBylonicLiquidSchema

# 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.

Returns

**Return type** The data set which has had the component applied to it.

# **FilterBySmilesSchema**

# **FilterBySmiles**

 ${\bf class}\ open ff. evaluator. datasets. curation. components. filtering. {\bf Filter By Smiles}$ 

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.

- **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.

#### Returns

Return type The data set which has had the component applied to it.

# **FilterBySmirksSchema**

# **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__()
```

et		

init()	
V	
<pre>apply(data_set, schema[, n_processes])</pre>	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.

#### Returns

Return type The data set which has had the component applied to it.

# **FilterByNComponentsSchema**

 $\textbf{class} \ \, \texttt{openff.evaluator.datasets.curation.components.filtering.} \\ \textbf{FilterByNComponentsSchema} (*\textit{args:} \texttt{openff.evaluator.datasets.curation.components.filtering.}) \\ \textbf{filterByNComponents.curation.curati$ 

Any, \*\*kwargs: Any)

**\_\_init\_\_**(\*args: Any, \*\*kwargs: Any) 
$$\rightarrow$$
 None

#### Methods

#### **Attributes**

n\_components

type

# **FilterByNComponents**

 ${\bf class}\ open ff. evaluator. datasets. curation. components. filtering. {\bf FilterByNComponents}$ 

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.

## Returns

Return type The data set which has had the component applied to it.

# **FilterBySubstancesSchema**

 $\textbf{class} \ \ \textbf{open} \textbf{ff.evaluator.datasets.curation.components.filtering.} \textbf{FilterBySubstancesSchema} (*\textit{args:} \textbf{args:} \textbf{open} \textbf{filterBySubstancesSchema}) (*\textit{args:} \textbf{open} \textbf{filterBySubstancesSchema}) (*\textit{args:}$ 

Any,
\*\*kwargs:
Any)

**\_\_init\_\_**(\*args: Any, \*\*kwargs: Any)  $\rightarrow$  None

#### **Methods**

```
__init__(*args, **kwargs)
```

# **Attributes**

```
substances_to_exclude
substances_to_include
type
```

# **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')]
>>> )
```

```
__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.

- 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.

#### Returns

**Return type** The data set which has had the component applied to it.

# **FilterByEnvironmentsSchema**

s openff.evaluator.datasets.curation.components.filtering.Fi	_
	Any,
	**kwai Any)
	Any)
$\_$ <b>init</b> $\_$ (*args: Any, **kwargs: Any) $\rightarrow$ None	
Methods	
init(*args, **kwargs)	
Attributes	
Attributes	
at_least_one_environment	
environments	
per_component_environments	
strictly_specified_environments	

# **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.

#### Returns

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**

\*\*kwargs:
Any)

**\_\_init\_\_**(\*args: Any, \*\*kwargs: Any)  $\rightarrow$  None

## **Methods**

\_\_init\_\_(\*args, \*\*kwargs)

## **Attributes**

t	y	p	$\epsilon$

# **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.

# Returns

**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

\*\*kwargs:

Any)

**\_\_init\_\_(\***args: Any, \*\*kwargs: Any)  $\rightarrow$  None

M	ethods
i	init(*args, **kwargs)
At	ttributes
cac	he_file_name
ret	ain_uncertainties
roo	t_archive_url
typ	pe
Import <sup>1</sup>	ThermoMLData
A	openff.evaluator.datasets.curation.components.thermoml.ImportThermoMLData component which will import all supported data from the NIST ThermoML archive for (optionally) specified urnals.
	init()
M	ethods
i	init()
app	Apply this curation component to a data set.
cl	Lassmethod 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.
	<ul> <li>n_processes – The number of processes that this component is allowed to parallelize across.</li> </ul>
	Returns
	Return type The data set which has had the component applied to it.
Data Po	int Selection
Selec	tSubstancesSchema
	continues on next nage

measured as close as possible to a particular set of states.

Table 127 – continued from previous page		
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	

State

SelectDataPoints

TargetState

FingerPrintType	An enumeration.

# SelectSubstancesSchema

 $\textbf{class} \ \ \textbf{openff.evaluator.datasets.curation.components.selection.} \textbf{SelectSubstancesSchema} (*args: args: args:$ 

Any, \*\*kwargs: Any)

**\_\_init\_\_(\***
$$args: Any, **kwargs: Any) \rightarrow None$$

# **Methods**

init(*args, **kwargs)	

# **Attributes**

finger_print_type
n_per_environment
per_property
substances_to_exclude
target_environments
type

## **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.

#### Returns

**Return type** The data set which has had the component applied to it.

## SelectDataPointsSchema

class openff.evaluator.datasets.curation.components.selection.SelectDataPointsSchema(\*args:

Any,
\*\*kwargs:
Any)

```
__init__(*args: Any, **kwargs: Any) \rightarrow None
```

# **Methods**

\_\_init\_\_(\*args, \*\*kwargs)

### **Attributes**

target_states		
3 -		
type		

# **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.

#### **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.

### Returns

**Return type** The data set which has had the component applied to it.

# State

```
__init__(*args: Any, **kwargs: Any) \rightarrow None
```



# **FingerPrintType**

<pre>class openff.evaluator.datasets.curation</pre>	n.components.selection. <b>FingerPrintType</b> (value)
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	

 $\textbf{class} \ \ \textbf{open} \textbf{ff.evaluator.datasets.curation.components.conversion.} \textbf{ConvertExcessDensityDataSchema} (*\textit{args:} \textbf{class}) \\$ 

Any,\*\*kwargs: Any)

```
__init__(*args: Any, **kwargs: Any) \rightarrow None
```

# **Methods**

\_\_init\_\_(\*args, \*\*kwargs)

# **Attributes**

pressure\_precision

temperature\_precision

type

# 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()	
<pre>apply(data_set, schema[, n_processes])</pre>	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.

#### Returns

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 Na-
	tive Open Force Field (SMIRNOFF) specification.
TLeapForceFieldSource	A wrapper around Amber force fields which may be ap-
	plied via the <i>tleap</i> software package.
LigParGenForceFieldSource	A wrapper and the OPLSAAM force field which can be
	applied via the LigParGen server.
	applied via tile Ligrarden 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.

\_\_init\_\_()

### **Methods**

init()	
<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

### **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
from_json(file_path)	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 openff.toolkit.typing.engines.smirnoff.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**(openff.toolkit.typing.engines.smirnoff.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=<Quantity(9.0, '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=<Quantity(9.0, '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.

init([preferred_charge_model, cutoff,])	Constructs a new LigParGenForceFieldSource 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.
<pre>parse_json(string_contents)</pre>	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.

# 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.

\_\_init\_\_()

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** (*boo1*) 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.
	continues on next page

Table 157 - continued from previous page

	· · · · · · · · · · · · · · · · · · ·
<pre>validate([attribute_type])</pre>	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**

init()	
	G III II G YGOY 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.
<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**

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 * measured_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

 ${\tt openff.evaluator.layers.register\_calculation\_schema} ({\it property\_class, layer\_class, schema})$ 

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 CalculationLayer should use when es-
	timating 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.

\_\_init\_\_()

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.
<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 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 <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

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.

\_\_init\_\_()

init()	
from_json(file_path)	Create this object from a JSON file.
get_attributes([attribute_type])	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**

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 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.
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\_\_()

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.
workflow_to_layer_result(queued_properties,	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

**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.

### **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**

absolute_tolerance	The absolute uncertainty that the property should be
	estimated to within.
maximum_data_points	The maximum number of data points to include as
	part of the multi-state reweighting calculations.
relative_tolerance	The relative uncertainty that the property should be
	estimated to within, i.e relative_tolerance * mea-
	sured_property.uncertainty.
storage_queries	The queries to perform when retrieving data for each
	of the components in the system from the storage
	backend.
temperature_cutoff	The maximum difference between the target temper-
	ature and the temperature at which cached data was
	collected to.
workflow_schema	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 K.

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**

 ${\bf class} \ {\bf openff.evaluator.backends.CalculationBackend} ({\it number\_of\_workers}{=}1,$ 

resources\_per\_worker=None)

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.

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.CUDA) 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 (ComputeResources.GPUToolkit, optional) The preferred toolkit to use when running on GPUs.

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.
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.

### 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 ComputeResources.GPUToolkit

#### 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

```
 \textbf{class} \  \, \textbf{openff.evaluator.backends.} \\ \textbf{QueueWorkerResources}(number\_of\_threads=1, number\_of\_gpus=0, \\ preferred\_gpu\_toolkit=None, \\ per\_thread\_memory\_limit=<Quantity(1, \\ 'gigabyte')>, wallclock\_time\_limit='01:00') \\
```

An extended resource object with properties specific to calculations which will run on queue based resources, such as LSF, PBS or SLURM.

#### **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**

|--|

#### **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_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 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\_toolkit

The preferred toolkit to use when running on GPUs.

Type ComputeResources.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**

class openff.evaluator.backends.dask.BaseDaskBackend(number\_of\_workers=1, re-

sources\_per\_worker=<openff.evaluator.backends.backends.Comp object>)

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=<openff.evaluator.backends.backends.ComputeResources object>)
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

 ${\bf class}\ open {\bf ff.evaluator.backends.dask.BaseDaskJobQueueBackend} ({\it minimum\_number\_of\_workers=1},$ 

 $maximum\_number\_of\_workers{=}1,$ 

re-

sources\_per\_worker=<openff.evaluator.backends.back

object>, 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=<openff.evaluator.backends.backends.QueueWorkerResources object>, 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.

#### Returns

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, re-

sources\_per\_worker=<openff.evaluator.backends.backends.Com
object>)

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=<openff.evaluator.backends.backends.ComputeResources object>)
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**

|--|

### 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,
>>>
                                      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',
>>> 1
>>>
>>> # Define extra options to only run on certain node groups
>>> extra_script_options = [
        '-m "ls-gpu lt-gpu"'
>>>
>>> ]
>>>
>>>
>>> # Create the backend which will adaptively try to spin up between one and
>>> # ten workers with the requested resources depending on the calculation.
→load.
```

(continues on next page)

(continued from previous page)

```
>>> 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
<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.

# Returns

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**

Constructs a new DaskLSFBackend object

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.
>>>
                                      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',
>>> 1
>>>
>>> # Create the backend which will adaptively try to spin up between one and
>>> # ten workers with the requested resources depending on the calculation.
\rightarrowload.
>>> from openff.evaluator.backends.dask import DaskPBSBackend
>>>
>>> pbs_backend = DaskPBSBackend(minimum_number_of_workers=1,
                                 maximum_number_of_workers=10,
>>>
```

(continues on next page)

(continued from previous page)

>>>	resources_per_worker=resources,
>>>	queue_name=' <mark>gpuqueue'</mark> ,
>>>	<pre>setup_script_commands=setup_script_commands)</pre>

# **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.

## Returns

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.

\_\_init\_\_()

Constructs a new StorageBackend object.

### **Methods**

cking whether the
ct is stored in the
object exists in the
data matching the
ieving ForceField-
estimators storage
g ForceFieldSource
stem, returning the
i

### 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\_force\_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.
<pre>retrieve_force_field(storage_key)</pre>	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.
<pre>store_force_field(force_field)</pre>	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 StorageBackend 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.

\_\_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.
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>parse_json(string_contents)</pre>	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.

## 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.

 $\begin{tabular}{ll} \textbf{Parameters} & \textbf{string\_contents} & (str & or & bytes) - \textbf{The typed json string}. \end{tabular}$ 

**Returns** The parsed class.

**Return type** Any

# 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()	
Grand de con (G1 - mode)	Constation ship of forms a ICON 61.
<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>has_ancillary_data()</pre>	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>parse_json(string_contents)</pre>	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.

# 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()	
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.
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.
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
<pre>validate([attribute_type])</pre>	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()	
<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.
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()	
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>	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**

coordinate_file_name	The name of a coordinate file which encodes the
	topology information of the system.
force_field_id	The id of the force field parameters used to generate
	the data.
number_of_molecules	The total number of molecules in the system.
observables	A frame of observables collected over the duration of
	the simulation.
property_phase	The phase of the system (e.g.
source_calculation_id	The server id of the calculation which yielded this
	data.
statistical_inefficiency	The statistical inefficiency of the collected data.
substance	A description of the composition of the stored sys-
	tem.
thermodynamic_state	The state at which the data was collected.
trajectory_file_name	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.

Returns

```
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

## class openff.evaluator.storage.data.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.

# **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 attribute_type.
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.

continues on next page

Table	206 -	<ul> <li>continued</li> </ul>	from	previous	nage
Iabic	200	COLLINIACA	110111	DICVIOUS	payc

14515 255 55111114	sa nom providad pago
<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.
<pre>validate([attribute_type])</pre>	Validate the values of the attributes.

### **Attributes**

end_state_trajectory	The name of a .dcd trajectory file containing config-
	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

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

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 attribute_type.
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.
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.

### **Returns**

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.
<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**

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()	
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.

### **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.

### Returns

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  ${\tt BaseSimulationData}$  derived data.

\_\_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.
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	<i>query</i> 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.

Returns

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()	
<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.
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.
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.
	continues on next nego

continues on next page

# Table 217 - continued from previous page

thermodynamic\_state

The state at which the data should have been collected.

### classmethod data\_class()

The type of data class that this query can be applied to.

Returns

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.
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.

### classmethod data\_class()

The type of data class that this query can be applied to.

### Returns

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 at-
	tributes of a storage query.

### **FilePath**

 ${\bf class}\ {\bf open ff. evaluator. storage. attributes. \textbf{FilePath}$ 

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
V	comparisons.
<pre>center(width[, fillchar])</pre>	Return a centered string of length width.
count(sub[, start[, end]])	Return the number of non-overlapping occurrences
	of substring sub in string S[start:end].
encode([encoding, errors])	Encode the string using the codec registered for en-
	coding.
	continues on next page

continues on next page

Table	221	<ul> <li>continued from</li> </ul>	om previous page

	- continued from previous page
<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
	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,
	False otherwise.
isalpha()	Return True if the string is an alphabetic string, False
	otherwise.
isascii()	Return True if all characters in the string are ASCII,
	False otherwise.
<pre>isdecimal() isdigit()</pre>	Return True if the string is a decimal string, False
	otherwise.
	Return True if the string is a digit string, False other-
	wise.
isidentifier()	Return True if the string is a valid Python identifier,
	False otherwise.
islavar()	
islower()	Return True if the string is a lowercase string, False otherwise.
isnumeric()	Return True if the string is a numeric string, False
	otherwise.
isprintable() isspace()	Return True if the string is printable, False otherwise.
	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
	otherwise.
<pre>join(iterable, /)</pre>	Concatenate any number of strings.
ljust(width[, fillchar])	Return a left-justified string of length width.
lower()	Return a copy of the string converted to lowercase.
lstrip([chars])	Return a copy of the string with leading whitespace
	removed.
maketrans(x[, y, z])	Return a translation table usable for str.translate().
partition(sep,/)	Partition the string into three parts using the given
	separator.
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.
2 Jac c (widdie, miende)	continues on next page
	continues on next page

\_\_\_\_

Table 221 - continued from previous page

	To the desired and desired
rpartition(sep,/)	Partition the string into three parts using the given
	separator.
rsplit([sep, maxsplit])	Return a list of the words in the string, using sep as
	the delimiter string.
rstrip([chars])	Return a copy of the string with trailing whitespace
	removed.
split([sep, maxsplit])	Return a list of the words in the string, using sep as
	the delimiter 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 low-
	ercase characters to uppercase.
title()	Return a version of the string where each word is ti-
	tlecased.
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(encoding='utf-8', errors='strict')

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.

Use keyword.iskeyword() to test for reserved identifiers such as "def" and "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(x, y=None, z=None,/)

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.

#### 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 words in the string, using sep as the delimiter string.

**sep** The delimiter according which to split the string. None (the default value) means split according to any whitespace, and discard empty strings from the result.

maxsplit Maximum number of splits to do. -1 (the default value) means no limit.

Splits are done starting at the end of the string and working 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 words in the string, using sep as the delimiter string.

**sep** The delimiter according which to split the string. None (the default value) means split according to any whitespace, and discard empty strings from the result.

maxsplit Maximum number of splits to do. -1 (the default value) means no limit.

## 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*, *typing.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**

## **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 (bool) – 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

Encapsulates and prepares a workflow which is able to
estimate a physical property.
An exception which was raised while executing a work-
flow protocol.
A hierarchical structure for storing and submitting the
workflows which will estimate a set of physical proper-
ties
The result of executing a Workflow as part of a Work-
flowGraph.
The base class for a protocol which would form one step
of a larger property calculation workflow.
A graph of connected protocols which may be executed
together.
A group of workflow protocols to be executed in one
batch.
A decorator which registers a class as being a protocol
which may be included in workflows.
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.
)	
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**

- C: 1 1	TOTAL ALL ALL ALL ALL ALL ALL ALL ALL ALL
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*.

 $\verb|static generate_default_metadata|| physical\_property, force\_field\_path, parametadata|| physical\_property, force\_field\_path, parametadata|| physical\_property|| property|| pr$ 

ter\_gradient\_keys=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, target\_uncertainty=None)

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

ison(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**

init()	
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.

**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 <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**

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)	
<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 <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).
	continues on next page

2.32. API 261

continues on next page

Table 231 - continued from previous page

	1 1 9
<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.
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 uuid 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 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

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** (*bool*) 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.
<pre>add_protocols(*protocols)</pre>	Add protocols to this group.
<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.
	continues on next page

Table 235 - continued from previous page

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.
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.
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 uuid 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

#### merae(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

 ${\tt openff.evaluator.workflow.register\_workflow\_protocol(} {\it protocol\_class})$ 

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 icon(file noth)	Create this phicat from a ISON file
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.
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.
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**

 $\textbf{class} \ open ff. evaluator. work flow. schemas. \textbf{ProtocolGroupSchema} (\textit{unique\_id=None}, \\$ 

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.
<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.
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.
apply(protocols[, template_values,])	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.
<pre>update_references(protocols,)</pre>	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()	
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>replace_protocol_types(protocol_replacements)</pre>	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

BaseMergeBehaviour	A base class for enums which will describes how at-
	tributes should be handled when attempting to merge
	similar protocols.
MergeBehaviour	A enum which describes how attributes should be han-
	dled when attempting to merge similar protocols.
InequalityMergeBehaviour	A enum which describes how attributes which can be
	compared with inequalities should be merged.
InputAttribute	A descriptor used to mark an attribute of an object as an
	input to that object.
OutputAttribute	A descriptor used to mark an attribute of an as an output
	of that object.

# BaseMergeBehaviour

3
<b>class</b> openff.evaluator.workflow.attributes. <b>BaseMergeBehaviour</b> ( <i>value</i> )  A base class for enums which will describes how attributes should be handled when attempting to merge similar protocols.
init()
MergeBehaviour
<pre>class openff.evaluator.workflow.attributes.MergeBehaviour(value)    A enum which describes how attributes should be handled when attempting to merge similar protocols.</pre>
This enum may take values of
• ExactlyEqual: This attribute must be exactly equal between two protocols for them to be able to merge.
<ul> <li>Custom: This attribute will be ignored by the built-in merging code such that user specified behavior ca be implemented.</li> </ul>
init()
Attributes
ExactlyEqual
Custom
InequalityMergeBehaviour
<pre>class openff.evaluator.workflow.attributes.InequalityMergeBehaviour(value)     A enum which describes how attributes which can be compared with inequalities should be merged.</pre>
This enum may take values of
<ul> <li>SmallestValue: When two protocols are merged, the smallest value of this attribute from either protocol i retained.</li> </ul>
<ul> <li>LargestValue: When two protocols are merged, the largest value of this attribute from either protocol i retained.</li> </ul>
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:

```
>>> from openff.evaluator.attributes import AttributeClass
>>> from openff.evaluator.workflow.attributes import InputAttribute
>>>
>>> class MyObject(AttributeClass):
>>>
>>> my_input = InputAttribute(
>>> docstring='An input will be used.',
>>> type_hint=float,
>>> default_value=0.1
>>> )
```

**Parameters merge\_behavior** (BaseMergeBehaviour) – 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**

#### Placeholder Values

ReplicatorValue	A placeholder value which will be set by a protocol repli-
	cator with the specified id.
ProtocolPath	Represents a pointer to the output of another protocol.

## ReplicatorValue

```
\textbf{class} \ \ \textbf{open} \textbf{ff.evaluator.workflow.utils.} \textbf{ReplicatorValue} (\textit{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.
<pre>from_string(existing_path_string)</pre>	
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.
replace_protocol(old_id, new_id)	Redirect the input to point at a new protocol.

#### **Attributes**

full_path is_global	The full path referenced by this object.
13_910001	
last_protocol	The end protocol id of the path.
path_separator	
property_name	The property name pointed to by the path.
property_separator	
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 uuid** (str) – The uuid 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.

continues on next page

Table	255 -	continued	from	previous page

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**

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>
	continues on next page

Table 256 – continued from previous page

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.
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 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 -

## 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

ed with
ed with
on.
vpe.
ldren's,
ts / out-
otocols
l by <i>in-</i>
ng class
otocol
d.

# **Attributes**

allow_merging	Input - 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.
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.
	continues on next page

Table 255 –	continued from previous page
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.

Table 250 - continued from provious page

### 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** typing.Union[int, float, openff.evaluator.utils.units.Quantity]

### 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. 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 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 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

## class openff.evaluator.protocols.analysis.AverageDielectricConstant(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)	
<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.
	continues on next page

Table 260 - continued from previous page

<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.
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	Input - 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	Input - 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**

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>
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_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[openff.evaluator.utils.observables.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

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.

## 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)	
	A II D II I
<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.
<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).

continues on next page

Table 264 – continued from previous page

	<u> </u>
<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.
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	<b>Input</b> - 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 *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 -

### **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**

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 <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).
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>
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.
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** typing.Union[list, openff.evaluator.utils.timeseries.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

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 *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**

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.
<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.
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_coordinate_file	<b>Input</b> - The file path to the starting coordinates of a
	trajectory.
input_trajectory_path	<b>Input</b> - The file path to the trajectory to subsample.
output_trajectory_path	Output - The file path to the subsampled 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.
time_series_statistics	<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

**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.

## 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** typing.Union[list, openff.evaluator.utils.timeseries.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 -

# **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.
<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_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_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** typing.Union[openff.evaluator.utils.observables.ObservableArray, openff.evaluator.utils.observables.ObservableFrame]

### output\_observables

**Output** - The decorrelated observables. The default value of this attribute is not set and must be set by the user..

**Type** typing.Union[openff.evaluator.utils.observables.ObservableArray, openff.evaluator.utils.observables.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

**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.

## 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** typing.Union[list, openff.evaluator.utils.timeseries.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 -

### **Coordinate Generation**

BuildCoordinatesPackmol	Creates a set of 3D coordinates with a specified composition 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**

**class** openff.evaluator.protocols.coordinates.**BuildCoordinatesPackmol**(*protocol\_id*) Creates a set of 3D coordinates with a specified composition 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 <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.
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	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.
	continues on next page

Table 274 – continued from previous page

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.
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.

#### 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 g / ml.

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 Å.

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 <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.
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.
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.
	continues on next page

Table	276 -	continued	from	previous page

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 *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

### mass\_density

Input - The target density of the created system. The default value of this attribute is 0.95 g / ml.

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  $2.0\,\text{ Å}$ .

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.
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 <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.
	continues on next page

Table 277 – continued from previous page

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	<b>Input</b> - A substance containing only the ligand to
	dock.
number_of_ligand_conformers	Input - 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	<b>Output</b> - The residue name assigned to the receptor.
required_inputs	The inputs which must be set on this protocol.
	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** BuildDockedCoordinates.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

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.
•	

continues on next page

Table	270	aantinuad	from	provious	
rabie	2/9-	continued	1110111	previous	page

	<u> </u>
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.
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.
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.

# 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

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**

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.
<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.
<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	<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 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

## 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 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 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.
<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 <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).
get_value(reference_path)	Returns the value of one of this protocols inputs / out-
	puts.
	continues on next page

Table 284 – continued from previous page

<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

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

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** TemplateBuildSystem.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.
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.
charge_backend	<b>Input</b> - The backend framework to use to assign par-
	tial charges.
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 ChargeBackend(value)

The framework to use to assign partial charges.

## charge\_backend

Input - The backend framework to use to assign partial charges.

Type BuildTLeapSystem.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.
- **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** TemplateBuildSystem.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 <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).
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>
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.
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	Input - 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.

# 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** typing.Union[openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray]

## 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** typing.Union[openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.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.
- $\mathbf{new\_id}(str)$  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.

## 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**

init(protocol_id)	Constructs a new ProtocolGroup.
add_condition(condition_to_add)	Adds a condition to this groups list of conditions if it
	not already in the condition list.
add_protocols(*protocols)	Add protocols to this group.
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 <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).
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 ProtocolGroup 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.
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.
	continues on next page

Table 293 - continued from previous page

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** typing.Union[int, float, openff.evaluator.utils.units.Quantity]

## 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** typing.Union[int, float, openff.evaluator.utils.units.Quantity]

### type

The right-hand value to compare. The default value of this attribute is Type.LessThan.

**Type** Conditional Group. Condition. 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 (ConditionalGroup.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 unid 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**

**class** openff.evaluator.protocols.miscellaneous.**AddValues**(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.
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 <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>
json([file_path, format])	Creates a JSON representation of this class.
	continues on next page

Table 295 – continued from previous page

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.
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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.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

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**

 $\textbf{class} \ \ \textbf{openff.evaluator.protocols.miscellaneous.} \\ \textbf{SubtractValues} (\textit{protocol\_id})$ 

A protocol to subtract one value from another such that:

result = value\_b - value\_a
\_\_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 <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).
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>
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.
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.
	continues on next page

Table 298 –	continued	from	previous p	age
-------------	-----------	------	------------	-----

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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray]

#### 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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray]

#### 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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.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

## 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.

 $\label{eq:protocol} \textbf{Parameters} \ \ \textbf{other} \ (\texttt{Protocol}) - \textbf{The protocol to merge into this one}.$ 

**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 -

# **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)	
<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.
<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.
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	<b>Input</b> - 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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray]

## multiplier

**Input** - The scalar to multiply by. The default value of this attribute is not set and must be set by the user..

**Type** typing.Union[int, float, openff.evaluator.utils.units.Quantity]

#### result

**Output** - The result of the multiplication. The default value of this attribute is not set and must be set by the user..

**Type** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.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

#### 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**

```
\textbf{class} \ \ \textbf{openff.evaluator.protocols.miscellaneous.} \textbf{DivideValue}(\textit{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 <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).
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>
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.
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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray]

### divisor

**Input** - The scalar to divide by. The default value of this attribute is not set and must be set by the user..

**Type** typing.Union[int, float, openff.evaluator.utils.units.Quantity]

### result

**Output** - The result of the division. The default value of this attribute is not set and must be set by the user..

**Type** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.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

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 -

# WeightByMoleFraction

 $\textbf{class} \ open ff. evaluator. protocols. \texttt{miscellaneous}. \textbf{WeightByMoleFraction} (\textit{protocol\_id})$ 

Multiplies a value by the mole fraction of a component in a Substance.

```
__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.
	continues on next page

Table 303 – continued from previous page

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.
<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.
component	Input - 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	<b>Input</b> - The value to be weighted.
weighted_value	Output - The value weighted by the component's
	$mole\ fraction\ as\ determined\ from\ the\ `full\_substance.$

## value

Input - The value to be weighted. The default value of this attribute is not set and must be set by the user..

**Type** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray]

### 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** typing.Union[int, float, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.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 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 -

# **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**

ed with
ed with
on.
vpe.
ldren's,
ts / out-
otocols
l by <i>in-</i>
ng class
otocol
d.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
component_roles	Input - 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

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 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.

# **DummyProtocol**

```
\textbf{class} \ \ \textbf{open} \textbf{ff.evaluator.protocols.miscellaneous.} \textbf{\textit{DummyProtocol}} (\textit{protocol\_id})
```

A protocol whose only purpose is to return an input value as an output value.

```
__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 <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>
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.
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** typing.Union[str, int, float, openff.evaluator.utils.units.Quantity, openff.evaluator.utils.units.Measurement, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.forcefield.gradients.ParameterGradientKey, list, tuple, dict, set, frozenset]

### output\_value

Output - A dummy output. The default value of this attribute is not set and must be set by the user..

Type typing.Union[str, int, float, openff.evaluator.utils.units.Quantity,

openff.evaluator.utils.units.Measurement, openff.evaluator.utils.observables.Observable, openff.evaluator.utils.observables.ObservableArray, openff.evaluator.forcefield.gradients.ParameterGradient, openff.evaluator.forcefield.gradients.ParameterGradientKey, list, tuple, dict, set, frozenset]

## 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 -

# **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**

class openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation(protocol\_id)

A protocol to minimise the potential energy of a system using  $\mbox{\rm Open} \mbox{\rm MM}.$ 

\_\_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.
<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.
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	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.
output_coordinate_file	Output - The file path to the minimised coordinates.
outputs	A dictionary of the outputs of this property.
	continues on next page

Tabla	211	- continued	from	previous page
Iable	011	- continued	110111	previous page

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.
tolerance	<b>Input</b> - 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 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).

 $\label{parameters} \textbf{Parameters} \ \ \textbf{reference\_path} \ (\texttt{ProtocolPath}) - \textbf{The path pointing to the attribute to return}.$ 

**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 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.

#### tolerance

Input - The energy tolerance to which the system should be minimized. The default value of this attribute is  $10.0 \, \text{kJ} / \text{mol}$ .

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.
	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).
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>
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.
	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 a merge with other protocols.         checkpoint_frequency       Input - The frequency (in multiples put_frequency) with which to write to a chile, e.g.         dependencies       A list of pointers to the protocols which this takes input from.         enable_pbc       Input - If true, periodic boundary condition enabled.         ensemble       Input - The thermodynamic ensemble to sin gradient_parameters         Input - An optional list of parameters to diff the evaluated energies with respect to.         high_precision       Input - If true, the simulation will be run us ble precision.         id       The unique id of this protocol.	of out- neckpoint a protocol ns will be mulate in. ferentiate		
put_frequency) with which to write to a characteristic file, e.g.  dependencies  A list of pointers to the protocols which this takes input from.  enable_pbc  Input - If true, periodic boundary condition enabled.  ensemble  Input - The thermodynamic ensemble to sin gradient_parameters  Input - An optional list of parameters to differ the evaluated energies with respect to.  high_precision  Input - If true, the simulation will be run us ble precision.	protocol as will be mulate in. ferentiate		
takes input from.  Input - If true, periodic boundary condition enabled.  ensemble  gradient_parameters  Input - An optional list of parameters to diff the evaluated energies with respect to.  high_precision  Input - If true, the simulation will be run us ble precision.	ns will be nulate in.		
ensemble Input - The thermodynamic ensemble to sir gradient_parameters Input - An optional list of parameters to diff the evaluated energies with respect to.  high_precision Input - If true, the simulation will be run us ble precision.	nulate in. ferentiate		
gradient_parametersInput - An optional list of parameters to diffhigh_precisionInput - If true, the simulation will be run usble precision.	ferentiate		
the evaluated energies with respect to.  high_precision  Input - If true, the simulation will be run us ble precision.			
ble precision.	sing dou-		
id The unique id of this protocol.	•		
<u> </u>			
input_coordinate_file			
observables  Output - The observables collected during ulation.	the sim-		
output_coordinate_fileOutput - The file path to the coordinates of system configuration.			
output_frequency Input - The frequency (in number of ste which to write to the output statistics and t files.			
outputs A dictionary of the outputs of this property.			
parameterized_systemInput - The parameterized system object vcodes the systems potential energy function			
required_inputs The inputs which must be set on this protoc	ol.		
schema A serializable schema for this object.			
steps_per_iterationInput - The number of steps to propogate theby at each iteration.	ie system		
thermodynamic_state  Input - The thermodynamic conditions to under The default value of this attribute is no must be set by the user			
thermostat_friction			
<b>Input</b> - The timestep to evolve the system be step.	y at each		
forward by the steps_per_iteration number	<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 saming the simulation.	nled dur-		

# 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

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\_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 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.

#### 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 / ps.

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 fs.

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**

 $\textbf{class} \ \ \textbf{openff.evaluator.protocols.openmm.} \textbf{OpenMMEvaluateEnergies} (\textit{protocol\_id})$ 

Re-evaluates the energy of a series of configurations for a given set of force field parameters using OpenMM.

\_\_init\_\_(protocol\_id)

# **Methods**

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 <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.
<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.
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	<b>Input</b> - 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.
	continues on next page

T-1-1-	~4=	- continued	f		
בוחבו	315 -	. CONTINUEA	trom	nravinie	nana
Iabic	010 -	COHUHUCU	110111	DIEVIOUS	Dauc

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.

# 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**

**class** openff.evaluator.protocols.paprika.coordinates.**PreparePullCoordinates**(*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**

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.
	continues on next page

Table 317 – continued from previous page

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.
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 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

## 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 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 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

apply_replicator(replicator, template_values)  can_merge(other[, path_replacements])  Determines whether this protocol can be merged with another.  execute([directory, available_resources])  from_json(file_path)  Create this object from a JSON file.  from_schema(schema)  get_attributes([attribute_type])  get_class_attribute(reference_path)  Returns all attributes of a specific attribute_type.  get_value(reference_path)  Returns one of this protocols, or any of its children's, 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.  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.  validate([attribute_type])  Validate the values of the attributes.		
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.	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.		
another.  execute([directory, available_resources])	<pre>apply_replicator(replicator, template_values)</pre>	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.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.	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 / 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.		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.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.	<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.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.  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>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.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.  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 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_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 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_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_uuid(value)       Prepend a unique identifier to this protocols id.         set_value(reference_path, value)       Sets the value of one of this protocols inputs.		which one of this protocols inputs (specified by in-
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_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 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>json([file_path, format])</pre>	
structure.  replace_protocol(old_id, new_id)  set_uuid(value)  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)	Finds each input which came from a given protocol
	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)	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.
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.

continues on next page

## Table 320 - continued from previous page

substance

**Input** - 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 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

### 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 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 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**

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.
<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**

-11	Innex Deferes whather this weeks calculated at the
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_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.
	continues on next page

Table	322 –	<ul> <li>continued</li> </ul>	from	previous	page
	<u> </u>	oon an aca		picticac	Page

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 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 -

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.
<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 <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.
	continues on next name

continues on next page

Table	324 -	<ul> <li>continued</li> </ul>	from	previous page
IUDIO	OL :	COLLULIACA		providuo pago

<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.
attach_lambdas	Input - 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.

#### **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 -

# **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)	
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).
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>
	continues on next page

Table 326 – continued from previous page

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**

-11	Innex Defense whather this waste calculations of the
allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
	merge with other protocols.
attach_lambdas	Input - 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	<b>Input</b> - 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 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 -

# **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)	
<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.
<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.
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.
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.
	continues on next page

Table 329 – continued from previous page

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.

## 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

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

## 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**

**class** openff.evaluator.protocols.paprika.restraints.**ApplyRestraints**(*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.
<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.
<pre>load_restraints(file_path)</pre>	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.
	continues on next page

Table	331	<ul> <li>continued</li> </ul>	from	previous page
IUDIC	OO I	CONTINUCC	110111	providus page

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.

Returns

**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

## 

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\_ison(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 -

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 calculation.
ComputeSymmetryCorrection	Computes the symmetry correction for an APR calcula-
	tion which involves a guest with symmetry.
ComputeReferenceWork	Computes the reference state work.

## **AnalyzeAPRPhase**

class openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase(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**

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 <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>
json([file_path, format])	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
·	continues on next page

Table 333 - continued from previous page

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.
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 *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 -

# 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**

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.
	continues on next page

Table 335 – continued from previous page

	1 1 5
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>
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.
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	Input - 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

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 *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 -

# ComputeReferenceWork

```
__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.
	continues on next page

Table 337 - continued from previous page

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.
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.
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

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 -

## 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

**class** openff.evaluator.protocols.reweighting.**ConcatenateTrajectories**(*protocol\_id*) A protocol which concatenates multiple trajectories into a single one.

\_\_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 <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.
<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.
dependencies	A list of pointers to the protocols which this protocol
	takes input from.
id	The unique id of this protocol.
input_coordinate_paths	<b>Input</b> - 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.
	continues on next page

## Table 341 – continued from previous page

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 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 -

## 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)	
<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.
<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_observables	<b>Input</b> - A list of observable arrays to concatenate.
output_observables	Output - The concatenated observable array.
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\_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** typing.Union[openff.evaluator.utils.observables.ObservableArray, openff.evaluator.utils.observables.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

**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.

**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

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.

# **Attributes**

allow_merging	<b>Input</b> - Defines whether this protocols is allowed to
_ 5 5	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	<b>Input</b> - 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.
	continues on next page

# Table 345 - continued from previous page

trajectory\_file\_path

**Input** - 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

**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.

**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)	
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).
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>
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 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.
	continues on next page

Table 347 - continued from previous page

dependencies	A list of pointers to the protocols which this protocol
acpenacieres	1 1
	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	Input - 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[openff.evaluator.utils.observables.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

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

class openff.evaluator.protocols.reweighting.ReweightObservable(protocol id)

Reweight an array of observables to a new state using MBAR.

```
__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.
	continues on next page

Table 348 – continued from previous page

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.
bootstrap_iterations	Input - 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	<b>Input</b> - 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	Input - 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[openff.evaluator.utils.observables.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 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.

### 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

# $\textbf{class} \ open ff. evaluator. protocols. reweighting. \textbf{ReweightDielectricConstant} (protocol\_id)$

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**

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.
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	<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.
dipole_moments	Input - 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** typing.Union[openff.evaluator.utils.observables.ObservableArray, list]

#### 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** typing.Union[openff.evaluator.utils.observables.ObservableArray, list]

## 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

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

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[openff.evaluator.utils.observables.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

class openff.evaluator.protocols.simulation.BaseEnergyMinimisation(protocol\_id)

A base class for protocols which will minimise the potential energy of a given 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).
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.
	continues on next page

Table 353 - continued from previous page

set_value(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.
Input - 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.

### 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  $10.0 \, kJ / mol$ .

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

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 -

#### **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.
<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).
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>
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_gpu_platforms	<b>Input</b> - If true, the simulation will be performed
	using a GPU if available, otherwise it will be con-
	strained to only using CPUs.
allow_merging	<b>Input</b> - 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	<b>Input</b> - If true, periodic boundary conditions will be
	enabled.
ensemble	<b>Input</b> - The thermodynamic ensemble to simulate in.
gradient_parameters	Input - 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.
	continues on next page

Table	356 -	continued	from	previous page	9
-------	-------	-----------	------	---------------	---

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	<b>Input</b> - 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 fs.

**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 / ps.

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 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 -

### 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

ed with
ed with
on.
ype.
ldren's,
ts / out-
otocols
l by <i>in-</i>
ng class
otocol
d.
1

# **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.
	continues on next page

Table 359 - continued from previous page

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** typing.Union[list, tuple]

#### 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

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**

class openff.evaluator.protocols.yank.BaseYankProtocol(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**

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.
<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_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.
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.

continues on next page

Table 362 - continued from previous page

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	Input - Controls whether or not to run YANK at high
	verbosity.

### 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 fs.

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 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.

### LigandReceptorYankProtocol

class openff.evaluator.protocols.yank.LigandReceptorYankProtocol(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.
<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.
	continues on poyt page

2.32. API 541

continues on next page

Table 363 – continued from previous page

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 <i>attribute_type</i> .
	1
<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>
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.
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	<b>Input</b> - The list of electrostatic alchemical states that
	YANK should sample at when calculating the free
	energy of the ligand in complex with the receptor.
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	Input - 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.
	continues on next page

Table	364 -	continued	from	previous	nage
Iabic	007	CONTINUCC	110111	picvious	page

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	Input - 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 apply_restraints is True The default value of this
	attribute is RestraintType.Harmonic.
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.
solvated_complex_coordinates	<b>Input</b> - The file path to the solvated complex coordi-
	nates.
solvated_complex_system	<b>Input</b> - The parameterized solvated complex system
	object.
solvated_complex_trajectory_path	Output - The file path to the generated ligand trajec-
	tory.
solvated_ligand_coordinates	Input - The file path to the solvated ligand coordi-
	nates.
solvated_ligand_system	<b>Input</b> - The parameterized solvated ligand system ob-
	ject.
solvated_ligand_trajectory_path	Output - The file path to the generated ligand trajec-
	tory.
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	Input - The length of the timestep to take.
verbose	Input - Controls whether or not to run YANK at high
	verbosity.

## class RestraintType(value)

The types of ligand restraints available within yank.

## 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** *LigandReceptorYankProtocol.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 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\_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 *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 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 fs.

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**

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.
<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.
	continues on payt page

continues on next page

Table 365 – continued from previous page

<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**

s between saving which this protocol nemical states that nemical states that regy difference beand the second sol-
which this protocol memical states that memical states that rgy difference be-
which this protocol memical states that memical states that rgy difference be-
nemical states that nemical states that rgy difference be-
nemical states that
nemical states that
gy difference be-
gy difference be-
nd the second sol-
ers to differentiate
pect to.
ised for equilibra-
ations to perform.
property.
is protocol.
ct.
reate and validate
any simulations.
he composition of
nates of the solute
e of transforming
o fully intomostin
a fully interacting
a runy interacting
1

Table 366 – continued from previous page

solution_1_system	Input - The parameterized system object of the solute
50140101-1-5/5004	embedded in the first solvent.
solution_1_trajectory_path	Output - The file path to the trajectory containing the
	solute in the first solvent.
solution_2_coordinates	<b>Input</b> - The file path to the coordinates of the solute
	embedded in the second solvent.
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.
solution_2_system	<b>Input</b> - The parameterized system object of the solute
	embedded in the second solvent.
solution_2_trajectory_path	Output - The file path to the trajectory containing the
	solute in the second solvent.
solvent_1	<b>Input</b> - The substance describing the composition of
	the first solvent.
solvent_1_coordinate_path	Output - The file path to the coordinates of only the
	first solvent.
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

**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

### 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 fs.

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**

class openff.evaluator.protocols.utils.SimulationProtocols(build\_coordinates:

openff.evaluator.protocols.coordinates.BuildCoordinatesPassign\_parameters:
openff.evaluator.protocols.forcefield.BaseBuildSystem,
energy\_minimisation:
openff.evaluator.protocols.openmm.OpenMMEnergyMini
equilibration\_simulation:
openff.evaluator.protocols.openmm.OpenMMSimulation,
production\_simulation:
openff.evaluator.protocols.openmm.OpenMMSimulation,
analysis\_protocol:
openff.evaluator.protocols.utils.S,
converge\_uncertainty:
openff.evaluator.workflow.protocols.ProtocolGroup,
decorrelate\_trajectory:
openff.evaluator.protocols.analysis.DecorrelateTrajectory,

openff.evaluator.protocols.analysis.DecorrelateObservable

The common set of protocols which would be required to estimate an observable by running a new molecule simulation.

decorrelate\_observables:

```
__init__(build_coordinates: openff.evaluator.protocols.coordinates.BuildCoordinatesPackmol, assign_parameters: openff.evaluator.protocols.forcefield.BaseBuildSystem, energy_minimisation: openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation, equilibration_simulation: openff.evaluator.protocols.openmm.OpenMMSimulation, production_simulation: openff.evaluator.protocols.openmm.OpenMMSimulation, analysis_protocol: openff.evaluator.protocols.utils.S, converge_uncertainty: openff.evaluator.workflow.protocols.ProtocolGroup, decorrelate_trajectory: openff.evaluator.protocols.analysis.DecorrelateTrajectory, decorrelate_observables: openff.evaluator.protocols.analysis.DecorrelateObservables) → None
```

#### **Methods**

init(build_coordinates,)
Attributes
Attributes
build_coordinates
assign_parameters
energy_minimisation
equilibration_simulation
production_simulation
production_simulation
analysis_protocol
converge_uncertainty
decorrelate_trajectory
decorrelate_observables

### ReweightingProtocols

class openff.evaluator.protocols.utils.ReweightingProtocols(unpack\_stored\_data:

openff.evaluator.protocols.storage.UnpackStoredSimulati join\_trajectories: openff.evaluator.protocols.reweighting.ConcatenateTraje join\_observables: openff.evaluator.protocols.reweighting.ConcatenateObse build\_reference\_system: openff.evaluator.protocols.forcefield.BaseBuildSystem, evaluate\_reference\_potential: openff.evaluator.protocols.reweighting.BaseEvaluateEne build\_target\_system: openff.evaluator.protocols.forcefield.BaseBuildSystem, evaluate\_target\_potential: openff.evaluator.protocols.reweighting.BaseEvaluateEne statistical inefficiency: openff.evaluator.protocols.utils.S, replicate\_statistics: openff.evaluator.protocols.miscellaneous.DummyProtoco decorrelate\_reference\_potential: openff.evaluator.protocols.analysis.DecorrelateObservab decorrelate\_target\_potential: openff.evaluator.protocols.analysis.DecorrelateObservab decorrelate\_observable: openff.evaluator.protocols.analysis.DecorrelateObservab zero\_gradients: Optional/openff.evaluator.protocols.gradients.ZeroGradient reweight observable: openff.evaluator.protocols.utils.T)

The common set of protocols which would be required to re-weight an observable from cached simulation data.

### **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_observable
zero_gradients
```

### generate base reweighting protocols

reweight\_observable

openff.evaluator.protocols.utils.generate\_base\_reweighting\_protocols(statistical\_inefficiency:

openff.evaluator.protocols.utils.S,
reweight\_observable:
openff.evaluator.protocols.utils.T,
replicator\_id: str =
'data\_replicator',
id\_suffix: str = ") → Tuple[openff.evaluator.protocols.utils.Reweight
openff.evaluator.protocols.utils.T],
openff.evaluator.workflow.schemas.ProtocolF

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:

```
replicator_id: str =
'data_replicator', id_suffix: str =
'') → Tu-
ple[openff.evaluator.protocols.utils.ReweightingPro
openff.evaluator.protocols.reweighting.ReweightOb
```

openff.evaluator.workflow.schemas.ProtocolReplica

openff.evaluator.utils.observables.ObservableType,

### generate\_simulation\_protocols

openff.evaluator.protocols.utils.generate\_simulation\_protocols(analysis\_protocol:

```
openff.evaluator.protocols.utils.S,
use_target_uncertainty: bool,
id_suffix: str = ",
conditional_group: Op-
tional[openff.evaluator.protocols.groups.Conditional
= None, n_molecules: int = 1000)
→ Tu-
ple[openff.evaluator.protocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationProtocols.utils.SimulationPro
```

openff.evaluator.workflow.utils.ProtocolPath, openff.evaluator.storage.data.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 attributes with additional metadata.
UNDEFINED	A custom type used to differentiate between None values, 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**

class openff.evaluator.attributes.Attribute(docstring, type\_hint, de-

fault\_value=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, optional=False, read\_only=False)

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=<openff.evaluator.attributes.attributes.UndefinedAttribute object>, 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*, *typing.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.
<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.

### 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

A class which stores the mean value of an observable as
well as the standard error in the mean.
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.
An enumeration of the common observables which may
be extracted from molecular simulations (or simulation
data) and stored in an ObservableFrame.
A data object for storing and retrieving frames of
the thermodynamic observables enumerated by the
ObservableType enum.
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: Op-

tional[Union[openff.evaluator.utils.units.Measurement, openff.evaluator.utils.units.Quantity]] = None, gradients: Optional[List[openff.evaluator.forcefield.gradients.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.

#### **Methods**

init([value, gradients])	
clear_gradients()	Clears all gradient information.
Attributes	
error	
gradients	
value	

### clear\_gradients()

Clears all gradient information.

### **ObservableArray**

class openff.evaluator.utils.observables.ObservableArray(value: Op-

tional[openff.evaluator.utils.units.Quantity]
= None, gradients: Optional[List[openff.evaluator.forcefield.gradients.ParameterGn
= None)

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[openff.evaluator.utils.units.Quantity] = None, gradients:
Optional[List[openff.evaluator.forcefield.gradients.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.

gradients	
value	The value(s) of the observable.
<pre>property value: openff.e     The value(s) of the observ</pre>	valuator.utils.units.Quantity able.
	$\rightarrow$ openff.evaluator.utils.observables.ObservableArray values stored for this observable at the specified indices.
Parameters indice	$\mathbf{s}$ – The indices of the entries to extract.
Returns	
Return type The sub	oset of the observable values.
openff.eve	bles: openff.evaluator.utils.observables.ObservableArray) → aluator.utils.observables.ObservableArray servables together in the order that they appear in the args list.
	ables – The observables to join.
Returns	•
	ncatenated observable object.
clear_gradients() Clears all gradient informa	
oservableType	
=	observables.ObservableType(value) observables which may be extracted from molecular simulations (or simulation bleFrame.
init()	
Attributes	
PotentialEnergy	
KineticEnergy	
TotalEnergy	
Temperature	
Volume	
Density	

	Table 381 – continued from previous page
Enthalpy	
ReducedPotential	

### **ObservableFrame**

A data object for storing and retrieving frames of the thermodynamic observables enumerated by the ObservableType enum.

**\_\_init\_\_**(*observables: Optional[Dict[Union[str*, openff.evaluator.utils.observables.ObservableType], openff.evaluator.utils.observables.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.
get(k[,d])	
items()	
join(*observable_frames)	Joins multiple observable frames together in the order that they appear in the args list.
keys()	
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 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\_openmm**( $file\_path: str, pressure: Optional[openff.evaluator.utils.units.Quantity] = None) <math>\rightarrow openff.evaluator.utils.observables.ObservableFrame$ 

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.

#### Returns

**Return type** The imported observables.

 $subset(indices: Iterable[int]) \rightarrow openff.evaluator.utils.observables.ObservableFrame$ 

Extracts the subset of the the array which is located at the specified indices.

**Parameters indices** – The indices of the entries to extract.

**Returns** 

**Return type** The subset of data.

**classmethod join**(\*observable\_frames: openff.evaluator.utils.observables.ObservableFrame)  $\rightarrow$  openff.evaluator.utils.observableFrame

Joins multiple observable frames together in the order that they appear in the args list.

**Parameters observable\_frames** – The observable frames to join.

Returns

**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

 $\mathbf{update}(\big[E\,\big],\, **F) \to \text{None. Update D from mapping/iterable E and F}.$ 

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: openff.evaluator.utils.observables.ObservableArray) \rightarrow openff.evaluator.utils.observables.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.

### Returns

**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 0.3.9

## **Bugfixes**

• PR #402: Fix importing full ThermoML archive

### **Behaviour 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.

### 2.33.2 0.3.8

## **Bugfixes**

• PR #390: Fix excluding v-sites from OpenMM positions

### 2.33.3 0.3.7

### **Bugfixes**

• PR #389: Fix v-site positions not set by OpenMM

### 2.33.4 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

### 2.33.5 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.

### 2.33.6 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.

### **Behaviour Changes**

• PR #349: Working files are deleted by default after an estimation batch completes.

### 2.33.7 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.

### 2.33.8 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.

### 2.33.9 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.

### 2.33.10 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.

#### **Behaviour 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).

### 2.33.11 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.

#### 2.33.12 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.

### 2.33.13 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.

### 2.33.14 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.

#### 2.33.15 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.

### 2.33.16 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.

## 2.33.17 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.

## 2.33.18 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.

## 2.33.19 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.

### 2.33.20 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

### 2.33.21 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.

## 2.33.22 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=...)

### 2.33.23 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

## 2.33.24 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.

### 2.33.25 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:

```
X.Y.Z - Descriptive Title
This release...
New Features
* PR #X: Feature summary
Bugfixes
* PR #Y: Fix Summary
Breaking Changes
* PR #Z: Descriptive summary of the breaking change
Migration Guide
This release contained several public API breaking changes. For the most part, these can-
remedied by the follow steps:
* A somewhat verbose guide on how users should upgrade their code given the new breaking.
```

### 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 585

# 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.

588 Bibliography

# **INDEX**

init_() (openff.evaluator.attributes.Attribute method), 564	Symb	ools	init_	_() (openff.evaluator.datasets.PropertyPhase
method), 564 init_() (openff.evaluator.attributes.AttributeClass method), 564 init_() (openff.evaluator.attributes.PlaceholderValue method), 566 init_() (openff.evaluator.backends.CalculationBackend method), 203 init_() (openff.evaluator.backends.CalculationBackend method), 203 init_() (openff.evaluator.backends.ComputeResources method), 204 init_() (openff.evaluator.backends.QueueWorkerResources method), 205 init_() (openff.evaluator.backends.QueueWorkerResources method), 206 init_() (openff.evaluator.backends.QueueWorkerResources method), 207 init_() (openff.evaluator.backends.dask.BaseDaskBackend method), 207 init_() (openff.evaluator.backends.dask.BaseDaskBackend method), 207 init_() (openff.evaluator.backends.dask.DaskLsFBackend method), 210 init_() (openff.evaluator.backends.dask.DaskLsFBackend method), 211 init_() (openff.evaluator.backends.dask.DaskLocalCluster method), 213 init_() (openff.evaluator.backends.dask.DaskLocalCluster method), 213 init_() (openff.evaluator.client.BatchMode method), 80 init_() (openff.evaluator.client.ConnectionOptions method), 80 init_() (openff.evaluator.client.Request method), 82 init_() (openff.evaluator.client.Request method), 82 init_() (openff.evaluator.client.Request method), 83 init_() (openff.evaluator.client.Request Result method), 84 init_() (openff.evaluator.client.Request Result method), 84 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 84 init_() (openff.evaluator.client.Request Result method), 84 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 85 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 86 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 80 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 80 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 80 init_() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 80 init_() (openff.evaluator.datasets.C	init_	() (openff.evaluator.attributes.Attribute		
method), 566 init_O (openff, evaluator. attributes. Placeholder Value method), 566 init_O (openff, evaluator. backends. CalculationBackend method), 203 init_O (openff, evaluator. backends. CalculationBackend method), 203 init_O (openff, evaluator. backends. ComputeResources method), 204 init_O (openff, evaluator. backends. ComputeResources method), 205 init_O (openff, evaluator. backends. QueueWorkerResources method), 205 init_O (openff, evaluator. backends. QueueWorkerResources method), 205 init_O (openff, evaluator. backends. dask. BaseDaskBackend method), 207 init_O (openff, evaluator. backends. dask. BaseDaskBackend method), 207 init_O (openff, evaluator. backends. dask. BaseDaskDob_queueBackReft@od), 164 method), 207 init_O (openff, evaluator. backends. dask. DaskLSFBackend method), 210 init_O (openff, evaluator. backends. dask. DaskLSFBackend method), 210 init_O (openff, evaluator. backends. dask. DaskDaskDaskDaskDaskDaskDaskDaskDaskDask			init_	=
init() (openff, evaluator.backends.CalculationBackend method), 303init() (openff, evaluator.backends.CalculationBackend method), 203init() (openff, evaluator.backends.ComputeResources method), 203init() (openff, evaluator.backends.ComputeResources method), 205init() (openff, evaluator.backends.ComputeResources method), 205init() (openff, evaluator.backends.QueueWorkerResources init() (openff, evaluator.backends.dask.BaseDaskBackend method), 205init() (openff, evaluator.backends.dask.BaseDaskBackend method), 206init() (openff, evaluator.backends.dask.BaseDaskJob)ueueBackEndpod), 164init() (openff, evaluator.backends.dask.DaskLocalCluster method), 211init() (openff, evaluator.backends.dask.DaskLocalCluster method), 211init() (openff, evaluator.backends.dask.DaskLocalCluster method), 210init() (openff, evaluator.backends.dask.DaskLocalCluster method), 210init() (openff, evaluator.client.BatchMode method), 380init() (openff, evaluator.client.EvaluatorClient method), 380init() (openff, evaluator.client.EvaluatorClient method), 380init() (openff, evaluator.client.EvaluatorClient method), 380init() (openff, evaluator.client.Request method), 82init() (openff, evaluator.client.Request method), 82init() (openff, evaluator.client.Request method), 830init() (openff, evaluator.datasets.CalculationSource method), 84init() (openff, evaluator.datasets.CalculationSource method), 85init() (openff, evaluator.datasets.CalculationSource method), 86init() (openff, evaluator.datasets.CalculationSource method), 99init() (openff, evaluator.datasets.PhysicalProperty	init_		init_	
method), 566  _init() (openff, evaluator. backends. CalculationBackend method), 203 _init() (openff, evaluator. backends. CalculationBackend method), 204 _init() (openff, evaluator. backends. ComputeResources method), 204 _init() (openff, evaluator. backends. QueueWorkerResources method), 205 _init() (openff, evaluator. backends. QueueWorkerResources method), 205 _init() (openff, evaluator. backends. QueueWorkerResources method), 179 _init() (openff, evaluator. backends. dask. BaseDaskBackend method), 207 _init() (openff, evaluator. backends. dask. BaseDaskBobQueueBackend method), 208 _init() (openff, evaluator. backends. dask. DaskLSFBackend method), 210 _init() (openff, evaluator. backends. dask. DaskLSFBackend method), 210 _init() (openff, evaluator. backends. dask. DaskLocalCluster method), 210 _init() (openff, evaluator. backends. dask. DaskPBSBackend method), 210 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 171 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 180 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 180 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 169 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 169 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 169 _init() (openff, evaluator. datasets. curation. components. filtering. Filter method), 169 _init() (openff, evaluator. datasets. curat	init		,	method), 152
method), 204  init _() (openff.evaluator.backends. ComputeResources method), 180  init _() (openff.evaluator.backends. ComputeResources method), 204  init _() (openff.evaluator.backends. QueueWorkerResources method), 179  method), 205  init _() (openff.evaluator.backends. QueueWorkerResources method), 179  method), 205  init _() (openff.evaluator.backends. dask. BaseDaskBackend method), 164  method), 207  init _() (openff.evaluator.backends. dask. BaseDaskBackend method), 208  init _() (openff.evaluator.backends.dask. BaseDaskJobQueueBackeffpod), 164  method), 208  init _() (openff.evaluator.backends.dask. DaskLSFBackend method), 161  init _() (openff.evaluator.backends.dask. DaskLSFBackend method), 210  init _() (openff.evaluator.backends.dask. DaskLocalCluster method), 210  init _() (openff.evaluator.backends.dask. DaskPBSBackend method), 171  init _() (openff.evaluator.backends.dask. DaskPBSBackend method), 171  init _() (openff.evaluator.client. Requestend method), 180  init _() (openff.evaluator.client. ConnectionOptions method), 80  init _() (openff.evaluator.client. EvaluatorClient method), 165  init _() (openff.evaluator.client. Request method), 82  init _() (openff.evaluator.client. Request method), 82  init _() (openff.evaluator.datasets. Curation.components. filtering. Filter method), 169  init _() (openff.evaluator.datasets. curation.components. filtering. Filter meth		method), 566	init_	
init_() (openff. evaluator.backends. ComputeResources method), 204  init_() (openff. evaluator.backends. QueueWorkerResources method), 179  init_() (openff. evaluator.backends. QueueWorkerResources method), 179  init_() (openff. evaluator.backends. dask. BaseDaskBackend method), 207  init_() (openff. evaluator.backends. dask. BaseDaskBackend method), 208  init_() (openff. evaluator.datasets. curation.components. filtering. Filter method), 208  init_() (openff. evaluator.datasets. curation.components. filtering. Filter method), 210  init_() (openff. evaluator.backends. dask. DaskLSFBackend method), 210  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 210  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 213  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 80  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 80  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 80  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 171  init_() (openff. evaluator.backends. dask. DaskPBSBackend method), 180	init_		nd	
method), 204  _init() (openff.evaluator.backends.QueueWorkerResources method), 179     method), 205  init() (openff.evaluator.backends.QueueWorkerResources method), 164     method), 207  _init() (openff.evaluator.backends.dask.BaseDaskBackend method), 207  _init() (openff.evaluator.backends.dask.BaseDaskJobQueueBackethod), 164     method), 208  _init() (openff.evaluator.backends.dask.BaseDaskJobQueueBackethod), 164     method), 208  _init() (openff.evaluator.backends.dask.DaskLSFBackend method), 211  _init() (openff.evaluator.backends.dask.DaskLSFBackend method), 210  _init() (openff.evaluator.backends.dask.DaskLocalCluster method), 213  _init() (openff.evaluator.backends.dask.DaskLocalCluster method), 213  _init() (openff.evaluator.client.BatchMode method), 80  _init() (openff.evaluator.client.ConnectionOptions method), 80  _init() (openff.evaluator.client.EvaluatorClient method), 171  _init() (openff.evaluator.datasets.curation.components.filtering.Filter method), 165  _init() (openff.evaluator.client.Request method), 84  _init() (openff.evaluator.client.Request method), 84  _init() (openff.evaluator.datasets.CalculationSource method), 98  _init() (openff.evaluator.datasets.CalculationSource method), 99  _init() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 169  _init() (openff.evaluator.datasets.CalculationSource method), 169  _init() (openff.evaluator.datasets.Curation.components.filtering.Filter method), 169  _init() (op				
init() (operiff.evaluator.backends.QueueWorkerResources	init_		'S	
method), 205				
init() (openff_evaluator.backends.dask.BaseDaskBackend method), 207init() (openff_evaluator.backends.dask.BaseDaskJobQueueBackerdpod), 164method), 208init() (openff_evaluator.backends.dask.DaskLSFBackend method), 211init() (openff_evaluator.backends.dask.DaskLocalCluster method), 210init() (openff_evaluator.backends.dask.DaskLocalCluster method), 210init() (openff_evaluator.backends.dask.DaskPBSBackend method), 213init() (openff_evaluator.backends.dask.DaskPBSBackend method), 213init() (openff_evaluator.client.BatchMode method), 80init() (openff_evaluator.client.ConnectionOptions method), 78init() (openff_evaluator.client.EvaluatorClient method), 78init() (openff_evaluator.client.Request method), 82init() (openff_evaluator.client.Request method), 84init() (openff_evaluator.client.Request method), 85init() (openff_evaluator.datasets.CalculationSource method), 98init() (openff_evaluator.datasets.CalculationSource method), 99init() (openff_evaluator.datasets.Descriptions method), 99init() (openff_evaluator.datasets.PhysicalPropertyinit() (openff_evaluator.datasets.curation.components.filtering.Filter method), 158init() (openff_evaluator.datasets.Caration.components.filtering.Filter method), 158init() (openff_evaluator.datasets.curation.components.filtering.Filter method), 158init() (openff_evaluator.datas	init_		ources	
method), 207 init_() (openff.evaluator.backends.dask.BaseDaskJobQueueBackfelfad), 164 method), 208 init_() (openff.evaluator.backends.dask.BaseDaskJobQueueBackfelfad), 164 method), 211 init_() (openff.evaluator.backends.dask.DaskLSFBackend method), 211 init_() (openff.evaluator.backends.dask.DaskLSFBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 211 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 211 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.backends.dask.DaskLSFBackend method), 210 init_() (openff.evaluator.client.Backends.dask.DaskPBSBackend method), 210 init_() (openff.evaluator.datasets.curation.components.filtering.Filter method), 161 init_() (openff.evaluator.datasets.curation.components.filtering.Filter method), 165 init_() (openff.evaluator.datasets.curation.components.filtering.Filter method), 159 init_() (openff.evaluator.datasets.curation.components.filtering.Filter method), 169 init_() (openff.evaluator.datasets.curation.components.filtering.Filter method), 169 i				
init() (openff.evaluator.backends.dask.BaseDaskJobQueueBackendbod), 164init() (openff.evaluator.backends.dask.DaskLSFBackend method), 211init() (openff.evaluator.backends.dask.DaskLocalCluster method), 210init() (openff.evaluator.backends.dask.DaskLocalCluster method), 210init() (openff.evaluator.backends.dask.DaskDaskDaskDaskDaskDaskDaskDaskDaskDask	init_		ckend	
method), 208 init() (openff.evaluator.backends.dask.DaskLSFBackend method), 211 init() (openff.evaluator.backends.dask.DaskLSFBackend method), 210 init() (openff.evaluator.backends.dask.DaskLocalCluster method), 210 init() (openff.evaluator.backends.dask.DaskDaskDaskDaskDaskDaskDaskDaskDaskDask		method), 207		
method), 208 init() (openff.evaluator.backends.dask.DaskLSFBackend method), 211 init() (openff.evaluator.backends.dask.DaskLSFBackend method), 210 init() (openff.evaluator.backends.dask.DaskLocalCluster method), 210 init() (openff.evaluator.backends.dask.DaskDaskDaskDaskDaskDaskDaskDaskDaskDask	init_	() (openff.evaluator.backends.dask.BaseDaskJob	bQueueBa	ckendod), 164
method), 211			1n1t_	_() (openff.evaluator.aatasets.curation.components.futering.Futer
init() (openff.evaluator.backends.dask.DaskLocalCluster method), 210	init_	() (openff.evaluator.backends.dask.DaskLSFBac	ckend	
method), 210init() (openff.evaluator.backends.dask.DaskPBSBackend method), 213		method), 211	init_	
init() (openff.evaluator.backends.dask.DaskPBSBackend method), 213init() (openff.evaluator.client.BatchMode method), 80init() (openff.evaluator.client.ConnectionOptions method), 80init() (openff.evaluator.client.ConnectionOptions method), 80init() (openff.evaluator.client.EvaluatorClient method), 78init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.RequestOptions method), 84init() (openff.evaluator.client.RequestResult method), 86init() (openff.evaluator.datasets.CalculationSource method), 98init() (openff.evaluator.datasets.MeasurementSource method), 99init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter method), 158init() (openff.evaluator.datasets.curation.components.filtering.Filter.gataseta.curation.components.filtering.Filter.gataseta.curation.components.filtering.Filter.gataseta.curation.components.filtering.Filter.gataseta.curation.compon	init_	() (openff.evaluator.backends.dask.DaskLocalC	luster	
method), 213init() (openff.evaluator.client.BatchMode		method), 210	init_	
init() (openff.evaluator.client.BatchMode method), 80init() (openff.evaluator.client.ConnectionOptions method), 80init() (openff.evaluator.client.ConnectionOptions method), 78init() (openff.evaluator.client.EvaluatorClient method), 78init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.Request method), 84init() (openff.evaluator.client.RequestResult method), 86init() (openff.evaluator.datasets.CalculationSource method), 98init() (openff.evaluator.datasets.MeasurementSource method), 99init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter method), 158init() (openff.evaluator.datasets.curation.components.filtering.Filter method), 168init() (openff.evaluator.datasets.curation.components.filtering.Filter	init_	() (openff.evaluator.backends.dask.DaskPBSBac	ckend	
<pre>init() (openff.evaluator.client.ConnectionOptions     method), 80    init() (openff.evaluator.client.ConnectionOptions         method), 80    init() (openff.evaluator.client.EvaluatorClient         method), 78    init() (openff.evaluator.client.Request method), 82    init() (openff.evaluator.client.Request method), 82    init() (openff.evaluator.client.Request method), 82    init() (openff.evaluator.client.RequestOptions         method), 159    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 159    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 169    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 168    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 169    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 169    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 168    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 168    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 158    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 158    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 158    init() (openff.evaluator.datasets.curation.components.filtering.Filter         method), 158    init() (openff.evaluator.datasets.curation.components.f</pre>			init_	
method), 80 init() (openff.evaluator.client.ConnectionOptions method), 80 init() (openff.evaluator.client.EvaluatorClient method), 78 init() (openff.evaluator.client.EvaluatorClient method), 78 init() (openff.evaluator.client.Request method), 82 init() (openff.evaluator.client.Request method), 82 init() (openff.evaluator.client.Request method), 84 init() (openff.evaluator.client.RequestResult method), 86 init() (openff.evaluator.datasets.CalculationSource method), 98 init() (openff.evaluator.datasets.MeasurementSource method), 99 init() (openff.evaluator.datasets.Calculatior.Source method), 158	init_	() (openff.evaluator.client.BatchMode		
init() (openff.evaluator.client.ConnectionOptions method), 80			init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
method), 80 init() (openff.evaluator.client.EvaluatorClient method), 78 init() (openff.evaluator.client.Request method), 82 init() (openff.evaluator.client.Request method), 82 init() (openff.evaluator.client.RequestOptions method), 84 init() (openff.evaluator.client.RequestResult method), 86 init() (openff.evaluator.datasets.CalculationSource method), 98 init() (openff.evaluator.datasets.MeasurementSource method), 99 init() (openff.evaluator.datasets.MeasurementSource method), 158 init() (openff.evaluator.datasets.curation.components.filtering.Filter method), 158	init			method), 165
init() (openff.evaluator.client.EvaluatorClient method), 78init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.RequestOptions method), 84init() (openff.evaluator.client.RequestResult method), 86init() (openff.evaluator.datasets.CalculationSource method), 98init() (openff.evaluator.datasets.MeasurementSource method), 99init() (openff.evaluator.datasets.PhysicalProperty			init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
method), 78 init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.RequestOptions method), 84 init() (openff.evaluator.client.RequestResult method), 86 init() (openff.evaluator.datasets.CalculationSource method), 98 init() (openff.evaluator.datasets.MeasurementSource method), 99 init() (openff.evaluator.datasets.Calculator.datasets.PhysicalProperty init() (openff.evaluator.datasets.curation.components.filtering.Filter method), 158	init			method), 165
init() (openff.evaluator.client.Request method), 82init() (openff.evaluator.client.RequestOptions method), 84init() (openff.evaluator.client.RequestResult method), 86init() (openff.evaluator.datasets.CalculationSource method), 98init() (openff.evaluator.datasets.MeasurementSource method), 99init() (openff.evaluator.datasets.PhysicalProperty init() (openff.evaluator.datasets.curation.components.filtering.Filter method), 158init() (openff.evaluator.datasets.curation.components.filtering.Filtering.			init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init() (openff.evaluator.client.RequestOptions method), 84	init			method), 159
method), 84 init() (openff.evaluator.client.RequestResult method), 86 init() (openff.evaluator.datasets.CalculationSource method), 98 init() (openff.evaluator.datasets.CalculationSource method), 98 init() (openff.evaluator.datasets.MeasurementSource—init() (openff.evaluator.datasets.curation.components.filtering.Filter method), 158 init() (openff.evaluator.datasets.PhysicalProperty init() (openff.evaluator.datasets.curation.components.filtering.Filterin			init_	$\_()$ (openff.evaluator.datasets.curation.components.filtering.Filter
init() (openff.evaluator.client.RequestResult method), 86				method), 159
method), 86 init() (openff.evaluator.datasets.CalculationSourceinit() (openff.evaluator.datasets.curation.components.filtering.Filter method), 98 init() (openff.evaluator.datasets.MeasurementSourceinit() (openff.evaluator.datasets.curation.components.filtering.Filter method), 99 init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter	init		init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
init() (openff.evaluator.datasets.CalculationSource				method), 169
method), 98 init() (openff.evaluator.datasets.MeasurementSourceinit() (openff.evaluator.datasets.curation.components.filtering.Filter  method), 168 init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter	init	<i>"</i>	init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
init() (openff.evaluator.datasets.MeasurementSourceinit() (openff.evaluator.datasets.curation.components.filtering.Filter method), 99 method), 158init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter		mathad) 00		method), 168
method), 99 init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter	ini+	() (one off evaluator datasets Measurement Source	init_	_() (openff.evaluator.datasets.curation.components.filtering.Filter
init() (openff.evaluator.datasets.PhysicalPropertyinit() (openff.evaluator.datasets.curation.components.filtering.Filter		() (openy.evanunor.aanuseis.measurememsourc mothod) 00	c	method), 158
IIII (() (Openiji.evaliaator.aatasets.i rtysteati roperty	ini+		init_	
method), 93 method), 158				method), 158

\_\_init\_\_() (openff.evaluator.datasets.PhysicalPropertyDatasets.U( (openff.evaluator.datasets.curation.components.filtering.Filtering.

method), 138

*method*), 162

\_\_init\_\_() (openff.evaluator.datasets.curation.components.filming\_film method), 162 method), 154 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filming\_Kilk@BerMieenlieator.datasets.taproom.TaproomDataSet method), 160 method), 147 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filmiing\_Ki)t@PenKfi.evalueSucheduatasets.taproom.TaproomSource method), 160 method), 151 \_\_init\_\_() (openff.evaluator.datasets.curation.components.fi**irai**ing.f(i)teoBexSffièlexluator.datasets.thermoml.ThermoMLDataSet *method*), 166 method), 142 \_\_init\_\_() (openff.evaluator.datasets.curation.component<u>s.filmitrg\_f()</u>te**vBexSfhèlesSaltemfo**rcefield.ForceFieldSource method), 166 method), 181 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filming\_film method), 167 method), 185 \_\_init\_\_() (openff.evaluator.datasets.curation.component<u>s.fi**lmi**ng.Ki)t@**pe**xMfpe**rksSuchemf**wrcefield.ParameterGradient</u> method), 167 *method*), 188 \_\_init\_\_() (openff.evaluator.datasets.curation.component<u>s.fi**imi**ing.f(i)</u>teoperSffeeealhamisforcefield.ParameterGradientKey method), 163 method), 187 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filmiing.fi)te@BenSffeeealhamisfomScfheebhSmirnoffForceFieldSource method), 163 method), 182 \_\_init\_\_\_() (openff.evaluator.datasets.curation.component<u>s.fi**lmi**ring</u>\_Kilt@**pexs(filestathaato**r.forcefield.TLeapForceFieldSource *method*), 170 method), 183 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filming\_KilterByEufffwunderSchenwers.CalculationLayer method), 169 method), 188 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filming\_Ki)tenBenffinpahaatuve.layers.CalculationLayerResult method), 157 method), 189 \_\_init\_\_() (openff.evaluator.datasets.curation.component<u>s.fi**lmi**ring.f(i)te**oBe**Affm**pahaautoeSahern**aCalculationLayerSchema</u> *method*), 157 method), 191 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filming\_KilleoDanfflevalenator.layers.reweighting.ReweightingLayer method), 199 *method*), 156 \_\_init\_\_() (openff.evaluator.datasets.curation.components.filmiing.f()teopanffiewden8tchehayers.reweighting.ReweightingSchema method), 156 method), 201 \_\_init\_\_() (openff.evaluator.datasets.curation.components.friexixly\_fnpenffx8odvuator.layers.simulation.SimulationLayer *method*), 173 method), 196 \_\_init\_\_() (openff.evaluator.datasets.curation.components.friexialv\_16pp@pFnfe8odvSahemlayers.simulation.SimulationSchema *method*), 172 method), 197 \_\_init\_\_() (openff.evaluator.datasets.curation.components.selation.FinapenffineNalpetor.layers.workflow.WorkflowCalculationLayer method), 179 method), 193 \_\_init\_\_() (openff.evaluator.datasets.curation.components<u>.sakrition</u>SelegtDaff@Palutsor.layers.workflow.WorkflowCalculationSchema method), 177 method), 194 \_\_init\_\_() (openff.evaluator.datasets.curation.components.sakaixion.\( \SelectData\( \Brightarrow\) (openff.evaluator.datasets.curation.components.\( \SelectData\( \Brightarrow\) (openff.evaluator.datasets.curation.components.\( \SelectData\( \Brightarrow\) (openff.evaluator.datasets.curation.components.\( \SelectData\( \Brightarrow\) (openff.evaluator.datasets.curation.components.\( \SelectData\( \Brightarrow\) (openff.evaluator.datasets.curation method), 176 method), 100 \_\_init\_\_() (openff.evaluator.datasets.curation.component<u>s.s.abai.tion.Selegt&uffxtvuolous</u>tor.properties.DielectricConstant *method*), 176 method), 107 \_\_init\_\_() (openff.evaluator.datasets.curation.component<u>s.såkrition.Selept&ndff.stvonkvas&alpemp</u>erties.EnthalpyOfMixing *method*), 175 method), 111 \_\_init\_\_() (openff.evaluator.datasets.curation.components.sikition.()) (tepenff.evaluator.properties.EnthalpyOfVaporization method), 177 method), 114 \_\_init\_\_() (openff.evaluator.datasets.curation.components<u>.sakrition</u>.Qu(gptStfftevaluator.properties.ExcessMolarVolume *method*), 178 *method*), 104 \_\_init\_\_\_() (openff.evaluator.datasets.curation.component<u>s.th**arirt**om</u>()rhppet/ffievalaMbDpnaperties.HostGuestBindingAffinity method), 174 *method*), 121 \_\_init\_\_() (openff.evaluator.datasets.curation.components\_therirtom() (openff.evaluallb.DuraySchiees&olvationFreeEnergy *method*), 173 method), 118 \_init\_\_() (openff.evaluator.datasets.curation.workflow.C<u>ura**tia**itWo</u>ftflowenff.evaluator.protocols.analysis.AverageDielectricConstar *method*), 153 method), 297

```
__init__() (openff.evaluator.protocols.analysis.AverageFreeEnirgies() (openff.evaluator.protocols.paprika.analysis.ComputeReference
         method), 303
                                                                   method), 479
__init__() (openff.evaluator.protocols.analysis.AverageObseiwaltle_() (openff.evaluator.protocols.paprika.analysis.ComputeSymmetri
         method), 291
                                                                   method), 474
__init__() (openff.evaluator.protocols.analysis.BaseAverage this trya Dicopenff.evaluator.protocols.paprika.coordinates.AddDummyA
         method), 286
                                                                   method), 443
__init___() (openff.evaluator.protocols.analysis.BaseDeco<u>rrelmirProt</u>0chbpenff.evaluator.protocols.paprika.coordinates.PreparePullC
         method), 313
                                                                    method), 432
__init___() (openff.evaluator.protocols.analysis.Compute<u>Dipoh:Mom</u>(Mopenff.evaluator.protocols.paprika.coordinates.PrepareRelea
         method), 308
                                                                   method), 438
__init__() (openff.evaluator.protocols.analysis.Decorrela<u>teObserva</u>kDe(openff.evaluator.protocols.paprika.restraints.ApplyRestraints
         method), 324
                                                                    method), 464
__init__() (openff.evaluator.protocols.analysis.DecorrelateTinjettor()) (openff.evaluator.protocols.paprika.restraints.GenerateAttach.
         method), 318
                                                                    method), 448
_init__() (openff.evaluator.protocols.coordinates.BuildC<u>oordintates</u>Palahpenlff.evaluator.protocols.paprika.restraints.GeneratePullRe
         method), 329
                                                                    method), 453
__init__() (openff.evaluator.protocols.coordinates.BuildDockailCookJihapesnff.evaluator.protocols.paprika.restraints.GenerateReleas
         method), 341
                                                                   method), 459
_init__() (openff.evaluator.protocols.coordinates.SolvateExixixittgStQdtopenff.evaluator.protocols.reweighting.BaseEvaluateEnergies
         method), 335
                                                                   method), 494
__init__() (openff.evaluator.protocols.forcefield.BaseBuildSixtart__() (openff.evaluator.protocols.reweighting.BaseMBARProtocol
         method), 347
                                                                   method), 500
__init__() (openff.evaluator.protocols.forcefield.BuildLigPaiGirtSys@n(openff.evaluator.protocols.reweighting.ConcatenateObserval
         method), 357
                                                                   method), 490
__init__() (openff.evaluator.protocols.forcefield.BuildSmi<u>rnoffSysterh</u>) (openff.evaluator.protocols.reweighting.ConcatenateTrajector.
         method), 352
                                                                   method), 485
__init__() (openff.evaluator.protocols.forcefield.BuildTLeap$\sqrt{m}init_m() (openff.evaluator.protocols.reweighting.ReweightDielectricCo
         method), 363
                                                                   method), 511
__init__() (openff.evaluator.protocols.gradients.ZeroGradieitsit__() (openff.evaluator.protocols.reweighting.ReweightObservable
         method), 368
                                                                    method), 505
__init__() (openff.evaluator.protocols.groups.ConditionalGranit__() (openff.evaluator.protocols.simulation.BaseEnergyMinimisati
         method), 374
                                                                    method), 517
__init__() (openff.evaluator.protocols.miscellaneous.Add<u>Va</u>luxist__() (openff.evaluator.protocols.simulation.BaseSimulation
         method), 380
                                                                   method), 522
__init__() (openff.evaluator.protocols.miscellaneous.Divide\nhte__() (openff.evaluator.protocols.storage.UnpackStoredSimulationD
                                                                   method), 529
         method), 394
__init__() (openff.evaluator.protocols.miscellaneous.Dumm\dagge\textbf{RictionCol}) (openff.evaluator.protocols.utils.ReweightingProtocols
         method), 409
                                                                    method), 560
__init__() (openff.evaluator.protocols.miscellaneous.FilterSithstanceByRpenff.evaluator.protocols.utils.SimulationProtocols
         method), 404
                                                                   method), 558
__init__() (openff.evaluator.protocols.miscellaneous.Mult<u>ipl</u>xVatue_() (openff.evaluator.protocols.yank.BaseYankProtocol
         method), 390
                                                                    method), 536
__init___() (openff.evaluator.protocols.miscellaneous.Subt<u>raċtVatues</u>() (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
         method), 385
                                                                   method), 541
__init__() (openff.evaluator.protocols.miscellaneous.WeightByMoleFydotpenff.evaluator.protocols.yank.SolvationYankProtocol
         method), 399
                                                                    method), 549
__init__() (openff.evaluator.protocols.openmm.OpenMMEnargyMix()) (septemff.evaluator.server.Batch method), 90
                                                                          (openff.evaluator.server.EvaluatorServer
                                                          __init__()
__init__() (openff.evaluator.protocols.openmm.OpenMMEvaluateEmentyied), 89
         method), 427
                                                          __init__() (openff.evaluator.storage.LocalFileStorage
__init__() (openff.evaluator.protocols.openmm.OpenMMSimulationmethod), 217
                                                          __init__() (openff.evaluator.storage.StorageBackend
         method), 420
 __init__() (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRiPithoel), 215
         method), 469
                                                          __init__() (openff.evaluator.storage.attributes.FilePath
```

	1 1 247		1 1) 266
	method), 247	.,	method), 266
1n1t	() (openff.evaluator.storage.attributes.QueryAttr	ı <u>bu</u> lanıt_	
	method), 254		method), 267
1n1t	() (openff.evaluator.storage.attributes.StorageAt	<i>r<u>ıb</u><b>un</b></i> ıt_	
	method), 253	5.1.1	method), 255
1n1t	() (openff.evaluator.storage.data.BaseSimulation	ı <i>Da</i> tanıt_	
	method), 225		method), 258
init	() (openff.evaluator.storage.data.BaseStoredDat	ainit_	
	method), 219		method), 259
init		<i></i> init_	_() (openff.evaluator.workflow.attributes.BaseMergeBehaviour
	method), 222		method), 281
init		d <u>Da</u> t <b>n</b> it_	_() (openff.evaluator.workflow.attributes.InequalityMergeBehavio
	method), 221		method), 281
init		<i>t<u>a_</u>init_</i>	_() (openff.evaluator.workflow.attributes.InputAttribute
	method), 224		method), 282
init	() (openff.evaluator.storage.data.StoredFreeEne	rgy <b>Dni</b> at_	_() (openff.evaluator.workflow.attributes.MergeBehaviour
	method), 231		method), 281
init		o <u>n D</u> iannait_	_() (openff.evaluator.workflow.attributes.OutputAttribute
	method), 228		method), 283
init	() (openff.evaluator.storage.query.BaseDataQue	<i>r</i> y_init_	_() (openff.evaluator.workflow.schemas.ProtocolGroupSchema
	method), 235		method), 274
init	() (openff.evaluator.storage.query.BaseSimulation	nDinniQu	(openff.evaluator.workflow.schemas.ProtocolReplicator
	method), 239		method), 276
init	() (openff.evaluator.storage.query.ForceFieldQu	<i>e<u>r</u>y</i> init_	_() (openff.evaluator.workflow.schemas.ProtocolSchema
	method), 238		method), 272
init	() (openff.evaluator.storage.query.FreeEnergyDa	a <u>ta<b>Qnė</b>rty</u>	_() (openff.evaluator.workflow.schemas.WorkflowSchema
	method), 245		method), 278
init	() (openff.evaluator.storage.query.SimulationDa	t <u>aQimeit</u> t_	_() (openff.evaluator.workflow.utils.ProtocolPath
	method), 242		method), 284
init	() (openff.evaluator.storage.query.SubstanceQue	<u>ry</u> init_	_() (openff.evaluator.workflow.utils.ReplicatorValue
	method), 236		method), 283
init	() (openff.evaluator.substances.Amount		
	method), 131	Α	
init	() (openff.evaluator.substances.Component	absolut	e_tolerance(openff.evaluator.layers.CalculationLayerSchema
	method), 129		attribute), 191
init	() (openff.evaluator.substances.ExactAmount	absolut	e_tolerance(openff.evaluator.layers.reweighting.ReweightingSc
	method), 132		attribute), 202
init	() (openff.evaluator.substances.MoleFraction	absolut	e_tolerance(openff.evaluator.layers.simulation.SimulationScher
	method), 134		attribute), 198
init	() (openff.evaluator.substances.Substance	absolut	e_tolerance(openff.evaluator.layers.workflow.WorkflowCalcular
	method), 125		attribute), 195
init	() (openff.evaluator.thermodynamics.Thermodyn	aniGState	e site location
	method), 136		(open ff. evaluator. protocols. coordinates. BuildDockedCoordinates
init	() (openff.evaluator.utils.observables.Observabl	e	attribute), 342
	method), 566	add com	ponent() (openff.evaluator.substances.Substance
init	() (openff.evaluator.utils.observables.Observabl	eArray	method), 127
	method), 567	add con	dition() (openff.evaluator.protocols.groups.ConditionalGroup
init	() (openff.evaluator.utils.observables.Observabl	eFrame	method), 376
	method), 569		operties() (openff.evaluator.datasets.PhysicalPropertyDataSet
init	() (openff.evaluator.utils.observables.Observabl	eType	method), 139
	method), 568	• •	operties()(openff.evaluator.datasets.taproom.TaproomDataSet
init		add_pro	method), 148
	method), 261	add nro	perties() (openff.evaluator.datasets.thermoml.ThermoMLDataS
init	() (openff.evaluator.workflow.ProtocolGraph	~~~pro	method), 143
	- ···		, , = -=

add_protocols() (openff.evaluator.protocols.groups.ConditionalGrampibute), 376  method), 376  allow_merging (openff.evaluator.protocols.miscellaneous.AddValues
add_protocols() (openff.evaluator.workflow.ProtocolGraph attribute), 381
method), 266 allow_merging (openff.evaluator.protocols.miscellaneous.DivideValue
add_protocols() (openff.evaluator.workflow.ProtocolGroup attribute), 396
method), 268 allow_merging (openff.evaluator.protocols.miscellaneous.DummyProtoco
add_schema() (openff.evaluator.client.RequestOptions attribute), 411
method), 84 allow_merging (openff.evaluator.protocols.miscellaneous.FilterSubstance
add_workflows() (openff.evaluator.workflow.WorkflowGraph attribute), 406
method), 258 allow_merging (openff.evaluator.protocols.miscellaneous.MultiplyValue
AddDummyAtoms (class in attribute), 391 openff.evaluator.protocols.paprika.coordinates), allow_merging(openff.evaluator.protocols.miscellaneous.SubtractValues
443 attribute), 386
AddValues (class in openff.evaluator.protocols.miscellaneouss),low_merging (openff.evaluator.protocols.miscellaneous.WeightByMolelatribute), 401
allow_gpu_platforms allow_merging(openff.evaluator.protocols.openmm.OpenMMEnergyMinu
(openff.evaluator.protocols.openmm.OpenMMSimulation attribute), 416
attribute), 421 allow_merging(openff.evaluator.protocols.openmm.OpenMMEvaluateEn
allow_gpu_platforms attribute), 428
(openff.evaluator.protocols.simulation.BaseSimulatlorow_merging (openff.evaluator.protocols.openmm.OpenMMSimulation attribute), 525 attribute), 421
allow_merging (openff.evaluator.protocols.analysis.Average\(\mathbb{Limble}\) in the distribute), 299 attribute), 470
allow_merging(openff.evaluator.protocols.analysis.AveragelFrowEmerging(openff.evaluator.protocols.paprika.analysis.ComputeRefe
attribute), 304 attribute), 480
allow_merging (openff.evaluator.protocols.analysis.Average\(\mathbb{Q}\) lowermetriging (openff.evaluator.protocols.paprika.analysis.ComputeSymattribute), 293  attribute), 475
allow_merging (openff.evaluator.protocols.analysis.BaseAndrlage Obergiadg (openff.evaluator.protocols.paprika.coordinates.AddDumi
attribute), 288 attribute), 444
allow_merging(openff.evaluator.protocols.analysis.BaseDækhowelmerging(obenff.evaluator.protocols.paprika.coordinates.PrepareP
attribute), 314 attribute), 434
allow_merging (openff.evaluator.protocols.analysis.Compath Dipolw Aging (openff.evaluator.protocols.paprika.coordinates.PrepareR attribute), 309 attribute), 439
$\textbf{allow\_merging} (\textit{openff.evaluator.protocols.analysis.Decor{\textbf{red}lumQ} \textit{lneargibly}} (openff.evaluator.protocols.paprika.restraints.ApplyRestrations and \textit{openff.evaluator.protocols.paprika.restraints.ApplyRestrations and \textit{openff.evaluator.protocols.paprika.protocols$
attribute), 325 attribute), 465
allow_merging (openff.evaluator.protocols.analysis.Decor <b>aellutwTrmjercginyg</b> (openff.evaluator.protocols.paprika.restraints.GenerateAttattribute), 320 attribute), 449
allow_merging(openff.evaluator.protocols.coordinates.BuillCoordinates.Buil
attribute), 331 attribute), 454
allow_merging(openff.evaluator.protocols.coordinates.BuillDowkmdCgindjhapenff.evaluator.protocols.paprika.restraints.GenerateRe
attribute), 343 attribute), 460
$\textbf{allow\_merging} (\textit{openff.evaluator.protocols.coordinates.Sobblicowine} \textbf{winergSing} (\textit{inpre} \textit{nff.evaluator.protocols.reweighting.BaseEvaluateEnergy}) and \textit{inpre} \textit{nff.evaluator.protocols.reweighting.BaseEvaluateEnergy}) and \textit{inpre} \textit{nff.evaluator.protocols.reweighting.BaseEvaluateEnergy}) and \textit{inpre} \textit{nff.evaluator.protocols.coordinates.SobblicowineergSing} (\textit{inpre} \textit{nff.evaluator.protocols.coordinates.SobblicowineergSing}) and \textit{inpre} \textit{nff.evaluator.protocols.coordinates.SobblicowineergSing} (\textit{nff.evaluator.protocols.coordinates.SobblicowineergSing)} (\textit{nff.evaluator.protocols.coordinates.SobblicowineergSing)} (\textit{nff.evaluator.protocols.coordinates.SobblicowineergSing)} (\textit{nff.evaluator.protocols.coordinates.SobblicowineergSing)} (\textit{nf.evaluator.protocols.coordinates.SobblicowineergSing)} (nf.evaluator.protocols.coordinates.SobblicowineergSing)$
attribute), 336 attribute), 496
allow_merging (openff.evaluator.protocols.forcefield.BaseRhildNymernging (openff.evaluator.protocols.reweighting.BaseMBARProtocols.tribute), 348 attribute), 502
allow_merging (openff.evaluator.protocols.forcefield.Buildalig.Bur@iensgistegn(openff.evaluator.protocols.reweighting.ConcatenateObse attribute), 358 attribute), 491
allow_merging(openff.evaluator.protocols.forcefield.Build <b>&amp;hlionoffi&amp;rsging</b> (openff.evaluator.protocols.reweighting.ConcatenateTrajec
attribute), 353 attribute), 486
allow_merging (openff.evaluator.protocols.forcefield.BuildattleonpSmereging (openff.evaluator.protocols.reweighting.ReweightDielectric
attribute), 364 attribute), 513

attribute), 370

 ${\tt allow\_merging} \ (open \textit{ff.evaluator.protocols.gradients.Zero} \textbf{\textit{Gkhdow\_mmerging}} \ (open \textit{ff.evaluator.protocols.reweighting.ReweightObservations}) \ (open \textit{ff.evaluator.protocols.gradients.Zero} \textbf{\textit{Gkhdow\_mmerging}} \ (open \textit{ff.evaluator.protocols.reweighting.ReweightObservations}) \ (open \textit{ff.evaluator.protocols.gradients.Zero} \textbf{\textit{Gkhdow\_mmerging}} \ (open \textit{ff.evaluator.protocols.gradients.Zero} \ \textbf{\textit{Gkhdow\_mmerging}} \ (open \textit{ff.evaluator.protocols.gradients.Zero} \ \textbf{\textit{Ghdow\_mmerging}} \ (open \textit{ff.evaluator.protocols.gradients.gradients.Zero} \ \textbf{\textit{Ghdow\_mmerging}} \ (open \textit{ff.evaluator.protocols.gradients.gradien$ 

 $\textbf{allow\_merging} (open \textit{ff.evaluator.protocols.groups.} Conditi \textbf{alld blive one} \textbf{pr} \textbf{ging} (open \textit{ff.evaluator.protocols.simulation.} Base \textit{Energy Minimization} \textbf{groups.} Condition \textbf{groups.} \textbf{grou$ 

attribute), 507

attribute), 519

attribute), 526

attribute), 531

attribute), 538

attribute), 545

attribute), 553

tribute), 262

attribute), 270

Amount (class in openff.evaluator.substances), 131

AnalyzeAPRPhase (class in method), openff.evaluator.protocols.paprika.analysis), apply() (openff.evaluator.protocols.paprika.analysis), apply() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.datasets.curation.components.composition) (Composition class method), 180 method), apply() (openff.evaluator.datasets.curation.components.Cuapply(Complements) class method), 152 method), apply() (openff.evaluator.datasets.curation.components.filtapinly).FitiepBiyChang	aluator.storage.query.FreeEnergyDataQuery 245 aluator.storage.query.SimulationDataQuery 243 <b>whssDensitylflata</b> schemas.ProtocolReplicator 277 <b>r()</b> (openff.evaluator.protocols.analysis.AverageDielect
openff.evaluator.protocols.paprika.analysis), apply() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.datasets.curation.components.components.components.components.components.components.components.components.curation.components.curation.components.curation.components.curation.components.curation.components.curation.components.filtapiplyFittepBiyChand.gapply() (openff.evaluator.datasets.curation.components.filtapiplyFittepBiyChand.gapply() (openff.evaluator.datasets.curation.gapply() (openff.evaluator.datasets.curation.gapply() (openff.evaluator.datasets.curation.gapply() (openff.evaluator.datasets.curation.gapply() (openff.evaluator.datasets.curation.gapply() (openff.evaluator.datasets.curation.gapp	aluator.storage.query.FreeEnergyDataQuery 245 aluator.storage.query.SimulationDataQuery 243 <b>whssDensitylflata</b> schemas.ProtocolReplicator 277 <b>r()</b> (openff.evaluator.protocols.analysis.AverageDielect
469 method), append_uuid() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.workflow.utils.ProtocolPathply() (openff.evaluator.datasets.curation.components.coapplyid) (openff.evaluator.datasets.curation.components.Cuapply() (openff.evaluator.datasets.curation.components.Cuapply() (openff.evaluator.datasets.curation.components.filtapinlyFittepBiyCauplicattor.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.curation.datasets.cu	245 aluator.storage.query.SimulationDataQuery 243 <b>wkssDerssinJfflata</b> schemas.ProtocolReplicator 277 <b>r()</b> (openff.evaluator.protocols.analysis.AverageDielect
append_uuid() (openff.evaluator.workflow.utils.ProtocolPatpply() (openff.evaluator.workflow.utils.ProtocolPatpply() (openff.evaluator.datasets.curation.components.coappply() (openff.evaluator.datasets.curation.components.Coappply() (openff.evaluator.datasets.curation.components.Cuapply() (openff.evaluator.datasets.curation.components.filtappplyFittepByCoapplyCoapply() (openff.evaluator.datasets.curation.components.filtappplyFittepByCoapplyCoapply() (openff.evaluator.datasets.curation.components.filtappplyFittepByCoapply() (openff.evaluator.datasets.curation.components.filtappplyFittepByCoapply() (openff.evaluator.datasets.curation.components.filtappplyFittepByCoapply() (openff.evaluator.datasets.curation.components.filtappplyFittepByCoapply() (openff.evaluator.datasets.curation.components.filtappply() (openff.evaluator.datasets.curation.components.curation.datasets.curation.datasets.curation.d	aluator.storage.query.SimulationDataQuery 243 <b>whssDerssitylflata</b> schemas.ProtocolReplicator 277 <b>r</b> () (openff.evaluator.protocols.analysis.AverageDielect
method), 285 method), apply() (openff.evaluator.datasets.curation.components.coapplyici) Coppenffiex class method), 180 method), apply() (openff.evaluator.datasets.curation.components.Cuapplyicioapplyicioanto class method), 152 method), apply() (openff.evaluator.datasets.curation.components.filtapplyyFittepByChang	243 <b>whssDenssitylflata</b> schemas.ProtocolReplicator 277 <b>r()</b> (openff.evaluator.protocols.analysis.AverageDielect
apply() (openff.evaluator.datasets.curation.components.coapplyici) (copposeff.Exclass method), 180 method), apply() (openff.evaluator.datasets.curation.components.Cuapplyicipentoclass method), 152 method), apply() (openff.evaluator.datasets.curation.components.filtapinkyFittepRicChange	<b>wksstDerwsbtylflata</b> schemas.ProtocolReplicator 277 <b>r()</b> (openff.evaluator.protocols.analysis.AverageDielect
class method), 180 method), apply() (openff.evaluator.datasets.curation.components.CuappibyComploireatto class method), 152 method), apply() (openff.evaluator.datasets.curation.components.filtapiplyFittepByChung	277 r() (openff.evaluator.protocols.analysis.AverageDielect
apply() (openff.evaluator.datasets.curation.components.CuapplyComplinated class method), 152 method), apply() (openff.evaluator.datasets.curation.components.filtapinkyFitteplkyCatao	r() (openff.evaluator.protocols.analysis.AverageDielect
class method), 152 method), apply() (openff.evaluator.datasets.curation.components.filtapinly/Fittep/BixChano	
apply() (openff.evaluator.datasets.curation.components.filtappinbyEithepRichtto	
class method), 164 method),	
apply() (openff.evaluator.datasets.curation.components.filtapingyEittepByEitepB	
class method), 161 method),	
apply() (openff.evaluator.datasets.curation.components.filtappylyFirtepByEavo	
class method), 172 method),	
apply() (openff.evaluator.datasets.curation.components.filtappylyFirtepBykatio	
class method), 165 method),	
apply() (openff.evaluator.datasets.curation.components.filtappplyEpitepBykbate	
class method), 159 method),	
apply() (openff.evaluator.datasets.curation.components.filtapplyyEittepByNava class method), 169 method),	
apply() (openff.evaluator.datasets.curation.components.filtapinkyFittepByPatso	
class method), 158 method),	* * **
apply() (openff.evaluator.datasets.curation.components.filtapinkyFittepByPatpo	
class method), 162 method),	
apply() (openff.evaluator.datasets.curation.components.filtapinkyFittepByRate	
class method), 160 method),	
apply() (openff.evaluator.datasets.curation.components.filtapingyFittepBixSatila	
class method), 166 method),	
apply() (openff.evaluator.datasets.curation.components.filtapinlyFittepRixSatist	
class method), 168 method),	
apply() (openff.evaluator.datasets.curation.components.filtapinlyFittepRyState	
class method), 163 method),	
apply() (openff.evaluator.datasets.curation.components.filtapplyEithepBySalts	<b>ጀሲ</b> ንፈውpenff.evaluator.protocols.forcefield.BuildSmirnof
class method), 170 method),	353
apply() (openff.evaluator.datasets.curation.components.filtapinly/Eithep By Eartip	<b>Enathor</b> enff.evaluator.protocols.forcefield.BuildTLeapSy

class method), 157

class method), 156

class method), 173

class method), 177

class method), 176

class method), 174

class method), 153

apply() (openff.evaluator.storage.query.BaseSimulationDataQuery

method), 235

allow\_merging (openff.evaluator.protocols.simulation.Basaspiployl@picopenff.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\polor()) (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

method), 364	<i>method</i> ), 513
apply_replicator() (openff.evaluator.protocols.gradientapplyGreeners)	apkitator() (openff.evaluator.protocols.reweighting.ReweightO
method), 370	method), 507
apply_replicator() (openff.evaluator.protocols.groups.Capplition	elGricaptor() (openff.evaluator.protocols.simulation.BaseEnergy
method), 376	method), 519
${\tt apply\_replicator()} \ (\textit{openff.evaluator.protocols.miscella} \\ {\tt apply\_Ad} \\ {\tt def}$	dpulueator() (openff.evaluator.protocols.simulation.BaseSimulat
method), 381	method), 526
apply_replicator() (openff.evaluator.protocols.miscellaapply_pri	eipleValueor() (openff.evaluator.protocols.storage.UnpackStoredSi
method), 396	method), 531
apply_replicator() (openff.evaluator.protocols.miscellaapplsy_national protocols.miscellaapplsy_national pro	<b>ephyRaxior(d)</b> (openff.evaluator.protocols.yank.BaseYankProtocol
method), 411	method), 538
apply_replicator() (openff.evaluator.protocols.miscellaapplsyEid	<b>ድፓ Σ i b stuar &amp; B ଏ R p k</b> aff. evaluator. protocols. yank. Ligand Receptor Ya
method), 406	method), 545
apply_replicator() (openff.evaluator.protocols.miscellaapplsyMn	
method), 391	method), 553
apply_replicator() (openff.evaluator.protocols.miscellanpplsySml	
method), 386	method), 264
apply_replicator()(openff.evaluator.protocols.miscellanepplsyWe	
method), 401	method), 271
apply_replicator() (openff.evaluator.protocols.openmmapplnyMth	· · · · · · · · · · · · · · · · · · ·
method), 416	attribute), 544
apply_replicator() (openff.evaluator.protocols.openmmMpplnyRe	
method), 428	openff.evaluator.protocols.paprika.restraints),
apply_replicator() (openff.evaluator.protocols.openmm.OpenMM	
	d_residue_names
apply_replicator() (openff.evaluator.protocols.paprika.analysis.A	
method), 471	attribute), 331
apply_replicator() (openff.evaluator.protocols.paprika.asslygice	
method), 481	(openff.evaluator.protocols.coordinates.SolvateExistingStructure
apply_replicator() (openff.evaluator.protocols.paprika.analysis.	
	lambdas ( <i>openff.evaluator.protocols.paprika.restraints.GenerateA</i>
apply_replicator() (openff.evaluator.protocols.paprika.coordinat	
	lambdas (openff.evaluator.protocols.paprika.restraints.GenerateP
apply_replicator() (openff.evaluator.protocols.paprika.coordinat	
	te (class in openff.evaluator.attributes), 564
apply_replicator() (openff.evaluator.protocols.paprika.Adamlibus	
method), 439	564
apply_replicator() (openff.evaluator.protocols.paprika.Accentiges	
	openff.evaluator.protocols.analysis), 297
apply_replicator() (openff.evaluator.protocols.paprika. **Asseming**	
method), 449	openff.evaluator.protocols.analysis), 303
apply_replicator() (openff.evaluator.protocols.paprika. Assemaigns	
	openff.evaluator.protocols.analysis), 291
${\tt apply\_replicator()} \ (open \textit{ff.evaluator.protocols.paprika.restraints})$	.GenerateReleaseRestraints
<i>method</i> ), 460	
apply_replicator() (openff.evaluator.protocols.reweightings Base	Frageobservaire (class in
method), 496	openff.evaluator.protocols.analysis), 286
apply_replicator() (openff.evaluator.protocols.reweighting Bross	YBAR Entocol (class in
method), 502	openff.evaluator.protocols.forcefield), 347
apply_replicator() (openff.evaluator.protocols.reweightings Cons	REBUCKEALISET VALUES (class in
method), 491	openff.evaluator.backends.dask), 207
apply_replicator() (openff.evaluator.protocols.reweightings [page]	atomato Tuai estoniaa
	RF696CHEGEBACKENd (class in
method), 486	หรือขั้งเอเนยมัลckend (class in openff.evaluator.backends.dask), 208

BaseDataQuery	(class	in	(openff.evaluator.protocols.analysis.AverageObservable
	torage.query), 235		attribute), 294
BaseDecorrelateProtoco	,	in	bootstrap_sample_size
	protocols.analysis), 313		(openff.evaluator.protocols.analysis.BaseAverageObservable
BaseEnergyMinimisation	,	in	attribute), 287
	protocols.simulation), 517		bootstrap_uncertainties
BaseEvaluateEnergies	(class	in	$(open {\it ff.evaluator.protocols.reweighting.} Base MBAR {\it Protocol}$
	protocols.reweighting), 494		attribute), 501
BaseMBARProtocol	(class		bootstrap_uncertainties
	protocols.reweighting), 500	0	$(open \emph{ff}. evaluator. protocols. reweighting. Reweight Dielectric Constant auch auch des finite des finite$
BaseMergeBehaviour	(class	in	attribute), 512
openff.evaluator.w	vorkflow.attributes), 281		bootstrap_uncertainties
BaseSimulation	(class	in	$(open {\it ff.evaluator.protocols.reweighting.} Reweight Observable$
openff.evaluator.p	protocols.simulation), 522		attribute), 507
BaseSimulationData	(class	in	$\verb box_aspect_ratio   (open ff.evaluator.protocols.coordinates.Build Coordinates. Build Coordinates. Buil$
openff.evaluator.si	torage.data), 225		attribute), 330
BaseSimulationDataQue	ry (class	in	$\verb box_aspect_ratio   (open ff.evaluator.protocols.coordinates.Solvate Existing Solvate $
openff.evaluator.si	torage.query), 239		attribute), 337
BaseStoredData	(class	in	BuildCoordinatesPackmol (class in
openff.evaluator.si	torage.data), 219		openff.evaluator.protocols.coordinates), 329
BaseYankProtocol	(class	in	BuildDockedCoordinates (class in
openff.evaluator.p	protocols.yank), 536		openff.evaluator.protocols.coordinates), 341
Batch (class in openff.evalu	uator.server), 90		BuildDockedCoordinates.ActivateSiteLocation
batch_mode (openff.evalua	itor.client.RequestOptions	at-	(class in openff.evaluator.protocols.coordinates),
tribute), 84			342
BatchMode (class in openff.	Levaluator.client), 80		BuildLigParGenSystem (class in
beta (openff.evaluator.therm	nodynamics.Thermodynai	micSta	ate openff.evaluator.protocols.forcefield), 357
property), 137			BuildLigParGenSystem.WaterModel (class in
bootstrap()	(in mod	dule	openff.evaluator.protocols.forcefield), 358
openff.evaluator.u	tils.observables), 571		BuildSmirnoffSystem (class in
<pre>bootstrap_cycles(openf)</pre>	f.evaluator.protocols.anal	ysis.A	verageFree <b>&amp;pærffies</b> aluator.protocols.forcefield), 352
attribute), 304			BuildTLeapSystem (class in
bootstrap_iterations			openff.evaluator.protocols.forcefield), 363
(openff.evaluator.p	protocols.analysis.Average	eDiele	e <b>BruicKdWisearp</b> System.ChargeBackend ( <i>class in</i>
attribute), 299			openff.evaluator.protocols.forcefield), 364
bootstrap_iterations			BuildTLeapSystem.WaterModel (class in
(openff.evaluator. <sub>l</sub>	protocols.analysis.Average	eObse	ervable openff.evaluator.protocols.forcefield), 364
attribute), 294			
bootstrap_iterations			C
(openff.evaluator. <sub>l</sub>	protocols.analysis.BaseAv	erage	Carrighte_aqueous_ionic_mole_fraction()
attribute), 287			(openff.evaluator.substances.Substance static
bootstrap_iterations			method), 128
	protocols.reweighting.Bas	seMBA	ARRICATED (in module
attribute), 501			openff.evaluator.layers), 192
bootstrap_iterations			calculation_layers (openff.evaluator.client.RequestOptions
(openff.evaluator.)	protocols.reweighting.Rew	veight	Dielectric Ganstant) 84
attribute), 513		Ü	calculation_schemas
bootstrap_iterations			(openff.evaluator.client.RequestOptions at-
	protocols.reweighting.Rew	veight	Observableribute) 84
attribute), 507		9	CalculationBackend (class in
bootstrap_sample_size			openff.evaluator.backends), 203
	protocols.analysis.Average	eDiele	ectification (class in openff.evaluator.layers),
attribute), 299	- 0		188
bootstrap_sample_size			

- CalculationLayerResult (class in can\_merge() (openff.evaluator.protocols.openmm.OpenMMEnergyMinimi openff.evaluator.layers), 189 method), 416
- CalculationLayerSchema (class in can\_merge() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnerge openff.evaluator.layers), 191 method), 428
- CalculationSource (class in can\_merge() (openff.evaluator.protocols.openmm.OpenMMSimulation openff.evaluator.datasets), 98 in can\_merge() (openff.evaluator.protocols.openmm.OpenMMSimulation method), 422
- can\_merge() (openff.evaluator.protocols.analysis.AverageDialectering@fi3t(openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhomethod), 299 method), 471
- can\_merge() (openff.evaluator.protocols.analysis.AverageFranEmergie() (openff.evaluator.protocols.paprika.analysis.ComputeReferent method), 305 method), 481
- can\_merge() (openff.evaluator.protocols.analysis.Average@lase\_method), 294 (openff.evaluator.protocols.paprika.analysis.ComputeSymmethod), 476
- can\_merge() (openff.evaluator.protocols.analysis.BaseAverage() (openff.evaluator.protocols.paprika.coordinates.AddDummyzmethod), 289 method), 445
- can\_merge() (openff.evaluator.protocols.analysis.BaseDeconnelmePgenOrOpenff.evaluator.protocols.paprika.coordinates.PreparePull method), 315 method), 434
- can\_merge() (openff.evaluator.protocols.analysis.ComputeDip\_oheMge()\_1(openff.evaluator.protocols.paprika.coordinates.PrepareRele method), 310 method), 439
- can\_merge() (openff.evaluator.protocols.analysis.DecorrelateQhergel@eopenff.evaluator.protocols.paprika.restraints.ApplyRestraint method), 325 method), 466
- can\_merge() (openff.evaluator.protocols.analysis.DecorrelateTimjerge(y) (openff.evaluator.protocols.paprika.restraints.GenerateAttac method), 320 method), 450
- can\_merge() (openff.evaluator.protocols.coordinates.BuildCam\_rdinxgesPa(uppenff.evaluator.protocols.paprika.restraints.GeneratePullR method), 332 method), 455
- can\_merge() (openff.evaluator.protocols.coordinates.Buildencemetre () (openff.evaluator.protocols.paprika.restraints.GenerateReleanuethod), 344 method), 461
- can\_merge() (openff.evaluator.protocols.coordinates.Solva@Evimerg@f() (toprenff.evaluator.protocols.reweighting.BaseEvaluateEnergimethod), 337

  method), 497
- can\_merge() (openff.evaluator.protocols.forcefield.BaseBuildfsymerage() (openff.evaluator.protocols.reweighting.BaseMBARProtocol method), 349 method), 502
- 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.BuildSnointoff(Exagen()) (openff.evaluator.protocols.reweighting.ConcatenateTrajecto method), 354 method), 487
- can\_merge() (openff.evaluator.protocols.forcefield.BuildTl@apSmetege() (openff.evaluator.protocols.reweighting.ReweightDielectricC method), 365 method), 513
- can\_merge() (openff.evaluator.protocols.gradients.ZeroGradia\_mmerge() (openff.evaluator.protocols.reweighting.ReweightObservable method), 371 method), 507
- can\_merge() (openff.evaluator.protocols.groups.ConditionalGivomerge() (openff.evaluator.protocols.simulation.BaseEnergyMinimisation.method), 377

  merge() (openff.evaluator.protocols.miscellaneous Additalmerge() (openff.evaluator.protocols.simulation.BaseSimulat
- can\_merge() (openff.evaluator.protocols.miscellaneous.Add\u00e4nlumerge() (openff.evaluator.protocols.simulation.BaseSimulation method), 382 method), 526
- can\_merge() (openff.evaluator.protocols.miscellaneous.DividhVmleege() (openff.evaluator.protocols.storage.UnpackStoredSimulation.

  method), 396

  method), 532
- can\_merge() (openff.evaluator.protocols.miscellaneous.DuranyRerrge() (openff.evaluator.protocols.yank.BaseYankProtocol method), 411 method), 539
- can\_merge() (openff.evaluator.protocols.miscellaneous.FilterSuchertnee(B)(Rplenff.evaluator.protocols.yank.LigandReceptorYankProtocols.yank.digandReceptorYankProtoco
- can\_merge() (openff.evaluator.protocols.miscellaneous.MudtiplyNertge() (openff.evaluator.protocols.yank.SolvationYankProtocol method), 391 method), 554
- can\_merge() (openff.evaluator.protocols.miscellaneous.Sukranctheriges() (openff.evaluator.workflow.Protocol method), 387 (openff.evaluator.workflow.Protocol
- can\_merge() (openff.evaluator.protocols.miscellaneous.WeightByAfoleF)a(aipunff.evaluator.workflow.ProtocolGroup method), 401 method), 269

${\tt capitalize()} \ (open {\it ff. evaluator. storage. attributes. File Particle Particle}) \ (open {\it ff. evaluator. storage. attributes. File Particle Part$	th 129		
method), 249	component_roles(openff.evaluate	r.protocols.misce	ellaneous.FilterSubsta
${\tt casefold()} \ (open \textit{ff.evaluator.storage.attributes.FilePath}$	attribute), 405		
method), 249	components (openff.evaluator.subs	stances.Substance	e at-
<pre>center() (openff.evaluator.storage.attributes.FilePath</pre>	tribute), 126		
method), 249	components_only(openff.evaluate	r.storage.query.S	SubstanceQuery
center_solute_in_box	attribute), 237	0 1 1	~ .
(openff.evaluator.protocols.coordinates.SolvateE		(class	in
attribute), 336	openff.evaluator.protocols	*	
charge_backend (openff.evaluator.protocols.forcefield.Bu		(class	in
attribute), 364	openff.evaluator.protocols	`	s) <b>.</b>
checkpoint_frequency	479	T - T	,,
(openff.evaluator.protocols.openmm.OpenMMSi		class	in
attribute), 422	openff.evaluator.backends		
checkpoint_frequency	ComputeResources.GPUToolkit		in
(openff.evaluator.protocols.simulation.BaseSimu	<del>-</del>	*	· · ·
attribute), 524	ComputeSymmetryCorrection	(class	in
checkpoint_interval		`	
	openff.evaluator.protocols col 474	.paprika.anatysis	6),
(openff.evaluator.protocols.yank.BaseYankProto		(-1	ž
attribute), 537	ConcatenateObservables	(class	in
checkpoint_interval	openff.evaluator.protocols		
(openff.evaluator.protocols.yank.LigandReceptor		(class	in
attribute), 546	openff.evaluator.protocols		
checkpoint_interval	•	class	in
(open ff. evaluator. protocols. yank. Solvation Yank Facilities and the protocol solution for the protocol solution of			
attribute), 554	ConditionalGroup.Condition	(class	in
$\verb clear()  (open ff. evaluator. utils. observables. Observable Francisco   Observables   Observabl$			
method), 570	ConditionalGroup.Condition.		in
<pre>clear_gradients() (openff.evaluator.utils.observables.</pre>			
method), 567	conditions (openff.evaluator.proto	ocols.groups.Con	ditionalGroup
<pre>clear_gradients() (openff.evaluator.utils.observables.</pre>	ObservableAuttribute), 376		
method), 568	confidence_intervals		
<pre>clear_gradients() (openff.evaluator.utils.observables.</pre>	Observable <b>Kognene</b> ff.evaluator.protocol	s.analysis.Averag	geFreeEnergies
method), 570	attribute), 304		
complex_coordinate_path	connection_options (openff.eve	aluator.client.Red	quest
(openff.evaluator.protocols.paprika.restraints.Ge			•
attribute), 449	ConnectionOptions (class in op	enff.evaluator.cli	ent),
complex_coordinate_path	80		
(openff.evaluator.protocols.paprika.restraints.Ge	en@ <b>come@nt/RecersiDe</b> nsityData	(class	in
attribute), 455	openff.evaluator.datasets.c	•	ents.conversion).
complex_electrostatic_lambdas	180		,,
(openff.evaluator.protocols.yank.LigandReceptor	· <i>YanhRentbEnk</i> essDensityDataSche	ema (class	in
attribute), 544	openff.evaluator.datasets.c		
complex_file_path(openff.evaluator.protocols.paprika.		лишт.сотрон	inis.conversion),
attribute), 434	coordinate_file_name		
complex_file_path(openff.evaluator.protocols.paprika.		skora StoredSimul	ationData
attribute), 440	attribute), 229	ade sci. Storeasimu	anonDaia
complex_steric_lambdas	coordinate_file_path		
		s acondinates Du	ildCoordinatas Daokm
(openff.evaluator.protocols.yank.LigandReceptor		s.cooramates.Bu	naCooramatesPackm
attribute), 544	attribute), 331		
Component (class in openff.evaluator.substances), 129	coordinate_file_path	a a a a u Jim - / C i	hanta Eniction - Ct
component (openff.evaluator.protocols.miscellaneous.Wei		s.coorainates.Sol	vateExistingStructure
attribute), 400	attribute), 337		
Component.Role (class in openff.evaluator.substances),	coordinate_file_path		

(openff.evaluator.protocols.forcefield.BaseBuildSy <b>st</b> a attribute), 348	ata_class() (openff.evaluator.storage.query.FreeEnergyDataQuery class method), 245
	ata_class() (openff.evaluator.storage.query.SimulationDataQuery
(openff.evaluator.protocols.forcefield.BuildLigParG	
	ata_to_store(openff.evaluator.layers.CalculationLayerResult
coordinate_file_path	attribute), 190
	Stratento_store (openff.evaluator.workflow.WorkflowResult
attribute), 354	attribute), 260
	ecorrelateObservables (class in
(openff.evaluator.protocols.forcefield.BuildTLeapSy.	· ·
	ecorrelateTrajectory (class in
coordinate_file_path	openff.evaluator.protocols.analysis), 318
(openff.evaluator.protocols.storage.UnpackStoredStored	
attribute), 531	(openff.evaluator.properties. Host Guest Binding Affinity
copy() (openff.evaluator.workflow.utils.ProtocolPath	class method), 122
method), 285	efault_request_options()
count() (openff.evaluator.storage.attributes.FilePath method), 249	(openff.evaluator.client.EvaluatorClient static method), 78
count_exact_amount (openff.evaluator.protocols.coordinate	· · · · · · · · · · · · · · · · · · ·
attribute), 330	(openff.evaluator.properties.Density static
<pre>count_exact_amount(openff.evaluator.protocols.coordinate</pre>	
	efault_reweighting_schema()
CurationComponent (class in	(openff.evaluator.properties.DielectricConstant
openff.evaluator.datasets.curation.components),	static method), 109
152 de	efault_reweighting_schema()
CurationComponentSchema (class in	(openff.evaluator.properties.EnthalpyOfMixing
openff.evaluator.datasets.curation.components),	class method), 112
	efault_reweighting_schema()
CurationWorkflow (class in	(openff.evaluator.properties. Enthalpy Of Vaporization
$open {\it ff. evaluator.} datasets. curation. work {\it flow}),$	class method), 116
	efault_reweighting_schema()
CurationWorkflowSchema (class in	(openff.evaluator.properties.ExcessMolarVolume
openff.evaluator.datasets.curation.workflow),	class method), 105
	efault_simulation_schema()
current_iteration(openff.evaluator.protocols.groups.Con attribute), 376	method), 101
$\verb cutoff  (open ff. evaluator. force field. LigPar Gen Force Field So \verb decoration )    Continuous field of the continuous f$	
	(openff. evaluator. properties. Die lectric Constant
$\verb"cutoff" (open ff. evaluator. force field. TLeap Force Field Source") \\$	static method), 108
property), 184 de	efault_simulation_schema()
D	(openff.evaluator.properties.EnthalpyOfMixing
	class method), 112
	efault_simulation_schema()
openff.evaluator.backends.dask), 210	(openff.evaluator.properties.EnthalpyOfVaporization
DaskLSFBackend (class in	static method), 115
openy, evaluator such en as a custo, , 211	efault_simulation_schema()
DaskPBSBackend (class in	class method), 105
openff.evaluator.backends.dask), 213	
data_class() (openff.evaluator.storage.query.BaseDataQte	(openff.evaluator.properties.SolvationFreeEnergy
class method), 235 data_class() (openff.evaluator.storage.query.BaseSimulation	
class method), 241	efault_storage_query() (in module
$\verb data_class()  (open ff. evaluator. storage. query. Force Field Que$	openff.evaluator.layers.reweighting), 203
class method), 238	efault_unit() (openff.evaluator.datasets.PhysicalProperty

- class method), 94 property), 382 default\_unit() (openff.evaluator.properties.Density dependencies (openff.evaluator.protocols.miscellaneous.DivideValue property), 397 class method), 101 default\_unit() (openff.evaluator.properties.DielectricCodespoendencies (openff.evaluator.protocols.miscellaneous.DummyProtocol class method), 108 property), 411 default\_unit() (openff.evaluator.properties.EnthalpyOfMixpendencies (openff.evaluator.protocols.miscellaneous.FilterSubstanceB class method), 112 property), 407 default\_unit() (openff.evaluator.properties.EnthalpyOfVappendtionies (openff.evaluator.protocols.miscellaneous.MultiplyValue property), 392 class method), 115 default\_unit() (openff.evaluator.properties.ExcessMolardependencies (openff.evaluator.protocols.miscellaneous.SubtractValues class method), 105 property), 387 default\_unit() (openff.evaluator.properties.HostGuestBiddpexAffinities (openff.evaluator.protocols.miscellaneous.WeightByMoleFr property), 402 class method), 122 default\_unit() (openff.evaluator.properties.SolvationFredEperglencies (openff.evaluator.protocols.openmm.OpenMMEnergyMinin class method), 119 property), 417 default\_yank\_schema()  ${\tt dependencies} \ (open {\it ff. evaluator. protocols. open mm. Open MME valuate Energy}) \ and \ are the {\it final protocols. open mm. Open MME valuate Energy}) \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy}) \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy}) \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy}) \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protocols. open mm. Open MME valuate Energy} \ are the {\it final protoc$ (openff.evaluator.properties.HostGuestBindingAffinity property), 429 static method), 122 dependencies (openff.evaluator.protocols.openmm.OpenMMSimulation Density (class in openff.evaluator.properties), 100 property), 423 property), 300 property), 471
- dependencies (openff.evaluator.protocols.analysis.Averaga Diplandrin Coast (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRP)
- dependencies (openff.evaluator.protocols.analysis.Average Grape Edergiaes (openff.evaluator.protocols.paprika.analysis.Compute Reference property), 305 property), 481
- dependencies (openff.evaluator.protocols.analysis.Averaga Dependencies (openff.evaluator.protocols.paprika.analysis.ComputeSymn property), 294 property), 476
- dependencies (openff.evaluator.protocols.analysis.BaseAveneenbanciads(openff.evaluator.protocols.paprika.coordinates.AddDumm property), 289 property), 445
- dependencies (openff.evaluator.protocols.analysis.BaseDedependencies (obenff.evaluator.protocols.paprika.coordinates.PreparePu property), 315 property), 435
- dependencies (openff.evaluator.protocols.analysis.Computation protocols.paprika.coordinates.PrepareRea property), 310 property), 440
- dependencies (openff.evaluator.protocols.analysis.Decorredependencies (openff.evaluator.protocols.paprika.restraints.ApplyRest property), 326 property), 466
- dependencies (openff.evaluator.protocols.analysis.Decorredependencies (openff.evaluator.protocols.paprika.restraints.GenerateAtta property), 321 property), 450
- dependencies (openff.evaluator.protocols.coordinates.Build Coundinates.Build Coundin property), 332 property), 455
- dependencies (openff.evaluator.protocols.coordinates.Build Dowked Coived in (appending per lucion) dependencies (openff.evaluator.protocols.paprika.restraints.GenerateRele property), 344 property), 461
- dependencies (openff.evaluator.protocols.coordinates.SolvaepenstemesStementf.evaluator.protocols.reweighting.BaseEvaluateEnerg property), 337 property), 497
- dependencies (openff.evaluator.protocols.forcefield.BaseBaitpEntdencies (openff.evaluator.protocols.reweighting.BaseMBARProtoco property), 349 property), 503
- dependencies (openff.evaluator.protocols.forcefield.BuildIdeBerGemSixesmopenff.evaluator.protocols.reweighting.ConcatenateObserv property), 359 property), 492
- dependencies (openff.evaluator.protocols.forcefield.BuildSaripanffSysteines (openff.evaluator.protocols.reweighting.ConcatenateTraject property), 354 property), 487
- dependencies (openff.evaluator.protocols.forcefield.BuildTilepupSolstnaties (openff.evaluator.protocols.reweighting.ReweightDielectric property), 365 property), 514
- dependencies (openff.evaluator.protocols.gradients.ZeroGdediendencies (openff.evaluator.protocols.reweighting.ReweightObservable) property), 508 property), 371
- dependencies (openff.evaluator.protocols.groups.Conditiodal@ndencies (openff.evaluator.protocols.simulation.BaseEnergyMinimis property), 377 property), 519
- dependencies (openff.evaluator.protocols.miscellaneous.Add\u00e4\u00e4ndencies (openff.evaluator.protocols.simulation.BaseSimulation

	operty), 527			attribute), 552				
_		ator.protocols.storage.U	npack	SanabiSendatiandaahing	g ( <i>openff.e</i>	evaluator.serv	ver.Batch	
	operty), 532		¥7. 1.1	attribute), 91	7		0 100	3.60
_	ies (openff.evalue operty), 539	ator.protocols.yank.Base	e Yank I	P <b>enalobė_</b> pbc (openff.eva attribute), 417	-	ocols.openmn	n.OpenMMEnergy	/Minimis
_	ies (openff.evalue operty), 546	ator.protocols.yank.Liga	ndRec	c <b>eprabKæ<u>n</u>kRro(opeh</b> ff.eva attribute), 429		ocols.openmn	n.OpenMMEvalua	ıteEnergi
dependenci	ies(openff.evalue	ator.protocols.yank.Solv	ationY	Y <b>ænlaPrbeophi</b> c (openff.eva attribute), 423	aluator.prote	ocols.openmn	n.OpenMMSimula	ıtion
<i>pr</i> dependenci	operty), 554 ies (openff.	evaluator.workflow.Prot	ocol	enable_pbc(openff.eva	aluator.prote	ocols.reweigh	ting.BaseEvaluate	eEnergie:
	operty), 262			attribute), 496				
-	ies (openff.evalud operty), 268	ator.workflow.ProtocolG	Froup	enable_pbc (openff.eva attribute), 518		ocols.simulat	ion.BaseEnergyM	inimisati
Dielectri		(class	in	enable_pbc(openff.eva	aluator.prote	ocols.simulat	ion.BaseSimulatio	n
op	enff.evaluator.pro	operties), 107		attribute), 525	5			
	ments ( <i>openff.eva</i> tribute), 298	luator.protocols.analysi	s.Aver	rægæddid læC) ric Caprestiffne va	ıluator.stora	ige.attributes.	FilePath	
	//	duator protocols analysi	s Con	<i>method</i> ), 249 n <b>endd£sipelteMtunæije</b> sctoi	mi,			
ati	tribute), 309	•		(openff.evalua	tor.storage.	data.StoredF	reeEnergyData	
		lluator.protocols.reweigi	hting.I	ReweightD <b>ietterclouite</b> Go <b>ns</b> a		•7	Ed D. J	
	tribute), 512	( 1		endswith()(openff.eva	iluator.store	ige.attributes	.FilePath	
DivideValı		(class	in	method), 249	atau muata a.	.la an annum (	on an MMCinnel ation	
<i>op</i> 39		otocols.miscellaneous),		ensemble (openff.evalua attribute), 423	-	ns.openmm.C	урентызтинано	'n
	penff.evaluator.pr tribute), 293	otocols.analysis.Average	eObsei	r <b>eadsle</b> mble (openff.evalud attribute), 525	-	ols.simulation	.BaseSimulation	
	* *	otocols.miscellaneous.D	ivideV	7.		(class	in	
_	tribute), 395			openff.evaluat	or.propertie	es), 111		
	mplex_coordina	=		EnthalpyOfVaporizat		(class	in	
_		otocols.coordinates.Bui	ldDoc	ckedCoordi <b>npæ</b> sff.evaluat		es), 114		
	tribute), 343			estimated_propertie				
	gand_coordinat	=		(openff.evalua	tor.client.R	equestResult	at-	
		otocols.coordinates.Bui	ldDoc		, or	7	70 . 7	
	tribute), 343	. M		estimated_propertie	es ( <i>openff.e</i>	evaluator.ser	er.Batch	
		ets.MeasurementSource		attribute), 91	es in anang	Savaluaton ali	ant) 76	
	bute), 99	ator forcefield Lia ParCa		EvaluatorClient( <i>cla</i> : <i>AFvaldSavor</i> Exception,		.evaниают.сы	ent), 70	
	operty), 186	aior.jorcejieia.Ligi arGe		EvaluatorServer (cla		evaluator se	rver) 88	
DummyProte		(class		ExactAmount (class in				
op	enff.evaluator.pro	otocols.miscellaneous),	iii	exceptions (openff.e				
40 _	99			tribute), 86 exceptions (openff.eva	aluator.lave	rs.Calculatio	nLaverResult	
E				attribute), 190	)			
	_samples( <i>openff</i> tribute), 502	f.evaluator.protocols.rew	veighti	in <b>ex.Bent.MBAR/Ppoinff.o</b> lvo exceptions ( <i>openff.e</i> v				
		f.evaluator.protocols.rew	eighti	ing.Reweig <b>atDibloei</b> ki260		· ·		
	tribute), 514		Ü	ExcessMolarVolume		(class	in	
effective_	_samples( <i>openff</i>	f.evaluator.protocols.rew	eighti	ing.Reweig <b>hpObffeevable</b> t				
	tribute), 508	1		execute() (openff.eval method), 300	uator.proto	cols.analysis.	AverageDielectric	Constant
	atic_lambdas_1		ank De	meinoa), 500 r <b>esseci</b> ute() (openff.eval	uator proto	cols analysis	Average Free Fner	oies
ati	tribute), 552		инкГТ	method), 305				
	atic_lambdas_2 penff.evaluator.pr	2 cotocols.yank.SolvationY	ankPr	execute() (openff.eval. rotocol method), 294	uator.proto	cols.analysis	AverageObservab.	le

- execute() (openff.evaluator.protocols.analysis.BaseAverageEtsutæDleopenff.evaluator.protocols.paprika.coordinates.AddDummyAtomethod), 289 method), 445
- execute() (openff.evaluator.protocols.analysis.BaseDecorrectereProtocols.paprika.coordinates.PreparePullComethod), 315 method), 435
- execute() (openff.evaluator.protocols.analysis.ComputeDipaledItae()) (openff.evaluator.protocols.paprika.coordinates.PrepareRelease method), 310 method), 440
- execute() (openff.evaluator.protocols.analysis.Decorrelate (Nexartable Copenff.evaluator.protocols.paprika.restraints.ApplyRestraints method), 326 method), 466
- execute() (openff.evaluator.protocols.analysis.Decorrelate (Strajeutor(y)) (openff.evaluator.protocols.paprika.restraints.GenerateAttachR method), 321 method), 450
- execute() (openff.evaluator.protocols.coordinates.BuildCoexdinatesPackpeolff.evaluator.protocols.paprika.restraints.GeneratePullRest method), 332 method), 455
- execute() (openff.evaluator.protocols.coordinates.BuildDackedGoordin
- execute() (openff.evaluator.protocols.coordinates.SolvateExisting.St() (toppenff.evaluator.protocols.reweighting.BaseEvaluateEnergies method), 337 method), 497
- execute() (openff.evaluator.protocols.forcefield.BaseBuildExstenate() (openff.evaluator.protocols.reweighting.BaseMBARProtocol method), 349 method), 503
- execute() (openff.evaluator.protocols.forcefield.BuildLigParGeruSys(en(openff.evaluator.protocols.reweighting.ConcatenateObservable method), 359

  method), 492
- execute() (openff.evaluator.protocols.forcefield.BuildSmir@affSystem) (openff.evaluator.protocols.reweighting.ConcatenateTrajectorie method), 354 method), 487
- execute() (openff.evaluator.protocols.forcefield.BuildTLeaps@state() (openff.evaluator.protocols.reweighting.ReweightDielectricConmethod), 365 method), 514
- execute() (openff.evaluator.protocols.gradients.ZeroGradientscute() (openff.evaluator.protocols.reweighting.ReweightObservable method), 371 method), 508
  execute() (openff.evaluator.protocols.groups.Conditional@xxxxxvute() (openff.evaluator.protocols.simulation.BaseEnergyMinimisation

execute() (openff.evaluator.protocols.miscellaneous.DivideXekurte() (openff.evaluator.protocols.storage.UnpackStoredSimulationDa

execute() (openff.evaluator.protocols.miscellaneous.Filter&nbsrate&B\Rp&aff.evaluator.protocols.yank.LigandReceptorYankProtocol

- method), 377

  method), 520

  execute() (openff evaluator protocols miscellaneous AddVehessute() (openff evaluator protocols simulation BaseSimulation
- execute() (openff.evaluator.protocols.miscellaneous.AddVakæsute() (openff.evaluator.protocols.simulation.BaseSimulation method), 382 method), 527
- method), 397 method), 532 execute() (openff.evaluator.protocols.miscellaneous.DumnexProtocol) (openff.evaluator.protocols.yank.BaseYankProtocol
- execute() (openff.evaluator.protocols.miscellaneous.Dum**nyErutbe**@) (openff.evaluator.protocols.yank.BaseYankProtocol method), 412 method), 539
- method), 407 method), 546

  execute() (openff evaluator protocols miscellaneous MultiplicAddite() (openff evaluator protocols vank SolvationYankProtocol
- execute() (openff.evaluator.protocols.miscellaneous.Multi**pkydalte**() (openff.evaluator.protocols.yank.SolvationYankProtocol method), 392 method), 554
- execute() (openff.evaluator.protocols.miscellaneous.Subtr**existalities(**) (openff.evaluator.workflow.Protocol method), method), 387 265
- execute() (openff.evaluator.protocols.miscellaneous.WeiglerByWhitleF)racti(openff.evaluator.workflow.ProtocolGraph method), 402 method), 266
- execute() (openff.evaluator.protocols.openmm.OpenMMEnxegyMin(i)nisa(impenff.evaluator.workflow.ProtocolGroup method), 417 method), 270
- execute() (openff.evaluator.protocols.openmm.OpenMME**exhacter(e)** rgies (openff.evaluator.workflow.Workflow method), 429 method), 257
- execute() (openff.evaluator.protocols.openmm.OpenMMSiemelative() (openff.evaluator.workflow.WorkflowGraph method), 423 method), 258
- execute() (openff.evaluator.protocols.paprika.analysis.AnelypentRABsi() (openff.evaluator.storage.attributes.FilePath method), 471 method), 249
- execute() (openff.evaluator.protocols.paprika.analysis.Comprediction in the complements
  - method), 481 (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRole

execute() (openff.evaluator.protocols.paprika.analysis.ComputeSym**atetihyGo**);r&O6on method), 476

F		openff.evaluator.datase	ets.curation.compone	nts.filtering),
fidelity (openff.evaluator.datasets.CalculationSource		160		
attribute), 98	Filter	ByRacemicSchema	(class	in
FilePath (class in openff.evaluator.storage.attributes), 247		openff.evaluator.datase 160	ers.curation.compone	nts.futering),
FilterByCharged (class in	Filter	BySmiles	(class	in
openff.evaluator.datasets.curation.components.fi 164		openff.evaluator.datase 166	ets.curation.compone	nts.filtering),
FilterByChargedSchema (class in	Filter	BySmilesSchema	(class	in
openff.evaluator.datasets.curation.components.fi 164		openff.evaluator.datase 166	ets.curation.compone	nts.filtering),
FilterByElements (class in	Filter		(class	in
openff.evaluator.datasets.curation.components.fi 161		openff.evaluator.datase 167	ets.curation.compone	nts.filtering),
FilterByElementsSchema (class in	Filter	BySmirksSchema	(class	in
openff.evaluator.datasets.curation.components.fi.		openff.evaluator.datase 167	ets.curation.compone	nts.filtering),
FilterByEnvironments (class in	Filter	ByStereochemistry	(class	in
openff.evaluator.datasets.curation.components.fi	iltering),	openff.evaluator.datase 163	ets.curation.compone	nts.filtering),
FilterByEnvironmentsSchema (class in	Filter	${f SyStereochemistrySc}$		in
openff.evaluator.datasets.curation.components.fi 171	_	openff.evaluator.datase 163	ets.curation.compone	nts.filtering),
FilterByIonicLiquid (class in	Filter	BySubstances	(class	in
openff.evaluator.datasets.curation.components.fi 165		openff.evaluator.datase 170	ets.curation.compone	nts.filtering),
FilterByIonicLiquidSchema (class in	Filter	BySubstancesSchema	(class	in
openff.evaluator.datasets.curation.components.fi 165	_	openff.evaluator.datase 169		nts.filtering),
FilterByMoleFraction (class in	Filter	ByTemperature	(class	in
openff.evaluator.datasets.curation.components.fi 159		openff.evaluator.datase 157	_	nts.filtering),
FilterByMoleFractionSchema (class in	Filter	ByTemperatureSchema		in
openff.evaluator.datasets.curation.components.fi 159		openff.evaluator.datase 157		nts.filtering),
FilterByNComponents (class in	Filter	Ouplicates	(class	in
openff.evaluator.datasets.curation.components.fi 169		openff.evaluator.datase		nts.filtering),
FilterByNComponentsSchema (class in	Filter	OuplicatesSchema	(class	in
openff.evaluator.datasets.curation.components.fi 168		openff.evaluator.datase		
FilterByPressure (class in		ed_substance(openff.e	valuator.protocols.mi	scellaneous.FilterSub
openff.evaluator.datasets.curation.components.fi	iltering),	<pre>attribute), 406 SubstanceByRole</pre>	(class	in
158	riiteis	openff.evaluator.protoc	`	iri
FilterByPressureSchema (class in	:14: )	404	ois.miscenuncous),	
openff.evaluator.datasets.curation.components.fi 158	final_v	value_source(openff.e	valuator.workflow.scl	nemas.WorkflowSchen
FilterByPropertyTypes (class in	. final s	attribute), 279	waluator workflow Wo	orkflow
openff.evaluator.datasets.curation.components.fi	ilt <del>ering),</del> –	property), 255	variation.workjiow.wc	TKJIOW
162 FilterByPropertyTypesSchema (class in	find()	·	orage.attributes.FileI	Path
FilterByPropertyTypesSchema (class in openff.evaluator.datasets.curation.components.fi	iltering),	method), 250	3	
162	FingerF	PrintType	(class	in
FilterByRacemic (class in		openff.evaluator.datase	ets.curation.compone	nts.selection),

179			attrib	ute), 546		
<pre>force_field_id (open</pre>	nff.evaluator.server.Batch	at-	free_energy_	difference		
tribute), 91			(open	ff.evaluator.prote	ocols.yank.Solvatio	nYankProtocol
<pre>force_field_id(openff.</pre>	evaluator.storage.data.Bas	seSimu		-	•	
attribute), 226	Ü		free_energy_			
<pre>force_field_id(openff.</pre>	evaluator.storage.data.Sto	redFre	eEnergyDd <b>top</b> en	ff.evaluator.stora	ige.data.StoredFree	eEnergyData
attribute), 233				ute), 232	· · · · · · · · · · · · · · · · · · ·	3,
force_field_id(openff.	evaluator.storage.data.Sto	redSin			(class	in
attribute), 230				evaluator.storas	,	
force_field_id(openff.	evaluator.storage.auerv.Bo	aseSim				Substance
attribute), 241				method), 126		
force_field_id(openff.	evaluator storage auery Fi	reeEne		//	evaluator storage a	uerv BaseDataOuerv
attribute), 245	evaluation storage (quer y.17	ссынс		method), 235	randon siorage.q	ner y.BuseBura quer y
force_field_id(openff.	evaluator storage auery Si	mulati		* *	evaluator storage a	uery RaseSimulationDat
attribute), 243	evanuaior.siorage.quer y.si	muun		method), 241	vanaior.siorage.q	acı y.DasesimaianonDai
force_field_path(open	off avaluator protocols for	rafiald		* *	evaluator storage a	uery ForceFieldOuery
attribute), 348	ijj.evataator.protocots.jore	ејши.		method), 238	evaluator.storage.q	uery.rorcertetaQuery
force_field_path(open	off analystan protosols for	ofold		//	maluaton stonago a	uam EnacEnanos DataOs
attribute), 360	ijj.evatuator.protocots.jore	енена.			evatuator.storage.q	uery.FreeEnergyDaiaQu
/ /	off analystan mustosals for	1 d		method), 246	ovalvatov stovaco s	am. Cimulatian Data Ou
force_field_path(open	ijj.evatuator.protocots.jorc	енена.			evatuator.storage.q	uery.simuiaiionDaiaQue
attribute), 354	or 1 , 1 c	C 11		method), 243		EL MID C
force_field_path(open	ıff.evaiuator.protocois.forc	епена.			atasets.tnermomi.1	nermoMLDataSet
attribute), 365	m 1	7.		method), 143	1	
<pre>force_field_path(open</pre>	ıff.evaluator.protocols.gra	dients.			lluator.utils.excepti	ons.EvaluatorException
attribute), 369				method), 87		
<pre>force_field_path(open</pre>	ıff.evaluator.protocols.stor	age.Ui			lluator.workflow.Wo	orkflowException
attribute), 531				method), 257		
<pre>force_field_path(open</pre>	ıff.evaluator.protocols.yan	k.Liga	_		datasets.thermoml	.ThermoMLDataSet
attribute), 544				method), 143		
${\tt force\_field\_source}(option{ \cite{bigs.png}}\ \cite{bigs.png})$	oenff.evaluator.storage.dat	ta.Forc			:attributes.Attribut	eClass
attribute), 223				method), 565		
${\tt force\_field\_source}(option{ \cite{bigs.png}}$	วenff.evaluator.storage.que	ery.Foi			client.Connection(	Options
attribute), 238				method), 81		
ForceFieldData	(class	in	<pre>from_json()</pre>	(openff.evalua	tor.client.Request	class
	storage.data), 222		metho			
ForceFieldQuery	(class	in	<pre>from_json()</pre>	(openff.evalua	tor.client.RequestO	Pptions
openff.evaluator.	storage.query), 238		class	method), 85		
ForceFieldSource	(class	in	<pre>from_json()</pre>	(openff.evalu	uator.client.Requesi	tResult
openff.evaluator.	forcefield), 181		class	method), 87		
<pre>format() (openff.evalue</pre>	ator.storage.attributes.File	<i>Path</i>	<pre>from_json()(</pre>	openff.evaluator.	datasets.Calculation	onSource
method), 250				method), 98		
<pre>format_map() (openff.eva</pre>	aluator.storage.attributes.l	FilePat	hfrom_json()(	openff.evaluator.	datasets.Measuren	nentSource
method), 250				method), 99		
frame_counts(openff.eva	aluator.protocols.reweight	ing.Ba		* *	datasets.PhysicalP	Property
attribute), 501	1 0	O		method), 95	Ž	1 2
<pre>frame_counts(openff.eva</pre>	aluator.protocols.reweight	ing.Re		* * * * * * * * * * * * * * * * * * * *	datasets.PhysicalP	PropertyDataSet
attribute), 514				method), 140		
frame_counts(openff.eva	aluator protocols reweight	ing Re		//	or datasets Source	class
attribute), 508			metho			
free_energy_differen	Ce				datasets.taproom.T	TanroomDataSet
	r.protocols.yank.BaseYank	Protoc		method), 148		.ap. roomDamber
attribute), 538	.p. otocow.yank.Dase tank	. 10100		* *	datasets.taproom.T	TaproomSource
free_energy_differen	Ce			method), 151	amuseis.iuproom.1	.ap. compouree
	ce r.protocols.yank.LigandRe	contor			datasets thermoul	ThermoMI Data Set
(оренуј.ечинано	.р.оюсов.ушк.ыдинике	τεριστ	TOTAL ENTITE (	эренуу.е чининот.	adiaseis.mermomi	.1 nermomeduusei

- class method), 143 class method), 338
- from\_json() (openff.evaluator.forcefield.LigParGenForceFirbiSojuson() (openff.evaluator.protocols.forcefield.BuildLigParGenSystem class method), 186 class method), 360
- from\_json() (openff.evaluator.forcefield.SmirnoffForceFielEtSomrjson() (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem class method), 183 class method), 354
- from\_json() (openff.evaluator.forcefield.TLeapForceField\( \) (openff.evaluator.protocols.forcefield.BuildTLeapSystem class method), 184 class method), 366
- from\_json() (openff.evaluator.layers.CalculationLayerResfittom\_json() (openff.evaluator.protocols.gradients.ZeroGradients class method), 190 class method), 371
- from\_json() (openff.evaluator.layers.CalculationLayerSch&rom\_json() (openff.evaluator.protocols.groups.ConditionalGroup class method), 192 class method), 378
- from\_json() (openff.evaluator.layers.reweighting.Reweightingticljeson() (openff.evaluator.protocols.groups.ConditionalGroup.C
- from\_json() (openff.evaluator.layers.simulation.SimulationEschnerieson() (openff.evaluator.protocols.miscellaneous.AddValues class method), 198 class method), 382
- from\_json() (openff.evaluator.layers.workflow.WorkflowCalicatutiysosichemupenff.evaluator.protocols.miscellaneous.DivideValue class method), 195 class method), 397
- from\_json() (openff.evaluator.properties.Density class from\_json() (openff.evaluator.protocols.miscellaneous.DummyProtocol class method), 102 class method), 412
- from\_json() (openff.evaluator.properties.DielectricConstafarom\_json() (openff.evaluator.protocols.miscellaneous.FilterSubstanceBy class method), 109 class method), 407
- from\_json() (openff.evaluator.properties.EnthalpyOfVapofication) (openff.evaluator.protocols.miscellaneous.SubtractValues class method), 116 class method), 387
- class method), 116
  class method), 38/
  from\_json() (openff.evaluator.properties.ExcessMolarVolumem\_json() (openff.evaluator.protocols.miscellaneous.WeightByMoleFraclass method), 106
  class method), 402
- from\_json() (openff.evaluator.properties.HostGuestBindingAffiningson() (openff.evaluator.protocols.openmm.OpenMMEnergyMinimic class method), 123 class method), 417
- from\_json() (openff.evaluator.properties.SolvationFreeEn&rgom\_json() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergenetation), 119

  class method), 429
- from\_json() (openff.evaluator.protocols.analysis.AveragePiebmtrjs6hh)t(openff.evaluator.protocols.openmm.OpenMMSimulation class method), 300 class method), 423
- from\_json() (openff.evaluator.protocols.analysis.AverageFfredingsgins() (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhaclass method), 306

  class method), 472
- from\_json() (openff.evaluator.protocols.analysis.Average@hxamyajskan() (openff.evaluator.protocols.paprika.analysis.ComputeReference class method), 294 class method), 482
- from\_json() (openff.evaluator.protocols.analysis.BaseAver**ingenObjsorabl**(openff.evaluator.protocols.paprika.analysis.ComputeSymme class method), 289 class method), 477
- from\_json() (openff.evaluator.protocols.analysis.BaseDec**bran**taj**ePont**(): (bpenff.evaluator.protocols.paprika.coordinates.AddDummy.class method), 316

  class method), 445
- from\_json() (openff.evaluator.protocols.analysis.Compute**Eripmle)Mon**(e) (openff.evaluator.protocols.paprika.coordinates.PreparePull class method), 311 class method), 435
- from\_json() (openff.evaluator.protocols.analysis.Decorrelfinantsiysonkle (openff.evaluator.protocols.paprika.coordinates.PrepareRelectors method), 326 class method), 440
- from\_json() (openff.evaluator.protocols.analysis.Decorreliancinajescon() (openff.evaluator.protocols.paprika.restraints.ApplyRestraints class method), 321 class method), 466
- from\_json() (openff.evaluator.protocols.coordinates.BuildEromdinates.Pachpeolff.evaluator.protocols.paprika.restraints.GenerateAttac.class method), 332 class method), 450
- from\_json() (openff.evaluator.protocols.coordinates.BuildPronkedContdinapenff.evaluator.protocols.paprika.restraints.GeneratePullR class method), 344 class method), 456
- ${\tt from\_json()}\ (open \textit{ff.evaluator.protocols.coordinates.Solva \textit{fadonist} \textit{jsgSM()} \\ (\textit{toppen} \textit{ff.evaluator.protocols.paprika.restraints.GenerateReleants)} \\ (\textit{toppen} \textit{ff.evaluator.protocols.paprika.restraints.GenerateReleants)} \\ (\textit{toppen} \textit{ff.evaluator.protocols.paprika.restraints.GenerateReleants)} \\ (\textit{toppen} \textit{ff.evaluator.protocols.paprika.restraints)} \\ (\textit{toppen} \textit{ff.evaluator.protocols.paprika.postate.posta$

class method), 235

- class method), 461 method), 131
- from\_json() (openff.evaluator.protocols.reweighting.BaseErodnajesCore() ies(openff.evaluator.substances.Component class method), 497 class method), 130
- from\_json() (openff.evaluator.protocols.reweighting.BaseMBARF som() (openff.evaluator.substances.ExactAmount class method), 503

  class method), 133
- from\_json() (openff.evaluator.protocols.reweighting.Conclinomte@bset) (ubbenff.evaluator.substances.MoleFraction class method), 492

  class method), 135
- from\_json() (openff.evaluator.protocols.reweighting.Conclinomic@Bouje@oriekopenff.evaluator.substances.Substance class method), 487 class method), 128
- from\_json() (openff.evaluator.protocols.reweighting.Rewefgland)igkort() (Openff.evaluator.thermodynamics.ThermodynamicState class method), 514 class method), 137
- from\_json() (openff.evaluator.protocols.reweighting.Rewefglandbjsordbl&openff.evaluator.utils.exceptions.EvaluatorException class method), 508 class method), 87
- from\_json() (openff.evaluator.protocols.simulation.BaseEftrogyMisom()at(openff.evaluator.workflow.Protocol class class method), 520 method), 265
- from\_json() (openff.evaluator.protocols.simulation.BaseSifmontijpson() (openff.evaluator.workflow.ProtocolGroup class method), 527 class method), 270
- from\_json() (openff.evaluator.protocols.storage.UnpackStdmotSirjmotn()n(Dpenff.evaluator.workflow.schemas.ProtocolGroupSchema class method), 533 class method), 275
- from\_json() (openff.evaluator.protocols.yank.BaseYankPromoljson() (openff.evaluator.workflow.schemas.ProtocolReplicator class method), 539 class method), 277
- from\_json() (openff.evaluator.protocols.yank.LigandReceptromailsPow()colpenff.evaluator.workflow.schemas.ProtocolSchema class method), 546 class method), 273
- from\_json() (openff.evaluator.protocols.yank.SolvationYankRmvtjcxdn() (openff.evaluator.workflow.schemas.WorkflowSchema class method), 554 class method), 280
- from\_json() (openff.evaluator.storage.data.BaseSimulationEDouta\_json() (openff.evaluator.workflow.WorkflowResult class method), 227 class method), 260
- from\_json() (openff.evaluator.storage.data.ForceFieldDat&rom\_openmm() (openff.evaluator.utils.observables.ObservableFrame class method), 223 class method), 569
- from\_json() (openff.evaluator.storage.data.HashableStoreffDomapandas() (openff.evaluator.datasets.PhysicalPropertyDataSet class method), 221 class method), 141
- from\_json() (openff.evaluator.storage.data.ReplaceableDdfnom\_pandas() (openff.evaluator.datasets.taproom.TaproomDataSet class method), 224 class method), 148
- from\_json() (openff.evaluator.storage.data.StoredFreeEnefgyMapandas() (openff.evaluator.datasets.thermoml.ThermoMLDataSet class method), 233 class method), 143
- from\_json() (openff.evaluator.storage.data.StoredSimulati**bnDnutp**ath() (openff.evaluator.forcefield.SmirnoffForceFieldSource class method), 230 class method), 182
- class method), 230 class method), 182 from\_json() (openff.evaluator.storage.query.BaseDataQueryom\_schema() (openff.evaluator.protocols.analysis.AverageDielectricCon
- from\_json() (openff.evaluator.storage.query.BaseSimulati@nDntusQhema() (openff.evaluator.protocols.analysis.AverageFreeEnergies class method), 241 class method), 306

class method), 300

- from\_json() (openff.evaluator.storage.query.ForceFieldQuferom\_schema() (openff.evaluator.protocols.analysis.AverageObservable class method), 239 class method), 294
- class method), 239 class method), 294
  from\_json() (openff.evaluator.storage.query.FreeEnergyDfirm@usschema() (openff.evaluator.protocols.analysis.BaseAverageObservation)
- class method), 246 class method), 289 from\_json() (openff.evaluator.storage.query.SimulationDataQmesy.hema() (openff.evaluator.protocols.analysis.BaseDecorrelateProtocols.BaseDecorrelatePr
- class method), 243 class method), 316
  from\_json() (openff.evaluator.storage.query.SubstanceQuency\_SubstanceQue
- $\verb|from_json()| (open ff. evaluator. substances. Amount class & \verb|from_schema()| (open ff. evaluator. protocols. analysis. Decorrelate Observable (open ff. evaluator. protocols. protocols.$

```
class method), 326
                                                                                                                                class method), 467
from_schema() (openff.evaluator.protocols.analysis.DecorfedonteTsafreenter(y) (openff.evaluator.protocols.paprika.restraints.GenerateAtt
                  class method), 321
                                                                                                                                class method), 451
from_schema() (openff.evaluator.protocols.coordinates.Buflrl@nosdhæmes@achpenff.evaluator.protocols.paprika.restraints.GeneratePu
                  class method), 332
                                                                                                                                class method), 456
from_schema() (openff.evaluator.protocols.coordinates.Bufldbnockell@nacdinapenff.evaluator.protocols.paprika.restraints.GenerateRe
                  class method), 344
                                                                                                                                class method), 461
from_schema() (openff.evaluator.protocols.coordinates.SolimonExistingnia() (toppenff.evaluator.protocols.reweighting.BaseEvaluateEnergy
                  class method), 338
                                                                                                                                class method), 497
from_schema() (openff.evaluator.protocols.forcefield.BaseBridinlSssdbrema() (openff.evaluator.protocols.reweighting.BaseMBARProtocols.
                  class method), 349
                                                                                                                                class method), 503
from_schema() (openff.evaluator.protocols.forcefield.BuildfipRasGementaeObse.
                  class method), 360
                                                                                                                                class method), 492
from_schema() (openff.evaluator.protocols.forcefield.BuildSpoint_sff.ivented() (openff.evaluator.protocols.reweighting.ConcatenateTrajec
                  class method), 354
                                                                                                                                class method), 487
from_schema() (openff.evaluator.protocols.forcefield.Buildfilamp$chtema() (openff.evaluator.protocols.reweighting.ReweightDielectri
                  class method), 366
                                                                                                                                class method), 514
from_schema() (openff.evaluator.protocols.gradients.Zero&routienchema() (openff.evaluator.protocols.reweighting.ReweightObservation
                  class method), 371
                                                                                                                                class method), 508
from_schema() (openff.evaluator.protocols.groups.ConditionalGraphema() (openff.evaluator.protocols.simulation.BaseEnergyMinimi
                  class method), 378
                                                                                                                                class method), 520
from_schema() (openff.evaluator.protocols.miscellaneous. AtdMakarema() (openff.evaluator.protocols.simulation.BaseSimulation
                                                                                                                                class method), 527
                  class method), 382
from_schema() (openff.evaluator.protocols.miscellaneous. Dronde Kahnema() (openff.evaluator.protocols.storage. UnpackStoredSimulati
                  class method), 397
                                                                                                                                class method), 533
from_schema() (openff.evaluator.protocols.miscellaneous. Dromn_schema() (openff.evaluator.protocols.yank.BaseYankProtocol
                  class method), 412
                                                                                                                                class method), 539
from_schema() (openff.evaluator.protocols.miscellaneous. Filtan Subwema & Rykplenff.evaluator.protocols.yank.LigandReceptorYankProt
                  class method), 407
                                                                                                                                class method), 546
from_schema() (openff.evaluator.protocols.miscellaneous. Madniphydhema() (openff.evaluator.protocols.yank.SolvationYankProtocol
                  class method), 392
                                                                                                                                class method), 555
from_schema() (openff.evaluator.protocols.miscellaneous. SurlomassObjectus()
                                                                                                                                                      (openff.evaluator.workflow.Protocol
                  class method), 387
                                                                                                                                class method), 262
from_schema() (openff.evaluator.protocols.miscellaneous. Weight By Wenta () (openff.evaluator.workflow.ProtocolGroup
                  class method), 402
                                                                                                                                class method), 270
from_schema() (openff.evaluator.protocols.openmm.OpenMMolineschemia()nisat(openff.evaluator.workflow.Workflow
                  class method), 417
                                                                                                                                class method), 256
from_schema() (openff.evaluator.protocols.openmm.OpenMinosity.stunialiy@reigenff.evaluator.datasets.PropertyPhase
                  class method), 429
                                                                                                                                class method), 96
from_schema() (openff.evaluator.protocols.openmm.OpenMMMsignaphif.evaluator.datasets.thermoml.ThermoMLDataSet
                                                                                                                                class method), 143
                  class method), 423
from_schema() (openff.evaluator.protocols.paprika.analystromalymal() Refferente evaluator.datasets.thermoml.ThermoMLDataSet
                  class method), 472
                                                                                                                                class method), 144
from_schema() (openff.evaluator.protocols.paprika.analysitut@mpartaRafpeenff.eWubuktor.workflow.utils.ProtocolPath
                  class method), 482
                                                                                                                                property), 285
from_schema() (openff.evaluator.protocols.paprika.analys isut township with the schema () (openff.evaluator.protocols.paprika.analys isut township with township with the schema () (openff.evaluator.protocols.paprika.analys isut township with the schema () (openff.evaluator.protocols.paprika.analys is township with th
                  class method), 477
                                                                                                                                attribute), 400
from_schema() (openff.evaluator.protocols.paprika.coordinates.AddDummyAtoms
                  class method), 446
from_schema() (openff.evaluator.protocols.paprika.coordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalcoordinaks.paprikalc
                  class method), 435
                                                                                                                                ule openff.evaluator.protocols.utils), 561
from_schema() (openff.evaluator.protocols.paprika.coordinates.Prepare relates ()
                  class method), 440
                                                                                                                                (openff.evaluator.workflow.Workflow
                                                                                                                                                                                                              static
```

from\_schema() (openff.evaluator.protocols.paprika.restraints.ApplyRastrajats256

- generate\_reweighting\_protocols() (in module class method), 300 openff.evaluator.protocols.utils), 562 get\_attributes() (openff.evaluator.protocols.analysis.AverageFreeEnerg generate\_simulation\_protocols() (in module class method), 306  $\verb"get_attributes"() (open fi. evaluator. protocols. analysis. Average Observab in the protocol of the protoc$ openff.evaluator.protocols.utils), 562 GenerateAttachRestraints (class inclass method), 294 openff.evaluator.protocols.paprika.restraints), get\_attributes() (openff.evaluator.protocols.analysis.BaseAverageObse 448 class method), 289 GeneratePullRestraints (class get\_attributes() (openff.evaluator.protocols.analysis.BaseDecorrelateI openff.evaluator.protocols.paprika.restraints), class method), 316  $\verb"get_attributes"() (open ff. evaluator. protocols. analysis. Compute Dipole Matter is a compute Dip$ 453 GenerateReleaseRestraints (class class method), 311 in openff.evaluator.protocols.paprika.restraints), get\_attributes() (openff.evaluator.protocols.analysis.DecorrelateObser class method), 326  $\verb|get()| (openff. evaluator. utils. observables. Observable Frame \verb|get_attributes()| (openff. evaluator. protocols. analysis. Decorrelate Trajection of the protocol of the$ class method), 321 *method*), 570  $\verb|get_amounts()| (open \textit{ff.evaluator.substances.Substance} \quad \verb|get_attributes()| (open \textit{ff.evaluator.protocols.coordinates.Build Coordinates.Build Coord$ *method*), 127 class method), 332  $\verb|get_attributes()| (open \textit{ff.evaluator.attributes.AttributeClgoex}\_\texttt{attributes()}) (open \textit{ff.evaluator.protocols.coordinates.BuildDockedClgoex}\_\texttt{attributes()}) (open \textit{ff.evaluator.protocols.coordinates.BuildDockedClgoex}\_$ class method), 565 class method), 344  $\verb|get_attributes(|)| (open \textit{ff.evaluator.client.ConnectionOptions_attributes(|)})| (open \textit{ff.evaluator.protocols.coordinates.SolvateExisting_attributes(|)})| (open \textit{ff.evaluator.protocols.coordinates.SolvateSolvates(|)})| (open \textit{ff.evaluator.protocols.coordinates.Solvates(|)})| (open \textit{ff.evaluator.protocols.coordinates)| (open$ class method), 81 class method), 338 get\_attributes()  $(openff. evaluator. client. Request \quad \verb|get_attributes()| (openff. evaluator. protocols. forcefield. Base Build System (openff. evaluator. protocols. p$ class method), 83 class method), 350 get\_attributes() (openff.evaluator.client.RequestOptionsget\_attributes() (openff.evaluator.protocols.forcefield.BuildLigParGen class method), 85 class method), 360 get\_attributes() (openff.evaluator.client.RequestResult get\_attributes() (openff.evaluator.protocols.forcefield.BuildSmirnoffSy. class method), 87 class method), 354 get\_attributes() (openff.evaluator.datasets.PhysicalPropertyattributes() (openff.evaluator.protocols.forcefield.BuildTLeapSyste class method), 95 class method), 366 get\_attributes() (openff.evaluator.layers.CalculationLagertRastdributes() (openff.evaluator.protocols.gradients.ZeroGradients
- class method), 190 class method), 371
- get\_attributes() (openff.evaluator.layers.CalculationLa**ger**Schennibutes() (openff.evaluator.protocols.groups.ConditionalGroup class method), 192 class method), 378

- get\_attributes() (openff.evaluator.layers.workflow.Workflow.Workflow.GaltributeSchefupenff.evaluator.protocols.miscellaneous.DivideValue class method), 195 class method), 397
- get\_attributes() (openff.evaluator.properties.DielectricGenstattributes() (openff.evaluator.protocols.miscellaneous.FilterSubstattributes() (openff.evaluator.protocols.miscellaneous.FilterSubstattri
- get\_attributes() (openff.evaluator.properties.EnthalpyOfMixingtributes() (openff.evaluator.protocols.miscellaneous.MultiplyValuator.protocols.miscellaneous.miscel
- get\_attributes() (openff.evaluator.properties.EnthalpyOf&uparitmibutes() (openff.evaluator.protocols.miscellaneous.SubtractVal class method), 116 class method), 387
- get\_attributes() (openff.evaluator.properties.ExcessMogetVolumeibutes() (openff.evaluator.protocols.miscellaneous.WeightByM class method), 106
  class method), 402
- get\_attributes() (openff.evaluator.properties.HostGuestBindingetffibites() (openff.evaluator.protocols.openmm.OpenMMEnergyl class method), 123 class method), 417
- get\_attributes() (openff.evaluator.properties.SolvationFgetEatrgyibutes() (openff.evaluator.protocols.openmm.OpenMMEvaluator.class method), 119 class method), 429
- $\verb|get_attributes()| (open ff. evaluator. protocols. analysis. A \textit{vgetgeDietechnitEx}() \textit{timp} en \textit{ff. evaluator. protocols. open mm. Open MMS imulations and the protocols of the protocols$

class method), 423	class method), 221
get_attributes()( <i>openff.evaluator.protocols.paprika.an<b>gb</b>±is</i>	aArtarlybeAPRPh(aspenff.evaluator.storage.data.ReplaceableData
class method), 472	class method), 224
get_attributes()(openff.evaluator.protocols.paprika.an <b>gb</b> is	aCturijbootleskije(repellifikvaluator.storage.data.StoredFreeEnergyData
class method), 482	class method), 233
	aCturiphovetSess(in) & typeCofficercollinoutor.storage.data.StoredSimulationData
class method), 477	class method), 230
get_attributes() (openff.evaluator.protocols.paprika.co <b>gra</b> tin	
class method), 446	class method), 235
**	atexxPioptiesRi)l[Opemflinudenator.storage.query.BaseSimulationDataQ
class method), 435	class method), 241
	atex H. Propries Release Office dilutter. storage. query. Force Field Query
class method), 440	class method), 239
**	tass memoa', 239 tastApplyReets (im(openff.evaluator.storage.query.FreeEnergyDataQuer
class method), 467	class method), 246
	tatGributesAddupestffciuuduator.storage.query.SimulationDataQuery
class method), 451	class method), 243
	tatti ributeBi(I)Ropaffievaluator.storage.query.SubstanceQuery
class method), 456	class method), 237
get_attributes()( <i>openff.evaluator.protocols.paprika.res<b>gæt</b>in.</i>	
class method), 461	class method), 132
get_attributes()( <i>openff.evaluator.protocols.reweightin<b>g&amp;u</b>s</i>	
class method), 498	class method), 130
get_attributes()( <i>openff.evaluator.protocols.reweightin<b>g&amp;t</b>s</i>	<b>eMBARRtoto(</b> )l(openff.evaluator.substances.ExactAmount
class method), 503	class method), 133
$\mathtt{get\_attributes()}$ (openff.evaluator.protocols.reweightin $\mathtt{geo}$	atterisbet(bse(r):abpenff.evaluator.substances.MoleFraction
class method), 492	class method), 135
$\mathtt{get\_attributes()}$ ( $\mathit{openff.evaluator.protocols.reweightin}$ $\mathtt{geo}$	atterichestiese(c)(cipenff.evaluator.substances.Substance
class method), 487	class method), 128
get_attributes()( <i>openff.evaluator.protocols.reweightinգ</i> <b>gՔե</b> <u>տ</u>	atghtDielees(dCopsuffievaluator.thermodynamics.ThermodynamicSta
class method), 514	class method), 137
get_attributes()( <i>openff.evaluator.protocols.reweightin</i> <b>gR</b> e <u>и</u>	ataln Dbutesable openff.evaluator.workflow.Protocol
class method), 508	class method), 265
get_attributes()( <i>openff.evaluator.protocols.simulation.</i> <b>@ex</b> e	
class method), 520	class method), 270
	Situalibiaties() (openff.evaluator.workflow.schemas.ProtocolGroupSci
class method), 527	class method), 275
**	Stortectissing these Conformation is a contractive statement of the contractiv
class method), 533	class method), 273
	autoributes() (openff.evaluator.workflow.schemas.WorkflowSchema
class method), 539	class method), 280
get_attributes() (openff.evaluator.protocols.yank.Ligan <b>d&amp;t</b> co	
class method), 546	class method), 260
get_attributes()( <i>openff.evaluator.protocols.yank.Solva<b>tjen</b>tY</i>	
class method), 555	(openff.evaluator.protocols.analysis.AverageDielectricConstant
get_attributes() (openff.evaluator.server.Batch class	method), 300
<i>,,</i>	class_attribute()
get_attributes()(openff.evaluator.storage.data.BaseSimulati	
class method), 227	method), 306
get_attributes() ( <i>openff.evaluator.storage.data.BaseSto<b>get</b>D</i>	
class method), 220	(openff. evaluator. protocols. analysis. Average Observable
get_attributes()(openff.evaluator.storage.data.ForceFieldDd	ata method), 295
class method), 223 get_	class_attribute()
get_attributes()(openff.evaluator.storage.data.HashableStor	redDuamenff.evaluator.protocols.analysis.BaseAverageObservable

```
method), 289
                                                                                                                                                                                method), 392
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (open ff. evaluator. protocols. analysis. Base Decorrelate Protocol benefi. evaluator. protocols. miscellaneous. Subtract Values
                        method), 316
                                                                                                                                                                                 method), 388
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.analysis.ComputeDipoleMoment@openff.evaluator.protocols.miscellaneous.WeightByMoleFraction
                        method), 311
                                                                                                                                                                                method), 402
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.analysis.DecorrelateObservable&openff.evaluator.protocols.openmm.OpenMMEnergyMinimisatio
                        method), 326
                                                                                                                                                                                method), 417
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.analysis.DecorrelateTrajectory (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies
                                                                                                                                                                                method), 429
                        method), 321
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.coordinates.BuildCoordinatesPachpenlff.evaluator.protocols.openmm.OpenMMSimulation
                         method), 333
                                                                                                                                                                                 method), 423
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.coordinates.BuildDockedCoordinates.ff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase
                        method), 345
                                                                                                                                                                                method), 472
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.coordinates.SolvateExistingStruator.protocols.paprika.analysis.ComputeReferenceW
                        method), 338
                                                                                                                                                                                method), 482
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.forcefield.BaseBuildSystem
                                                                                                                                                                                (openff.evaluator.protocols.paprika.analysis.ComputeSymmetryC
                        method), 350
                                                                                                                                                                                method), 477
                                                                                                                                                       get_class_attribute()
get_class_attribute()
                         (openff.evaluator.protocols.force field.BuildLigParGenSystem (openff.evaluator.protocols.paprika.coordinates.AddDummyAtom)
                        method), 360
                                                                                                                                                                                method), 446
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                        (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem (openff.evaluator.protocols.paprika.coordinates.PreparePullCoor
                        method), 355
                                                                                                                                                                                method), 435
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.forcefield.BuildTLeapSystem
                                                                                                                                                                                (open ff. evaluator. protocols. paprika. coordinates. Prepare Release Contract of the contra
                        method), 366
                                                                                                                                                                                method), 440
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.gradients.ZeroGradients
                                                                                                                                                                                (openff.evaluator.protocols.paprika.restraints.ApplyRestraints
                        method), 371
                                                                                                                                                                                method), 467
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.groups.ConditionalGroup
                                                                                                                                                                                (openff.evaluator.protocols.paprika.restraints.GenerateAttachResset (a) and a substitution of the context of 
                        method), 378
                                                                                                                                                                                method), 451
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.miscellaneous.AddValues
                                                                                                                                                                                 (openff.evaluator.protocols.paprika.restraints.GeneratePullRestra
                        method), 383
                                                                                                                                                                                method), 456
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (open {\it ff. evaluator. protocols. miscellaneous.} Divide Value
                                                                                                                                                                                (open ff. evaluator. protocols. paprika. restraints. Generate Release Releas
                        method), 397
                                                                                                                                                                                method), 461
                                                                                                                                                       get_class_attribute()
get_class_attribute()
                         (openff.evaluator.protocols.miscellaneous.DummyProtocol (openff.evaluator.protocols.reweighting.BaseEvaluateEnergies
                        method), 412
                                                                                                                                                                                method), 498
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                         (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRpenff.evaluator.protocols.reweighting.BaseMBARProtocol
                        method), 407
                                                                                                                                                                                method), 503
get_class_attribute()
                                                                                                                                                       get_class_attribute()
                                                                                                                                                                               (openff.evaluator.protocols.reweighting.ConcatenateObservables
                         (openff.evaluator.protocols.miscellaneous.MultiplyValue
```

method), 492	method), 333
<pre>get_class_attribute()</pre>	$\verb"get_value" () (open ff. evaluator. protocols. coordinates. Build Docked Coordinates. Build D$
(openff.evaluator.protocols.reweighting.Concate	
method), 488	$\verb"get_value" () (open ff. evaluator. protocols. coordinates. Solvate Existing Structure) () (open ff. evaluator. protocols. coordinates. Solvate Existing Structure) () () () () () () () () () () () () ()$
<pre>get_class_attribute()</pre>	method), 338
	t <b>Djætecval:Geh3</b> t( <b>vp</b> enff.evaluator.protocols.forcefield.BaseBuildSystem
method), 514	method), 350
<pre>get_class_attribute()</pre>	$\verb"get_value()" (open fi.evaluator. protocols. force field. Build Lig Par Gen System for the protocol of the $
$(open {\it ff.evaluator.protocols.reweighting.} Reweigh$	
method), 509	$\verb"get_value()" (open ff. evaluator. protocols. force field. Build Smirn off System")$
<pre>get_class_attribute()</pre>	method), 355
	g <b>yldin iwa kuwa (n)</b> (openff.evaluator.protocols.forcefield.BuildTLeapSystem
method), 520	method), 366
<pre>get_class_attribute()</pre>	<pre>get_value() (openff.evaluator.protocols.gradients.ZeroGradients</pre>
$(open {\it ff. evaluator. protocols. simulation. Base Simulation}. Base {\it Simulation. Base Simulation. protocols. simulation. protocols.$	
method), 527	<pre>get_value() (openff.evaluator.protocols.groups.ConditionalGroup</pre>
<pre>get_class_attribute()</pre>	method), 378
(openff. evaluator. protocols. storage. Unpack Storeduction (a) the protocol of the protocol	d Sietul wad up (a) dopenff.evaluator.protocols.miscellaneous.AddValues
method), 533	method), 383
<pre>get_class_attribute()</pre>	$\verb"get_value" () (open ff. evaluator. protocols. miscellane ous. Divide Value$
(openff.evaluator.protocols.yank. Base Yank Protocols.yank. Base Yan	col method), 397
method), 539	$\verb"get_value" () \ (open \textit{ff.evaluator.protocols.miscellaneous.Dummy Protocol}$
<pre>get_class_attribute()</pre>	method), 412
(openff.evaluator.protocols.yank.Ligand Receptor)	Y <b>gekP.nabuel()</b> (openff.evaluator.protocols.miscellaneous.FilterSubstanceB
<i>method</i> ), 546	method), 408
<pre>get_class_attribute()</pre>	<pre>get_value() (openff.evaluator.protocols.miscellaneous.MultiplyValue</pre>
(openff.evaluator.protocols.yank.Solvation Yank P	rotocol method), 392
method), 555	$\verb"get_value" () (open ff. evaluator. protocols. miscellane ous. Subtract Values$
<pre>get_class_attribute()</pre>	method), 388
(openff.evaluator.workflow.Protocol method),	$\verb"get_value"() (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Weight By Mole Francisco () (open fi. evaluator. protocols. miscellane ous. Misce$
264	method), 402
<pre>get_class_attribute()</pre>	<pre>get_value() (openff.evaluator.protocols.openmm.OpenMMEnergyMinin</pre>
(openff.evaluator.workflow.ProtocolGroup	method), 417
method), 269	<pre>get_value() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergy</pre>
<pre>get_molecules_per_component()</pre>	method), 429
(openff.evaluator.substances.Substance	<pre>get_value() (openff.evaluator.protocols.openmm.OpenMMSimulation</pre>
method), 127	method), 423
<pre>get_value() (openff.evaluator.protocols.analysis.Average</pre>	e <b>Djetecral Gohyt(vp</b> enff.evaluator.protocols.paprika.analysis.AnalyzeAPRPi
method), 300	method), 472
	e F <b>get Enangies()</b> (openff.evaluator.protocols.paprika.analysis.ComputeRefer
method), 306	method), 482
	e <b>(Yese_rvalilee ()</b> (openff.evaluator.protocols.paprika.analysis.ComputeSymn
method) 295	method) 477

method), 326 method), 467
get\_value() (openff.evaluator.protocols.analysis.DecorrelgetTrajetragy) (openff.evaluator.protocols.paprika.restraints.GenerateAttac method), 321 method), 451

get\_value() (openff.evaluator.protocols.analysis.BaseAverger\_ObskwaDl (openff.evaluator.protocols.paprika.coordinates.AddDummy.

get\_value() (openff.evaluator.protocols.analysis.BaseDecgete[staPwet@copenff.evaluator.protocols.paprika.coordinates.PreparePull

 $\verb|get_value()| (open ff. evaluator. protocols. analysis. Compute \textit{QeipoleMum}(open ff. evaluator. protocols. paprika. coordinates. Prepare Relevant for the protocols of the$ 

get\_value() (openff.evaluator.protocols.analysis.DecorrelgetQhyernakle (openff.evaluator.protocols.paprika.restraints.ApplyRestraint

method), 446

method), 435

method), 440

get\_value() (openff.evaluator.protocols.coordinates.Build@vorsialutesPachpenff.evaluator.protocols.paprika.restraints.GeneratePullK

get\_value() (openij.evananor.pronocois.coorannaes.Duna**ge**02**annaes), papron**ij.evananor.pronocois.paprika.restranis.Generatet in

Index 611

method), 289

method), 316

*method*), 311

method), 456	method), 321
get_value() (openff.evaluator.protocols.paprika.restraintsg@tnwalt	
method), 461	(openff.evaluator.protocols.coordinates.BuildCoordinatesPackmo
get_value() (openff.evaluator.protocols.reweighting.BaseEvaluatel	
· · · · · · · · · · · · · · · · · · ·	ue_references()
<pre>get_value() (openff.evaluator.protocols.reweighting.BaseMBARPro method), 503</pre>	method), 345
get_value() (openff.evaluator.protocols.reweighting.Conc <b>ge</b> tasteO method), 492	theer was bless ences () (openff.evaluator.protocols.coordinates.Solvate Existing Structure
$\verb"get_value" ()  (open ff. evaluator. protocols. reweighting. Concatenate Transfer of the protocols of t$	
	ue_references()
$\verb"get_value()" (open fi. evaluator. protocols. reweight in g. Reweight Diele and the protocol of the protoco$	· · · · · · · · · · · · · · · · · · ·
method), 515	method), 350
get_value() (openff.evaluator.protocols.reweighting.ReweightiQbak	
method), 509	(openff.evaluator.protocols.forcefield.BuildLigParGenSystem
get_value() (openff.evaluator.protocols.simulation.BaseEnergyMin	
,,	ue_references()
<pre>get_value() (openff.evaluator.protocols.simulation.BaseSimulation</pre>	method), 355
get_value() (openff.evaluator.protocols.storage.UnpackStgredSinal	
method), 533	(openff.evaluator.protocols.forcefield.BuildTLeapSystem
<pre>get_value() (openff.evaluator.protocols.yank.BaseYankProtocol</pre>	method), 366
	ue_references()
<pre>get_value() (openff.evaluator.protocols.yank.LigandReceptorYankl</pre>	
method), 547	method), 372
$\verb"get_value"() (open \textit{ff.evaluator.protocols.yank.Solvation} Yan \textit{get} \texttt{"roward} and "roward" and "$	de_references()
method), 555	(openff.evaluator.protocols.groups.ConditionalGroup
<pre>get_value() (openff.evaluator.workflow.Protocol</pre>	method), 376
	ue_references()
<pre>get_value() (openff.evaluator.workflow.ProtocolGroup</pre>	(openff.evaluator.protocols.miscellaneous.AddValues method), 383
<pre>get_value_references()</pre>	ue_references()
(openff.evaluator.protocols.analysis.AverageDielectricConsmethod), 301	xt( <b>xp</b> enff.evaluator.protocols.miscellaneous.DivideValue method), 397
	ue_references()
(openff.evaluator.protocols.analysis.AverageFreeEnergies method), 306	(openff.evaluator.protocols.miscellaneous.DummyProtocol method), 412
	ue_references()
(openff.evaluator.protocols.analysis.AverageObservable method), 295	(openff.evaluator.protocols.miscellaneous.FilterSubstanceByRole method), 408
	ue_references()
(openff.evaluator.protocols.analysis.BaseAverageObservable method), 290	
	ue_references()
(openff.evaluator.protocols.analysis.BaseDecorrelateProtocols.BaseDecorrelateProtocols.analysis.BaseDecorrelateProtocols.BaseDecorrel	
	ue_references()
-	n(copenff.evaluator.protocols.miscellaneous.WeightByMoleFraction method), 402
	ue_references()
	e&openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation method), 418
	ue_references()
	(openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies

method), 430	method), 520
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.openmm.OpenMMSin	nulation (openff.evaluator.protocols.simulation.BaseSimulation
method), 424	method), 527
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.paprika.analysis.Anal method), 472	lyzeAPRPh(aspenff.evaluator.protocols.storage.UnpackStoredSimulationData method), 533
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.paprika.analysis.Com	puteRefere(www.Mfrkvaluator.protocols.yank.BaseYankProtocol
method), 482	method), 540
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.paprika.analysis.Com	puteSymm <b>&amp;typ&amp;rdfi:eecdiwa</b> tor.protocols.yank.LigandReceptorYankProtocol
method), 477	method), 547
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.paprika.coordinates. An anti-control of the control	AddDummy(Atpensf.evaluator.protocols.yank.SolvationYankProtocol
method), 446	method), 555
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.paprika.coordinates.F method), 435	PreparePul <b>l&amp;ponffinaden</b> ator.workflow.Protocol method), 263
<pre>get_value_references()</pre>	<pre>get_value_references()</pre>
(openff.evaluator.protocols.paprika.coordinates.F	PrepareRel@ape@iff@relihattsr.workflow.ProtocolGroup
method), 440	method), 269
<pre>get_value_references()</pre>	<pre>gpu_device_indices (openff.evaluator.backends.ComputeResources</pre>
(openff.evaluator.protocols.paprika.restraints.App	plyRestrain <b>pr</b> operty), 205
method), 467	$\verb"gpu_device_indices" (open ff.evaluator.backends.Queue Worker Resources of the property o$
<pre>get_value_references()</pre>	property), 206
(openff.evaluator.protocols.paprika.restraints.Gen	ngnadAenthRenamenters
method), 451	(openff. evaluator. protocols. analysis. Compute Dipole Moments
<pre>get_value_references()</pre>	attribute), 309
(openff. evaluator. protocols. paprika. restraints. General and the protocol states and the protocol states are also become a protocol states and the protocol states are also become a protocol states are also	n <b>gnadPahiRepaniane</b> ters
method), 456	$(open {\it ff. evaluator. protocols. gradients.} Zero {\it Gradients}$
<pre>get_value_references()</pre>	attribute), 370
(open ff. evaluator. protocols. paprika. restraints. General contents for the protocol state of the protocol	
method), 462	$(open {\it ff. evaluator. protocols. open mm. Open MME valuate Energies}$
<pre>get_value_references()</pre>	attribute), 430
$(open {\it ff.evaluator.protocols.reweighting.} Base Eval$	
method), 498	(open ff. evaluator. protocols. open mm. Open MMS imulation
<pre>get_value_references()</pre>	attribute), 424
(open ff. evaluator. protocols. reweighting. Base MBA for the substitution of the su	
method), 504	(openff.evaluator.protocols.reweighting.BaseEvaluateEnergies
<pre>get_value_references()</pre>	attribute), 496
(openff.evaluator.protocols.reweighting.Concaten	
method), 492	(openff.evaluator.protocols.simulation.BaseSimulation
<pre>get_value_references()</pre>	attribute), 525
(openff.evaluator.protocols.reweighting.Concaten	
method), 488	(openff.evaluator.protocols.yank.BaseYankProtocol
<pre>get_value_references()</pre>	attribute), 537
(openff.evaluator.protocols.reweighting.Reweight	
method), 515	(openff.evaluator.protocols.yank.LigandReceptorYankProtocol
<pre>get_value_references()</pre>	attribute), 547
(openff.evaluator.protocols.reweighting.Reweight	
method), 509	(openff.evaluator.protocols.yank.SolvationYankProtocol
get_value_references()  (openff evaluator protocols simulation BaseEnero	attribute), 555 <b>cylliwinasi</b> on(openff.evaluator.datasets.PhysicalProperty
CODERII. E VALUALOT. DI OLOCOIS. SIMULALION. DASE ENETS	VALLONGARGEDIGEDOTA ODERTITE VALUATOT. AATASETS, ETTVSTCALETODETTV

attribute), 95	high_precision(openff.evaluator.protocols.openmm.OpenMMSimulation
gradients (openff.evaluator.properties.Density attribute), 102	attribute), 424
gradients (openff.evaluator.properties.DielectricConstant	high_precision(openff.evaluator.protocols.simulation.BaseSimulation attribute), 525
attribute), 109	host_coordinate_path
gradients (openff.evaluator.properties.EnthalpyOfMixing	
attribute), 113	attribute), 460
gradients (openff.evaluator.properties.EnthalpyOfVaporiatribute), 116	zHDon/GuestBindingAffinity (class in openff.evaluator.properties), 121
gradients (openff.evaluator.properties.ExcessMolarVolum	1 00 1 1
attribute), 106	
${\tt gradients} (\textit{openff.evaluator.properties.HostGuestBinding.} \\$	Afantipenff.evaluator.client.Request attribute), 82
attribute), 123	$\verb"id" (open ff. evaluator. datasets. Physical Property attribute),$
gradients (openff.evaluator.properties.SolvationFreeEner	
attribute), 119	id (openff.evaluator.properties.Density attribute), 102
gradients (openff.evaluator.workflow.WorkflowResult attribute), 260	id (openff.evaluator.properties.DielectricConstant attribute), 109
guest_orientation_mask	id (openff.evaluator.properties.EnthalpyOfMixing
$(open {\it ff.evaluator.protocols.paprika.coordinates.} Is a particular of the contract of the$	PreparePullGA@Bdjpqtex3
attribute), 433	id (openff.evaluator.properties.EnthalpyOfVaporization
Н	attribute), 116
	id (openff.evaluator.properties.ExcessMolarVolume at-
has_ancillary_data()	tribute), 106
	tad (openff.evaluator.properties.HostGuestBindingAffinity
class method), 227	attribute), 123
has_ancillary_data()	id (openff.evaluator.properties.SolvationFreeEnergy at-
(openff.evaluator.storage.data.BaseStoredData	tribute), 119
<pre>class method), 219 has_ancillary_data()</pre>	id (openff.evaluator.protocols.analysis.AverageDielectricConstant attribute), 301
(openff.evaluator.storage.data.ForceFieldData	id (openff.evaluator.protocols.analysis.AverageFreeEnergies
class method), 223	attribute), 307
has_ancillary_data()	id(openff.evaluator.protocols.analysis.AverageObservable
(openff.evaluator.storage.data.HashableStoredDa	
class method), 221	$\verb"id" (open ff. evaluator. protocols. analysis. Base Average Observable$
has_ancillary_data()	attribute), 290
	$\verb"id" (open ff. evaluator. protocols. analysis. Base Decorrelate Protocol$
class method), 225	attribute), 317
has_ancillary_data()	id (openff.evaluator.protocols.analysis.ComputeDipoleMoments
(openff.evaluator.storage.data.StoredFreeEnergy)	<i>"</i>
class method), 233	id (openff.evaluator.protocols.analysis.DecorrelateObservables
has_ancillary_data()	attribute), 327
	Dava (openff.evaluator.protocols.analysis.DecorrelateTrajectory
class method), 230	attribute), 322 t <b>àx</b> læpenff.evaluator.protocols.coordinates.BuildCoordinatesPackmol
method), 217	attribute), 333
	kich (openff.evaluator.protocols.coordinates.BuildDockedCoordinates
method), 216	attribute), 345
	id (openff.evaluator.protocols.coordinates.SolvateExistingStructure
method), 217	attribute), 339
has_object() (openff.evaluator.storage.StorageBackend method), 216	attribute), 350
HashableStoredData (class in	id (openff.evaluator.protocols.forcefield.BuildLigParGenSystem
openff.evaluator.storage.data), 221	attribute), 361

<pre>id(openff.evaluator.protocols.forcefield.BuildSmirnoffSyste</pre>	nid (openff.evaluator.protocols.reweighting.ConcatenateTrajectories attribute), 488
$\label{eq:condition} \begin{tabular}{l} id (open ff. evaluator. protocols. force field. Build TLeap System \\ attribute), 366 \end{tabular}$	$\verb id   (open ff. evaluator. protocols. reweighting. Reweight Dielectric Constant attribute), 515$
id (openff.evaluator.protocols.gradients.ZeroGradients attribute), 372	<pre>id(openff.evaluator.protocols.reweighting.ReweightObservable</pre>
id (openff.evaluator.protocols.groups.ConditionalGroup attribute), 378	<pre>id(openff.evaluator.protocols.simulation.BaseEnergyMinimisation</pre>
id (openff.evaluator.protocols.miscellaneous.AddValues attribute), 383	id (openff.evaluator.protocols.simulation.BaseSimulation attribute), 528
id (openff.evaluator.protocols.miscellaneous.DivideValue attribute), 398	<pre>id(openff.evaluator.protocols.storage.UnpackStoredSimulationData</pre>
id (openff.evaluator.protocols.miscellaneous.DummyProtoc attribute), 413	tribute), 540
attribute), 408	attribute), 547
id (openff.evaluator.protocols.miscellaneous.MultiplyValue attribute), 393	attribute), 555
$\label{eq:continuous} \textbf{id}  (\textit{openff.evaluator.protocols.miscellaneous.SubtractValue}                   $	id (openff.evaluator.workflow.Protocol attribute), 262
id (openff.evaluator.protocols.miscellaneous.WeightByMole attribute), 403	270
attribute), 418	riidu(saptenff.evaluator.workflow.schemas.ProtocolGroupSchema attribute), 275
id (openff.evaluator.protocols.openmm.OpenMMEvaluateE attribute), 430	attribute), 273
id (openff.evaluator.protocols.openmm.OpenMMSimulation attribute), 424	erty), 131
id (openff.evaluator.protocols.paprika.analysis.AnalyzeAPI attribute), 473	property), 130
id (openff.evaluator.protocols.paprika.analysis.ComputeRe attribute), 483	property), 133
id (openff.evaluator.protocols.paprika.analysis.ComputeSynattribute), 478	property), 135
id (openff.evaluator.protocols.paprika.coordinates.AddDun attribute), 446	property), 126
id (openff.evaluator.protocols.paprika.coordinates.Prepare attribute), 436	$open {\it ff. evaluator. datasets. curation. components. free solv)},$
id (openff.evaluator.protocols.paprika.coordinates.Prepare attribute), 441	ReleaseCobrainates ImportFreeSolvSchema (class in
${\tt id}(open f\!f\!. evaluator. protocols. paprika. restraints. Apply Restraints. Apply Restraints), 467$	raints openff.evaluator.datasets.curation.components.freesolv), 172
id (openff.evaluator.protocols.paprika.restraints.GenerateA attribute), 451	######################################
id (openff.evaluator.protocols.paprika.restraints.GeneratePattribute), 457	PullRestrainfs4 ImportThermoMLDataSchema (class in
id (openff.evaluator.protocols.paprika.restraints.GenerateRattribute), 462	CeleaseRest <b>opint</b> ff.evaluator.datasets.curation.components.thermoml), 173
<pre>id(openff.evaluator.protocols.reweighting.BaseEvaluateEn attribute), 498</pre>	eirgdex() (openff.evaluator.storage.attributes.FilePath method), 250
${\tt id}(open f\!f\!. evaluator. protocols. reweighting. Base MBAR Protocols. reweighting. Reweig$	oFnlequalityMergeBehaviour (class in openff.evaluator.workflow.attributes), 281
<pre>id(openff.evaluator.protocols.reweighting.ConcatenateObs</pre>	<pre>cimputescoordinate_file</pre>
,	Spengie in the control of the contro

attribute), 319	method), 250
<pre>input_coordinate_file</pre>	isdigit() (openff.evaluator.storage.attributes.FilePath
(openff.evaluator.protocols.openmm.OpenMMEn	
attribute), 418	isidentifier() (openff.evaluator.storage.attributes.FilePath
input_coordinate_file	method), 250
<del>-</del>	niustioner() (openff.evaluator.storage.attributes.FilePath
	method), 250
<pre>attribute), 424 input_coordinate_file</pre>	
<del>-</del>	isnumeric() (openff.evaluator.storage.attributes.FilePath
(openff.evaluator.protocols.simulation.BaseEnerg	
attribute), 518	isprintable() (openff.evaluator.storage.attributes.FilePath
<pre>input_coordinate_file</pre>	method), 251
	laisspace() (openff.evaluator.storage.attributes.FilePath
attribute), 525	method), 251
input_coordinate_path	istitle() (openff.evaluator.storage.attributes.FilePath
(openff.evaluator.protocols.paprika.coordinates.A	
attribute), 444	isupper() (openff.evaluator.storage.attributes.FilePath
<pre>input_coordinate_paths</pre>	method), 251
(openff.evaluator.protocols.reweighting.Concater	n <b>aiteEms&amp;At@jæs</b> nff.evaluator.utils.observables.ObservableFrame
attribute), 486	method), 570
$\verb"input_observables" (open \textit{ff.evaluator.protocols.analysis}.$	DecorrelateObservables
attribute), 324	J
<pre>input_observables(openff.evaluator.protocols.gradient.</pre>	s.Zeto_Gradiste() (openff.evaluator.backends.dask.BaseDaskJobQueueBacke
attribute), 369	method), 209
	irss Concurrence (Performent Section 2007)
attribute), 490	method), 212
	memod), 212 อนุราธิ <u>il</u> ระรั <b>บ</b> ธุรเตรเตรายะ (epekglevaluator.backends.dask.DaskPBSBackend
attribute), 405	method), 214
$\verb"input_system" (open ff. evaluator. protocols. paprika. coording the control of the control o$	methou), 214
attribute), 444	method), 251
	memoa), 251 nt <b>yAnyly Rsperiji.e</b> \$aluator.utils.observables.ObservableArray
attribute), 465	1 1 1 5 60
input_trajectory_path	class method), 568
	join() (openff.evaluator.utils.observables.ObservableFrame
(openff.evaluator.protocols.analysis.Decorrelate	
attribute), 319	json() (openff.evaluator.attributes.AttributeClass
<pre>input_trajectory_paths</pre>	method), 565
(openff.evaluator.protocols.reweighting.Concater	nafeTragectories(openff.evaluator.client.ConnectionOptions
attribute), 486	method), 81
$\verb"input_value" (open ff. evaluator. protocols. miscellane ous. D$	นทุษอน (ทุงเอรอกff.evaluator.client.Request method), 83
attribute), 410	<pre>json() (openff.evaluator.client.RequestOptions method),</pre>
InputAttribute (class in	85
$open {\it ff. evaluator.} work {\it flow. attributes}), 282$	<pre>json() (openff.evaluator.client.RequestResult method),</pre>
${\tt inputs} \ (open {\it ff.evaluator.work flow.schemas. Protocol Group} \\$	Schema 87
attribute), 275	json() (openff.evaluator.datasets.CalculationSource
${\tt inputs} \ (open {\it ff.evaluator.work flow.schemas. Protocol Schemas}. Protocol Schemas. Protocol $	na method), 98
attribute), 273	ison() (openff.evaluator.datasets.MeasurementSource
$\verb"inverse_beta" (open ff. evaluator. the rmodynamics. Thermodynamics. The rmodynamics and the state of the $	dynamicStateethod), 99
property), 137	json() (openff.evaluator.datasets.PhysicalProperty
isalnum() (openff.evaluator.storage.attributes.FilePath	method), 95
method), 250	json() (openff.evaluator.datasets.PhysicalPropertyDataSet
isalpha() (openff.evaluator.storage.attributes.FilePath	method), 141
method), 250	json() (openff.evaluator.datasets.Source method), 97
isascii() (openff.evaluator.storage.attributes.FilePath	
method), 250	json() (openff.evaluator.datasets.taproom.TaproomDataSet
isdecimal() (openff.evaluator.storage.attributes.FilePath	method), 149
13accimar() (openy.evananoi.snoiage.anrionies.rnerani	

- json() (openff.evaluator.datasets.thermoml.ThermoMLDat**ġSet**n() (openff.evaluator.protocols.coordinates.SolvateExistingStructure method), 144 method), 339
- json() (openff.evaluator.forcefield.ForceFieldSource json() (openff.evaluator.protocols.forcefield.BaseBuildSystem method), 181 method), 350
- json() (openff.evaluator.forcefield.LigParGenForceFieldSojuson() (openff.evaluator.protocols.forcefield.BuildLigParGenSystem method), 186 method), 361
- json() (openff.evaluator.forcefield.SmirnoffForceFieldSourgson() (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem method), 183 method), 355
- json() (openff.evaluator.forcefield.TLeapForceFieldSourcejson() (openff.evaluator.protocols.forcefield.BuildTLeapSystem method), 184
  method), 367
- json() (openff.evaluator.layers.CalculationLayerResult json() (openff.evaluator.protocols.gradients.ZeroGradients method), 190 method), 372
- json() (openff.evaluator.layers.CalculationLayerSchema json() (openff.evaluator.protocols.groups.ConditionalGroup method), 192
  method), 378
- json() (openff.evaluator.layers.reweighting.ReweightingScljeson() (openff.evaluator.protocols.groups.ConditionalGroup.Condition method), 202
  method), 375
- json() (openff.evaluator.layers.simulation.SimulationScherijson() (openff.evaluator.protocols.miscellaneous.AddValues method), 198 method), 383
- json() (openff.evaluator.layers.workflow.WorkflowCalculat**iyack())** (upenff.evaluator.protocols.miscellaneous.DivideValue method), 196 method), 398
- json() (openff.evaluator.properties.Density method), json() (openff.evaluator.protocols.miscellaneous.DummyProtocol method), 413
- json() (openff.evaluator.properties.DielectricConstant json() (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRole

method), 408

method), 483

json() (openff.evaluator.properties.EnthalpyOfMixing json() (openff.evaluator.protocols.miscellaneous.MultiplyValue method), 113 method), 393

method), 109

method), 295

- json() (openff.evaluator.properties.EnthalpyOfVaporizationjson() (openff.evaluator.protocols.miscellaneous.SubtractValues method), 116 method), 388
- method), 116 method), 388 json() (openff.evaluator.properties.ExcessMolarVolume json() (openff.evaluator.protocols.miscellaneous.WeightByMoleFraction
- method), 106 method), 403
  json() (openff.evaluator.properties.HostGuestBindingAffinifson() (openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation method), 123 method), 418
- json() (openff.evaluator.properties.SolvationFreeEnergy json() (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies method), 119 method), 430
- json() (openff.evaluator.protocols.analysis.AverageDielectris6tr()st(openff.evaluator.protocols.openmm.OpenMMSimulation method), 301 method), 424
- json() (openff.evaluator.protocols.analysis.AverageFreeEnergistics() (openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase
- method), 307 method), 473 json() (openff.evaluator.protocols.analysis.AverageObserv**glske**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.analysis.ComputeReferenceWorld (openff.evaluator.protocols.anal
- json() (openff.evaluator.protocols.analysis.BaseAverageObjsondDleopenff.evaluator.protocols.paprika.analysis.ComputeSymmetryCommethod), 290 method), 478
- json() (openff.evaluator.protocols.analysis.BaseDecorrelat**yPont(): (b**penff.evaluator.protocols.paprika.coordinates.AddDummyAtoms
- method), 317 method), 446 json() (openff.evaluator.protocols.analysis.ComputeDipole**json()** wopenff.evaluator.protocols.paprika.coordinates.PreparePullCoordinat
- method), 312 method), 436 json() (openff.evaluator.protocols.analysis.DecorrelateObs**jsson(**)(expenff.evaluator.protocols.paprika.coordinates.PrepareReleaseCo
- method), 327 method), 441

  json() (openff.evaluator.protocols.analysis.DecorrelateTrajeson()) (openff.evaluator.protocols.paprika.restraints.ApplyRestraints
- method), 322 method), 467
  json() (openff.evaluator.protocols.coordinates.BuildCoordinates.B

json()	(openff.evaluator.protocols.paprika.restraints.Gene method), 457		Restetlints), 237 (openff.evaluator.substances.Amount method),
json()	$(open {\it ff. evaluator. protocols. paprika. restraints. General constraints and the protocols of the proto$	erateRelea	askRestraints
ison()	method), 462 (openff.evaluator.protocols.reweighting.BaseEvalu	json() ateEnergi	(openff.evaluator.substances.Component esnethod), 130
3 17	method), 498	json()	(openff.evaluator.substances.ExactAmount
<pre>json()</pre>	$(open {\it ff.evaluator.protocols.reweighting.} Base {\it MBAB} \\$		
	method), 504	json()	(openff.evaluator.substances.MoleFraction
<pre>json()</pre>	(openff.evaluator.protocols.reweighting.Concatena		
ison()	method), 493 (openff.evaluator.protocols.reweighting.Concatena		(openff.evaluator.substances.Substance method),
J3011()	method), 488	-	(openff.evaluator.thermodynamics.ThermodynamicState
json()	(openff.evaluator.protocols.reweighting.ReweightD		
	method), 515		(openff.evaluator.utils.exceptions.EvaluatorException
json()	(open ff. evaluator. protocols. reweighting. Reweight Open ff. evaluator. protocols. protocols		
	method), 509	json()	* * **
<pre>json()</pre>	(openff.evaluator.protocols.simulation.BaseEnergy		
ison()	method), 521 (openff.evaluator.protocols.simulation.BaseSimula.	json()	(openff.evaluator.workflow.ProtocolGroup method), 270
J3011()	method), 528		(openff.evaluator.workflow.schemas.ProtocolGroupSchema
json()	(openff.evaluator.protocols.storage.UnpackStoredS		
	method), 534		openff.evaluator.workflow.schemas.ProtocolReplicator
<pre>json()</pre>	(open ff. evaluator. protocols. yank. Base Yank Protocols. yank. Yank Protoc		method), 278
	method), 540		(openff.evaluator.workflow.schemas.ProtocolSchema
json()	(openff.evaluator.protocols.yank.LigandReceptorYo		
ison()	method), 547 (openff.evaluator.protocols.yank.SolvationYankPro		(openff.evaluator.workflow.schemas.WorkflowSchema method), 280
13011()	method), 556	json()	(openff.evaluator.workflow.WorkflowException
json()	(openff.evaluator.server.Batch method), 92	10011()	method), 257
	(openff.evaluator.storage.data.BaseSimulationData	ijson()	(openff.evaluator.workflow.WorkflowResult
	method), 227		method), 260
json()	. 1 00	K	
	method), 220		
json()	(openff.evaluator.storage.data.ForceFieldData method), 223	keys()	(openff.evaluator.utils.observables.ObservableFrame
ison()	(openff.evaluator.storage.data.HashableStoredData	q	method), 570
10011()	method), 221	L	
<pre>json()</pre>	(openff.evaluator.storage.data.ReplaceableData	last nr	cotocol (openff.evaluator.workflow.utils.ProtocolPath
	method), 225	_	1) 205
json()	(openff.evaluator.storage.data.StoredFreeEnergyDe	<i>ata</i> Teap_so	purce (openff.evaluator.forcefield.TLeapForceFieldSource
	memou j, 255		man aut.) 101
json()	(openff.evaluator.storage.aata.storeaSimulationDa method), 230	<sup>16</sup> left_ha	property), 184 and_value(openff.evaluator.protocols.groups.ConditionalGroup.C
json()	memoa), 250		attribute), 375
30011()	method), 236	ligand_	_electrostatic_lambdas
<pre>json()</pre>	(openff.evaluator.storage.query.BaseSimulationDa	taQuery	(openff.evaluator.protocols.yank.LigandReceptorYankProtocol attribute), 544
	method), 241	ligand	residue_name
json()	(openff.evaluator.storage.query.ForceFieldQuery	J	(openff.evaluator.protocols.coordinates.BuildDockedCoordinates
·()	method), 239		attribute), 343
Json()	(openff.evaluator.storage.query.FreeEnergyDataQuethod), 246	"1"gand_	
ison()	(openff.evaluator.storage.query.SimulationDataQu	erv	(openff.evaluator.protocols.yank.LigandReceptorYankProtocol
5()	method), 243		attribute), 543 _steric_lambdas
<pre>json()</pre>	$(open f\!f. evaluator. storage. query. Substance Query$	119anu_	(openff.evaluator.protocols.yank.LigandReceptorYankProtocol
			1 3

attribute), 544		(openff.evaluator.protocols.analysis.AverageObservable
${\tt ligand\_substance} \ (open \textit{ff.evaluator.protocols.coordinate}) \\$		
attribute), 342		(openff.evaluator.protocols.analysis.BaseAverageObservable
LigandReceptorYankProtocol (class in		method), 290
openff.evaluator.protocols.yank), 541	_	(openff.evaluator.protocols.analysis.BaseDecorrelateProtocol
LigandReceptorYankProtocol.RestraintType		method), 317
(class in openff.evaluator.protocols.yank), 543	_	(openff.evaluator.protocols.analysis.ComputeDipoleMoments
LigParGenForceFieldSource (class in		method), 312
openff.evaluator.forcefield), 185 LigParGenForceFieldSource.ChargeModel (class in	_	(openff.evaluator.protocols.analysis.DecorrelateObservables method), 327
openff.evaluator.forcefield), 186		(openff.evaluator.protocols.analysis.DecorrelateTrajectory
ljust() (openff.evaluator.storage.attributes.FilePath		method), 322
method), 251		(openff.evaluator.protocols.coordinates.BuildCoordinatesPackm
load_restraints() (openff.evaluator.protocols.paprika.		
class method), 465		(openff.evaluator.protocols.coordinates.BuildDockedCoordinate.
LocalFileStorage (class in openff.evaluator.storage),	_	method), 345
217		(openff.evaluator.protocols.coordinates.SolvateExistingStructure
lower() (openff.evaluator.storage.attributes.FilePath		method), 339
method), 251	merge()	(openff.evaluator.protocols.forcefield.BaseBuildSystem
<pre>lstrip() (openff.evaluator.storage.attributes.FilePath</pre>		method), 351
method), 251	merge()	(openff.evaluator.protocols.forcefield.BuildLigParGenSystem
M		method), 361
M		(openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
$\verb maketrans()  (open ff. evaluator. storage. attributes. File Path) $	,	method), 355
static method), 251		(openff.evaluator.protocols.forcefield.BuildTLeapSystem
mass_density (openff.evaluator.protocols.coordinates.Bu attribute), 330	<pre>ildCoordin   merge()</pre>	MeSPACkm61 (openff.evaluator.protocols.gradients.ZeroGradients
mass_density(openff.evaluator.protocols.coordinates.Soc	lvateExistir	rysthachore72
attribute), 339	merge()	(openff.evaluator.protocols.groups.ConditionalGroup
$\verb max_iterations   (open \textit{ff.evaluator.protocols.groups.Conditions}   (open ff.evaluator.protocols.groups.Conditions.groups.Conditions.groups.Conditions.groups.Conditions.groups.Conditions.groups.Conditions.groups.Conditions.groups$	litionalGro	method), 376
attribute), 376	merge()	(openff.evaluator.protocols.miscellaneous.AddValues
$\verb max_iterations  (open ff. evaluator. protocols. open mm. Open for the context of the context$	enMMEner	gyMinahisation
attribute), 418		(openff.evaluator.protocols.miscellaneous.DivideValue
max_iterations (openff.evaluator.protocols.simulation.B	BaseEnergy.	MHMMsatiଟନ (openff.evaluator.protocols.miscellaneous.DummyProtocol
attribute), 518		
max_molecules (openff.evaluator.protocols.coordinates.B	merge()	กนพรรคมหาดเ (openff.evaluator.protocols.miscellaneous.FilterSubstanceByRol
attribute), 330 max_molecules (openff.evaluator.protocols.coordinates.Sc		
attribute), 339	merae()	(openff.evaluator.protocols.miscellaneous.MultiplyValue
maximum_data_points		method), 393
(onenff evaluator layers reweighting Reweighting	merge()	(openff.evaluator.protocols.miscellaneous.SubtractValues
attribute), 201	Denerate v	method), 388
MeasurementSource (class in		(openff.evaluator.protocols.miscellaneous.WeightByMoleFractio
openff.evaluator.datasets), 99		method), 403
merge() (openff.evaluator.datasets.PhysicalPropertyDatasets.PhysicalPro	gmerge()	(openff.evaluator.protocols.openmm.OpenMMEnergyMinimisati
method), 139		method), 418
merge() (openff.evaluator.datasets.taproom.TaproomData	smerge()	(openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies
method), 149		method), 430
merge() (openff.evaluator.datasets.thermoml.ThermoMLD	marge()	(openff.evaluator.protocols.openmm.OpenMMSimulation
method), 144		method), 424
$\verb merge()  (open ff. evaluator. protocols. analysis. Average Dielection (a) the state of the s$	૯ <b>ભગસ્ ઉ</b> ષ્ઠિમકે દ	Appenff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase
method), 301		method), 473
merge() (openff.evaluator.protocols.analysis.AverageFree method), 307	EMERGIES )	(openff.evaluator.protocols.paprika.analysis.ComputeReferenceV method), 483

```
merge() (openff.evaluator.protocols.paprika.analysis.CompwexSydvarbathyyCanffectaidomator.properties.EnthalpyOfVaporization
         method), 478
                                                                   attribute), 117
merge() (openff.evaluator.protocols.paprika.coordinates.AdulDadutxA(topenff.evaluator.properties.ExcessMolarVolume
         method), 447
                                                                   attribute), 106
merge() (openff.evaluator.protocols.paprika.coordinates.PmptredPaddCopedffiataduator.properties.HostGuestBindingAffinity
         method), 436
                                                                   attribute), 124
merge() (openff.evaluator.protocols.paprika.coordinates.Pmpt.ededaaspenffrebalussor.properties.SolvationFreeEnergy
         method), 441
                                                                   attribute), 120
merge() (openff.evaluator.protocols.paprika.restraints.AppMoLeGradion (class in openff.evaluator.substances),
         method), 468
merge() (openff.evaluator.protocols.paprika.restraints.GenrosstAiradoRrosstAiratiors() (openff.evaluator.storage.data.BaseSimulationData
         method), 452
                                                                   class method), 227
merge() (openff.evaluator.protocols.paprika.restraints.GenmosaPaillRosmaition() (openff.evaluator.storage.data.ReplaceableData
                                                                   class method), 224
         method), 457
merge() (openff.evaluator.protocols.paprika.restraints.GenrosstRålndisathastikan(s) (openff.evaluator.storage.data.StoredFreeEnergyDa
         method), 462
                                                                   class method), 232
merge() (openff.evaluator.protocols.reweighting.BaseEvalumosfineinghormation() (openff.evaluator.storage.data.StoredSimulationDat
         method), 498
                                                                   class method), 229
merge() (openff.evaluator.protocols.reweighting.BaseMBAINDLotippli er (openff.evaluator.protocols.miscellaneous.MultiplyValue
         method), 504
                                                                   attribute), 391
merge() (openff.evaluator.protocols.reweighting.Concatend Mara thing ly the law
                                                                                         (class
                                                                                                              in
         method), 493
                                                                   openff.evaluator.protocols.miscellaneous),
merge() (openff.evaluator.protocols.reweighting.ConcatenateTrajectortes
         method), 488
merge() (openff.evaluator.protocols.reweighting.ReweightDielectricConstant
         method), 515
                                                         n\_microstates (openff.evaluator.protocols.paprika.analysis.ComputeSym
merge() (openff.evaluator.protocols.reweighting.ReweightObservableattribute), 475
         method), 509
                                                         n_pull_windows (openff.evaluator.protocols.paprika.coordinates.Prepare.
merge() (openff.evaluator.protocols.simulation.BaseEnergyMinimisationibute), 433
         method), 521
                                                         n_pull_windows (openff.evaluator.protocols.paprika.restraints.GenerateP
merge() (openff.evaluator.protocols.simulation.BaseSimulation
                                                                   attribute), 454
         method), 528
                                                         number_of_components
{\tt merge}() (openff. evaluator. protocols. storage. Unpack Stored Simulation {\tt Papen} ff. evaluator. substances. Substance
         method), 534
                                                                   erty), 126
merge() (openff.evaluator.protocols.yank.BaseYankProtocohumber_of_equilibration_iterations
         method), 540
                                                                   (openff.evaluator.protocols.yank.BaseYankProtocol
merge() (openff.evaluator.protocols.yank.LigandReceptorYankProtocaltribute), 537
         method), 547
                                                         number_of_equilibration_iterations
merge() (openff.evaluator.protocols.yank.SolvationYankProtocol
                                                                   (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
         method), 556
                                                                   attribute), 547
merge() (openff.evaluator.workflow.Protocol method),
                                                        number_of_equilibration_iterations
         263
                                                                   (openff.evaluator.protocols.yank.SolvationYankProtocol
              (openff.evaluator.workflow.ProtocolGroup
merge()
                                                                   attribute), 556
         method), 269
                                                         number\_of\_gpus (openff.evaluator.backends.ComputeResources
MergeBehaviour
                                (class
                                                     in
                                                                   property), 205
         openff.evaluator.workflow.attributes), 281
                                                         number_of_gpus (openff.evaluator.backends.QueueWorkerResources
metadata
            (openff.evaluator.datasets.PhysicalProperty
                                                                   property), 206
         attribute), 94
                                                         number_of_iterations
metadata (openff.evaluator.properties.Density attribute),
                                                                   (openff.evaluator.protocols.yank.BaseYankProtocol
                                                                   attribute), 537
metadata(openff.evaluator.properties.DielectricConstant number_of_iterations
         attribute), 110
                                                                   (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
metadata (openff.evaluator.properties.EnthalpyOfMixing
                                                                   attribute), 548
         attribute), 113
                                                         number_of_iterations
```

```
(openff.evaluator.protocols.yank.SolvationYankProtocol
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 518
                                                               attribute), 556
                                                                                                                                                                                                                                                                                                                                                                                                        output_coordinate_file
number_of_ligand_conformers
                                                                                                                                                                                                                                                                                                                                                                                                                                                                          (openff.evaluator.protocols.simulation.BaseSimulation
                                                                  (openff.evaluator.protocols.coordinates.BuildDockedCoordinatthalbute), 525
                                                                 attribute), 342
                                                                                                                                                                                                                                                                                                                                                                                                        output_coordinate_path
number_of_molecules
                                                                                                                                                                                                                                                                                                                                                                                                                                                                          (openff.evaluator.protocols.paprika.coordinates.AddDummyAtom
                                                                  (openff.evaluator.storage.data.StoredSimulationData
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 444
                                                                                                                                                                                                                                                                                                                                                                                                        output_coordinate_path
                                                               attribute), 229
number_of_molecules
                                                                                                                                                                                                                                                                                                                                                                                                                                                                          (openff.evaluator.protocols.paprika.coordinates.PreparePullCoor
                                                                 (openff.evaluator.storage.query.SimulationDataQuery
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 436
                                                                 attribute), 243
                                                                                                                                                                                                                                                                                                                                                                                                        output_coordinate_path
number\_of\_threads. \textit{ComputeResources} (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareReleaseComputeResources}) (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareReleaseComputeResources})) (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareResources})) (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareResources})) (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareResources})) (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareResources})) (\textit{openff.evaluator.protocols.paprika.coordinates.PrepareResources}) (\textit{openff.evaluator.protocol
                                                               property), 205
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 441
number_of_threads(openff.evaluator.backends.QueueWookerfletopcoordinate_path
                                                               property), 207
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         (open ff. evaluator. protocols. reweighting. Concatenate Trajectories
                                                                                                                                                                                                                                                                                                                                                                                                                                                                          attribute), 486
O
                                                                                                                                                                                                                                                                                                                                                                                                      output_frequency (openff.evaluator.protocols.openmm.OpenMMSimulat
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 425
 Observable (class in openff.evaluator.utils.observables),
                                                                                                                                                                                                                                                                                                                                                                                                        \verb"output_frequency" (open ff. evaluator. protocols. simulation. Base Simulation for the substitution of 
observable ({\it open ff. evaluator. protocols. analysis. Average Observable\ attribute}), 524
                                                                                                                                                                                                                                                                                                                                                                                                        output_number_of_molecules
                                                                  attribute), 293
observable (\it open ff. evaluator. protocols. reweighting. Reweight Observable (\it open ff. evaluator. protocols. coordinates. Build Coordinates Packmonton for the contraction of the c
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 331
                                                                 attribute), 506
                                                                                                                                                                                                                                                                                                                                                                                                        output_number_of_molecules
ObservableArray
                                                                                                                                                                                                                                  (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         (openff.evaluator.protocols.coordinates.SolvateExistingStructure
                                                                  openff.evaluator.utils.observables), 567
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 339
ObservableFrame
                                                                                                                                                                                                                                   (class
                                                                                                                                                                                                                                                                                                                                                                          in
                                                                                                                                                                                                                                                                                                                                                                                                        \verb"output_observables" (open ff. evaluator. protocols. analysis. Decorrelate Observables (open ff. evaluator. protocols. analysis. Decorrelate Observables) (open ff. evaluator. protocols. protoco
                                                                 openff.evaluator.utils.observables), 569
observables (\it open ff. evaluator. protocols. open mm. Open MMS imulation that the best of the control of the
                                                                                                                                                                                                                                                                                                                                                                                                        output_observables(openff.evaluator.protocols.gradients.ZeroGradient.
                                                                  attribute), 425
observables ({\it open ff. evaluator. protocols. simulation. Base Simulation\ attribute}), 370
                                                                                                                                                                                                                                                                                                                                                                                                        output_observables(openff.evaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.OpenMMEvaluator.protocols.openmm.Openmm.OpenMMEvaluator.protocols.openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.Openmm.O
                                                                 attribute), 526
observables (openff.evaluator.protocols.storage.UnpackStoredSimulativihybeita 430
                                                                                                                                                                                                                                                                                                                                                                                                        \verb"output_observables" (open ff. evaluator. protocols. reweighting. Base Evaluation for the protocols of th
                                                                  attribute), 531
observables (openff.evaluator.storage.data.StoredSimulationData attribute), 496
                                                                                                                                                                                                                                                                                                                                                                                                        output_observables (openff.evaluator.protocols.reweighting.Concatenat
                                                               attribute), 229
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 491
ObservableType
                                                                                                                                                                                                                              (class
                                                                                                                                                                                                                                                                                                                                                                          in
                                                                                                                                                                                                                                                                                                                                                                                                        output_substance(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.B
                                                                  openff.evaluator.utils.observables), 568
offset (openff.evaluator.protocols.paprika.coordinates.AddDummyAftthisbute), 331
                                                                                                                                                                                                                                                                                                                                                                                                        output_substance (openff.evaluator.protocols.coordinates.SolvateExistin
                                                               attribute), 444
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 339
OpenMMEnergyMinimisation
                                                                                                                                                                                                                                                                        (class
                                                                                                                                                                                                                                                                                                                                                                          in
                                                                                                                                                                                                                                                                                                                                                                                                        \verb"output_system" (open ff. evaluator. protocols. paprika. coordinates. Add Dumner for the protocols of the
                                                                 openff.evaluator.protocols.openmm), 415
                                                                                                                                                                                                                                                                                                                                                                                                                                                                          attribute), 444
OpenMMEvaluateEnergies
                                                                                                                                                                                                                                                                (class
                                                                                                                                                                                                                                                                                                                                                                         in
                                                                                                                                                                                                                                                                                                                                                                                                      \verb"output_system" (open ff. evaluator. protocols. paprika. restraints. Apply Restra
                                                                 openff.evaluator.protocols.openmm), 427
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 465
OpenMMSimulation
                                                                                                                                                                                                                                       (class
                                                                                                                                                                                                                                                                                                                                                                          in
                                                                                                                                                                                                                                                                                                                                                                                                        output_trajectory_path
                                                                  openff.evaluator.protocols.openmm), 420
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         (openff.evaluator.protocols.analysis.DecorrelateTrajectory
options (openff.evaluator.server.Batch attribute), 91
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 320
output_coordinate_file
                                                                  (open ff. evaluator. protocols. reweighting. Concatenate Trajectories
                                                               attribute), 418
                                                                                                                                                                                                                                                                                                                                                                                                                                                                         attribute), 486
output_coordinate_file
                                                                 (open \textit{ff.evaluator.protocols.openmm.} Open \textit{MMSimputpyt-} \textbf{value} (open \textit{ff.evaluator.protocols.miscellaneous.} Dummy \textit{Protocols.openmm.} \textbf{Open \textit{MMSimputpyt-}} \textbf{value} (open \textit{ff.evaluator.protocols.miscellaneous.} \textbf{Dummy Protocols.} \textbf{Open \textit{ff.evaluator.protocols.miscellaneous.} \textbf{Dummy Protocols.} \textbf{Open \textit{ff.evaluator.protocols.}} \textbf{Open \textit{ff.evaluator.}} \textbf{Open \textit{ff.evaluator.protocols.}} \textbf{Open \textit{ff.evaluator.proto
                                                                                                                                                                                                                                                                                                                                                                                                                                                                        attribute), 410
                                                               attribute), 425
                                                                                                                                                                                                                                                                                                                                                                                                      OutputAttribute
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 in
output_coordinate_file
```

(open ff.evaluator.protocols.simulation. Base EnergyMinimisa PR ff.evaluator.workflow.attributes), 282

```
outputs (openff.evaluator.protocols.analysis.AverageDielecontronts(openff.evaluator.protocols.paprika.analysis.AnalyzeAPRPhase property), 301

property), 473

outputs (openff.evaluator.protocols.analysis.AverageFraeFoutpits (openff.evaluator.protocols.paprika.analysis.ComputePeferance)
```

- outputs (openff.evaluator.protocols.analysis.AverageFreeEnergiests (openff.evaluator.protocols.paprika.analysis.ComputeReferenceWebsites), 307 property), 483
- outputs (openff.evaluator.protocols.analysis.AverageObseroublputs (openff.evaluator.protocols.paprika.analysis.ComputeSymmetryCoproperty), 295 property), 478
- outputs (openff.evaluator.protocols.analysis.BaseAverageObxeputls (openff.evaluator.protocols.paprika.coordinates.AddDummyAtomsproperty), 290 property), 447
- outputs (openff.evaluator.protocols.analysis.BaseDecorrelovePputos (openff.evaluator.protocols.paprika.coordinates.PreparePullCoordinates.property), 317 property), 436
- outputs (openff.evaluator.protocols.analysis.ComputeDipol**eMpwes**(openff.evaluator.protocols.paprika.coordinates.PrepareReleaseC property), 312 property), 441
- outputs (openff.evaluator.protocols.analysis.DecorrelateOloxetpuble (openff.evaluator.protocols.paprika.restraints.ApplyRestraints property), 327 property), 468
- outputs (openff.evaluator.protocols.analysis.DecorrelateTrajetparys (openff.evaluator.protocols.paprika.restraints.GenerateAttachRest property), 322 property), 452
- outputs (openff.evaluator.protocols.coordinates.BuildCoordinatesBuildCoordinat
- outputs (openff.evaluator.protocols.coordinates.BuildDockodCpatdinapesnff.evaluator.protocols.paprika.restraints.GenerateReleaseRe.

  property), 345

  property), 462
- outputs (openff.evaluator.protocols.coordinates.SolvateExistingStrsc(nopenff.evaluator.protocols.reweighting.BaseEvaluateEnergies property), 339 property), 499
- outputs (openff.evaluator.protocols.forcefield.BaseBuildSystemputs (openff.evaluator.protocols.reweighting.BaseMBARProtocol property), 351 property), 504
- property), 351 property), 504
  outputs (openff.evaluator.protocols.forcefield.BuildLigParGertSyxtsn(openff.evaluator.protocols.reweighting.ConcatenateObservables property), 361 property), 493
- outputs (openff.evaluator.protocols.forcefield.BuildSmirnoffisitsperts (openff.evaluator.protocols.reweighting.ConcatenateTrajectories property), 355 property), 488

outputs (openff.evaluator.protocols.forcefield.BuildTLeapSoxteputs (openff.evaluator.protocols.reweighting.ReweightDielectricConsta

- property), 367 property), 515
  outputs (openff.evaluator.protocols.gradients.ZeroGradientsutputs (openff.evaluator.protocols.reweighting.ReweightObservable
- property), 372 property), 509

  outputs (openff.evaluator.protocols.groups.ConditionalGrouptputs (openff.evaluator.protocols.simulation.BaseEnergyMinimisation
- $property), 378 \\ property), 521 \\ \text{outputs} \ (open \! f\! f\! .evaluator. protocols. miscellaneous. Add Values \it tputs} \ (open \! f\! f\! .evaluator. protocols. simulation. Base Simulation$
- property), 384 property), 528

  outputs (openff.evaluator.protocols.miscellaneous.DivideVoluteputs (openff.evaluator.protocols.storage.UnpackStoredSimulationData
- property), 398 property), 534 outputs (openff.evaluator.protocols.miscellaneous.DummyBratiputs (openff.evaluator.protocols.yank.BaseYankProtocol
- property), 413 property), 540

  outputs (openff.evaluator.protocols.miscellaneous.FilterSubsutmatB(Rplenff.evaluator.protocols.yank.LigandReceptorYankProtocol
- property), 408 property), 548 outputs (openff.evaluator.protocols.miscellaneous.Multiply**dialpe**ts (openff.evaluator.protocols.yank.SolvationYankProtocol
- property), 393 property), 556
  outputs (openff.evaluator.protocols.miscellaneous.SubtractNulpests (openff.evaluator.workflow.Protocol property), 389
  262
- outputs (openff.evaluator.protocols.miscellaneous.WeightByWfplaFsaction(openff.evaluator.workflow.ProtocolGroup property), 403 property), 268
- outputs (openff.evaluator.protocols.openmm.OpenMMEnergythinististatione (openff.evaluator.workflow.schemas.WorkflowSchema property), 418

  attribute), 279
- $outputs (open ff. evaluator. protocols. open mm. Open MME valuator fires: \verb|gtex|| store|| (open ff. evaluator. work flow. Work flow property), 430 property), 255$
- outputs (openff.evaluator.protocols.openmm.OpenMMSimulation property), 425

P	class method), 100
parameter_gradient_keys	<pre>parse_json() (openff.evaluator.datasets.PhysicalProperty</pre>
	attribute), class method), 95
91	parse_json() (openff.evaluator.datasets.PhysicalPropertyDataSet
ParameterGradient (class	in class method), 141
openff.evaluator.forcefield), 188	parse_json() (openff.evaluator.datasets.Source class
ParameterGradientKey (class	in method), 97
openff.evaluator.forcefield), 187	$\verb"parse_json()" (open ff. evaluator. datasets. taproom. Taproom Data Set$
namamatani nad arratam	class method), 149
(openff.evaluator.protocols.analysis.C attribute), 309	ComputeDipoleMoments () (openff.evaluator.datasets.taproom.TaproomSource class method), 151
parameterized_system	$\verb"parse_json()" (open ff. evaluator. datasets. the rmoml. The rmoMLD at a Set$
(openff.evaluator.protocols.forcefield.	BaseBuildSystem class method), 144
attribute), 348	<pre>parse_json() (openff.evaluator.forcefield.ForceFieldSource</pre>
naramotorized system	class method), 181
(openff.evaluator.protocols.forcefield.	BuildLigParGenSystem() (openff.evaluator.forcefield.LigParGenForceFieldSource
attribute), 361	ciass methoa), 187
parameterized system	$\verb"parse_json()" (open ff. evaluator. force field. Smirn off Force Field Source")$
(openff.evaluator.protocols.forcefield.	BuildSmirnoffSystem class method), 183
attribute), 356	par se_json() (openjj.evananor.jorcejteta.rLeaprorcertetasource
parameterized_system	class method), 184
(openff.evaluator.protocols.forcefield.	BuildTLeap Narse_json() (openff.evaluator.layers.CalculationLayerResult
attribute), 367	ctass method), 191
parameterized_system	parse_json() (openff.evaluator.layers.CalculationLayerSchema
(openff.evaluator.protocols.openmm. Color of the color	OpenMMEnergyMinimslatismethod), 192
attribute), 419	parse_json() (openff.evaluator.layers.reweighting.ReweightingSchema class method), 202
parameterized_system	cuiss memou), 202
(openff.evaluator.protocols.openmm.(	OpenMMEvaludieEnison() (openff.evaluator.layers.simulation.SimulationSchema class method), 198
attribute), 430	parse_json() (openff.evaluator.layers.workflow.WorkflowCalculationSch
parameterized_system	1 106
(openff.evaluator.protocols.openmm.(	parse_json() (openff.evaluator.properties.Density
attribute), 425	class method), 103
parameterized_system  (openff avaluator protocols reveightis	ng.BaseEvalunaEnergign() (openff.evaluator.properties.DielectricConstant
attribute), 496	class method), 110
parameterized_system	<pre>parse_json() (openff.evaluator.properties.EnthalpyOfMixing</pre>
(openff evaluator protocols simulation	n.BaseEnergyMinimisations method), 113
attribute), 518	parse_json() (openff.evaluator.properties.EnthalpyOfVaporization
narameterized system	class method), 117
(openff.evaluator.protocols.simulation	n.BaseSimulationse_json() (openff.evaluator.properties.ExcessMolarVolume
attribute) 525	class method), 106
parse_json() (openff.evaluator.attributes.Attr	ributeClass parse_json() (openff.evaluator.properties.HostGuestBindingAffinity class method), 124
narse ison() (one off evaluator client Connec	tionOptions parse_json() (openff.evaluator.properties.SolvationFreeEnergy
class method), 81	class method), 120
parse_json() (openff.evaluator.client.Requestion), 83	est class parse_json() (openff.evaluator.protocols.analysis.AverageDielectricCon. class method), 301
parse_json() (openff.evaluator.client.Reque	
class method), 85	class method), 307
parse_json() (openff.evaluator.client.Req.	uestResult parse_json()(openff.evaluator.protocols.analysis.AverageObservable
class method) 87	class method), 296
<pre>parse_json() (openff.evaluator.datasets.Calc</pre>	$ulationSourc$ $ exttt{parse\_json()}$ $(openff.evaluator.protocols.analysis.BaseAverageObservabulationSource$
class method), 98	class method), 290

parse\_json() (openff.evaluator.datasets.MeasurementSource\_json() (openff.evaluator.protocols.analysis.BaseDecorrelateProtocols.BaseDecorrelateProtocols.BaseDe

class method), 408

- class method), 317 class method), 447
- parse\_json() (openff.evaluator.protocols.analysis.Compu**ratipe**le**json**() (openff.evaluator.protocols.paprika.coordinates.PreparePu class method), 312 class method), 436
- parse\_json() (openff.evaluator.protocols.analysis.Decorredates@bsjsonkle.copenff.evaluator.protocols.paprika.coordinates.PrepareRecclass method), 327 class method), 441
- parse\_json() (openff.evaluator.protocols.analysis.Decorr**clateStrajjeson()** (openff.evaluator.protocols.paprika.restraints.ApplyRestrain class method), 322 class method), 468
- parse\_json() (openff.evaluator.protocols.coordinates.BuilpharsedijssoesPachpeolff.evaluator.protocols.paprika.restraints.GenerateAtta class method), 334 class method), 452
- parse\_json() (openff.evaluator.protocols.coordinates.BuilpharskedSon(dinapenff.evaluator.protocols.paprika.restraints.GeneratePull class method), 345

  class method), 457
- parse\_json() (openff.evaluator.protocols.coordinates.Solv**pæds**astjssost()) (topenff.evaluator.protocols.paprika.restraints.GenerateRelectors method), 340 class method), 462
- parse\_json() (openff.evaluator.protocols.forcefield.BaseBpildSextgron() (openff.evaluator.protocols.reweighting.BaseEvaluateEnerg class method), 351 class method), 499
- parse\_json() (openff.evaluator.protocols.forcefield.BuildIpgPssGjysSyx(en(openff.evaluator.protocols.reweighting.BaseMBARProtococlass method), 361 class method), 504
- parse\_json() (openff.evaluator.protocols.forcefield.BuildSpairseffSystem() (openff.evaluator.protocols.reweighting.ConcatenateObservaluator.protocols.protocols.protocols.protocols.reweighting.ConcatenateObservaluator.protocols.proto
- parse\_json() (openff.evaluator.protocols.forcefield.BuildTflæmp&ysjson() (openff.evaluator.protocols.reweighting.ConcatenateTraject class method), 367 class method), 488
- parse\_json() (openff.evaluator.protocols.gradients.ZeroGpadisetsjson() (openff.evaluator.protocols.reweighting.ReweightDielectric class method), 372 class method), 515
- parse\_json() (openff.evaluator.protocols.groups.Conditio**pal@wij**son() (openff.evaluator.protocols.reweighting.ReweightObservable class method), 379 class method), 509

parse\_json() (openff.evaluator.protocols.miscellaneous.Divaids&aljason() (openff.evaluator.protocols.storage.UnpackStoredSimulation

- parse\_json() (openff.evaluator.protocols.groups.Conditio**pal@wujs6nfditape**nff.evaluator.protocols.simulation.BaseEnergyMinimise class method), 375

  class method), 521

  parse\_json() (openff.evaluator.protocols.miscellaneous AddVsheison() (openff.evaluator.protocols.simulation.BaseSimulation.
- parse\_json() (openff.evaluator.protocols.miscellaneous.ApdYshejson() (openff.evaluator.protocols.simulation.BaseSimulation class method), 384 class method), 528
- class method), 398 class method), 534
  parse\_json() (openff.evaluator.protocols.miscellaneous.DpansePijston()) (openff.evaluator.protocols.yank.BaseYankProtocol
- parse\_json() (openff.evaluator.protocols.miscellaneous.DpanseLysion&l) (openff.evaluator.protocols.yank.BaseYankProtocols class method), 413

  class method), 540
- class method), 413 class method), 540
  parse\_json() (openff.evaluator.protocols.miscellaneous.FilearSalsisson(ByRplenff.evaluator.protocols.yank.LigandReceptorYankProto
- parse\_json() (openff.evaluator.protocols.miscellaneous.Mpdtiply\_Yjslon() (openff.evaluator.protocols.yank.SolvationYankProtocol class method), 393 class method), 556

class method), 548

- parse\_json() (openff.evaluator.protocols.miscellaneous.SpharsetVydsoms() (openff.evaluator.server.Batch class class method), 389 method), 92
- parse\_json() (openff.evaluator.protocols.miscellaneous. Waright By Moda F) (ction ff.evaluator.storage.data.Base Simulation Data class method), 403 class method), 227
- parse\_json() (openff.evaluator.protocols.openmm.OpenMp4Esergys6in()n(sqtenff.evaluator.storage.data.BaseStoredData class method), 419 class method), 220
- parse\_json() (openff.evaluator.protocols.openmm.OpenMpAEsælujusæfin@rgopenff.evaluator.storage.data.ForceFieldData class method), 431 class method), 223
- parse\_json() (openff.evaluator.protocols.openmm.OpenMp4Sisseuliysion() (openff.evaluator.storage.data.HashableStoredData class method), 425 class method), 221
- parse\_json() (openff.evaluator.protocols.paprika.analysispamskyzejsBRP)h(uspenff.evaluator.storage.data.ReplaceableData class method), 473 class method), 225
- parse\_json() (openff.evaluator.protocols.paprika.analysis**p@vspujeRafk**)e(upeMiffkvaluator.storage.data.StoredFreeEnergyData class method), 483 class method), 233
- parse\_json() (openff.evaluator.protocols.paprika.analysispanspujeSom()) (typenff.evaluator.storage.data.StoredSimulationData class method), 478 class method), 230
- parse\_json() (openff.evaluator.protocols.paprika.coordin**paes.sed\_dJ3om(n)**y(**Appemf**f.evaluator.storage.query.BaseDataQuery

nanca	class method), 236 json() (openff.evaluator.storage.query.BaseSimulo		openff.evaluator.propertie	s.EnthalpyOfVapa	orization	
parse_				ina EuganaMalanV	la luma	
	class method), 242		(openff.evaluator.propert	ies.Excessmoiarv	оште	
parse_	json() (openff.evaluator.storage.query.ForceField		attribute), 106	II .C .D: I	100	
	class method), 239		openff.evaluator.propertie	s.HostGuestBindi	ngAffinity	
parse_	ison() (openff.evaluator.storage.query.FreeEnergy		*			
	class method), 246		(openff.evaluator.properti	es.SolvationFreeE	inergy	
parse_	son() (openff.evaluator.storage.query.Simulation.					
	class method), 244		openff.evaluator.protocols	.paprika.analysis.	AnalyzeAPRPhase	
parse_	<pre>json() (openff.evaluator.storage.query.SubstanceQ</pre>	•	attribute), 470			
	class method), 237	phase (	openff.evaluator.protocols	.paprika.restraint	s.ApplyRestraints	
parse_			attribute), 465			
	class method), 132	physic	cal_property(openff.eva	luator.layers.Calc	rulationLayerResult	
parse_	ison() (openff.evaluator.substances.Component		attribute), 190			
	class method), 130		calProperty (class in op-	enff.evaluator.data	asets),	
parse_	<pre>json() (openff.evaluator.substances.ExactAmount</pre>		93			
	class method), 134		alPropertyDataSet	(class	in	
parse_	${\sf json()}\ (open ff. evaluator. substances. Mole Fraction$		openff.evaluator.dataset			
	class method), 136	placeh	older_id(openff.evaluat	or.workflow.schen	ıas.ProtocolReplicatoı	۴
parse_	ison() (openff.evaluator.substances.Substance		property), 277			
	class method), 128	Placeh	olderValue	(class	in	
parse_	${\sf json()}$ (openff. evaluator. the rmody namics. The rmody	dynamic	Sta <b>op</b> enff.evaluator.attribu	tes), 566		
	class method), 138	pop()(	openff.evaluator.utils.obse	rvables.Observab	oleFrame	
parse_	${\sf json()}$ (openff.evaluator.utils.exceptions.Evaluator	rExcepti	on method), 570			
	class method), 88	pop_ne	ext_in_path()(openff.ev	aluator.workflow.i	utils.ProtocolPath	
parse_	<pre>json() (openff.evaluator.workflow.Protocol class</pre>		method), 285			
	method), 265	popite	em() (openff.evaluator.utils	s.observables.Obs	ervableFrame	
parse_	<pre>json() (openff.evaluator.workflow.ProtocolGroup</pre>		method), 570			
	class method), 270	potent	ial_energies( <i>openff.ev</i>	aluator.protocols.	analysis.AverageDiele	Cl
parse_	${\sf json()}$ (openff.evaluator.workflow.schemas.Protoc	olGroup	Schuernidbute), 301			
	class method), 275	potent	ial_energies( <i>openff.ev</i>	aluator.protocols.	analysis.AverageObse	$r_{\nu}$
parse_	json() (openff.evaluator.workflow.schemas.Protoc	olReplic	atattribute), 296	_		
	class method), 278		ial_energies( <i>openff.ev</i>	aluator.protocols.	analysis.BaseAverage	Oi
parse_	son() (openff.evaluator.workflow.schemas.Protoc	olSchem	a attribute), 287	•		
	class method), 273		red_charge_model			
parse_	json() (openff.evaluator.workflow.schemas.Workflo			eld.LigParGenFo	rceFieldSource	
•	class method), 280		property), 186	o .		
parse_	json() (openff.evaluator.workflow.WorkflowExcept	<i>ti</i> <b>pr</b> efer				
• –	class method), 258	•	(openff.evaluator.backer	nds.ComputeResor	urces	
parse	json() (openff.evaluator.workflow.WorkflowResult		property), 205	1		
<u>.</u>	class method), 260		red_gpu_toolkit			
partit	ion() (openff.evaluator.storage.attributes.FilePath		(openff.evaluator.backet	nds.OueueWorker	Resources	
P 4 2 2 2 2 .	method), 251		property), 207	ido, Que ile 1, e 1.te 1.		
ner th	read_memory_limit	Prepar	rePullCoordinates	(class	in	
PCI_CII	(openff.evaluator.backends.QueueWorkerResourc	_	openff.evaluator.protoco	*		
	property), 206		432	ois.paprina.coorai	,,,	
phase	(openff.evaluator.datasets.PhysicalProperty	Prenar		(class	in	
pridoc	attribute), 94	rrcpur	openff.evaluator.protoco	`		
phase	(openff.evaluator.properties.Density attribute),		438	на приртика соота	naics),	
piidoc	103	nrener	d_protocol_id()			
phase	(openff.evaluator.properties.DielectricConstant	br cher	(openff.evaluator.workfl	ow utils Protocoll	Path	
hirase	attribute), 110		method), 285		Court	
nhase (		nressi	re (openff.evaluator.thern	nodynamics Thern	nodynamicState	
P.1.40C ((	tribute), 113	P1 0000	attribute), 137			

```
properties (openff.evaluator.datasets.PhysicalPropertyDataSet
                                                                      attribute), 274
          property), 139
                                                            {\tt protocol\_schemas}. \textit{Uopenff.evaluator.workflow.schemas}. \textit{WorkflowSchema}
properties (openff.evaluator.datasets.taproom.TaproomDataSet
                                                                      attribute), 279
          property), 149
                                                            ProtocolGraph (class in openff.evaluator.workflow),
properties (openff.evaluator.datasets.thermoml.ThermoMLDataSet 266
          property), 144
                                                            ProtocolGroup (class in openff.evaluator.workflow),
properties_by_substance()
          (open ff. evaluator. datas ets. Physical Property Data SP {\it r}{\it o}{\it tocolGroupSchema})
                                                                                                  (class
                                                                                                                    in
          method), 139
                                                                      openff.evaluator.workflow.schemas), 274
                                                            ProtocolPath (class in openff.evaluator.workflow.utils),
properties_by_substance()
          (openff.evaluator.datasets.taproom.TaproomDataSet
                                                                      284
          method), 149
                                                            ProtocolReplicator
                                                                                                 (class
                                                                                                                    in
                                                                      openff.evaluator.workflow.schemas), 276
properties_by_substance()
          (openff.evaluator.datasets.thermoml.ThermoMLDataSetCols (openff.evaluator.protocols.groups.ConditionalGroup
          method), 144
                                                                      property), 379
properties_by_type()
                                                            protocols
                                                                           (openff.evaluator.workflow.ProtocolGraph
          (openff.evaluator.datasets.PhysicalPropertyDataSet
                                                                      property), 266
          method), 140
                                                            protocols
                                                                           (openff.evaluator.workflow.ProtocolGroup
properties_by_type()
                                                                      property), 268
          (openff.evaluator.datasets.taproom.TaproomDataSprotocols (openff.evaluator.workflow.Workflow prop-
          method), 149
                                                                      erty), 255
properties_by_type()
                                                            protocols (openff.evaluator.workflow.WorkflowGraph
          (open f\!\!f\!. evaluator. datasets. thermoml. ThermoMLD at a Set
                                                                      property), 258
          method), 145
                                                            ProtocolSchema
                                                                                               (class
property\_name (openff.evaluator.workflow.utils.ProtocolPath
                                                                      openff.evaluator.workflow.schemas), 272
          property), 284
                                                            provenance (openff.evaluator.datasets.CalculationSource
\verb|property_phase| (open ff. evaluator. storage. data. Base Simulation Data attribute), 98
                                                            pull\_distance (openff.evaluator.protocols.paprika.coordinates.PrepareP
          attribute), 226
property_phase (openff.evaluator.storage.data.StoredFreeEnergyDattribute), 433
                                                            pull_window_index (openff.evaluator.protocols.paprika.coordinates.Prep
          attribute), 234
property_phase (openff.evaluator.storage.data.StoredSimulationDatattribute), 433
          attribute), 230
property_phase(openff.evaluator.storage.query.BaseSimulationDataQuery
          attribute), 240
                                                            query()
                                                                           (openff.evaluator.storage.LocalFileStorage
property_phase(openff.evaluator.storage.query.FreeEnergyDataQuerethod), 218
          attribute), 246
                                                                            (openff.evaluator.storage.StorageBackend
                                                            query()
property_phase (openff.evaluator.storage.query.SimulationDataQuantithod), 216
          attribute), 244
                                                            QueryAttribute
                                                                                               (class
\verb|property_types|| (open \textit{ff.evaluator.datasets.Physical Property Data Set_{pen \textit{ff.evaluator.storage.attributes}}), 254 \\
          property), 139
                                                            queued_properties(openff.evaluator.client.RequestResult
property_types (openff.evaluator.datasets.taproom.TaproomDataSettribute), 86
          property), 150
                                                            queued_properties (openff.evaluator.server.Batch at-
property_types (openff.evaluator.datasets.thermoml.ThermoMLDatasets), 91
          property), 145
                                                            QueueWorkerResources
                                                                                                   (class
                                                                                                                    in
PropertyPhase (class in openff.evaluator.datasets), 96
                                                                      openff.evaluator.backends), 205
Protocol (class in openff.evaluator.workflow), 261
                                                            QueueWorkerResources.GPUToolkit
                                                                                                          (class
                                                                                                                    in
protocol_ids (openff.evaluator.workflow.utils.ProtocolPath
                                                                      openff.evaluator.backends), 206
          property), 284
protocol_path(openff.evaluator.workflow.utils.ProtocolPath
          property), 285
                                                            receptor_coordinate_file
protocol_replicators
                                                                      (openff.evaluator.protocols.coordinates.BuildDockedCoordinates
          (openff.evaluator.workflow.schemas.WorkflowSchema
                                                                      attribute), 342
attribute), 279 \\ \texttt{receptor\_residue\_name} \\ \texttt{protocol\_schemas}. (open \textit{ff.evaluator.workflow.schemas.ProtocolGroupSchema}, (open \textit{ff.evaluator.workflow.schemas.ProtocolGroupSchema}, (open \textit{ff.evaluator.protocols.coordinates.BuildDockedCoordinates.buildDockedCoordinates}) \\
```

attribute), 343	method), 322
receptor_residue_name	${\tt replace\_protocol()} \ (open \textit{ff.evaluator.protocols.coordinates.BuildCoordinates.} \\$
(openff.evaluator.protocols.yank.LigandRecept	torYankProtoankthod), 334
attribute), 543	${\tt replace\_protocol()} \ (open \textit{ff.evaluator.protocols.coordinates.BuildDocket}) \\$
${\tt reference} (openff.evaluator.datasets.MeasurementSouthern Southern $	rce method), 346
attribute), 99	${\tt replace\_protocol()}\ (open \textit{ff.evaluator.protocols.coordinates.Solvate Existing Protocols.})$
reference_reduced_potentials	method), 340
	$\emph{MBARephase}_{D}$ protocol ( ) ( $openff.evaluator.protocols.forcefield.BaseBuildSystems$
attribute), 501	method), 351
reference_reduced_potentials	${\tt replace\_protocol()} \ (open \textit{ff.evaluator.protocols.force} \textit{field.BuildLigParG}$
(openff.evaluator.protocols.reweighting.Reweighting)	
attribute), 516	${\tt replace\_protocol()} \ (open \textit{ff.evaluator.protocols.force} \textit{field.BuildSmirnoff}$
reference_reduced_potentials	method), 356
	$ght$ $\Omega$ here $ au$ and $ au$ eprotocol ( ) (openff. evaluator. protocols. for cefield. Build TL eap System 1.0. For each of the protocol $ au$ and $ au$ is the protocol $ au$ and $ au$ and $ au$ is the protocol $ au$ and $ au$ are the protocol $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ are the protocol $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ are the protocol $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ and $ au$ are the protocol $ au$ and $ au$ are the protocological $ au$ are the protoco
attribute), 510	method), 367
	${\it le}$ replace_protocol() (openff. evaluator. protocols. gradients. Zero Gradients.
openff.evaluator.layers), 193	method), 372
register_calculation_schema() (in modulo openff.evaluator.layers), 193	<pre>le replace_protocol() (openff.evaluator.protocols.groups.ConditionalGroumethod), 379</pre>
register_default_plugins() (in modul	
openff.evaluator.plugins), 572	method), 384
register_external_plugins() (in modul	le replace_protocol() (openff.evaluator.protocols.miscellaneous.DivideVa
openff.evaluator.plugins), 572	method), 398
1 0	le replace_protocol()(openff.evaluator.protocols.miscellaneous.DummyP
openff.evaluator.datasets.thermoml), 146	method), 413
register_workflow_protocol() (in modul	le replace_protocol() (openff.evaluator.protocols.miscellaneous.FilterSub
openff.evaluator.workflow), 272	method), 409
$\verb"relative_tolerance" (open ff. evaluator. layers. Calcula in the control of th$	tion <b>Lepda&amp;ch_epro</b> tocol() (openff.evaluator.protocols.miscellaneous.Multiply\
attribute), 191	method), 393
$\verb"relative_tolerance" (open ff. evaluator. layers. reweight a property of the property of th$	ting <b>rAplæightipgSvloaod</b> () (openff.evaluator.protocols.miscellaneous.SubtractV
attribute), 202	method), 389
$\verb"relative_tolerance" (open ff. evaluator. layers. simulation of the control of$	ion.Si <b>epllatiensdertoc</b> ol() (openff.evaluator.protocols.miscellaneous.WeightBy
attribute), 199	method), 403
$\verb"relative_tolerance" (open \textit{ff.evaluator.layers.work flow} and \textit{tolerance}) and \textit{tolerance}) are the lattice of the latt$	w.W <b>wrkylbad&amp;alpnb:ttonoSchyrhu</b> penff.evaluator.protocols.openmm.OpenMMEnerg
attribute), 196	method), 419
${\tt release\_lambdas} \ (open \textit{ff.evaluator.protocols.paprika}.$	rest <b>neplsaGenpnotBelouGReprenifin</b> svaluator.protocols.openmm.OpenMMEvalu

replace\_protocol() (openff.evaluator.protocols.analysis: Capitante Dipote Monte (popenff.evaluator.protocols.paprika.coordinates.Premethod), 312 method), 441
replace\_protocol() (openff.evaluator.protocols.analysis: Deplace\_aprobaced\_aprobaced\_aprotocolf.evaluator.protocols.paprika.restraints.Apply

replace() (openff.evaluator.storage.attributes.FilePath replace\_protocol() (openff.evaluator.protocols.openmm.OpenMMSimu

replace\_protocol() (openff.evaluator.protocols.analysis.Analyzis.A

replace\_protocol() (openff.evaluator.protocols.analysis. \*Applager\*:rprEntergols() (openff.evaluator.protocols.paprika.analysis.Compi

replace\_protocol() (openff.evaluator.protocols.analysis. \*Applagen() pseurtokkal() (openff.evaluator.protocols.paprika.analysis. Compu

replace\_protocol() (openff.evaluator.protocols.analysis: BayskAverage Observable (openff.evaluator.protocols.paprika.coordinates.Ad

replace\_protocol() (openff.evaluator.protocols.analysis: Paysk Decopredate Pada () (obpenff.evaluator.protocols.paprika.coordinates.Pre

*method*), 431

method), 425

method), 473

method), 483

method), 478

method), 447

method), 436

method), 327 method), 468 replace\_protocol() (openff.evaluator.protocols.analysis: Pepthacelaperfortipecodry) (openff.evaluator.protocols.paprika.restraints.Gene

Index 627

attribute), 460

method), 251

*method*), 302

method), 307

method), 296

method), 290

method), 317

required\_effective\_samples

attribute), 501

```
method), 452
                                                                                                                                                                                                                                                          required_effective_samples
replace_protocol() (openff.evaluator.protocols.paprika.restraints.@pneuffiteRululkRostpninttcols.reweighting.ReweightDielectricConsta
                                         method), 457
                                                                                                                                                                                                                                                                                                    attribute), 516
replace_protocol() (openff.evaluator.protocols.paprika.resquiintedGent&eattReleaseMephesints
                                         method), 462
                                                                                                                                                                                                                                                                                                     (openff.evaluator.protocols.reweighting.ReweightObservable
replace_protocol() (openff.evaluator.protocols.reweighting.BaseExtatibuteE)n&rigies
                                         method), 499
                                                                                                                                                                                                                                                          required_inputs (openff.evaluator.protocols.analysis.AverageDielectricC
replace_protocol() (openff.evaluator.protocols.reweighting.BaseMpBapPpp)tokol
                                         method), 504
                                                                                                                                                                                                                                                          required_inputs (openff.evaluator.protocols.analysis.AverageFreeEnergy
replace_protocol() (openff.evaluator.protocols.reweighting.ConcateapterOpsetOcols)
                                         method), 493
                                                                                                                                                                                                                                                          \verb"required_inputs" (open ff. evaluator. protocols. analysis. A verage Observable and the protocol of the pro
replace_protocol() (openff.evaluator.protocols.reweighting.ConcatenperTynje&fories
                                         method), 489
                                                                                                                                                                                                                                                          required_inputs(openff.evaluator.protocols.analysis.BaseAverageObser
replace_protocol() (openff.evaluator.protocols.reweighting.Reweighting.Reweighting.Constant
                                         method), 516
                                                                                                                                                                                                                                                          replace_protocol() (openff.evaluator.protocols.reweighting.Reweightophstyn)yable
                                                                                                                                                                                                                                                          required_inputs (openff.evaluator.protocols.analysis.ComputeDipoleMo
                                         method), 510
replace_protocol() (openff.evaluator.protocols.simulation.BaseEnergyMin) misation
                                                                                                                                                                                                                                                          required_inputs (openff.evaluator.protocols.analysis.DecorrelateObserv
                                         method), 521
replace_protocol() (openff.evaluator.protocols.simulation.BaseSinprolpticty), 328
                                         method), 528
                                                                                                                                                                                                                                                          required\_inputs (openff.evaluator.protocols.analysis.DecorrelateTraject
replace_protocol() (openff.evaluator.protocols.storage.UnpackStoppedSentulationData
                                                                                                                                                                                                                                                          required\_inputs (openff.evaluator.protocols.coordinates.BuildCoordinates)
                                         method), 534
replace_protocol() (openff.evaluator.protocols.yank.BaseYankPropooplerty), 334
                                                                                                                                                                                                                                                          required\_inputs (openff.evaluator.protocols.coordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDockedCoordinates.BuildDoc
                                         method), 540
method), 548
                                                                                                                                                                                                                                                           required_inputs (openff.evaluator.protocols.coordinates.SolvateExisting
replace_protocol() (openff.evaluator.protocols.yank.SolvationYankPopenty), 340
                                                                                                                                                                                                                                                          required_inputs (openff.evaluator.protocols.forcefield.BaseBuildSystem
                                         method), 556
replace_protocol() (openff.evaluator.workflow.Protocol
                                                                                                                                                                                                                                                                                                    property), 351
                                         method), 263
                                                                                                                                                                                                                                                           required_inputs (openff.evaluator.protocols.forcefield.BuildLigParGenS
replace_protocol() (openff.evaluator.workflow.ProtocolGroup
                                                                                                                                                                                                                                                                                                   property), 362
                                         method), 269
                                                                                                                                                                                                                                                          required_inputs (openff.evaluator.protocols.forcefield.BuildSmirnoffSyst
replace_protocol() (openff.evaluator.workflow.utils.ProtocolPath property), 356
                                         method), 285
                                                                                                                                                                                                                                                           required_inputs (openff.evaluator.protocols.forcefield.BuildTLeapSystem
{\tt replace\_protocol()} \ (open {\it ff.evaluator.work flow}. Work {\it flow}. Work {\it flow}. Work {\it flow}. Work {\it flow}. {\it fl
                                                                                                                                                                                                                                                                                                    property), 367
                                         method), 255
                                                                                                                                                                                                                                                           required_inputs (openff.evaluator.protocols.gradients.ZeroGradients
replace_protocol_types()
                                                                                                                                                                                                                                                                                                    property), 373
                                         (open \textit{ff.evaluator.workflow.schemas.WorkflowSchemauired\_inputs} \ (open \textit{ff.evaluator.protocols.groups.ConditionalGroup}) \ (open \textit{ff.evaluator.workflow.schemas.WorkflowSchemauired\_inputs}) \ (open \textit{ff.evaluator.protocols.groups.ConditionalGroups}) \ (open \textit{ff.evaluator.protocols.groups.ConditionalGroups)}) \ (open \textit{ff.evaluator.protocols.groups.ConditionalGroups)}) \ (open \textit{ff.evaluator.protocols.groups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.ConditionalGroups.Co
                                         method), 279
                                                                                                                                                                                                                                                                                                    property), 379
ReplaceableData
                                                                                                                                                                                                                                                         required_inputs(openff.evaluator.protocols.miscellaneous.AddValues
                                                                                                                                                 (class
                                          openff.evaluator.storage.data), 224
                                                                                                                                                                                                                                                                                                    property), 384
ReplicatorValue
                                                                                                                                                 (class
                                                                                                                                                                                                                                                          required_inputs(openff.evaluator.protocols.miscellaneous.DivideValue
                                         openff.evaluator.workflow.utils), 283
                                                                                                                                                                                                                                                                                                    property), 399
Request (class in openff.evaluator.client), 82
                                                                                                                                                                                                                                                          {\tt required\_inputs} \ (open \textit{ff.evaluator.protocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.DummyProtocols.miscellaneous.Dummy
request_estimate() (openff.evaluator.client.EvaluatorClient
                                                                                                                                                                                                                                                                                                    property), 414
                                                                                                                                                                                                                                                          \verb"required_inputs" (open ff. evaluator. protocols. miscellaneous. Filter Substantial Control of the control o
                                         method), 79
request_url (openff.evaluator.forcefield.LigParGenForceFieldSourcproperty), 409
                                         property), 186
                                                                                                                                                                                                                                                           required_inputs (openff.evaluator.protocols.miscellaneous.MultiplyValu
RequestOptions (class in openff.evaluator.client), 84
                                                                                                                                                                                                                                                                                                    property), 394
RequestResult (class in openff.evaluator.client), 86
                                                                                                                                                                                                                                                           {\tt required\_inputs} \ (open \textit{ff.evaluator.protocols.miscellaneous.SubtractValuations}) \ and \ an extraction of the protocol of the protoco
```

628 Index

property), 389

property), 404

(openff.evaluator.protocols.reweighting.BaseMBARPquixced\_inputs (openff.evaluator.protocols.miscellaneous.WeightByMo

(openff.evaluator.protocols.coordinates.SolvateExistingStructure

```
required_inputs (openff.evaluator.protocols.openmm.Openet(MIEadrs) oblinia istype()
                                                                                                                                                                                                                         (openff.evaluator.layers.CalculationLayer
                              property), 419
required_inputs (openff.evaluator.protocols.openmm.OpenMMEvaluats Enterligid), 189
                                                                                                                                                                                          required_schema_type()
                              property), 431
required_inputs (openff.evaluator.protocols.openmm.OpenMMSim(dptioff.evaluator.layers.reweighting.ReweightingLayer
                              property), 425
                                                                                                                                                                                                                         class method), 200
required_inputs (openff.evaluator.protocols.paprika.analyein/Ared/zetPMMdatype()
                              property), 474
                                                                                                                                                                                                                          (openff.evaluator.layers.simulation.SimulationLayer
required_inputs (openff.evaluator.protocols.paprika.analysis.CompellatsRepletelnock)\( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) \( \) 
                              property), 484
                                                                                                                                                                                          required_schema_type()
required_inputs (openff.evaluator.protocols.paprika.analysis.CompapeSyfiumalryConhayeixuworkflow.WorkflowCalculationLayer
                                                                                                                                                                                                                          class method), 194
                              property), 479
required_inputs (openff.evaluator.protocols.paprika.coom/istateai/Add//IsalmemyAst@ppenff.evaluator.protocols.paprika.restraints.Genera
                              property), 447
                                                                                                                                                                                                                         attribute), 452
required_inputs (openff.evaluator.protocols.paprika.coom/istaneaiPrepschPenldS(opplinffinexaluator.protocols.paprika.restraints.Genera
                              property), 437
                                                                                                                                                                                                                          attribute), 458
required_inputs (openff.evaluator.protocols.paprika.coombisturaiPrepschPedassaGonfflevalexator.protocols.paprika.restraints.Genera
                                                                                                                                                                                                                         attribute), 463
                              property), 442
required_inputs (openff.evaluator.protocols.paprika.restmastsx\dipplyReytpaintpenff.evaluator.protocols.yank.LigandReceptorYankPro
                              property), 468
                                                                                                                                                                                                                         attribute), 544
required_inputs (openff.evaluator.protocols.paprika.rest naistsx@intersupArtholoRestff.cinusduator.protocols.paprika.analysis.AnalyzeAF
                                                                                                                                                                                                                         attribute), 470
                              property), 452
\textbf{required\_inputs} (\textit{openff.evaluator.protocols.paprika.analysis.} Compute \textit{R} \textbf{interstite} \textbf{At the parafite valuator.protocols.paprika.analysis.} Compute \textit{R} \textbf{interstite} \textbf{in
                                                                                                                                                                                                                         attribute), 480
                              property), 458
required_inputs (openff.evaluator.protocols.paprika.restmints.ApplyRes
                              property), 463
                                                                                                                                                                                                                         attribute), 465
required_inputs (openff.evaluator.protocols.reweighting. Berst Evaluator.protocols.paprika.restraints. Generate.
                              property), 499
                                                                                                                                                                                                                          attribute), 452
required_inputs (openff.evaluator.protocols.reweighting.BestMBARB_quatch(openff.evaluator.protocols.paprika.restraints.Generate
                                                                                                                                                                                                                         attribute), 458
                              property), 505
required_inputs (openff.evaluator.protocols.reweighting. Cost natural Openff.evaluator.protocols.paprika.restraints. Generate
                              property), 494
                                                                                                                                                                                                                         attribute), 463
required_inputs (openff.evaluator.protocols.reweighting. Cossulte (aprellification inputs) (openff.evaluator.protocols.analysis. Average Free Energies) (openff.evalu
                                                                                                                                                                                                                         attribute), 304
                              property), 489
required_inputs (openff.evaluator.protocols.reweighting. Resulght Diehrf.evia Diomonomontocols.miscellaneous. Add Values
                              property), 516
                                                                                                                                                                                                                          attribute), 381
required_inputs (openff.evaluator.protocols.reweighting.Resulst(the first fixed) leator.protocols.miscellaneous.DivideValue
                                                                                                                                                                                                                         attribute), 395
                              property), 510
required_inputs (openff.evaluator.protocols.simulation.BreasElber(gyMinthibenisdutiator.protocols.miscellaneous.MultiplyValue
                              property), 522
                                                                                                                                                                                                                         attribute), 391
\textbf{required\_inputs} (\textit{openff.evaluator.protocols.simulation.} \textit{Bree-Silrtu lapionff.evaluator.protocols.miscellaneous.} \textit{SubtractValues} (\textit{openff.evaluator.protocols.miscellaneous.} \textit{openff.evaluator.protocols.miscellaneous.} \textit{openff.evaluator.protoc
                              property), 529
                                                                                                                                                                                                                          attribute), 386
required_inputs (openff.evaluator.protocols.storage.Unpacelsstate(ts)innffaevahlouwa.protocols.paprika.analysis.AnalyzeAPRPhase
                              property), 535
                                                                                                                                                                                                                          attribute), 470
required_inputs (openff.evaluator.protocols.yank.BaseYankBudoto(oppenff.evaluator.protocols.paprika.analysis.ComputeReferenceWood
                                                                                                                                                                                                                         attribute), 480
                              property), 541
required_inputs (openff.evaluator.protocols.yank.LigandResignt) (Speckffrevialuator.protocols.paprika.analysis.ComputeSymmetryCon
                              property), 548
                                                                                                                                                                                                                         attribute), 475
required_inputs (openff.evaluator.protocols.yank.Solvative_KarlkFs()o(aplenff.evaluator.client.Request method), 82
                              property), 557
                                                                                                                                                                                          retain_packmol_files
                                                                                                                                                                                                                         (open {\it ff. evaluator. protocols. coordinates. Build Coordinates Packmo}
required_inputs (openff.evaluator.workflow.Protocol
                              property), 262
                                                                                                                                                                                                                         attribute), 330
```

Index 629

required\_inputs(openff.evaluator.workflow.ProtocolGroupetain\_packmol\_files

property), 268

attribute), 340	$(open f\!f\!. evaluator. layers. simulation. Simulation Layer$
retrieve_force_field()	class method), 197
(openff. evaluator. storage. Local File Storage	<pre>schedule_calculation()</pre>
method), 218	$(open {\it ff.evaluator.layers.work flow.} Work {\it flowCalculationLayer}$
retrieve_force_field()	class method), 194
(openff.evaluator.storage.StorageBackend method), 216	schema (openff.evaluator.protocols.analysis.AverageDielectricConstant property), 302
	tsxiyema (openff.evaluator.protocols.analysis.AverageFreeEnergies
method), 218	property), 308
	kathema (openff.evaluator.protocols.analysis.AverageObservable
method), 216	property), 296
retrieve_results() (openff.evaluator.client.EvaluatorC	lisatiema (openff.evaluator.protocols.analysis.BaseAverageObservable
method), 79	property), 291
	${\tt schema} \ (open \textit{ff.evaluator.protocols.analysis.BaseDecorrelateProtocol}$
openff.evaluator.protocols.reweighting), 511	property), 318
	schema (openff.evaluator.protocols.analysis.ComputeDipoleMoments
openff.evaluator.layers.reweighting), 199	property), 313
ReweightingProtocols (class in openff.evaluator.protocols.utils), 560	schema (openff.evaluator.protocols.analysis.DecorrelateObservables property), 328
	schema (openff.evaluator.protocols.analysis.DecorrelateTrajectory
openff.evaluator.layers.reweighting), 201	property), 323
	${\tt schema}(open \textit{ff.evaluator.protocols.coordinates.BuildCoordinatesPackmol}$
openff.evaluator.protocols.reweighting), 505	property), 334
rfind() (openff.evaluator.storage.attributes.FilePath method), 252	schema (openff.evaluator.protocols.coordinates.BuildDockedCoordinates property), 346
	ratitiemal Greenff. Evaluation. protocols.coordinates. Solvate Existing Structure
attribute), 375	property), 340
rindex() (openff.evaluator.storage.attributes.FilePath method), 252	schema (openff.evaluator.protocols.forcefield.BaseBuildSystem property), 351
rjust() (openff.evaluator.storage.attributes.FilePath	schema (openff.evaluator.protocols.forcefield.BuildLigParGenSystem
method), 252	property), 362
role (openff.evaluator.substances.Component attribute),	schema (openff.evaluator.protocols.forcefield.BuildSmirnoffSystem
129	property), 356
	grehema (openff.evaluator.protocols.forcefield.BuildTLeapSystem
property), 217	property), 367  blachema (openff.evaluator.protocols.gradients.ZeroGradients
property), 266	property), 373
	plachema (openff.evaluator.protocols.groups.ConditionalGroup
property), 258	property), 379
	hschema (openff.evaluator.protocols.miscellaneous.AddValues
method), 252	property), 384
	schema (openff.evaluator.protocols.miscellaneous.DivideValue
method), 252	property), 399
rstrip() (openff.evaluator.storage.attributes.FilePath	schema (openff.evaluator.protocols.miscellaneous.DummyProtocol
method), 252	property), 414
6	${\tt schema}(open \textit{ff.evaluator.protocols.miscellaneous.Filter Substance By Role}$
S	property), 409
<pre>schedule_calculation()</pre>	schema (openff.evaluator.protocols.miscellaneous.MultiplyValue
$(open {\it ff.evaluator.} layers. Calculation Layer$	property), 394
class method), 189	schema (openff.evaluator.protocols.miscellaneous.SubtractValues
schedule_calculation()	property), 389
	ISE pema (openff.evaluator.protocols.miscellaneous.WeightByMoleFraction property), 404
class method), 200	schema (openff.evaluator.protocols.openmm.OpenMMEnergyMinimisation
	, - p. D. VIII C. COLONIO

property), 419			openff.evaluator.data	sets.curation.compo	onents.selection),
schema (openff.evaluator.pr	otocols.openmm.OpenMMEv	aluateEnerg	gi <b>&amp;</b> 37		
property), 431			OataPointsSchema	(class	in
schema ( <i>openff.evaluator.pr</i> property), 425	otocols.openmm.OpenMMSii	nulation	openff.evaluator.data 176	sets.curation.compo	onents.selection),
	otocols.paprika.analysis.Ana	lyz <b>&amp;&amp;P&amp;P</b>	Burbstances	(class	in
property), 474	• •	•	openff.evaluator.data	sets.curation.compo	onents.selection),
	otocols.paprika.analysis.Con	iputeRefere		•	
property), 484	• •	Selects	SubstancesSchema	(class	in
	otocols.paprika.analysis.Con	ıputeSymme		sets.curation.compo	onents.selection),
property), 479	. 1	4 1 ID	175	1: . C	
property), 447	otocols.paprika.coordinates	·	attribute), 81		•
schema (openff.evaluator.pr property), 437	otocols.paprika.coordinates	Pre <b>parede</b> nd <u>l</u>	<b>L&amp;ddn&amp;ss(as</b> penff.evalt property), 78	uator.client.Evaluato	orClient
	otocols.paprika.coordinates	Pre <b>parede</b> ale		r.client.Connection(	Options
property), 442		1	attribute), 81		•
	otocols.paprika.restraints.Ap	ply <b>Rerwei</b> n		ator.client.Evaluator	rClient
property), 468			property), 78		
schema (openff.evaluator.pr	otocols.paprika.restraints.Ge	ner <b>sæ</b> æ <b>Auu</b> a	d <b>dRe</b> s <b>(rapients</b> f.evaluator.	protocols.analysis.A	werageDielectricConstan
property), 453	• •		method), 302		
schema (openff.evaluator.pr	otocols.paprika.restraints.Ge	ner <b>sæ</b> æ <b>Pulli</b>	R <b>est)</b> a <b>iope</b> nff.evaluator.	protocols.analysis.A	werageFreeEnergies
property), 458	• •		method), 308		
schema (openff.evaluator.pr	otocols.paprika.restraints.Ge	ner <b>sæ</b> æ <b>Ræbei</b>	u <b>deBe(søpæinnffs</b> evaluator.	protocols.analysis.A	verageObservable
property), 463			method), 296		
schema ( <i>openff.evaluator.pr</i> property), 499	otocols.reweighting.BaseEva	lua <b>seFnengi</b>	(openff.evaluator. method), 291	protocols.analysis.E	BaseAverageObservable
	otocols.reweighting.BaseMB	ARBeat <u>o</u> uuli	* *	protocols.analysis.E	
property), 505	3 3		method), 318	ı ,	
	otocols.reweighting.Concater	nat <b>es@bse.nvi</b>	* *	protocols.analysis.C	ComputeDipoleMoments
property), 494			method), 313		
	otocols.reweighting.Concate	nat <b>eFrajemi</b> a	de (openff.evaluator.	protocols.analysis.L	DecorrelateObservables
property), 489		v	method), 328		
schema (openff.evaluator.pr	otocols.reweighting.Reweigh	tDi <b>sk</b> et <u>tr</u> uicu6	E <b>drístuap</b> enff.evaluator.	protocols.analysis.L	DecorrelateTrajectory
property), 516			method), 323		
schema (openff.evaluator.pr	otocols.reweighting.Reweigh	tOb <b>set</b> vabli	d() (openff.evaluator.	protocols.coordinat	es.BuildCoordinatesPack
property), 510			method), 334		
schema ( <i>openff.evaluator.pr</i> property), 522	otocols.simulation.BaseEner	gyM <b>se</b> timixui	t <b>ic</b> (n) (openff.evaluator. method), 346	protocols.coordinat	es.BuildDockedCoording
	otocols.simulation.BaseSimu	<i>lati<b>se</b>t_</i> uui	* *	protocols.coordinat	es.SolvateExistingStructi
property), 529			method), 340	1	Ü
. 1 00 1	otocols.storage.UnpackStored	dSi <i>n</i> smerta <i>ti</i> nomi	1	protocols.forcefield.	BaseBuildSystem
property), 535	esta a ala mante Dan aVanteDusta		method), 351	musta sala fansafiald	DuildLia DauCon Sustan
	otocols.yank.BaseYankProtoc	oi set_uu:		protocois.jorcejieta.	BuitaLigParGenSystem
property), 541	esta a ala manik Lia an dD a antar	Varala Processor	method), 362	musta sala fansafiald	Devil dConsister of Constant
property), 548	otocols.yank.LigandReceptor		method), 356		
	otocols.yank.SolvationYankP	rot <b>se</b> otl_uui		protocols.forcefield.	BuildTLeapSystem
property), 557			method), 368		
	workflow.Protocol property)	, set_uui	d() (openff.evaluator.	protocols.gradients.	ZeroGradients
262			method), 373	_	
schema (openff.evaluator.w erty), 271	orkflow.ProtocolGroup prop	set_uui	d() (openff.evaluator. method), 379	protocols.groups.Co	onditionalGroup
SelectDataPoints	(class in	set_uui	d() (openff.evaluator.	protocols.miscellan	eous.AddValues

method), 474

- method), 384 method), 529
- set\_uuid() (openff.evaluator.protocols.miscellaneous.DivideValuatid() (openff.evaluator.protocols.storage.UnpackStoredSimulationD method), 399 method), 535
- set\_uuid() (openff.evaluator.protocols.miscellaneous.Dun**sexPuntiod()** (openff.evaluator.protocols.yank.BaseYankProtocol method), 414 method), 541
- set\_uuid() (openff.evaluator.protocols.miscellaneous.Filtes & the waird & Replenff.evaluator.protocols.yank.LigandReceptorYankProtocolomethod), 409 method), 548
- set\_uuid() (openff.evaluator.protocols.miscellaneous.Mulsphy.Vauhiel() (openff.evaluator.protocols.yank.SolvationYankProtocol method), 394 method), 557
- set\_uuid() (openff.evaluator.protocols.miscellaneous.Subtsett\(\subsett\) (openff.evaluator.workflow.Protocol method), 389 method), 263
- set\_uuid() (openff.evaluator.protocols.miscellaneous.WeiglerByWhildF)rac(inpenff.evaluator.workflow.ProtocolGroup method), 404 method), 268
- set\_uuid() (openff.evaluator.protocols.openmm.OpenMMEntrgyd/Initia)s(atpenff.evaluator.protocols.analysis.AverageDielectricConstant method), 419 method), 302
- set\_uuid() (openff.evaluator.protocols.openmm.OpenMMEetluntelfre@lexpenff.evaluator.protocols.analysis.AverageFreeEnergies method), 431 method), 308
- set\_uuid() (openff.evaluator.protocols.openmm.OpenMMSiertulmaibue() (openff.evaluator.protocols.analysis.AverageObservable method), 426 method), 296
- method), 426 method), 296
  set\_uuid() (openff.evaluator.protocols.paprika.analysis.A**sel**yz**yAPRP()**ukopenff.evaluator.protocols.analysis.BaseAverageObservable
- set\_uuid() (openff.evaluator.protocols.paprika.analysis.ComputaRufer@n@Weafkevaluator.protocols.analysis.BaseDecorrelateProtocol method), 484 method), 318

method), 291

- set\_uuid() (openff.evaluator.protocols.paprika.analysis.ComputeDywer();twy@ovff.extiabnator.protocols.analysis.ComputeDipoleMoment. method), 479 method), 313
- set\_uuid() (openff.evaluator.protocols.paprika.coordinate**s&t**dd**lalime**(y)A(openff.evaluator.protocols.analysis.DecorrelateObservables method), 447 method), 328
- set\_uuid() (openff.evaluator.protocols.paprika.coordinate**ssPrepakePe(I)Copedifiates**luator.protocols.analysis.DecorrelateTrajectory method), 437 method), 323
- set\_uuid() (openff.evaluator.protocols.paprika.coordinatessPrepakel@dsqConffrebinlutasor.protocols.coordinates.BuildCoordinatesPacemethod), 442

  method), 334
- set\_uuid() (openff.evaluator.protocols.paprika.restraints.AsptlyRestraints

set\_uuid() (openff.evaluator.protocols.paprika.restraints.GeneralAnn()/Repanffintsaluator.protocols.coordinates.SolvateExistingStruc

- method), 453 method), 340 set\_uuid() (openff.evaluator.protocols.paprika.restraints.General Mel(Restpainff.evaluator.protocols.forcefield.BaseBuildSystem
- set\_uuid() (openff.evaluator.protocols.paprika.restraints.**GeneralBal(Restpainff.**evaluator.protocols.forcefield.BaseBuildSystem method), 458 method), 351
- set\_uuid() (openff.evaluator.protocols.paprika.restraints. **General Releas (Pestificinus** uator.protocols.forcefield.BuildLigParGenSystem method), 463 method), 362
- set\_uuid() (openff.evaluator.protocols.reweighting.BaseEsceltuateEnce()e(openff.evaluator.protocols.forcefield.BuildSmirnoffSystem method), 499 method), 356
- set\_uuid() (openff.evaluator.protocols.reweighting.BaseMsARRadroe(l) (openff.evaluator.protocols.forcefield.BuildTLeapSystem method), 505 method), 368
- set\_uuid() (openff.evaluator.protocols.reweighting.Conca**setus@bse**()abbeenff.evaluator.protocols.gradients.ZeroGradients method), 494 method), 373
- set\_uuid() (openff.evaluator.protocols.reweighting.Concasants/Flipse(A)riepenff.evaluator.protocols.groups.ConditionalGroup method), 489 method), 379
- set\_uuid() (openff.evaluator.protocols.reweighting.ReweightDislartue(O(openff.evaluator.protocols.miscellaneous.AddValues method), 516 method), 384
- set\_uuid() (openff.evaluator.protocols.reweighting.ReweightObsalud() (openff.evaluator.protocols.miscellaneous.DivideValue method), 510 method), 399
- set\_uuid() (openff.evaluator.protocols.simulation.BaseEnergy.Windiwis@tiopenff.evaluator.protocols.miscellaneous.DummyProtocol method), 522 method), 414
- set\_uuid() (openff.evaluator.protocols.simulation.BaseSinselutivalue() (openff.evaluator.protocols.miscellaneous.FilterSubstanceBy

1 0 100	I D 740
<i>''</i>	method), 548
set_value() (openff.evaluator.protocols.miscellaneous.MustiplyWallu	
	method), 557
set_value() (openff.evaluator.protocols.miscellaneous.Subsetc.Nalu method), 389	mes() (openff.evaluator.workflow.Protocol method), 264
set_value() (openff.evaluator.protocols.miscellaneous.We <b>se/ttByAI</b> valuator.protocols.miscellaneous.We <b>se/ttByAIv</b> aluator.protocols.miscellaneous.We <b>se/ttByAIvaluator.protocols.miscellaneous.Wese/ttByAIvaluator.protocols.miscellaneous.Wese/ttByAIvaluator.protocols.miscellaneous.Wese/ttByAIvaluator.protocols.miscellaneous.Wese/ttByAIvaluator.protocols.miscellaneous.wine.protocols.miscellaneous.Wese/ttByAIvaluator.protocols.miscellaneous.wine.protocols.mi</b>	ฟ <b>eF)</b> ra( <b>ชาฺณ</b> nff.evaluator.workflow.ProtocolGroup method), 271
set_value() (openff.evaluator.protocols.openmm.OpenMMEndeeffalu	
method), 419	method), 570
set_value() (openff.evaluator.protocols.openmm.OpenMMEtuhpapa method), 431	Hyengiestf.evaluator.protocols.yank.BaseYankProtocol attribute), 537
set_value() (openff.evaluator.protocols.openmm.OpenMMs@ivmpladia	
	attribute), 548
set_value() (openff.evaluator.protocols.paprika.analysis. Asatlype Afmethod), 474	RRP(upsenff.evaluator.protocols.yank.SolvationYankProtocol attribute), 557
set_value() (openff.evaluator.protocols.paprika.analysis. Simplicati	
	(openff.evaluator.protocols.storage.UnpackStoredSimulationData
set_value() (openff.evaluator.protocols.paprika.analysis.ComputeS	
	ionDataQuery (class in
set_value() (openff.evaluator.protocols.paprika.coordinates.AddDu	· · · · · · · · · · · · · · · · · · ·
method), 447 Simulati	
set_value() (openff.evaluator.protocols.paprika.coordinates.Prepar	· · · · · · · · · · · · · · · · · · ·
	ionProtocols (class in
<pre>set_value() (openff.evaluator.protocols.paprika.coordinates.Prepar</pre>	re Reinflier Cloucit disporter to cols. utils), 558
method), 442 Simulati	ionSchema (class in
$\verb set_value()  (open ff. evaluator. protocols. paprika. restraints. Apply Restrain$	sopeinffsevaluator.layers.simulation), 197
	(openff.evaluator.substances.Component at-
$\verb set_value()  (open ff. evaluator. protocols. paprika. restraints. Generated and the protocol of the protoc$	
	fForceFieldSource (class in
$\verb set_value()  (open ff. evaluator. protocols. paprika. restraints. Generate for the protocol of the protoco$	
	openff.evaluator.protocols.yank.SolvationYankProtocol
set_value() (openff.evaluator.protocols.paprika.restraints.Generate	
	coordinate_file
set_value() (openff.evaluator.protocols.reweighting.BaseEvaluateE	
· · · · · · · · · · · · · · · · · · ·	attribute), 336
set_value() (openff.evaluator.protocols.reweighting.BaselsGBARCROW method), 505	(openff.evaluator.protocols.yank.SolvationYankProtocol
set_value() (openff.evaluator.protocols.reweighting.ConcatenateOl	
	n_1_free_energy
set_value() (openff.evaluator.protocols.reweighting.ConcatenateTra	
	attribute), 552
set_value() (openff.evaluator.protocols.reweighting.ReweightnDialen	atr <b>ic Gys. tiem (</b> openff.evaluator.protocols.yank.SolvationYankProto
method), 516	attribute), 552
$\verb set_value()  (open \textit{ff.evaluator.protocols.reweighting.Reweight \textit{Objects})    \textit{open ff.evaluator.protocols.reweighting.Reweight \textit{Objects})    \textit{open ff.evaluator.protocols.reweighting.Reweight \textit{Objects})    \textit{open ff.evaluator.protocols.reweighting.Reweight \textit{Objects})    \textit{open ff.evaluator.protocols.reweighting.Reweight \textit{Objects})    open ff.evaluator.protocols.reweight \textit{open ff.evaluator.protocols.reweight \textit{open ff.evaluator.protocols.reweight \textit{open ff.evaluator.protocols.reweight \textit{open ff.evaluator.protocols.reweight \textit{open ff.evaluator.protocols.reweight \textit{open ff.evaluator.protocols.protocols.reweight \textit{open ff.evaluator.protocols.proto$	nv <i>dbl</i> trajectory_path
	(openff.evaluator.protocols.yank.SolvationYankProtocol
$\verb set_value()  (open ff. evaluator. protocols. simulation. Base Energy Minimum and States and Sta$	
	n_2_coordinates
<pre>set_value() (openff.evaluator.protocols.simulation.BaseSimulation     method), 529</pre>	(openff.evaluator.protocols.yank.SolvationYankProtocol attribute), 552
set_value() (openff.evaluator.protocols.storage.UnpackStorehString)	
	(openff. evaluator. protocols. yank. Solvation Yank Protocol
	attribute), 553
	$\verb"n_2_system" (open \textit{ff.evaluator.protocols.yank.Solvation} Y ank \textit{Protocols.yank.Solvation} Y ank Protocols.yank.Yank.Yank.Yank.Yank.Yank.Yank.Yank.Y$
$\verb set_value()  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. protocols. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. yank. yank. Ligand Receptor Yank Parameter)  (open ff. evaluator. yank. $	Paoticibolte), 552

solutio	n_2_trajectory_path		(openff.evaluator.properties.ExcessMolarVolume
	(openff.evaluator.protocols.yank.SolvationYankPr		attribute), 107
	attribute), 553	source	(openff.evaluator.properties.HostGuestBindingAffinity
solvate	d_complex_coordinates		attribute), 124
	(openff.evaluator.protocols.yank.LigandReceptor	YsokiProto	
	attribute), 544		attribute), 120
solvate	d_complex_system		_calculation_id
		YankProto	odalpenff.evaluator.storage.data.BaseSimulationData
_	attribute), 544		attribute), 226
solvate	d_complex_trajectory_path		_calculation_id
		YankProto	odalpenff.evaluator.storage.data.StoredFreeEnergyData
_	attribute), 545		attribute), 234
solvate	d_ligand_coordinates		_calculation_id
		YankProto	odalpenff.evaluator.storage.data.StoredSimulationData
	attribute), 543		attribute), 230
solvate	d_ligand_system		_calculation_id
		YankProto	odalpenff.evaluator.storage.query.BaseSimulationDataQuery
<b>.</b>	attribute), 543		attribute), 240
solvate	d_ligand_trajectory_path		_calculation_id
	(openff.evaluator.protocols.yank.LiganaReceptor attribute), 544	<i>YankProte</i>	odalpenff.evaluator.storage.query.FreeEnergyDataQuery attribute), 246
Solvato		cource	_calculation_id
JOIVACE	openff.evaluator.protocols.coordinates), 335	Source_	(openff.evaluator.storage.query.SimulationDataQuery
Solvati	onFreeEnergy (class in		attribute), 244
JOIVACI	openff.evaluator.properties), 118	cources	s (openff.evaluator.datasets.PhysicalPropertyDataSet
Solvati	onYankProtocol (class in	30ul cc.	property), 139
JOIVACI	openff.evaluator.protocols.yank), 549	sources	s (openff.evaluator.datasets.taproom.TaproomDataSet
solvent	_1 (openff.evaluator.protocols.yank.SolvationYank		
502.0110	attribute), 551		s (openff.evaluator.datasets.thermoml.ThermoMLDataSet
solvent	_1_coordinate_path		property), 145
	(openff.evaluator.protocols.yank.SolvationYankPi	ra <b>sol</b> ide (	
	attribute), 552		method), 252
solvent	_1_trajectory_path	splitli	ines() (openff.evaluator.storage.attributes.FilePath
	(openff.evaluator.protocols.yank.SolvationYankPr		method), 252
	attribute), 552		(openff.evaluator.backends.CalculationBackend
solvent	_2 (openff.evaluator.protocols.yank.SolvationYank		
	attribute), 551		) (openff.evaluator.backends.dask.BaseDaskBackend
solvent	_2_coordinate_path		method), 208
		ræsteaælt ()	) (openff.evaluator.backends.dask.BaseDaskJobQueueBackend
	attribute), 553		method), 209
solvent	_2_trajectory_path	start()	) (openff.evaluator.backends.dask.DaskLocalCluster
	(openff.evaluator.protocols.yank.SolvationYankPi	rotocol	method), 210
	attribute), 553	start()	) (openff.evaluator.backends.dask.DaskLSFBackend
Source (	(class in openff.evaluator.datasets), 97		method), 212
source	(openff.evaluator.datasets.PhysicalProperty at-	start()	) (openff.evaluator.backends.dask.DaskPBSBackend
	tribute), 95		method), 214
source	(openff.evaluator.properties.Density attribute),	start()	(openff.evaluator.server.EvaluatorServer
	103		method), 90
source	$(open {\it ff. evaluator. properties.} Dielectric {\it Constant}$	start_p	protocol (openff.evaluator.workflow.utils.ProtocolPath
	attribute), 110		property), 284
source	(openff.evaluator.properties.EnthalpyOfMixing	start_s	state_trajectory
	attribute), 114		(openff. evaluator. storage. data. Stored Free Energy Data
source(	openff.evaluator.properties. Enthalpy Of Vaporization for the properties of the pr		attribute), 232
	attribute) 117	started	(openff evaluator backends CalculationBackend

property), 204			openff.evaluator.stora	ge.attributes), 253	
started(openff.evaluator.ba	ckends.dask.BaseDaskBacker	a&torage	Backend (class in oper	ıff.evaluator.storage), 2	215
property), 208		store_f	orce_field()		
started(openff.evaluator.ba	ckends.dask.BaseDaskJobQu	eueBacker	ndopenff.evaluator.store	age.LocalFileStorage	
property), 210			method), 218		
started(openff.evaluator.ba	ckends.dask.DaskLocalCluste	store_f	orce_field()		
property), 210			(openff.evaluator.store	age.StorageBackend	
started(openff.evaluator.ba	ckends.dask.DaskLSFBacken	d	method), 215		
property), 212		store_o	bject()(openff.evalu	ator.storage.LocalFileS	Storage
started(openff.evaluator.ba	ckends.dask.DaskPBSBacken	d	method), 218		
property), 214		store_o	bject()(openff.evalu	ator.storage.StorageBa	ckend
startswith()(openff.evalue	ator.storage.attributes.FilePat	h	method), 215		
method), 252		StoredF	reeEnergyData	(class	in
State (class in openff.evaluat	or.datasets.curation.compone	nts.selecti	ompenff.evaluator.stora	ge.data), 231	
177		StoredS	imulationData	(class	in
statistical_inefficiend	cy		openff.evaluator.stora	ge.data), 228	
(openff.evaluator.st	orage.data. Stored Simulation L	Octurip()	(openff.evaluator.s	torage.attributes.FileP	ath
attribute), 229			method), 252		
steps_per_iteration		submit_	task() (openff.evalua	tor.backends.Calculati	onBackend
	otocols.openmm.OpenMMSin	nulation	method), 204		
attribute), 426	•	submit_	task() (openff.evalua	tor.backends.dask.Base	eDaskBackend
steps_per_iteration			method), 208		
	otocols.simulation.BaseSimul	<i>a<b>siub</b>m</i> it_	task() (openff.evalua	tor.backends.dask.Base	eDaskJobQueueBack
attribute), 524			method), 209		
steps_per_iteration		submit_	task() (openff.evalua	tor.backends.dask.Das	kLocalCluster
	otocols.yank.BaseYankProtoc		method), 210		
attribute), 537	,		task() (openff.evalua	tor.backends.dask.Das	kLSFBackend
steps_per_iteration			method), 212		
	otocols.yank.LigandReceptor	Y <b>subi</b> Riiot <u>o</u>	talsk() (openff.evalua	tor.backends.dask.Das	kPBSBackend
attribute), 549			method), 214		
steps_per_iteration		subset(	) (openff.evaluator.util	s.observables.Observa	bleArray
	otocols.yank.SolvationYankPi		method), 568		·
attribute), 557	•		) (openff.evaluator.util	s.observables.Observa	bleFrame
steric_lambdas_1(openff.e	evaluator.protocols.yank.Solv	ationYankl	Pnoethood), 570		
attribute), 552	•		ce (class in openff.eval	luator.substances), 125	5
steric_lambdas_2(openff.e	evaluator.protocols.yank.Solv				
attribute), 552			attribute), 94		•
stop() (openff.evaluator.ba	ckends.CalculationBackend	substan	ce (openff.evaluator	r.properties.Density	at-
method), 204			tribute), 103		
stop() (openff.evaluator.bac	kends.dask.BaseDaskBackend	<i>l</i> substan	ce (openff.evaluator.pr	operties.DielectricCon	istant
method), 208			attribute), 110	•	
stop() (openff.evaluator.bac	kends.dask.BaseDaskJobQue	u <b>eBbektæn</b>	ke (openff.evaluator.pr	operties.EnthalpyOfM	ixing
method), 210	_		attribute), 114	1 11 0	
stop() (openff.evaluator.bac	kends.dask.DaskLocalCluster	substan	* * * * * * * * * * * * * * * * * * * *	operties.EnthalpyOfVa	porization
method), 210			attribute), 117	1 1 0	
stop() (openff.evaluator.bac	kends.dask.DaskLSFBackend	substan	* *	operties.ExcessMolar	Volume
method), 212			attribute), 107	1	
stop() (openff.evaluator.bac	kends.dask.DaskPBSBackend	substan		operties.HostGuestBin	dingAffinity
method), 214			attribute), 124	1	0 33 3
, , , , , , , , , , , , , , , , , , ,	ator.server.EvaluatorServer	substan	* *	operties.SolvationFree	Energy
method), 90			attribute), 120		
storage_queries(openff.ev	valuator.layers.reweighting.Re	vsæilg latian	* * * * * * * * * * * * * * * * * * * *	otocols.coordinates.Bi	uildCoordinatesPackn
attribute), 201	. 0 0	0	attribute), 330		
StorageAttribute	(class in	substan	ce (openff.evaluator.pr	otocols.coordinates.So	lvateExistingStructur

attribute), 340			target_reduced_potentials
substance (openff.evaluator attribute), 348	protocols.forcefield.Bas	eBuild.	ldSystem (openff.evaluator.protocols.reweighting.BaseMBARProtocol attribute), 501
, · · · · · · · · · · · · · · · · · · ·	nrotocols forcefield Rui	ldI ia Pi	Parange System duced_potentials
attribute), 362	.ргогосогз.јогсејгега.Виг	iuLigi	(openff.evaluator.protocols.reweighting.ReweightDielectricConst
substance (openff.evaluator	nrotocols forcefield Rui	ldSmir	
attribute), 356	.proiocois.jorcejieia.bui		target_reduced_potentials
, · · · · · · · · · · · · · · · · · · ·	nrotocols forcefield Rui		eapSystem (openff.evaluator.protocols.reweighting.ReweightObservable
attribute), 368	.protocots.jorecjicia.but	штысц	attribute), 510
substance (openff.evaluator	:protocols.paprika.coord	linates'.	
attribute), 444	.protocots.pup.t.tencoore		openff.evaluator.datasets.curation.components.selection),
substance (openff.evaluator	:protocols.paprika.coore	linates.	
attribute), 437	1 1 1		temperature(openff.evaluator.thermodynamics.ThermodynamicState
, · · · · · · · · · · · · · · · · · · ·	:protocols.paprika.coore		es.PrepareR <b>atvaiseCa)</b> oidinates
attribute), 442	1 1 1		temperature_cutoff(openff.evaluator.layers.reweighting.ReweightingSe
substance (openff.evaluator	protocols.storage.Unpa:		
attribute), 531			thermodynamic_state
substance (openff.evaluator	:storage.data.BaseSimul	ationD	Data (openff.evaluator.datasets.PhysicalProperty
attribute), 226	-		attribute), 94
substance (openff.evaluator	:storage.data.StoredFree	Energy	rg <b>ther</b> modynamic_state
attribute), 234			(openff.evaluator.properties.Density attribute),
substance (openff.evaluator	storage.data.StoredSim	ulation	onData 103
attribute), 231		-	thermodynamic_state
substance (openff.evaluator	.storage.query.BaseSimi	ılationl	nDataQuer\openff.evaluator.properties.DielectricConstant
attribute), 240			attribute), 110
substance (openff.evaluator	:storage.query.FreeEner	gyData	
attribute), 246			(openff.evaluator.properties.EnthalpyOfMixing
substance (openff.evaluator	storage.query.Simulatio		
attribute), 244			thermodynamic_state
	valuator.storage.query.B	aseSim	imulationDatap@nffrgvaluator.properties.EnthalpyOfVaporization
attribute), 240			attribute), 117
substance_query(openff.e	valuator.storage.query.F	reeEne	
attribute), 247			(openff.evaluator.properties.ExcessMolarVolume
substance_query (openff.e	valuator.storage.query.S		
attribute), 244	/ 1		thermodynamic_state
SubstanceQuery	(class	in	(openff.evaluator.properties.HostGuestBindingAffinity
openff.evaluator.sto		. D	attribute), 124
substances (openff.evaluate	r.aatasets.PhysicalProp	ertyDa	
property), 139		D	(openff.evaluator.properties.SolvationFreeEnergy
substances (openff.evaluate	r.aatasets.taproom.1apr		
property), 150	on datasets they moved The		thermodynamic_state  MI DataSet (one of a valuation protected and basis Average Dielectric Constant
	r.aaiaseis.inermomi.1 ne	ermowi.	MLDataSet (openff.evaluator.protocols.analysis.AverageDielectricConstant attribute), 302
<pre>property), 145 SubtractValues</pre>	(class	in .	thermodynamic_state
	otocols.miscellaneous),	ırı	(openff.evaluator.protocols.analysis.AverageFreeEnergies
оренуу.evaнаают.рт 385	nocois.miscenaneous),		attribute), 304
swapcase() (openff.evaluate	or storage attributes File	Dath .	
method), 253	vi.storage.attributes.File.	ıum	(openff.evaluator.protocols.analysis.AverageObservable
memoa), 255			attribute), 296
Т		-	thermodynamic_state
	(class		(openff.evaluator.protocols.analysis.BaseAverageObservable
TaproomDataSet	(class	in	attribute), 287
TaproomSource	tasets.taproom), 147 (class	in	thermodynamic_state
_	tasets.taproom), 151	ııı	(openff.evaluator.protocols.openmm.OpenMMEvaluateEnergies
ορειήμεναιμαίοι.αα	iascis.iapioonij, 131		1 1

```
attribute), 431
                                                                 openff.evaluator.datasets.thermoml), 142
thermodynamic_state
                                                        thermostat friction
         (openff.evaluator.protocols.openmm.OpenMMSimulation
                                                                 (openff.evaluator.protocols.openmm.OpenMMSimulation
         attribute), 426
                                                                 attribute), 426
thermodynamic_state
                                                        thermostat_friction
         (openff.evaluator.protocols.paprika.analysis.ComputeReferefroeMffkvaluator.protocols.simulation.BaseSimulation
         attribute), 480
                                                                 attribute), 525
thermodynamic_state
                                                       time_series_statistics
         (openff.evaluator.protocols.paprika.analysis.ComputeSymm&tpp@ff.evaliwator.protocols.analysis.AverageDielectricConstant
         attribute), 475
                                                                 attribute), 302
thermodynamic_state
                                                        time_series_statistics
         (openff.evaluator.protocols.reweighting.BaseEvaluateEnerg@penff.evaluator.protocols.analysis.AverageObservable
                                                                 attribute), 297
         attribute), 496
thermodynamic_state
                                                       time_series_statistics
         (openff.evaluator.protocols.reweighting.ReweightDielectricQopsnffievaluator.protocols.analysis.BaseAverageObservable
         attribute), 512
                                                                 attribute), 288
thermodynamic_state
                                                       time_series_statistics
         (openff.evaluator.protocols.simulation.BaseSimulation
                                                                 (openff.evaluator.protocols.analysis.BaseDecorrelateProtocol
         attribute), 525
                                                                 attribute), 314
thermodynamic_state
                                                        time_series_statistics
         (openff.evaluator.protocols.storage.UnpackStoredSimulationDpenff.evaluator.protocols.analysis.DecorrelateObservables
         attribute), 531
                                                                 attribute), 328
thermodynamic_state
                                                        time_series_statistics
         (openff.evaluator.protocols.yank.BaseYankProtocol
                                                                 (openff.evaluator.protocols.analysis.DecorrelateTrajectory
         attribute), 537
                                                                 attribute), 323
thermodynamic_state
                                                       timestep(openff.evaluator.protocols.openmm.OpenMMSimulation
         (openff.evaluator.protocols.yank.LigandReceptorYankProtocoltribute), 426
         attribute), 549
                                                        timestep(openff.evaluator.protocols.simulation.BaseSimulation
thermodynamic_state
                                                                 attribute), 524
         (openff.evaluator.protocols.yank.SolvationYankProtimelstep (openff.evaluator.protocols.yank.BaseYankProtocol
         attribute), 557
                                                                 attribute), 537
thermodynamic_state
                                                        timestep (openff.evaluator.protocols.yank.LigandReceptorYankProtocol
         (openff.evaluator.storage.data.BaseSimulationData
                                                                 attribute), 549
         attribute), 226
                                                       \verb|timestep| (open ff. evaluator. protocols. yank. Solvation Yank Protocol|
thermodynamic_state
                                                                 attribute), 557
         (openff.evaluator.storage.data.StoredFreeEnergyDtatale()
                                                                    (openff.evaluator.storage.attributes.FilePath
         attribute), 234
                                                                 method), 253
thermodynamic_state
                                                       TLeapForceFieldSource
                                                                                            (class
                                                                                                           in
         (openff.evaluator.storage.data.StoredSimulationData
                                                                 openff.evaluator.forcefield), 183
                                                        to\_force\_field() (openff.evaluator.forcefield.SmirnoffForceFieldSource)
         attribute), 231
thermodynamic_state
                                                                 method), 182
         (openff.evaluator.storage.query.BaseSimulationDatoQqcaph()
                                                                           (openff.evaluator.workflow.Workflow
         attribute), 240
                                                                 method), 256
thermodynamic_state
                                                        to_number_of_molecules()
         (openff.evaluator.storage.query.FreeEnergyDataQuery
                                                                 (openff.evaluator.substances.Amount method),
         attribute), 247
                                                                 131
                                                        to_number_of_molecules()
thermodynamic_state
         (openff.evaluator.storage.query.SimulationDataQuery
                                                                 (openff.evaluator.substances.ExactAmount
         attribute), 244
                                                                 method), 133
ThermodynamicState
                                                       to_number_of_molecules()
                                  (class
         openff.evaluator.thermodynamics), 136
                                                                 (openff.evaluator.substances.MoleFraction
thermoml_property()
                                              module
                                                                 method), 135
                                                        to_pandas() (openff.evaluator.datasets.PhysicalPropertyDataSet
         openff.evaluator.datasets.thermoml), 146
ThermoMLDataSet
                                                                 method), 140
                                (class
                                                   in
```

```
to_pandas() (openff.evaluator.datasets.taproom.TaproomDataSet attribute), 496
                                  method), 150
                                                                                                                                                                                                                  trajectory_file_path
to_pandas() (openff.evaluator.datasets.thermoml.ThermoMLDataSetopenff.evaluator.protocols.simulation.BaseSimulation
                                                                                                                                                                                                                                                     attribute), 525
                                  method), 145
to_protocol() (openff.evaluator.workflow.schemas.Protocolf@jeogt@chenfile_path
                                  method), 275
                                                                                                                                                                                                                                                     (openff.evaluator.protocols.storage.UnpackStoredSimulationData
to_protocol() (openff.evaluator.workflow.schemas.ProtocolSchemaattribute), 531
                                                                                                                                                                                                                  {\tt trajectory\_path}\ (open ff. evaluator. protocols. analysis. Compute Dipole Months and Compu
                                   method), 273
to_storage_query() (openff.evaluator.storage.data.BaseSimulationDaribute), 309
                                                                                                                                                                                                                  \verb|trajectory_paths|| (open \textit{ff.evaluator.protocols.paprika.analysis.Analyze Analyze Analyze
                                   method), 227
to_storage_query() (openff.evaluator.storage.data.BaseStoredDatattribute), 470
                                                                                                                                                                                                                  translate() (openff.evaluator.storage.attributes.FilePath
                                   method), 220
to_storage_query() (openff.evaluator.storage.data.ForceFieldDatamethod), 253
                                                                                                                                                                                                                  type (openff.evaluator.protocols.groups.ConditionalGroup.Condition
                                  method), 223
to_storage_query() (openff.evaluator.storage.data.HashableStored#Datbute), 375
                                   method), 222
                                                                                                                                                                                                                   type (openff.evaluator.workflow.schemas.ProtocolGroupSchema
to_storage_query() (openff.evaluator.storage.data.ReplaceableDatatribute), 275
                                  method), 225
                                                                                                                                                                                                                  type (openff.evaluator.workflow.schemas.ProtocolSchema
to_storage_query() (openff.evaluator.storage.data.StoredFreeEnergy Polatone), 273
                                   method), 233
to_storage_query() (openff.evaluator.storage.data.Stored imulationData
                                  method), 230
                                                                                                                                                                                                                  uncertainty (openff.evaluator.datasets.PhysicalProperty
tolerance (openff.evaluator.protocols.coordinates.BuildCoordinatesPagkmal), 94
                                   attribute), 330
                                                                                                                                                                                                                  uncertainty
                                                                                                                                                                                                                                                                                              (openff.evaluator.properties.Density
tolerance (openff.evaluator.protocols.coordinates.SolvateExistingStructurate), 103
                                   attribute), 340
                                                                                                                                                                                                                  uncertainty (openff.evaluator.properties.DielectricConstant
tolerance (openff.evaluator.protocols.openmm.OpenMMEnergyMinimistation), 110
                                   attribute), 419
                                                                                                                                                                                                                  uncertainty (openff.evaluator.properties.EnthalpyOfMixing
tolerance (openff.evaluator.protocols.simulation.BaseEnergyMinimisationute), 114
                                   attribute), 518
                                                                                                                                                                                                                  uncertainty (openff.evaluator.properties.EnthalpyOfVaporization
topology_file_name (openff.evaluator.storage.data.StoredFreeEnergyPotae), 117
                                   attribute), 232
                                                                                                                                                                                                                  uncertainty (openff.evaluator.properties.ExcessMolarVolume
{\tt topology\_path} (\textit{openff.evaluator.protocols.paprika.analysis.Analyze} \underline{\textit{Apppphase}}_{107}
                                   attribute), 470
                                                                                                                                                                                                                  uncertainty (openff.evaluator.properties.HostGuestBindingAffinity
total_number_of_iterations
                                                                                                                                                                                                                                                      attribute), 124
                                  (openff.evaluator.protocols.openmm. Open MMS im {\tt uktior} {\tt tainty} (openff.evaluator.properties. Solvation Free Energy) {
                                   attribute), 426
                                                                                                                                                                                                                                                      attribute), 120
total_number_of_iterations
                                                                                                                                                                                                                  UNDEFINED (in module openff.evaluator.attributes), 566
                                   (open \emph{ff.} evaluator. protocols. simulation. Base Simulation \texttt{PackStoredSimulationData}) and \texttt{figure} ackStoredSimulationData \texttt{figure} ackStoredSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulationSimulat
                                                                                                                                                                                                                                                                                                                                                                                                                    in
                                   attribute), 524
                                                                                                                                                                                                                                                      openff.evaluator.protocols.storage), 529
total_number_of_molecules
                                                                                                                                                                                                                  unsuccessful_properties
                                   (open {\it ff.} evaluator. protocols. storage. Unpack {\it Stored Simulation} {\it Question for the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} we also a constraint {\it the protocols.} when {\it the protocols.} wh
                                                                                                                                                                                                                                                                                                                                                                                                                  at-
                                   attribute), 531
                                                                                                                                                                                                                                                     tribute), 86
trajectory_file_name
                                                                                                                                                                                                                  unsuccessful_properties
                                   (openff.evaluator.storage.data.StoredSimulationData
                                                                                                                                                                                                                                                      (openff.evaluator.server.Batch
                                                                                                                                                                                                                                                                                                                                                                                         attribute),
                                  attribute), 229
trajectory_file_path
                                                                                                                                                                                                                  update() (openff.evaluator.utils.observables.ObservableFrame
                                   (openff.evaluator.protocols.openmm.OpenMMEvaluateEnergieshod), 570
                                  attribute), 431
                                                                                                                                                                                                                  update_references()
trajectory_file_path
                                                                                                                                                                                                                                                     (openff.evaluator.workflow.schemas.ProtocolReplicator
                                   (open {\it ff. evaluator. protocols. open mm. Open MMS imulation}
                                                                                                                                                                                                                                                     method), 277
                                  attribute), 426
                                                                                                                                                                                                                  upper()
                                                                                                                                                                                                                                                                 (openff.evaluator.storage.attributes.FilePath
trajectory_file_path
                                                                                                                                                                                                                                                     method), 253
                                   (openff.evaluator.protocols.reweighting.BaseEvaluateEnergies
```

validate() (openff.evaluator.protocols.analysis.DecorrelateObservables

validate() (openff.evaluator.protocols.analysis.DecorrelateTrajectory

validate() (openff.evaluator.protocols.coordinates.BuildCoordinatesPack

validate() (openff.evaluator.protocols.coordinates.BuildDockedCoordi

validate() (openff.evaluator.protocols.coordinates.SolvateExistingStructure)

validate() (openff.evaluator.protocols.forcefield.BaseBuildSystem

V

- validate() (openff.evaluator.attributes.AttributeClass *method*), 565
- validate() (openff.evaluator.client.ConnectionOptions method), 81
- validate() (openff.evaluator.client.Request method), 83
- validate() (openff.evaluator.client.RequestOptions method), 85
- validate() (openff.evaluator.client.RequestResult method), 86
- ${\tt validate()}\ (open {\it ff. evaluator. datasets. Physical Property}$
- method), 95  $\textbf{validate()} \ (open \textit{ff.evaluator.datasets.PhysicalPropertyDataSet} \textbf{date()} \ (open \textit{ff.evaluator.protocols.forcefield.BuildLigParGenSystem}) \ (open \textit{ff.evaluator.datasets.PhysicalPropertyDataSet} \textbf{date()}) \ (open \textit{ff.evaluator.protocols.forcefield.BuildLigParGenSystem}) \ (open \textit{ff.evaluator.datasets.PhysicalPropertyDataSet}) \ (open \textit{ff.evaluato$
- method), 362 *method*), 140  ${\tt validate()}\ (open \textit{ff.evaluator.datasets.taproom.} \textit{TaproomD} \textit{Validate()}\ (open \textit{ff.evaluator.protocols.force} field. \textit{BuildSmirnoffSystem} \textit{Supplementary} \textit{Supp$

method), 328

method), 323

method), 334

method), 346

method), 341

method), 352

- method), 357 *method*), 150  $\textbf{validate()} \ (open \textit{ff.evaluator.datasets.thermoml.} The \textit{rmoMLP} \ \textit{idate()} \ (open \textit{ff.evaluator.protocols.force} field. \textit{BuildTLeapSystem} \ \textit{validate()} \ (open \textit{ff.evaluator.protocols.force} field. \textit{BuildTLeapSystem} \ \textit{validate()} \ (open \textit{ff.evaluator.datasets.thermoml.} \ \textit{ThermomLP} \ \textit{validate()} \ (open \textit{ff.evaluator.datasets.thermoml.} \ \textit{validate()} \ open \textit{ff.evaluator.datasets.thermoml.} \ open \textit{ff.evaluato$ method), 368
- method), 146  $\verb|validate()| (open \textit{ff.evaluator.layers.CalculationLayerResult}| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.layers.CalculationLayerResult})| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.protocols.gradients.ZeroGradients})| (open \textit{ff.evaluator.protocols.gradients})| (open \textit{ff.evaluator.$ method), 373
- *method*), 190  ${\tt validate()}\ (open\!f\!f\!.evaluator.layers. Calculation Layer Sche \color="chevaluator" Alberta Conditional Group to the color of th$
- method), 379 method), 191  $\textbf{validate()} \ (\textit{openff.evaluator.layers.reweighting.Reweighting.} \\ \textbf{king Schema ()} \ (\textit{openff.evaluator.protocols.groups.ConditionalGroup.ConditionalG$
- method), 376 method), 202 $\verb|validate()| (open ff. evaluator. layers. simulation. Simulation Schema \\ \verb|validate()| (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Add Values \\ | (open ff. evaluator. protocols. miscellaneous. Miscellaneous.$
- method), 384 method), 199  $\textbf{validate()} \ (open \textit{ff.evaluator.layers.workflow.WorkflowCallalidate()} \ (open \textit{ff.evaluator.protocols.miscellaneous.DivideValue}) \ (open \textit{ff.evaluator.layers.workflow.WorkflowCallalidate()}) \ (open \textit{ff.evaluator.protocols.miscellaneous.DivideValue}) \ (open \textit{ff.evaluator.layers.workflow.WorkflowCallalidate()}) \ (open \textit{ff.evaluator.layers.workflow.WorkflowCallalidate()}) \ (open \textit{ff.evaluator.layers.workflow.WorkflowCallalidate()}) \ (open \textit{ff.evaluator.layers.workflow.WorkflowCallalidate()}) \ (open \textit{ff.evaluator.layers.workflowCallalidate()}) \ (open \textit{ff.evalua$
- method), 399 method), 195
- validate() (openff.evaluator.protocols.miscellaneous.DummyProtocol (openff.evaluator.properties.Density validate() method), 414 method), 103  $\verb|validate()| (open \textit{ff.evaluator.properties.DielectricConstant}| \textbf{alidate()}| (open \textit{ff.evaluator.protocols.miscellaneous.FilterSubstanceByReleases)| \textbf{alidate()}| (open \textit{ff.evaluator.protocols.miscellaneous.FilterSubstanceByReleases)| \textbf{alidate()}| \textbf{aligner}| \textbf{align$
- method), 406 *method*), 110  $\verb|validate()| (open \textit{ff.evaluator.properties.EnthalpyOfMixing} \\ \textbf{validate()} (open \textit{ff.evaluator.protocols.miscellaneous.MultiplyValue} \\ \textbf{validate()} \\ \textbf{validate()} (open \textit{ff.evaluator.protocols.miscellaneous.MultiplyValue} \\ \textbf{validate()} \\ \textbf{validate$
- method), 394method), 114
- $\textbf{validate()} \ (\textit{openff.evaluator.protocols.miscellaneous.SubtractValues} \ (\textit{openff.evaluator.protocols.miscellaneous.SubtractValues}) \ (\textit{openff.evalues.protocols.miscellaneous.SubtractValues}) \ (\textit{openff.evalues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.SubtractValues.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneous.protocols.miscellaneo$ method), 389 method), 117
- $\verb|validate()| (open \textit{ff.evaluator.properties.ExcessMolarVolume} \\ | \textbf{lidate()}| (open \textit{ff.evaluator.protocols.miscellaneous.WeightByMoleFractions and } \\ | \textbf{fig.evaluator.protocols.miscellaneous.WeightByMoleFraction} \\ | \textbf{fig.evaluator.protocols.miscellaneous.MeightByMoleFraction} \\ | \textbf{fig.evaluator.protocols.miscellaneo$ method), 404 method), 107
- $\textbf{validate()} \ (\textit{openff.evaluator.properties.HostGuestBindingMaffinitate()} \ (\textit{openff.evaluator.protocols.openmm.OpenMMEnergyMinimised}) \ (\textit{openff.evaluator.protocols.openmm.Op$ method), 420 method), 124  ${\tt validate()}\ (open\!f\!f\!.evaluator.properties. Solvation Free Energian Language Control of the Control of t$
- method), 432 method), 120 validate() (openff.evaluator.protocols.analysis.AverageDietechnetechnication
- method), 426 method), 302
- $\textbf{validate()} \ (open \textit{ff.evaluator.protocols.analysis.AverageFreeEnergies()} \ (open \textit{ff.evaluator.protocols.paprika.analysis.AnalyzeAPRP has been expected as a protocol of the protocol$ method), 474 *method*), 304
- $\textbf{validate()} \ (\textit{openff.evaluator.protocols.analysis.AverageObservable} \textbf{(} \textit{openff.evaluator.protocols.paprika.analysis.ComputeReference and the protocols.paprika.analysis.ComputeReference and the protocols analysis.AverageObservable \textbf{(}) \textit{(} \textit{openff.evaluator.protocols.paprika.analysis.ComputeReference analysis.ComputeReference and the protocols.paprika.analysis.ComputeReference and the protocols analysis.AverageObservable \textbf{(}) \textit{(} \textit{openff.evaluator.protocols.paprika.analysis.ComputeReference analysis.ComputeReference analysis.ComputeReference$ method), 484*method*), 297
- validate() (openff.evaluator.protocols.analysis.BaseAverage biglate()) (openff.evaluator.protocols.paprika.analysis.ComputeSymmetric method), 479 method), 291
- $\textbf{validate()} \ (\textit{openff.evaluator.protocols.analysis.BaseDeco} \ \textit{PetateProtocolf.evaluator.protocols.paprika.coordinates.AddDummyAnalysis.BaseDecorrelateProtocolf.evaluator.protocols.paprika.coordinates.AddDummyAnalysis.BaseDecorrelateProtocolf.evaluator.proto$ *method*), 448 *method*), 318
- $\textbf{validate()} \ (open \textit{ff.evaluator.protocols.analysis.} Compute \textit{DipoleMoments} \ (open \textit{ff.evaluator.protocols.paprika.coordinates.} Prepare \textit{Pull Computed DipoleMoments} \ (open \textit{ff.evaluator.protocols.paprika.coordinates.} \ Prepare \textit{Pull Computed DipoleMoments} \ Prepa$ method), 437 *method*), 313

```
validate() (openff.evaluator.protocols.paprika.coordinates.PrepareRethroxe(Cabadinates
         method), 442
                                                        validate() (openff.evaluator.storage.query.FreeEnergyDataQuery
validate() (openff.evaluator.protocols.paprika.restraints.ApplyRestrainthod), 247
                                                        validate() (openff.evaluator.storage.query.SimulationDataQuery
         method), 469
validate() (openff.evaluator.protocols.paprika.restraints.GenerateAntathnRestraints
         method), 453
                                                        validate() (openff.evaluator.storage.query.SubstanceQuery
validate() (openff.evaluator.protocols.paprika.restraints.GeneratePmldfkext)กูลนิสิริ
                                                        validate()
         method), 458
                                                                            (openff.evaluator.substances.Amount
validate() (openff.evaluator.protocols.paprika.restraints.GenerateRelethsuRestraints
                                                        validate()
                                                                        (openff.evaluator.substances.Component
         method), 463
validate() (openff.evaluator.protocols.reweighting.BaseEvaluateEnergellesd), 130
                                                        validate() (openff.evaluator.substances.ExactAmount
         method), 499
validate() (openff.evaluator.protocols.reweighting.BaseMBARProtonethod), 134
         method), 505
                                                        validate() (openff.evaluator.substances.MoleFraction
validate() (openff.evaluator.protocols.reweighting.ConcatenateObsmethble), 135
         method), 494
                                                        validate()
                                                                          (openff.evaluator.substances.Substance
validate() (openff.evaluator.protocols.reweighting.ConcatenateTrajpnathroids), 128
         method), 489
                                                        validate() (openff.evaluator.thermodynamics.ThermodynamicState
validate() (openff.evaluator.protocols.reweighting.ReweightDielectnie@cols.taki37
         method), 516
                                                        validate()
                                                                             (openff.evaluator.workflow.Protocol
validate() (openff.evaluator.protocols.reweighting.ReweightObservaluthod), 265
         method), 511
                                                        validate() (openff.evaluator.workflow.ProtocolGroup
validate() (openff.evaluator.protocols.simulation.BaseEnergyMinimixathioat), 271
         method), 522
                                                        validate() (openff.evaluator.workflow.schemas.ProtocolGroupSchema
validate() (openff.evaluator.protocols.simulation.BaseSimulation method), 274
         method), 529
                                                        validate() (openff.evaluator.workflow.schemas.ProtocolSchema
validate() (openff.evaluator.protocols.storage.UnpackStoredSimulativentDoalin, 274
         method), 535
                                                        validate() (openff.evaluator.workflow.schemas.WorkflowSchema
validate() (openff.evaluator.protocols.yank.BaseYankProtocol
                                                                  method), 279
                                                        validate() (openff.evaluator.workflow.WorkflowResult
         method), 538
validate() (openff.evaluator.protocols.yank.LigandReceptorYankPromoclood), 260
         method), 549
                                                        value
                                                                     (openff.evaluator.datasets.PhysicalProperty
validate() (openff.evaluator.protocols.yank.SolvationYankProtocol attribute), 94
         method), 557
                                                        value
                                                                (openff.evaluator.properties.Density attribute),
validate() (openff.evaluator.server.Batch method), 92
validate() (openff.evaluator.storage.data.BaseSimulationDalme
                                                                  (openff.evaluator.properties.DielectricConstant
         method), 228
                                                                  attribute), 110
validate() (openff.evaluator.storage.data.BaseStoredDatavalue (openff.evaluator.properties.EnthalpyOfMixing at-
         method), 220
                                                                  tribute), 114
validate() (openff.evaluator.storage.data.ForceFieldDatavalue (openff.evaluator.properties.EnthalpyOfVaporization
         method), 223
                                                                  attribute), 117
validate() (openff.evaluator.storage.data.HashableStoredDatue (openff.evaluator.properties.ExcessMolarVolume
                                                                  attribute), 107
         method), 222
validate() (openff.evaluator.storage.data.ReplaceableDataalue (openff.evaluator.properties.HostGuestBindingAffinity
                                                                  attribute), 124
         method), 225
validate() (openff.evaluator.storage.data.StoredFreeEnerweData (openff.evaluator.properties.SolvationFreeEnergy
         method), 234
                                                                  attribute), 120
validate() (openff.evaluator.storage.data.StoredSimulationDlue (openff.evaluator.protocols.analysis.AverageDielectricConstant
         method), 231
                                                                  attribute), 302
validate() (openff.evaluator.storage.query.BaseDataQueryalue (openff.evaluator.protocols.analysis.AverageObservable
         method), 236
                                                                  attribute), 297
validate() (openff.evaluator.storage.query.BaseSimulationDlaceQapenff.evaluator.protocols.analysis.BaseAverageObservable
         method), 241
                                                                  attribute), 288
```

validate() (openff.evaluator.storage.query.ForceFieldQueralue (openff.evaluator.protocols.miscellaneous.DivideValue

	attribute), 395	WeightB	yMoleFraction	(class	in
value(a	ppenff.evaluator.protocols.miscellaneous.MultiplyV attribute), 390	/alue	openff.evaluator.protoc 399	ols.miscellaneou	us),
value(a	ppenff.evaluator.protocols.miscellaneous.WeightBy attribute), 400		dorvalue (openff.evalua attribute), 401	tor.protocols.mis	cellaneous.WeightByMole
value (a	ppenff.evaluator.protocols.reweighting.BaseMBAR attribute), 501		index (openff.evaluator attribute), 465	protocols.paprii	ka.restraints.ApplyRestrai
value(a	ppenff.evaluator.protocols.reweighting.ReweightDia attribute), 517		agaback() (openff.evaluethod), 88	luator.utils.excep	tions.EvaluatorException
value(a	ppenff.evaluator.protocols.reweighting.ReweightOb attribute), 511	swintdb <u>l</u> ter		luator.workflow.V	VorkflowException
value	(openff.evaluator.substances.Amount attribute), 131	Workflow	พ (class in openff.evalud พ_id(openff.evaluator.)		
value	(openff.evaluator.substances.ExactAmount at-		attribute), 259	· ·	
	tribute), 133	workflow	w_protocol()	(in	module
value	(openff.evaluator.substances.MoleFraction at-		openff.evaluator.workfl	ow), 272	
	tribute), 135	workflo	w_schema( <i>openff.evalu</i>	ator.layers.rewei	ighting.ReweightingSchem
value(a	ppenff.evaluator.utils.observables.Observable Array		attribute), 202		
value	property), 568 (openff.evaluator.workflow.WorkflowResult at-		w_schema ( <i>openff.evalu</i> attribute), 199	ator.layers.simul	lation.SimulationSchema
	tribute), 260	workflo	w_schema( <i>openff.evalu</i>	ator.layers.work	flow.WorkflowCalculation
value_a	a (openff.evaluator.protocols.miscellaneous.Subtradattribute), 386		<pre>attribute), 195 w_to_layer_result(</pre>	)	
value_k	o (openff.evaluator.protocols.miscellaneous.Subtrac attribute), 386		(openff.evaluator.layer. static method), 200	s.reweighting.Re	weightingLayer
values	openff.evaluator.protocols.analysis.AverageFreeE			)	
	attribute), 304		(openff.evaluator.layer.	s.simulation.Sim	ulationLayer
values	(openff.evaluator.protocols.miscellaneous.Add Valuation for the contraction of the cont	ies	static method), 197		
	attribute), 381	workflo	w_to_layer_result(	)	
values	() (openff.evaluator.utils.observables.ObservableF method), 570		(openff.evaluator.layer.static method), 193	s.workflow.Workf	AowCalculationLayer
verbose	e (openff.evaluator.protocols.yank.BaseYankProtoc attribute), 537		wCalculationLayer openff.evaluator.layers	(class workflow), 193	in
verbose	e (openff.evaluator.protocols.yank.LigandReceptor) attribute), 549	Yawoki/ku£lao		(class	in
verbose	e (openff.evaluator.protocols.yank.SolvationYankPi			,,	
	attribute), 557		wGraph (class in ope	enff.evaluator.wo	orkflow),
verbose	e_packmol (openff.evaluator.protocols.coordinates				•
	attribute), 330		wResult (class in ope	enff.evaluator.wo	orkflow),
verbose	e_packmol (openff.evaluator.protocols.coordinates				
	attribute), 341	Workflow	wSchema	(class	in
volumes	s (openff.evaluator.protocols.analysis.AverageDiele attribute), 298	ectricConst	t <b>wpt</b> enff.evaluator.workfl	ow.schemas), 27	8
volumes	s (openff.evaluator.protocols.reweighting.Reweight	D <b>Z</b> electric <b>(</b>	Constant		
	attribute), 512	ZeroGra		(class	in
			openff.evaluator.protoc	`	
W		zfill()	(openff.evaluator.ste		
wallclo	ock_time_limit		method), 253		
	(openff.evaluator.backends.QueueWorkerResource property), 206	ces			
water_r	nodel (openff.evaluator.protocols.forcefield.BuildL attribute), 362	.igParGenS	System		
water_r	nodel (openff.evaluator.protocols.forcefield.BuildT attribute), 368	LeapSyster	m		