Fedorov

Release 0.0.0

Aug 10, 2023

Contents

CHAPTER 1

Contents

1.1 Install with pip

To install the package with the package manager pip, execute

```
$ pip install fedorov --user
```

To upgrade the package, simply execute the same command with the `--upgrade` option.

```
$ pip install fedorov --user --upgrade
```

1.2 Install from source

Alternatively you can clone the git repository and execute the `setup.py` script to install the package.

```
git clone https://github.com/glotzerlab/fedorov.git
cd fedorov
python setup.py install --user
```

1.3 API Reference

This is the API for the **Fedorov** package.

1.3.1 Classes for 3D crystal initialization

This section contains three classes that allow user to initialize a 3D crystal structure with different user input.

class `fedorov.`**`SpaceGroup`**(*space_group_number=1*)

A class for space group symmetry operation.

This class provides method to initialize a crystal unit cell with space group and Wyckoff possition information.

Parameters **`space_group_number`** (*`int`*) – Space group number between 1 and 230.

All the space group information was obtained from Bilbao Crystallographic Server

`get_basis_vectors`(*base_positions*, *base_type=[]*, *base_quaternions=None*, *is_complete=False*, *apply_orientation=False*)

Get the basis vectors for the defined crystall structure.

Parameters

- **`base_positions`** (*`np.ndarray`*) – N by 3 np array of the Wyckoff postions
- **`base_type`** (*`list`*) – a list of string for particle type name
- **`base_quaternions`** (*`np.ndarray`*) – N by 4 np array of quaternions, default None
- **`is_complete`** (*`bool`*) – bool value to indicate if the positions are complete postions in a unitcell
- **`apply_orientations`** (*`bool`*) – bool value to indicate if the space group symmetry should be applied to orientatioin

Returns basis_vectors

Return type np.ndarray

`get_lattice_vectors`(***user_lattice_params*)

Initialize the unitcell and return lattice vectors [a1, a2, a3].

Parameters **`user_lattice_params`** (*`float`*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class `fedorov.`**`Prototype`**(*space_group_number=1*, *wyckoff_site="*, *type_by_site="*)

Crystal prototype class.

This class uses the minimal necessay information needed to fully define a crystal structures with space group number, wyckoff postions(in letter name convention) and free parameters for each relavent wyckoff postions.

Parameters

- **`space_group_number`** – space group number between 1 and 230
- **`wyckoff_site`** (*`str`*) – wyckoff site letters included in the prototype
- **`type_by_site`** (*`str`*) – type name letter for each site set in wyckoff_sites

All the space group and Wyckoff positions information was obtained from Bilbao Crystallographic Server

`get_basis_vectors`(***user_basis_params*)

Initialize fractional coordinates of the particles in the unitcell.

Parameters **`user_basis_params`** (*`float`*) – user defined parameters for different Wyckoff site degree of freedom, when applicable

Returns basis_vectors

Return type np.ndarray

get_lattice_vectors(***user_lattice_params*)

Initialize the unitcell and return lattice vectors [a1, a2, a3]

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class fedorov.**AflowPrototype**(*prototype_index=0*, *set_type=False*)

Bases: `fedorov.fedorov.Prototype`

Aflow prototype class.

This class uses the crystal prototypes in Aflow database to initialize crystal structures.

Parameters

- **prototype_index** (*int*) – prototype index [0, 589] for all 590 prototypes in AFLOW.
- **set_type** (*bool*) – allow setting different type name(in A, B, C order) for different atoms in AFLOW prototype

The list of crystal structures available in Aflow are summarized below:

classmethod from_query(*pearson_symbol: str | None = None*, *space_group: int | None = None*, *prototype: str | None = None*, *set_type: bool = False*)

Create all *AflowPrototype* matching the given query.

Args:

pearson_symbol (*str*, optional): The Pearson symbol to search for, defaults to `None` which accepts any Pearson symbol.

space_group (*int*, optional): The space group to search for, defaults to `None` which accepts any space group.

prototype (*str*, optional): The chemical prototype to search for, defaults to `None` which accepts any prototype.

set_type (*bool*, optional): Set different type name (in alphabetic order starting with "A") for different atoms in AFLOW prototype.

Returns:

lattices (list[*AflowPrototype*]): The list of all *AflowPrototype*'s with a given Pearson symbol.

1.3.2 Classes for 3D unit cell

class fedorov.**Triclinic**

A class for constructing a triclinic unitcell.

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a triclinic unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class `fedorov.`**`Monoclinic`**

A class for constructing a monoclinic unitcell

This class provides method to initialize a monoclinic unitcell

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a monoclinic unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class `fedorov.`**`Orthorhombic`**

A class for constructing a orthorhombic unitcell.

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a orthorhombi unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class `fedorov.`**`Tetragonal`**

A class for constructing a tetragonal unitcell.

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a tetragona unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class `fedorov.`**`Hexagonal`**

A class for constructing a hexagonal unitcell.

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a hexagonal unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class `fedorov.`**`Rhombohedral`**

A class for constructing a rhombohedral unitcell.

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a rhombohedral unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

class fedorov.**Cubic**

A class for constructing a cubic unitcell.

classmethod get_lattice_vectors(***user_lattice_params*)

Initialize a cubicc unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, c, alpha, beta, gamma where applicable

Returns lattice_vectors

Return type np.ndarray

1.3.3 Class for 2D crystal initialization

This section contains one class that allows user to initialize a 2D crystal structure with different user input.

class fedorov.**PlaneGroup**(*plane_group_number=1*)

A class for plane group symmetry operation.

This class provides method to initialize a crystal unit cell with plane group and Wyckoff possition information.

Parameters **plane_group_number** (*int*) – Plane group number between 1 and 17.

All the plane group information was obtained from Bilbao Crystallographic Server

get_basis_vectors(*base_positions*, *base_type=[]*, *base_quaternions=None*, *is_complete=False*, *apply_orientation=False*)

Get the basis vectors for the defined crystall structure.

Parameters

- **base_positions** (*np.ndarray*) – N by 2 np array of the Wyckoff postions
- **base_type** (*list*) – a list of string for particle type name
- **base_quaternions** (*np.ndarray*) – N by 4 np array of quaternions, default None
- **is_complete** (*bool*) – bool value to indicate if the positions are complete postions in a unitcell
- **apply_orientations** (*bool*) – bool value to indicate if the space group symmetry should be applied to orientation

Returns basis_vectors

Return type np.ndarray

get_lattice_vectors(***user_lattice_params*)

Initialize the unitcell and return lattice vectors [a1, a2].

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, theta where applicable

Returns lattice_vectors

Return type np.ndarray

1.3.4 Classes for 2D unit cell

class fedorov.**Oblique2D**

A class for constructing a 2D oblique unitcell

This class provides method to initialize a 2D oblique unitcell

classmethod get_lattice_vectors(***user_lattice_params*)
Initialize a 2D oblique unitcell and return lattice vectors [a1, a2].

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, theta where applicable

Returns lattice_vectors

Return type np.ndarray

class fedorov.**Rectangular2D**
A class for constructing a 2D rectangular unitcell

This class provides method to initialize a 2D rectangular unitcell

classmethod get_lattice_vectors(***user_lattice_params*)
Initialize a 2D rectangular unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, theta where applicable

Returns lattice_vectors

Return type np.ndarray

class fedorov.**Hexagonal2D**
A class for constructing a 2D hexagonal unitcell

This class provides method to initialize a 2D hexagonal unitcell

classmethod get_lattice_vectors(***user_lattice_params*)
Initialize a 2D hexagonal unitcell and return lattice vectors.

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, theta where applicable

Returns lattice_vectors

Return type np.ndarray

class fedorov.**Square2D**
A class for constructing a 2D square unitcell

This class provides method to initialize a 2D square unitcell

classmethod get_lattice_vectors(***user_lattice_params*)
Initialize a 2D square unitcell and return lattice vectors [a1, a2].

Parameters **user_lattice_params** (*float*) – unit cell parameters, provide a, b, theta where applicable

Returns lattice_vectors

Return type np.ndarray

1.3.5 Class for Point group symmetry operations

This section contains one class that allows user to obtain all point group symmetry operations.

class fedorov.**PointGroup**(*point_group_number=1*)
A class to access all point group symmetry operations.

This class provides method to access all point group symmetry operation in both rotational matrix form or quaternion form.

Parameters **`point_group_number`** (*`int`*) – Point group number between 1 and 32.

All the point group information was obtained from Bilbao Crystallographic Server

`get_quaternion``()`
Get the quaternions for the point group symmetry.

Returns list of quaternions

Return type list

`get_rotation_matrix``()`
Get the rotation matrixes for the point group symmetry.

Returns n by 3 by 3 numpy array containing n rotational matrixes

Return type numpy.ndarray

1.3.6 Some methods for crystal initialization

1.4 License

Fedorov is licensed under the **BSD-3-Clause License**:

1.5 Credits

1.5.1 Fedorov Developers

The following people contributed to the fedorov package.

Pengji Zhou <zhoupj@umich.edu>, University of Michigan - **Lead developer**.

Vyas Ramasubramani <vramasub@umich.edu>, University of Michigan

Brandon Butler <butlerbr@umich.edu>, University of Michigan

1.5.2 Libraries

Fedorov utilizes the following crystallographic database for different crystal structure information and symmetry operations:

- The pre-defined crystal structures are obtained by Aflow
- The space group, plane group and point group symmetry information are obtained from the Bilbao Crystallographic Server :

Users are encouraged to cite these references per the authors' guidelines when using fedorov.

1.5.3 Acknowledgments

Development for federov was supported in part by the National Science Foundation, Division of Materials Research Award # DMR 1808342, “CDS&E: Fast, scalable GPU-enabled software for predictive materials design & discovery”.

CHAPTER 2

Indices and tables

- genindex
- search

Index