pyEMsoft

Release 1.0

Dec 29, 2020

Contents:

EMsoft is an open source program for computation and visualization of scanning electron microscopy diffraction patterns such as EBSD,ECP,TKD, and EKP. pyEMsoft is a Python interface to EMsoft that provides some access of the Fortran types and subrouines. pyEMsoft is automatically generated from a Python interface generator for Fortran (f90wrap).

CHAPTER 1

Financial Support

EBSD/ECP/EKP development of this package, including dictionary indexing for EBSD/ECP, was started with support from an AFOSR/MURI grant, FA9550-12-1-0458; the original EBSD code from CTEMsoft 2.0 was developed with support from an ONR grant, N00014-12-1-0075. All recent development of TKD and related modalities, including the creation of routines that can generate PoVRay visualization script files, was performed with support from an ONR Vannevar Bush Fellowship, N00014-16-1-2821.

1.1 Installation

The pyEMsoft modules can only be generated after EMsoft has been built:

- To compile EMsoft, you need to first build the Software Developer Kit on your operating system.
- Then, follow the instructions in EMsoft to compile the EMsoft modules and programs.
- Toggle ON the build option to build dynamically linked libraris `BUILD_SHARED_LIBS=ON` e.g. libEMsoftLib.so in the CMakeCache.txt.
- Next, complete the EMsoft package configuration following the EMsoft Wiki Package Configuration.
- To test if EMsoft has been configured correctly,run a simple example such as Crystal Data Entry Example.
- Create a Ni.xtal crystal file in the XtalFolder folder and this will be later used for unittests.
- Have Python 3.x installed then pip install numpy (other packages are needed to run examples include: jupyter notebook, pyyaml, scikit-image, h5py, matplotlib).
- Git clone the f90wrap from our repository which contains minor changes for pyEMsoft. Install f90wrap Install with the setup.py (`python setup.py install`) and check if f90wrap and f2py-f90wrap have been added to path. These are the two important executables to generate the wrappers.
- In the EMsoftBuild, use the auto-generated shell scripts run_pyEMsoft.sh (for pyEMsoft module) to compile the extension modules (_pyEMsoft.*.so and pyEMsoft.py)
- In the EMsoftBuild, run_docs.sh can be used to generate a local copy of the documentation.

- In the Anaconda environment (if f90wrap is install in Anaconda environment), there might potentially be an issue in linking some of the dynamic libraries (see the Debugging section about how to fix these).
- Install pyEMsoft as a Python package with `python setup.py install` from the pyEMsoft build folder.

1.1.1 Dependencies

- Github
- EMsoft_SKD
- EMsoft
- Python 3.x (unittest files and examples provided in Python 3.x)
- recent version of numpy which includes support for f2py
- Fortran compiler gfortran 6.3+ or ifort 12+
- f90wrap

1.1.2 Supported Platforms

Currently, Windows system is not fully supported because f90wrap has only been tested on Mac and Linux system.

For Windows users, we recommend using the Windows Subsystem for Linux (WSL).

Operating System	C/C++ Compiler	Fortran Compiler
macOS (10.12)	Xcode Native tools (8.3.x)	GFortran 6.3.0 and above
Windows (10)	Visual Studio 2015 (CE/Pro)	Intel Fortran v17
Linux (Ubuntu 16.x, CentOS 7.x)	GCC 4.8 and Above/Clang 3.8 and greater	GNU Fortran 5.4.1 20160904 or newer

1.1.3 Debugging

For more technical aspects of the build process, please refer to a journal paper by Pearu Peterson

The error information regarding the build is logged in the build_error.log file. However, if incorrect libraries are linked, you are not gonna find any clue in the log file unless you import the built pyEMsoft module.

For macOC (Darwin):

To check what dynamic libraries are linked to the shared library file (.so), you can use `otool -L *.so` to check the linked dynamic libraries. Note that the Accelerate.framework is for the lapack library.

```
_pyEMsoft.cpython-37m-darwin.so:
/usr/local/gfortran/lib/libgomp.1.dylib (compatibility version 2.0.0, current version␣
↪2.0.0)
/usr/local/gfortran/lib/libgfortran.3.dylib (compatibility version 4.0.0, current␣
↪version 4.0.0)
/Applications/Build_SDK/EMsoft_SDK/fftw-3.3.8/lib/libfftw3.3.dylib (compatibility␣
↪version 9.0.0, current version 9.8.0)
/System/Library/Frameworks/Accelerate.framework/Versions/A/Accelerate (compatibility␣
↪version 1.0.0, current version 4.0.0)
/usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1238.60.2)
```

(continues on next page)

(continued from previous page)

```
/usr/local/gfortran/lib/libgcc_s.1.dylib (compatibility version 1.0.0, current␣
↪version 1.0.0)
/usr/local/gfortran/lib/libquadmath.0.dylib (compatibility version 1.0.0, current␣
↪version 1.0.0)
```

In Anaconda environment, the f90wrap will link against libraries in the ../Anaconda/lib folder which contains outdated version of libgfortran library (does not support ieee_arithmetic) and probably an incomplete libgomp (openMP) library.

If you are not certain what's missing in the library (usually import pyEMsoft will tell you), use the `nm` command to reveal all the linked (statically and dynamically) contents and compare with the linked contents (also `nm` command) of the shared library (.so). In mac, you can also use `install_name_tool -change old new` to fix the libraries without needing to rebuild pyEMsoft.

For Linux (Ubuntu):

To check what dynamic libraries are linked to the shared library file (.so), you can use `ldd *.so` to check the dynamically linked dynamic libraries. One potential issue might be related to the location of fftw3 library since for some systems it is located in the lib64 folder instead of lib folder.

Here is an example successfully built from WSL. Because a program needs to know where to look for library files, we must add that location to the environmental variable. For missing link to any .so file, add the environment variable using the following command: `export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/full path/to/so.file/`

```
linux-vdso.so.1 (0x00007ffecc7a6000)
libgfortran.so.5 => /lib/x86_64-linux-gnu/libgfortran.so.5 (0x00007f46f6f4a000)
libEMsoftLib.so => /home/chaoyi/EMsoftBuild/Release/Bin/libEMsoftLib.so␣
↪(0x00007f46f6143000)
libEMsoftHDFLib.so => /home/chaoyi/EMsoftBuild/Release/Bin/libEMsoftHDFLib.so␣
↪(0x00007f46f5bdf000)
libEMOpenCLLib.so => /home/chaoyi/EMsoftBuild/Release/Bin/libEMOpenCLLib.so␣
↪(0x00007f46f5bbc000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f46f59ca000)
libquadmath.so.0 => /lib/x86_64-linux-gnu/libquadmath.so.0 (0x00007f46f5980000)
libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007f46f582f000)
libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x00007f46f5814000)
/lib64/ld-linux-x86-64.so.2 (0x00007f46f7e17000)
libfftw3.so.3.5.7 => /home/chaoyi/EMsoft_SDK/fftw-3.3.8/lib/libfftw3.so.3.5.7␣
↪(0x00007f46f56ed000)
liblapack.so.3 => /lib/x86_64-linux-gnu/liblapack.so.3 (0x00007f46f5049000)
libmvec.so.1 => /lib/x86_64-linux-gnu/libmvec.so.1 (0x00007f46f501d000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f46f5015000)
libgomp.so.1 => /lib/x86_64-linux-gnu/libgomp.so.1 (0x00007f46f4fd3000)
libOpenCL.so.1 => /lib/x86_64-linux-gnu/libOpenCL.so.1 (0x00007f46f4dc8000)
libblas.so.3 => /lib/x86_64-linux-gnu/libblas.so.3 (0x00007f46f4d5b000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f46f4d38000)
```

1.1.4 Unit tests

A collection of unit tests can be found in the pyEMsoft/unittests folder. Execute any of them with the following command: `python test_XXXX.py`. Some of them might require a crystal file or euler angle file relative to the EMdatapathname.

1.1.5 Examples

We have included several examples of how pyEMsoft can be used as a python package.

- Example 1: how to create a cystal (xtal) file step by step with pyEMsoft.
- Example 2: basic crystallographic computations
- Example 3: plot a Kikuchi sphere from EBSD master pattern
- Example 4: read and analyze crystal data
- Example 5: how to use image filters
- Example 6: global optimization for pattern center, orientation and deformation tensor inference

1.2 pyEMsoft

1.2.1 Executing EMsoft program in Python

This is directly calling the built EMsoft functions from within Python. First, the `.../EMsoftBuild/Release/Bin` folder needs to be added to the path. Second, complete the EMsoft package configuration following the EMsoft Wiki Package Configuration. Then, create a Ni.xtal file as defined in Crystal Data Entry Example. This crystal file will later be read into some of the unittests files.

```
ChaoyideAir:source chaoyizhu$ python3.7
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)]
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> os.system('EMmkxtal')
```

1.2.2 How to use the pyEMsoft module?

Install pyEMsoft as a python package called EMsoft and then one can easily import the pyEMsoft module from any directoary.

```
# import the pyEMsoft module
from EMsoft import pyEMsoft
# import utilitymodule
from EMsoft import pyEMsoftTools
```

1.2.3 Constants (constants.f90)

This module contains physical and mathematical constants used by various programs such as periodic table information, atomic weights, etc.

A few examples have been tested in the unittest file. For instance, several examples of basic constants (in SI units) can be found in this module:

```
# value of pi, speed of light, Planck constant, Boltzmann constant
print('Value of \u03C0 is %.19f' % (pyEMsoft.constants.cpi), '\n')
print('Speed of light (c) is %.1f (m/s)' % (pyEMsoft.constants.clight), '\n')
```

(continues on next page)

(continued from previous page)

```
print('Planck constant (h) is %e (Js)' % (pyEMsoft.constants.cplanck), '\n')
print('Boltzmann Constant (k) is %e (m^2kgs^(-1)K^(-1))' % (pyEMsoft.constants.
↪cboltzmann), '\n')
```

For a complete list of constants, please refer to the original fortran file (constants.f90).

1.2.4 Typedefs (typedefs.f90)

It contains definition of all variables and types for crystallographic computations in EMsoft. More specifically, it defines the unitcell type and the orientation type (class type objects in python) , as well as the main cell variable used by all crystallographic computations.

For instance, one important variable that is used extensively within EMsoft is the `unitcell` type.

```
# use the unitcell class defined in the typedefs module
LatCell=pyEMsoft.typedefs.unitcell()
```

This `unitcell` can now be used as input/output variable in other associated routines.

Another special case of variable is the character array such as a list of space group name. The output numpy.array is in ASCII encoded format. To retrieve the characters, get_character_array function from the pyEMsoftTools can be used.

```
# convert the space group names from ASCII encoded numpy array to character array
Tools.get_character_array(pyEMsoft.typedefs.sym_sgname)
```

1.2.5 Quaternions (quaternions.f90)

Quaternions module contains basic quaternion functions such as quaternion normalization/complex conjugate/multiplication/division/inner product/rotation, interquaternion angle, random quaternion and interpolation between quaternions.

One function defined in the Quaternions module is to determine the norm of a given quaternion.

```
# define an arbitrary quaternion (single precision)
q = np.asarray([1, 2, 3, 4], dtype=np.float32)
# use the Quaternions module to find norm of the quaternion and obtain a normalized␣
↪quaternion
q  = q / pyEMsoft.quaternions.cabs(q)
# check the precision
print(q.dtype)
```

Note that the interface function e.g. `cabs` have both single precision (`_quat_norm`) and double precision (`_quat_norm_d`) routines 'glued together' in the Fortran script. Passing either single precision array (float32) or double precision array (float64) will default into the first single precision routine `_quat_norm` (at the moment)unless the `_quat_norm_d` is explicitly defined.

```
# define an arbitrary quaternion (double precision)
q = np.asarray([1, 2, 3, 4], dtype=np.float64)
# use the Quaternions module to find norm of the quaternion and obtain a normalized␣
↪quaternion
q  = q / pyEMsoft.quaternions._quat_norm_d(q)
# check the precision
print(q.dtype)
```

1.2.6 Rotations (rotations.f90)

This Rotations module contains everything that has to do with rotations and conversions between rotations. Details regarding this can be referred to the book[1] or a more recent tutorial paper[2]

For conversion from quaternion to orientation matrix, the `qu2eu` function can be called:

```
# define an arbitrary quaternion (single or double precision)
q = np.asarray([1, 2, 3, 4], dtype=np.float32)
# use the Quaternion module to find norm of the quaternion and obtain a normalized␣
↪quaternion
q  = q / pyEMsoft.Quaternions.cabs(q)
# determining the corresponding orientation matrix of the arbitrary quaternion
om = pyEMsoft.Rotations.qu2om(q)
```

To see if the lapack library is correctly linked, you can check if the `om2ax` routine outputs the correct value because it uses lapack to calculate the eigenvalue of a given matrix. A specific unittest is added in the test_rotations.py file for the verification of the correct use of lapack library.

In the rotations module, the `init_orientation` and `init_orientaiton_om` functions can be used to communicate with all the rotation conversion functions in the rotations module. By providing a random quaterion and looping over the rotation methods, all the rotation conversion functions can be tested. A double precision quaternion is defined first for the following example (see unittest script test_rotations.py):

```
# define some rotation methods (om has its dedicated routine)
# qu is not included because we are using it as a inputtype
# and qu2om, qu2eu, ... can be used to
rotation_method = ['qu','eu', 'ax', 'ro', 'ho', 'cu', 'st','om']
# get function from the pyEMsoft.rotations module
def get_function(str):
    return getattr(pyEMsoft.rotations, str)
# loop over the rotation method list
for i in rotation_method:
    # qu is already an input so it does not require conversion
    if i == 'qu':
        res = pyEMsoft.rotations._init_orientation_d(self.q, i, rotcheck=False)
        print(i, 'to other types\n', res,'\n')
    else:
        f = get_function('_qu2'+i+'_d')
        inputtype = f(self.q)
        # init_orientation_om is a separate function
        if i == 'om':
            res = pyEMsoft.rotations._init_orientation_om_d(inputtype, i,␣
↪rotcheck=True)
            print(i, 'to other types\n', res, '\n')
        else:
            res = pyEMsoft.rotations._init_orientation_d(inputtype, i, rotcheck=False)
            print(i, 'to other types\n', res, '\n')
```

In addition, functions in the rotations module involving checking the bound and norm of a given type of rotation, coordinate transformation of tensor and vectors (passive or active transformation), and a way of computing the geometrical mean of a list of quaternions (including its standard deviation quaternion).

[1] Morawiec, A., 2003. Orientations and rotations. Springer-Verlag.

[2] Rowenhorst, D., Rollett, A.D., Rohrer, G.S., Groeber, M., Jackson, M., Konijnenberg, P.J. and De Graef, M., 2015. Consistent representations of and conversions between 3D rotations. Modelling and Simulation in Materials Science and Engineering, 23(8), p.083501.

1.2.7 Math (math.f90)

Math module is a collection of mathematical/numerical routines. For instance, mathematical operations to obtain matrix inverse, cross-product, matrix determinant, cubic roots, etc.

Polar decomposition of a deformation gradient tensor can be carried out using the getpolardecomposition function in the math module.

Polar decomposition is one of the special cases (also the minvert function) where a matrix needs to be preallocated in python but the results are filled in the fortran code. The preallocation essentially defines an empty numpy array in Fortran order in memory.

```
>>> # define the deformation gradient tensor
>>> F=np.array([[1, 0.495, 0.5],[-0.333,1,-0.247],[0.959,0,1.5]], dtype=np.double)
>>> Rmatrix= np.asarray(np.zeros([3, 3]), dtype=np.double, order='F')
>>> Smatrix= np.asarray(np.zeros([3, 3]), dtype=np.double, order='F')
>>> pyEMsoft.Math.getpolardecomposition(F, Rmatrix, Smatrix)
>>> print('Polar decomposition of:\n',F, '\n\ngives rotation matrix\n',Rmatrix,
↪'\n\nand stretch matrix\n',Smatrix)
Polar decomposition of:
[[ 1.     0.495  0.5  ]
[-0.333  1.    -0.247]
[ 0.959  0.     1.5  ]]

gives rotation matrix
[[ 0.91432887  0.37693049 -0.14807474]
[-0.37389189  0.92618061  0.04893185]
[ 0.15558786  0.01062414  0.98776492]]

and stretch matrix
[[ 1.18804362  0.0787009   0.78289752]
[ 0.0787009   1.11276121 -0.02436515]
[ 0.78289752 -0.02436515  1.39552385]]
```

In addition, one can check the flags (part of numpy) for the Rmatrix or the Smatrix. In this case, Rmatrix is in a single, Fortran-style contiguous segment.

```
>>> print('\n Rmatrix FLAGS:\n', Rmatrix.flags, '\n')
Rmatrix FLAGS:
C_CONTIGUOUS : False
F_CONTIGUOUS : True
OWNDATA : True
WRITEABLE : True
ALIGNED : True
WRITEBACKIFCOPY : False
UPDATEIFCOPY : False
```

Whenever the output variable is preallocated in python, the interface function such as `minvert` can now switch between the single precision (`_minvert`) and double precision routines (`_minvert_d`), depending on the precision of the preallocated out variable. The try and except python function is wrapped around these routines of different precisions such that the prompted `ValueError` is used as a switch.

1.2.8 HDFsupport (HDFsupport.f90)

This module contains some of the HDF5 helper routines that can be used to export or import HDF5 data set. The routines within HDFsupport can already read/write EMsoft specific format data. This is probably more convenient than writing hdf5 specific module (see Example4_read_crystal_data)

One example routine from this module is able to save crystal data unitcell (Example 1 below shows how to create a crystal unitcell in python) to a .xtal file in the default XtalFolder (EMsoft package configuration is required).

```
# use routine from HDFsupport to save crystal data
pyEMsoft.hdfsupport.savedatahdf(LatCell)
```

In addition, it is also possible to read crystal data from a .xtal file from the XtalFolder using `crystaldata` funciton, written based on the `readdatahdf` function from the same HDFsupport module. Within this `crystaldata` function, addtional function `calcmatrices` (from crystal.f90) is called to computes the direct/reciprocal lattice/structure matrices for a given crystal.

```
# readin the existing hdf5 data (in the XtalFolder)
pyEMsoft.hdfsupport.crystaldata(LatCell)
```

1.2.9 EBSDmod (EBSDmod.f90)

This module contains several functions to work with EBSD input and output data. For instance, we can use it to read in a list of Euler angles, Monte Carlo data and master pattern data.

A list of Euler angles (.txt) needs to be first created in the EMsoft data folder (EMdatapathname). In the unittests file, the euler.txt is created, which contains two sets of Euler angles.

```
    # EBSD name list types
enl = pyEMsoft.namelisttypedefs.EBSDNameListType()
# define name of the file containing angles
enl.anglefile='euler.txt'
enl.eulerconvention='hkl'
#enl.anglefiletype = 'orientations'
angles=pyEMsoft.ebsdmod.EBSDAngleType()
# verbose=True converts eu to qu, hkl to tsl
numangles = pyEMsoft.ebsdmod.ebsdreadangles(enl,angles,verbose=True)
# the quaternions are saved in columns
print(angles,'\n')
```

It is required to first open the hdf5 interface through the `h5open_emsoft` (HDFsupport module) before we can use these functions such as `readebsdmasterpatternfile`.

```
# MPfile=input('Master pattern file (path relative to EMdatapathname):')
MPfile = 'Ni-master-20kV.h5'
# master pattern namelist types
mpnl = pyEMsoft.namelisttypedefs.EBSDMasterNameListType()
# Monte Carlo namelist types
mcnl = pyEMsoft.namelisttypedefs.MCCLNameListType()
# master pattern data types
EBSDMPdata = pyEMsoft.typedefs.EBSDMPdataType()
# Monte Carlo data types
EBSDMCdata = pyEMsoft.typedefs.EBSDMCdataType()
# hdferror (inout int), hdferror=0, no error; hdferror=1 means error returned
hdferr = np.asarray([0], dtype=int, order='F')
# open the hdf5 interface first
pyEMsoft.hdfsupport.h5open_emsoft(hdferr)
# readebsdmasterpatternfile is a Fortran routine that exports all relevant␣
↪information from
# Monte Carlo data. The following example shows how to get accum_e
pyEMsoft.ebsdmod.readebsdmontecarlofile(MPfile, mcnl, EBSDMCdata, getaccume=True)
# readebsdmasterpatternfile is a Fortran routine that exports all the information␣
↪from the master pattern
```

(continues on next page)

(continued from previous page)

```
# if keep4=True, this keeps the original rank 4 tensor can has an extra dimension␣
↪associated with atom types
# (e.g.EBSDMPdata.mlpnh4, EBSDMPdata.mlpsh4 )
pyEMsoft.ebsdmod.readebsdmasterpatternfile(MPfile, mpnl, EBSDMPdata,
getmlpnh=True, getmlpsh=True, getmasterspnh=True, getmasterspsh=True)
pyEMsoft.hdfsupport.h5close_emsoft(hdferr)
```

1.2.10 Crystal (crystal.f90)

The Crystal module includes distance and angle computations, coordinate transformations, normalizations, dot and cross products, generation of asymmetric positions; also some routines that deal with reading lattice parameters and atom coordinates and such.

Given the space group of a crystal, we can find out the corresponding crystal system using the `getemsoftxtalsystem`:

```
# input a space group for fcc crystal (should be in the cubic crystal system=1)
>>> pyEMsoft.crystal.getemsoftxtalsystem(225)
1
```

To display the periodic table, a `displayelements` function can be called from the crystal module. This routine simply uses message routine defined in io.f90 to directly print out strings to the terminal.

```
# display the elements in the periodic table
pyEMsoft.crystal.displayelements()
```

A more complicated scenario involves use of `unitcell` defined in the Typedefs module. The following example uses the unitcell as an input/output [intent(inout)] in the `getlatparm` function to define crystal structure and lattice parameters/angles.

```
LatCell=pyEMsoft.typedefs.unitcell()
pyEMsoft.crystal.getlatparm(LatCell)
```

The crystal structure information can be obtained in two ways: 1) either read from an existing .xtal file (as in the unittest file), 2) or go through the steps in Example1_make_crystal.py.

```
# a crystal unitcell needs to be created before testing the routines
# define the unitcell using typedefs to store crystallographic data
LatCell = pyEMsoft.typedefs.unitcell()
# file name of the crystal data file
LatCell.fname = 'Ni.xtal'
# readin the existing hdf5 data (in the XtalFolder)
# this function also uses readDataHDF (HDFsupport.f90) and CalcMatrices (crystal.f90)
pyEMsoft.hdfsupport.crystaldata(LatCell)
```

In some cases, the direct lattice vectors may need to be transformed to reciprocal space or cartesian reference frame. The `transspace` routine can be used to convert a vector between the three spaces with a single character as a switch: direct space ('d'), reciprocal space ('r'), cartesian reference frame ('c').

```
# define an arbitrary input vector in the direct space
input_vector = np.asarray([1, 1, 1], dtype=np.float32)
# define the output array first in fortran order in memory
output_vector = np.asarray([0, 0, 0], dtype=np.float32, order='F')
# define the space of input vector (direct space)
```

(continues on next page)

(continued from previous page)

```
inspace = 'd'
# define hte space of the output vector (standard cartesian reference frame)
outspace = 'r'
# now call the transspace from crystal module to convert the input vector into␣
↪another space
pyEMsoft.crystal.transspace(self.LatCell, input_vector, output_vector, inspace,␣
↪outspace)
print('The', input_vector, 'in the ', Tools.get_space_string(inspace), 'has been␣
↪converted to', output_vector,'in', Tools.get_space_string(outspace), '\n')
```

If coordinate transformation is needed in a given space ('d', 'c' or 'r'), the `transcoor` function can be called to defined a coordinate transformed vector from old to new ('on') or new to old ('no').

```
# vector components involving a transformation matrix
# define the output array first in fortran order in memory (double precision)
output_vector_t = np.asarray([0, 0, 0], dtype=np.float64, order='F')
# the transformation here is defined from a random quaternion (ideally this should
# be matrix with directional cosines formed by the basis vectors of two coordinates␣
↪systems (old and new)
trans_m = pyEMsoft.rotations.qu2om(self.q)
# call the transcoor function for the coordinate transformation in a defined space (
↪'on'=old to new, 'no'=new to old)
pyEMsoft.crystal.transcoor(self.LatCell, output_vector, output_vector_t, trans_m, 'c',
↪ 'on')
print('The output vector is', output_vector_t, 'under the transformation matrix\n',␣
↪trans_m, '\n')
```

Furthermore, there is a module called `milbrav` to help with conversion between Miller indices and Miller-Bravais indices ('34' or '43' is the switch).

```
# first we do a Miller to Miller-Bravais indices conversion (switch:'34')
Miller_index = np.asarray([1, 0, 1], dtype=np.int32)
Miller_Bravais_index = np.asarray([0, 0, 0, 0], dtype=np.int32, order='F')
pyEMsoft.crystal.milbrav(Miller_index, Miller_Bravais_index, '34')
print('Miller indices', Miller_index, 'is converted to Miller-Bravais indices:',␣
↪Miller_Bravais_index,'\n')
# then we do a Miller-Bravais to Miller indices conversion (switch:'43')
Miller_index = np.asarray([0, 0, 0], dtype=np.int32, order='F')
pyEMsoft.crystal.milbrav(Miller_index, Miller_Bravais_index, '43')
print('Miller-Bravais indices', Miller_Bravais_index, 'is converted to Miller indices:
↪', Miller_index,'\n')
```

To obtain density of a crystal structure (and average atomic weight), we can first find all the atom positions in a unit cell (symmetry.calcpositions) and then use `calcdensity` from the crystal module.

```
# calculate positions of atoms in the unit cell
pyEMsoft.symmetry.calcpositions(self.LatCell, 'v')
# calculate density, average atomic number, average atomic weight
density, avZ, avA = pyEMsoft.crystal.calcdensity(self.LatCell)
print(density, avZ, avA)
print('Density=', density, '(g/cm^3)', 'average atomic number=', avZ, 'average atomic␣
↪weight=', avA, '(g/mol)\n')
print('unit cell volume', self.LatCell.vol)
```

Moreover, the Crystal module contains many other useful tools to work with crystallography such vector normalization, length of vector, angle between vectors, cross product of two vectors, etc for any given space. An example is given below in Example2_basic_crystallography to solve some of the problems in the textbook written by Marc De

Graef[3]

1.2.11 Symmetry (symmetry.f90)

The Symmetry module deals with all symmetry-related routines. This includes routines to generate a space group based on the generator string; computation of orbits and families; computation of all atoms in a single or multiple unit cells.

In the corresponding unittest file (test_symmetry.py), the crystal structure information is directly read from an existing Ni.xtal file (if this does not exist, you need to create one).

The `isgallowed` function helps to determine whether an input (integer array) diffraction vector is forbidden due to precense of a certain type of atom centering.

```
# define three vectors in reciprocal space (integer arrays)
g1 = np.array([1, 1, 1])
g2 = np.array([1, 0, 1])
g3 = np.array([2, 2, 4])
# decode the bytes to utf-8 strings to get space symbols
space_group_name = (self.LatCell.sg.sym_name).decode('utf-8')
print('Is reflection g1', g1, 'allowed in', '\'', space_group_name[1],
'\'', 'type centering?')
print('Answer:', bool(pyEMsoft.symmetry.isgallowed(self.LatCell, g1)),
'\n')
print('Is reflection g2', g2, 'allowed in', '\'', space_group_name[1],
'\'', 'type centering?')
print('Answer:', bool(pyEMsoft.symmetry.isgallowed(self.LatCell, g2)),
'\n')
print('Is reflection g3', g3, 'allowed in', '\'', space_group_name[1],
'\'', 'type centering?')
print('Answer:', bool(pyEMsoft.symmetry.isgallowed(self.LatCell, g3)),
'\n')
```

With the `getpatternsymmetry` function, diffraction group, crystal point group, Laune group, projection diffraction group, and many 2D symmetry point groups (for bright field, dark field, whole pattern diffraction) can be accessed through some input variable e.g. crystal structure data (unitcell), crystal point group number, and zone axis ([uvw]).

Another useful routine is the `calcpositions`, which is used to calculate a list of atom positions for every atom type in the crystal. For example, this is used in `calcdensity`.

1.2.12 Lambert (lambert.f90)

This module contains a number of projection functions for the modified Lambert projection between square lattice and 3D hemisphere[4], hexagonal lattice and 3D hemisphere, as well as the more complex mapping between a 3D cubic grid and the unit quaternion hemisphere with positive scalar component. In addition, there are some other projections, such as the stereographic ones. Each function is named by the projection, the dimensionality of the starting grid, and the forward or inverse character. For each function, there is also a single precision and a double precision version, but we use the interface formalism to have only a single call. The Forward mapping is taken to be the one from the simple grid to the curved grid. Since the module deals with various grids, we also add a few functions/subroutines that apply symmetry operations on those grids.

[3] De Graef, M., 2003. Introduction to conventional transmission electron microscopy. Cambridge University Press.

[4] Callahan, P.G. and De Graef, M., 2013. Dynamical electron backscatter diffraction patterns. Part I: Pattern simulations. Microscopy and Microanalysis, 19(5), pp.1255-1265.

```
# 2D square coordinates
xy = np.asarray([0.3, 0.2], dtype=np.float32)
# return 1 if the point lies outside the bounds
ierr = np.asarray([0], dtype=np.int32, order='F')
# 2D square coordinates to 3D hemisphere transformation
xyz = pyEMsoft.lambert.lambertsquaretosphere(xy, ierr)
print('2D square coordinates', xy, 'is transformed into 3D coordinates',
xyz, '\n')
# 3D hemisphere to 2D square coordinates transformation
xy = pyEMsoft.lambert.lambertspheretosquare(xyz, ierr)
print('3D coordinates', xyz, 'is transformed into 2D square coordinates',
xy, '\n')
```

An example that involves use of the `lambertsquaretosphere` is given in the jupyter notebook file to plot Kikuchi sphere Example3_plot_Kikuchi_Sphere. First, master lamber projection patterns (weighted by atom occupancy) are weighted average based on Monte Carlo yield. 2D square coordinates are prescaled into the bounds and are then converted into 3D hemispherical coordiantes for the southern and northern hemispheress respectively.

1.2.13 Diffraction (diffraction.f90)

This module contains many routines used in the dynamical diffraction. In the unittests script, two classes of derived are used: `unitcell` (crystal) and `gnode` (diffraction related quantaties). Moreover, an example crystal data file (Ni.xtal) is read from the XtalFolder.

Most of the physical quantaties related to dyanmical diffraction can be obtained through the following `getvoltage` function, which communates to many other routines in the diffraction module.

```
# get the accelerating voltage, relativistic correction factor
# relativistic accelerating potential, electron wavelength
# obtain scattering factor sets, mean inner potential, interaction constant
# this routine uses many other rountines such as getscatfac, CalcUcg, Calcwavelength
pyEMsoft.diffraction.getvoltage(self.LatCell, self.rlp, True)
```

From the kinematical diffraction theory, the 2 theta diffraction angle can be calculated for any diffracting plane:

```
# calculate 2theta diffraction angle for a plane (hkl)
Angle = pyEMsoft.diffraction.calcdiffangle(self.LatCell, 1, 1, 1)
print('\nDiffraction angle for (111) is:', Angle, '(rad)\n')
```

1.2.14 Examples

The Symmetry module in combination with some functions in the Crystal module. A `unitcell` containing all the crystllographic information can be generated. User can either interact with the terminal to populate the unitcell with crystallographic information or define the values in the `unicell` manually e.g. LatCell.xtal_system=1.

Example 1: Make a crystal

```
from EMsoft import pyEMsoft
# define the unitcell usinge typedefs
LatCell = pyEMsoft.typedefs.unitcell()
# set the crystal system and lattice parameters/angles
pyEMsoft.crystal.getlatparm(LatCell)
# set the space group number
pyEMsoft.symmetry.getspacegroup(LatCell)
```

(continues on next page)

(continued from previous page)

```
# set space group setting
pyEMsoft.symmetry.generatesymmetry(LatCell, True)
# set atom types, fractional coordiantesm, site occupation and Debye-Waller factor
pyEMsoft.crystal.getasympos(LatCell)
# input file name (.xtal)
LatCell.fname=input('File name of the crystal file (.xtal):')
# source of crystal data
LatCell.source=input('Source of crystal data:')
# print the unitcell containing crystallographic information
print('\nCrystal System:', LatCell.xtal_system,'\n')
print('Lattice Parameters:\na= %.04f nm\nb= %.04f nm\nc= %.04f nm\n'
% (LatCell.a, LatCell.b, LatCell.c), '\n')
print('Angles:\n\u03B1= %.04f\u00b0\n\u03B2= %.04f\u00b0\n\u03B3= %.04f\u00b0\n'
% (LatCell.alpha, LatCell.beta, LatCell.gamma),'\n')
print('Space Group Number:', LatCell.sym_sgnum, '\n')
print('Space Group Setting:', LatCell.sym_sgset, '\n')
print('Number of Atom Types', LatCell.atom_ntype, '\n')
print('Atom Type (atomic number):', LatCell.atom_type[0:LatCell.atom_ntype], '\n')
print('Fractional coordiantes, Site occupation and Debye-Waller factor: \n',
LatCell.atom_pos[0:LatCell.atom_ntype, 0:5], '\n')
# empty spaces are reserved for more input characters
print('File name:', LatCell.fname.decode('utf-8'), '\n')
print('Source of crystal data:', LatCell.source.decode('utf-8'))
# use routine from HDFsupport to save crystal data in the XtalFolder
pyEMsoft.hdfsupport.savedatahdf(LatCell)
```

Example 2: Basic Crystallography Computations

In the following example, several routines in the Crystal module are utilized to solve some simple crystallography problems[3] .

```
from EMsoft import pyEMsoft
from EMsoft.pyEMsoftTools import Tools
import numpy as np
import math

# Examples here are taken from A textbook written by Professor Marc De Graef:
# De Graef, M., 2003. Introduction to conventional transmission electron microscopy.
# Cambridge University Press.
```

Task 1: a tetragonal crystal has lattice parameters a=1/2 nm and c=1 nm. Compute its metric tensor:

```
# Example 1.1
# define the unitcell usinge typedefs
LatCell = pyEMsoft.typedefs.unitcell()
# set the crystal system and lattice parameters/angles
# use a=0.5, c=1
pyEMsoft.crystal.getlatparm(LatCell)
# calculate the reciprocal/direct metric/structure tensors
pyEMsoft.crystal.calcmatrices(LatCell)
print('\nDirect metric tensor of the tetragonal crystal with a=0.5, c=1 is:\n',
np.round(LatCell.dmt,2), '\n')
```

Task 2: compute the distance between the points (1/2,0,1/2) and (1/2,1/2,0) in direct lattice space for the crystal defined in Task 1.

```
# Example 1.2
# distance between the two points is equal to the length of the vector
# connecting the two points in direct space
p1 = np.array([0.5, 0, 0.5])
p2 = np.array([0.5, 0.5, 0])
v = p1-p2
# define the space (d=direct space, r=reciprocal space, c=cartesian reference frame)
space = 'd'
v_length = pyEMsoft.crystal.calclength(LatCell, v, space)
print('Length between', p1, 'and', p2, 'is', v_length, '(nm) in the',
Tools.get_space_string(space))
```

Task 3: compute the dot product and angles between the vectors [120] and [311] for the crystal defined in Task 1.

```
# Example 1.3
# dot product and angle between two vectors in the direct space
v1 = np.array([1, 2, 0])
v2 = np.array([3, 1, 1])
v1_dot_v2 = pyEMsoft.crystal.calcdot(LatCell, v1, v2, space)
print('Dot product of', v1, 'and', v2, 'is', v1_dot_v2, '(nm^2) in the',
Tools.get_space_string(space))
# angle between two vectors in the direct space
a = pyEMsoft.crystal.calcangle(LatCell, v1, v2, space)
print('The angle between', v1, 'and', v2, 'is %4f' % math.degrees(a))
```

Task 4: compute the reciprocal metric tensor for the crystal defined in Task 1

```
# Example 1.5 & 1.9
print('\nReciprocal metric tensor of the tetragonal crystal with a=0.5,c=1 is:\n',
np.round(LatCell.rmt,2), '\n')
# The reciprocal lattice vectors are just rows of the the recirpocal metric tensor
```

Task 5: compute the angle between the (120) and (311) plane normals for the crystal defined in Task 1.

```
# Example 1.6
# given two plane normals
n1 = np.asarray([1, 2, 0])
n2 = np.asarray([3, 1, 1])
space = 'r'
a = pyEMsoft.crystal.calcangle(LatCell, n1, n2, space)
print('The angle between', n1, 'and', n2, 'is %4f' % math.degrees(a), '\n')
```

Task 6: write down the reciprocal components of the lattice vector [114] for the crystal defined in Task 1.

```
# Example 1.8
v = np.asarray([1, 1, 4])
n = np.asarray([0, 0, 0], dtype=np.float32, order='F')
inspace = 'd'
outspace = 'r'
pyEMsoft.crystal.transspace(LatCell, v, n, inspace, outspace)
print('The', v, 'in the ', Tools.get_space_string(inspace), 'has been converted to',
n,'in', Tools.get_space_string(outspace), '\n')
```

Task 7: determine the cross product of the vectors [110] and [111] in the crystal defined in Task 1.

```
# Example 1.10
 v1 = np.asarray([1, 1, 0], dtype=np.float32)
```

(continues on next page)

(continued from previous page)

```
v2 = np.asarray([1, 1, 1], dtype=np.float32)
output_vector = np.asarray([0, 0, 0], dtype=np.float32, order='F')
# define the space of input vector (direct space)
inspace = 'd'
# define the space of the output vector (reciprocal space)
outspace = 'r'
pyEMsoft.crystal.calccross(LatCell, v1, v2, output_vector, inspace, outspace, True)
print('The output vector cross product is', output_vector,
'in the', Tools.get_space_string(outspace))
pyEMsoft.crystal.calccross(LatCell, v1, v2, output_vector, inspace, inspace, True)
print('or',output_vector,'in the', Tools.get_space_string(inspace), '\n')
```

Task 8: determine the cross product of the reciprocal lattice vectors [110] and [111] in the crystal defined in Task 1.

```
# Example 1.12
g1 = np.asarray([0, 0, 0], dtype=np.float32, order='F')
g2 = np.asarray([0, 0, 0], dtype=np.float32, order='F')
output_vector = np.asarray([0, 0, 0], dtype=np.float32, order='F')
# determine the reciprocal space vectors for v1 and v2
pyEMsoft.crystal.transspace(LatCell, v1, g1, inspace, outspace)
pyEMsoft.crystal.transspace(LatCell, v2, g2, inspace, outspace)
# redefine the space of input vector (reciprocal space)
inspace = 'r'
# redefine the space of the output vector (direct space)
outspace = 'd'
pyEMsoft.crystal.calccross(LatCell, g1, g2, output_vector, inspace, outspace, True)
print('The output vector cross product is', output_vector,
'in the', Tools.get_space_string(outspace))
pyEMsoft.crystal.calccross(LatCell, g1, g2, output_vector, inspace, inspace, True)
print('or',output_vector,'in the', Tools.get_space_string(inspace), '\n')
```

1.3 pyEMsoftLib

This page contains all the wrappped types and subroutines

class pyEMsoft.**Clsupport**(**args*, ***kwargs*)

Module clsupport

Defined at CLsupport.f90 lines 28-745

static clerror_check(*routine*, *ierr*[, *nonfatal*])

Defined at CLsupport.f90 lines 604-745

Parameters

- **routine** (*str*) –
- **ierr** (*int*) –
- **nonfatal** (*bool*) –

static clquery_platform_info(*platform_id*)

Defined at CLsupport.f90 lines 75-270

Parameters **platform_id** (*int*) –

static clread_source_file(*sourcefile*)

csource, slength = clread_source_file(sourcefile)

Defined at CLsupport.f90 lines 287-364

Parameters **sourcefile** (*str*) –

Returns

- **csource** (*str*)
- **slength** (*int*)

static clread_source_file_wrapper(*sourcefile*)
csource, slength = clread_source_file_wrapper(sourcefile)

Defined at CLsupport.f90 lines 381-418

Parameters **sourcefile** (*str*) –

Returns

- **csource** (*str*)
- **slength** (*int*)

class pyEMsoft.**Commonmod**(**args*, ***kwargs*)
Module commonmod

Defined at commonmod.f90 lines 1-539

static ebsdgetaverageorientations(*ipar*, *eulers*, *tmi*, *dplist*, *aveuler*[, *disorient*])
Defined at commonmod.f90 lines 155-271

Parameters

- **ipar** (*int array*) –
- **eulers** (*float array*) –
- **tmi** (*int array*) –
- **dplist** (*float array*) –
- **aveuler** (*float array*) –
- **disorient** (*float array*) –
- **==========** – set up the symmetry quaternions for this rotational symmetry allocate the dict structure

static ebsdgetindexingsuccessmap(*ipar*, *tmi*, *ea*, *ebsdnl*, *ism*)
Defined at commonmod.f90 lines 389-443

Parameters

- **ipar** (*int array*) –
- **tmi** (*int array*) –
- **ea** (*float array*) –
- **ebsdnl** (*Ebsdindexingnamelisttype*) –
- **ism** (*float array*) –

static ebsdgetkammap(*numeu*, *eulers*, *ipf_wd*, *ipf_ht*, *dict*, *kam*)
Defined at commonmod.f90 lines 463-539

Parameters

- **numeu** (*int*) –

- **eulers** (*float array*) –
- **ipf_wd** (*int*) –
- **ipf_ht** (*int*) –
- **dict** (*Dicttype*) –
- **kam** (*float array*) –

static ebsdgetorientationsimilaritymap(*idims*, *tmi*, *nm*, *ipf_wd*, *ipf_ht*, *osm*)
Defined at commonmod.f90 lines 290-371

Parameters

- **idims** (*int array*) –
- **tmi** (*int array*) –
- **nm** (*int*) –
- **ipf_wd** (*int*) –
- **ipf_ht** (*int*) –
- **osm** (*float array*) –

class pyEMsoft.**Constants**(**args*, ***kwargs*)
Module constants

Defined at constants.f90 lines 63-412

atom_color
Element atom_color ftype=character(3) pytype=str

Defined at constants.f90 line 184

atom_colors
Element atom_colors ftype=real(kind=sgl) pytype=float

Defined at constants.f90 line 277

atom_mtradii
Element atom_mtradii ftype=real(kind=sgl) pytype=float

Defined at constants.f90 line 172

atom_spradii
Element atom_spradii ftype=real(kind=sgl) pytype=float

Defined at constants.f90 line 160

atom_sym
Element atom_sym ftype=character(2) pytype=str

Defined at constants.f90 line 148

atom_weights
Element atom_weights ftype=real(kind=sgl) pytype=float

Defined at constants.f90 line 299

butterfly9x9
Element butterfly9x9 ftype=real(kind=sgl) pytype=float

Defined at constants.f90 line 398

cavogadro
Element cavogadro ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cboltzmann
Element cboltzmann ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

ccharge
Element ccharge ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cj2ev
Element cj2ev ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

clight
Element clight ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cmoment
Element cmoment ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cpermea
Element cpermea ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cpermit
Element cpermit ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cpi
Element cpi ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

cplanck
Element cplanck ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

crestmass
Element crestmass ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 126

epsijk
Element epsijk ftype=real(kind=sgl) pytype=float

Defined at constants.f90 line 80

epsijkd
Element epsijkd ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 81

fzoarray
Element fzoarray ftype=integer(kind=irg) pytype=int

Defined at constants.f90 line 388

fztarray
Element fztarray ftype=integer(kind=irg) pytype=int

Defined at constants.f90 line 385

icovertices
Element icovertices ftype=real(kind=dbl) pytype=float

Defined at constants.f90 line 412

class pyEMsoft.**Crystal**(**args*, ***kwargs*)
Module crystal

Defined at crystal.f90 lines 52-1868

static calcangle(**args*, ***kwargs*)
Defined at crystal.f90 lines 72-74

Overloaded interface containing the following procedures: _calcanglesingle _calcangledouble

static calccross(**args*, ***kwargs*)
Defined at crystal.f90 lines 76-78

Overloaded interface containing the following procedures: _calccrosssingle _calccrossdouble

static calcdensity(*self*)
dens, avz, ava = calcdensity(self)

Defined at crystal.f90 lines 1452-1484

Parameters **cell** (*Unitcell*) –

Returns

- **dens** (*float*)
- **avz** (*float*)
- **ava** (*float*)

static calcdot(**args*, ***kwargs*)
Defined at crystal.f90 lines 60-62

Overloaded interface containing the following procedures: _calcdotsingle _calcdotdouble

static calclength(**args*, ***kwargs*)
Defined at crystal.f90 lines 68-70

Overloaded interface containing the following procedures: _calclengthsingle _calclengthdouble

static calcmatrices(*self*)
Defined at crystal.f90 lines 191-275

Parameters **cell** (*Unitcell*) –

static calcsgholz(*self*, *holzdata*, *gg*, *kt*, *lambda_*)
exer = calcsgholz(self, holzdata, gg, kt, **lambda_**)

Defined at crystal.f90 lines 1634-1661

Parameters

- **cell** (*Unitcell*) –

- **holzdata** (*Holzentries*) –
- **gg** (*float array*) –
- **kt** (*float array*) –
- **lambda** (*float*) –

Returns exer

Return type float

static computeor (*self*, *cella*, *cellb*, *direction*)
tt = computeor(self, cella, cellb, direction)

Defined at crystal.f90 lines 1557-1612

Parameters

- **orel** (*Orientation*) –
- **cella** (*Unitcell*) –
- **cellb** (*Unitcell*) –
- **direction** (*str*) –

Returns

- **tt** (*float array*)
- *==========NEW==========*

static convert_kgs_to_substrate (*self*, *cells*, *kg*, *ttinv*, *fn*)
kgs = convert_kgs_to_substrate(self, cells, kg, ttinv, fn)

Defined at crystal.f90 lines 1836-1868

Parameters

- **cell** (*Unitcell*) –
- **cells** (*Unitcell*) –
- **kg** (*float array*) –
- **ttinv** (*float array*) –
- **fn** (*float array*) –

Returns kgs

Return type float array

static displayelements ()
Defined at crystal.f90 lines 1267-1292

static extractposition (*list*, *pt*)
Defined at crystal.f90 lines 1315-1426

Parameters

- **list** (*str array*) –
- **pt** (*float array*) –

static getasympos (*self*)
Defined at crystal.f90 lines 1203-1249

Parameters **cell** (*Unitcell*) –

static getemsoftxtalsystem(*isg*)
xs = getemsoftxtalsystem(isg)

Defined at crystal.f90 lines 91-110

Parameters **isg** (*int*) –

Returns **xs**

Return type int

static getholzcoordinates(*self*, *holzdata*, *gg*, *kt*, *lambda_*)
pxy = getholzcoordinates(self, holzdata, gg, kt, **lambda_**)

Defined at crystal.f90 lines 1786-1814

Parameters

- **cell** (*Unitcell*) –
- **holzdata** (*Holzentries*) –
- **gg** (*float array*) –
- **kt** (*float array*) –
- **lambda** (*float*) –

Returns **pxy**

Return type float array

static getholzgeometry(*self*, *holzdata*, *g1*, *g2*, *uvw*, *fn*)
Defined at crystal.f90 lines 1684-1764

Parameters

- **cell** (*Unitcell*) –
- **holzdata** (*Holzentries*) –
- **g1** (*float array*) –
- **g2** (*float array*) –
- **uvw** (*int array*) –
- **fn** (*int array*) –

static getlatparm(*self*)
Defined at crystal.f90 lines 1066-1182

Parameters **cell** (*Unitcell*) –

static getor()
orel = getor()

Defined at crystal.f90 lines 1509-1537

Returns **orel**

Return type Orientation

static milbrav(*p*, *q*, *d*)
Defined at crystal.f90 lines 995-1036

Parameters

- **p** (*int array*) –

- **q** (*int array*) –
- **d** (*str*) –

static normvec (**args*, ***kwargs*)
Defined at crystal.f90 lines 64-66

Overloaded interface containing the following procedures: _normvecsingle _normvecdouble

static resetcell (*self*)
Defined at crystal.f90 lines 128-171

Parameters **cell** (*Unitcell*) –

static transcoor (*self*, *t*, *d*, *talpha*, *space*, *direction*)
Defined at crystal.f90 lines 458-486

Parameters

- **cell** (*Unitcell*) –
- **t** (*float array*) –
- **d** (*float array*) –
- **talpha** (*float array*) –
- **space** (*str*) –
- **direction** (*str*) –

static transspace (**args*, ***kwargs*)
Defined at crystal.f90 lines 56-58

Overloaded interface containing the following procedures: _transspacesingle _transspacedouble

class pyEMsoft.**Detectors** (**args*, ***kwargs*)
Module detectors

Defined at detectors.f90 lines 40-573

static ebsdfullgeneratedetector (*self*, *ebsddetector*, *numebins*, *numzbins*[, *verbose*])
Defined at detectors.f90 lines 490-573

Parameters

- **enl** (*Ebsdfullnamelisttype*) –
- **ebsddetector** (*Ebsddetectortype*) –
- **numebins** (*int*) –
- **numzbins** (*int*) –
- **verbose** (*bool*) –
- **==========** – —— generate the detector arrays
- **==========** – This needs to be done only once for a given detector geometry

static generatedefectebsddetector (*self*, *mcnl*, *nsx*, *nsy*, *tgx*, *tgy*, *tgz*, *patcntr*)
Defined at detectors.f90 lines 401-472

Parameters

- **enl** (*Ebsddefectnamelisttype*) –
- **mcnl** (*Mcclnamelisttype*) –

- **nsx** (*int*) –
- **nsy** (*int*) –
- **tgx** (*float array*) –
- **tgy** (*float array*) –
- **tgz** (*float array*) –
- **patcntr** (*float array*) –
- **==========** – —— generate the detector arrays
- **==========** –

static generateebsddetector (*self*, *mcnl*, *ebsdmcdata*, *ebsddetector*[, *verbose*])
Defined at detectors.f90 lines 65-209

Parameters

- **enl** (*Ebsdnamelisttype*) –
- **mcnl** (*Mcclnamelisttype*) –
- **ebsdmcdata** (*Ebsdmcdatatype*) –
- **ebsddetector** (*Ebsddetectortype*) –
- **verbose** (*bool*) –
- **==========** – —— generate the detector arrays
- **==========** – This needs to be done only once for a given detector geometry

static generatemyebsddetector (*self*, *mcnl*, *ebsdmcdata*, *nsx*, *nsy*, *nume*, *tgx*, *tgy*, *tgz*, *accum_e_detector*, *patcntr*[, *bg*])
Defined at detectors.f90 lines 228-387

Parameters

- **enl** (*Ebsdnamelisttype*) –
- **mcnl** (*Mcclnamelisttype*) –
- **ebsdmcdata** (*Ebsdmcdatatype*) –
- **nsx** (*int*) –
- **nsy** (*int*) –
- **nume** (*int*) –
- **tgx** (*float array*) –
- **tgy** (*float array*) –
- **tgz** (*float array*) –
- **accum_e_detector** (*float array*) –
- **patcntr** (*float array*) –
- **bg** (*bool*) –
- **==========** – —— generate the detector arrays
- **==========** –

class pyEMsoft.**Dictmod**(**args*, ***kwargs*)
Module dictmod

Defined at dictmod.f90 lines 107-1700

static dd_density(*x*, *nums*, *mu*, *kappa*, *c*, *dtype*)
y = dd_density(x, nums, mu, kappa, c, dtype)

Defined at dictmod.f90 lines 970-991

Parameters

- **x** (*float array*) –
- **nums** (*int*) –
- **mu** (*float array*) –
- **kappa** (*float*) –
- **c** (*float*) –
- **dtype** (*str*) –

Returns y

Return type float array

static di_emfordd(*x*, *dict*, *nums*, *seed*, *muhat*, *dtype*)
kappahat = di_emfordd(x, dict, nums, seed, muhat, dtype)

Defined at dictmod.f90 lines 629-732

Parameters

- **x** (*float array*) –
- **dict** (*Dicttype*) –
- **nums** (*int*) –
- **seed** (*int*) –
- **muhat** (*float array*) –
- **dtype** (*str*) –

Returns kappahat

Return type float

static di_init(*self*, *dtype*)
Defined at dictmod.f90 lines 464-605

Parameters

- **dict** (*Dicttype*) –
- **dtype** (*str*) –
- **----------** – IMPORTANT NOTE: the original von Mises-Fischer (VMF) approach requires that q and -q are considered to be separate quaternions, so the original Matlab code included the negatives of all quaternion symmetry operators as well, leading to a cardinality of twice the rotational point group order. It appears that we do not have to do so if we replace the exponential in the VMF by a hyperbolic cosine function, which would account directly for the q, -q duplicity... Alternatively, one can use the axial Watson distribution.

- **------------** – identity operator is part of all point groups

static di_sampledd(*n*, *seed*, *mu*, *kappa*, *dtype*)
sdd = di_sampledd(n, seed, mu, kappa, dtype)

Defined at dictmod.f90 lines 221-252

Parameters

- **n** (*int*) –
- **seed** (*int*) –
- **mu** (*float array*) –
- **kappa** (*float*) –
- **dtype** (*str*) –

Returns **sdd**

Return type float array

static di_similarity_classifier(*array*, *k*, *npx*, *npy*, *returnarr*)
Defined at dictmod.f90 lines 1073-1102

Parameters

- **array** (*int array*) –
- **k** (*int*) –
- **npx** (*int*) –
- **npy** (*int*) –
- **returnarr** (*float array*) –

static getaveragedisorientationmap(**args*, ***kwargs*)
Defined at dictmod.f90 lines 118-120

Overloaded interface containing the following procedures: _getaveragedisorientationmapsingle _getaveragedisorientationmapdouble

static getdisorientationangle(**args*, ***kwargs*)
Defined at dictmod.f90 lines 114-116

Overloaded interface containing the following procedures: _getdisorientationanglesingle _getdisorientationangledouble

static getdisorientationangleaxis(*eu1*, *eu2*, *dict*, *disax*)
Defined at dictmod.f90 lines 1459-1504

Parameters

- **eu1** (*float array*) –
- **eu2** (*float array*) –
- **dict** (*Dicttype*) –
- **disax** (*float array*) –

static getdisorientationangleaxistwophases(*eu1*, *eu2*, *dict1*, *dict2*, *disax*)
Defined at dictmod.f90 lines 1522-1562

Parameters

- **eu1** (*float array*) –

- **eu2** (*float array*) –
- **dict1** (*Dicttype*) –
- **dict2** (*Dicttype*) –
- **disax** (*float array*) –

static reducedisorientationtomfz(*ro*, *cell*, *fztype*, *fzorder*, *romfz*)
Defined at dictmod.f90 lines 1163-1203

Parameters

- **ro** (*float array*) –
- **cell** (*Unitcell*) –
- **fztype** (*int*) –
- **fzorder** (*int*) –
- **romfz** (*float array*) –

static reduceorientationtocubicefz(*eu*, *dict*, *eufz*)
Defined at dictmod.f90 lines 1219-1247

Parameters

- **eu** (*float array*) –
- **dict** (*Dicttype*) –
- **eufz** (*float array*) –

static reduceorientationtorfz(*eu*, *dict*, *fztype*, *fzorder*, *eufz*[, *mfz*])
Defined at dictmod.f90 lines 1268-1310

Parameters

- **eu** (*float array*) –
- **dict** (*Dicttype*) –
- **fztype** (*int*) –
- **fzorder** (*int*) –
- **eufz** (*float array*) –
- **mfz** (*bool*) –

class pyEMsoft.**Diffraction**(**args*, ***kwargs*)
Module diffraction

Defined at diffraction.f90 lines 47-1891

static bwsolve(*m*, *w*, *cgg*, *cginv*, *nn*, *ipiv*)
Defined at diffraction.f90 lines 1672-1737

Parameters

- **m** (*complex array*) –
- **w** (*complex array*) –
- **cgg** (*complex array*) –
- **cginv** (*complex array*) –
- **nn** (*int*) –

- **ipiv** (*int array*) –

static calcdiffangle (*self*, *h*, *k*, *l*)
tt = calcdiffangle(self, h, k, l)

Defined at diffraction.f90 lines 300-307

Parameters

- **cell** (*Unitcell*) –
- **h** (*int*) –
- **k** (*int*) –
- **l** (*int*) –

Returns tt

Return type float

static calcfresnelpropagator (*beam*, *dimi*, *dimj*, *dz*, *scl*, *propname*, *lambda_*)
Defined at diffraction.f90 lines 1843-1891

Parameters

- **beam** (*float array*) –
- **dimi** (*int*) –
- **dimj** (*int*) –
- **dz** (*float*) –
- **scl** (*float*) –
- **propname** (*str*) –
- **lambda** (*float*) –

static calcsg (**args*, ***kwargs*)
Defined at diffraction.f90 lines 159-161

Overloaded interface containing the following procedures: _calcsgsingle _calcsgdouble

static calcucg (*self*, *rlp*, *hkl*[, *applyqgshift*, *interpolate*])
Defined at diffraction.f90 lines 367-656

Parameters

- **cell** (*Unitcell*) –
- **rlp** (*Gnode*) –
- **hkl** (*int array*) –
- **applyqgshift** (*bool*) –
- **interpolate** (*bool*) –
- **----------** – first the simplest case: kinematical X-ray scattering factors this option was added to accomodate the HEDM forward projector needs

static calcwavelength (*self*, *rlp*[, *skip*, *verbose*])
Defined at diffraction.f90 lines 218-281

Parameters

- **cell** (*Unitcell*) –
- **rlp** (*Gnode*) –
- **skip** (*int*) –
- **verbose** (*bool*) –

static diffpage (*self*, *cell*, *rlp*, *camlen*)
Defined at diffraction.f90 lines 1013-1287

Parameters

- **ps** (*Postscript_Type*) –
- **cell** (*Unitcell*) –
- **rlp** (*Gnode*) –
- **camlen** (*float*) –

static dumppp (*self*, *cell*, *xo*, *yo*, *np*, *lal*, *icnt*, *vgsave*, *rg*, *rnumfam*)
Defined at diffraction.f90 lines 1471-1536

Parameters

- **ps** (*Postscript_Type*) –
- **cell** (*Unitcell*) –
- **xo** (*float*) –
- **yo** (*float*) –
- **np** (*bool*) –
- **lal** (*float*) –
- **icnt** (*int*) –
- **vgsave** (*float array*) –
- **rg** (*float array*) –
- **rnumfam** (*int array*) –

static dumpzap (*self*, *cell*, *xo*, *yo*, *u*, *v*, *w*, *p*, *np*, *first*, *indi*, *lal*, *icnt*, *dbdiff*, *vg*, *vgsave*, *rg*, *rfamily*, *rnumfam*, *hhcc*)
Defined at diffraction.f90 lines 1318-1447

Parameters

- **ps** (*Postscript_Type*) –
- **cell** (*Unitcell*) –
- **xo** (*float*) –
- **yo** (*float*) –
- **u** (*int*) –
- **v** (*int*) –
- **w** (*int*) –
- **p** (*int*) –
- **np** (*bool*) –
- **first** (*bool*) –

- **indi** (*int*) –
- **lal** (*float*) –
- **icnt** (*int*) –
- **dbdiff** (*bool array*) –
- **vg** (*float array*) –
- **vgsave** (*float array*) –
- **rg** (*float array*) –
- **rfamily** (*int array*) –
- **rnumfam** (*int array*) –
- **hhcc** (*int*) –

static getscatfac (*self*, *s*, *sfarray*, *ntypes*)
Defined at diffraction.f90 lines 736-760

Parameters

- **cell** (*Unitcell*) –
- **s** (*float*) –
- **sfarray** (*complex array*) –
- **ntypes** (*int*) –

static getvoltage (*self*, *rlp*[, *verbose*])
Defined at diffraction.f90 lines 178-194

Parameters

- **cell** (*Unitcell*) –
- **rlp** (*Gnode*) –
- **verbose** (*bool*) –

static lorentzpf (*theta*, *hedm=None*)
tt = lorentzpf(theta[, hedm])

Defined at diffraction.f90 lines 322-333

Parameters

- **theta** (*float*) –
- **hedm** (*str*) –

Returns **tt**

Return type float

static precalcfscatt (*self*, *dmin*, *gstep*)
Defined at diffraction.f90 lines 672-720

Parameters

- **cell** (*Unitcell*) –
- **dmin** (*float*) –
- **gstep** (*float*) –

static studylist(*list*, *slect*, *ppat*)
np = studylist(list, slect, ppat)

Defined at diffraction.f90 lines 1559-1638

Parameters

- **list** (*str array*) –
- **slect** (*int array*) –
- **ppat** (*bool*) –

Returns **np**

Return type int

static tbcalcinten(*sg*, *z*, *xig*, *xigp*, *xizero*, *betag*)
it, is_ = tbcalcinten(sg, z, xig, xigp, xizero, betag)

Defined at diffraction.f90 lines 951-992

Parameters

- **sg** (*float*) –
- **z** (*float*) –
- **xig** (*float*) –
- **xigp** (*float*) –
- **xizero** (*float*) –
- **betag** (*float*) –

Returns

- **it** (*float*)
- **is_** (*float*)

static tbcalcsm(*ar*, *ai*, *sg*, *z*, *xig*, *xigp*, *xizero*, *betag*)
Defined at diffraction.f90 lines 864-921

Parameters

- **ar** (*float array*) –
- **ai** (*float array*) –
- **sg** (*float*) –
- **z** (*float*) –
- **xig** (*float*) –
- **xigp** (*float*) –
- **xizero** (*float*) –
- **betag** (*float*) –

class pyEMsoft.**Ebsdmod**(**args*, ***kwargs*)
Module ebsdmod

Defined at EBSDmod.f90 lines 44-1430

static calcebsdpatterndefect(**args*, ***kwargs*)
Defined at EBSDmod.f90 lines 72-74

Overloaded interface containing the following procedures: _calcebsdpatterndefect_zint _calcebsdpatterndefect_noint

static calcebsdpatternsinglefull(*ipar*, *qu*, *accum*, *mlpnh*, *mlpsh*, *rgx*, *rgy*, *rgz*, *binned*, *emin*, *emax*, *mask*, *prefactor*[, *fmatrix*, *removebackground*, *applynoise*])

Defined at EBSDmod.f90 lines 899-1015

Parameters

- **ipar** (*int array*) –
- **qu** (*float array*) –
- **accum** (*float array*) –
- **mlpnh** (*float array*) –
- **mlpsh** (*float array*) –
- **rgx** (*float array*) –
- **rgy** (*float array*) –
- **rgz** (*float array*) –
- **binned** (*float array*) –
- **emin** (*int*) –
- **emax** (*int*) –
- **mask** (*float array*) –
- **prefactor** (*float*) –
- **fmatrix** (*float array*) –
- **removebackground** (*str*) –
- **applynoise** (*int*) –

static calcebsdpatternsinglefullfast(*ipar*, *qu*, *accum*, *mlpnh*, *mlpsh*, *rgx*, *rgy*, *rgz*, *binned*, *emin*, *emax*, *prefactor*)

Defined at EBSDmod.f90 lines 1182-1242

Parameters

- **ipar** (*int array*) –
- **qu** (*float array*) –
- **accum** (*float array*) –
- **mlpnh** (*float array*) –
- **mlpsh** (*float array*) –
- **rgx** (*float array*) –
- **rgy** (*float array*) –
- **rgz** (*float array*) –
- **binned** (*float array*) –
- **emin** (*int*) –
- **emax** (*int*) –

- **prefactor** (*float*) –

static ebsdcopymcdata (*inputfile*, *outputfile*, *h5*)
Defined at EBSDmod.f90 lines 1257-1332

Parameters

- **inputfile** (*str*) –
- **outputfile** (*str*) –
- **h5** (*str*) –

static ebsdcopympdata (*inputfile*, *outputfile*, *h5*[, *skipcrystaldata*])
Defined at EBSDmod.f90 lines 1348-1430

Parameters

- **inputfile** (*str*) –
- **outputfile** (*str*) –
- **h5** (*str*) –
- **skipcrystaldata** (*bool*) –

static ebsdfullreadangles (*self*, *angles*, *verbose=None*)
numangles = ebsdfullreadangles(self, angles[, verbose])

Defined at EBSDmod.f90 lines 179-244

Parameters

- **enl** (*Ebsdfullnamelisttype*) –
- **angles** (*Ebsdangletype*) –
- **verbose** (*bool*) –

Returns

- **numangles** (*int*)
- **==========** – get the angular information, either in Euler angles or in quaternions, from a text file
- **==========** – open the angle file

static ebsdreadangles (*self*, *angles*, *verbose=None*)
numangles = ebsdreadangles(self, angles[, verbose])

Defined at EBSDmod.f90 lines 90-164

Parameters

- **enl** (*Ebsdnamelisttype*) –
- **angles** (*Ebsdangletype*) –
- **verbose** (*bool*) –

Returns

- **numangles** (*int*)
- **==========** – get the angular information, either in Euler angles or in quaternions, from a text file
- **==========** – open the angle file

static ebsdreadorpcdef(*self*, *orpcdef*, *verbose=None*)
numangles = ebsdreadorpcdef(self, orpcdef[, verbose])

Defined at EBSDmod.f90 lines 259-322

Parameters

- **enl** (*Ebsdnamelisttype*) –
- **orpcdef** (*Ebsdanglepcdeftype*) –
- **verbose** (*bool*) –

Returns

- **numangles** (*int*)
- ========== – get the angular information, either in Euler angles or in quaternions, from a text file
- ========== – open the angle file

static readebsdmasterpatternfile(*mpfile*, *mpnl*, *ebsdmpdata*, *getkevs=None*, *getmlpnh=None*, *getmlpsh=None*, *getmasterspnh=None*, *getmasterspsh=None*, *keep4=None*, *defectmp=None*, *verbose=None*)
hdferr = readebsdmasterpatternfile(mpfile, mpnl, ebsdmpdata[, getkevs, getmlpnh, getmlpsh, getmasterspnh, getmasterspsh, keep4, defectmp, verbose])

Defined at EBSDmod.f90 lines 598-880

Parameters

- **mpfile** (*str*) –
- **mpnl** (*Ebsdmasternamelisttype*) –
- **ebsdmpdata** (*Ebsdmpdatatype*) –
- **getkevs** (*bool*) –
- **getmlpnh** (*bool*) –
- **getmlpsh** (*bool*) –
- **getmasterspnh** (*bool*) –
- **getmasterspsh** (*bool*) –
- **keep4** (*bool*) –
- **defectmp** (*bool*) –
- **verbose** (*bool*) –

Returns

- **hdferr** (*int*)
- ========== – make sure this is a Master Pattern file
- ==========

static readebsdmontecarlofile(*mcfile*, *mcnl*, *ebsdmcdata*, *getaccume=None*, *getaccumz=None*, *getaccumsp=None*, *getaccumxyz=None*, *verbose=None*)

hdferr = readebsdmontecarlofile(mcfile, mcnl, ebsdmcdata[, getaccume, getaccumz, getaccumsp, getaccumxyz, verbose])

Defined at EBSDmod.f90 lines 340-579

Parameters

- **mcfile** (*str*) –
- **mcnl** (*Mcclnamelisttype*) –
- **ebsdmcdata** (*Ebsdmcdatatype*) –
- **getaccume** (*bool*) –
- **getaccumz** (*bool*) –
- **getaccumsp** (*bool*) –
- **getaccumxyz** (*bool*) –
- **verbose** (*bool*) –

Returns

- **hdferr** (*int*)
- **====================================** – make sure this is a Monte Carlo file
- **====================================**

class pyEMsoft.**Error**(**args*, ***kwargs*)

Module error

Defined at error.f90 lines 46-76

static fatalerror(*var1*, *var2*)

Defined at error.f90 lines 68-75

Parameters

- **var1** (*str*) –
- **var2** (*str*) –

class pyEMsoft.**Files**(**args*, ***kwargs*)

Module files

Defined at files.f90 lines 49-568

static convertwiki2pdf(*nt*, *wikilist*)

Defined at files.f90 lines 150-234

Parameters

- **nt** (*int*) –
- **wikilist** (*int array*) –

static copytemplatefiles(*nt*, *templatelist*[, *json*])

Defined at files.f90 lines 260-361

Parameters

- **nt** (*int*) –

- **templatelist** (*int array*) –
- **json** (*bool*) –

static dumpxtalinfo(*self*)
Defined at files.f90 lines 73-135

Parameters **cell** (*Unitcell*) –

static interpret_program_arguments(**args*, ***kwargs*)
Defined at files.f90 lines 52-54

Overloaded interface containing the following procedures: _interpret_program_arguments_with_nml _interpret_program_arguments_no_nml

class pyEMsoft.**Filters**(**args*, ***kwargs*)
Module filters

Defined at filters.f90 lines 44-1069

static adhisteq(*nr*, *dimx*, *dimy*, *im*, *verbose=None*)
output = adhisteq(nr, dimx, dimy, im[, verbose])

Defined at filters.f90 lines 389-492

Parameters

- **nr** (*int*) –
- **dimx** (*int*) –
- **dimy** (*int*) –
- **im** (*int array*) –
- **verbose** (*bool*) –

Returns **output**

Return type int array

static applygaussianbeamspread(*ipar*, *fpar*, *vxyz*, *w*[, *verbose*])
Defined at filters.f90 lines 63-125

Parameters

- **ipar** (*int array*) –
- **fpar** (*float array*) –
- **vxyz** (*int array*) –
- **w** (*float*) –
- **verbose** (*bool*) –

static applypoissonnoise(*image*, *nx*, *ny*, *idum*)
noisy = applypoissonnoise(image, nx, ny, idum)

Defined at filters.f90 lines 141-158

Parameters

- **image** (*float array*) –
- **nx** (*int*) –
- **ny** (*int*) –

- **idum** (*int*) –

Returns noisy

Return type float array

static butterflymask9x9(*input*, *output*, *dims*)
Defined at filters.f90 lines 1004-1030

Parameters

- **input** (*float array*) –
- **output** (*float array*) –
- **dims** (*int*) –

static calchoughlut(*dimx*, *lut*)
Defined at filters.f90 lines 658-693

Parameters

- **dimx** (*int*) –
- **lut** (*float array*) –

static cumul_histogram(*nx*, *ny*, *im*)
h = cumul_histogram(nx, ny, im)

Defined at filters.f90 lines 238-275

Parameters

- **nx** (*int*) –
- **ny** (*int*) –
- **im** (*int array*) –

Returns h

Return type int array

static getadpmap(*iunit*, *nexpt*, *l*, *wd*, *ht*, *dpmap*)
Defined at filters.f90 lines 511-567

Parameters

- **iunit** (*int*) –
- **nexpt** (*int*) –
- **l** (*int*) –
- **wd** (*int*) –
- **ht** (*int*) –
- **dpmap** (*float array*) –

static getadpmapram(*epatterns*, *nexpt*, *cs*, *l*, *wd*, *ht*, *dpmap*)
Defined at filters.f90 lines 586-643

Parameters

- **epatterns** (*float array*) –
- **nexpt** (*int*) –
- **cs** (*int*) –

- **l** (*int*) –
- **wd** (*int*) –
- **ht** (*int*) –
- **dpmap** (*float array*) –

static houghtransform(*dimx*, *lut*, *im*, *ht*)
Defined at filters.f90 lines 710-726

Parameters

- **dimx** (*int*) –
- **lut** (*float array*) –
- **im** (*float array*) –
- **ht** (*float array*) –

static image_entropy(*h*)
e = image_entropy(h)

Defined at filters.f90 lines 289-305

Parameters **h** (*int array*) –

Returns **e**

Return type float

static image_histogram(*nx*, *ny*, *im*)
h = image_histogram(nx, ny, im)

Defined at filters.f90 lines 174-189

Parameters

- **nx** (*int*) –
- **ny** (*int*) –
- **im** (*int array*) –

Returns **h**

Return type int array

static image_jointentropy(*h*)
e = image_jointentropy(h)

Defined at filters.f90 lines 319-337

Parameters **h** (*int array*) –

Returns **e**

Return type float

static image_jointhistogram(*nx*, *ny*, *im1*, *im2*)
h = image_jointhistogram(nx, ny, im1, im2)

Defined at filters.f90 lines 206-222

Parameters

- **nx** (*int*) –
- **ny** (*int*) –

- **im1** (*int array*) –
- **im2** (*int array*) –

Returns h

Return type int array

static image_mutualinformation(*nx*, *ny*, *im1*, *im2*)

mi = image_mutualinformation(nx, ny, im1, im2)

Defined at filters.f90 lines 353-367

Parameters

- **nx** (*int*) –
- **ny** (*int*) –
- **im1** (*int array*) –
- **im2** (*int array*) –

Returns mi

Return type float

static inversiondivision(*input*, *output*, *dims*)

Defined at filters.f90 lines 1046-1069

Parameters

- **input** (*float array*) –
- **output** (*float array*) –
- **dims** (*int*) –

class pyEMsoft.**Hdfsupport**(**args*, ***kwargs*)

Module hdfsupport

Defined at HDFsupport.f90 lines 66-5773

static checkfixedlengthflag(*dataset*, *hdf_head*)

itis = checkfixedlengthflag(dataset, hdf_head)

Defined at HDFsupport.f90 lines 5397-5419

Parameters

- **dataset** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns itis

Return type bool

static crystaldata(*self*[, *verbose*, *existinghdfhead*])

Defined at HDFsupport.f90 lines 5094-5134

Parameters

- **cell** (*Unitcell*) –
- **verbose** (*bool*) –
- **existinghdfhead** (*Hdfobjectstacktype*) –

static cstringify (*strin*)
cstrout = cstringify(strin)

Defined at HDFsupport.f90 lines 173-186

Parameters **strin** (*str*) –

Returns **cstrout**

Return type str

static fstringify (*strin*)
fstrout = fstringify(strin)

Defined at HDFsupport.f90 lines 229-240

Parameters **strin** (*str array*) –

Returns **fstrout**

Return type str

static h5_read_integer_dataset (*fname*, *dsetnm*)
hdferr, rdata = h5_read_integer_dataset(fname, dsetnm)

Defined at HDFsupport.f90 lines 5565-5593

Parameters

- **fname** (*str*) –
- **dsetnm** (*str*) –

Returns

- **hdferr** (*int*)
- **rdata** (*int*)

static h5_tsl_read_ebsd_pattern (*fname*, *dsetnm*, *rdata*, *offset*, *szx*, *szy*)
hdferr = h5_tsl_read_ebsd_pattern(fname, dsetnm, rdata, offset, szx, szy)

Defined at HDFsupport.f90 lines 5504-5550

Parameters

- **fname** (*str*) –
- **dsetnm** (*str*) –
- **rdata** (*int array*) –
- **offset** (*int*) –
- **szx** (*int*) –
- **szy** (*int*) –

Returns **hdferr**

Return type int

static h5_write_pseudo_bse_image (*fname*, *dsetnm*, *wdata*)
hdferr = h5_write_pseudo_bse_image(fname, dsetnm, wdata)

Defined at HDFsupport.f90 lines 5447-5485

Parameters

- **fname** (*str*) –

- **dsetnm** (*str*) –
- **wdata** (*float array*) –

Returns hdferr

Return type int

static h5close_emsoft (*hdferr*)
Defined at HDFsupport.f90 lines 146-158

Parameters **hdferr** (*int*) –

static h5open_emsoft (*hdferr*)
Defined at HDFsupport.f90 lines 118-132

Parameters **hdferr** (*int*) –

static hdf_addstringattributetogroup (*dataname*, *stratt*, *hdf_head*, *overwrite=None*)
success = hdf_addstringattributetogroup(dataname, stratt, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 5611-5670

Parameters

- **dataname** (*str*) –
- **stratt** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_createfile (*hdfname*, *hdf_head*)
success = hdf_createfile(hdfname, hdf_head)

Defined at HDFsupport.f90 lines 689-709

Parameters

- **hdfname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_creategroup (*groupname*, *hdf_head*)
success = hdf_creategroup(groupname, hdf_head)

Defined at HDFsupport.f90 lines 765-797

Parameters

- **groupname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_extractdatasettextfile(*dataname*, *textfile*, *hdf_head*)
success = hdf_extractdatasettextfile(dataname, textfile, hdf_head)

Defined at HDFsupport.f90 lines 1129-1190

Parameters

- **dataname** (*str*) –
- **textfile** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_getstringattributefromgroup(*dataname*, *stratt*, *slen_bn*, *hdf_head*)
success = hdf_getstringattributefromgroup(dataname, stratt, slen_bn, hdf_head)

Defined at HDFsupport.f90 lines 5688-5743

Parameters

- **dataname** (*str*) –
- **stratt** (*str*) –
- **slen_bn** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_handleerror(*error*, *offendingroutine*[, *nonfatal*])
Defined at HDFsupport.f90 lines 653-667

Parameters

- **error** (*int*) –
- **offendingroutine** (*str*) –
- **nonfatal** (*bool*) –

static hdf_opendataset(*dataname*, *hdf_head*)
success = hdf_opendataset(dataname, hdf_head)

Defined at HDFsupport.f90 lines 845-863

Parameters

- **dataname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_openfile(*hdfname*, *hdf_head*, *readonly=None*)
success = hdf_openfile(hdfname, hdf_head[, readonly])

Defined at HDFsupport.f90 lines 725-750

Parameters

- **hdfname** (*str*) –

- **hdf_head** (*Hdfobjectstacktype*) –
- **readonly** (*bool*) –

Returns success

Return type int

static hdf_opengroup (*groupname*, *hdf_head*)
success = hdf_opengroup(groupname, hdf_head)

Defined at HDFsupport.f90 lines 812-830

Parameters

- **groupname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_pop (*self*[, *closeall*, *verbose*])
Defined at HDFsupport.f90 lines 506-569

Parameters

- **hdf_head** (*Hdfobjectstacktype*) –
- **closeall** (*bool*) –
- **verbose** (*bool*) –

static hdf_read2dimage (*dataset*, *image*, *numx*, *numy*, *hdf_head*)
Defined at HDFsupport.f90 lines 5755-5773

Parameters

- **dataset** (*str*) –
- **image** (*int array*) –
- **numx** (*int*) –
- **numy** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –

static hdf_readdatasetdouble (*dataname*, *hdf_head*)
hdferr, rdata = hdf_readdatasetdouble(dataname, hdf_head)

Defined at HDFsupport.f90 lines 3541-3571

Parameters

- **dataname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns

- **hdferr** (*int*)
- **rdata** (*float*)

static hdf_readdatasetfloat (*dataname*, *hdf_head*)
hdferr, rdata = hdf_readdatasetfloat(dataname, hdf_head)

Defined at HDFsupport.f90 lines 3274-3304

Parameters

- **dataname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns

- **hdferr** (*int*)
- **rdata** (*float*)

static hdf_readdatasetinteger(*dataname*, *hdf_head*)
hdferr, rdata = hdf_readdatasetinteger(dataname, hdf_head)

Defined at HDFsupport.f90 lines 3010-3040

Parameters

- **dataname** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns

- **hdferr** (*int*)
- **rdata** (*int*)

static hdf_stackdump(*self*)
Defined at HDFsupport.f90 lines 583-603

Parameters **hdf_head** (*Hdfobjectstacktype*) –

static hdf_writedatasetchararray1d(*dataname*, *chararray*, *dims*, *hdf_head*, *overwrite=None*)
success = hdf_writedatasetchararray1d(dataname, chararray, dims, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1305-1352

Parameters

- **dataname** (*str*) –
- **chararray** (*str array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns **success**

Return type int

static hdf_writedatasetchararray2d(*dataname*, *chararray*, *dims*, *hdf_head*, *overwrite=None*)
success = hdf_writedatasetchararray2d(dataname, chararray, dims, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1370-1416

Parameters

- **dataname** (*str*) –
- **chararray** (*str array*) –
- **dims** (*int array*) –

- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetchararray3d(*dataname*, *chararray*, *dims*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetchararray3d(dataname, chararray, dims, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1434-1480

Parameters

- **dataname** (*str*) –
- **chararray** (*str array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetchararray4d(*dataname*, *chararray*, *dims*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetchararray4d(dataname, chararray, dims, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1498-1544

Parameters

- **dataname** (*str*) –
- **chararray** (*str array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetdouble(*dataname*, *dblval*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetdouble(dataname, dblval, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2058-2109

Parameters

- **dataname** (*str*) –
- **dblval** (*float*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetdoublearray1d(*dataname*, *dblarr*, *dim0*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetdoublearray1d(dataname, dblarr, dim0, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2498-2551

Parameters

- **dataname** (*str*) –
- **dblarr** (*float array*) –
- **dim0** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetdoublearray2d(*dataname*, *dblarr*, *dim0*, *dim1*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetdoublearray2d(dataname, dblarr, dim0, dim1, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2569-2623

Parameters

- **dataname** (*str*) –
- **dblarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetdoublearray3d(*dataname*, *dblarr*, *dim0*, *dim1*, *dim2*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetdoublearray3d(dataname, dblarr, dim0, dim1, dim2, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2641-2696

Parameters

- **dataname** (*str*) –
- **dblarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **dim2** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetdoublearray4d(*dataname*, *dblarr*, *dim0*, *dim1*, *dim2*, *dim3*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetdoublearray4d(dataname, dblarr, dim0, dim1, dim2, dim3, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2714-2770

Parameters

- **dataname** (*str*) –
- **dblarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **dim2** (*int*) –
- **dim3** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetfloat(*dataname*, *fltval*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetfloat(dataname, fltval, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1989-2040

Parameters

- **dataname** (*str*) –
- **fltval** (*float*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetfloatarray1d(*dataname*, *fltarr*, *dim0*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetfloatarray1d(dataname, fltarr, dim0, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2127-2181

Parameters

- **dataname** (*str*) –
- **fltarr** (*float array*) –
- **dim0** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetfloatarray2d(*dataname*, *fltarr*, *dim0*, *dim1*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetfloatarray2d(dataname, fltarr, dim0, dim1, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2199-2254

Parameters

- **dataname** (*str*) –
- **fltarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetfloatarray3d(*dataname*, *fltarr*, *dim0*, *dim1*, *dim2*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetfloatarray3d(dataname, fltarr, dim0, dim1, dim2, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2272-2328

Parameters

- **dataname** (*str*) –
- **fltarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **dim2** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetfloatarray4d(*dataname*, *fltarr*, *dim0*, *dim1*, *dim2*, *dim3*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetfloatarray4d(dataname, fltarr, dim0, dim1, dim2, dim3, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2346-2403

Parameters

- **dataname** (*str*) –
- **fltarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **dim2** (*int*) –
- **dim3** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –

- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetfloatarray6d(*dataname*, *fltarr*, *dim0*, *dim1*, *dim2*, *dim3*, *dim4*, *dim5*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetfloatarray6d(dataname, fltarr, dim0, dim1, dim2, dim3, dim4, dim5, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 2422-2480

Parameters

- **dataname** (*str*) –
- **fltarr** (*float array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **dim2** (*int*) –
- **dim3** (*int*) –
- **dim4** (*int*) –
- **dim5** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetinteger(*dataname*, *intval*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetinteger(dataname, intval, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1562-1612

Parameters

- **dataname** (*str*) –
- **intval** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetinteger1bytearray1d(*dataname*, *intarr*, *dim0*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetinteger1bytearray1d(dataname, intarr, dim0, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1630-1681

Parameters

- **dataname** (*str*) –
- **intarr** (*int array*) –

- **dim0** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetintegerarray1d(*dataname*, *intarr*, *dim0*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetintegerarray1d(dataname, intarr, dim0, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1699-1752

Parameters

- **dataname** (*str*) –
- **intarr** (*int array*) –
- **dim0** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetintegerarray2d(*dataname*, *intarr*, *dim0*, *dim1*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetintegerarray2d(dataname, intarr, dim0, dim1, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1770-1824

Parameters

- **dataname** (*str*) –
- **intarr** (*int array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetintegerarray3d(*dataname*, *intarr*, *dim0*, *dim1*, *dim2*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetintegerarray3d(dataname, intarr, dim0, dim1, dim2, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1842-1897

Parameters

- **dataname** (*str*) –
- **intarr** (*int array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –

- **dim2** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetintegerarray4d(*dataname*, *intarr*, *dim0*, *dim1*, *dim2*, *dim3*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetintegerarray4d(dataname, intarr, dim0, dim1, dim2, dim3, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1915-1971

Parameters

- **dataname** (*str*) –
- **intarr** (*int array*) –
- **dim0** (*int*) –
- **dim1** (*int*) –
- **dim2** (*int*) –
- **dim3** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasetstringarray(*dataname*, *inputarray*, *nlines*, *hdf_head*, *overwrite=None*)

success = hdf_writedatasetstringarray(dataname, inputarray, nlines, hdf_head[, overwrite])

Defined at HDFsupport.f90 lines 1209-1287

Parameters

- **dataname** (*str*) –
- **inputarray** (*str array*) –
- **nlines** (*int*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **overwrite** (*bool*) –

Returns success

Return type int

static hdf_writedatasettextfile(*dataname*, *filename*, *hdf_head*)

success = hdf_writedatasettextfile(dataname, filename, hdf_head)

Defined at HDFsupport.f90 lines 881-954

Parameters

- **dataname** (*str*) –

- **filename** (*str*) –
- **hdf_head** (*Hdfobjectstacktype*) –

Returns success

Return type int

static hdf_writeemheader(*self*, *dstr*, *tstrb*, *tstre*, *prn*[, *dataname*])
Defined at HDFsupport.f90 lines 279-429

Parameters

- **hdf_head** (*Hdfobjectstacktype*) –
- **dstr** (*str*) –
- **tstrb** (*str*) –
- **tstre** (*str*) –
- **prn** (*str*) –
- **dataname** (*str*) –

static hdf_writehyperslabchararray2d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)
success = hdf_writehyperslabchararray2d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 3822-3857

Parameters

- **dataname** (*str*) –
- **wdata** (*str array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabchararray3d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)
success = hdf_writehyperslabchararray3d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 3881-3914

Parameters

- **dataname** (*str*) –
- **wdata** (*str array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –

- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabchararray4d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabchararray4d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 3938-3971

Parameters

- **dataname** (*str*) –
- **wdata** (*str array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabdoublearray2d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabdoublearray2d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4334-4367

Parameters

- **dataname** (*str*) –
- **wdata** (*float array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabdoublearray3d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabdoublearray3d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4391-4424

Parameters

- **dataname** (*str*) –
- **wdata** (*float array*) –

- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabdoublearray4d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabdoublearray4d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4448-4481

Parameters

- **dataname** (*str*) –
- **wdata** (*float array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabfloatarray2d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabfloatarray2d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4163-4196

Parameters

- **dataname** (*str*) –
- **wdata** (*float array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabfloatarray3d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabfloatarray3d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4220-4253

Parameters

- **dataname** (*str*) –
- **wdata** (*float array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabfloatarray4d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabfloatarray4d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4277-4310

Parameters

- **dataname** (*str*) –
- **wdata** (*float array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabintegerarray2d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabintegerarray2d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 3995-4027

Parameters

- **dataname** (*str*) –
- **wdata** (*int array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabintegerarray3d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabintegerarray3d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4051-4083

Parameters

- **dataname** (*str*) –
- **wdata** (*int array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdf_writehyperslabintegerarray4d(*dataname*, *wdata*, *hdims*, *offset*, *dims*, *hdf_head*, *insert=None*)

success = hdf_writehyperslabintegerarray4d(dataname, wdata, hdims, offset, dims, hdf_head[, insert])

Defined at HDFsupport.f90 lines 4107-4139

Parameters

- **dataname** (*str*) –
- **wdata** (*int array*) –
- **hdims** (*int array*) –
- **offset** (*int array*) –
- **dims** (*int array*) –
- **hdf_head** (*Hdfobjectstacktype*) –
- **insert** (*bool*) –

Returns success

Return type int

static hdferror_check(*offendingroutine*, *error*[, *fatal*])

Defined at HDFsupport.f90 lines 620-636

Parameters

- **offendingroutine** (*str*) –
- **error** (*int*) –
- **fatal** (*bool*) –

static readdatahdf(*self*[, *existinghdfhead*])

Defined at HDFsupport.f90 lines 5276-5383

Parameters

- **cell** (*Unitcell*) –

- **existinghdfhead** (*Hdfobjectstacktype*) –

static resetfixedlengthflag()
Defined at HDFsupport.f90 lines 5431-5433

static savedatahdf(*self*[, *existinghdfhead*])
Defined at HDFsupport.f90 lines 5157-5253

Parameters

- **cell** (*Unitcell*) –
- **existinghdfhead** (*Hdfobjectstacktype*) –

class pyEMsoft.**Io**(**args*, ***kwargs*)
Module io

Defined at io.f90 lines 52-649

static message(*mess*[, *frm*, *advance*])
Defined at io.f90 lines 101-115

Parameters

- **mess** (*str*) –
- **frm** (*str*) –
- **advance** (*str*) –

static printmatrixcd(*s*, *a*)
Defined at io.f90 lines 637-648

Parameters

- **s** (*str*) –
- **a** (*complex array*) –

static printmatrixd(*s*, *a*)
Defined at io.f90 lines 624-635

Parameters

- **s** (*str*) –
- **a** (*float array*) –

static readvalue(**args*, ***kwargs*)
Defined at io.f90 lines 58-64

Overloaded interface containing the following procedures: _readvalueintshort _readvalueintlong _readvaluerealsingle _readvaluerealdouble _readvaluestring _readvaluestringarray

static writevalue(**args*, ***kwargs*)
Defined at io.f90 lines 66-73

Overloaded interface containing the following procedures: _writevalueintshort _writevalueintlong _writevalueintlonglong _writevaluerealsingle _writevaluerealdouble _writevaluerealcomplex _writevaluestring

class pyEMsoft.**Lambert**(**args*, ***kwargs*)
Module lambert

Defined at Lambert.f90 lines 58-2474

static apply2dlauesymmetry(*ipx*, *ipy*, *isym*, *iequiv*)
nequiv = apply2dlauesymmetry(ipx, ipy, isym, iequiv)

Defined at Lambert.f90 lines 1690-1781

Parameters

- **ipx** (*int*) –
- **ipy** (*int*) –
- **isym** (*int*) –
- **iequiv** (*int array*) –

Returns nequiv

Return type int

static apply2dpgsymmetry(*self*, *ipx*, *ipy*, *isym*, *iequiv*)
nequiv = apply2dpgsymmetry(self, ipx, ipy, isym, iequiv)

Defined at Lambert.f90 lines 1805-1851

Parameters

- **tdpg** (*Symdata2D*) –
- **ipx** (*int*) –
- **ipy** (*int*) –
- **isym** (*int*) –
- **iequiv** (*int array*) –

Returns nequiv

Return type int

static apply3dpgsymmetry(*self*, *ipx*, *ipy*, *ipz*, *npx*, *iequiv*, *usehex=None*, *stereographic=None*, *cubictype=None*)
nequiv = apply3dpgsymmetry(self, ipx, ipy, ipz, npx, iequiv[, usehex, stereographic, cubictype])

Defined at Lambert.f90 lines 1876-1998

Parameters

- **cell** (*Unitcell*) –
- **ipx** (*int*) –
- **ipy** (*int*) –
- **ipz** (*int*) –
- **npx** (*int*) –
- **iequiv** (*int array*) –
- **usehex** (*bool*) –
- **stereographic** (*bool*) –
- **cubictype** (*int*) –

Returns nequiv

Return type int

static getpyramiddouble(*xyz*)
res = getpyramiddouble(xyz)

Defined at Lambert.f90 lines 1156-1197

Parameters **xyz** (*float array*) –

Returns res

Return type int

static getpyramidsingle(*xyz*)
res = getpyramidsingle(xyz)

Defined at Lambert.f90 lines 1101-1142

Parameters **xyz** (*float array*) –

Returns res

Return type int

static getsextantdouble(*xy*)
res = getsextantdouble(xy)

Defined at Lambert.f90 lines 697-724

Parameters **xy** (*float array*) –

Returns res

Return type int

static getsextantsingle(*xy*)
res = getsextantsingle(xy)

Defined at Lambert.f90 lines 655-682

Parameters **xy** (*float array*) –

Returns res

Return type int

static hemicheck(*ix*, *iy*, *npx*, *north*)
xyz = hemicheck(ix, iy, npx, north)

Defined at Lambert.f90 lines 2416-2473

Parameters

- **ix** (*int*) –
- **iy** (*int*) –
- **npx** (*int*) –
- **north** (*bool*) –

Returns xyz

Return type float array

static interpolatelambert(**args*, ***kwargs*)
Defined at Lambert.f90 lines 158-164

Overloaded interface containing the following procedures: _interpolationlambert2dsingle _interpolationlambert2ddouble _interpolationlambert3dsingle _interpolationlambert3dinteger _interpolationlambert4dsingle _interpolationlambert4ddouble4b4

static lambertballtocube(**args*, ***kwargs*)
Defined at Lambert.f90 lines 98-100

Overloaded interface containing the following procedures: _lambert3dcubeinversesingle _lambert3dcubeinversedouble

static lambertballtoquaternion(**args*, ***kwargs*)
Defined at Lambert.f90 lines 104-106

Overloaded interface containing the following procedures: _lambert3dballtoquaternionsingle _lambert3dballtoquaterniondouble

static lambertcubetoball(**args*, ***kwargs*)
Defined at Lambert.f90 lines 93-95

Overloaded interface containing the following procedures: _lambert3dcubeforwardsingle _lambert3dcubeforwarddouble

static lambertcubetoquaternion(**args*, ***kwargs*)
Defined at Lambert.f90 lines 115-117

Overloaded interface containing the following procedures: _lambert3dcubetoquaternionsingle _lambert3dcubetoquaterniondouble

static lambertforward(**args*, ***kwargs*)
Defined at Lambert.f90 lines 143-145

Overloaded interface containing the following procedures: _lambertforwardsingle _lambertforwarddouble

static lambertgetinterpolation(**args*, ***kwargs*)
Defined at Lambert.f90 lines 153-155

Overloaded interface containing the following procedures: _lambertgetinterpolationsingle _lambertgetinterpolationdouble

static lamberthextosphere(**args*, ***kwargs*)
Defined at Lambert.f90 lines 79-81

Overloaded interface containing the following procedures: _lambert2dhexforwardsingle _lambert2dhexforwarddouble

static lambertinverse(**args*, ***kwargs*)
Defined at Lambert.f90 lines 148-150

Overloaded interface containing the following procedures: _lambertinversesingle _lambertinversedouble

static lambertspheretohex(**args*, ***kwargs*)
Defined at Lambert.f90 lines 84-86

Overloaded interface containing the following procedures: _lambert2dhexinversesingle _lambert2dhexinversedouble

static lambertspheretosquare(**args*, ***kwargs*)
Defined at Lambert.f90 lines 73-75

Overloaded interface containing the following procedures: _lambert2dsquareinversesingle _lambert2dsquareinversedouble

static lambertsquaretosphere(**args*, ***kwargs*)
Defined at Lambert.f90 lines 68-70

Overloaded interface containing the following procedures: _lambert2dsquareforwardsingle _lambert2dsquareforwarddouble

static samplevmf(*mu*, *kappa*, *vmfscale*, *inten*, *npx*, *nix*, *niy*, *w*, *mlpnh*, *mlpsh*, *legendrearray*)
Defined at Lambert.f90 lines 2360-2400

Parameters

- **mu** (*float array*) –
- **kappa** (*float*) –
- **vmfscale** (*float*) –
- **inten** (*float*) –
- **npx** (*int*) –
- **nix** (*int*) –
- **niy** (*int*) –
- **w** (*int*) –
- **mlpnh** (*float array*) –
- **mlpsh** (*float array*) –
- **legendrearray** (*float array*) –

static stereographicforward(**args*, ***kwargs*)
Defined at Lambert.f90 lines 132-134

Overloaded interface containing the following procedures: _stereographicforwardsingle _stereographicforwarddouble

static stereographicinverse(**args*, ***kwargs*)
Defined at Lambert.f90 lines 137-139

Overloaded interface containing the following procedures: _stereographicinversesingle _stereographicinversedouble

class pyEMsoft.**Local**(**args*, ***kwargs*)
Module local

Defined at local.f90 lines 68-1478

configstructurenames
Element configstructurenames ftype=character(30) pytype=str

Defined at local.f90 line 182

dataunit
Element dataunit ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 113

dataunit2
Element dataunit2 ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 113

dataunit3
Element dataunit3 ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 113

dbl
Element dbl ftype=integer pytype=int

Defined at local.f90 line 79

displayconfigfilemissingmessage
Element displayconfigfilemissingmessage ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 106

displayemsoftwarningmessages
Element displayemsoftwarningmessages ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 105

static emsoft(*progname*, *progdesc*[, *emconfig*, *stdout*, *makeconfig*])
Defined at local.f90 lines 1104-1171

Parameters

- **progname** (*str*) –
- **progdesc** (*str*) –
- **emconfig** (*Configstructuretype*) –
- **stdout** (*int*) –
- **makeconfig** (*bool*) –

static emsoft_fromnativepath(*inpath*)
outpath = emsoft_fromnativepath(inpath)

Defined at local.f90 lines 1222-1241

Parameters **inpath** (*str*) –

Returns **outpath**

Return type str

static emsoft_getconfigpath()
configpath = emsoft_getconfigpath()

Defined at local.f90 lines 434-444

Returns **configpath**

Return type str

static emsoft_getemdatapathname()
emdatapathname = emsoft_getemdatapathname()

Defined at local.f90 lines 504-531

Returns **emdatapathname**

Return type str

static emsoft_getemdevelop()
emdevelop = emsoft_getemdevelop()

Defined at local.f90 lines 713-728

Returns **emdevelop**

Return type bool

static emsoft_getemsoftbuilddate()
builddate = emsoft_getemsoftbuilddate()

Defined at local.f90 lines 394-398

Returns builddate

Return type str

static emsoft_getemsofthdftest()
dohdftest = emsoft_getemsofthdftest()

Defined at local.f90 lines 318-329

Returns dohdftest

Return type bool

static emsoft_getemsoftnativedelimiter()
emsoftnativedelimiter = emsoft_getemsoftnativedelimiter()

Defined at local.f90 lines 412-420

Returns emsoftnativedelimiter

Return type str

static emsoft_getemsoftpathname()
emsoftpathname = emsoft_getemsoftpathname()

Defined at local.f90 lines 460-488

Returns emsoftpathname

Return type str

static emsoft_getemsoftplatform()
platform = emsoft_getemsoftplatform()

Defined at local.f90 lines 254-258

Returns platform

Return type str

static emsoft_getemsoftrevision()
revision = emsoft_getemsoftrevision()

Defined at local.f90 lines 376-380

Returns revision

Return type str

static emsoft_getemsofttestingpath()
buildpath = emsoft_getemsofttestingpath()

Defined at local.f90 lines 339-344

Returns buildpath

Return type str

static emsoft_getemsofttestpath()
testpath = emsoft_getemsofttestpath()

Defined at local.f90 lines 294-303

Returns testpath

Return type str

static emsoft_getemsoftversion()
version = emsoft_getemsoftversion()

Defined at local.f90 lines 358-362

Returns version

Return type str

static emsoft_getemtmppathname()
emtmppathname = emsoft_getemtmppathname()

Defined at local.f90 lines 588-615

Returns emtmppathname

Return type str

static emsoft_getemxtalfolderpathname()
emxtalfolderpathname = emsoft_getemxtalfolderpathname()

Defined at local.f90 lines 546-573

Returns emxtalfolderpathname

Return type str

static emsoft_getfftwwisdomfilename()
fftwwisdomfilename = emsoft_getfftwwisdomfilename()

Defined at local.f90 lines 1001-1005

Returns fftwwisdomfilename

Return type str

static emsoft_geth5copypath()
h5copypath = emsoft_geth5copypath()

Defined at local.f90 lines 272-280

Returns h5copypath

Return type str

static emsoft_getjsonparameter(*ep*, *nobackslash=None*)
param = emsoft_getjsonparameter(ep[, nobackslash])

Defined at local.f90 lines 768-838

Parameters

- **ep** (*str*) –
- **nobackslash** (*bool*) –

Returns param

Return type str

static emsoft_getnotify()
notify = emsoft_getnotify()

Defined at local.f90 lines 197-202

Returns notify

Return type str

static emsoft_getopenclpathname()
openclpathname = emsoft_getopenclpathname()

Defined at local.f90 lines 964-968

Returns openclpathname

Return type str

static emsoft_getrandomseedfilename()
randomseedfilename = emsoft_getrandomseedfilename()

Defined at local.f90 lines 1073-1077

Returns randomseedfilename

Return type str

static emsoft_getrelease()
release = emsoft_getrelease()

Defined at local.f90 lines 742-750

Returns release

Return type bool

static emsoft_getresourcepathname()
resourcepathname = emsoft_getresourcepathname()

Defined at local.f90 lines 899-903

Returns resourcepathname

Return type str

static emsoft_getslackchannel()
slackchannel = emsoft_getslackchannel()

Defined at local.f90 lines 235-240

Returns slackchannel

Return type str

static emsoft_getslackwebhookurl()
slackwebhookurl = emsoft_getslackwebhookurl()

Defined at local.f90 lines 216-221

Returns slackwebhookurl

Return type str

static emsoft_gettemplatecodefilename()
templatecodefilename = emsoft_gettemplatecodefilename()

Defined at local.f90 lines 1019-1023

Returns templatecodefilename

Return type str

static emsoft_gettemplatepathname(*json=None*)
templatepathname = emsoft_gettemplatepathname([json])

Defined at local.f90 lines 853-866

Parameters **json** (*bool*) –

Returns **templatepathname**

Return type str

static emsoft_getuser()
username = emsoft_getuser()

Defined at local.f90 lines 944-950

Returns **username**

Return type str

static emsoft_getuseremail()
useremail = emsoft_getuseremail()

Defined at local.f90 lines 690-698

Returns **useremail**

Return type str

static emsoft_getuserhomepath()
userhomepathname = emsoft_getuserhomepath()

Defined at local.f90 lines 917-930

Returns **userhomepathname**

Return type str

static emsoft_getuserlocation()
userlocation = emsoft_getuserlocation()

Defined at local.f90 lines 660-675

Returns **userlocation**

Return type str

static emsoft_getusername()
username = emsoft_getusername()

Defined at local.f90 lines 630-645

Returns **username**

Return type str

static emsoft_getwikicodefilename()
wikicodefilename = emsoft_getwikicodefilename()

Defined at local.f90 lines 1037-1041

Returns **wikicodefilename**

Return type str

static emsoft_getwikipathname()
wikipathname = emsoft_getwikipathname()

Defined at local.f90 lines 881-885

Returns **wikipathname**

Return type str

static emsoft_getwyckoffpositionsfilename()
wpfilename = emsoft_getwyckoffpositionsfilename()

Defined at local.f90 lines 1055-1059

Returns wpfilename

Return type str

static emsoft_getxtalpathname()
xtalpathname = emsoft_getxtalpathname()

Defined at local.f90 lines 983-987

Returns xtalpathname

Return type str

static emsoft_path_init([*config*])
Defined at local.f90 lines 1269-1400

Parameters **config** (*bool*) –

static emsoft_tonativepath(*inpath*)
outpath = emsoft_tonativepath(inpath)

Defined at local.f90 lines 1187-1206

Parameters **inpath** (*str*) –

Returns outpath

Return type str

fnlen
Element fnlen ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 101

fortran_real_4
Element fortran_real_4 ftype=integer pytype=int

Defined at local.f90 line 94

fortran_real_8
Element fortran_real_8 ftype=integer pytype=int

Defined at local.f90 line 96

ill
Element ill ftype=integer pytype=int

Defined at local.f90 line 88

irg
Element irg ftype=integer pytype=int

Defined at local.f90 line 86

ish
Element ish ftype=integer pytype=int

Defined at local.f90 line 84

jsonck
Element jsonck ftype=integer pytype=int

Defined at local.f90 line 98

psunit
Element psunit ftype=integer(kind=irg) pytype=int

Defined at local.f90 line 113

sgl
Element sgl ftype=integer pytype=int

Defined at local.f90 line 77

static timestamp([*stdout*, *timestring*, *datestring*])
Defined at local.f90 lines 1423-1478

Parameters

- **stdout** (*int*) –
- **timestring** (*str*) –
- **datestring** (*str*) –

wraparraysize
Element wraparraysize ftype=integer(c_int32_t) pytype=int

Defined at local.f90 line 110

class pyEMsoft.**Math**(**args*, ***kwargs*)
Module math

Defined at math.f90 lines 52-3077

static besseli0(*x*)
bessi0 = besseli0(x)

Defined at math.f90 lines 700-720

Parameters **x** (*float*) –

Returns **bessi0**

Return type float

static besseli1(*x*)
bessi1 = besseli1(x)

Defined at math.f90 lines 743-763

Parameters **x** (*float*) –

Returns **bessi1**

Return type float

static besselin(*x*, *n*)
bessi = besselin(x, n)

Defined at math.f90 lines 628-677

Parameters

- **x** (*float*) –
- **n** (*int*) –

Returns **bessi**

Return type float

static c4_normal_01(*seed*)
c4_normal_01 = c4_normal_01(seed)

Defined at math.f90 lines 773-812

Parameters **seed** (*int*) –

Returns **c4_normal_01**

Return type complex

static c8_normal_01(*seed*)
c8_normal_01 = c8_normal_01(seed)

Defined at math.f90 lines 814-853

Parameters **seed** (*int*) –

Returns **c8_normal_01**

Return type complex

static calcdeterminant(*a*, *m*, *n*)
determinant = calcdeterminant(a, m, n)

Defined at math.f90 lines 2993-3014

Parameters

- **a** (*float array*) –
- **m** (*int*) –
- **n** (*int*) –

Returns **determinant**

Return type float

static cinvert(*a*, *b*)
Defined at math.f90 lines 467-490

Parameters

- **a** (*complex array*) –
- **b** (*complex array*) –

static cross3(**args*, ***kwargs*)
Defined at math.f90 lines 59-61

Overloaded interface containing the following procedures: _cross3 _cross3_d

static cubicroots(*co*, *x*)
Defined at math.f90 lines 2537-2555

Parameters

- **co** (*float array*) –
- **x** (*complex array*) –

static el1k(*phi*, *k*)
res = el1k(phi, k)

Defined at math.f90 lines 2464-2479

Parameters

- **phi** (*float*) –
- **k** (*float*) –

Returns res

Return type float

static el2 (*x*, *qqc*, *aa*, *bb*)
res = el2(x, qqc, aa, bb)

Defined at math.f90 lines 2404-2462

Parameters

- **x** (*float*) –
- **qqc** (*float*) –
- **aa** (*float*) –
- **bb** (*float*) –

Returns res

Return type float

static el2k (*phi*, *k*)
res = el2k(phi, k)

Defined at math.f90 lines 2481-2496

Parameters

- **phi** (*float*) –
- **k** (*float*) –

Returns res

Return type float

static get_bit_parameters (*bd*, *verbose=None*)
numbits, bitrange, bitmode = get_bit_parameters(bd[, verbose])

Defined at math.f90 lines 278-324

Parameters

- **bd** (*str*) –
- **verbose** (*bool*) –

Returns

- **numbits** (*int*)
- **bitrange** (*float*)
- **bitmode** (*str*)
- **==========** – analyze the bitdepth parameter; if we have integers, then we need to analyze the digits in the string to figure out how to scale the data. '10int' means that we store the data as 32-bit integers, but the scaled values range from 0 to 2^(10)-1

static getpolardecomposition (*f*, *rmatrix*, *smatrix*)
Defined at math.f90 lines 226-261

Parameters

- **f** (*float array*) –
- **rmatrix** (*float array*) –
- **smatrix** (*float array*) –

static i4_huge()
i4_huge = i4_huge()

Defined at math.f90 lines 855-890

Returns i4_huge

Return type int

static i4_normal_ab(*a*, *b*, *seed*)
i4_normal_ab = i4_normal_ab(a, b, seed)

Defined at math.f90 lines 892-942

Parameters

- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –

Returns i4_normal_ab

Return type int

static i8_normal_ab(*a*, *b*, *seed*)
i8_normal_ab = i8_normal_ab(a, b, seed)

Defined at math.f90 lines 944-995

Parameters

- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –

Returns i8_normal_ab

Return type int

static infty()
infinity = infty()

Defined at math.f90 lines 152-157

Returns infinity

Return type float

static inftyd()
infinity = inftyd()

Defined at math.f90 lines 169-174

Returns infinity

Return type float

static jaccard_distance(*img1*, *img2*, *nn*, *mutualinformation=None*)
jd = jaccard_distance(img1, img2, nn[, mutualinformation])

Defined at math.f90 lines 3031-3077

Parameters

- **img1** (*int array*) –
- **img2** (*int array*) –
- **nn** (*int*) –
- **mutualinformation** (*bool*) –

Returns **jd**

Return type float

static kcluster(*matrix*, *nrow*, *ncol*, *nclusters*, *niter*, *indexarray*)
Defined at math.f90 lines 2654-2706

Parameters

- **matrix** (*float array*) –
- **nrow** (*int*) –
- **ncol** (*int*) –
- **nclusters** (*int*) –
- **niter** (*int*) –
- **indexarray** (*int array*) –

static kclusterweights(*matrix*, *nrow*, *ncol*, *nclusters*, *niter*, *wts*)
Defined at math.f90 lines 2573-2635

Parameters

- **matrix** (*float array*) –
- **nrow** (*int*) –
- **ncol** (*int*) –
- **nclusters** (*int*) –
- **niter** (*int*) –
- **wts** (*float array*) –

static kdelta(*i*, *j*)
k = kdelta(i, j)

Defined at math.f90 lines 2511-2518

Parameters

- **i** (*int*) –
- **j** (*int*) –

Returns **k**

Return type int

static laguer(*a*, *m*, *x*, *eps*, *polish*)
Defined at math.f90 lines 2083-2137

Parameters

- **a** (*complex array*) –
- **m** (*int*) –
- **x** (*complex*) –
- **eps** (*float*) –
- **polish** (*bool*) –

static matrixexponential(*a*, *e*, *z0*, *tp*, *nn*)
Defined at math.f90 lines 515-604

Parameters

- **a** (*complex array*) –
- **e** (*complex array*) –
- **z0** (*float*) –
- **tp** (*str*) –
- **nn** (*int*) –

static minvert(**args*, ***kwargs*)
Defined at math.f90 lines 55-57

Overloaded interface containing the following procedures: _minvert _minvert_d

static nan()
x = nan()

Defined at math.f90 lines 186-192

Returns x

Return type float

static nan_d()
x = nan_d()

Defined at math.f90 lines 204-210

Returns x

Return type float

static point_inside_polygon(*px*, *py*, *xx*, *yy*)
inorout = point_inside_polygon(px, py, xx, yy)

Defined at math.f90 lines 2367-2396

Parameters

- **px** (*float*) –
- **py** (*float*) –
- **xx** (*float array*) –
- **yy** (*float array*) –

Returns inorout

Return type int

static point_inside_triangle(*v0*, *v1*, *v2*)
point_inside_triangle = point_inside_triangle(v0, v1, v2)

Defined at math.f90 lines 2240-2257

Parameters

- **v0** (*float array*) –
- **v1** (*float array*) –
- **v2** (*float array*) –

Returns point_inside_triangle

Return type bool

static printmatrix(*s*, *a*)
Defined at math.f90 lines 2207-2219

Parameters

- **s** (*str*) –
- **a** (*float array*) –

static r4_normal_01(*seed*)
r4_normal_01 = r4_normal_01(seed)

Defined at math.f90 lines 997-1040

Parameters **seed** (*int*) –

Returns r4_normal_01

Return type float

static r4_normal_ab(*a*, *b*, *seed*)
r4_normal_ab = r4_normal_ab(a, b, seed)

Defined at math.f90 lines 1042-1090

Parameters

- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –

Returns r4_normal_ab

Return type float

static r4_uniform_01(*seed*)
r4_uniform_01 = r4_uniform_01(seed)

Defined at math.f90 lines 1092-1185

Parameters **seed** (*int*) –

Returns r4_uniform_01

Return type float

static r4vec_normal_ab(*n*, *a*, *b*, *seed*, *x*)
Defined at math.f90 lines 1271-1378

Parameters

- **n** (*int*) –
- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –
- **x** (*float array*) –

static r4vec_uniform_01 (*n*, *seed*, *r*)
Defined at math.f90 lines 1187-1269

Parameters

- **n** (*int*) –
- **seed** (*int*) –
- **r** (*float array*) –

static r8_normal_01 (*seed*)
r8_normal_01 = r8_normal_01(seed)

Defined at math.f90 lines 1380-1423

Parameters **seed** (*int*) –

Returns r8_normal_01

Return type float

static r8_normal_ab (*a*, *b*, *seed*)
r8_normal_ab = r8_normal_ab(a, b, seed)

Defined at math.f90 lines 1425-1473

Parameters

- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –

Returns r8_normal_ab

Return type float

static r8_uniform_01 (*seed*)
r8_uniform_01 = r8_uniform_01(seed)

Defined at math.f90 lines 1475-1564

Parameters **seed** (*int*) –

Returns r8_uniform_01

Return type float

static r8mat_normal_01 (*m*, *n*, *seed*, *r*)
Defined at math.f90 lines 1566-1629

Parameters

- **m** (*int*) –
- **n** (*int*) –
- **seed** (*int*) –

- **r** (*float array*) –

static r8mat_normal_ab (*m*, *n*, *a*, *b*, *seed*, *r*)
Defined at math.f90 lines 1631-1698

Parameters

- **m** (*int*) –
- **n** (*int*) –
- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –
- **r** (*float array*) –

static r8vec_normal_01 (*n*, *seed*, *x*)
Defined at math.f90 lines 1700-1802

Parameters

- **n** (*int*) –
- **seed** (*int*) –
- **x** (*float array*) –

static r8vec_normal_ab (*n*, *a*, *b*, *seed*, *x*)
Defined at math.f90 lines 1804-1909

Parameters

- **n** (*int*) –
- **a** (*float*) –
- **b** (*float*) –
- **seed** (*int*) –
- **x** (*float array*) –

static r8vec_uniform_01 (*n*, *seed*, *r*)
Defined at math.f90 lines 1911-1981

Parameters

- **n** (*int*) –
- **seed** (*int*) –
- **r** (*float array*) –

static rank_points (*p1*, *p2*, *p3*, *p4*, *xx*, *yy*)
Defined at math.f90 lines 2281-2327

Parameters

- **p1** (*float array*) –
- **p2** (*float array*) –
- **p3** (*float array*) –
- **p4** (*float array*) –
- **xx** (*float array*) –

- **yy** (*float array*) –

static reorganizeclusters(*matrix*, *nrow*, *ncol*, *nclusters*, *nsc*, *nsr*, *indexarray*)
Defined at math.f90 lines 2727-2835

Parameters

- **matrix** (*float array*) –
- **nrow** (*int*) –
- **ncol** (*int*) –
- **nclusters** (*int*) –
- **nsc** (*int*) –
- **nsr** (*int*) –
- **indexarray** (*int array*) –
- **set things up for the Monte Carlo part -------** (*--------*) –
- **----------** – reshape the index array for easier neighbor finding
- **= reshape(IndexArray, (/ NSC, NSR /))** (*IA*) –
- **(,) 'reshaped IndexArray ', shape (IA)** (*write*) – count the number of points in each cluster and add the x-y coordinates to get the cluster centroid

static transfourthranktensor(*al*, *cin*, *cout*)
Defined at math.f90 lines 2005-2060

Parameters

- **al** (*float array*) –
- **cin** (*float array*) –
- **cout** (*float array*) –

static trilinear_splat(*r*, *dr*, *init=None*)
grid3 = trilinear_splat(r, dr[, init])

Defined at math.f90 lines 2940-2971

Parameters

- **r** (*float array*) –
- **dr** (*float array*) –
- **init** (*bool*) –

Returns **grid3**

Return type float array

static vecnorm(**args*, ***kwargs*)
Defined at math.f90 lines 63-67

Overloaded interface containing the following procedures: _vecnorm _vecnorm_d _vecnorm2 _vecnorm2_d

static vectormatch(*n*, *va*, *vb*)
nce = vectormatch(n, va, vb)

Defined at math.f90 lines 2900-2917

Parameters

- **n** (*int*) –
- **va** (*int array*) –
- **vb** (*int array*) –

Returns nce

Return type int

static zroots(*a*, *roots*)
Defined at math.f90 lines 2157-2201

Parameters

- **a** (*complex array*) –
- **roots** (*complex array*) –

class pyEMsoft.**Namelisthandlers**(**args*, ***kwargs*)
Module namelisthandlers

Defined at NameListHandlers.f90 lines 42-9091

static getadpnamelist(*nmlfile*, *adpnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 5949-6041

Parameters

- **nmlfile** (*str*) –
- **adpnl** (*Adpnamelisttype*) –
- **initonly** (*bool*) –

static getangnamelist(*nmlfile*, *csnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 237-280

Parameters

- **nmlfile** (*str*) –
- **csnl** (*Angnamelisttype*) –
- **initonly** (*bool*) –

static getaverageorientationnamelist(*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2115-2168

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Averageorientationnamelisttype*) –
- **initonly** (*bool*) –

static getbsenamelist(*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3473-3567

Parameters

- **nmlfile** (*str*) –
- **enl** (*Bsenamelisttype*) –
- **initonly** (*bool*) –

static getcbednamelist (*nmlfile*, *cbednl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4686-4753

Parameters

- **nmlfile** (*str*) –
- **cbednl** (*Cbednamelisttype*) –
- **initonly** (*bool*) –

static getchangesettingnamelist (*nmlfile*, *csnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 125-162

Parameters

- **nmlfile** (*str*) –
- **csnl** (*Changesettingnamelisttype*) –
- **initonly** (*bool*) –

static getconvertorientationsnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1582-1657

Parameters

- **nmlfile** (*str*) –
- **enl** (*Convertorientationsnamelisttype*) –
- **initonly** (*bool*) –

static getcplmmasternamelist (*nmlfile*, *omnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 908-971

Parameters

- **nmlfile** (*str*) –
- **omnl** (*Cplmmasternamelisttype*) –
- **initonly** (*bool*) –

static getcplmnamelist (*nmlfile*, *omnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1327-1377

Parameters

- **nmlfile** (*str*) –
- **omnl** (*Cplmnamelisttype*) –
- **initonly** (*bool*) –

static getctemqcnamelist (*nmlfile*, *ctemqcnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3009-3070

Parameters

- **nmlfile** (*str*) –
- **ctemqcnl** (*Ctemqcnamelisttype*) –
- **initonly** (*bool*) –

static getctfnamelist (*nmlfile*, *csnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 178-221

Parameters

- **nmlfile** (*str*) –
- **csnl** (*Ctfnamelisttype*) –
- **initonly** (*bool*) –

static getdictindxopenclnamelist(*nmlfile*, *dictindxnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 5202-5283

Parameters

- **nmlfile** (*str*) –
- **dictindxnl** (*Dictindxopencllisttype*) –
- **initonly** (*bool*) –

static getdisorientationsnamelist(*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2060-2100

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Disorientationsnamelisttype*) –
- **initonly** (*bool*) –

static getdpmergenamelist(*nmlfile*, *dpmnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 6108-6157

Parameters

- **nmlfile** (*str*) –
- **dpmnl** (*Dpmergenamelisttype*) –
- **initonly** (*bool*) –

static getdvsdnamelist(*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2290-2345

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Dvsdnamelisttype*) –
- **initonly** (*bool*) –

static getebsd2dqcmasternamelist(*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8720-8769

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsd2Dqcmasternamelisttype*) –
- **initonly** (*bool*) –

static getebsdclusternamelist(*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2878-2927

Parameters

- **nmlfile** (*str*) –

- **emnl** (*Ebsdclusternamelisttype*) –
- **initonly** (*bool*) –

static getebsddefectnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3582-3676

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsddefectnamelisttype*) –
- **initonly** (*bool*) –

static getebsddenamelist (*nmlfile*, *enl*, *de*, *p*[, *initonly*])
Defined at NameListHandlers.f90 lines 3691-3936

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsdnamelisttype*) –
- **de** (*Ebsddenamelisttype*) –
- **p** (*Ebsddipreviewnamelisttype*) –
- **initonly** (*bool*) –

static getebsddipreviewnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 5418-5503

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsddipreviewnamelisttype*) –
- **initonly** (*bool*) –

static getebsdfullnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7753-7887

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsdfullnamelisttype*) –
- **initonly** (*bool*) –

static getebsdindexingnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 5520-5756

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsdindexingnamelisttype*) –
- **initonly** (*bool*) –

static getebsdmasternamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2362-2431

Parameters

- **nmlfile** (*str*) –

- **emnl** (*Ebsdmasternamelisttype*) –
- **initonly** (*bool*) –

static getebsdmasteropenclnamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2805-2863

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Ebsdmasteropenclnamelisttype*) –
- **initonly** (*bool*) –

static getebsdmastershtnamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2520-2593

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Ebsdmastershtnamelisttype*) –
- **initonly** (*bool*) –

static getebsdnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3290-3458

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsdnamelisttype*) –
- **initonly** (*bool*) –

static getebsdoverlapnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4093-4204

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsdoverlapnamelisttype*) –
- **initonly** (*bool*) –

static getebsdqcmasternamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8667-8705

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ebsdqcmasternamelisttype*) –
- **initonly** (*bool*) –

static getebsdsinglemasternamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2608-2663

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Ebsdsinglemasternamelisttype*) –
- **initonly** (*bool*) –

static getecciname list (*nmlfile*, *eccinl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4993-5098

Parameters

- **nmlfile** (*str*) –
- **eccinl** (*Eccinamelisttype*) –
- **initonly** (*bool*) –

static getecpindexingnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 6513-6611

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ecpindexingnamelisttype*) –
- **initonly** (*bool*) –

static getecpmasternamelist (*nmlfile*, *ecpnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3087-3146

Parameters

- **nmlfile** (*str*) –
- **ecpnl** (*Ecpmasternamelisttype*) –
- **initonly** (*bool*) –

static getecpnamelist (*nmlfile*, *ecpnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4475-4589

Parameters

- **nmlfile** (*str*) –
- **ecpnl** (*Ecpnamelisttype*) –
- **initonly** (*bool*) –

static getecppatternnamelist (*nmlfile*, *ecpnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4768-4811

Parameters

- **nmlfile** (*str*) –
- **ecpnl** (*Ecppatternnamelisttype*) –
- **initonly** (*bool*) –

static getecpqcmasternamelist (*nmlfile*, *ecpnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2945-2991

Parameters

- **nmlfile** (*str*) –
- **ecpnl** (*Ecpqcmasternamelisttype*) –
- **initonly** (*bool*) –

static getecpsinglenamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7004-7073

Parameters

- **nmlfile** (*str*) –
- **enl** (*Ecpsinglenamelisttype*) –
- **initonly** (*bool*) –

static getecpzanamelist (*nmlfile*, *ecpnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4405-4458

Parameters

- **nmlfile** (*str*) –
- **ecpnl** (*Ecpzanamelisttype*) –
- **initonly** (*bool*) –

static geteecmasternamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2446-2504

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Eecmasternamelisttype*) –
- **initonly** (*bool*) –

static getemcbed2dqcnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8603-8652

Parameters

- **nmlfile** (*str*) –
- **enl** (*Emcbed2Dqcnamelisttype*) –
- **initonly** (*bool*) –

static getemcbedqcnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8541-8588

Parameters

- **nmlfile** (*str*) –
- **enl** (*Emcbedqcnamelisttype*) –
- **initonly** (*bool*) –

static getemdpfit4namelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 6626-6827

Parameters

- **nmlfile** (*str*) –
- **enl** (*Emdpfit4Listtype*) –
- **initonly** (*bool*) –

static getemdpfitnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 6842-6989

Parameters

- **nmlfile** (*str*) –

- **enl** (*Emdpfitlisttype*) –
- **initonly** (*bool*) –

static getemgammanamelist (*nmlfile*, *epf*[, *initonly*])
Defined at NameListHandlers.f90 lines 8090-8158

Parameters

- **nmlfile** (*str*) –
- **epf** (*Emgammanamelisttype*) –
- **initonly** (*bool*) –

static getemgammaopenclnamelist (*nmlfile*, *epf*[, *initonly*])
Defined at NameListHandlers.f90 lines 8173-8236

Parameters

- **nmlfile** (*str*) –
- **epf** (*Emgammaopenclnamelisttype*) –
- **initonly** (*bool*) –

static getemgammastemnamelist (*nmlfile*, *epf*[, *initonly*])
Defined at NameListHandlers.f90 lines 8400-8457

Parameters

- **nmlfile** (*str*) –
- **epf** (*Emgammastemnamelisttype*) –
- **initonly** (*bool*) –

static getemhh4namelist (*nmlfile*, *hhnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8942-9091

Parameters

- **nmlfile** (*str*) –
- **hhnl** (*Emhh4Namelisttype*) –
- **initonly** (*bool*) –

static getemintegratestemnamelist (*nmlfile*, *isnml*[, *initonly*])
Defined at NameListHandlers.f90 lines 8879-8926

Parameters

- **nmlfile** (*str*) –
- **isnml** (*Emintegratestemnamelisttype*) –
- **initonly** (*bool*) –

static getemmdstemnamelist (*nmlfile*, *msnml*[, *initonly*])
Defined at NameListHandlers.f90 lines 8781-8867

Parameters

- **nmlfile** (*str*) –
- **msnml** (*Emmdstemnamelisttype*) –
- **initonly** (*bool*) –

static getemtgbstemnamelist (*nmlfile*, *epf*[, *initonly*])
Defined at NameListHandlers.f90 lines 8472-8526

Parameters

- **nmlfile** (*str*) –
- **epf** (*Emtgbstemnamelisttype*) –
- **initonly** (*bool*) –

static getemtwophasenamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8251-8301

Parameters

- **nmlfile** (*str*) –
- **enl** (*Emtwophasenamelisttype*) –
- **initonly** (*bool*) –

static geteulersnamelist (*nmlfile*, *csnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 296-342

Parameters

- **nmlfile** (*str*) –
- **csnl** (*Eulersnamelisttype*) –
- **initonly** (*bool*) –

static getfitalphavariantsnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7343-7393

Parameters

- **nmlfile** (*str*) –
- **enl** (*Fitalphavarianttype*) –
- **initonly** (*bool*) –

static getfitorientationpsnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7408-7458

Parameters

- **nmlfile** (*str*) –
- **enl** (*Fitorientationpstype*) –
- **initonly** (*bool*) –

static getgbodmnamelist (*nmlfile*, *gbonl*[, *initonly*])
Defined at NameListHandlers.f90 lines 420-466

Parameters

- **nmlfile** (*str*) –
- **gbonl** (*Gbodmnamelisttype*) –
- **initonly** (*bool*) –

static getgbonamelist (*nmlfile*, *gbonl*[, *initonly*])
Defined at NameListHandlers.f90 lines 358-404

Parameters

- **nmlfile** (*str*) –
- **gbonl** (*Gbonamelisttype*) –
- **initonly** (*bool*) –

static getgrainviznamelist (*nmlfile*, *gvnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 60-109

Parameters

- **nmlfile** (*str*) –
- **gvnl** (*Grainviznamelisttype*) –
- **initonly** (*bool*) –

static getkamnamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2235-2275

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Kamnamelisttype*) –
- **initonly** (*bool*) –

static getkinematicalnamelist (*nmlfile*, *knl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3228-3274

Parameters

- **nmlfile** (*str*) –
- **knl** (*Kinematicalnamelisttype*) –
- **initonly** (*bool*) –

static getkosselmasternamelist (*nmlfile*, *knl*[, *initonly*])
Defined at NameListHandlers.f90 lines 832-893

Parameters

- **nmlfile** (*str*) –
- **knl** (*Kosselmasternamelisttype*) –
- **initonly** (*bool*) –

static getkosselnamelist (*nmlfile*, *knl*[, *initonly*])
Defined at NameListHandlers.f90 lines 747-817

Parameters

- **nmlfile** (*str*) –
- **knl** (*Kosselnamelisttype*) –
- **initonly** (*bool*) –

static getlacbednamelist (*nmlfile*, *lacbednl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4604-4671

Parameters

- **nmlfile** (*str*) –

- **lacbednl** (*Lacbednamelisttype*) –
- **initonly** (*bool*) –

static getlauemasternamelist (*nmlfile*, *lmnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 986-1074

Parameters

- **nmlfile** (*str*) –
- **lmnl** (*Lauemasternamelisttype*) –
- **initonly** (*bool*) –

static getlauenamelist (*nmlfile*, *lnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1090-1169

Parameters

- **nmlfile** (*str*) –
- **lnl** (*Lauenamelisttype*) –
- **initonly** (*bool*) –

static getlaueslitnamelist (*nmlfile*, *lnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1185-1312

Parameters

- **nmlfile** (*str*) –
- **lnl** (*Laueslitnamelisttype*) –
- **initonly** (*bool*) –

static getlocalosmmasternamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2678-2721

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Localosmnamelisttype*) –
- **initonly** (*bool*) –

static getlorentznamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 565-669

Parameters

- **nmlfile** (*str*) –
- **enl** (*Lorentznamelisttype*) –
- **initonly** (*bool*) –

static getmcclfoilnamelist (*nmlfile*, *mcnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7654-7738

Parameters

- **nmlfile** (*str*) –
- **mcnl** (*Mcclnamelisttype*) –
- **initonly** (*bool*) –

static getmcclmultilayernamelist (*nmlfile*, *mcnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1960-2045

Parameters

- **nmlfile** (*str*) –
- **mcnl** (*Mcclmultilayernamelisttype*) –
- **initonly** (*bool*) –

static getmcclnamelist (*nmlfile*, *mcnl*[, *initonly*, *writetofile*])
Defined at NameListHandlers.f90 lines 1801-1945

Parameters

- **nmlfile** (*str*) –
- **mcnl** (*Mcclnamelisttype*) –
- **initonly** (*bool*) –
- **writetofile** (*str*) –

static getmcclspherenamelist (*nmlfile*, *mcnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7542-7639

Parameters

- **nmlfile** (*str*) –
- **mcnl** (*Mcclnamelisttype*) –
- **initonly** (*bool*) –

static getmclipssnamelist (*nmlfile*, *mcnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1479-1567

Parameters

- **nmlfile** (*str*) –
- **mcnl** (*Mclipssnamelisttype*) –
- **initonly** (*bool*) –

static getmcnamelist (*nmlfile*, *mcnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1392-1464

Parameters

- **nmlfile** (*str*) –
- **mcnl** (*Mcnamelisttype*) –
- **initonly** (*bool*) –

static getmdelectronpropnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 8316-8385

Parameters

- **nmlfile** (*str*) –
- **enl** (*Mdelectronpropnamelisttype*) –
- **initonly** (*bool*) –

static getmultiphasenamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 685-731

Parameters

- **nmlfile** (*str*) –
- **enl** (*Multiphasenamelisttype*) –
- **initonly** (*bool*) –

static getorientationsimilaritynamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2183-2220

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Orientationsimilaritynamelisttype*) –
- **initonly** (*bool*) –

static getorientationviznamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 1673-1784

Parameters

- **nmlfile** (*str*) –
- **enl** (*Orientationviznamelisttype*) –
- **initonly** (*bool*) –

static getoslerpnamelist (*nmlfile*, *onl*[, *initonly*])
Defined at NameListHandlers.f90 lines 482-549

Parameters

- **nmlfile** (*str*) –
- **onl** (*Oslerpnamelisttype*) –
- **initonly** (*bool*) –

static getosmnamelist (*nmlfile*, *osmnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 6056-6093

Parameters

- **nmlfile** (*str*) –
- **osmnl** (*Osmnamelisttype*) –
- **initonly** (*bool*) –

static getpedindxnamelist (*nmlfile*, *pednl*[, *initonly*])
Defined at NameListHandlers.f90 lines 5298-5403

Parameters

- **nmlfile** (*str*) –
- **pednl** (*Pedkinindxlisttype*) –
- **initonly** (*bool*) –

static getpedkinnamelist (*nmlfile*, *pednl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4826-4888

Parameters

- **nmlfile** (*str*) –
- **pednl** (*Pedkinnamelisttype*) –
- **initonly** (*bool*) –

static getpedzanamelist (*nmlfile*, *pednl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4903-4977

Parameters

- **nmlfile** (*str*) –
- **pednl** (*Pedzanamelisttype*) –
- **initonly** (*bool*) –

static getpfinversionnamelist (*nmlfile*, *epf*[, *initonly*])
Defined at NameListHandlers.f90 lines 8025-8075

Parameters

- **nmlfile** (*str*) –
- **epf** (*Pfinversionnamelisttype*) –
- **initonly** (*bool*) –

static getrefinemartensitenamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7473-7527

Parameters

- **nmlfile** (*str*) –
- **enl** (*Refinemartensitetype*) –
- **initonly** (*bool*) –

static getrefineorientationnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7251-7328

Parameters

- **nmlfile** (*str*) –
- **enl** (*Refineorientationtype*) –
- **initonly** (*bool*) –

static getreflectornamelist (*nmlfile*, *rnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3161-3213

Parameters

- **nmlfile** (*str*) –
- **rnl** (*Reflectornamelisttype*) –
- **initonly** (*bool*) –

static getrfznamelist (*nmlfile*, *rfznl*[, *initonly*])
Defined at NameListHandlers.f90 lines 5117-5187

Parameters

- **nmlfile** (*str*) –

- **rfznl** (*Rfznamelisttype*) –
- **initonly** (*bool*) –

static getsrdefectnamelist (*nmlfile*, *srdnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7901-8010

Parameters

- **nmlfile** (*str*) –
- **srdnl** (*Srdefectnamelisttype*) –
- **initonly** (*bool*) –

static getstemdcinamelist (*nmlfile*, *dcinl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7087-7167

Parameters

- **nmlfile** (*str*) –
- **dcinl** (*Stemdcinamelisttype*) –
- **initonly** (*bool*) –

static getstemgeometrynamelist (*nmlfile*, *dcinl*[, *initonly*])
Defined at NameListHandlers.f90 lines 7181-7236

Parameters

- **nmlfile** (*str*) –
- **dcinl** (*Stemgeometrynamelisttype*) –
- **initonly** (*bool*) –

static gettkdindexingnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 6172-6373

Parameters

- **nmlfile** (*str*) –
- **enl** (*Tkdindexingnamelisttype*) –
- **initonly** (*bool*) –

static gettkdmasternamelist (*nmlfile*, *emnl*[, *initonly*])
Defined at NameListHandlers.f90 lines 2737-2790

Parameters

- **nmlfile** (*str*) –
- **emnl** (*Tkdmasternamelisttype*) –
- **initonly** (*bool*) –

static gettkdnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 3951-4077

Parameters

- **nmlfile** (*str*) –
- **enl** (*Tkdnamelisttype*) –
- **initonly** (*bool*) –

static gettkdoverlapnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4219-4289

Parameters

- **nmlfile** (*str*) –
- **enl** (*Tkdoverlapnamelisttype*) –
- **initonly** (*bool*) –

static gettkdspotsnamelist (*nmlfile*, *enl*[, *initonly*])
Defined at NameListHandlers.f90 lines 4304-4390

Parameters

- **nmlfile** (*str*) –
- **enl** (*Tkdspotsnamelisttype*) –
- **initonly** (*bool*) –

static getzadefectnamelist (*nmlfile*, *zadefect*[, *initonly*])
Defined at NameListHandlers.f90 lines 6388-6497

Parameters

- **nmlfile** (*str*) –
- **zadefect** (*Zadefectnamelisttype*) –
- **initonly** (*bool*) –

class pyEMsoft.**Namelisttypedefs** (**args*, ***kwargs*)
Module namelisttypedefs

Defined at NameListTypedefs.f90 lines 42-2068

class pyEMsoft.**Others** (**args*, ***kwargs*)
Module others

Defined at others.f90 lines 49-1151

static ei (*x*)
ei = ei(x)

Defined at others.f90 lines 340-370

Parameters **x** (*float*) –

Returns ei

Return type float

static fcore (*g*, *z*, *accvlt*)
fcore = fcore(g, z, accvlt)

Defined at others.f90 lines 595-665

Parameters

- **g** (*float*) –
- **z** (*int*) –
- **accvlt** (*float*) –

Returns fcore

Return type float

static fphon(*g*, *ul*, *a*, *b*)
fphon = fphon(g, ul, a, b)

Defined at others.f90 lines 186-205

Parameters

- **g** (*float*) –
- **ul** (*float*) –
- **a** (*float array*) –
- **b** (*float array*) –

Returns fphon

Return type float

static fscatt(*g*, *ul*, *z*, *symbol*, *accvlt*, *absflg*, *accflg*, *dwflg*)
fscatt = fscatt(g, ul, z, symbol, accvlt, absflg, accflg, dwflg)

Defined at others.f90 lines 67-161

Parameters

- **g** (*float*) –
- **ul** (*float*) –
- **z** (*int*) –
- **symbol** (*str*) –
- **accvlt** (*float*) –
- **absflg** (*int*) –
- **accflg** (*bool*) –
- **dwflg** (*bool*) –

Returns fscatt

Return type complex

static getwk(*z*, *symbol*, *a*, *b*)
Defined at others.f90 lines 373-592

Parameters

- **z** (*int*) –
- **symbol** (*str*) –
- **a** (*float array*) –
- **b** (*float array*) –

static qsortd(*x*, *ind*, *n*)
Defined at others.f90 lines 1001-1151

Parameters

- **x** (*float array*) –
- **ind** (*int array*) –

- **n** (*int*) –

static ri1 (*bi*, *bj*, *g*)
ri1 = ri1(bi, bj, g)

Defined at others.f90 lines 208-239

Parameters

- **bi** (*float*) –
- **bj** (*float*) –
- **g** (*float*) –

Returns ri1

Return type float

static ri2 (*bi*, *bj*, *g*, *u*)
ri2 = ri2(bi, bj, g, u)

Defined at others.f90 lines 242-292

Parameters

- **bi** (*float*) –
- **bj** (*float*) –
- **g** (*float*) –
- **u** (*float*) –

Returns ri2

Return type float

static rih1 (*x1*, *x2*, *x3*)
rih1 = rih1(x1, x2, x3)

Defined at others.f90 lines 295-311

Parameters

- **x1** (*float*) –
- **x2** (*float*) –
- **x3** (*float*) –

Returns rih1

Return type float

static rih2 (*x*)
rih2 = rih2(x)

Defined at others.f90 lines 314-326

Parameters **x** (*float*) –

Returns rih2

Return type float

static rih3 (*x*)
rih3 = rih3(x)

Defined at others.f90 lines 329-337

Parameters **x** (*float*) –

Returns **rih3**

Return type float

static ssort(*x*, *y*, *n*, *kflag*)
Defined at others.f90 lines 680-999

Parameters

- **x** (*float array*) –
- **y** (*int array*) –
- **n** (*int*) –
- **kflag** (*int*) –

static weko(*a*, *b*, *s*)
weko = weko(a, b, s)

Defined at others.f90 lines 164-183

Parameters

- **a** (*float array*) –
- **b** (*float array*) –
- **s** (*float*) –

Returns **weko**

Return type float

class pyEMsoft.**Patternmod**(**args*, ***kwargs*)
Module patternmod

Defined at patternmod.f90 lines 57-2020

static closeexppatternfile(*inputtype*, *funit*)
Defined at patternmod.f90 lines 1257-1284

Parameters

- **inputtype** (*str*) –
- **funit** (*int*) –

static getemsoftpccoordinates(*pctr*, *vendor*, *delta*, *nx*, *ny*)
emsoftpc = getemsoftpccoordinates(pctr, vendor, delta, nx, ny)

Defined at patternmod.f90 lines 1992-2020

Parameters

- **pctr** (*float array*) –
- **vendor** (*str*) –
- **delta** (*float*) –
- **nx** (*int*) –
- **ny** (*int*) –

Returns **emsoftpc**

Return type float array

static getexppatternrow(*iii*, *wd*, *patsz*, *l*, *dims3*, *offset3*, *funit*, *inputtype*, *hdfstrings*, *exppatarray*[, *roi*, *flipy*])

Defined at patternmod.f90 lines 369-623

Parameters

- **iii** (*int*) –
- **wd** (*int*) –
- **patsz** (*int*) –
- **l** (*int*) –
- **dims3** (*int array*) –
- **offset3** (*int array*) –
- **funit** (*int*) –
- **inputtype** (*str*) –
- **hdfstrings** (*str array*) –
- **exppatarray** (*float array*) –
- **roi** (*int array*) –
- **flipy** (*bool*) –

static getmetadata(*self*, *sem*, *patterndata*, *angles*, *numangles*, *filename*, *inputtype*, *funit*, *hdfstrings*, *istat*[, *verbose*])

Defined at patternmod.f90 lines 883-1242

Parameters

- **enl** (*Ebsdnamelisttype*) –
- **sem** (*Ebsdsemarray*) –
- **patterndata** (*Ebsddipreviewnamelisttype*) –
- **angles** (*Ebsdangletype*) –
- **numangles** (*int*) –
- **filename** (*str*) –
- **inputtype** (*str*) –
- **funit** (*int*) –
- **hdfstrings** (*str array*) –
- **istat** (*int*) –
- **verbose** (*bool*) –

static getsingleexppattern(*iii*, *wd*, *patsz*, *l*, *dims3*, *offset3*, *funit*, *inputtype*, *hdfstrings*, *exppat*)

Defined at patternmod.f90 lines 648-861

Parameters

- **iii** (*int*) –
- **wd** (*int*) –
- **patsz** (*int*) –

- **l** (*int*) –
- **dims3** (*int array*) –
- **offset3** (*int array*) –
- **funit** (*int*) –
- **inputtype** (*str*) –
- **hdfstrings** (*str array*) –
- **exppat** (*float array*) –

static invert_ordering_arrays(*npat*)
Defined at patternmod.f90 lines 142-164

Parameters **npat** (*int*) –

static openexppatternfile(*filename*, *npat*, *l*, *inputtype*, *recsize*, *funit*, *hdfstrings*, *verbose=None*)
istat = openexppatternfile(filename, npat, l, inputtype, recsize, funit, hdfstrings[, verbose])

Defined at patternmod.f90 lines 193-337

Parameters

- **filename** (*str*) –
- **npat** (*int*) –
- **l** (*int*) –
- **inputtype** (*str*) –
- **recsize** (*int*) –
- **funit** (*int*) –
- **hdfstrings** (*str array*) –
- **verbose** (*bool*) –

Returns **istat**

Return type int

static patternmod_errormessage(*istat*)
Defined at patternmod.f90 lines 1298-1310

Parameters **istat** (*int*) –

static preprocesspatterns(*nthreads*, *inram*, *ebsdnl*, *binx*, *biny*, *masklin*, *correctsize*, *totnumexpt*[, *epatterns*, *exptiq*])
Defined at patternmod.f90 lines 1347-1709

Parameters

- **nthreads** (*int*) –
- **inram** (*bool*) –
- **ebsdnl** (*Ebsdindexingnamelisttype*) –
- **binx** (*int*) –
- **biny** (*int*) –
- **masklin** (*float array*) –

- **correctsize** (*int*) –
- **totnumexpt** (*int*) –
- **epatterns** (*float array*) –
- **exptiq** (*float array*) – define a bunch of mostly integer parameters for the input patterns we have the following parameters (P for experimental Pattern)

static preprocesstkdpatterns (*nthreads*, *inram*, *tkdnl*, *binx*, *biny*, *masklin*, *correctsize*, *totnumexpt*[, *epatterns*, *exptiq*])

Defined at patternmod.f90 lines 1737-1974

Parameters

- **nthreads** (*int*) –
- **inram** (*bool*) –
- **tkdnl** (*Tkdindexingnamelisttype*) –
- **binx** (*int*) –
- **biny** (*int*) –
- **masklin** (*float array*) –
- **correctsize** (*int*) –
- **totnumexpt** (*int*) –
- **epatterns** (*float array*) –
- **exptiq** (*float array*) – define a bunch of mostly integer parameters

class pyEMsoft.**Quaternions** (**args*, ***kwargs*)

Module quaternions

Defined at quaternions.f90 lines 127-659

static cabs (**args*, ***kwargs*)

Defined at quaternions.f90 lines 152-154

Overloaded interface containing the following procedures: _quat_norm _quat_norm_d

static conjg (**args*, ***kwargs*)

Defined at quaternions.f90 lines 145-147

Overloaded interface containing the following procedures: _quat_conjg _quat_conjg_d

static quat_angle (**args*, ***kwargs*)

Defined at quaternions.f90 lines 170-172

Overloaded interface containing the following procedures: _quat_angle _quat_angle_d

static quat_div (**args*, ***kwargs*)

Defined at quaternions.f90 lines 158-160

Overloaded interface containing the following procedures: _quat_div _quat_div_d

static quat_innerproduct (**args*, ***kwargs*)

Defined at quaternions.f90 lines 164-166

Overloaded interface containing the following procedures: _quat_innerproduct _quat_innerproduct_d

static quat_lp (**args*, ***kwargs*)

Defined at quaternions.f90 lines 176-178

Overloaded interface containing the following procedures: _quat_lp _quat_lp_d

static quat_marsaglia(*self*)
q = quat_marsaglia(self)

Defined at quaternions.f90 lines 630-659

Parameters **seed** (*Rng_T*) –

Returns **q**

Return type float array

static quat_mult(**args*, ***kwargs*)
Defined at quaternions.f90 lines 138-140

Overloaded interface containing the following procedures: _quat_mult _quat_mult_d

static quat_slerp(**args*, ***kwargs*)
Defined at quaternions.f90 lines 182-184

Overloaded interface containing the following procedures: _quat_slerp _quat_slerp_d

static quaternion_print(**args*, ***kwargs*)
Defined at quaternions.f90 lines 132-134

Overloaded interface containing the following procedures: _quaternion_print _quaternion_print_d

class pyEMsoft.**Rng**(**args*, ***kwargs*)
Module rng

Defined at rng.f90 lines 28-64

static rng_seed(*self*, *seed*)
Defined at rng.f90 lines 44-49

Parameters

- **self** (*Rng_T*) –
- **seed** (*int*) –

static rng_uniform(*self*)
u = rng_uniform(self)

Defined at rng.f90 lines 52-64

Parameters **self** (*Rng_T*) –

Returns **u**

Return type float

class pyEMsoft.**Rotations**(**args*, ***kwargs*)
Module rotations

Defined at rotations.f90 lines 114-6274

static ax2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 313-315

Overloaded interface containing the following procedures: _ax2cu _ax2cu_d

static ax2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 283-285

Overloaded interface containing the following procedures: _ax2eu _ax2eu_d

static ax2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 307-309

Overloaded interface containing the following procedures: _ax2ho _ax2ho_d

static ax2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 289-291

Overloaded interface containing the following procedures: _ax2om _ax2om_d

static ax2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 301-303

Overloaded interface containing the following procedures: _ax2qu _ax2qu_d

static ax2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 295-297

Overloaded interface containing the following procedures: _ax2ro _ax2ro_d

static ax2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 325-327

Overloaded interface containing the following procedures: _ax2rv _ax2rv_d

static ax2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 319-321

Overloaded interface containing the following procedures: _ax2st _ax2st_d

static ax_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 147-149

Overloaded interface containing the following procedures: _ax_check _ax_check_d

static cu2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 491-493

Overloaded interface containing the following procedures: _cu2ax _cu2ax_d

static cu2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 479-481

Overloaded interface containing the following procedures: _cu2eu _cu2eu_d

static cu2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 509-511

Overloaded interface containing the following procedures: _cu2ho _cu2ho_d

static cu2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 485-487

Overloaded interface containing the following procedures: _cu2om _cu2om_d

static cu2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 503-505

Overloaded interface containing the following procedures: _cu2qu _cu2qu_d

static cu2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 497-499

Overloaded interface containing the following procedures: _cu2ro _cu2ro_d

static cu2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 521-523

Overloaded interface containing the following procedures: _cu2rv _cu2rv_d

static cu2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 515-517

Overloaded interface containing the following procedures: _cu2st _cu2st_d

static cu_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 137-139

Overloaded interface containing the following procedures: _cu_check _cu_check_d

static eu2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 191-193

Overloaded interface containing the following procedures: _eu2ax _eu2ax_d

static eu2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 215-217

Overloaded interface containing the following procedures: _eu2cu _eu2cu_d

static eu2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 209-211

Overloaded interface containing the following procedures: _eu2ho _eu2ho_d

static eu2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 185-187

Overloaded interface containing the following procedures: _eu2om _eu2om_d

static eu2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 203-205

Overloaded interface containing the following procedures: _eu2qu _eu2qu_d

static eu2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 197-199

Overloaded interface containing the following procedures: _eu2ro _eu2ro_d

static eu2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 227-229

Overloaded interface containing the following procedures: _eu2rv _eu2rv_d

static eu2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 221-223

Overloaded interface containing the following procedures: _eu2st _eu2st_d

static eu_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 122-124

Overloaded interface containing the following procedures: _eu_check _eu_check_d

static genrot(**args*, ***kwargs*)
Defined at rotations.f90 lines 170-172

Overloaded interface containing the following procedures: _genrot _genrot_d

static ho2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 442-444

Overloaded interface containing the following procedures: _ho2ax _ho2ax_d

static ho2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 460-462

Overloaded interface containing the following procedures: _ho2cu _ho2cu_d

static ho2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 430-432

Overloaded interface containing the following procedures: _ho2eu _ho2eu_d

static ho2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 436-438

Overloaded interface containing the following procedures: _ho2om _ho2om_d

static ho2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 454-456

Overloaded interface containing the following procedures: _ho2qu _ho2qu_d

static ho2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 448-450

Overloaded interface containing the following procedures: _ho2ro _ho2ro_d

static ho2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 472-474

Overloaded interface containing the following procedures: _ho2rv _ho2rv_d

static ho2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 466-468

Overloaded interface containing the following procedures: _ho2st _ho2st_d

static ho_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 132-134

Overloaded interface containing the following procedures: _ho_check _ho_check_d

static init_orientation(**args*, ***kwargs*)
Defined at rotations.f90 lines 176-180

Overloaded interface containing the following procedures: _init_orientation _init_orientation_om _init_orientation_d _init_orientation_om_d

static om2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 240-242

Overloaded interface containing the following procedures: _om2ax _om2ax_d

static om2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 264-266

Overloaded interface containing the following procedures: _om2cu _om2cu_d

static om2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 234-236

Overloaded interface containing the following procedures: _om2eu _om2eu_d

static om2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 258-260

Overloaded interface containing the following procedures: _om2ho _om2ho_d

static om2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 252-254

Overloaded interface containing the following procedures: _om2qu _om2qu_d

static om2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 246-248

Overloaded interface containing the following procedures: _om2ro _om2ro_d

static om2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 276-278

Overloaded interface containing the following procedures: _om2rv _om2rv_d

static om2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 270-272

Overloaded interface containing the following procedures: _om2st _om2st_d

static om_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 152-154

Overloaded interface containing the following procedures: _om_check _om_check_d

static print_orientation(**args*, ***kwargs*)
Defined at rotations.f90 lines 651-653

Overloaded interface containing the following procedures: _print_orientation _print_orientation_d

static qu2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 393-395

Overloaded interface containing the following procedures: _qu2ax _qu2ax_d

static qu2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 411-413

Overloaded interface containing the following procedures: _qu2cu _qu2cu_d

static qu2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 381-383

Overloaded interface containing the following procedures: _qu2eu _qu2eu_d

static qu2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 405-407

Overloaded interface containing the following procedures: _qu2ho _qu2ho_d

static qu2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 387-389

Overloaded interface containing the following procedures: _qu2om _qu2om_d

static qu2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 399-401

Overloaded interface containing the following procedures: _qu2ro _qu2ro_d

static qu2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 423-425

Overloaded interface containing the following procedures: _qu2rv _qu2rv_d

static qu2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 417-419

Overloaded interface containing the following procedures: _qu2st _qu2st_d

static qu_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 142-144

Overloaded interface containing the following procedures: _qu_check _qu_check_d

static quat_average(**args*, ***kwargs*)
Defined at rotations.f90 lines 644-646

Overloaded interface containing the following procedures: _quat_average _quat_average_d

static ro2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 344-346

Overloaded interface containing the following procedures: _ro2ax _ro2ax_d

static ro2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 362-364

Overloaded interface containing the following procedures: _ro2cu _ro2cu_d

static ro2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 332-334

Overloaded interface containing the following procedures: _ro2eu _ro2eu_d

static ro2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 356-358

Overloaded interface containing the following procedures: _ro2ho _ro2ho_d

static ro2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 338-340

Overloaded interface containing the following procedures: _ro2om _ro2om_d

static ro2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 350-352

Overloaded interface containing the following procedures: _ro2qu _ro2qu_d

static ro2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 374-376

Overloaded interface containing the following procedures: _ro2rv _ro2rv_d

static ro2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 368-370

Overloaded interface containing the following procedures: _ro2st _ro2st_d

static ro_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 127-129

Overloaded interface containing the following procedures: _ro_check _ro_check_d

static rodriguesproduct(**args*, ***kwargs*)
Defined at rotations.f90 lines 626-628

Overloaded interface containing the following procedures: _rodriguesproduct _rodriguesproduct_d

static rotatetensor2(**args*, ***kwargs*)
Defined at rotations.f90 lines 638-640

Overloaded interface containing the following procedures: _rottensor2_om _rottensor2_om_d

static rotatevector(**args*, ***kwargs*)
Defined at rotations.f90 lines 632-634

Overloaded interface containing the following procedures: _rotvec_om _rotvec_om_d

static rv2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 577-579

Overloaded interface containing the following procedures: _rv2ax _rv2ax_d

static rv2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 607-609

Overloaded interface containing the following procedures: _rv2cu _rv2cu_d

static rv2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 589-591

Overloaded interface containing the following procedures: _rv2eu _rv2eu_d

static rv2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 619-621

Overloaded interface containing the following procedures: _rv2ho _rv2ho_d

static rv2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 601-603

Overloaded interface containing the following procedures: _rv2om _rv2om_d

static rv2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 583-585

Overloaded interface containing the following procedures: _rv2qu _rv2qu_d

static rv2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 595-597

Overloaded interface containing the following procedures: _rv2ro _rv2ro_d

static rv2st(**args*, ***kwargs*)
Defined at rotations.f90 lines 613-615

Overloaded interface containing the following procedures: _rv2st _rv2st_d

static rv_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 162-164

Overloaded interface containing the following procedures: _rv_check _rv_check_d

static st2ax(**args*, ***kwargs*)
Defined at rotations.f90 lines 540-542

Overloaded interface containing the following procedures: _st2ax _st2ax_d

static st2cu(**args*, ***kwargs*)
Defined at rotations.f90 lines 564-566

Overloaded interface containing the following procedures: _st2cu _st2cu_d

static st2eu(**args*, ***kwargs*)
Defined at rotations.f90 lines 528-530

Overloaded interface containing the following procedures: _st2eu _st2eu_d

static st2ho(**args*, ***kwargs*)
Defined at rotations.f90 lines 558-560

Overloaded interface containing the following procedures: _st2ho _st2ho_d

static st2om(**args*, ***kwargs*)
Defined at rotations.f90 lines 534-536

Overloaded interface containing the following procedures: _st2om _st2om_d

static st2qu(**args*, ***kwargs*)
Defined at rotations.f90 lines 552-554

Overloaded interface containing the following procedures: _st2qu _st2qu_d

static st2ro(**args*, ***kwargs*)
Defined at rotations.f90 lines 546-548

Overloaded interface containing the following procedures: _st2ro _st2ro_d

static st2rv(**args*, ***kwargs*)
Defined at rotations.f90 lines 570-572

Overloaded interface containing the following procedures: _st2rv _st2rv_d

static st_check(**args*, ***kwargs*)
Defined at rotations.f90 lines 157-159

Overloaded interface containing the following procedures: _st_check _st_check_d

class pyEMsoft.**So3**(**args*, ***kwargs*)
Module so3

Defined at so3.f90 lines 48-1817

static cubochoricneighbors(*cubneighbor*, *nn*, *cub*, *stepsize*)
Defined at so3.f90 lines 656-683

Parameters

- **cubneighbor** (*float array*) –
- **nn** (*int*) –
- **cub** (*float array*) –
- **stepsize** (*float*) –

static getfztypeandorder(*pgnum1*, *pgnum2=None*)
fztype, fzorder = getfztypeandorder(pgnum1[, pgnum2])

Defined at so3.f90 lines 86-148

Parameters

- **pgnum1** (*int*) –
- **pgnum2** (*int*) –

Returns

- **fztype** (*int*)
- **fzorder** (*int*)

static getmackenziedistribution(*pgnum*, *nmisor*, *misor*, *mk*)
Defined at so3.f90 lines 1715-1817

Parameters

- **pgnum** (*int*) –
- **nmisor** (*int*) –
- **misor** (*float array*) –
- **mk** (*float array*) –

static getvertex(*order*, *vertex*)
Defined at so3.f90 lines 1552-1563

Parameters

- **order** (*int*) –
- **vertex** (*float array*) –

static insidecubehexfz(*rod*)
res = insidecubehexfz(rod)

Defined at so3.f90 lines 431-447

Parameters **rod** (*float array*) –

Returns **res**

Return type bool

static insidecubicfz(*rod*, *ot*)
res = insidecubicfz(rod, ot)

Defined at so3.f90 lines 392-413

Parameters

- **rod** (*float array*) –
- **ot** (*str*) –

Returns **res**

Return type bool

static insidecubicmfz(*rod*, *ot*)
res = insidecubicmfz(rod, ot)

Defined at so3.f90 lines 1468-1492

Parameters

- **rod** (*float array*) –
- **ot** (*str*) –

Returns **res**

Return type bool

static insidecyclicfz(*rod*, *fztype*, *fzorder*)
res = insidecyclicfz(rod, fztype, fzorder)

Defined at so3.f90 lines 268-292

Parameters

- **rod** (*float array*) –
- **fztype** (*int*) –
- **fzorder** (*int*) –

Returns **res**

Return type bool

static insidedihedralfz(*rod*, *order*)
res = insidedihedralfz(rod, order)

Defined at so3.f90 lines 312-374

Parameters

- **rod** (*float array*) –
- **order** (*int*) –

Returns res

Return type bool

static insidedihedralmfz(*rod*, *mfzorder*)
res = insidedihedralmfz(rod, mfzorder)

Defined at so3.f90 lines 1508-1537

Parameters

- **rod** (*float array*) –
- **mfzorder** (*int*) –

Returns res

Return type bool

static insideicosahedralfz(*rod*)
res = insideicosahedralfz(rod)

Defined at so3.f90 lines 236-250

Parameters **rod** (*float array*) –

Returns res

Return type bool

static isinsidefz(*rod*, *fztype*, *fzorder*, *qfz=None*)
insidefz = isinsidefz(rod, fztype, fzorder[, qfz])

Defined at so3.f90 lines 179-222

Parameters

- **rod** (*float array*) –
- **fztype** (*int*) –
- **fzorder** (*int*) –
- **qfz** (*float array*) –

Returns insidefz

Return type bool

static isinsidemfz(*rod*, *mfztype*, *mfzorder*)
insidemfz = isinsidemfz(rod, mfztype, mfzorder)

Defined at so3.f90 lines 1433-1453

Parameters

- **rod** (*float array*) –
- **mfztype** (*int*) –

- **mfzorder** (*int*) –

Returns insidemfz

Return type bool

static mkcc(*a*, *b*, *c*)
cc = mkcc(a, b, c)

Defined at so3.f90 lines 1662-1670

Parameters

- **a** (*float*) –
- **b** (*float*) –
- **c** (*float*) –

Returns cc

Return type float

static mks2(*a*, *b*, *c*)
s2 = mks2(a, b, c)

Defined at so3.f90 lines 1688-1696

Parameters

- **a** (*float*) –
- **b** (*float*) –
- **c** (*float*) –

Returns s2

Return type float

static pnpoly(*px*, *py*, *xx*, *yy*, *n*)
inout = pnpoly(px, py, xx, yy, n)

Defined at so3.f90 lines 1611-1644

Parameters

- **px** (*float*) –
- **py** (*float*) –
- **xx** (*float array*) –
- **yy** (*float array*) –
- **n** (*int*) –

Returns inout

Return type int

class pyEMsoft.**Stringconstants**(**args*, ***kwargs*)
Module stringconstants

Defined at stringconstants.f90 lines 46-1318

sc_accume
Element sc_accume ftype=character(7) pytype=str

Defined at stringconstants.f90 line 510

sc_accumedetector
Element sc_accumedetector ftype=character(14) pytype=str

Defined at stringconstants.f90 line 520

sc_accumenh
Element sc_accumenh ftype=character(10) pytype=str

Defined at stringconstants.f90 line 514

sc_accumesh
Element sc_accumesh ftype=character(10) pytype=str

Defined at stringconstants.f90 line 516

sc_accumesp
Element sc_accumesp ftype=character(10) pytype=str

Defined at stringconstants.f90 line 518

sc_accumsp
Element sc_accumsp ftype=character(7) pytype=str

Defined at stringconstants.f90 line 512

sc_accumt
Element sc_accumt ftype=character(7) pytype=str

Defined at stringconstants.f90 line 522

sc_accumt1
Element sc_accumt1 ftype=character(8) pytype=str

Defined at stringconstants.f90 line 524

sc_accumt2
Element sc_accumt2 ftype=character(8) pytype=str

Defined at stringconstants.f90 line 526

sc_accumt3
Element sc_accumt3 ftype=character(8) pytype=str

Defined at stringconstants.f90 line 528

sc_accumxyz
Element sc_accumxyz ftype=character(9) pytype=str

Defined at stringconstants.f90 line 530

sc_accumz
Element sc_accumz ftype=character(7) pytype=str

Defined at stringconstants.f90 line 532

sc_accumznh
Element sc_accumznh ftype=character(10) pytype=str

Defined at stringconstants.f90 line 534

sc_accumzsh
Element sc_accumzsh ftype=character(10) pytype=str

Defined at stringconstants.f90 line 536

sc_angab
Element sc_angab ftype=character(5) pytype=str

Defined at stringconstants.f90 line 538

sc_angfile
Element sc_angfile ftype=character(7) pytype=str

Defined at stringconstants.f90 line 540

sc_anglefile
Element sc_anglefile ftype=character(9) pytype=str

Defined at stringconstants.f90 line 542

sc_anglefiletype
Element sc_anglefiletype ftype=character(13) pytype=str

Defined at stringconstants.f90 line 544

sc_aperturepositon
Element sc_aperturepositon ftype=character(16) pytype=str

Defined at stringconstants.f90 line 546

sc_applydeformation
Element sc_applydeformation ftype=character(16) pytype=str

Defined at stringconstants.f90 line 548

sc_astigmatism
Element sc_astigmatism ftype=character(11) pytype=str

Defined at stringconstants.f90 line 552

sc_atomdata
Element sc_atomdata ftype=character(8) pytype=str

Defined at stringconstants.f90 line 144

sc_atomtypes
Element sc_atomtypes ftype=character(9) pytype=str

Defined at stringconstants.f90 line 146

sc_avctffile
Element sc_avctffile ftype=character(9) pytype=str

Defined at stringconstants.f90 line 554

sc_avdotproductmap
Element sc_avdotproductmap ftype=character(15) pytype=str

Defined at stringconstants.f90 line 148

sc_averageorientations
Element sc_averageorientations ftype=character(19) pytype=str

Defined at stringconstants.f90 line 150

sc_axialsymmetry
Element sc_axialsymmetry ftype=character(14) pytype=str

Defined at stringconstants.f90 line 51

sc_axisangle
Element sc_axisangle ftype=character(9) pytype=str

Defined at stringconstants.f90 line 556

sc_axoutname
Element sc_axoutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 558

sc_beamcurrent
Element sc_beamcurrent ftype=character(11) pytype=str

Defined at stringconstants.f90 line 560

sc_beamdc
Element sc_beamdc ftype=character(6) pytype=str

Defined at stringconstants.f90 line 562

sc_bethelist
Element sc_bethelist ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1097

sc_betheparameters
Element sc_betheparameters ftype=character(15) pytype=str

Defined at stringconstants.f90 line 154

sc_betheparametersfile
Element sc_betheparametersfile ftype=character(19) pytype=str

Defined at stringconstants.f90 line 156

sc_bfimage
Element sc_bfimage ftype=character(7) pytype=str

Defined at stringconstants.f90 line 152

sc_binning
Element sc_binning ftype=character(7) pytype=str

Defined at stringconstants.f90 line 564

sc_binx
Element sc_binx ftype=character(4) pytype=str

Defined at stringconstants.f90 line 566

sc_biny
Element sc_biny ftype=character(4) pytype=str

Defined at stringconstants.f90 line 568

sc_bitdepth
Element sc_bitdepth ftype=character(8) pytype=str

Defined at stringconstants.f90 line 570

sc_braggangle
Element sc_braggangle ftype=character(10) pytype=str

Defined at stringconstants.f90 line 158

sc_braggga
Element sc_braggga ftype=character(8) pytype=str

Defined at stringconstants.f90 line 160

sc_brightfield
Element sc_brightfield ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1099

sc_bse1
Element sc_bse1 ftype=character(4) pytype=str

Defined at stringconstants.f90 line 572

sc_bzero
Element sc_bzero ftype=character(5) pytype=str

Defined at stringconstants.f90 line 162

sc_c1
Element sc_c1 ftype=character(2) pytype=str

Defined at stringconstants.f90 line 576

sc_c2
Element sc_c2 ftype=character(2) pytype=str

Defined at stringconstants.f90 line 578

sc_c3
Element sc_c3 ftype=character(2) pytype=str

Defined at stringconstants.f90 line 580

sc_cameraazimuthalangle
Element sc_cameraazimuthalangle ftype=character(22) pytype=str

Defined at stringconstants.f90 line 174

sc_cameraelevationangle
Element sc_cameraelevationangle ftype=character(22) pytype=str

Defined at stringconstants.f90 line 176

sc_cbednamelist
Element sc_cbednamelist ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1101

sc_cbedqc
Element sc_cbedqc ftype=character(6) pytype=str

Defined at stringconstants.f90 line 164

sc_cbedqcnamelist
Element sc_cbedqcnamelist ftype=character(14) pytype=str

Defined at stringconstants.f90 line 166

sc_cellatomtype
Element sc_cellatomtype ftype=character(14) pytype=str

Defined at stringconstants.f90 line 574

sc_centering
Element sc_centering ftype=character(9) pytype=str

Defined at stringconstants.f90 line 582

sc_char1d
Element sc_char1d ftype=character(6) pytype=str

Defined at stringconstants.f90 line 584

sc_char2d
Element sc_char2d ftype=character(6) pytype=str

Defined at stringconstants.f90 line 586

sc_char3d
Element sc_char3d ftype=character(6) pytype=str

Defined at stringconstants.f90 line 588

sc_char4d
Element sc_char4d ftype=character(6) pytype=str

Defined at stringconstants.f90 line 590

sc_chararray2d
Element sc_chararray2d ftype=character(11) pytype=str

Defined at stringconstants.f90 line 592

sc_ci
Element sc_ci ftype=character(2) pytype=str

Defined at stringconstants.f90 line 168

sc_cimap
Element sc_cimap ftype=character(5) pytype=str

Defined at stringconstants.f90 line 170

sc_clarray
Element sc_clarray ftype=character(7) pytype=str

Defined at stringconstants.f90 line 172

sc_colormapfile
Element sc_colormapfile ftype=character(12) pytype=str

Defined at stringconstants.f90 line 594

sc_combinesites
Element sc_combinesites ftype=character(12) pytype=str

Defined at stringconstants.f90 line 596

sc_compgridtype
Element sc_compgridtype ftype=character(12) pytype=str

Defined at stringconstants.f90 line 598

sc_compmode
Element sc_compmode ftype=character(8) pytype=str

Defined at stringconstants.f90 line 600

sc_compnsteps
Element sc_compnsteps ftype=character(10) pytype=str

Defined at stringconstants.f90 line 602

sc_config
Element sc_config ftype=character(7) pytype=str

Defined at stringconstants.f90 line 63

sc_configpath
Element sc_configpath ftype=character(10) pytype=str

Defined at stringconstants.f90 line 87

sc_convergenceangle
Element sc_convergenceangle ftype=character(16) pytype=str

Defined at stringconstants.f90 line 1293

sc_coordinatesystem
Element sc_coordinatesystem ftype=character(16) pytype=str

Defined at stringconstants.f90 line 1103

sc_copyfromenergyfile
Element sc_copyfromenergyfile ftype=character(18) pytype=str

Defined at stringconstants.f90 line 658

sc_correctedeulerangles
Element sc_correctedeulerangles ftype=character(20) pytype=str

Defined at stringconstants.f90 line 178

sc_cplm
Element sc_cplm ftype=character(4) pytype=str

Defined at stringconstants.f90 line 1219

sc_cplmimages
Element sc_cplmimages ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1221

sc_cplmintensities
Element sc_cplmintensities ftype=character(15) pytype=str

Defined at stringconstants.f90 line 1223

sc_cplmmaster
Element sc_cplmmaster ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1229

sc_cplmmasterlpnh
Element sc_cplmmasterlpnh ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1233

sc_cplmmasternamelist
Element sc_cplmmasternamelist ftype=character(18) pytype=str

Defined at stringconstants.f90 line 1225

sc_cplmmasternml
Element sc_cplmmasternml ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1231

sc_cplmmasterspnh
Element sc_cplmmasterspnh ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1235

sc_cplmnamelist
Element sc_cplmnamelist ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1227

sc_creationdate
Element sc_creationdate ftype=character(12) pytype=str

Defined at stringconstants.f90 line 180

sc_creationtime
Element sc_creationtime ftype=character(12) pytype=str

Defined at stringconstants.f90 line 182

sc_creator
Element sc_creator ftype=character(7) pytype=str

Defined at stringconstants.f90 line 184

sc_crystaldata
Element sc_crystaldata ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1105

sc_crystalsystem
Element sc_crystalsystem ftype=character(13) pytype=str

Defined at stringconstants.f90 line 186

sc_ctffile
Element sc_ctffile ftype=character(7) pytype=str

Defined at stringconstants.f90 line 604

sc_cubochoricstepsize
Element sc_cubochoricstepsize ftype=character(18) pytype=str

Defined at stringconstants.f90 line 188

sc_cuoutname
Element sc_cuoutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 606

sc_d2or3
Element sc_d2or3 ftype=character(5) pytype=str

Defined at stringconstants.f90 line 204

sc_data
Element sc_data ftype=character(4) pytype=str

Defined at stringconstants.f90 line 1107

sc_datafile
Element sc_datafile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 608

sc_dataname
Element sc_dataname ftype=character(8) pytype=str

Defined at stringconstants.f90 line 610

sc_datasetname
Element sc_datasetname ftype=character(11) pytype=str

Defined at stringconstants.f90 line 614

sc_date
Element sc_date ftype=character(4) pytype=str

Defined at stringconstants.f90 line 190

sc_defectfilename
Element sc_defectfilename ftype=character(14) pytype=str

Defined at stringconstants.f90 line 616

sc_defecttest
Element sc_defecttest ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1279

sc_deformationfile
Element sc_deformationfile ftype=character(15) pytype=str

Defined at stringconstants.f90 line 618

sc_delta
Element sc_delta ftype=character(5) pytype=str

Defined at stringconstants.f90 line 620

sc_depthmax
Element sc_depthmax ftype=character(8) pytype=str

Defined at stringconstants.f90 line 622

sc_depthstep
Element sc_depthstep ftype=character(9) pytype=str

Defined at stringconstants.f90 line 624

sc_detparms
Element sc_detparms ftype=character(8) pytype=str

Defined at stringconstants.f90 line 192

sc_detparmscan
Element sc_detparmscan ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1109

sc_detparmstepsize
Element sc_detparmstepsize ftype=character(15) pytype=str

Defined at stringconstants.f90 line 194

sc_develop
Element sc_develop ftype=character(7) pytype=str

Defined at stringconstants.f90 line 115

sc_devid
Element sc_devid ftype=character(5) pytype=str

Defined at stringconstants.f90 line 626

sc_dictfile
Element sc_dictfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 628

sc_dictindxopenclnamelist
Element sc_dictindxopenclnamelist ftype=character(22) pytype=str

Defined at stringconstants.f90 line 1111

sc_dictionaryeulerangles
Element sc_dictionaryeulerangles ftype=character(21) pytype=str

Defined at stringconstants.f90 line 274

sc_diffractionpattern
Element sc_diffractionpattern ftype=character(18) pytype=str

Defined at stringconstants.f90 line 196

sc_discsize
Element sc_discsize ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1291

sc_dismaxvals
Element sc_dismaxvals ftype=character(10) pytype=str

Defined at stringconstants.f90 line 630

sc_disminvals
Element sc_disminvals ftype=character(10) pytype=str

Defined at stringconstants.f90 line 632

sc_dispfield
Element sc_dispfield ftype=character(9) pytype=str

Defined at stringconstants.f90 line 198

sc_dispfile
Element sc_dispfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 634

sc_displacementfield
Element sc_displacementfield ftype=character(2) pytype=str

Defined at stringconstants.f90 line 1113

sc_dispmode
Element sc_dispmode ftype=character(8) pytype=str

Defined at stringconstants.f90 line 636

sc_dispx
Element sc_dispx ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1115

sc_dispy
Element sc_dispy ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1117

sc_disvals
Element sc_disvals ftype=character(7) pytype=str

Defined at stringconstants.f90 line 638

sc_dmin
Element sc_dmin ftype=character(4) pytype=str

Defined at stringconstants.f90 line 640

sc_dotproducts
Element sc_dotproducts ftype=character(11) pytype=str

Defined at stringconstants.f90 line 200

sc_double1d
Element sc_double1d ftype=character(8) pytype=str

Defined at stringconstants.f90 line 644

sc_double2d
Element sc_double2d ftype=character(8) pytype=str

Defined at stringconstants.f90 line 646

sc_double3d
Element sc_double3d ftype=character(8) pytype=str

Defined at stringconstants.f90 line 648

sc_double4d
Element sc_double4d ftype=character(8) pytype=str

Defined at stringconstants.f90 line 650

sc_doubletype
Element sc_doubletype ftype=character(6) pytype=str

Defined at stringconstants.f90 line 642

sc_dt
Element sc_dt ftype=character(2) pytype=str

Defined at stringconstants.f90 line 652

sc_duration
Element sc_duration ftype=character(8) pytype=str

Defined at stringconstants.f90 line 202

sc_dwelltime
Element sc_dwelltime ftype=character(9) pytype=str

Defined at stringconstants.f90 line 654

sc_dwflag
Element sc_dwflag ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1301

sc_ebinsize
Element sc_ebinsize ftype=character(8) pytype=str

Defined at stringconstants.f90 line 264

sc_ebsd
Element sc_ebsd ftype=character(4) pytype=str

Defined at stringconstants.f90 line 1119

sc_ebsdclusternamelist
Element sc_ebsdclusternamelist ftype=character(19) pytype=str

Defined at stringconstants.f90 line 1135

sc_ebsddefectmaster
Element sc_ebsddefectmaster ftype=character(16) pytype=str

Defined at stringconstants.f90 line 1139

sc_ebsddefectnamelist
Element sc_ebsddefectnamelist ftype=character(18) pytype=str

Defined at stringconstants.f90 line 252

sc_ebsdenergy
Element sc_ebsdenergy ftype=character(10) pytype=str

Defined at stringconstants.f90 line 216

sc_ebsdindexingnamelisttype
Element sc_ebsdindexingnamelisttype ftype=character(24) pytype=str

Defined at stringconstants.f90 line 1121

sc_ebsdmaster
Element sc_ebsdmaster ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1137

sc_ebsdmasterjson
Element sc_ebsdmasterjson ftype=character(14) pytype=str

Defined at stringconstants.f90 line 210

sc_ebsdmasternamelist
Element sc_ebsdmasternamelist ftype=character(18) pytype=str

Defined at stringconstants.f90 line 1129

sc_ebsdmasternml
Element sc_ebsdmasternml ftype=character(13) pytype=str

Defined at stringconstants.f90 line 212

sc_ebsdmasteropenclnamelist
Element sc_ebsdmasteropenclnamelist ftype=character(24) pytype=str

Defined at stringconstants.f90 line 1131

sc_ebsdmastervars
Element sc_ebsdmastervars ftype=character(14) pytype=str

Defined at stringconstants.f90 line 214

sc_ebsdnamelist
Element sc_ebsdnamelist ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1133

sc_ebsdoverlapnamelist
Element sc_ebsdoverlapnamelist ftype=character(19) pytype=str

Defined at stringconstants.f90 line 220

sc_ebsdoverlapnml
Element sc_ebsdoverlapnml ftype=character(14) pytype=str

Defined at stringconstants.f90 line 218

sc_ebsdpatterns
Element sc_ebsdpatterns ftype=character(12) pytype=str

Defined at stringconstants.f90 line 206

sc_ebsdscannamelist
Element sc_ebsdscannamelist ftype=character(16) pytype=str

Defined at stringconstants.f90 line 1143

sc_ebsdsinglemasternamelist
Element sc_ebsdsinglemasternamelist ftype=character(24) pytype=str

Defined at stringconstants.f90 line 1141

sc_ebsdviewreferenceframe
Element sc_ebsdviewreferenceframe ftype=character(22) pytype=str

Defined at stringconstants.f90 line 208

sc_ecci
Element sc_ecci ftype=character(4) pytype=str

Defined at stringconstants.f90 line 1145

sc_eccidefectjson
Element sc_eccidefectjson ftype=character(14) pytype=str

Defined at stringconstants.f90 line 222

sc_eccifoiljson
Element sc_eccifoiljson ftype=character(12) pytype=str

Defined at stringconstants.f90 line 224

sc_ecciimages
Element sc_ecciimages ftype=character(10) pytype=str

Defined at stringconstants.f90 line 226

sc_eccimasternml
Element sc_eccimasternml ftype=character(13) pytype=str

Defined at stringconstants.f90 line 228

sc_eccinamelist
Element sc_eccinamelist ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1147

sc_ecp
Element sc_ecp ftype=character(3) pytype=str

Defined at stringconstants.f90 line 230

sc_ecpatterns
Element sc_ecpatterns ftype=character(10) pytype=str

Defined at stringconstants.f90 line 242

sc_ecpindexingnamelisttype
Element sc_ecpindexingnamelisttype ftype=character(23) pytype=str

Defined at stringconstants.f90 line 1149

sc_ecpkinematic
Element sc_ecpkinematic ftype=character(12) pytype=str

Defined at stringconstants.f90 line 232

sc_ecpmaster
Element sc_ecpmaster ftype=character(9) pytype=str

Defined at stringconstants.f90 line 236

sc_ecpmasternamelist
Element sc_ecpmasternamelist ftype=character(17) pytype=str

Defined at stringconstants.f90 line 1151

sc_ecpmasternml
Element sc_ecpmasternml ftype=character(12) pytype=str

Defined at stringconstants.f90 line 238

sc_ecpname
Element sc_ecpname ftype=character(7) pytype=str

Defined at stringconstants.f90 line 240

sc_ecpnamelist
Element sc_ecpnamelist ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1153

sc_ecppatternnamelist
Element sc_ecppatternnamelist ftype=character(18) pytype=str

Defined at stringconstants.f90 line 1159

sc_ecpsingle
Element sc_ecpsingle ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1161

sc_ecpza
Element sc_ecpza ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1155

sc_ecpzanamelist
Element sc_ecpzanamelist ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1157

sc_ecpzanml
Element sc_ecpzanml ftype=character(8) pytype=str

Defined at stringconstants.f90 line 234

sc_eecmaster
Element sc_eecmaster ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1123

sc_eecmasternamelist
Element sc_eecmasternamelist ftype=character(17) pytype=str

Defined at stringconstants.f90 line 1127

sc_eecmasternml
Element sc_eecmasternml ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1125

sc_ehistmin
Element sc_ehistmin ftype=character(8) pytype=str

Defined at stringconstants.f90 line 266

sc_ekev
Element sc_ekev ftype=character(4) pytype=str

Defined at stringconstants.f90 line 268

sc_ekevs
Element sc_ekevs ftype=character(5) pytype=str

Defined at stringconstants.f90 line 270

sc_emdata
Element sc_emdata ftype=character(6) pytype=str

Defined at stringconstants.f90 line 244

sc_emdatapathname
Element sc_emdatapathname ftype=character(14) pytype=str

Defined at stringconstants.f90 line 75

sc_emebsddefectnml
Element sc_emebsddefectnml ftype=character(15) pytype=str

Defined at stringconstants.f90 line 250

sc_emebsdfullnml
Element sc_emebsdfullnml ftype=character(13) pytype=str

Defined at stringconstants.f90 line 246

sc_emebsdnml
Element sc_emebsdnml ftype=character(9) pytype=str

Defined at stringconstants.f90 line 248

sc_emebsdscannml
Element sc_emebsdscannml ftype=character(13) pytype=str

Defined at stringconstants.f90 line 254

sc_emecp
Element sc_emecp ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1163

sc_emecpnml
Element sc_emecpnml ftype=character(8) pytype=str

Defined at stringconstants.f90 line 256

sc_emheader
Element sc_emheader ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1165

sc_emkinematical
Element sc_emkinematical ftype=character(13) pytype=str

Defined at stringconstants.f90 line 262

sc_emkinematicalnamelist
Element sc_emkinematicalnamelist ftype=character(21) pytype=str

Defined at stringconstants.f90 line 1167

sc_emnotify
Element sc_emnotify ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1314

sc_empedza
Element sc_empedza ftype=character(7) pytype=str

Defined at stringconstants.f90 line 258

sc_emshtdoi
Element sc_emshtdoi ftype=character(34) pytype=str

Defined at stringconstants.f90 line 1316

sc_emslackchannel
Element sc_emslackchannel ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1312

sc_emslackwebhookurl
Element sc_emslackwebhookurl ftype=character(17) pytype=str

Defined at stringconstants.f90 line 1310

sc_emsoft
Element sc_emsoft ftype=character(6) pytype=str

Defined at stringconstants.f90 line 65

sc_emsoftlibrarylocation
Element sc_emsoftlibrarylocation ftype=character(21) pytype=str

Defined at stringconstants.f90 line 71

sc_emsoftnativedelimiter
Element sc_emsoftnativedelimiter ftype=character(21) pytype=str

Defined at stringconstants.f90 line 107

sc_emsoftpathname
Element sc_emsoftpathname ftype=character(14) pytype=str

Defined at stringconstants.f90 line 69

sc_emsoftplatform
Element sc_emsoftplatform ftype=character(14) pytype=str

Defined at stringconstants.f90 line 79

sc_emsofttestingpath
Element sc_emsofttestingpath ftype=character(17) pytype=str

Defined at stringconstants.f90 line 83

sc_emsofttestpath
Element sc_emsofttestpath ftype=character(14) pytype=str

Defined at stringconstants.f90 line 81

sc_emsoftversion
Element sc_emsoftversion ftype=character(13) pytype=str

Defined at stringconstants.f90 line 85

sc_emtkdnml
Element sc_emtkdnml ftype=character(8) pytype=str

Defined at stringconstants.f90 line 260

sc_emtmppathname
Element sc_emtmppathname ftype=character(13) pytype=str

Defined at stringconstants.f90 line 77

sc_emxtalfolderpathname
Element sc_emxtalfolderpathname ftype=character(20) pytype=str

Defined at stringconstants.f90 line 73

sc_energyaverage
Element sc_energyaverage ftype=character(13) pytype=str

Defined at stringconstants.f90 line 656

sc_energyfile
Element sc_energyfile ftype=character(10) pytype=str

Defined at stringconstants.f90 line 660

sc_energymax
Element sc_energymax ftype=character(9) pytype=str

Defined at stringconstants.f90 line 662

sc_energymin
Element sc_energymin ftype=character(9) pytype=str

Defined at stringconstants.f90 line 664

sc_esel
Element sc_esel ftype=character(4) pytype=str

Defined at stringconstants.f90 line 272

sc_eulerangles
Element sc_eulerangles ftype=character(11) pytype=str

Defined at stringconstants.f90 line 276

sc_eulerconvention
Element sc_eulerconvention ftype=character(15) pytype=str

Defined at stringconstants.f90 line 666

sc_eulerfile
Element sc_eulerfile ftype=character(9) pytype=str

Defined at stringconstants.f90 line 668

sc_eulertriplet
Element sc_eulertriplet ftype=character(12) pytype=str

Defined at stringconstants.f90 line 278

sc_euoutname
Element sc_euoutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 670

sc_exptfile
Element sc_exptfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 672

sc_fftwwisdomtxt
Element sc_fftwwisdomtxt ftype=character(15) pytype=str

Defined at stringconstants.f90 line 129

sc_filemode
Element sc_filemode ftype=character(8) pytype=str

Defined at stringconstants.f90 line 674

sc_filmfile
Element sc_filmfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 676

sc_filterpattern
Element sc_filterpattern ftype=character(13) pytype=str

Defined at stringconstants.f90 line 678

sc_fit
Element sc_fit ftype=character(3) pytype=str

Defined at stringconstants.f90 line 282

sc_fixedlength
Element sc_fixedlength ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1169

sc_float1d
Element sc_float1d ftype=character(7) pytype=str

Defined at stringconstants.f90 line 682

sc_float2d
Element sc_float2d ftype=character(7) pytype=str

Defined at stringconstants.f90 line 684

sc_float3d
Element sc_float3d ftype=character(7) pytype=str

Defined at stringconstants.f90 line 686

sc_float4d
Element sc_float4d ftype=character(7) pytype=str

Defined at stringconstants.f90 line 688

sc_floattype
Element sc_floattype ftype=character(5) pytype=str

Defined at stringconstants.f90 line 680

sc_fn
Element sc_fn ftype=character(2) pytype=str

Defined at stringconstants.f90 line 690

sc_fnf
Element sc_fnf ftype=character(4) pytype=str

Defined at stringconstants.f90 line 692

sc_fns
Element sc_fns ftype=character(4) pytype=str

Defined at stringconstants.f90 line 694

sc_formula
Element sc_formula ftype=character(7) pytype=str

Defined at stringconstants.f90 line 284

sc_frfo
Element sc_frfo ftype=character(4) pytype=str

Defined at stringconstants.f90 line 698

sc_ftensor
Element sc_ftensor ftype=character(7) pytype=str

Defined at stringconstants.f90 line 286

sc_full
Element sc_full ftype=character(4) pytype=str

Defined at stringconstants.f90 line 696

sc_fzcnt
Element sc_fzcnt ftype=character(5) pytype=str

Defined at stringconstants.f90 line 280

sc_ga
Element sc_ga ftype=character(2) pytype=str

Defined at stringconstants.f90 line 704

sc_gammavalue
Element sc_gammavalue ftype=character(10) pytype=str

Defined at stringconstants.f90 line 706

sc_gb
Element sc_gb ftype=character(2) pytype=str

Defined at stringconstants.f90 line 708

sc_geometry
Element sc_geometry ftype=character(8) pytype=str

Defined at stringconstants.f90 line 710

sc_gf
Element sc_gf ftype=character(2) pytype=str

Defined at stringconstants.f90 line 700

sc_globalworkgrpsz
Element sc_globalworkgrpsz ftype=character(15) pytype=str

Defined at stringconstants.f90 line 712

sc_gridtype
Element sc_gridtype ftype=character(9) pytype=str

Defined at stringconstants.f90 line 288

sc_groupname
Element sc_groupname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 714

sc_gs
Element sc_gs ftype=character(2) pytype=str

Defined at stringconstants.f90 line 702

sc_h5copy
Element sc_h5copy ftype=character(11) pytype=str

Defined at stringconstants.f90 line 53

sc_h5copypath
Element sc_h5copypath ftype=character(10) pytype=str

Defined at stringconstants.f90 line 55

sc_h5ebsd
Element sc_h5ebsd ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1281

sc_h5pfinversion
Element sc_h5pfinversion ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1283

sc_hdffileversion
Element sc_hdffileversion ftype=character(15) pytype=str

Defined at stringconstants.f90 line 294

sc_hdfname
Element sc_hdfname ftype=character(7) pytype=str

Defined at stringconstants.f90 line 290

sc_hdfstrings
Element sc_hdfstrings ftype=character(10) pytype=str

Defined at stringconstants.f90 line 292

sc_header
Element sc_header ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1171

sc_hipassw
Element sc_hipassw ftype=character(7) pytype=str

Defined at stringconstants.f90 line 716

sc_hipasswmax
Element sc_hipasswmax ftype=character(10) pytype=str

Defined at stringconstants.f90 line 718

sc_hipasswnsteps
Element sc_hipasswnsteps ftype=character(13) pytype=str

Defined at stringconstants.f90 line 720

sc_hkl
Element sc_hkl ftype=character(3) pytype=str

Defined at stringconstants.f90 line 722

sc_hklarray
Element sc_hklarray ftype=character(8) pytype=str

Defined at stringconstants.f90 line 726

sc_hklfamilies
Element sc_hklfamilies ftype=character(11) pytype=str

Defined at stringconstants.f90 line 724

sc_hklmax
Element sc_hklmax ftype=character(6) pytype=str

Defined at stringconstants.f90 line 728

sc_homepathname
Element sc_homepathname ftype=character(12) pytype=str

Defined at stringconstants.f90 line 95

sc_hooutname
Element sc_hooutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 730

sc_horizontalaxisa
Element sc_horizontalaxisa ftype=character(15) pytype=str

Defined at stringconstants.f90 line 298

sc_hostname
Element sc_hostname ftype=character(8) pytype=str

Defined at stringconstants.f90 line 296

sc_hypercarr2
Element sc_hypercarr2 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 732

sc_hypercarr3
Element sc_hypercarr3 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 734

sc_hypercarr4
Element sc_hypercarr4 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 736

sc_hyperdarr2
Element sc_hyperdarr2 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 738

sc_hyperdarr3
Element sc_hyperdarr3 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 740

sc_hyperdarr4
Element sc_hyperdarr4 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 742

sc_hyperfarr2
Element sc_hyperfarr2 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 744

sc_hyperfarr3
Element sc_hyperfarr3 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 746

sc_hyperfarr4
Element sc_hyperfarr4 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 748

sc_hyperiarr2
Element sc_hyperiarr2 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 750

sc_hyperiarr3
Element sc_hyperiarr3 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 752

sc_hyperiarr4
Element sc_hyperiarr4 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 754

sc_hypslab
Element sc_hypslab ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1299

sc_id
Element sc_id ftype=character(2) pytype=str

Defined at stringconstants.f90 line 300

sc_includebackground
Element sc_includebackground ftype=character(17) pytype=str

Defined at stringconstants.f90 line 550

sc_indexingmode
Element sc_indexingmode ftype=character(12) pytype=str

Defined at stringconstants.f90 line 756

sc_indices
Element sc_indices ftype=character(7) pytype=str

Defined at stringconstants.f90 line 314

sc_info
Element sc_info ftype=character(4) pytype=str

Defined at stringconstants.f90 line 316

sc_inputfilename
Element sc_inputfilename ftype=character(13) pytype=str

Defined at stringconstants.f90 line 758

sc_inputpolefigures
Element sc_inputpolefigures ftype=character(16) pytype=str

Defined at stringconstants.f90 line 1173

sc_inputtype
Element sc_inputtype ftype=character(9) pytype=str

Defined at stringconstants.f90 line 760

sc_intarr3d
Element sc_intarr3d ftype=character(8) pytype=str

Defined at stringconstants.f90 line 762

sc_intarray
Element sc_intarray ftype=character(8) pytype=str

Defined at stringconstants.f90 line 764

sc_intarray1d
Element sc_intarray1d ftype=character(10) pytype=str

Defined at stringconstants.f90 line 766

sc_intarray2d
Element sc_intarray2d ftype=character(10) pytype=str

Defined at stringconstants.f90 line 768

sc_intarray3d
Element sc_intarray3d ftype=character(10) pytype=str

Defined at stringconstants.f90 line 770

sc_intbxfile
Element sc_intbxfile ftype=character(9) pytype=str

Defined at stringconstants.f90 line 772

sc_intbyfile
Element sc_intbyfile ftype=character(9) pytype=str

Defined at stringconstants.f90 line 774

sc_integer
Element sc_integer ftype=character(7) pytype=str

Defined at stringconstants.f90 line 776

sc_integer1d
Element sc_integer1d ftype=character(9) pytype=str

Defined at stringconstants.f90 line 778

sc_integer2d
Element sc_integer2d ftype=character(9) pytype=str

Defined at stringconstants.f90 line 780

sc_integer3d
Element sc_integer3d ftype=character(9) pytype=str

Defined at stringconstants.f90 line 782

sc_integer4d
Element sc_integer4d ftype=character(9) pytype=str

Defined at stringconstants.f90 line 784

sc_intensities
Element sc_intensities ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1175

sc_intfactor
Element sc_intfactor ftype=character(9) pytype=str

Defined at stringconstants.f90 line 302

sc_ipfht
Element sc_ipfht ftype=character(6) pytype=str

Defined at stringconstants.f90 line 786

sc_ipfwd
Element sc_ipfwd ftype=character(6) pytype=str

Defined at stringconstants.f90 line 788

sc_iq
Element sc_iq ftype=character(2) pytype=str

Defined at stringconstants.f90 line 304

sc_iqmap
Element sc_iqmap ftype=character(5) pytype=str

Defined at stringconstants.f90 line 306

sc_isangle
Element sc_isangle ftype=character(7) pytype=str

Defined at stringconstants.f90 line 790

sc_ism
Element sc_ism ftype=character(3) pytype=str

Defined at stringconstants.f90 line 308

sc_ismap
Element sc_ismap ftype=character(5) pytype=str

Defined at stringconstants.f90 line 310

sc_isr
Element sc_isr ftype=character(19) pytype=str

Defined at stringconstants.f90 line 312

sc_ivolstepx
Element sc_ivolstepx ftype=character(9) pytype=str

Defined at stringconstants.f90 line 900

sc_ivolstepy
Element sc_ivolstepy ftype=character(9) pytype=str

Defined at stringconstants.f90 line 902

sc_ivolstepz
Element sc_ivolstepz ftype=character(9) pytype=str

Defined at stringconstants.f90 line 904

sc_ivolx
Element sc_ivolx ftype=character(5) pytype=str

Defined at stringconstants.f90 line 894

sc_ivoly
Element sc_ivoly ftype=character(5) pytype=str

Defined at stringconstants.f90 line 896

sc_ivolz
Element sc_ivolz ftype=character(5) pytype=str

Defined at stringconstants.f90 line 898

sc_jsonfilename
Element sc_jsonfilename ftype=character(17) pytype=str

Defined at stringconstants.f90 line 119

sc_jsonfiles
Element sc_jsonfiles ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1177

sc_jsontemplates
Element sc_jsontemplates ftype=character(13) pytype=str

Defined at stringconstants.f90 line 121

sc_k
Element sc_k ftype=character(1) pytype=str

Defined at stringconstants.f90 line 792

sc_kam
Element sc_kam ftype=character(3) pytype=str

Defined at stringconstants.f90 line 320

sc_kappavmf
Element sc_kappavmf ftype=character(8) pytype=str

Defined at stringconstants.f90 line 318

sc_keeptmpfile
Element sc_keeptmpfile ftype=character(11) pytype=str

Defined at stringconstants.f90 line 794

sc_kevs
Element sc_kevs ftype=character(4) pytype=str

Defined at stringconstants.f90 line 796

sc_kij
Element sc_kij ftype=character(3) pytype=str

Defined at stringconstants.f90 line 800

sc_kk
Element sc_kk ftype=character(2) pytype=str

Defined at stringconstants.f90 line 802

sc_klist
Element sc_klist ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1185

sc_kosselmaster
Element sc_kosselmaster ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1183

sc_kosselmasterlist
Element sc_kosselmasterlist ftype=character(16) pytype=str

Defined at stringconstants.f90 line 322

sc_kosselmasternamelist
Element sc_kosselmasternamelist ftype=character(20) pytype=str

Defined at stringconstants.f90 line 1179

sc_kosselmode
Element sc_kosselmode ftype=character(10) pytype=str

Defined at stringconstants.f90 line 324

sc_kosselnamelist
Element sc_kosselnamelist ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1181

sc_kv
Element sc_kv ftype=character(2) pytype=str

Defined at stringconstants.f90 line 798

sc_l
Element sc_l ftype=character(1) pytype=str

Defined at stringconstants.f90 line 326

sc_lacbednamelist
Element sc_lacbednamelist ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1187

sc_lambdamax
Element sc_lambdamax ftype=character(9) pytype=str

Defined at stringconstants.f90 line 330

sc_lambdamin
Element sc_lambdamin ftype=character(9) pytype=str

Defined at stringconstants.f90 line 332

sc_lambdas
Element sc_lambdas ftype=character(7) pytype=str

Defined at stringconstants.f90 line 328

sc_lastenergy
Element sc_lastenergy ftype=character(10) pytype=str

Defined at stringconstants.f90 line 804

sc_latgridtype
Element sc_latgridtype ftype=character(11) pytype=str

Defined at stringconstants.f90 line 814

sc_latticeconstanta
Element sc_latticeconstanta ftype=character(16) pytype=str

Defined at stringconstants.f90 line 334

sc_latticeconstantalpha
Element sc_latticeconstantalpha ftype=character(20) pytype=str

Defined at stringconstants.f90 line 336

sc_latticeconstantb
Element sc_latticeconstantb ftype=character(16) pytype=str

Defined at stringconstants.f90 line 338

sc_latticeconstantbeta
Element sc_latticeconstantbeta ftype=character(19) pytype=str

Defined at stringconstants.f90 line 340

sc_latticeconstantc
Element sc_latticeconstantc ftype=character(16) pytype=str

Defined at stringconstants.f90 line 342

sc_latticeconstantgamma
Element sc_latticeconstantgamma ftype=character(20) pytype=str

Defined at stringconstants.f90 line 344

sc_latticeparameters
Element sc_latticeparameters ftype=character(17) pytype=str

Defined at stringconstants.f90 line 346

sc_lauec
Element sc_lauec ftype=character(5) pytype=str

Defined at stringconstants.f90 line 806

sc_lauec2
Element sc_lauec2 ftype=character(6) pytype=str

Defined at stringconstants.f90 line 808

sc_lauegroup
Element sc_lauegroup ftype=character(9) pytype=str

Defined at stringconstants.f90 line 358

sc_lauemaster
Element sc_lauemaster ftype=character(10) pytype=str

Defined at stringconstants.f90 line 348

sc_lauemasternamelist
Element sc_lauemasternamelist ftype=character(18) pytype=str

Defined at stringconstants.f90 line 356

sc_lauemasternml
Element sc_lauemasternml ftype=character(13) pytype=str

Defined at stringconstants.f90 line 352

sc_lauenamelist
Element sc_lauenamelist ftype=character(12) pytype=str

Defined at stringconstants.f90 line 354

sc_lauenml
Element sc_lauenml ftype=character(7) pytype=str

Defined at stringconstants.f90 line 350

sc_lenga
Element sc_lenga ftype=character(5) pytype=str

Defined at stringconstants.f90 line 810

sc_lengb
Element sc_lengb ftype=character(5) pytype=str

Defined at stringconstants.f90 line 812

sc_lorentznamelist
Element sc_lorentznamelist ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1189

sc_magfile
Element sc_magfile ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1191

sc_magnetization
Element sc_magnetization ftype=character(13) pytype=str

Defined at stringconstants.f90 line 818

sc_magnetization2d
Element sc_magnetization2d ftype=character(15) pytype=str

Defined at stringconstants.f90 line 816

sc_magnetizationdata
Element sc_magnetizationdata ftype=character(17) pytype=str

Defined at stringconstants.f90 line 1193

sc_makedictionary
Element sc_makedictionary ftype=character(14) pytype=str

Defined at stringconstants.f90 line 820

sc_manufacturer
Element sc_manufacturer ftype=character(12) pytype=str

Defined at stringconstants.f90 line 372

sc_maskfile
Element sc_maskfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 826

sc_maskpattern
Element sc_maskpattern ftype=character(11) pytype=str

Defined at stringconstants.f90 line 828

sc_maskradius
Element sc_maskradius ftype=character(10) pytype=str

Defined at stringconstants.f90 line 830

sc_masterfile
Element sc_masterfile ftype=character(10) pytype=str

Defined at stringconstants.f90 line 840

sc_masternh
Element sc_masternh ftype=character(8) pytype=str

Defined at stringconstants.f90 line 832

sc_mastersh
Element sc_mastersh ftype=character(8) pytype=str

Defined at stringconstants.f90 line 834

sc_masterspnh
Element sc_masterspnh ftype=character(10) pytype=str

Defined at stringconstants.f90 line 836

sc_masterspsh
Element sc_masterspsh ftype=character(10) pytype=str

Defined at stringconstants.f90 line 838

sc_materialname
Element sc_materialname ftype=character(12) pytype=str

Defined at stringconstants.f90 line 374

sc_maxnumincell
Element sc_maxnumincell ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1297

sc_mccldata
Element sc_mccldata ftype=character(8) pytype=str

Defined at stringconstants.f90 line 360

sc_mcclfoilnamelist
Element sc_mcclfoilnamelist ftype=character(16) pytype=str

Defined at stringconstants.f90 line 1199

sc_mcclmultilayernamelist
Element sc_mcclmultilayernamelist ftype=character(22) pytype=str

Defined at stringconstants.f90 line 1195

sc_mcclnamelist
Element sc_mcclnamelist ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1197

sc_mcfoil
Element sc_mcfoil ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1209

sc_mcfoilnml
Element sc_mcfoilnml ftype=character(9) pytype=str

Defined at stringconstants.f90 line 366

sc_mclipss
Element sc_mclipss ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1201

sc_mclipssnamelist
Element sc_mclipssnamelist ftype=character(15) pytype=str

Defined at stringconstants.f90 line 1203

sc_mcmode
Element sc_mcmode ftype=character(6) pytype=str

Defined at stringconstants.f90 line 368

sc_mcnamelist
Element sc_mcnamelist ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1205

sc_mcopencl
Element sc_mcopencl ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1207

sc_mcopencljson
Element sc_mcopencljson ftype=character(12) pytype=str

Defined at stringconstants.f90 line 362

sc_mcopenclnml
Element sc_mcopenclnml ftype=character(11) pytype=str

Defined at stringconstants.f90 line 364

sc_mcspherenml
Element sc_mcspherenml ftype=character(11) pytype=str

Defined at stringconstants.f90 line 370

sc_meansubtraction
Element sc_meansubtraction ftype=character(15) pytype=str

Defined at stringconstants.f90 line 376

sc_meanvals
Element sc_meanvals ftype=character(8) pytype=str

Defined at stringconstants.f90 line 842

sc_microstructure
Element sc_microstructure ftype=character(14) pytype=str

Defined at stringconstants.f90 line 378

sc_microstructuredata
Element sc_microstructuredata ftype=character(18) pytype=str

Defined at stringconstants.f90 line 1211

sc_microstructurefile
Element sc_microstructurefile ftype=character(18) pytype=str

Defined at stringconstants.f90 line 1213

sc_miso
Element sc_miso ftype=character(4) pytype=str

Defined at stringconstants.f90 line 380

sc_mlpnh
Element sc_mlpnh ftype=character(5) pytype=str

Defined at stringconstants.f90 line 822

sc_mlpsh
Element sc_mlpsh ftype=character(5) pytype=str

Defined at stringconstants.f90 line 824

sc_mode
Element sc_mode ftype=character(4) pytype=str

Defined at stringconstants.f90 line 846

sc_modfsquared
Element sc_modfsquared ftype=character(11) pytype=str

Defined at stringconstants.f90 line 844

sc_montagename
Element sc_montagename ftype=character(11) pytype=str

Defined at stringconstants.f90 line 848

sc_mpfile
Element sc_mpfile ftype=character(6) pytype=str

Defined at stringconstants.f90 line 850

sc_mrcfile
Element sc_mrcfile ftype=character(7) pytype=str

Defined at stringconstants.f90 line 852

sc_multidevid
Element sc_multidevid ftype=character(10) pytype=str

Defined at stringconstants.f90 line 906

sc_multiplier
Element sc_multiplier ftype=character(10) pytype=str

Defined at stringconstants.f90 line 854

sc_mutualinformation
Element sc_mutualinformation ftype=character(17) pytype=str

Defined at stringconstants.f90 line 382

sc_namelisttemplates
Element sc_namelisttemplates ftype=character(17) pytype=str

Defined at stringconstants.f90 line 123

sc_natomtypes
Element sc_natomtypes ftype=character(10) pytype=str

Defined at stringconstants.f90 line 384

sc_ncolumns
Element sc_ncolumns ftype=character(8) pytype=str

Defined at stringconstants.f90 line 856

sc_ncubochoric
Element sc_ncubochoric ftype=character(11) pytype=str

Defined at stringconstants.f90 line 386

sc_ncubochoricvariable
Element sc_ncubochoricvariable ftype=character(11) pytype=str

Defined at stringconstants.f90 line 908

sc_nism
Element sc_nism ftype=character(4) pytype=str

Defined at stringconstants.f90 line 868

sc_nmlfiles
Element sc_nmlfiles ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1215

sc_nmlparameters
Element sc_nmlparameters ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1217

sc_nnav
Element sc_nnav ftype=character(4) pytype=str

Defined at stringconstants.f90 line 866

sc_nnk
Element sc_nnk ftype=character(3) pytype=str

Defined at stringconstants.f90 line 870

sc_nosm
Element sc_nosm ftype=character(4) pytype=str

Defined at stringconstants.f90 line 872

sc_notes
Element sc_notes ftype=character(5) pytype=str

Defined at stringconstants.f90 line 388

sc_npix
Element sc_npix ftype=character(4) pytype=str

Defined at stringconstants.f90 line 874

sc_npiy
Element sc_npiy ftype=character(4) pytype=str

Defined at stringconstants.f90 line 876

sc_npx
Element sc_npx ftype=character(3) pytype=str

Defined at stringconstants.f90 line 878

sc_nref
Element sc_nref ftype=character(4) pytype=str

Defined at stringconstants.f90 line 886

sc_nregions
Element sc_nregions ftype=character(8) pytype=str

Defined at stringconstants.f90 line 888

sc_nregionsmax
Element sc_nregionsmax ftype=character(11) pytype=str

Defined at stringconstants.f90 line 860

sc_nregionsmin
Element sc_nregionsmin ftype=character(11) pytype=str

Defined at stringconstants.f90 line 858

sc_nregionsstepsize
Element sc_nregionsstepsize ftype=character(16) pytype=str

Defined at stringconstants.f90 line 862

sc_nrows
Element sc_nrows ftype=character(5) pytype=str

Defined at stringconstants.f90 line 864

sc_nsteps
Element sc_nsteps ftype=character(6) pytype=str

Defined at stringconstants.f90 line 890

sc_nthreads
Element sc_nthreads ftype=character(8) pytype=str

Defined at stringconstants.f90 line 892

sc_num_el
Element sc_num_el ftype=character(6) pytype=str

Defined at stringconstants.f90 line 910

sc_numangle
Element sc_numangle ftype=character(8) pytype=str

Defined at stringconstants.f90 line 914

sc_numangledictionary
Element sc_numangledictionary ftype=character(19) pytype=str

Defined at stringconstants.f90 line 916

sc_numangles
Element sc_numangles ftype=character(9) pytype=str

Defined at stringconstants.f90 line 918

sc_numberfamilies
Element sc_numberfamilies ftype=character(14) pytype=str

Defined at stringconstants.f90 line 392

sc_numdetparm
Element sc_numdetparm ftype=character(10) pytype=str

Defined at stringconstants.f90 line 920

sc_numdictsingle
Element sc_numdictsingle ftype=character(13) pytype=str

Defined at stringconstants.f90 line 922

sc_numebins
Element sc_numebins ftype=character(8) pytype=str

Defined at stringconstants.f90 line 912

sc_numeuler
Element sc_numeuler ftype=character(8) pytype=str

Defined at stringconstants.f90 line 924

sc_numexptpatterns
Element sc_numexptpatterns ftype=character(15) pytype=str

Defined at stringconstants.f90 line 390

sc_numexptsingle
Element sc_numexptsingle ftype=character(13) pytype=str

Defined at stringconstants.f90 line 926

sc_numg
Element sc_numg ftype=character(4) pytype=str

Defined at stringconstants.f90 line 928

sc_numk
Element sc_numk ftype=character(4) pytype=str

Defined at stringconstants.f90 line 930

sc_numreflections
Element sc_numreflections ftype=character(14) pytype=str

Defined at stringconstants.f90 line 932

sc_numset
Element sc_numset ftype=character(6) pytype=str

Defined at stringconstants.f90 line 934

sc_numsx
Element sc_numsx ftype=character(5) pytype=str

Defined at stringconstants.f90 line 936

sc_numsy
Element sc_numsy ftype=character(5) pytype=str

Defined at stringconstants.f90 line 938

sc_numthick
Element sc_numthick ftype=character(8) pytype=str

Defined at stringconstants.f90 line 940

sc_numzbins
Element sc_numzbins ftype=character(8) pytype=str

Defined at stringconstants.f90 line 942

sc_nx
Element sc_nx ftype=character(2) pytype=str

Defined at stringconstants.f90 line 880

sc_ny
Element sc_ny ftype=character(2) pytype=str

Defined at stringconstants.f90 line 882

sc_nz
Element sc_nz ftype=character(2) pytype=str

Defined at stringconstants.f90 line 884

sc_odf
Element sc_odf ftype=character(3) pytype=str

Defined at stringconstants.f90 line 394

sc_omega
Element sc_omega ftype=character(5) pytype=str

Defined at stringconstants.f90 line 944

sc_omoutname
Element sc_omoutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 946

sc_opencl
Element sc_opencl ftype=character(6) pytype=str

Defined at stringconstants.f90 line 127

sc_openclpathname
Element sc_openclpathname ftype=character(14) pytype=str

Defined at stringconstants.f90 line 97

sc_operator
Element sc_operator ftype=character(8) pytype=str

Defined at stringconstants.f90 line 398

sc_orderparameter
Element sc_orderparameter ftype=character(4) pytype=str

Defined at stringconstants.f90 line 1237

sc_orientation
Element sc_orientation ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1295

sc_osm
Element sc_osm ftype=character(3) pytype=str

Defined at stringconstants.f90 line 396

sc_outname
Element sc_outname ftype=character(7) pytype=str

Defined at stringconstants.f90 line 948

sc_outputfile
Element sc_outputfile ftype=character(10) pytype=str

Defined at stringconstants.f90 line 952

sc_outputformat
Element sc_outputformat ftype=character(12) pytype=str

Defined at stringconstants.f90 line 950

sc_patchw
Element sc_patchw ftype=character(6) pytype=str

Defined at stringconstants.f90 line 954

sc_path
Element sc_path ftype=character(4) pytype=str

Defined at stringconstants.f90 line 404

sc_patternaxisa
Element sc_patternaxisa ftype=character(12) pytype=str

Defined at stringconstants.f90 line 410

sc_patterncentercalibration
Element sc_patterncentercalibration ftype=character(24) pytype=str

Defined at stringconstants.f90 line 1249

sc_patternfile
Element sc_patternfile ftype=character(11) pytype=str

Defined at stringconstants.f90 line 956

sc_patternheight
Element sc_patternheight ftype=character(14) pytype=str

Defined at stringconstants.f90 line 406

sc_patternwidth
Element sc_patternwidth ftype=character(13) pytype=str

Defined at stringconstants.f90 line 408

sc_patx
Element sc_patx ftype=character(4) pytype=str

Defined at stringconstants.f90 line 958

sc_paty
Element sc_paty ftype=character(4) pytype=str

Defined at stringconstants.f90 line 960

sc_pedkin
Element sc_pedkin ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1243

sc_pedkinnamelist
Element sc_pedkinnamelist ftype=character(14) pytype=str

Defined at stringconstants.f90 line 400

sc_pedpattern
Element sc_pedpattern ftype=character(10) pytype=str

Defined at stringconstants.f90 line 962

sc_pedpatterns
Element sc_pedpatterns ftype=character(11) pytype=str

Defined at stringconstants.f90 line 402

sc_pedza
Element sc_pedza ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1239

sc_pedzanamelist
Element sc_pedzanamelist ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1241

sc_pfinversiondata
Element sc_pfinversiondata ftype=character(15) pytype=str

Defined at stringconstants.f90 line 1245

sc_pfinversionnamelist
Element sc_pfinversionnamelist ftype=character(19) pytype=str

Defined at stringconstants.f90 line 1247

sc_phase
Element sc_phase ftype=character(5) pytype=str

Defined at stringconstants.f90 line 412

sc_phasemethod
Element sc_phasemethod ftype=character(11) pytype=str

Defined at stringconstants.f90 line 964

sc_phi
Element sc_phi ftype=character(3) pytype=str

Defined at stringconstants.f90 line 414

sc_phi1
Element sc_phi1 ftype=character(4) pytype=str

Defined at stringconstants.f90 line 416

sc_phi2
Element sc_phi2 ftype=character(4) pytype=str

Defined at stringconstants.f90 line 418

sc_phiefile
Element sc_phiefile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 966

sc_phimfile
Element sc_phimfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 968

sc_phinum
Element sc_phinum ftype=character(6) pytype=str

Defined at stringconstants.f90 line 970

sc_pixellocation
Element sc_pixellocation ftype=character(13) pytype=str

Defined at stringconstants.f90 line 420

sc_pixelsize
Element sc_pixelsize ftype=character(9) pytype=str

Defined at stringconstants.f90 line 972

sc_platid
Element sc_platid ftype=character(6) pytype=str

Defined at stringconstants.f90 line 974

sc_pointgroup
Element sc_pointgroup ftype=character(10) pytype=str

Defined at stringconstants.f90 line 422

sc_pointgroupnumber
Element sc_pointgroupnumber ftype=character(16) pytype=str

Defined at stringconstants.f90 line 424

sc_poisson
Element sc_poisson ftype=character(7) pytype=str

Defined at stringconstants.f90 line 976

sc_positions
Element sc_positions ftype=character(9) pytype=str

Defined at stringconstants.f90 line 978

sc_progmode
Element sc_progmode ftype=character(8) pytype=str

Defined at stringconstants.f90 line 980

sc_programname
Element sc_programname ftype=character(11) pytype=str

Defined at stringconstants.f90 line 426

sc_qctype
Element sc_qctype ftype=character(6) pytype=str

Defined at stringconstants.f90 line 428

sc_quoutname
Element sc_quoutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 982

sc_qxy
Element sc_qxy ftype=character(3) pytype=str

Defined at stringconstants.f90 line 984

sc_randomseedfilename
Element sc_randomseedfilename ftype=character(18) pytype=str

Defined at stringconstants.f90 line 105

sc_randomseedsdata
Element sc_randomseedsdata ftype=character(16) pytype=str

Defined at stringconstants.f90 line 139

sc_readmefirst
Element sc_readmefirst ftype=character(11) pytype=str

Defined at stringconstants.f90 line 93

sc_reconstructed
Element sc_reconstructed ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1253

sc_refcnt
Element sc_refcnt ftype=character(6) pytype=str

Defined at stringconstants.f90 line 986

sc_refinedcorrecteddotproducts
Element sc_refinedcorrecteddotproducts ftype=character(27) pytype=str

Defined at stringconstants.f90 line 430

sc_refinedcorrectedeulerangles
Element sc_refinedcorrectedeulerangles ftype=character(27) pytype=str

Defined at stringconstants.f90 line 432

sc_refineddotproducts
Element sc_refineddotproducts ftype=character(18) pytype=str

Defined at stringconstants.f90 line 434

sc_refinedeulerangles
Element sc_refinedeulerangles ftype=character(18) pytype=str

Defined at stringconstants.f90 line 436

sc_release
Element sc_release ftype=character(7) pytype=str

Defined at stringconstants.f90 line 117

sc_resourcepathname
Element sc_resourcepathname ftype=character(16) pytype=str

Defined at stringconstants.f90 line 91

sc_resources
Element sc_resources ftype=character(9) pytype=str

Defined at stringconstants.f90 line 125

sc_restart
Element sc_restart ftype=character(7) pytype=str

Defined at stringconstants.f90 line 988

sc_rfznamelist
Element sc_rfznamelist ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1251

sc_rin
Element sc_rin ftype=character(3) pytype=str

Defined at stringconstants.f90 line 438

sc_roi
Element sc_roi ftype=character(3) pytype=str

Defined at stringconstants.f90 line 1255

sc_rooutname
Element sc_rooutname ftype=character(9) pytype=str

Defined at stringconstants.f90 line 990

sc_rout
Element sc_rout ftype=character(4) pytype=str

Defined at stringconstants.f90 line 440

sc_sampleid
Element sc_sampleid ftype=character(9) pytype=str

Defined at stringconstants.f90 line 444

sc_sampletilt
Element sc_sampletilt ftype=character(11) pytype=str

Defined at stringconstants.f90 line 446

sc_sampling
Element sc_sampling ftype=character(8) pytype=str

Defined at stringconstants.f90 line 992

sc_scalefactor
Element sc_scalefactor ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1289

sc_scalingmode
Element sc_scalingmode ftype=character(11) pytype=str

Defined at stringconstants.f90 line 994

sc_scan1
Element sc_scan1 ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1265

sc_scanid
Element sc_scanid ftype=character(7) pytype=str

Defined at stringconstants.f90 line 448

sc_schematic1
Element sc_schematic1 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 450

sc_schematic2
Element sc_schematic2 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 452

sc_schematic3
Element sc_schematic3 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 454

sc_schematic4
Element sc_schematic4 ftype=character(10) pytype=str

Defined at stringconstants.f90 line 456

sc_section
Element sc_section ftype=character(7) pytype=str

Defined at stringconstants.f90 line 996

sc_sele
Element sc_sele ftype=character(4) pytype=str

Defined at stringconstants.f90 line 998

sc_semsignal
Element sc_semsignal ftype=character(10) pytype=str

Defined at stringconstants.f90 line 458

sc_sg
Element sc_sg ftype=character(2) pytype=str

Defined at stringconstants.f90 line 1305

sc_sgdbdiff
Element sc_sgdbdiff ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1002

sc_sgname
Element sc_sgname ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1000

sc_sig
Element sc_sig ftype=character(3) pytype=str

Defined at stringconstants.f90 line 1004

sc_sigend
Element sc_sigend ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1006

sc_sigstart
Element sc_sigstart ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1008

sc_sigstep
Element sc_sigstep ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1010

sc_source
Element sc_source ftype=character(6) pytype=str

Defined at stringconstants.f90 line 99

sc_spacegroupnumber
Element sc_spacegroupnumber ftype=character(16) pytype=str

Defined at stringconstants.f90 line 460

sc_spacegroupsetting
Element sc_spacegroupsetting ftype=character(17) pytype=str

Defined at stringconstants.f90 line 462

sc_spatialaverage
Element sc_spatialaverage ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1012

sc_sphinxnamelist
Element sc_sphinxnamelist ftype=character(14) pytype=str

Defined at stringconstants.f90 line 464

sc_squhex
Element sc_squhex ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1014

sc_startthick
Element sc_startthick ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1016

sc_starttime
Element sc_starttime ftype=character(9) pytype=str

Defined at stringconstants.f90 line 466

sc_stdout
Element sc_stdout ftype=character(6) pytype=str

Defined at stringconstants.f90 line 468

sc_stemdci
Element sc_stemdci ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1257

sc_stemdcinamelist
Element sc_stemdcinamelist ftype=character(15) pytype=str

Defined at stringconstants.f90 line 1259

sc_stemdcinml
Element sc_stemdcinml ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1261

sc_stemgeometrynamelist
Element sc_stemgeometrynamelist ftype=character(20) pytype=str

Defined at stringconstants.f90 line 1263

sc_stemnmlfile
Element sc_stemnmlfile ftype=character(11) pytype=str

Defined at stringconstants.f90 line 442

sc_stepx
Element sc_stepx ftype=character(6) pytype=str

Defined at stringconstants.f90 line 470

sc_stepxvariable
Element sc_stepxvariable ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1018

sc_stepy
Element sc_stepy ftype=character(6) pytype=str

Defined at stringconstants.f90 line 472

sc_stepyvariable
Element sc_stepyvariable ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1020

sc_stereonh
Element sc_stereonh ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1022

sc_stereosh
Element sc_stereosh ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1024

sc_stoptime
Element sc_stoptime ftype=character(8) pytype=str

Defined at stringconstants.f90 line 474

sc_str
Element sc_str ftype=character(3) pytype=str

Defined at stringconstants.f90 line 612

sc_subsfile
Element sc_subsfile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1026

sc_summode
Element sc_summode ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1028

sc_symmetry
Element sc_symmetry ftype=character(8) pytype=str

Defined at stringconstants.f90 line 476

sc_templatecodefilename
Element sc_templatecodefilename ftype=character(20) pytype=str

Defined at stringconstants.f90 line 101

sc_templatecodestxt
Element sc_templatecodestxt ftype=character(17) pytype=str

Defined at stringconstants.f90 line 133

sc_templatepathname
Element sc_templatepathname ftype=character(16) pytype=str

Defined at stringconstants.f90 line 89

sc_temporary
Element sc_temporary ftype=character(9) pytype=str

Defined at stringconstants.f90 line 59

sc_test2group
Element sc_test2group ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1285

sc_testdata
Element sc_testdata ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1277

sc_testgroup
Element sc_testgroup ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1287

sc_testing
Element sc_testing ftype=character(7) pytype=str

Defined at stringconstants.f90 line 57

sc_testnml
Element sc_testnml ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1034

sc_tf
Element sc_tf ftype=character(2) pytype=str

Defined at stringconstants.f90 line 1030

sc_thetac
Element sc_thetac ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1036

sc_thickinc
Element sc_thickinc ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1038

sc_thk
Element sc_thk ftype=character(3) pytype=str

Defined at stringconstants.f90 line 1303

sc_tifffile
Element sc_tifffile ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1040

sc_tiffname
Element sc_tiffname ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1042

sc_tiffprefix
Element sc_tiffprefix ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1044

sc_tiltaxis
Element sc_tiltaxis ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1046

sc_tkd
Element sc_tkd ftype=character(3) pytype=str

Defined at stringconstants.f90 line 1267

sc_tkdindexingnamelisttype
Element sc_tkdindexingnamelisttype ftype=character(23) pytype=str

Defined at stringconstants.f90 line 1269

sc_tkdmaster
Element sc_tkdmaster ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1275

sc_tkdmasternamelist
Element sc_tkdmasternamelist ftype=character(17) pytype=str

Defined at stringconstants.f90 line 1271

sc_tkdmasternml
Element sc_tkdmasternml ftype=character(12) pytype=str

Defined at stringconstants.f90 line 478

sc_tkdnamelist
Element sc_tkdnamelist ftype=character(11) pytype=str

Defined at stringconstants.f90 line 1273

sc_tkdpatterns
Element sc_tkdpatterns ftype=character(11) pytype=str

Defined at stringconstants.f90 line 480

sc_tkdspots
Element sc_tkdspots ftype=character(8) pytype=str

Defined at stringconstants.f90 line 484

sc_tkdspotsnml
Element sc_tkdspotsnml ftype=character(11) pytype=str

Defined at stringconstants.f90 line 482

sc_tmp
Element sc_tmp ftype=character(3) pytype=str

Defined at stringconstants.f90 line 67

sc_tmpfile
Element sc_tmpfile ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1048

sc_topdotproductlist
Element sc_topdotproductlist ftype=character(17) pytype=str

Defined at stringconstants.f90 line 486

sc_topmatchindices
Element sc_topmatchindices ftype=character(15) pytype=str

Defined at stringconstants.f90 line 488

sc_totnumel
Element sc_totnumel ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1050

sc_totnumexpt
Element sc_totnumexpt ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1052

sc_trange
Element sc_trange ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1054

sc_ts
Element sc_ts ftype=character(2) pytype=str

Defined at stringconstants.f90 line 1032

sc_uniform
Element sc_uniform ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1056

sc_unitgvectors
Element sc_unitgvectors ftype=character(12) pytype=str

Defined at stringconstants.f90 line 1058

sc_useenergyweighting
Element sc_useenergyweighting ftype=character(18) pytype=str

Defined at stringconstants.f90 line 1060

sc_usenumd
Element sc_usenumd ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1064

sc_useremail
Element sc_useremail ftype=character(9) pytype=str

Defined at stringconstants.f90 line 113

sc_userlocation
Element sc_userlocation ftype=character(12) pytype=str

Defined at stringconstants.f90 line 111

sc_username
Element sc_username ftype=character(8) pytype=str

Defined at stringconstants.f90 line 109

sc_usetmpfile
Element sc_usetmpfile ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1062

sc_valid
Element sc_valid ftype=character(5) pytype=str

Defined at stringconstants.f90 line 490

sc_variants
Element sc_variants ftype=character(8) pytype=str

Defined at stringconstants.f90 line 492

sc_version
Element sc_version ftype=character(7) pytype=str

Defined at stringconstants.f90 line 494

sc_vmean
Element sc_vmean ftype=character(5) pytype=str

Defined at stringconstants.f90 line 1066

sc_voltage
Element sc_voltage ftype=character(7) pytype=str

Defined at stringconstants.f90 line 1068

sc_wavelength
Element sc_wavelength ftype=character(10) pytype=str

Defined at stringconstants.f90 line 1070

sc_wiki
Element sc_wiki ftype=character(4) pytype=str

Defined at stringconstants.f90 line 135

sc_wikicodestxt
Element sc_wikicodestxt ftype=character(13) pytype=str

Defined at stringconstants.f90 line 131

sc_windows
Element sc_windows ftype=character(7) pytype=str

Defined at stringconstants.f90 line 61

sc_workingdistance
Element sc_workingdistance ftype=character(16) pytype=str

Defined at stringconstants.f90 line 496

sc_wyckoffpositionsfilename
Element sc_wyckoffpositionsfilename ftype=character(23) pytype=str

Defined at stringconstants.f90 line 103

sc_wyckoffpositionstxt
Element sc_wyckoffpositionstxt ftype=character(20) pytype=str

Defined at stringconstants.f90 line 137

sc_xpc
Element sc_xpc ftype=character(3) pytype=str

Defined at stringconstants.f90 line 1074

sc_xpos
Element sc_xpos ftype=character(10) pytype=str

Defined at stringconstants.f90 line 500

sc_xposition
Element sc_xposition ftype=character(9) pytype=str

Defined at stringconstants.f90 line 498

sc_xstar
Element sc_xstar ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1072

sc_xtalname
Element sc_xtalname ftype=character(8) pytype=str

Defined at stringconstants.f90 line 1076

sc_xtalname2
Element sc_xtalname2 ftype=character(9) pytype=str

Defined at stringconstants.f90 line 1078

sc_xtalname_gamma
Element sc_xtalname_gamma ftype=character(14) pytype=str

Defined at stringconstants.f90 line 1084

sc_xtalname_gammap
Element sc_xtalname_gammap ftype=character(15) pytype=str

Defined at stringconstants.f90 line 1086

sc_xtalnamefilm
Element sc_xtalnamefilm ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1080

sc_xtalnamesubs
Element sc_xtalnamesubs ftype=character(13) pytype=str

Defined at stringconstants.f90 line 1082

sc_xyarray
Element sc_xyarray ftype=character(7) pytype=str

Defined at stringconstants.f90 line 502

sc_ypc
Element sc_ypc ftype=character(3) pytype=str

Defined at stringconstants.f90 line 1090

sc_ypos
Element sc_ypos ftype=character(10) pytype=str

Defined at stringconstants.f90 line 506

sc_yposition
Element sc_yposition ftype=character(9) pytype=str

Defined at stringconstants.f90 line 504

sc_ystar
Element sc_ystar ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1088

sc_zaimages
Element sc_zaimages ftype=character(8) pytype=str

Defined at stringconstants.f90 line 508

sc_zenododoi
Element sc_zenododoi ftype=character(38) pytype=str

Defined at stringconstants.f90 line 1318

sc_zstar
Element sc_zstar ftype=character(6) pytype=str

Defined at stringconstants.f90 line 1092

class pyEMsoft.**Symmetry**(**args*, ***kwargs*)
Module symmetry

Defined at symmetry.f90 lines 49-2703

static bfsymmetry(*self*, *uvw*, *j*)
isym, ir = bfsymmetry(self, uvw, j)

Defined at symmetry.f90 lines 1349-1370

Parameters

- **cell** (*Unitcell*) –
- **uvw** (*int array*) –
- **j** (*int*) –

Returns

- **isym** (*int*)
- **ir** (*int*)

static calcequivpos(*self*, *site*, *n*, *ctmp*)
Defined at symmetry.f90 lines 587-644

Parameters

- **cell** (*Unitcell*) –
- **site** (*float array*) –
- **n** (*int*) –
- **ctmp** (*float array*) –

static calcfamily(*self*, *ind*, *space*, *itmp*)
num = calcfamily(self, ind, space, itmp)

Defined at symmetry.f90 lines 444-487

Parameters

- **cell** (*Unitcell*) –
- **ind** (*int array*) –
- **space** (*str*) –
- **itmp** (*int array*) –

Returns num

Return type int

static calcorbit (*self*, *m*, *ctmp*)
n = calcorbit(self, m, ctmp)

Defined at symmetry.f90 lines 511-570

Parameters

- **cell** (*Unitcell*) –
- **m** (*int*) –
- **ctmp** (*float array*) –

Returns n

Return type int

static calcpositions (*self*, *switch_bn*)
Defined at symmetry.f90 lines 738-816

Parameters

- **cell** (*Unitcell*) –
- **switch_bn** (*str*) –

static calcstar (*self*, *kk*, *stmp*, *space*)
n = calcstar(self, kk, stmp, space)

Defined at symmetry.f90 lines 671-714

Parameters

- **cell** (*Unitcell*) –
- **kk** (*float array*) –
- **stmp** (*float array*) –
- **space** (*str*) –

Returns n

Return type int

static checkpatternsymmetry (*self*, *k*, *ga*, *isym*)
thetam = checkpatternsymmetry(self, k, ga, isym)

Defined at symmetry.f90 lines 2128-2167

Parameters

- **cell** (*Unitcell*) –
- **k** (*int array*) –
- **ga** (*int array*) –

- **isym** (*int*) –

Returns thetam

Return type float

static extractwyckoffposition(*wyckoffpos*, *pt*)
Defined at symmetry.f90 lines 2608-2659

Parameters

- **wyckoffpos** (*str*) –
- **pt** (*float array*) –

static generate2dsymmetry(*tdpg*, *pgn*)
Defined at symmetry.f90 lines 1991-2099

Parameters

- **tdpg** (*Symdata2D*) –
- **pgn** (*int*) –
- **------------** –

static generatesymmetry(*self*, *dopg*)
Defined at symmetry.f90 lines 349-421

Parameters

- **cell** (*Unitcell*) –
- **dopg** (*bool*) –

static getasymposwyckoff(*self*)
Defined at symmetry.f90 lines 2516-2590

Parameters **cell** (*Unitcell*) –

static getdiffractiongroup(*self*, *uvw*, *pgn*)
dgn = getdiffractiongroup(self, uvw, pgn)

Defined at symmetry.f90 lines 1448-1972

Parameters

- **cell** (*Unitcell*) –
- **uvw** (*int array*) –
- **pgn** (*int*) –

Returns

- **dgn** (*int*)
- ———

static gethexvsrho(*self*, *pgnum*)
stnum = gethexvsrho(self, pgnum)

Defined at symmetry.f90 lines 2185-2231

Parameters

- **cell** (*Unitcell*) –
- **pgnum** (*int*) –

Returns **stnum**

Return type int

static getlauegroupnumber(*sgnum*)
lgn = getlauegroupnumber(sgnum)

Defined at symmetry.f90 lines 2673-2703

Parameters **sgnum** (*int*) –

Returns **lgn**

Return type int

static getorder(*k*, *il*, *num*, *itmp*)
jcnt = getorder(k, il, num, itmp)

Defined at symmetry.f90 lines 1039-1057

Parameters

- **k** (*float array*) –
- **il** (*int array*) –
- **num** (*int*) –
- **itmp** (*int array*) –

Returns **jcnt**

Return type int

static getpatternsymmetry(*self*, *uvw*, *pgnum*, *verbose=None*)
dgn = getpatternsymmetry(self, uvw, pgnum[, verbose])

Defined at symmetry.f90 lines 1389-1418

Parameters

- **cell** (*Unitcell*) –
- **uvw** (*int array*) –
- **pgnum** (*int*) –
- **verbose** (*bool*) –

Returns **dgn**

Return type int

static getsetting(*self*)
iset = getsetting(self)

Defined at symmetry.f90 lines 840-884

Parameters **cell** (*Unitcell*) –

Returns **iset**

Return type int

static getspacegroup(*self*)
Defined at symmetry.f90 lines 947-1014

Parameters **cell** (*Unitcell*) –

static interpretwyckoffletter(*t*, *x*, *y*, *z*)
st = interpretwyckoffletter(t, x, y, z)

Defined at symmetry.f90 lines 2344-2380

Parameters

- **t** (*str*) –
- **x** (*float*) –
- **y** (*float*) –
- **z** (*float*) –

Returns st

Return type float

static isgallowed(*self*, *g*)
isgallowed = isgallowed(self, g)

Defined at symmetry.f90 lines 1292-1313

Parameters

- **cell** (*Unitcell*) –
- **g** (*int array*) –

Returns isgallowed

Return type bool

static isitnew(*self*, *nsym*)
isitnew = isitnew(self, nsym)

Defined at symmetry.f90 lines 304-325

Parameters

- **cell** (*Unitcell*) –
- **nsym** (*int*) –

Returns isitnew

Return type bool

static listpointgroups()
Defined at symmetry.f90 lines 896-910

static makegenerators(*self*)
Defined at symmetry.f90 lines 140-225

Parameters **cell** (*Unitcell*) –

static matrixmult(*self*, *k1*, *k2*)
Defined at symmetry.f90 lines 251-282

Parameters

- **cell** (*Unitcell*) –
- **k1** (*int*) –
- **k2** (*int*) –

static shortestg(*self*, *k*, *gone*, *gtwo*, *isym*)
Defined at symmetry.f90 lines 1109-1269

Parameters

- **cell** (*Unitcell*) –
- **k** (*int array*) –
- **gone** (*int array*) –
- **gtwo** (*int array*) –
- **isym** (*int*) –

static sym_fillgen(*self*, *t*, *isgn*)
Defined at symmetry.f90 lines 73-119

Parameters

- **cell** (*Unitcell*) –
- **t** (*str array*) –
- **isgn** (*int*) –

static sym_getmultiplicity(*t*)
stmult = sym_getmultiplicity(t)

Defined at symmetry.f90 lines 2245-2271

Parameters **t** (*str*) –

Returns **stmult**

Return type str

static sym_getposition(*t*)
st = sym_getposition(t)

Defined at symmetry.f90 lines 2285-2329

Parameters **t** (*str*) –

Returns **st**

Return type str

static sym_getwpstring(*sgnum*, *wpstring*)
Defined at symmetry.f90 lines 2395-2424

Parameters

- **sgnum** (*int*) –
- **wpstring** (*str*) –

static sym_printwyckoffpositions(*sgnum*, *wpstring*[, *wyckofflist*])
Defined at symmetry.f90 lines 2440-2499

Parameters

- **sgnum** (*int*) –
- **wpstring** (*str*) –
- **wyckofflist** (*str array*) –

class pyEMsoft.**Timing**(**args*, ***kwargs*)
Module timing

Defined at timing.f90 lines 42-302

static printtime(*tm*)
Defined at timing.f90 lines 249-273

Parameters **tm** (*float*) –

static time_estimate(*self*, *numk*)
Defined at timing.f90 lines 170-186

Parameters

- **tt** (*Timetype*) –
- **numk** (*int*) –

static time_remaining(*self*, *ik*, *numk*)
Defined at timing.f90 lines 204-233

Parameters

- **tt** (*Timetype*) –
- **ik** (*int*) –
- **numk** (*int*) –

static time_report(*self*, *interval*)
Defined at timing.f90 lines 124-132

Parameters

- **tt** (*Timetype*) –
- **interval** (*float*) –

static time_reset(*self*)
Defined at timing.f90 lines 95-107

Parameters **tt** (*Timetype*) –

static time_start(*self*)
Defined at timing.f90 lines 146-153

Parameters **tt** (*Timetype*) –

static time_stop(*self*, *numk*)
Defined at timing.f90 lines 290-302

Parameters

- **tt** (*Timetype*) –
- **numk** (*int*) –

static time_tick()
t = time_tick()

Defined at timing.f90 lines 58-62

Returns t

Return type int

static time_tock(*t*)
tock = time_tock(t)

Defined at timing.f90 lines 74-81

Parameters **t** (*int*) –

Returns tock

Return type float

class pyEMsoft.**Typedefs**(**args*, ***kwargs*)
Module typedefs

Defined at typedefs.f90 lines 65-1777

bfpg
Element bfpg ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 461

cslintegers
Element cslintegers ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 1622

csllabels
Element csllabels ftype=character(3) pytype=str

Defined at typedefs.f90 line 1592

cslnumberdefined
Element cslnumberdefined ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 1587

dfgn
Element dfgn ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 467

dfsp
Element dfsp ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 470

dg
Element dg ftype=character(5) pytype=str

Defined at typedefs.f90 line 453

dgorder
Element dgorder ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 458

dgpg
Element dgpg ftype=logical pytype=bool

Defined at typedefs.f90 line 514

extendedhmorthsymbols
Element extendedhmorthsymbols ftype=character(11) pytype=str

Defined at typedefs.f90 line 268

extendedorthsettings
Element extendedorthsettings ftype=character(8) pytype=str

Defined at typedefs.f90 line 207

fztypetable
Element fztypetable ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 819

maxdefects
Element maxdefects ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 79

maxpasym
Element maxpasym ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 76

pdg
Element pdg ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 474

pglaue
Element pglaue ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 435

pglaueinv
Element pglaueinv ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 439

pgrot
Element pgrot ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 431

pgsamplingtype
Element pgsamplingtype ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 445

pgthd
Element pgthd ftype=character(5) pytype=str

Defined at typedefs.f90 line 421

pgthdorder
Element pgthdorder ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 427

pgtwd
Element pgtwd ftype=character(10) pytype=str

Defined at typedefs.f90 line 400

pgtwdinverse
Element pgtwdinverse ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 412

pgtwdorder
Element pgtwdorder ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 403

sgpg
Element sgpg ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 358

sgsym
Element sgsym ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 366

sgsymnum
Element sgsymnum ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 394

sgxsym
Element sgxsym ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 353

sht_mirinv
Element sht_mirinv ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 540

sht_zrot
Element sht_zrot ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 528

sym_gl
Element sym_gl ftype=character(40) pytype=str

Defined at typedefs.f90 line 350

sym_qsymop
Element sym_qsymop ftype=real(kind=dbl) pytype=float

Defined at typedefs.f90 line 734

sym_sgname
Element sym_sgname ftype=character(11) pytype=str

Defined at typedefs.f90 line 203

tslsymtype
Element tslsymtype ftype=character(2) pytype=str

Defined at typedefs.f90 line 759

wppg
Element wppg ftype=integer(kind=irg) pytype=int

Defined at typedefs.f90 line 464

class pyEMsoft.**Utilities**(**args*, ***kwargs*)
Module utilities

Defined at utilities.f90 lines 40-117

static getxtaldata (*xtalname*, *cellparams*, *tslsymmetry=None*)
sgnum = getxtaldata(xtalname, cellparams[, tslsymmetry])

Defined at utilities.f90 lines 60-117

Parameters

- **xtalname** (*str*) –
- **cellparams** (*float array*) –
- **tslsymmetry** (*str*) –

Returns

- **sgnum** (*int*)
- **==========**

1.4 pyEMsoftToolsLib

This page contains all the wrappped types and subroutines

pyEMsoftTools.**EMEBSDGlobalOptimizationUpdate** (*string*, *Opt*, *EBSD*, *Pattern*)
loadOptimizationData(string, Opt, EBSD, Pattern)

Update the template file for global optimization

Parameters

- **string** (*str*) –
- **Opt** (*class*) –
- **EBSD** (*class*) –
- **Pattern** (*class*) –

pyEMsoftTools.**EMEBSDnamelistUpdate** (*EBSD*)
Update EMEBSD nml file with EBSD namelist class

Parameters **EBSD** (*class*) –

pyEMsoftTools.**EMsoftPCtoPC** (*str*, *enl*)
enl = EMsoftPCtoPC(str, PC, enl)

Convert EMsoft pattern center to vendor's convention

Parameters

- **str** (*str*) –
- **PC** (*class*) –
- **enl** (*class*) –

Returns **enl**

Return type class

class pyEMsoftTools.**ExtractData** (*master*)
Module ExtractData

Extract hdf5 data from crystal structure file or master EBSD file

crystal_data()
Crystal, AtomDict, Info = crystal_data()

Extract everything about the crystal structure information

Returns

- **Crystal** (*dict*)
- **AtomDict** (*dict*)
- **Info** (*dict*)

ebsd_master()
master_info, master_pattern = ebsd_master()

Extract 2D master patterns (stereographic projection and Lambert projection) as well as related information

Returns

- **master_info** (*dict*)
- **master_pattern** (*float array*)

pyEMsoftTools.**PCtoEMsoftPC**(*str*, *PC*, *enl*)
enl = PCtoEMsoftPC(str, PC, enl)

Convert pattern center to EMsoft convention

Parameters

- **str** (*str*) –
- **PC** (*class*) –
- **enl** (*class*) –

Returns **enl**

Return type class

class pyEMsoftTools.**Tools**
Module Tools

Some tools to help with pyEMsoft module

static get_character_array(*ASCII_array*)
CArray = get_character_array(ASCII_array)

Convert numpy array (ASCII) data into character array

Parameters **ASCII_array** (*float array*) –

Returns **CArray**

Return type character array

static get_crystal_system_name(*crystal_system_number*)
crystal_system_name = get_crystal_system_name(crystal_system_number)

Get the corresponding name of the crystal system

Parameters **crystal_system_number** (*int*) –

Returns **crystal_system_name**

Return type str

static get_point_group(*SGNUM*)
pgnum = get_point_group(SGNUM)

Determine the point group a space group number

Parameters **SGNUM** (*int*) –

Returns pgnum

Return type int

static get_space_string(*chr*)
transspace_str=get_space_string(chr)

Get the corresponding name of the space in strings

Parameters **chr** (*str*) –

Returns transspace_str

Return type str

pyEMsoftTools.**circular_mask**(*ht*, *wd*)
mask = circular_mask(ht, wd)

Create a circular mask for a pattern

Parameters **binned** (*array*) –

Returns mask

Return type int array

pyEMsoftTools.**createAngleFile**(*emdatapath*, *angle_type*, *Pattern*, *AnglesMatrix*)
Create Euler angle file for EMEBSD program

Parameters

- **emdatapath** (*str*) –
- **angle_type** (*str*) –
- **Pattern** (*class*) –
- **AnglesMatrix** (*float array*) –

pyEMsoftTools.**getGrayscale**(*binned*)
bpatint = getGrayscale(binned)

Get grayscale pattern [0,255]

Parameters **binned** (*array*) –

Returns bpatint

Return type int array

pyEMsoftTools.**getSingleEBSDPattern**(*str*, *EBSD*, *Pattern*, *path*)
TargetPattern = getSingleEBSDPattern(str, EBSD, Pattern, path)

Get a single EBSD pattern from a pattern file

Parameters

- **str** (*str*) –
- **EBSD** (*class*) –
- **Pattern** (*class*) –

- **path** (*str*) –

Returns TargetPattern

Return type array

pyEMsoftTools.**loadBrukerHDF**(*str*, *path*)
enl, patterndata, angles = loadBrukerHDF(str, path)

Load metadata from BrukerHDF pattern file

Parameters

- **str** (*str*) –
- **path** (*str*) –

Returns

- **enl** (*class*)
- **patterndata** (*class*)
- **angles** (*float array*)

pyEMsoftTools.**loadBrukerHDFSEM**(*str*, *path*)
PatternQuality, SEM= loadBrukerHDFSEM(str, path)

Load pattern quality map and SEM array from BrukerHDF file

Parameters

- **str** (*str*) –
- **path** (*str*) –

Returns

- **PatternQuality** (*float array*)
- **SEM** (*nt array*)

pyEMsoftTools.**loadOptimizationData**(*emdatapath*, *inputtype*, *EBSD*, *Opt*)
EBSD_Opt, quaternion =loadOptimizationData(emdatapath, inputtype, EBSD)

Load global optimization output data with default output file name

Parameters

- **emdatapath** (*str*) –
- **inputtype** (*str*) –
- **EBSD** (*class*) –
- **Opt** (*class*) –

Returns

- **EBSD_Opt** (*class*)
- **quaternion** (*float array*)
- **Ftensor** (*float array*)

pyEMsoftTools.**loadPattern**(*inputtype*, *data*, *path*)
enl, patterndata, angles = loadPattern(inputtype, data, path)

Load metadata from pattern file

Parameters

- **inputtype** (*str*) –
- **data** (*class*) –
- **path** (*str*) –

Returns

- **enl** (*class*)
- **patterndata** (*class*)
- **angles** (*float array*)

pyEMsoftTools.**loadSEM**(*inputtype*, *data*, *path*)

PatternQuality, SEM = loadSEM(inputtype, data, path)

Load pattern quality map and SEM array from pattern file

Parameters

- **inputtype** (*str*) –
- **data** (*class*) –
- **path** (*str*) –

Returns

- **PatternQuality** (*float array*)
- **SEM** (*nt array*)

pyEMsoftTools.**loadTSLHDF**(*str*, *path*)

enl, patterndata, angles = loadBrukerHDF(str, path)

Load metadata from TSLHDF pattern file

Parameters

- **str** (*str*) –
- **path** (*str*) –

Returns

- **enl** (*class*)
- **patterndata** (*class*)
- **angles** (*float array*)

pyEMsoftTools.**loadTSLHDFSEM**(*str*, *path*)

PatternQuality, SEM= loadTSLHDFSEM(str, path)

Load pattern quality map and SEM array from TSLHDF file

Parameters

- **str** (*str*) –
- **path** (*str*) –

Returns

- **PatternQuality** (*float array*)
- **SEM** (*nt array*)

CHAPTER 2

License

CHAPTER 3

Indices

- genindex
- search

Python Module Index

p

Index

D

E

F

G

H

I

J

K

L

R

S

T

U

V

W

Z