PYATB v1.1.2

PYATB

Dec 01, 2025

QUICK START

CHAPTER

ONE

INTRODUCTION

PYATB (Python ab initio tight binding simulation package) is an open-source software package designed for computing electronic structures and related properties based on the ab initio tight binding Hamiltonian. The Hamiltonian can be directly obtained after conducting self-consistent calculations with first-principles packages using numerical atomic orbital (NAO) bases, such as ABACUS. The package comprises three modules - Bands, Geometric, and Optical, each providing a comprehensive set of tools for analyzing different aspects of a material's electronic structure.

The Bands module enables users to calculate essential properties of band structures, including the partial density of states (PDOS), fat bands, Fermi surfaces, Wyel/Dirac points, and more. Additionally, the band unfolding method is utilized to obtain the energy band spectra of a supercell by projecting the electronic structure of the supercell onto the Brillouin zone of the primitive cell.

In the Geometric module, users have access to tools for computing Berry phase and Berry curvature-related quantities, such as electric polarization, Wilson loops, Chern numbers, anomalous Hall conductivities, and more.

The Optical module offers a range of optical property calculations, including optical conductivity and nonlinear optical responses, such as shift current and Berry curvature dipole.

PYATB is licensed under GPLv3.0, making it freely available for use and modification by the scientific community. The main developers of PYATB are Gan Jin, Hongsheng Pang, Yuyang Ji, Zujian Dai, Xudong Zhu, under the supervision of

Prof. Lixin He at the University of Science and Technology of China.

1.1 Capabilities

PYATB provides three major modules: *Bands module*, *Geometric module*, *Optical module*, each with its own set of functions:

- Bands module
  - **Band structre**
    Allows users to calculate the energy bands and wave functions using three different **k**-point modes: k-point, k-line, and k-mesh.
  - **Band unfolding**
    Calculates the spectra weight by unfolding the energy bands of a supercell into the Brillouin zone (BZ) of the primitive cell.
  - **Fermi energy**
    Calculates the Fermi energy at a given temperature.
  - **Fermi surface**
    Plots the Fermi surface.
  - **Find nodes**
    Allows users to search for degenerate points of the energy bands in the BZ within a specified energy window. This function can be used to find the Weyl/Dirac points in Weyl/Dirac semimetals.
  - **DOS and PDOS**
    Calculates the density of states (DOS) and partial density of states (PDOS) of particular orbitals.
  - **Fat band**
    Provides the contribution of each atomic orbital to the electronic wave functions at each **k**-point in the BZ.
  - **Spin texture**
    Plots the spin polarization vector as a function of momentum in the BZ.
  - **Surface states**
    Calculates the spectral weights using the iterative surface Green's function method.
- Geometric module
  - **Wilson loop**
    Enables users to calculate the $\mathbb{Z}_2$ number by tracking the Wannier centers along the Wilson loop.
  - **Electric polarization**
    Evaluates the electric polarization in various directions for non-centrosymmetric materials based on the Berry phase theory.
  - **Berry curvature**
    Computes the Berry curvature in the BZ.
  - **Anomalous Hall conductivity**
    Calculates the anomalous Hall conductivity (AHC) using Berry curvature.
  - **Spin Hall conductivity**
    Calculates the spin Hall conductivity (SHC) using the band-projected Berry curvature-like term.
  - **Anomalous Nernst conductivity**
    Calculates the anomalous Nernst conductivity (ANC) using the Berry-curvature–related thermoelectric kernel.

- **Chern number**
  Calculates the Chern number of a system for any given **k**-plane.
- **Chirality**
  Examines the chirality of Weyl points by calculating the Berry curvature on a sphere around the **k** point.
- **Orbital magnetization**
  Calculates the orbital magnetization using the Berry-phase formulation.

- Optical module
  - **JDOS**
    Calculates the joint density of states (JDOS), which characterizes both electronic states and optical transitions.
  - **Optical conductivity and dielectric function**
    Calculates the frequency-dependent optical conductivity and dielectric function.
  - **Shift current**
    Calculates the shift current conductivity tensor for the bulk photovoltaic effect.
  - **Berry curvature dipole**
    Calculates the Berry curvature dipole which leads to the nonlinear anomalous Hall effects.
  - **Second harmonic generation**
    Calculates the second-harmonic generation susceptibility using Berry-connection–based nonlinear optical matrix elements.

1.2 Methodology

PYATB is based on the ab initio tight binding model, where the parameters of the Hamiltonian are generated directly from the self-consistent calculations using first-principles software based on numerical atomic orbitals (NAO) bases, such as ABACUS.

NOTE: ABACUS (Atomic-orbital Based Ab-initio Computation at UStc) is an open-source package based on density functional theory (DFT). For a detailed introduction, please refer to https://abacus.ustc.edu.cn/.

In a periodic system, the Kohn–Sham equation at a given **k** point can be written as,

$$H|\Psi_{n\mathbf{k}}\rangle = E_{n\mathbf{k}}|\Psi_{n\mathbf{k}}\rangle.$$

Here $\Psi_{n\mathbf{k}}$ is the Bloch wave function of the n-th band, and can be expressed under NAO as,

$$|\Psi_{n\mathbf{k}}\rangle = \frac{1}{\sqrt{N}}\sum_{\mu} C_{n\mu}(\mathbf{k})\sum_{\mathbf{R}} \mathrm{e}^{i\mathbf{k}\cdot\mathbf{R}}|\mathbf{R}\mu\rangle,$$

where $|\mathbf{R}\mu\rangle \equiv \phi_\mu(\mathbf{r}-\tau_\mu-\mathbf{R})$ is the μ-th atomic orbital, in the $\mathbf{R}$-th unit cell, and τ_μ denotes the center position of this orbital. The composite index $\mu=(\alpha,i,\zeta,l,m)$, where α is the element type, i is the index of the atom of each element type, ζ is the multiplicity of the radial functions for the angular momentum l, and m is the magnetic quantum number. The coefficient of the NAO is given by $C_{n\mu}(\mathbf{k})$.

Under the NAO base, the Kohn-Sham equation becames a eigenvalue problem,

$$H(\mathbf{k})C_n(\mathbf{k}) = E_{n\mathbf{k}}S(\mathbf{k})C_n\ .$$

where $H(\mathbf{k})$, $S(\mathbf{k})$ and $C_n(\mathbf{k})$ are the Hamiltonian matrix, overlap matrix and eigenvectors of the n-th band, respectively.

To obtain the $H(\mathbf{k})$ and $S(\mathbf{k})$, we first calculate tight binding Hamiltonian in real space via first-principles softwares based on NAOs, such as ABACUS,

$$\begin{aligned}H_{\nu\mu}(\mathbf{R}) &= \langle \mathbf{0}\nu|H|\mathbf{R}\mu\rangle\,, \\ S_{\nu\mu}(\mathbf{R}) &= \langle \mathbf{0}\nu|\mathbf{R}\mu\rangle\,.\end{aligned}$$

Once we have the $H_{\nu\mu}(\mathbf{R})$ and $S_{\nu\mu}(\mathbf{R})$, we can obtain the Hamiltonian matrix and the overlap matrix at arbitrary **k** points using the following relation,

$$\begin{aligned}H_{\nu\mu}(\mathbf{k}) &= \sum_{\mathbf{R}} \mathrm{e}^{i\mathbf{k}\cdot\mathbf{R}} H_{\nu\mu}(\mathbf{R})\,, \\ S_{\nu\mu}(\mathbf{k}) &= \sum_{\mathbf{R}} \mathrm{e}^{i\mathbf{k}\cdot\mathbf{R}} S_{\nu\mu}(\mathbf{R})\,,\end{aligned}$$

When solving for the geometric properties of the bands, we also need the dipole matrix between the NAOs, namely:

$$r_{\nu\mu,a}(\mathbf{R}) = \langle \mathbf{0}\nu|r_a|\mathbf{R}\mu\rangle \quad a = x, y, z.$$

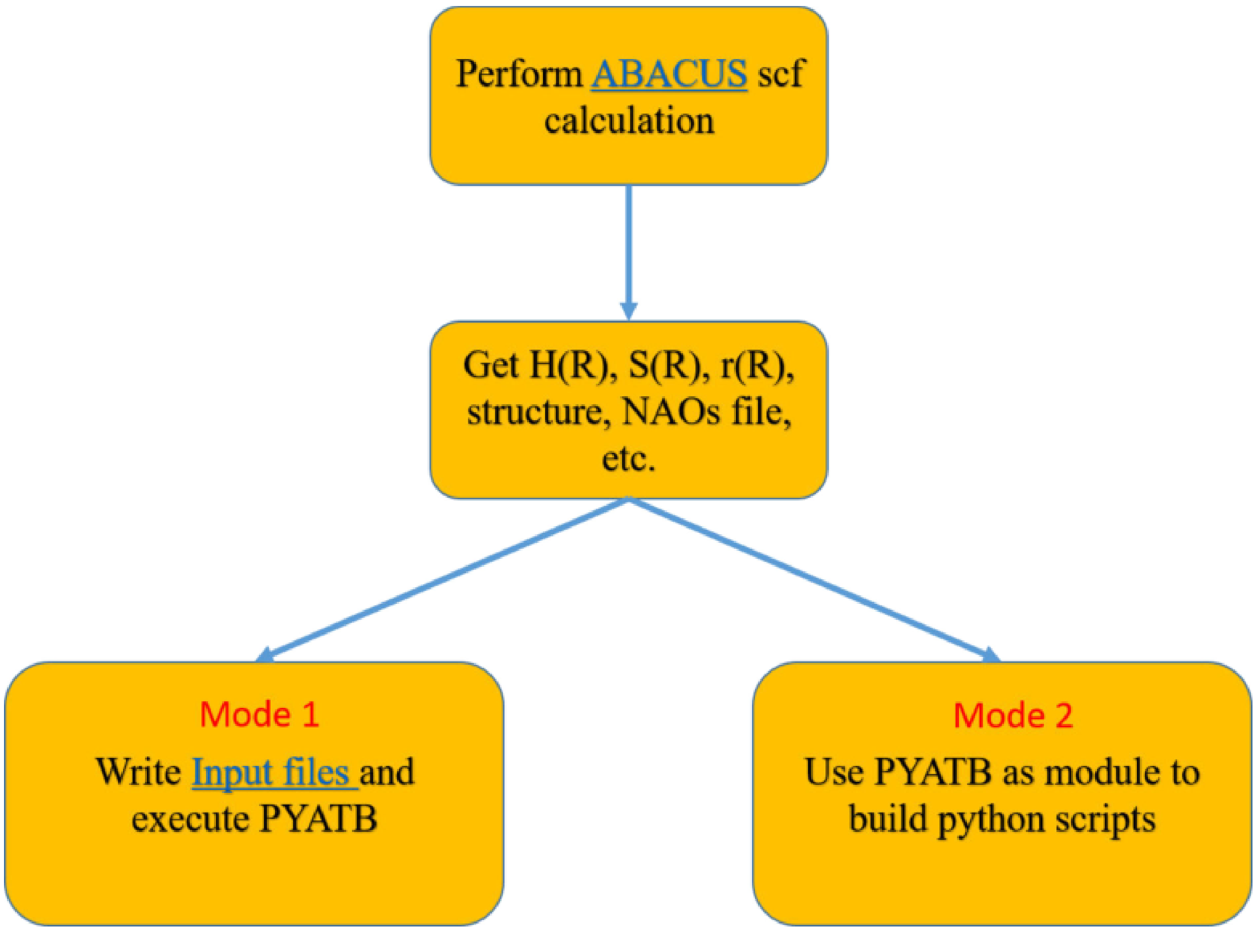
The dipole matrix $\mathbf{A}^R_{\nu\mu}(\mathbf{k})$ can then be obtained by Fourier transform,

$$A^R_{\nu\mu,a}(\mathbf{k}) = \sum_{\mathbf{R}} \mathrm{e}^{i\mathbf{k}\cdot\mathbf{R}} r_{\nu\mu,a}(\mathbf{R}).$$

After obtaining the tight binding parameters $H_{\nu\mu}(\mathbf{R})$, $S_{\nu\mu}(\mathbf{R})$, and $r_{\nu\mu,a}(\mathbf{R})$ from the first principles software, the electronic structures and related physical propertie can be calculated using PYATB. More description of the methods used in PYATB can be found in Ref. ???

1.3 Workflow

The workflow of PYATB involves several key steps. First,one need to perform self-consistent calculations using ABAUCS to generate tight binding Hamiltonan, including $H_{\nu\mu}(\mathbf{R})$, $S_{\nu\mu}(\mathbf{R})$, and $r_{\nu\mu,a}(\mathbf{R})$. Some of the functions may require crystal structur file and atomic orbital data. Once all the necessary files are obtained, the user can either write an `Input` file to perform the corresponding function computation using PYATB or write a script utilizing the PYATB module to carry out the calculations.


Perform ABACUS scf calculation
Get H(R), S(R), r(R), structure, NAOs file, etc.
Mode 1
Write Input files and execute PYATB
Mode 2
Use PYATB as module to build python scripts

CHAPTER TWO

INSTALLATION

2.1 Install via PyPI

To install **PYATB** from PyPI, ensure that `mpi4py` is available in your Python environment. We recommend installing it via conda:

```
conda install -c conda-forge mpi4py
pip install pyatb
```

2.2 Install from Source

The latest development version of PYATB can be obtained from the official GitHub repository:

```
git clone https://github.com/pyatb/pyatb.git
```

2.2.1 Prerequisites

Currently, PYATB is only supported on Linux systems and has not been tested on Windows or Mac systems. To use PYATB, the following prerequisites must be installed:

- Python 3.8 or newer
- C++ compiler
- pybind11
- Eigen3
- NumPy
- SciPy
- mpi4py
- Matplotlib
- ASE

2.2.2 Building Wheel Files (.whl)

PYATB provides a convenient Docker-based workflow for building **manylinux-compliant wheel packages**.

The `build_wheels.sh` script will automatically create a Docker environment and compile the wheel files:

```
./build_wheels.sh
```

If you need to build wheels for a specific Python version, modify the `CIBW_BUILD` variable inside the script. For example:

```
CIBW_BUILD="cp313-manylinux_x86_64"
```

will generate wheel files for **Python 3.13**.

CHAPTER

THREE

RUN

PYATB supports mixed parallelism of OpenMP and MPI, and you need to determine the number of threads and processes to run depending on the actual configuration of your computer.

For example, set the number of threads to 2,

```
export OMP_NUM_THREADS=2
```

and then use 6 processes to run PYATB,

```
mpirun -n 6 pyatb
```

CHAPTER

FOUR

BRIEF INTRODUCTION OF THE INPUT FILES

4.1 Input

The Input file describes the basic information about the system and the parameters required to calculate the function. For a complete list of the input parameters, please consult this *input list*.

The following is an example of a Input file:

```
INPUT_PARAMETERS
{
    nspin                          4
    package                        ABACUS
    fermi_energy                   9.481194886038594
    fermi_energy_unit              eV
    HR_route                       data-HR-sparse_SPIN0.csr
    SR_route                       data-SR-sparse_SPIN0.csr
    rR_route                       new-data-rR-tr_SPIN4
    HR_unit                        Ry
    rR_unit                        Bohr
    max_kpoint_num                 8000
}

LATTICE
{
    lattice_constant               1.8897162
    lattice_constant_unit          Bohr
    lattice_vector
    -2.069  -3.583614  0.000000
     2.069  -3.583614  0.000000
     0.000   2.389075  9.546667
}

BAND_STRUCTURE
{
    wf_collect                     0
    kpoint_mode                    line
    kpoint_num                     5
    high_symmetry_kpoint
    0.00000 0.00000 0.0000 100   # G
    0.00000 0.00000 0.5000 100   # Z
    0.50000 0.50000 0.0000 100   # F
    0.00000 0.00000 0.0000 100   # G
    0.50000 0.00000 0.0000 1     # L
}
```

The `Input` file consists of three main sections (**INPUT_PARAMETERS**, **LATTICE**, **FUNCTION**), each of which is represented by the `keyword + {}`, namely:

```
KEYWORD
{
    parameter  xx
}
```

The **INPUT_PARAMETERS** section is utilized to specify public parameters and indicate the HR, SR, and rR files required for the calculation.

The **LATTICE** section outlines lattice information and includes three parameters: lattice_constant, lattice_constant_unit, and lattice_vector.

The **FUNCTION** section lists the parameters necessary for each calculation function. These parameters differ depending on the specific function being used.

NOTE:

- The keyword of the section must be capitalized, and the parameters in the section must be lowercase.
- Each parameter value is provided by specifying the name of the input variable and then putting the value after the name, separated by one or more blank characters(space or tab).
- Any line starting with `#` or `//` will be ignored. To ignore a function section, add `#` in front of the section name. This will cause all parameters within that function section to be ignored.
- Proper case is required to avoid reading errors. For example, `eV` must not be written as `ev`, and `Bohr` must not be written as `bohr`.

4.2 HR, SR, rR

HR, SR, and rR are important input files for PYATB and contain information about the tight binding model, which can be generated by ABACUS.

In the PYATB program, if only the *Bands module* is used, it is enough to provide HR and SR, and specifying rR is not required. However, when the functions of the calculations are related to the *Geometric module* or *Optical module*, the rR file must be specified.

4.2.1 How to generate HR, SR, rR files using ABACUS

Run the **scf** calculation of ABACUS normally, specifies the parameters `out_mat_hs2` and `out_mat_r` in the `INPUT` file, and three files of sparse format `data-HR-sparse_SPIN0.csr`, `data-SR-sparse_ SPIN0.csr`, `data-rR-sparse.csr` (and `data-HR-sparse_SPIN1.csr` when `nspin` is 2), will be generated in OUT* folder.

NOTE:

- nspin=4 and nspin=1, nspin=2 output different files. nspin=4 files contain imaginary numbers, so the correct nspin parameter needs to be strictly specified when executing pyatb programs.

4.2.2 Format description of HR, SR, rR

We use the file `data-HR-sparse_SPIN0.csr` as an example to explain the format of HR. This file contains a series of $H(R)$ matrices, where each R corresponds to an $H(R)$ matrix. The specific format of the file is as follows:

1. The dimension of the $H(R)$ matrix is recorded on the first line, for example, `Matrix Dimension of H(R): 270`, indicating that each $H(R)$ matrix is a 270*270 matrix.
2. The number of R is recorded on the second line, for example, `Matrix number of H(R): 97`, indicating that there are a total of 97 $H(R)$ matrices.
3. Starting from the third line, every 4 lines form a group, recording the sparse matrix of $H(R)$. Following the example above, the file still has 4*97 lines left, divided into 97 groups.
   - The first line of each group contains 4 integers. The first three numbers represent the `(x, y, z)` fractional coordinates of R, and the last number represents the number of non-zero elements in the sparse matrix.
   - The second to fourth lines of each group record the `data`, `indices`, and `indptr` information of the sparse matrix in `csr` format.

The sparse matrix `csr` format is explained below, with reference to `Wikipedia`. Please refer to Sparse matrix.

We will use the following matrix as an example:

$$\begin{bmatrix} 4 & 2 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 4 & 5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The sparse matrix in `csr` format has three sets of data, corresponding to three lines in the output file. The first line contains the non-zero element `data`, the second line contains the column indices of the non-zero elements `indices`, and the third line contains the row offset `indptr`.

Note that our counting starts from 0.

First, the first row of the matrix is `4 2 0 0`, which has 2 non-zero elements `4 2`, with corresponding column indices `0 1`. The second row of the matrix is `0 3 0 0`, which has 1 non-zero element `3`, with corresponding column index `1`. The third row of the matrix has 2 non-zero elements `4 5`, with corresponding column indices `1 2`. The fourth row is all zeros and has no non-zero elements.

Therefore, we have:

```
data = [4, 2, 3, 4, 5]   # non-zero elements
indices = [0, 1, 1, 1, 2]   # column indices of each non-zero element

'''
The number of non-zero elements in each row from the first to
the fourth row is 2, 1, 2, 0, respectively;
the corresponding offset values are 0, 2, 3, 5, 5.
These values are used to record the starting offset of each
row's first element in the data array, and can also be used
for the indices array. The number of non-zero elements in
each row can be obtained by subtracting the previous offset value
from the current offset value, i.e., 2-0=2, 3-2=1, 5-3=2, 5-5=0.
'''

# the dimension of this array must be the number of rows of the matrix plus 1.
indptr = [0, 2, 3, 5, 5]
```

NOTE:

- When the `npsin` parameter in the PYATB `Input` file is set to `1` or `2`, the HR matrix read is purely real. When `nspin` is set to `4`, the HR matrix read is complex, with matrix elements in the format of `(real part, imaginary part)`.

The file format for the SR matrix is identical to that of the HR matrix. The rR matrix format is slightly different because it includes the r_x, r_y, and r_z directions. We use `data-rR-sparse.csr` as an example to illustrate the rR file format.

1. The first line is `Matrix Dimension of r(R): 270`.
2. The second line is `Matrix number of r(R): 97`.
3. The third line consists of three numbers, representing the fractional coordinates `(x, y, z)` of R.
4. Starting from the fourth line, every 4 lines form a group, with a total of three groups, recording the $r_x(R)$, $r_y(R)$, and $r_z(R)$ matrices respectively. After that, the loop is performed for different R.
   - The first line of each group is an integer, recording the number of non-zero elements in the sparse matrix.
   - The second to fourth lines of each group are in the `csr` format of the sparse matrix, recording the `data`, `indices`, and `indptr` information respectively.

4.3 Other file

For the calculation of some functions, structure files and NAOs (numerical atomic orbital bases) files are also required.

- The structure file

  This file describes the structural information about the system, e.g., lattice constant, lattice vectors, and positions of the atoms within a unit cell. The structure file format of ABACUS is currently used.

- The NAOs files

  This file describes the numerical atomic orbitals used to expand the Hamiltonian. The NAOs file format of ABACUS is currently used.

CHAPTER

FIVE

FUNCTIONS

5.1 BAND_STRUCTURE

An example of calculating the energy band structure of the topological insulator Bi_2Se_3 is provided in the folder `examples/Bi2Se3`.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                              4
    package                            ABACUS
    fermi_energy                       9.557219691497478
    fermi_energy_unit                  eV
    HR_route                           data-HR-sparse_SPIN0.csr
    SR_route                           data-SR-sparse_SPIN0.csr
    rR_route                           data-rR-sparse.csr
    HR_unit                            Ry
    rR_unit                            Bohr
    max_kpoint_num                     8000
}

LATTICE
{
    lattice_constant                   1.8897162
    lattice_constant_unit              Bohr
    lattice_vector
    -2.069  -3.583614  0.000000
     2.069  -3.583614  0.000000
     0.000   2.389075  9.546667
}

BAND_STRUCTURE
{
    wf_collect                         0
    kpoint_mode                        line
    kpoint_num                         5
    high_symmetry_kpoint
    0.00000 0.00000 0.0000 100   # G
    0.00000 0.00000 0.5000 100   # Z
    0.50000 0.50000 0.0000 100   # F
    0.00000 0.00000 0.0000 100   # G
    0.50000 0.00000 0.0000 1     # L
}
```

wf_collect: Whether to output wave function matrix information. The wave function file stores the expansion coefficients of NAOs.

There are three ways to set k-points: k-point, k-line, and k-mesh, with the keyword kpoint_mode used to define the mode. The setting parameters for each mode differ, so please refer to the INPUT for detailed instructions. In this example, the line mode is used. In this mode, kpoint_num specifies the number of high-symmetry points, and high_symmetry_kpoint records the direct coordinates of these points and the number of k-points between each pair of high-symmetry points. Each row in the setting consists of four numbers, where the first three indicate the coordinates, and the last number specifies the number of k-points between the given k-point and the next high symmetry point.

After the calculation is done, a number of files will be generated in the Out/Band_Structure folder, including kpt.dat and band.dat. If kpoint_mode is set to line, the output will also include the energy band diagram and its corresponding drawing script.

5.2 BANDUNFOLDING

5.2.1 introduction

Band unfolding is a method of projecting the wave function of the supercell to the coupled **k** points in the original unit cell to obtain the spectral function.

The relationships between the lattice vectors of the large cell (A) and primitive cell (a) are given by

$$\begin{bmatrix} A_1 \\ A_1 \\ A_3 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Detailed descriptions can be found in First-principles calculations of the surface states of doped and alloyed topological materials via band unfolding method.

5.2.2 example

Here is an example located in the examples/NV folder that showcases how to calculate the spectral function of the nitrogen-vacancy (NV) center in diamond.

The Input file is:

```
INPUT_PARAMETERS
{
    nspin                           1
    package                         ABACUS
    fermi_energy                    15.46412271260700
    fermi_energy_unit               eV
    HR_route                        data-HR-sparse_SPIN0.csr
    SR_route                        data-SR-sparse_SPIN0.csr
    rR_route                        data-rR-sparse.csr
    HR_unit                         Ry
    rR_unit                         Bohr
}

LATTICE
{
    lattice_constant                1.8897162
```

(continues on next page)

(continued from previous page)

```
    lattice_constant_unit          Bohr
    lattice_vector
    7.13366 0 0 #latvec1
    0 7.13366 0 #latvec2
    0 0 7.13366 #latvec3
}

BANDUNFOLDING
{
    stru_file                      STRU
    ecut                           60
    band_range                     10 250
    m_matrix                       -2 2 2 2 -2 2 2 2 -2
    kpoint_mode                    line
    kpoint_num                     5
    high_symmetry_kpoint
    0.500000  0.000000  0.500000 300  # X
    0.500000  0.250000  0.750000 300  # W
    0.500000  0.500000  0.500000 300  # L
    0.000000  0.000000  0.000000 300  # Gamma
    0.500000  0.000000  0.500000 1    # X
}
```

`stru_file`: The structure file name of the supercell. This file indicates the crystal structure of the supercell and the corresponding orbital file. Make sure that both the structure file and the orbital file exist.

`ecut`: Determine the number of projections to the plane wave basis group, the energy unit is Ry.

`band_range`: Specify the energy band range of band unfolding.

`m_matrix`: Transformation matrix of supercell and primitive cell lattice vector

The k-point setting is determined according to the structure of the original cell, not the k-point in the supercell. For the k point setting of this function, please refer to the `kpoint_mode` module.

Once the task calculation is finished, three files will be generated in the `Out/Bandunfolding` folder. These files include `kpt.dat` and `spectral_weight.dat`, representing the k-point and spectral function of the original cell, respectively. Additionally, a drawing script called `plot_unfold.py` will also be included.

NOTE: When calculating a slab material, it is necessary to remove the thickness of the vacuum layer and make necessary modifications to the crystal vector and atomic positions specified in the `LATTICE` and structure files. Failing to make these modifications accurately can result in inaccurate results.

5.3 FERMI_ENERGY

5.3.1 introduction

This function is to calculate the Fermi energy of solid materials given temperature and electronic occupation number. If the system is an insulator, fermi energy is given by the valence band maximum (VBM) .

For each **k** point, the probability of finding an electron in any energy state should obey the Fermi-Dirac distribution. The integration of occupied electrons over the entire Brillouin zone should be the occupation number. Though which, the exact Fermi energy could be attained following Newton interpolation.

$$f(E, E_f, T) = \frac{1}{1 + e^{\left(\frac{E - E_f}{k_B T}\right)}}$$

$$N[E_f] = \int_{BZ} [d\mathbf{k}] \sum_n f(E_n, E_f, T)$$

5.3.2 example

You can find an example of how to calculate the Fermi energy of Copper in the `examples/Cu` folder.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                   1
    package                 ABACUS
    fermi_energy            Auto
    fermi_energy_unit       eV
    HR_route                data-HR-sparse_SPIN0.csr
    SR_route                data-SR-sparse_SPIN0.csr
    HR_unit                 Ry
    rR_unit                 Bohr
    max_kpoint_num          8000
}

LATTICE
{
    lattice_constant        6.91640
    lattice_constant_unit   Bohr
    lattice_vector
    0.50        0.50         0.00
    0.50        0.00         0.50
    0.00        0.50         0.50
}

FERMI_ENERGY
{
    temperature             0
    electron_num            11
    grid                    50 50 50
    epsilon                 1e-4
}
```

`electron_num`: The number of the electrons in the system. The number of valence electrons, which is the sum of all atomic valence electrons of the system, can be obtained from the self-consistent output of the first-principles software.

`epsilon`: The max tolerable error of Newton interpolation. If two steps of Newton interpolation differs less than this epsilon, the calculation would stop and output the result.

`temperature`: The temperature of the system, unit in K.

After the task calculation is completed, there will be one file in the `Out/Fermi_Energy` folder, namely `fermi_energy.dat`, showing the calculated Fermi energy.

5.4 FERMI_SURFACE

5.4.1 introduction

The Fermi surface is a function that calculates the iso-energy surface of a given energy level. If the given energy level is the Fermi energy, then it will plot the Fermi surface.

5.4.2 example

You can find an example of how to calculate the Fermi surface of Copper in the `examples/Cu` folder.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                   1
    package                 ABACUS
    fermi_energy            Auto
    fermi_energy_unit       eV
    HR_route                data-HR-sparse_SPIN0.csr
    SR_route                data-SR-sparse_SPIN0.csr
    HR_unit                 Ry
    rR_unit                 Bohr
    max_kpoint_num          8000
}

LATTICE
{
    lattice_constant        6.91640
    lattice_constant_unit   Bohr
    lattice_vector
    0.50        0.50        0.00
    0.50        0.00        0.50
    0.00        0.50        0.50
}

FERMI_ENERGY
{
    temperature             0
    electron_num            11
    grid                    50 50 50
    epsilon                 1e-4
}

FERMI_SURFACE
{
    bar                     1e-5
    kpoint_mode             mp
    k_start                 0 0 0
    k_vect1                 1 0 0
    k_vect2                 0 1 0
    k_vect3                 0 0 1
    mp_grid                 50 50 50
}
```

`bar`: The max tolerable error bar for the Fermi surface.

nbands: If you know the energy band range where the Fermi energy is located, you can set this parameter to speed up the calculation. There are two numbers in total, indicating the range of the energy band. The default value is 0 0, that is, all energy bands are considered.

For the k point setting of this function, please refer to the kpoint_mode module.

After the task calculation is completed, there will be two files in the Out/Fermi_Surface folder, namely fermi_surface_kpt.dat and plot_fermi_surface.py, corresponding to the k-point found on the Fermi surface and a plotting script.

NOTE: To visualize iso-energy surface at specific energy levels, the fermi_energy parameter in INPUT_PARAMETERS can be customized.

5.5 FIND_NODES

5.5.1 introduction

Find nodes is a method of finding k points with degenerate energy bands in a given energy range. This function can be used to find the Weyl/Dirac points in Weyl/Dirac semimetals.

5.5.2 example

An example of how to locate the Weyl point of $MnSb_2Te_4$ is provided in the examples/MnSb2Te4 folder.

The Input file is:

```
INPUT_PARAMETERS
{
    nspin                           4
    package                         ABACUS
    fermi_energy                    9.9700823666762375
    fermi_energy_unit               eV
    HR_route                        data-HR-sparse_SPIN0.csr
    SR_route                        data-SR-sparse_SPIN0.csr
    rR_route                        data-rR-sparse.csr
    HR_unit                         Ry
    rR_unit                         Bohr
}

LATTICE
{
    lattice_constant                1.8897162
    lattice_constant_unit           Bohr
    lattice_vector
     4.2885731815169859   -0.0001203117360283   -0.0000592245216563
     2.1441823977387200    3.7140734770500492   -0.0000592245216895
     2.1439936829776851    1.2378353300160352   13.4147375436875418
}

FIND_NODES
{
    energy_range                    9.870 10.070
    k_start                         0.0 0.0 -0.2
    k_vect1                         0.0 0.0  0.0
    k_vect2                         0.0 0.0  0.0
```

(continues on next page)

(continued from previous page)

```
    k_vect3                         0.0 0.0  0.4
    initial_grid                    1  1  100
    initial_threshold               0.01
    adaptive_grid                   1  1  20
    adaptive_threshold              0.001
}
```

`energy_range`: The energy range in which the program searches for degenerate points, the energy unit is eV.

Set the search space of k points: Selecting a parallel hexahedron in k-space requires an origin (`k_start`) and three vectors (`k_vect1`, `k_vect2`, `k_vect3`) that are not parallel to each other in general. In this example, `k_vect2` and `k_vect3` are zero vectors, so the chosen search space is k-line from (0.0, 0.0, -0.2) to (0.0, 0.0, 0.4).

After the task calculation is completed, there will be three files in the `Out/Find_Nodes` folder, namely `nodes.dat`, `plot_nodes.py` and `nodes.pdf`, corresponding to the degenerate k-point(s) in direct coordinate and a plotting script with its plot.

5.6 PDOS

5.6.1 introduction

The distribution of electronic states at various energies is characterized by the density of states (DOS), while the partial density of states (PDOS) is a useful tool for analyzing the contribution of individual atomic orbitals to the DOS.

The implementation of PDOS is given by

$$g_{\mu}(E) = \frac{1}{N_{\mathbf{k}}} \sum_{\mathbf{k}} \sum_{n} \sum_{\nu} C_{n\nu}^{*}(\mathbf{k}) S_{\nu\mu}(\mathbf{k}) C_{n\mu}(\mathbf{k}) \delta(E - E_{n\mathbf{k}}),$$

where $C_{n\mu}(\mathbf{k})$ is the coefficient of the NAO.

5.6.2 example

An example (refer to folder `examples/Si2`) of calculating the PDOS of the diamond Si is given here.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                   1
    package                 ABACUS
    fermi_energy            6.389728305291531
    fermi_energy_unit       eV
    HR_route                data-HR-sparse_SPIN0.csr
    SR_route                data-SR-sparse_SPIN0.csr
    rR_route                data-rR-sparse.csr
    HR_unit                 Ry
    rR_unit                 Bohr
}

LATTICE
{
    lattice_constant            1.8897162
```

(continues on next page)

(continued from previous page)

```
    lattice_constant_unit   Bohr
    lattice_vector
    0.000000000000  2.715000000000  2.715000000000
    2.715000000000  0.000000000000  2.715000000000
    2.715000000000  2.715000000000  0.000000000000
}

PDOS
{
    stru_file      STRU
    e_range        -5.0 17.0
    de             0.01
    sigma          0.07
    kpoint_mode    mp
    mp_grid        12 12 12
}
```

`stru_file`: The structure file name of the supercell. This file indicates the crystal structure of the supercell and the corresponding orbital file. Make sure that both the structure file and the orbital file exist.

`e_range`: Specify the energy range of dos, the unit is eV.

`de`: specifies the energy interval.

`sigma`: Specify the parameters of Gaussian smearing.

For the k point setting of this function, please refer to the `kpoint_mode` module.

After the task calculation is completed, there will be three files in the `Out/PDOS` folder, namely `TDOS.dat` and `PDOS.xml`, `plot_dos.py`. Specify the projected atomic orbital index in the plot script, and then draw the PDOS plot.

5.7 FAT_BAND

5.7.1 introduction

A fat band can provide information about the contributions of specific atomic orbitals or groups of orbitals to the electronic bands of a material at given k points.

5.7.2 example

In the `examples/Si2` folder, there is an example of how to calculate the fat band of the diamond Si.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                 1
    package               ABACUS
    fermi_energy          6.389728305291531
    fermi_energy_unit     eV
    HR_route              data-HR-sparse_SPIN0.csr
    SR_route              data-SR-sparse_SPIN0.csr
    rR_route              data-rR-sparse.csr
    HR_unit               Ry
```

(continues on next page)

(continued from previous page)

```
    rR_unit             Bohr
}

LATTICE
{
    lattice_constant        1.8897162
    lattice_constant_unit   Bohr
    lattice_vector
    0.000000000000  2.715000000000  2.715000000000
    2.715000000000  0.000000000000  2.715000000000
    2.715000000000  2.715000000000  0.000000000000
}

FAT_BAND
{
    band_range                      1 8
    stru_file                       STRU
    kpoint_mode                     line
    kpoint_num                      5
    high_symmetry_kpoint
    0.50000  0.50000 0.5000 100  # L
    0.00000  0.00000 0.0000 100  # G
    0.50000  0.00000 0.5000 100  # X
    0.37500 -0.37500 0.0000 100  # K
    0.00000  0.00000 0.0000 1    # G
}
```

`band_range`: There are two numbers (separated by spaces) to indicate which bands are selected for projection, counting from 1.

`stru_file`: The structure file name. This file indicates the crystal structure and the corresponding orbital file. Make sure that both the structure file and the orbital file exist.

For the k point setting of this function, please refer to the `kpoint_mode` module.

Once the task calculation is finished, you will find four files in the `Out/Fat_Band` folder. These files include `band.dat` and `pband.dat`, `fatband.xml`, and `plot_fatband.py`. They contain valuable information about the original bands, the coefficients of the bands projected onto each atomic orbital (the number of atomic orbitals is equal to the number of basis sets), an XML formatted file of the projected bands, and a script to visualize the fat band.

5.8 SPIN_TEXTURE

5.8.1 introduction

Spin texture refers to the spatial distribution of electron spins in momentum space. In PYATB, the spin texture is calculated as follows,

$$\langle\Psi_{n\mathbf{k}}|\hat{\sigma}_i|\Psi_{n\mathbf{k}}\rangle = \sum_{\mu,\nu,s,s'} C^*_{n,\mu s}(\mathbf{k}) S_{\mu\nu,ss'}(\mathbf{k}) \hat{\sigma}_{i,ss'} C_{n,\nu s'}(\mathbf{k}),$$

where $\hat{\sigma}_i$ are the Pauli matrices, with i= x, y, z, and s=$\uparrow$, $\downarrow$ is the spin index.

5.8.2 example

Here, we provide an example of calculating the spin texture of WSSe (refer to folder `tutorial/WSSe_spin_texture`).

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                           4
    package                         ABACUS
    fermi_energy                    1.4232523137
    fermi_energy_unit               eV
    HR_route                        ../abacus/OUT.WSSe/data-HR-sparse_SPIN0.csr
    SR_route                        ../abacus/OUT.WSSe/data-SR-sparse_SPIN0.csr
    rR_route                        ../abacus/OUT.WSSe/data-rR-sparse.csr
    HR_unit                         Ry
    rR_unit                         Bohr
    max_kpoint_num                  4000
}

LATTICE
{
    lattice_constant                1.8897162
    lattice_constant_unit           Bohr
    lattice_vector
    3.2521424294         0.0000000000         0.0000000000
    1.6260712147         2.8164379605         0.0000000000
    0.0000000000         0.0000000000        18.2324867249
}

BAND_STRUCTURE
{
    wf_collect                0
    kpoint_mode               line
    kpoint_num                7
    high_symmetry_kpoint
    0.0000000000   0.0000000000   0.0000000000  100    # G
    0.0000000000   0.5000000000   0.0000000000  100    # M'
    0.3333333333   0.6666666666   0.0000000000  100    # K'
    0.0000000000   0.0000000000   0.0000000000  100    # G
    0.5000000000   0.5000000000   0.0000000000  100    # M
    0.6666666666   0.3333333333   0.0000000000  100    # K
    0.0000000000   0.0000000000   0.0000000000  1      # G
    kpoint_label                    G,M',K',G,M,K,G
}

SPIN_TEXTURE
{
    band_range                35 38
    kpoint_mode               line
    kpoint_num                7
    high_symmetry_kpoint
    0.0000000000   0.0000000000   0.0000000000  100    # G
    0.0000000000   0.5000000000   0.0000000000  100    # M'
    0.3333333333   0.6666666666   0.0000000000  100    # K'
    0.0000000000   0.0000000000   0.0000000000  100    # G
    0.5000000000   0.5000000000   0.0000000000  100    # M
```

(continues on next page)

(continued from previous page)

```
    0.6666666666   0.3333333333   0.0000000000  100    # K
    0.0000000000   0.0000000000   0.0000000000  1      # G
    kpoint_label                    G,M',K',G,M,K,G
}
```

band_range: Integer type, with two values, used to define the range of energy bands for calculation. Counting starts from 1. For example, "1 20" specifies that the calculation includes energy bands from the 1st to the 20th, a total of 21 bands.

For the k point setting of this function, please refer to the kpoint_mode module.

After the task calculation is completed, there will be three files in the Out/Spin_Texture folder, namely kpt.dat and spin_texture_x.dat, spin_texture_y.dat, spin_texture_z.dat, plot_spintexture_line.py, corresponding to the k-point and the spin texture, the drawing script.

5.9 WILSON_LOOP

5.9.1 introduction

We can determine the Z2 topology number of the topological insulator by Wilson loop. Computing the six time reversal invariant planes, we can obtain the Z2 topology metrics (ν_0, ν_1 ν_2 ν_3). These six planes are k1=0.0, k1=0.5, k2=0.0, k2=0.5, k3=0.0, k3=0.5, respectively. When selecting a plane, only half of the plane needs to be selected.

$$\nu_0 = Z2(ki = 0) + Z2(ki = 0.5) \quad mod \quad 2$$
$$\nu_i = Z2(ki = 0.5)$$

where $i = 1, 2, 3$ refers to the x, y and z directions.

The Wilson loop is implemented as follows: \$$W_n(\mathbf{k_2}) = \frac{i}{2\pi}\int_0^{2\pi} d\mathbf{k_1}\langle u_{n,\mathbf{k_1},\mathbf{k_2}}|\partial_{\mathbf{k_1}}|u_{n,\mathbf{k_1},\mathbf{k_2}}\rangle$.\$

5.9.2 example

Here is an example of how to calculate the Wilson loop of the topological insulator Bi_2Se_3 located in the examples/Bi2Se3 folder.

The Input file is:

```
INPUT_PARAMETERS
{
    nspin                           4
    package                         ABACUS
    fermi_energy                    9.557219691497478
    fermi_energy_unit               eV
    HR_route                        data-HR-sparse_SPIN0.csr
    SR_route                        data-SR-sparse_SPIN0.csr
    rR_route                        data-rR-sparse.csr
    HR_unit                         Ry
    rR_unit                         Bohr
    max_kpoint_num                  8000
}

LATTICE
{
```

(continues on next page)

(continued from previous page)

```
    lattice_constant                1.8897162
    lattice_constant_unit           Bohr
    lattice_vector
    -2.069  -3.583614  0.000000
     2.069  -3.583614  0.000000
     0.000   2.389075  9.546667
}

WILSON_LOOP
{
    occ_band            78
    k_start             0.0  0.0  0.5
    k_vect1             1.0  0.0  0.0
    k_vect2             0.0  0.5  0.0
    nk1                 101
    nk2                 101
}
```

`occ_band`: The number of occupied energy bands of an insulator.

`k_start`: The origin point coordinates used to describe a Brillouin zone plane.

`k_vect1`: The expansion vector is a vector used to define a Brillouin zone plane, and it is also the direction of integration for calculations.

`k_vect2`: The expansion vector is a vector used to define a Brillouin zone plane, and it is also the direction of Wilson loop evolution for calculations.

`nk1`: k_vect1 is divided into nk1 k-points.

`nk2`: k_vect2 is divided into nk2 k-points.

After the task calculation is completed, there will be two files in the `Out/Wilson_Loop` folder, namely `wilson_loop.dat` and `plot_wl.py`, corresponding to the Wilson loop of each k-point in the `k_vect2` direction and drawing script.

5.10 POLARIZATION

5.10.1 introduction

We calculate the spontaneous polarization of periodic solids by so-called Modern Theory of Polarization, namely Berry phase theory. The electric polarization **P** is a modulo a quantum $e\mathbf{R}/V_c$ multi-valued function, corresponding to the following implementation equation:

$$\mathbf{P} = \frac{-e}{(2\pi)^3} \sum_{n}^{occ} \int_{\mathrm{BZ}} \mathbf{A}_n(\mathbf{k}) d^3k,$$

where $\mathbf{A}_n(\mathbf{k})$ is the Berry connection of a single band.

5.10.2 example

Here, we present an example (located in the `examples/PbTiO3` folder) showcasing the calculation of the polarization of $PbTiO_3$.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                1
    package              ABACUS
    fermi_energy         13.38267075814371
    fermi_energy_unit    eV
    HR_route             data-HR-sparse_SPIN0.csr
    SR_route             data-SR-sparse_SPIN0.csr
    rR_route             data-rR-sparse.csr
    HR_unit              Ry
    rR_unit              Bohr
}

LATTICE
{
    lattice_constant        7.3699
    lattice_constant_unit   Bohr
    lattice_vector
    1.0000000000         0.0000000000         0.0000000000
    0.0000000000         1.0000000000         0.0000000000
    0.0000000000         0.0000000000         1.0000000000
}

POLARIZATION
{
    occ_band      22
    nk1           10
    nk2           10
    nk3           10
    atom_type     3
    stru_file     STRU
    valence_e     14 12 6
}
```

`occ_band`: The number of occupied energy bands of an insulator.

`nk1`: This refers to the number of samples taken in the x direction of the reciprocal lattice vector **G**.

`nk2`: This refers to the number of samples taken in the y direction of the reciprocal lattice vector **G**.

`nk3`: This refers to the number of samples taken in the z direction of the reciprocal lattice vector **G**.

`stru_file`: Specify the strucutre file. NAOs files are not required.

`atom_type`: The number of element types in the system.

`valence_e`: The number of valence electrons per element.

The parameters `nk1`, `nk2`, and `nk3` correspond to the number of k-points used for the integration in the three lattice directions, and increasing their values leads to a more accurate and convergent result.

After completing the task, `polarization.dat` appears in the `Out/Polarization` folder which contains the electric polarization of the three lattice directions.

5.11 BERRY_CURVATURE

5.11.1 introduction

Berry curvature is of fundamental importance for understanding some basic properties of solid materials and is essential for the description of the dynamics of Bloch electrons.

The Berry curvature of a single energy band is defined as follows:

$$\Omega_n(\mathbf{k}) = \nabla \times \mathbf{A}_n(\mathbf{k}),$$

where Berry connection $\mathbf{A}_n(\mathbf{k}) = i\langle u_{n\mathbf{k}}|\nabla_{\mathbf{k}}|u_{n\mathbf{k}}\rangle$, $|u_{nk}\rangle$ is the periodic part of the Bloch wave function.

We calculated the Berry curvature:

$$\Omega_{\alpha\beta}(\mathbf{k}) = \sum_n f_n(\mathbf{k})\Omega_{n,\alpha\beta}(\mathbf{k}),$$

where f_n is the Fermi occupation function.

Detailed descriptions can be found in Calculation of Berry curvature using non-orthogonal atomic orbitals.

5.11.2 example

An example (refer to folder `example/Fe`) of calculating the Berry curvature of the fcc-Fe is given here.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                   4
    package                 ABACUS
    fermi_energy            18.18839115931923
    fermi_energy_unit       eV
    HR_route                data-HR-sparse_SPIN0.csr
    SR_route                data-SR-sparse_SPIN0.csr
    rR_route                data-rR-sparse.csr
    HR_unit                 Ry
    rR_unit                 Bohr
}

LATTICE
{
    lattice_constant        5.4235
    lattice_constant_unit   Bohr
    lattice_vector
     0.5   0.5   0.5
    -0.5   0.5   0.5
    -0.5  -0.5   0.5
}

BERRY_CURVATURE
{
    method                  0
    kpoint_mode             line
    kpoint_num              10
    high_symmetry_kpoint
```

(continues on next page)

(continued from previous page)

```
    0.0    0.0     0.0    100  # G
    0.5   -0.5    -0.5    100  # H
    0.75   0.25   -0.25   100  # P
    0.5    0.0    -0.5    100  # N
    0.0    0.0     0.0    100  # G
    0.5    0.5     0.5    100  # H
    0.5    0.0     0.0    100  # N
    0.0    0.0     0.0    100  # G
    0.75   0.25   -0.25   100  # P
    0.5    0.0     0.0    1    # N
}
```

`method`: Method for calculating Berry curvature. `0` means direct calculation, `1` means calculation by Kubo formula.

`occ_band`: The number of occupied energy bands of an insulator. When this value is not set, it will be determined according to the Fermi energy.

For the k point setting of this function, please refer to the `kpoint_mode` module.

Upon completion of the task calculation, two files will be generated in the `Out/Berry_Curvature` directory: `kpt.dat` and `berry_curvature.dat`. These files respectively contain the k-point coordinates and the total Berry curvature for each k-point.

5.12 AHC

5.12.1 introduction

The dc anomalous Hall conductivity (AHC) is simply given as the Brillouin zone integral of the Berry curvature of occupying energy bands,

$$\sigma_{xy} = -\frac{e^2}{\hbar}\sum_n^{occ}\int_{\mathrm{BZ}}\frac{d\mathbf{k}}{(2\pi)^3}f_n(\mathbf{k})\Omega_{n,z}(\mathbf{k}).$$

5.12.2 example

An example (refer to folder `example/Fe`) of calculating the AHC of the fcc-Fe is given here.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                    4
    package                  ABACUS
    fermi_energy             18.18839115931923
    fermi_energy_unit        eV
    HR_route                 data-HR-sparse_SPIN0.csr
    SR_route                 data-SR-sparse_SPIN0.csr
    rR_route                 data-rR-sparse.csr
    HR_unit                  Ry
    rR_unit                  Bohr
}

LATTICE
```

(continues on next page)

(continued from previous page)

```
{
    lattice_constant         5.4235
    lattice_constant_unit    Bohr
    lattice_vector
     0.5   0.5   0.5
    -0.5   0.5   0.5
    -0.5  -0.5   0.5
}

AHC
{
    integrate_mode           Grid
    integrate_grid           100 100 100
    adaptive_grid            20 20 20
    adaptive_grid_threshold 100
}
```

`integrate_mode`: Specifies the mode of integration, which can be grid integration and adaptive integration.

`integrate_grid`: Specifies a uniform grid for grid integration.

`adaptive_grid`: Specifies the grid for adaptive densification.

`adaptive_grid_threshold`: Specifies the cut-off value of adaptive densification, the unit is ^{2}.

Once the calculation is complete, the resulting file `ahc.dat` will be generated in the `Out/AHC` directory.

5.13 CHERN_NUMBER

5.13.1 introduction

Chern number is a topological invariant used to explain the quantized Hall conductivity. The Chern invariant is the total Berry flux in the 2D Brillouin zone,

$$n = \frac{1}{2\pi} \oint_{\mathbf{S}} \mathbf{\Omega} \cdot d\mathbf{S}$$

where n is an integer, $\mathbf{S}$ is any closed 2D manifold, $\mathbf{\Omega}$ is total Berry curvature (flux).

To calculate the Chern number, you must first select a closed 2D surface in the Brillouin.

5.13.2 example

An example (refer to folder `examples/MnBi2Te4-weyl`) of calculating Chern number of the Weyl semimetal $MnBi_2Te_4$ is given here.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                          4
    package                        ABACUS
    fermi_energy                   9.2309138700265265
    fermi_energy_unit              eV
    HR_route                       data-HR-sparse_SPIN0.csr
```

(continues on next page)

(continued from previous page)

```
    SR_route                        data-SR-sparse_SPIN0.csr
    rR_route                        data-rR-sparse.csr
    HR_unit                         Ry
    rR_unit                         Bohr
}

LATTICE
{
    lattice_constant                1.8897162
    lattice_constant_unit           Bohr
    lattice_vector
    4.3773399999000002 0.0000000000000000  0.0000000000000000
    2.1886700000000001 3.7908876409999999  0.0000000000000000
    2.1886700000000001 1.2636292099999999 13.7730333333000008
}

CHERN_NUMBER
{
    method                          0
    occ_band                        109
    integrate_mode                  Grid
    integrate_grid                  100 100 1
    adaptive_grid                   20  20  1
    adaptive_grid_threshold         100
    k_start                         0 0 0
    k_vect1                         1 0 0
    k_vect2                         0 1 0
}
```

`method`: Method for calculating berry curvature. `0` means direct calculation, `1` means calculation by Kubo formula.

`occ_band`: The number of occupied energy bands of an insulator. When this value is not set, it will be determined according to the Fermi energy.

To determine a plane of k-space requires an origin (`k_start`) and two vectors that are not parallel to each other (`k_vect1`, `k_vect2`).

`k_start`: The origin point coordinates used to describe a Brillouin zone plane.

`k_vect1`: The expansion vector used to describe a Brillouin zone plane.

`k_vect2`: The expansion vector used to describe a Brillouin zone plane.

For k-point integration, please refer to the [Setting of integration] section.

After the task calculation is completed, the `chern_number.dat` file will appear in the `Out/Chern_Num` folder which contains the Chern number specific results.

5.14 CHIRALITY

5.14.1 introduction

Examines the chirality of Weyl points by calculating the Berry curvature on a sphere around the **k** point.

5.14.2 example

An example is provided in the `examples/MnSb2Te4` directory to investigate the chirality of the Weyl point in $MnSb_2Te_4$.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                           4
    package                         ABACUS
    fermi_energy                    9.9700823666762375
    fermi_energy_unit               eV
    HR_route                        data-HR-sparse_SPIN0.csr
    SR_route                        data-SR-sparse_SPIN0.csr
    rR_route                        data-rR-sparse.csr
    HR_unit                         Ry
    rR_unit                         Bohr
}

LATTICE
{
    lattice_constant                1.8897162
    lattice_constant_unit           Bohr
    lattice_vector
     4.2885731815169859   -0.0001203117360283   -0.0000592245216563
     2.1441823977387200    3.7140734770500492   -0.0000592245216895
     2.1439936829776851    1.2378353300160352   13.4147375436875418
}

CHIRALITY
{
    k_vect                          0.0000 0.0000 -0.0538
    radius                          0.02
    point_num                       100
}
```

`k_vect`: The k-point direct coordinates need to be calculated.

`radius`: The radius of the integrating sphere. The unit is $^{-1}$.

`point_num`: The number of k-points that are uniformly sampled on a spherical surface.

The `chirality.dat` file that record the results are in the `Out/Chirality` folder.

5.15 JDOS

5.15.1 introduction

Joint density of states (JDOS) is used to describe the density of states of electrons excited from the valence band to the conduction band, which relates to the absorption spectrum and the dielectric function of system.

The implementation of JDOS per crystal cell is given by

$$D_{joint}(\omega) = \frac{V_c}{\hbar} \int \frac{d^3 k}{(2\pi)^3} \sum_{n,m} f_{nm} \delta(\omega_{mn} - \omega).$$

where V_c is the cell volume, $f_{nm} = f_n - f_m$ and $\hbar\omega_{mn} = E_m - E_n$ are differences between occupation factors and band energies, respectively.

Currently JDOS is only used to calculate insulators and semiconductors.

5.15.2 example

An example of calculating the JDOS of a perovskite CsPbI$_3$ is presented here (refer to folder `examples/CsPbI3`).

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                   1
    package                 ABACUS
    fermi_energy            5.508975304340945
    fermi_energy_unit       eV
    HR_route                data-HR-sparse_SPIN0.csr
    SR_route                data-SR-sparse_SPIN0.csr
    rR_route                data-rR-sparse.csr
    HR_unit                 Ry
    rR_unit                 Bohr
}

LATTICE
{
    lattice_constant            1.8897261258369282
    lattice_constant_unit       Bohr
    lattice_vector
        6.2894000000       0.0000000000       0.0000000000
        0.0000000000       6.2894000000       0.0000000000
        0.0000000000       0.0000000000       6.2894000000
}

JDOS
{
    occ_band         37
    omega            0.5  10
    domega           0.01
    eta              0.2
    grid             30  30  30
}
```

`occ_band`: Specifies the occupied energy band of the system. Currently, only insulator or semiconductor materials can be calculated.

`omega`: Specifies the photon energy, the unit is eV.

`domega`: Specifies the energy interval of the omega.

`eta`: Specify the parameters of Gaussian smearing.

`grid`: Specifies the uniform k-point grid used to calculate the JDOS.

Once the task calculation is finished, two files will be generated in the `Out/JDOS` folder: `JDOS.dat` and `plot_jdos.py`. The former contains the JDOS data, while the latter is a script used for plotting the JDOS.

5.16 OPTICAL_CONDUCTIVITY

5.16.1 introduction

The frequency-dependent optical conductivity expressed by the Kubo-Greenwood formula can be formulated as

$$\sigma_{\alpha\beta}(\hbar\omega) = -\frac{ie^2\hbar}{NV_{\text{cell}}} \sum_{\mathbf{k}} \sum_{n,m} \left(\frac{f_{n\mathbf{k}} - f_{m\mathbf{k}}}{E_{n\mathbf{k}} - E_{m\mathbf{k}}} \right) \frac{\langle \psi_{n\mathbf{k}} | v_\alpha | \psi_{m\mathbf{k}} \rangle \langle \psi_{m\mathbf{k}} | v_\beta | \psi_{n\mathbf{k}} \rangle}{\hbar\omega + E_{n\mathbf{k}} - E_{m\mathbf{k}} + i\eta} .$$

The imaginary part of the dielectric function is

$$\epsilon_i^{\alpha\beta}(\omega) = -\frac{e^2\pi}{\epsilon_0\hbar} \int \frac{d\mathbf{k}}{(2\pi)^3} \sum_{nm} f_{nm} r_{nm}^{\alpha} r_{mn}^{\beta} \delta\left(\omega_{mn} - \omega\right) ,$$

The real part of the dielectric function is obtained by the Kramer-Kronig transformation,

$$\epsilon_r^{\alpha\beta}(\omega) = \delta_{\alpha\beta} + \frac{2}{\pi} \mathbf{P} \int_0^\infty d\omega' \frac{\omega' \epsilon_i^{\alpha\beta}(\omega')}{\omega'^2 - \omega^2} .$$

The linear optical spectrum can be calculated through the dielectric function, such as refractive index $n(\omega)$, extinction coefficient $\kappa(\omega)$, absorption coefficient $\alpha(\omega)$, energy-loss function $L(\omega)$, reflectivity $R(\omega)$:

$$\begin{aligned}
n(\omega) &= \left[\frac{\sqrt{\varepsilon_1^2 + \varepsilon_2^2} + \varepsilon_1}{2} \right]^{\frac{1}{2}} \\
\kappa(\omega) &= \left[\frac{\sqrt{\varepsilon_1^2 + \varepsilon_2^2} - \varepsilon_1}{2} \right]^{\frac{1}{2}} \\
\alpha(\omega) &= \frac{\sqrt{2}\omega}{c} \left[\sqrt{\varepsilon_1^2 + \varepsilon_2^2} - \varepsilon_1 \right]^{\frac{1}{2}} \\
L(\omega) &= \operatorname{Im}\left(\frac{-1}{\varepsilon(\omega)} \right) = \frac{\varepsilon_2}{\varepsilon_1^2 + \varepsilon_2^2} \\
R(\omega) &= \frac{(n-1)^2 + k^2}{(n+1)^2 + k^2}
\end{aligned}$$

5.16.2 example

Here, we provide an example (located in the `examples/Si` folder) demonstrating the calculation of the optical conductivity and dielectric function for diamond Si.

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                1
    package              ABACUS
    fermi_energy         6.389728305291531
    fermi_energy_unit    eV
    HR_route             data-HR-sparse_SPIN0.csr
    SR_route             data-SR-sparse_SPIN0.csr
    rR_route             data-rR-sparse.csr
    HR_unit              Ry
    rR_unit              Bohr
}

LATTICE
{
    lattice_constant        1.8897162
    lattice_constant_unit   Bohr
    lattice_vector
    0.000000000000  2.715000000000  2.715000000000
    2.715000000000  0.000000000000  2.715000000000
    2.715000000000  2.715000000000  0.000000000000
}

OPTICAL_CONDUCTIVITY
{
    occ_band       4
    omega          0   10
    domega         0.01
    eta            0.1
    grid           50 50 50
}
```

`occ_band`: Used to specify the occupied energy band of an insulator or semiconductor. Currently this function can only calculate insulators or semiconductors.

`omega`: Specifies the photon energy, the unit is eV.

`domega`: Specifies the energy interval of the omega.

`eta`: Specify the parameters of Gaussian smearing.

`grid`: Specifies the uniform k-point grid used to calculate the optical conductivity.

After completing the task, five main files are generated in the `Out/Optical_Conductivity` folder, namely `optical_conductivity_real_part.dat`, `optical_conductivity_imag_part.dat`, `dielectric_function_real_part.dat`, `dielectric_function_imag_part.dat` and `plot_optical.py`.

5.17 SHIFT_CURRENT

5.17.1 introduction

The shift current is an intrinsic contribution to the bulk photovoltaic effect (BPVE). It describes the photocurrent generated by light illumination on homogeneous non-centrosymmetric crystals. The shift current is a second-order optical response. It can be expressed as a DC current, generated by a monochromatic photoelectric field $\mathbf{E}(t) = \mathbf{E}(\omega)\mathrm{e}^{i\omega t} + \mathbf{E}(-\omega)\mathrm{e}^{-i\omega t}$, where

$$J^a = 2\sigma^{abc}(0; \omega, -\omega)E_b(\omega)E_c(-\omega).$$

Here, $a, b, c = x, y, z$, and $\sigma^{abc}(0; \omega, -\omega)$ is the shift current tensor,

$$\sigma^{abc}(0; \omega, -\omega) = \frac{\pi e^3}{\hbar^2} \int \frac{d\mathbf{k}}{8\pi^3} \sum_{n,m} f_{nm} \mathrm{Im}\left[I_{mn}^{abc} + I_{mn}^{acb}\right] \delta\left(\omega_{mn} - \omega\right)$$

where $I_{mn}^{abc} = r_{mn}^b r_{nm;a}^c$, r_{nm}^a is the inter-band dipole matrix, and $r_{nm;a}^b$ is the generalized derivative of the dipole matrix.

Detailed descriptions can be found in Second-order optical response in semiconductors.

5.17.2 example

Here, we provide an example of calculating the shift current conductivity of the monolayer WS_2 (refer to folder `examples/WS2`).

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                       1
    package                     ABACUS
    fermi_energy                0.3484302262859574
    fermi_energy_unit           eV
    HR_route                    data-HR-sparse_SPIN0.csr
    SR_route                    data-SR-sparse_SPIN0.csr
    rR_route                    data-rR-sparse.csr
    HR_unit                     Ry
    rR_unit                     Bohr
}

LATTICE
{
    lattice_constant            1.8897162
    lattice_constant_unit       Bohr
    lattice_vector
    3.183820900165    0.0                 0.0
   -1.591910450082    2.757269780643    0.0
    0.0               0.0                 20.086904001384
}

SHIFT_CURRENT
{
    occ_band                    13
    omega                       0    4
```

(continues on next page)

(continued from previous page)

```
    domega                  0.01
    smearing_method         1
    eta                     0.1
    grid                    1000 1000 1
}
```

`occ_band`: Used to specify the occupied energy band of an insulator or semiconductor. Currently this function can only calculate insulators or semiconductors.

`omega`: Specifies the photon energy, the unit is eV.

`domega`: Specifies the energy interval of the omega.

`eta`: Specify the parameters of Gaussian smearing.

`grid`: Specifies the uniform k-point grid used to calculate the shift current.

Upon completion of the task, two essential files are generated in the `Out/Shift_Current` folder. These files include `shift_current.dat` and `plot_shift_current.py`.

5.18 BERRY_CURVATURE_DIPOLE

5.18.1 introduction

In a system with time-reversal symmetry, the Berry curvature is an odd function of **k**, i.e., $\Omega_a(\mathbf{k}) = -\Omega_a(-\mathbf{k})$. As a result, the integration of the Berry curvature over the BZ is zero. However, if the system lacks a inversion symmetry, a higher-order nonlinear AHC can arise. More specifically, $j_a^0 = \chi_{abc}E_b(\omega)E_c(-\omega)$ and $j_a^{2\omega} = \chi_{abc}E_b(\omega)E_c(\omega)$, describe a rectified current and the second harmonic, respectively, whereas ω is the driving frequency. The coefficient χ_{abc} is given by

$$\chi_{abc} = -\varepsilon_{adc}\frac{e^3\tau}{2(1+i\omega\tau)}D_{bd}.$$

where

$$D_{ab} = \int_k f_0\left(\frac{\partial\Omega_b}{\partial k_a}\right)$$

is called the Berry curvature dipole.

In PYATB, the Berry curvature dipole at a given temperature T and chemical potential μ is calculated using the following formula:

$$D_{ab}(\mu, T) = -\int \frac{\partial f_0(E,\mu,T)}{\partial E}D_{ab}(E)dE.$$

5.18.2 example

Here, we provide an example of calculating the Berry curvature dipole of the trigonal Te (refer to folder `examples/Te`).

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                       4
    package                     ABACUS
    fermi_energy                9.574476774876963
    fermi_energy_unit           eV
    HR_route                    data-HR-sparse_SPIN0.csr
    SR_route                    data-SR-sparse_SPIN0.csr
    rR_route                    data-rR-sparse.csr
    HR_unit                     Ry
    rR_unit                     Bohr
    max_kpoint_num              28000
}

LATTICE
{
    lattice_constant            1.8897162
    lattice_constant_unit       Bohr
    lattice_vector
    2.22    -3.84515    0.000
    2.22     3.84515    0.000
    0.00     0.00000    5.910

}

BERRY_CURVATURE_DIPOLE
{
    omega                       9.474  10.074
    domega                      0.001
    integrate_mode              Grid
    integrate_grid              400 400 400
    adaptive_grid               20 20 20
    adaptive_grid_threshold     20000
}
```

`omega`: To set the energy range for the Berry curvature dipole, you can adjust it based on the Fermi energy level. the unit is eV.

`domega`: Specifies the energy interval of the omega.

For k-point integration, please refer to the [Setting of integration] section.

Once the task has been finished, three crucial files are produced in the `Out/Berry_Curvature_Dipole` directory. These files consist of `bcd_step2.dat`, `kpoint_list`, and `plot_bcd.py`. The first file stores the Berry curvature dipole's magnitude, while the second file records the refined k-point coordinates. The third file contains the script used for generating the visualization of the dipole.

5.19 SECOND_HARMONIC_GENERATION

5.19.1 introduction

Second harmonic generation (SHG), also called frequency doubling, is a nonlinear optical process, in which photons interacting with a nonlinear material are effectively ‘combined’ to form new photons having twice the frequency of initial photons. The SHG coefficient is defined:

$$P^c(2\omega) = \chi^{abc}(-2\omega;\omega,\omega)E^a(\omega)E^b(\omega)$$

In PYATB, the SHG calculated using two methods. One is the commonly used form which divides SHG into inter- and intra-band parts, shown in the following formula:

$$\chi^{abc}_{interband}(-2\omega,\omega,\omega) = \frac{e^3}{\hbar^2\Omega}\sum_{nml,\mathbf{k}} \frac{r^a_{nm}\left\{r^b_{ml}r^c_{ln}\right\}}{(\omega_{ln}-\omega_{ml})}\left[\frac{2f_{nm}}{\omega_{mn}-2\omega}+\frac{f_{ln}}{\omega_{ln}-\omega}+\frac{f_{ml}}{\omega_{ml}-\omega}\right]$$

$$\begin{aligned}\chi^{abc}_{intraband}(-2\omega,\omega,\omega) =& \frac{i}{2}\frac{e^3}{\hbar^2\Omega}\sum_{nm,\mathbf{k}} f_{nm}\left[\frac{2}{\omega_{mn}(\omega_{mn}-2\omega)}r^a_{nm}\left(r^b_{nm;c}+r^c_{mn;b}\right)+\frac{1}{\omega_{mn}(\omega_{mn}-\omega)}\left(r^a_{nm;c}r^b_{mn}+r^a_{nm;b}r^c_{mn}\right)\right.\\ &+\frac{1}{\omega^2_{mn}}\left(\frac{1}{\omega_{mn}-\omega}-\frac{4}{\omega_{mn}-2\omega}\right)r^a_{nm}\left(r^b_{mn}\Delta^c_{mn}+r^c_{mn}\Delta^b_{mn}\right)\\ &\left.-\frac{1}{2\omega_{mn}(\omega_{mn}-\omega)}\left(r^b_{nm;a}r^c_{mn}+r^c_{nm;a}r^b_{mn}\right)\right]\end{aligned}$$

This is the most widely used form and is the default method (method 0) of SHG calculation. Secondly SHG could be expressed in the form of multiplying velocity matrices:

$$\chi^{abc}(-2\omega;\omega,\omega) = -\sum_{n,m,l,\mathbf{k}}\frac{i}{2\omega^3(2\omega-\omega_{mn})}\left(\frac{f_{nl}}{\omega-\omega_{ln}}+\frac{f_{ml}}{\omega-\omega_{ml}}\right)v^c_{nm}v^a_{ml}v^b_{ln}$$

This form is generally aligns with the upper formula, while the calculation can be costly for a wide energy range can be contributing to a single energy point.

5.19.2 example

Here, we provide an example of calculating the SHG of the GaAs (refer to folder `tutorial/GaAs_SHG/`).

The `Input` file is:

```
INPUT_PARAMETERS
{
    nspin                   1
    package                 ABACUS
    fermi_energy            10.171348972
    fermi_energy_unit       eV
    HR_route                ../abacus/OUT.GaAs/data-HR-sparse_SPIN0.csr
    SR_route                ../abacus/OUT.GaAs/data-SR-sparse_SPIN0.csr
    rR_route                ../abacus/OUT.GaAs/data-rR-sparse.csr
    HR_unit                 Ry
    rR_unit                 Bohr
   max_kpoint_num           100000
}

LATTICE
```

(continues on next page)

(continued from previous page)

```
{
    lattice_constant        1.889727
    lattice_constant_unit   Bohr
    lattice_vector
    0.0000000000000000    2.7650000000000001    2.7650000000000001
    2.7650000000000001    0.0000000000000000    2.7650000000000001
    2.7650000000000001    2.7650000000000001    0.0000000000000000

}

SHG
{
    omega            0.01 4
    domega           0.01
    eta              0.05
    grid             50 50 50
}
```

`omega`: To set the energy range for the SHG, you can adjust it. the unit is eV.

`domega`: Specifies the energy interval of the omega.

`grid`: Specifies the uniform k-point grid used to calculate the SHG.

`eta`: $\hbar\omega \rightarrow \hbar\omega + i\eta$ is used to prevent numerical divergence caused by a zero denominator.

Once the task has been finished, three crucial files are produced in the `Out/Second_Harmonic_Generation` directory. These files consist of `shg_real.dat`, `shg_imag.dat` and `plot_shg.py`.

The first two files contain the real and imaginary parts of the SHG's magnitude, and the last one is a plotting script.

CHAPTER

SIX

DETAILED INTRODUCTION OF THE INPUT FILES

6.1 Full List of INPUT Keywords

- *INPUT_PARAMETERS*

  nspin | *package* | *fermi_energy* | *fermi_energy_unit* | *HR_route* | *SR_route* | *rR_route* | *binary* | *HR_unit* | *rR_unit* | *max_kpoint_num* | *sparse_format* | *w90_TB_route* | *w90_TB_has_r*
- *LATTICE*

  lattice_constant | *lattice_constant_unit* | *lattice_vector*
- *BAND_STRUCTURE*

  wf_collect | *band_range* | *kpoint_mode*
- *BANDUNFOLDING*

  stru_file | *ecut* | *band_range* | *m_matrix* | *kpoint_mode*
- *FERMI_ENERGY*

  temperature | *electron_num* | *grid* | *epsilon* |
- *FERMI_SURFACE*

  bar | *nbands* | *kpoint_mode*
- *FIND_NODES*

  energy_range | *k_start* | *k_vect1* | *k_vect2* | *k_vect3* | *initial_grid* | *initial_threshold* | *adaptive_gridadaptive_threshold* | *kpoint_mode*
- *PDOS*

  stru_file | *e_range* | *de* | *sigma* | *kpoint_mode*
- *FAT_BAND*

  band_range | *stru_file* | *kpoint_mode*
- *SPIN_TEXTURE*

  band_range | *kpoint_mode*
- *SURFACE_STATE*

  cal_surface_method | *surface_direction* | *energy_windows* | *de* | *eta* | *coupling_layers* | *calculate_layer* | *kpoint_mode*
- *WILSON_LOOP*

  occ_band | *k_start* | *k_vect1* | *k_vect2* | *nk1* | *nk2*

- *POLARIZATION*

  occ_band | *nk1* | *nk2* | *nk3* | *atom_type* | *stru_file* | *valence_e*

- *BERRY_CURVATURE*

  method | *occ_band* | *kpoint_mode*

- *AHC*

  method | *integrate_mode*

- *ANC*

  fermi_range | *de* | *eta* | *integrate_grid*

- *SHC*

  alpha | *beta* | *gamma* | *fermi_range* | *de* | *eta* | *integrate_grid*

- *CHERN_NUMBER*

  method | *occ_band* | *k_start* | *k_vect1* | *k_vect2* | *integrate_mode*

- *CHIRALITY*

  method | *occ_band* | *k_vect* | *radius* | *point_num*

- *JDOS*

  occ_band | *omega* | *domega* | *eta* | *grid*

- *OPTICAL_CONDUCTIVITY*

  occ_band | *omega* | *domega* | *eta* | *grid*

- *SHIFT_CURRENT*

  occ_band | *omega* | *domega* | *smearing_method* | *eta* | *grid* | *method*

- *BERRY_CURVATURE_DIPOLE*

  omega | *domega* | *grid*

- *SHG*

  method | *omega* | *domega* | *eta* | *grid*

- *POCKELS*

  omega1 | *omega* | *domega* | *grid*

Setting of k points

- *When kpoint_mode is 'mp'*

  mp_grid | *k_start* | *k_vect1* | *k_vect2* | *k_vect3*

- *When kpoint_mode is 'line'*

  kpoint_num | *high_symmetry_kpoint*

- *When kpoint_mode is 'direct'*

  kpoint_num | *kpoint_direct_coor*

Setting of integration

- *When integrate_mode is 'Grid'*

  integrate_grid | *adaptive_grid* | *adaptive_grid_threshold*

- *When integrate_mode is 'Adaptive'*

  relative_error | *absolute_error* | *initial_grid*

6.1.1 INPUT_PARAMETERS

nspin {#input_nspin}

- **Type**: Integer
- **Description**: Indicates the spin component of the wave function, related to the structure of the HR file.
  - 1: regardless of spin.
  - 2: the wave function is divided into two groups, one group is all up and one group is all down.
  - 4: the wave function has both up and down components.
- **Default**: No default value

package {#input_package}

- **Type**: String
- **Description**: Indicates data sources for HR, SR, rR.
- **Default**: ABACUS

fermi_energy {#input_fermi_energy}

- **Type**: Real
- **Description**: Indicates the Fermi energy of the system. When set to Auto, the FERMI_ENERGY function needs to be added.
- **Default**: Auto

fermi_energy_unit {#input_fermi_energy_unit}

- **Type**: String
- **Description**: The unit of Fermi. Can be set to Ry, eV.
- **Default**: eV

HR_route {#input_HR_route}

- **Type**: String
- **Description**: Path to HR matrix file. When nspin=2, two sets of paths need to be provided.
- **Default**: No default value

SR_route {#input_SR_route}

- **Type**: String
- **Description**: Path to the SR matrix file.
- **Default**: No default value

rR_route {#input_rR_route}

- **Type**: String
- **Description**: Path to the rR matrix file.
- **Default**: No default value

binary {#input_binary}

- **Type**: Boolean
- **Description**: Whether HR, SR, and rR files are binary files.
- **Default**: 0

HR_unit {#input_HR_unit}

- **Type**: String
- **Description**: The unit of HR. Can be set to Ry, eV.
- **Default**: Ry

rR_unit {#input_rR_unit}

- **Type**: String
- **Description**: The unit of rR. Can be set to Bohr, Angstrom.
- **Default**: Bohr

max_kpoint_num {#input_max_kpoint_num}

- **Type**: Integer
- **Description**: The upper limit of the number of k points stored in the memory during program calculation, which is used to control the memory consumption during calculation.
- **Default**: 8000

sparse_format {#input_sparse_format}

- **Type**: Boolean
- **Description**: Whether HR, SR, rR matrices are stored in memory is sparse storage.
- **Default**: 0

w90_TB_route {input_w90_tb_route}

- **Type**: String
- **Description**: Specify the tight-binding file generated by Wannier90.
- **Default**: No default value

w90_TB_has_r {input_w90_tb_has_r}

- **Type**: Boolean
- **Description**: Whether the tight-binding file generated by Wannier90 contain rR information.
- **Default**: 0

6.1.2 LATTICE

lattice_constant {#lattice_lattice_constant}

- **Type**: Real
- **Description**: The lattice constant of the system.
- **Default**: No default value

lattice_constant_unit {#lattice_lattice_constant_unit}

- **Type**: String
- **Description**: The unit of the lattice constant. Can be set to Bohr, Angstrom.
- **Default**: Bohr

lattice_vector {#lattice_lattice_vector}

- **Type**: Real
- **Description**: The 3 lattice vectors of the system. Each lattice vector is a row, with a total of 3 rows and 9 parameters.
- **Default**: No default value

6.1.3 BAND_STRUCTURE

wf_collect {#bandstructure_wf_collect}

- **Type**: Boolean
- **Description**: Whether to output wave function matrix information.
- **Default**: No default value

band_range {#bandstructure_band_range}

- **Type**: Integer
- **Description**: There are two numbers (separated by spaces) to indicate which bands are selected, counting from 1.
- **Default**: No default value

kpoint_mode {#bandstructure_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.4 BANDUNFOLDING

stru_file {#bandunfolding_stru_file}

- **Type**: String
- **Description**: Specify the strucutre file path.
- **Default**: No default value

ecut {#bandunfolding_ecut}

- **Type**: Real
- **Description**: Used to determine the number of plane wave basis sets. Unit is Ry.
- **Default**: 10

band_range {#bandunfolding_band_range}

- **Type**: Integer
- **Description**: Specifies the range of supercell energy band index within which the energy bands will be calculated. There are two parameters, representing the starting band index and the end band index, the index counts from 1.
- **Default**: No default value

m_matrix {#bandunfolding_m_matrix}

- **Type**: Real
- **Description**: The lattice vector transformation matrix between the supercell and the primitive cell, with 9 parameters, is written on the same line.
- **Default**: No default value

kpoint_mode {#bandunfolding_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point of unitcell. See *Setting of k points*
- **Default**: No default value

6.1.5 FERMI_ENERGY

temperature {#fermienergy_temperature}

- **Type**: Real
- **Description**: temperature. The unit is K
- **Default**: 0

electron_num {#fermienergy_electron_num}

- **Type**: Integer
- **Description**: The total number of electrons in the system.
- **Default**: No default value

grid {#fermienergy_grid}

- **Type**: Integer
- **Description**: The grid to use for Newton interpolation. There are three parameters.
- **Default**: 10 10 10

epsilon {#fermienergy_epsilon}

- **Type**: Real
- **Description**: Newton interpolation parameters, absolute accuracy.
- **Default**: 0.001

6.1.6 FERMI_SURFACE

bar {#fermisurface_bar}

- **Type**: Real
- **Description**: The max tolerable error bar for the Fermi surface
- **Default**: No default value

nbands {#fermisurface_nbands}

- **Type**: Integer
- **Description**: If you know the energy band range where the Fermi energy is located, you can set this parameter to speed up the calculation. There are two numbers in total, indicating the range of the energy band. The default value is `0 0`, that is, all energy bands are considered.
- **Default**: 0 0

kpoint_mode {#fermisurface_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.7 FIND_NODES

energy_range {#findnodes_energy_range}

- **Type**: Integer
- **Description**: The energy range in which the program searches for degenerate points, the energy unit is eV.
- **Default**: No default value

k_start {#findnodes_k_start}

- **Type**: Real
- **Description**: The origin point coordinates used to describe a Brillouin zone plane.
- **Default**: 0.0 0.0 0.0

k_vect1 {#findnodes_k_vect1}

- **Type**: Real
- **Description**: The expansion vector used to describe a Brillouin zone plane.
- **Default**: 1.0 0.0 0.0

k_vect2 {#findnodes_k_vect2}

- **Type**: Real
- **Description**: The expansion vector used to describe a Brillouin zone plane.
- **Default**: 0.0 1.0 0.0

k_vect3 {#findnodes_k_vect3}

- **Type**: Real
- **Description**: The expansion vector used to describe a Brillouin zone plane.
- **Default**: 0.0 0.0 1.0

initial_grid {#findnodes_initial_grid}

- **Type**: Integer
- **Description**: Set the initial grid for searching degenerate k points. There are three parameters.
- **Default**: 10 10 10

initial_threshold {#findnodes_initial_threshold}

- **Type**: Real
- **Description**: The energy unit is eV. In the initial grid, only the k-points whose band differences are less than the threshold can enter the search for the next round of degenerate points.
- **Default**: 0.1

adaptive_grid {#findnodes_adaptive_grid}

- **Type**: Integer
- **Description**: The refined grid will refine the k-points that reach the initial_threshold in the initial grid. There are three parameters.
- **Default**: 20 20 20

adaptive_threshold {#findnodes_adaptive_threshold}

- **Type**: Real
- **Description**: The minimum difference considered in independent bands, the energy unit is eV. This means if the band gap is below this bar, it will be recognized as degenerate bands.
- **Default**: 0.001

kpoint_mode {#findnodes_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.8 PDOS

stru_file {#pdos_stru_file}

- **Type**: String
- **Description**: The structure file name. This file records the structure of the lattice, the types of elements, and the atomic orbitals used. Make sure that both the structure file and the orbital file exist.
- **Default**: No default value

e_range {#pdos_e_range}

- **Type**: Real
- **Description**: The range of energy E. There are two parameters, indicating the starting point and the ending point.
- **Default**: No default value

de {#pdos_de}

- **Type**: Real
- **Description**: The interval dE for the energy E.
- **Default**: 0.01

sigma {#pdos_sigma}

- **Type**: Real
- **Description**: Parameters for gauss smearing.
- **Default**: 0.001

kpoint_mode {#pdos_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.9 FAT_BAND

band_range {#fatband_band_range}

- **Type**: Integer
- **Description**: There are two numbers (separated by spaces) to indicate which bands are selected for projection, counting from 1.
- **Default**: No default value

stru_file {#fatband_stru_file}

- **Type**: String
- **Description**: The structure file name. This file indicates the crystal structure and the corresponding orbital file. Make sure that both the structure file and the orbital file exist.
- **Default**: No default value

kpoint_mode {#fatband_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point of unitcell. See *Setting of k points*
- **Default**: No default value

6.1.10 SPIN_TEXTURE

band_range {#spintexture_band_range}

- **Type**: Integer
- **Description**: There are two numbers (separated by spaces) to indicate which bands are selected, counting from 1.
- **Default**: No default value

kpoint_mode {#spintexture_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.11 SURFACE_STATE

cal_surface_method {#surface_state_cal_surface_method}

- **Type**: String
- **Description**: Perform calculations using the surface Green's function.
- **Default**: green_fun

surface_direction {#surface_state_surface_direction}

- **Type**: String
- **Description**: Set the orientation of the material surface: a, b, and c represent the a, b, and c axes of the lattice vectors.
- **Default**: c

energy_windows {#surface_state_energy_windows}

- **Type**: Real
- **Description**: Set the energy range of the bands centered on the *fermi_energy*. There are two values.
- **Default**: -1.0 1.0

de {#surface_state_de}

- **Type**: Real
- **Description**: Energy intervals dividing the energy_windows.
- **Default**: 0.01

eta {#surface_state_eta}

- **Type**: Real
- **Description**: Set the broadening of the Green's function, $G = (E + i * \eta - H)^{-1}$.
- **Default**: 0.01

coupling_layers {#surface_state_coupling_layers}

- **Type**: Integer
- **Description**: By setting the number of principal layers, it is possible to ensure that interactions occur only between adjacent principal layers.
- **Default**: No default value

calculate_layer {#surface_state_calculate_layer}

- **Type**: Integer
- **Description**: Specify the number of layers of the spectral function to output.
- **Default**: No default value

kpoint_mode {#surface_state_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.12 WILSON_LOOP

occ_band {#wilsonloop_occ_band}

- **Type**: Integer
- **Description**: The number of occupied energy bands of an insulator.
- **Default**: No default value

k_start {#wilsonloop_k_start}

- **Type**: Real
- **Description**: The origin point coordinates used to describe a Brillouin zone plane.
- **Default**: 0.0 0.0 0.0

k_vect1 {#wilsonloop_k_vect1}

- **Type**: Real
- **Description**: The expansion vector is a vector used to define a Brillouin zone plane, and it is also the direction of integration for calculations.
- **Default**: 1.0 0.0 0.0

k_vect2 {#wilsonloop_k_vect2}

- **Type**: Real
- **Description**: The expansion vector is a vector used to define a Brillouin zone plane, and it is also the direction of Wilson loop evolution for calculations.
- **Default**: 0.0 1.0 0.0

nk1 {#wilsonloop_nk1}

- **Type**: Integer
- **Description**: k_vect1 is divided into nk1 k-points.
- **Default**: 100

nk2 {#wilsonloop_nk2}

- **Type**: Integer
- **Description**: k_vect2 is divided into nk2 k-points.
- **Default**: 100

6.1.13 POLARIZATION

occ_band {#polarization_occ_band}

- **Type**: Integer
- **Description**: The number of occupied energy bands of an insulator.
- **Default**: No default value

nk1 {#polarization_nk1}

- **Type**: Integer
- **Description**: The number of samples in the x direction of reciprocal lattice vector **G**.
- **Default**: No default value

nk2 {#polarization_nk2}

- **Type**: Integer
- **Description**: The number of samples in the y direction of reciprocal lattice vector **G**.
- **Default**: No default value

nk3 {#polarization_nk3}

- **Type**: Integer
- **Description**: The number of samples in the z direction of reciprocal lattice vector **G**.
- **Default**: No default value

atom_type {#polarization_atom_type}

- **Type**: Integer
- **Description**: The number of element types in the system.
- **Default**: No default value

stru_file {#polarization_stru_file}

- **Type**: String
- **Description**: Specify the strucutre file. NAOs files are not required.
- **Default**: No default value

valence_e {#polarization_valence_e}

- **Type**: Integer
- **Description**: The number of valence electrons per element.
- **Default**: No default value

6.1.14 BERRY_CURVATURE

method {#berrycurvature_method}

- **Type**: Integer
- **Description**: Method for calculating berry curvature. `0` means direct calculation, `1` means calculation by Kubo formula.
- **Default**: 0

occ_band {#berrycurvature_occ_band}

- **Type**: Integer
- **Description**: The number of occupied energy bands of an insulator. When this value is not set, it will be determined according to the Fermi energy.
- **Default**: -1

kpoint_mode {#berrycurvature_kpoint_mode}

- **Type**: String
- **Description**: Used to set the k point. See *Setting of k points*
- **Default**: No default value

6.1.15 AHC

method {#ahc_method}

- **Type**: Integer
- **Description**: Method for calculating berry curvature. `0` means direct calculation, `1` means calculation by Kubo formula.
- **Default**: 0

integrate_mode {#ahc_integrate_mode}

- **Type**: String
- **Description**: Used for integration settings. See *Setting of integration*.
- **Default**: No default value

6.1.16 ANC

fermi_range {#anc_fermi_range}

- **Type**: Real
- **Description**: Provide the range of the Fermi energy with respect to the reference point *fermi_energy*. Specify two floating-point numbers.
- **Default**: -1.0 1.0

de {#anc_de}

- **Type**: Real
- **Description**: Energy intervals dividing the Fermi energy range.
- **Default**: 0.01

eta {#anc_eta}

- **Type**: Real
- **Description**: When calculating the Berry curvature using the Kubo formula, the energy difference in the denominator is modified as $E_{n\mathbf{k}} - E_{m\mathbf{k}} \to E_{n\mathbf{k}} - E_{m\mathbf{k}} + i\eta$ to avoid numerical issues caused by degeneracy.
- **Default**: 0.01

integrate_grid {#anc_integrate_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.1.17 SHC

alpha {#shc_alpha}

- **Type**: String
- **Description**: Specify the current direction of the SHC, $\sigma_{\alpha\beta}^{\gamma}$.
- **Default**: x

beta {#shc_beta}

- **Type**: String
- **Description**: Specify the electric field direction of the SHC, $\sigma_{\alpha\beta}^{\gamma}$.
- **Default**: y

gamma {#shc_gamma}

- **Type**: String
- **Description**: Specify the spin polarization direction of the SHC, $\sigma_{\alpha\beta}^{\gamma}$.
- **Default**: z

fermi_range {#shc_fermi_range}

- **Type**: Real
- **Description**: Provide the range of the Fermi energy with respect to the reference point *fermi_energy*. Specify two floating-point numbers.
- **Default**: -1.0 1.0

de {#shc_de}

- **Type**: Real
- **Description**: Energy intervals dividing the Fermi energy range.
- **Default**: 0.01

eta {#shc_eta}

- **Type**: Real
- **Description**: When calculating the Berry curvature using the Kubo formula, the energy difference in the denominator is modified as $E_{n\mathbf{k}} - E_{m\mathbf{k}} \to E_{n\mathbf{k}} - E_{m\mathbf{k}} + i\eta$ to avoid numerical issues caused by degeneracy.
- **Default**: 0.01

integrate_grid {#shc_integrate_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.1.18 CHERN_NUMBER

method {#chernnumber_method}

- **Type**: Integer
- **Description**: Method for calculating berry curvature. `0` means direct calculation, `1` means calculation by Kubo formula.
- **Default**: 0

occ_band {#chernnumber_occ_band}

- **Type**: Integer
- **Description**: The number of occupied energy bands of an insulator. When this value is not set, it will be determined according to the Fermi energy.
- **Default**: -1

k_start {#chernnumber_k_start}

- **Type**: Real
- **Description**: The origin point coordinates used to describe a Brillouin zone plane.
- **Default**: 0.0 0.0 0.0

k_vect1 {#chernnumber_k_vect1}

- **Type**: Real
- **Description**: The expansion vector used to describe a Brillouin zone plane.
- **Default**: 1.0 0.0 0.0

k_vect2 {#chernnumber_k_vect2}

- **Type**: Real
- **Description**: The expansion vector used to describe a Brillouin zone plane.
- **Default**: 0.0 1.0 0.0

integrate_mode {#chernnumber_integrate_mode}

- **Type**: String
- **Description**: Used for integration settings. See *Setting of integration*.
- **Default**: No default value

6.1.19 CHIRALITY

method {#chirality_method}

- **Type**: Integer
- **Description**: Method for calculating berry curvature. `0` means direct calculation, `1` means calculation by Kubo formula.
- **Default**: 0

occ_band {#chirality_occ_band}

- **Type**: Integer
- **Description**: The number of occupied energy bands of an insulator. When this value is not set, it will be determined according to the Fermi energy.
- **Default**: -1

k_vect {#chirality_k_vect}

- **Type**: Real
- **Description**: The k-point coordinates need to be calculated. There are three parameters to represent the coordinates.
- **Default**: No default value

radius {#chirality_radius}

- **Type**: Real
- **Description**: The radius of the integrating sphere. The unit is $^{-1}$.
- **Default**: No default value

point_num {#chirality_point_num}

- **Type**: Integer
- **Description**: The number of k-points that are uniformly sampled on a spherical surface.
- **Default**: No default value

6.1.20 JDOS

occ_band {#jdos_occ_band}

- **Type**: Integer
- **Description**: Specifies the occupied energy band of the system. Currently, only insulator or semiconductor materials can be calculated.
- **Default**: No default value

omega {#jdos_omega}

- **Type**: Real
- **Description**: Specifies the photon energy, the unit is eV. There are two parameters, indicating the starting point and the ending point.
- **Default**: No default value

domega {#jdos_domega}

- **Type**: Real
- **Description**: The energy interval of ω.
- **Default**: No default value

eta {#jdos_eta}

- **Type**: Real
- **Description**: Specify the parameters of Gaussian smearing.
- **Default**: 0.01

grid {#jdos_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.1.21 OPTICAL_CONDUCTIVITY

occ_band {#opticalconductivity_occ_band}

- **Type**: Integer
- **Description**: The number of occupied energy bands of an insulator. When this value is not set, it will be determined according to the Fermi energy.
- **Default**: -1

omega {#opticalconductivity_omega}

- **Type**: Real
- **Description**: The range of ω. There are two parameters, indicating the starting point and the ending point. Unit is eV.
- **Default**: No default value

domega {#opticalconductivity_domega}

- **Type**: Real
- **Description**: The energy interval of ω.
- **Default**: No default value

eta {#opticalconductivity_eta}

- **Type**: Real
- **Description**: Parameters for triangular smearing.
- **Default**: 0.01

grid {#opticalconductivity_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.1.22 SHIFT_CURRENT

occ_band {#shiftcurrent_occ_band}

- **Type**: Integer
- **Description**: Used to specify the occupied energy band of an insulator or semiconductor. Currently this function can only calculate insulators or semiconductors.
- **Default**: No default value

omega {#shiftcurrent_omega}

- **Type**: Real
- **Description**: The range of ω. There are two parameters, indicating the starting point and the ending point. Unit is eV.
- **Default**: No default value

domega {#shiftcurrent_domega}

- **Type**: Real
- **Description**: The energy interval of ω.
- **Default**: No default value

smearing_method {#shiftcurrent_smearing_method}

- **Type**: Integer
- **Description**: The method of smearing. `0`: no smearing. `1`: Gaussian smearing. `2`: adaptive smearing.
- **Default**: 1

eta {#shiftcurrent_eta}

- **Type**: Real
- **Description**: Specify the parameters of Gaussian smearing.
- **Default**: 0.01

grid {#shiftcurrent_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

method {#shiftcurrent_method}

- **Type**: Integer
- **Description**: Specify the method to calculate the shift current. `0` represents calculation using the Sternheimer equation, `1` represents the first order partial derivative calculation.
- **Default**: 1

6.1.23 BERRY_CURVATURE_DIPOLE

omega {#berrycurvaturedipole_omega}

- **Type**: Real
- **Description**: To set the energy range for the Berry curvature dipole, you can adjust it based on the Fermi energy level. The unit is eV. There are two parameters.
- **Default**: No default value

domega {#berrycurvaturedipole_domega}

- **Type**: Real
- **Description**: Specifies the energy interval of the omega.
- **Default**: No default value

grid {#berrycurvaturedipole_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.1.24 SHG

method {#shg_method}

- **Type**: Integer
- **Description**: Specify the method to calculate the SHG. `0` represents calculation using the inter- and intra-band formula, `1` represents the velocity matrices formula.
- **Default**: 0

omega {#shg_omega}

- **Type**: Real
- **Description**: The range of ω. There are two parameters, indicating the starting point and the ending point. Unit is eV.
- **Default**: No default value

domega {#shg_domega}

- **Type**: Real
- **Description**: The energy interval of ω.
- **Default**: No default value

eta {#shg_eta}

- **Type**: Real
- **Description**: $\hbar\omega \rightarrow \hbar\omega + i\eta$ is used to prevent numerical divergence caused by a zero denominator.
- **Default**: 0.05

grid {#shg_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.1.25 POCKELS

omega1 {#pockels_omega1}

- **Type**: Real
- **Description**: The frequency of external electric field, normally a rather small number. Unit is eV.
- **Default**: 0

omega {#pockels_omega}

- **Type**: Real
- **Description**: The range of ω. There are two parameters, indicating the starting point and the ending point. Unit is eV.
- **Default**: No default value

domega {#pockels_domega}

- **Type**: Real
- **Description**: The energy interval of ω.
- **Default**: No default value

grid {#pockels_grid}

- **Type**: Integer
- **Description**: The grid for integration. There are 3 parameters in total.
- **Default**: No default value

6.2 Setting of k points

As long as the kpoint_mode parameter exists in FUNCTIONS, the following setting methods are to be followed.

6.2.1 When kpoint_mode is 'mp'

mp_grid

- **Type**: Integer
- **Description**: The grid dividing the Brillouin zone. There are three parameters to divide the three-dimensional Brillouin zone.
- **Default**: No default value

k_start

- **Type**: Real
- **Description**: The origin point coordinates of the Brillouin zone.
- **Default**: 0.0 0.0 0.0

k_vect1

- **Type**: Real
- **Description**: Expanded vector of the Brillouin zone.
- **Default**: 1.0 0.0 0.0

k_vect2

- **Type**: Real
- **Description**: Expanded vector of the Brillouin zone.
- **Default**: 0.0 1.0 0.0

k_vect3

- **Type**: Real
- **Description**: Expanded vector of the Brillouin zone.
- **Default**: 0.0 0.0 1.0

6.2.2 When kpoint_mode is 'line'

kpoint_num

- **Type**: Integer
- **Description**: The number of high symmetry points.
- **Default**: No default value

kpoint_label

- **Type**: String
- **Description**: Specify the labels of high symmetry points, separated by spaces.
- **Default**: No default value

high_symmetry_kpoint

- **Type**: Real
- **Description**: Fractional coordinates of high symmetry points and line densities of corresponding k-lines. The first three parameters are the fractional coordinates of the high symmetry points, and the fourth parameter is the line density.
- **Default**: No default value

6.2.3 When kpoint_mode is 'direct'

kpoint_num

- **Type**: Integer
- **Description**: the number of k points.
- **Default**: No default value

kpoint_direct_coor

- **Type**: Real
- **Description**: Fractional coordinates of the k point.
- **Default**: No default value

6.3 Setting of integration

As long as the integrate_mode parameter exists in FUNCTIONS, the following setting methods are followed.

6.3.1 When integrate_mode is 'Grid'

integrate_grid

- **Type**: Integer
- **Description**: Low precision grid for integration. There are three parameters.
- **Default**: 4 4 4

adaptive_grid

- **Type**: Integer
- **Description**: High precision grid for integration. There are three parameters.
- **Default**: 4 4 4

adaptive_grid_threshold

- **Type**: Real
- **Description**: If the value of a k point is greater than this value, then the k point will be adapted.
- **Default**: 50.0

6.3.2 When integrate_mode is 'Adaptive'

relative_error

- **Type**: Real
- **Description**: The relative error of the adaptive integral.
- **Default**: 1e-6

absolute_error

- **Type**: Real
- **Description**: The absolute error of the adaptive integral.
- **Default**: 0.1

initial_grid

- **Type**: Real
- **Description**: The initial grid for adaptive integration. There are three parameters.
- **Default**: 1 1 1