
Fedorov

Release 0.0.0

Aug 10, 2023

1	Contents	1
1.1	Install with pip	1
1.2	Install from source	1
1.3	API Reference	1
1.3.1	Classes for 3D crystal initialization	1
1.3.2	Classes for 3D unit cell	3
1.3.3	Class for 2D crystal initialization	5
1.3.4	Classes for 2D unit cell	5
1.3.5	Class for Point group symmetry operations	6
1.3.6	Some methods for crystal initialization	7
1.4	License	7
1.5	Credits	8
1.5.1	Fedorov Developers	8
1.5.2	Libraries	8
1.5.3	Acknowledgments	8
2	Indices and tables	9
	Index	11

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 position 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 necessary 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 orthorhombic 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 tetragonal 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 position 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**:

```

BSD 3-Clause License for Fedorov

Copyright (c) 2019–2020 The Regents of the University of Michigan All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,
   this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributors
   may be used to endorse or promote products derived from this software without
   specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```

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 fedorov 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`

AAflowPrototype (*class in fedorov*), 3**C**Cubic (*class in fedorov*), 4**F**from_query() (*fedorov.AflowPrototype class method*), 3**G**get_basis_vectors() (*fedorov.PlaneGroup method*), 5get_basis_vectors() (*fedorov.Prototype method*), 2get_basis_vectors() (*fedorov.SpaceGroup method*), 2get_lattice_vectors() (*fedorov.Cubic class method*), 5get_lattice_vectors() (*fedorov.Hexagonal class method*), 4get_lattice_vectors() (*fedorov.Hexagonal2D class method*), 6get_lattice_vectors() (*fedorov.Monoclinic class method*), 4get_lattice_vectors() (*fedorov.Oblique2D class method*), 6get_lattice_vectors() (*fedorov.Orthorhombic class method*), 4get_lattice_vectors() (*fedorov.PlaneGroup method*), 5get_lattice_vectors() (*fedorov.Prototype method*), 2get_lattice_vectors() (*fedorov.Rectangular2D class method*), 6get_lattice_vectors() (*fedorov.Rhombohedral class method*), 4get_lattice_vectors() (*fedorov.SpaceGroup method*), 2get_lattice_vectors() (*fedorov.Square2D class method*), 6get_lattice_vectors() (*fedorov.Tetragonal class method*), 4get_lattice_vectors() (*fedorov.Triclinic class method*), 3get_quaternion() (*fedorov.PointGroup method*), 7get_rotation_matrix() (*fedorov.PointGroup method*), 7**H**Hexagonal (*class in fedorov*), 4Hexagonal2D (*class in fedorov*), 6**M**Monoclinic (*class in fedorov*), 3**O**Oblique2D (*class in fedorov*), 5Orthorhombic (*class in fedorov*), 4**P**PlaneGroup (*class in fedorov*), 5PointGroup (*class in fedorov*), 6Prototype (*class in fedorov*), 2**R**Rectangular2D (*class in fedorov*), 6Rhombohedral (*class in fedorov*), 4**S**SpaceGroup (*class in fedorov*), 1Square2D (*class in fedorov*), 6**T**Tetragonal (*class in fedorov*), 4Triclinic (*class in fedorov*), 3