# **Authors**

John P. Sutter, Diamond Light Source Ltd ([john.sutter@diamond.ac.uk](mailto:john.sutter@diamond.ac.uk))

Jacob Filik, Diamond Light Source Ltd ([jacob.filik@diamond.ac.uk](mailto:jacob.filik@diamond.ac.uk))

Alfred Q. R. Baron, RIKEN SPring-8 Center ([baron@spring8.or.jp](mailto:baron@spring8.or.jp))

James Pittard, University of Bristol ([jp17358@bristol.ac.uk](mailto:jp17358@bristol.ac.uk))

We also thank Taishun Manjo of the RIKEN SPring-8 Center Precision Spectroscopy Division for providing the material files of diamond, silicon, and germanium.

# **Purpose**

This document provides current information and examples for running PyCSFEx, a Python 3 software package developed for calculations of X-ray structure factors of complex crystals. Details of its development and structure, with particular emphasis on its use for α quartz, may be found in J. P. Sutter, J. Pittard, J. Filik & A. Q. R. Baron, “Calculating temperature-dependent X-ray structure factors of α quartz with an extensible Python 3 package,” *J. Appl. Cryst.* **55** (4), 1011-1028 (2022).

# **Requirements**

To run this package, a Python 3 interpreter with the modules below must be available:

* \_\_future\_\_
* abc
* h5py
* importlib
* math
* matplotlib
* numpy
* os
* pandas
* pkgutil
* re
* sys
* xlrd

# **Code structure and use**

## **Directories**

The code is contained within the directory PyCSFEx\_06Aug22 and includes the packages below:

* general\_crystals. Here definitions basic to all crystals are provided by module general\_crystal.py. All material files, which define particular crystals, are also included here. Users may write their own material files based on the examples provided in this distribution:
  + α quartz
    - alphaquartz\_dextro\_zp.py (dextro quartz, coordinates given in *z*(+) convention, anisotropic thermal treatment)
    - alphaquartz\_laevo\_zm.py (laevo quartz, coordinates given in *z*(–) convention, anisotropic thermal treatment)
    - alphaquartz\_dextro\_zp\_isodwf.py (dextro quartz, coordinates given in *z*(+) convention, isotropic Debye thermal treatment)
    - alphaquartz\_laevo\_zm\_isodwf.py (laevo quartz, coordinates given in *z*(–) convention, isotropic Debye thermal treatment)
  + Diamond
    - Diamond\_Origin2\_isodwf.py (coordinates given by origin choice 2 of *International Tables for Crystallography* Vol. A space group 227, isotropic Debye thermal treatment)
  + Silicon
    - Si\_Origin2\_isodwf.py (coordinates given by origin choice 2 of *International Tables for Crystallography* Vol. A space group 227, isotropic Debye thermal treatment)
  + Germanium
    - Ge\_Origin2\_isodwf.py (coordinates given by origin choice 2 of *International Tables for Crystallography* Vol. A space group 227, isotropic Debye thermal treatment)
* Structure\_Factor\_Calculator. This is the stable core of the PyCSFEx package. It should not be altered by the user.
* Examples. This contains examples and outputs of Python 3 scripts that apply the material files listed above.

The Microsoft Excel spreadsheet Form\_factor\_coefficients.xlsx is also included here and is part of the core that should not be altered by the user. It contains the fitting parameters for the atomic scattering factors.

# **Definition of the crystal**

An abstract base class GeneralCrystal is defined in general\_crystal.py. This includes a basic set of attributes and methods for all types of crystal:

* crystal system (cubic, tetragonal, orthorhombic, hexagonal, rhombohedral/trigonal, monoclinic, triclinic)
* unit cell parameters (sides and angles)
* determination of lattice vectors (module latt\_vec\_A)
* determination of the metric matrix (module G\_matrix\_A2)
* determination of the wavelength, interplanar spacing, and Bragg angle for Bragg reflection of X-rays of given energy from atomic planes of given Miller indices (module angle\_finder)

GeneralCrystal also includes a set of abstract methods that are named but not implemented:

* information: to print out details about the crystal
* lattice\_unit\_cell\_params: to set the lattice parameters
* refatom\_coordinates: to set the position of the prototype atom
* atoms\_init\_and\_update: to set the element types and determine the positions and thermal vibrations of all the atoms.

Objects cannot be produced from an abstract base class. However, each material file in general\_crystals defines a subclass of GeneralCrystal from which objects can be created. For example, in the file alphaquartz\_dextro\_zp.py that defines dextro α quartz in the setting, one defines

class AlphaQuartz\_Dextro\_zp(GeneralCrystal)

Each material file must provide an implementation for the abstract methods in GeneralCrystal, but the choice of implementation is left to the user. Material files may also be written to add new attributes and methods not included in GeneralCrystal. Each material file should include the lines

import numpy as np

from .general\_crystal import GeneralCrystal

and should also import the modules in Structure\_Factor\_Calculator below:

from Structure\_Factor\_Calculator.checks import Check

from Structure\_Factor\_Calculator.atom import Atom

from Structure\_Factor\_Calculator.diffraction\_environment import Diff\_Environment

from Structure\_Factor\_Calculator.tools import Tools

The module general\_crystal.py also includes a class CrystalFactory. The module \_\_init\_\_.py in general\_crystals imports this class so that all material files in general\_crystals can be read. Each script in Examples must include the lines

import sys

sys.path.append(“..”)

from general\_crystals import CrystalFactory

and must also import the definition of the crystal from its material file, for example as below:

from general\_crystals.alphaquartz\_dextro\_zp import AlphaQuartz\_Dextro\_zp

In the user scripts in Examples, instances of a crystal class are initialized as in the example below:

AQzD1 = AlphaQuartz\_Dextro\_zp(temp\_K,hkl,energy\_eV)

where the arguments are, respectively, the temperature in Kelvin, the Miller indices of the Bragg reflection, and the X-ray photon energy in eV. This calls the class method set\_temp\_miller\_energy(self,temperature\_K,hkl,energy\_eV), which sets the crystal’s temperature and calculates the corresponding unit cell and atomic positions. Each time the crystal’s temperature is changed, this method must be called. However, if only the Bragg reflection or X-ray energy is changed while the temperature remains the same, a simpler updating method set\_miller\_energy(self,hkl,energy\_eV) can be executed.

# **Structure of the package’s core**

The core of the program is the Python package Structure\_Factor\_Calculator. It consists of the following modules:

* atom.py (defines a class Atom, which sets the atomic properties)
* checks.py (defines a class Check, which contains input checking methods)
* diffraction\_environment.py (defines a class Diff\_Environment, which sets the diffraction parameters)
* physical\_constants.py (contains a list of fundamental physical constants)
* structure\_factor\_calc.py (defines a class Structure\_Factor that provides methods for calculating the structure factor)
* tools.py (contains various methods used in other modules).
* xrpy (calculates the anomalous terms of the atomic scattering factors according to the procedures of Brennan and Cowan (1992). This includes anomalous dispersion, Compton scattering, and Rayleigh scattering.) WARNING: More accurate calculations will be required if the chosen X-ray energy is close to an absorption edge. The effects of crystal structure on the absorption, which would appear in EXAFS spectra, are not considered.

# **Definition of the atom**

Each atom is defined as an instance of class Atom. Its attributes are defined below:

|  |  |
| --- | --- |
| **Attribute** | **Definition** |
| element | Atomic species (e.g. “C”, “Si” or “O”) |
| coordinates | Atom’s fractional coordinates in unit cell |
| cartesian\_coordinates | Cartesian coordinates of atom in Å |
| Environment | Instance of class Diff\_Environment containing diffraction parameters (below) |
| form\_factor\_coefficients | Fitting parameters for the photoelectric contribution to the atom’s scattering factor. Input automatically from the included Excel spreadsheet Form\_factor\_coefficients.xlsx |
| form\_factor | Total atomic scattering factor in the direction of the diffracted beam. Excludes Debye-Waller factor. |
| f0 | Photoelectric atomic scattering factor in the direction of the diffracted beam. |
| form\_factor\_fwd | Total atomic scattering factor for the forward direction. |
| beta\_matrix | Specification matrix of thermal ellipsoid if given. See Sutter *et al* (2022) for details. |
| M\_and\_TD | Atomic mass in amu and Debye temperature in K if given. This is used for an isotropic Debye treatment of thermal vibrations. |

The arguments below are used to create an instance of class Atom. NOTE: Either beta\_matrix or M\_and\_TD must be provided by the user, but not both.

|  |  |
| --- | --- |
| **Argument** | **Definition** |
| element | Atomic species (e.g. “C”, “Si” or “O”) |
| coord | Atom’s fractional coordinates in unit cell |
| environment\_obj | Instance of class Diff\_Environment containing diffraction parameters (below) |
| latt\_vec\_A | Lattice vectors in Cartesian coordinates in Å |
| beta\_matrix | Specification matrix of thermal ellipsoid if given. See Sutter *et al* (2022) for details. |
| M\_and\_TD | Atomic mass in amu and Debye temperature in K if given. This is used for an isotropic Debye treatment of thermal vibrations. |

Class Atom has the following methods:

* update: updates an already existing instance of class Atom
* information: prints out the attribute values
* data\_selecter: inputs the values of form\_factor\_coefficients from the Excel spreadsheet Form\_factor\_coefficients.xlsx
* scattering\_factor: sets the attributes form\_factor, f0 and form\_factor\_fwd.

# **Definition of the environment**

The environment is defined as an instance of class Diff\_Environment. It describes the Bragg reflection of X-rays of specified wavelength from the crystal at a given temperature. Its attributes are as follows:

|  |  |
| --- | --- |
| **Attribute** | **Definition** |
| temp\_K | Temperature of crystal in K |
| angle\_rad | Bragg angle in radians |
| angle\_deg | Bragg angle in degrees |
| hkl | Miller indices of selected Bragg reflection |
| wavelength\_A | X-ray wavelength in Å |
| d\_A | Spacing of diffracting atomic planes (hkl) |

Instances of class Diff\_Environment are initialized using the following arguments:

|  |  |
| --- | --- |
| **Argument** | **Definition** |
| temperature\_K | Temperature of crystal in K |
| crystal\_system | Type of unit cell. NOTE: For trigonal crystals in which the hexagonal unit cell is used, this should be given as “Hexagonal”. |
| unit\_cell\_params | Lattice parameters (sides and angles of unit cell) |
| angle\_plane\_wavelength | List with Bragg angle in radians, Miller indices, and X-ray wavelength in Å |

Class Diff\_Environment has the following methods:

* update: updates an already existing instance of class Diff\_Environment
* information: prints out the attribute values
* d\_hkl: calculates the spacing of the diffracting atomic planes (hkl)
* cell\_volume: calculates the unit cell’s volume.

# **The structure factor**

Class Structure\_Factor has the following methods:

* debye\_waller: calculates the Debye-Waller factor of a particular atom by using thermal ellipsoids
* isotropic\_debye\_waller: calculates the Debye-Waller factor of a particular atom by using an isotropic Debye model
* atom\_scat\_phase\_DW: multiplies an atom’s scattering factor by its Debye-Waller factor and its phase. Repetition of the same calculation for identical atoms is avoided to save runtime.
* F\_hkl: calculates the structure factor of the specified Bragg reflection (hkl), the Bragg reflection (), and forward scattering.
* SF\_output: prints out the structure factor and other useful information.

# **References**

Brennan, S. & Cowan, P. L. (1992). *Rev. Sci. Instrum.* **63**, 850-853.

Sutter, J. P., Pittard, J., Filik, J. & Baron, A. Q. R. (2022). *J. Appl. Cryst.* **55**, 1011-1028. <https://doi.org/10.1107/S1600576722005945>