
WannierTools Documentation

Release 2.4.2

QuanSheng Wu

May 17, 2022

Contents

1	Updates	3
2	Preliminaries	5
2.1	Installation of WannierTools (Linux or Mac)	5
2.2	Introduction of input files	6
2.3	Capabilities of WannierTools	23
2.4	Symmetrization of wannier90_hr.dat New	45
3	Tutorial: learning WannierTools through examples	47
3.1	Bi2Se3 (3D strong TI)	47
3.2	MoS2 (2D QSHE)	56
3.3	WTe2 (Type II Weyl semimetal)	63
3.4	IrF4 (Nodal Chain metals)	63
3.5	FeSi (Weyl point in Phonon system)	63
4	Citation	65
5	Correspondence	67
6	Licence	69
7	Authors and contributions	71
8	Sponsors	73

WannierTools: an open-source software package for novel topological materials

Source on Github : https://github.com/quanshengwu/wannier_tools

Aiming to investigate topological properties of

- Electron systems (Tested)
- Phonon systems (Testing)

CHAPTER 1

Updates

WannierTools V2.4.0

Sep 1 2018 1. We have a big update in this version, since Changming Yue put his symmetrization code into WannierTools.

So now we can symmetrize the Wannier functions based tight binding model. This functionality is included in the wannhr_symm/ folder.

2. Fixed a bug about reading Rcut.
3. Distinguish two dual surface in the slab band calculation.
4. Update an example about calculating the mirror Chern number of ZrTe.

:blue:‘Develop branch 2018 June 27 We added the test version of phonon system. Welcome to git clone the develop branch and test it. git clone https://github.com/quanshengwu/wannier_tools.git‘

WannierTools V2.3.0

1. Fixed a bug.
2. Add Translate_to_WS_calc in the CONTROL namelist. This works for BulkFS_plane_calc.

WannierTools V2.2.9 1. Fixed several bugs.

2. Added two new functionalities:
 1. Calculate the energy levels at given k points in the KPOINTS_3D card. This is called the point mode.
 2. Calculate anomalous Hall conductivity (AHC).

WannierTools V2.2.6

1. Discard the Miller indices
2. Discard the third vector in the SURFACE card. The surface plane is specified only by two lattice vectors sitting on it.

2.1 Installation of WannierTools (Linux or Mac)

2.1.1 Prerequisites

You need to install the following (mandatory) packages:

- Fortran compiler (Gfortran or ifort)
- MPICH version higher than 2.1.5
- Lapack and Blas library

2.1.2 Compilation

First Check out the repository by

```
git clone https://github.com/quanshengwu/wannier_tools.git
```

Or download the .zip file directly from https://github.com/quanshengwu/wannier_tools, then uncompress it

Then Go into wannier_tools/src directory, Choose and Edit Makefile, Change the blas library "libs=" to your lapack+blas library

At present, we prepared 3 typical Makefiles, which are sequential+gfortran, sequential+ifort and mpi+ifort.

For the mpi compiler, you should switch on the compile flag "-DMPI", see Makefile.intel-mpi

After the compilation, the binary 'wann_tools' is copied to wannier_tools/bin/, you can put this path to the system PATH with

```
export PATH=/where/you/downloaded/wannier_tools/bin:$PATH
```

to the .bashrc file in your home directory.

2.1.3 Usage

Now you can enjoy your exploration for topological materials with WannierTools.

There are two files you have to prepare,

1. `wt.in`. All the control and user specified parameters are included in this file.
2. `wannier90_hr.dat`. Tight binding model constructed by Wannier90 or written in the format as `wannier90_hr.dat`.

After the preparation of these two files, you can just run `wann_tools` in the same folder

```
wt.x &
```

or in multi-cores

```
mpirun -np 4 wt.x &
```

The output information during the running are written in `WT.out`.

2.1.4 Plotting tools

1. `gnuplot`
2. `xmgrace`
3. `xcrysden`
4. `matlab`

2.2 Introduction of input files

Attention: From WannierTools 2.2, the name of input file changes from ‘`input.dat`’ to ‘`wt.in`’. The executable binary changes from ‘`wann_tools`’ to ‘`wt.x`’

There are two input files you should prepare *wt.in* and *wannier90_hr.dat*

- *Main input file wt.in*
- *Special tags for phonon system (under testing)*
- *Tight-binding model wannier90_hr.dat*

2.2.1 Main input file wt.in

- *Basic Parameters*
- *TB_FILE*
- *SYSTEM*
- *CONTROL*
- *PARAMETERS*
- *LATTICE*
- *ATOM_POSITIONS*
- *PROJECTORS*

- *SURFACE*
- *KPATH_BULK*
- *KPOINTS_3D*
- *KPATH_SLAB*
- *KPLANE_SLAB*
- *KPLANE_BULK*
- *KCUBE_BULK*
- *EFFECTIVE_MASS*
- *SELECTED_ATOMS*
- *WANNIER_CENTRES*

Before executing `wann_tools`, you should cp the `wt.in` file in the directory `wannier_tools/example` by your own necessary.

For version later than 2.0, we updated the format of `wt.in`. The input file is structured in a number of **NAMELIST** and **INPUT_CARDS**.

Here we introduce the `wt.in` for Bi_2Se_3 as an example

```
&TB_FILE
Hrfile = 'wannier90_hr.dat'
Package = 'VASP'           ! obtained from VASP, it could be 'VASP', 'QE', 'Wien2k',
  ↳ 'OpenMx'
/

LATTICE
Angstrom
-2.069 -3.583614 0.000000      ! crystal lattice information
 2.069 -3.583614 0.000000
 0.000  2.389075  9.546667

ATOM_POSITIONS
5                               ! number of atoms for projectors
Direct                         ! Direct or Cartesian coordinate
Bi 0.3990    0.3990    0.6970
Bi 0.6010    0.6010    0.3030
Se 0         0         0.5
Se 0.2060    0.2060    0.1180
Se 0.7940    0.7940    0.8820

PROJECTORS
3 3 3 3 3                     ! number of projectors
Bi px py pz                   ! projectors
Bi px py pz
Se px py pz
Se px py pz
Se px py pz

SURFACE                        ! See doc for details
1 0 0
0 1 0

&CONTROL
```

(continues on next page)

(continued from previous page)

```

BulkBand_calc      = T
BulkFS_calc        = T
BulkGap_cube_calc  = T
BulkGap_plane_calc = T
SlabBand_calc      = T
WireBand_calc      = T
SlabSS_calc        = T
SlabArc_calc       = T
SlabQPI_calc       = T
SlabSpintexture_calc = T
Wanniercenter_calc = T
BerryCurvature_calc = T
EffectiveMass_calc  = T
/

&SYSTEM
NSLAB = 10          ! for thin film system
NSLAB1= 4           ! nanowire system
NSLAB2= 4           ! nanowire system
NumOccupied = 18    ! NumOccupied
SOC = 1             ! soc
E_FERMI = 4.4195    ! e-fermi
Bx= 0, By= 0, Bz= 0 ! Bx By Bz
surf_onsite= 0.0    ! surf_onsite
/

&PARAMETERS
Eta_Arc = 0.001     ! infinite small value, like brodening
E_arc = 0.0         ! energy for calculate Fermi Arc
OmegaNum = 100      ! omega number
OmegaMin = -0.6     ! energy interval
OmegaMax = 0.5      ! energy interval
Nk1 = 21            ! number k points  odd number would be better
Nk2 = 21            ! number k points  odd number would be better
Nk3 = 21            ! number k points  odd number would be better
NP = 1              ! number of principle layers
Gap_threshold = 1.0 ! threshold for GapCube output
/

KPATH_BULK          ! k point path
4                   ! number of k line only for bulk band
G 0.00000 0.00000 0.0000 Z 0.00000 0.00000 0.5000
Z 0.00000 0.00000 0.5000 F 0.50000 0.50000 0.0000
F 0.50000 0.50000 0.0000 G 0.00000 0.00000 0.0000
G 0.00000 0.00000 0.0000 L 0.50000 0.00000 0.0000

KPATH_SLAB
2                   ! number of k line for 2D case
K 0.33 0.67 G 0.0 0.0 ! k path for 2D case
G 0.0 0.0 M 0.5 0.5

KPLANE_SLAB
-0.1 -0.1          ! Original point for 2D k plane
0.2 0.0            ! The first vector to define 2D k plane
0.0 0.2            ! The second vector to define 2D k plane  for arc plots

KPLANE_BULK

```

(continues on next page)

(continued from previous page)

```

-0.00 -0.00 0.00 ! Original point for 3D k plane
1.00 0.00 0.00 ! The first vector to define 3d k space plane
0.00 0.50 0.00 ! The second vector to define 3d k space plane

```

KCUBE_BULK

```

-0.50 -0.50 -0.50 ! Original point for 3D k plane
1.00 0.00 0.00 ! The first vector to define 3d k space plane
0.00 1.00 0.00 ! The second vector to define 3d k space plane
0.00 0.00 1.00 ! The third vector to define 3d k cube

```

```

EFFECTIVE_MASS ! optional
2 ! The i'th band to be calculated
0.01 ! k step in unit of (1/Angstrom)
0.0 0.0 0.0 ! k point where the effective mass calculated.

```

WANNIER_CENTRES ! copy from wannier90.wout

Cartesian

```

-0.000040 -1.194745 6.638646
0.000038 -1.196699 6.640059
-0.000032 -1.192363 6.640243
-0.000086 -3.583414 2.908040
0.000047 -3.581457 2.906587
-0.000033 -3.585864 2.906443
-0.000001 1.194527 4.773338
0.000003 1.194538 4.773336
-0.000037 1.194536 4.773327
0.000006 -1.194384 1.130261
-0.000018 -1.216986 1.140267
0.000007 -1.172216 1.140684
0.000011 -3.583770 8.416406
-0.000002 -3.561169 8.406398
-0.000007 -3.605960 8.405979
0.000086 -1.194737 6.638626
-0.000047 -1.196693 6.640080
0.000033 -1.192286 6.640223
0.000040 -3.583406 2.908021
-0.000038 -3.581452 2.906608
0.000032 -3.585788 2.906424
0.000001 1.194548 4.773330
-0.000003 1.194537 4.773332
0.000037 1.194539 4.773340
-0.000011 -1.194381 1.130260
0.000002 -1.216981 1.140268
0.000007 -1.172191 1.140687
-0.000006 -3.583766 8.416405
0.000018 -3.561165 8.406400
-0.000007 -3.605935 8.405982

```

Basic Parameters

TB_FILE, **LATTICE**, **ATOM_POSITIONS**, **PROJECTORS** and **SURFACE** are the necessary basic parameters. They are used by almost all functionalities listed in **CONTROL** namelist.

NAMELISTS

NAMELISTS are a standard input construct in Fortran90. The use of NAMELISTS is relatively flexible. All the variables in the NAMELISTS have default values. You should set them only when it is needed. Variables can be inserted in any order. Such as

```
&NAMELIST
needed_variable2=XX, needed_variable1=YY,
character_variable1='a suitable string'
/
```

There are 4 NAMELISTS included in wt.in. They are *TB_FILE*, **SYSTEM**, **CONTROL**, **PARAMETERS**.

Note: If you want to comment one line, please use '!' instead of '#', because our codes were written in Fortran.

TB_FILE

Set the filename of the tight-binding Hamiltonian. At present, we use the format of wannier90_hr.dat specified in Wannier90.

```
&TB_FILE
Hrfile = 'wannier90_hr.dat'
Package = 'VASP' ! obtained from VASP, it could be 'VASP', 'QE', 'Wien2k',
→ 'OpenMx'
/
```

The default value for Hrfile is 'wannier90_hr.dat'. You could specify the first-principle package that used for obtaining wannier90_hr.dat. Default value for Package is 'VASP'. We support VASP, QE, Wien2k, OpenMx, Abinit at present. Please report new software package to me if you needed.

Note: Package is very important if you use QE to generate your tight binding model. Because the orbital order in QE is different from VASP, Wien2k et al.. And it will affect the results of spin texture. If you got strange spin texture, please carefully check this tag.

SYSTEM

In this namelists, we specify the system you need to compute.

```
&SYSTEM
Nslab = 10
Nslab1= 6
Nslab2= 6
NumOccupied = 18 ! NumOccupied
SOC = 1 ! soc
E_FERMI = 4.4195 ! e-fermi
Bx= 0, By= 0, Bz= 0 ! Bx By Bz
surf_onsite= 0.0 ! surf_onsite
/
```

- NSlab : integer, Number of slabs for slab band, The default value is 10.

- Nslab1, Nslab2 : integers, The thickness of nano ribbon. If you don't want to calculate the band structure of it, then don't set it. The default values are Nslab1= 1, Nslab2= 1.
- NumOccupied : integer, Number of occupied Wannier bands. No default value.

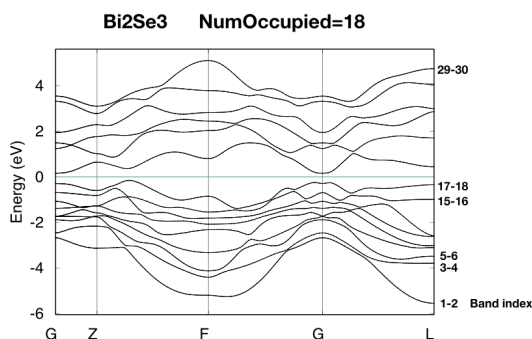
Important: please set NumOccupied correctly. It represents the "occupied" wannier bands, not the total number of electrons. In the calculation of Wilson loop, the Wilson loop is the trace of NumOccupied bands. If you want to study the topology between the 8th and the 9th band, then set NumOccupied=8.

When you search Weyl points, nodal line or study the gap in some k slices, NumOccupied is also a very important. WannierTools will look for touching point or calculate the energy gap between the NumOccupied'th band and the (NumOccupied+1)'th band.

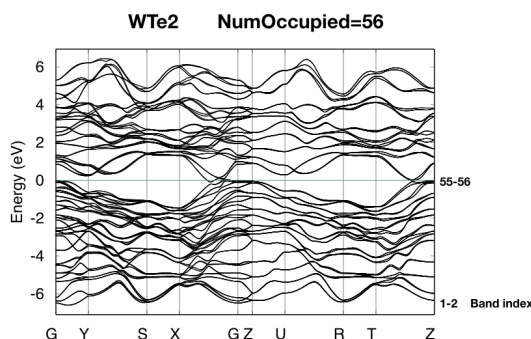
When you calculate the Fermi surface with BulkFS_calc= T, In order to save the storage, WannierTools only writes out 8(16) energy bands around NumOccupied'th band for SOC=0 (SOC=1) system into FS3D.bxsf.

If you don't put any physical meaning into this tag, then it is very easy to understand.

NumOccupied



Note: The energy band is calculated from wannier90_hr.dat, not from the DFT bands. In DFT calculations, there could be more bands below the Fermi level.



Here you could set NumOccupied =55 to study the nodal points between band 55 and band 56.

Or you could also set NumOccupied =57 to study the nodal points between band 57 and band 58.

Tips: xmgrace is a good software to figure out the band index.
xmgrace bulkek.dat &

- SOC : integer, Flag for spin-orbital coupling. If SOC=0, it means there is no SOC included in your given tight binding model. if SOC=1 or >0, it means SOC is already included in the tight binding model.
- E_FERMI : real-valued, Fermi level for the given tight binding model.
- Bx, By, Bz : real-valued, magnetic field value. Ignore it in this version.
- surf_onsite : real-valued, Additional onsite energy on the surface, you can set this to see how surface state changes. But don't set it if you don't know what it is.

CONTROL

In this name list, you can set the keywords to setup the tasks. All these tasks can be set to be true at the same time.

```
&CONTROL
BulkBand_calc           = T           ! bulk band structure calculation flag
```

(continues on next page)

(continued from previous page)

BulkFS_calc	= F
BulkGap_cube_calc	= F
BulkGap_plane_calc	= F
SlabBand_calc	= T
WireBand_calc	= F
SlabSS_calc	= T
SlabArc_calc	= F
SlabSpintexture_calc	= T
wanniercenter_calc	= F
BerryCurvature_calc	= F
/	

Note: New features : :red: FindNodes_calc; WeylChirality_calc; Z2_3D_calc; Chern_3D_calc

We listed those features in the table below.

Flag options	Function	Output	Tested
BulkBand_calc	Band structure for bulk	bulkek.dat, bulkek.gnu	yes
BulkFS_calc	Fermi surface for bulk system	FS3D.dat, FS3D.bxsf	yes
BulkGap_cube_calc	Energy gap for a given k cube for bulk system	GapCube.dat, GapCube.gnu	yes
BulkGap_plane_calc	Energy gap for a given k plane for bulk system	GapPlane.dat, GapPlane.gnu	yes
FindNodes_calc	Find touching points between the Numoccpued'th band and (Numoccpued+1)'th band	Nodes.dat, Nodes.gnu	yes
SlabBand_calc	Band structure for 2D slab system	slabek.dat,slabek.gnu	yes
WireBand_calc	Band structure for 1D ribbon system	ribbonek.dat,ribbonek.gnu	yes
Dos_calc	Density of state for 3D bulk system	dos.dat	yes
JDos_calc	Joint Density of state for 3D bulk system	jdos.dat	yes
SlabSS_calc	Surface spectrum A(k,E) along a kline and energy interval for slab system	dos.dat_l, dos.dat_r, dos.dat_bulk,surfdos_l.gnu, surfdos_r.gnu, surf- dos_l_only.gnu, surf- dos_r_only.gnu, surf- dos_bulk.gnu	yes
SlabArc_calc	Surface spectrum A(k,E0) for fixed energy E0 in 2D k-plane for slab system	arc.dat_l, arc.dat_r, arc_l.gnu, arc_r.gnu, arc_l_only.gnu, arc_l_only.gnu,	yes
SlabQPI_calc	Surface QPI for fixed energy E0 in 2D k-plane for slab system	arc.dat_l, arc.dat_r, arc_l.gnu, arc_r.gnu, arc_l_only.gnu, arc_l_only.gnu, arc_l_only.gnu, arc.jdat_l, arc.jdat_r, arc.jsdat_l, arc.jsdat_r, arc_l_jdos.gnu, arc_l_jsdos.gnu, arc_r_jdos.gnu, arc_r_jsdos.gnu,	yes
SlabSpintexture_calc	Spin texture in 2D k-plane for slab system	spindos_l.dat spindos_r.dat spintext_l.gnu spintext_r.gnu spintext_l.dat spintext_r.dat	yes
wanniercenter_calc	Wilson loop of a given 3D k-plane for bulk system	wcc.dat, wcc.gnu	yes
Z2_3D_calc	Wilson loop in all 6 3D k-planes for bulk system Z2 number calculation	wanniercenter3D_Z2.gnu, wanniercenter3D_Z2_{1-6}.dat	yes
Chern_3D_calc	Wilson loop in all 6 3D k-planes for bulk system Chern number calculation	wanniercenter3D_Z2.gnu, wanniercenter3D_Z2_{1-6}.dat	yes
WeylChirality_calc	Weyl Chirality calculation for given k points	find chiralities in WT.out, wannier- center3D_Weyl.dat, wan- niercenter3D_Weyl_*.gnu	yes
BerryPhase_calc	Berry phase with a 3D k path for bulk system	find Berry phase in WT.out	Yes
BerryCurvature_calc	Berry Curvature in 3D k-plane for bulk system	BerryCurvature.dat, BerryCurvature.gnu	yes
2.2. Introduction of input files		Berrycurvature- normalized.dat Berrycurvature- normalized.gnu	13

PARAMETERS

In this namelists, we listed some parameters necessary in the task you specified in namelists CONTROL.

```
&PARAMETERS
E_arc = 0.0           ! energy for calculate Fermi Arc
Eta_Arc = 0.001      ! infinite small value, like broadening
OmegaNum = 200        ! omega number
OmegaMin = -0.6       ! energy interval
OmegaMax = 0.5        ! energy interval
Nk1 = 50              ! number k points
Nk2 = 50              ! number k points
Nk3 = 50              ! number k points
NP = 2                ! number of principle layers
Gap_threshold = 1.0    ! threshold for GapCube output
/
```

E_arc : real-valued, energy for calculate Fermi arc, used if SlabArc_calc = T. The default value is 0.0.

Eta_Arc : real-valued, infinite same value for broadening used in Green's function calculation. used if SlabArc_calc = T. Default value is 0.001.

[OmegaMin, OmegaMax] : real-valued, energy interval for surface state calculation. used if SlabSS_calc= T. No default value.

OmegaNum : integer valued, Number of slices in the energy interval [OmegaMin, OmegaMax]. used if SlabSS_calc= T. The default value is 100.

Nk1, Nk2, Nk3 : integer valued, Number of k points for different purpose. I will explain that later. Default value is Nk1=20, Nk2=20, Nk3=20.

NP : integer valued, Number of principle layers, see details related to iterative green's function. Used if SlabSS_calc= T, SlabArc_calc=T, SlabSpintexture_calc=T. Default value is 2. You need to do a convergence test by setting Np= 1, Np=2, Np=3, and check the surface state spectrum. Basically, the value of Np depends on the spread of Wannier functions you constructed. One thing should be mentioned is that the computational time grows cubically of Np.

Gap_threshold : real valued. This value is used when you do energy gap calculation like BulkGap_cube_calc=T, BulkGap_plane_calc=T. The k points will be printed out in a file when the energy gap is smaller than Gap_threshold.

Input Card

The second important format in wt.in is the input_card format, which is relatively fixed format. First, we need a keyword like LATTICE, which is name of this card. After this keyword, the number of lines is fixed until it is done. There are several INPUT_CARDS in the wt.in. There is no order between the INPUT_CARDS. And any comments or blank lines could be added between the INPUT_CARDS. Lets introduce them one by one.

LATTICE

In this card, we set three lattice vectors coordinates. For the unit, you can use both Angstrom and Bohr. However, in the program, we use Angstrom. Bohr unit will be transformed to Angstrom automatically. No default values for the LATTICE CARD.

```
LATTICE
Angstrom
-2.069 -3.583614 0.000000 ! crystal lattice information
```

(continues on next page)

(continued from previous page)

2.069	-3.583614	0.000000
0.000	2.389075	9.546667

ATOM_POSITIONS

In this card, we set the atom's position.

```

ATOM_POSITIONS
5                      ! number of atoms for projectors
Direct                ! Direct or Cartesian coordinate
Bi 0.3990 0.3990 0.6970
Bi 0.6010 0.6010 0.3030
Se 0 0 0.5
Se 0.2060 0.2060 0.1180
Se 0.7940 0.7940 0.8820

```

Note: 1. Here the atom means that the atoms with projectors. not the whole atoms in the unit cell. 2. You can use “Direct” or “Cartesian” coordinates. “Direct” means the fractional coordinate based on the primitive lattice vector listed in LATTICE CARDS.

PROJECTORS

In this card, we set the Wannier projectors for the tight binding.

```

PROJECTORS
3 3 3 3              ! number of projectors
Bi pz px py         ! projectors
Bi pz px py
Se pz px py
Se pz px py
Se pz px py

```

Here we don't take into account the spin degeneracy, only consider the orbital part. The name of orbitals should be “s”, “px”, “py”, “pz”, “dxy”, “dxz”, “dyz”, “dx2-y2”, “dz2”. I will add f orbitals latter. The order of the orbitals is very important if you want to analyze the symmetry properties. The default order in Wannier90 is “s”, “pz”, “px”, “py”, “dz2”, “dxz”, “dyz”, “dx2-y2”, “dxy”. You can find the orbital order from wannier90.wout.

Note: If you don't care about the calculation related to symmetry like mirror chern number. The order or the name is not important. So for the f electrons, please write 7 random orbitals like px or dz2 or what else you want.

SURFACE

Attention: from version v2.2.6 on, you can specify a surface with SURFACE card with only two lattice vectors.

MILLER_INDICES CARD

Miller indices form a notation system in crystallography for planes in crystal (Bravais) lattices. You can find more information from Wikipedia https://en.wikipedia.org/wiki/Miller_index. In WannierTools, you only need to specify three integers like

```
MILLER_INDICES
0 0 1
```

Note: Since this is very confusing, we discard it from version V2.2.6. You should notice that the Miller indices mentioned here are based on three vectors specified in LATTICE card.

SURFACE CARD

This card is very important for slabs calculation. You need to read the following text carefully

```
SURFACE          ! See doc for details
1  0  0          a11, a12, a13
0  1  0          a21 a22 a23
```

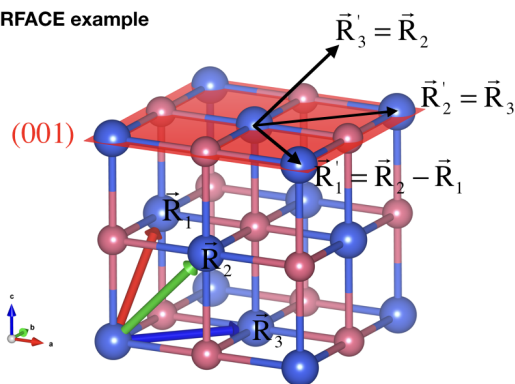
In this card, we specify the surface you want to investigate. Basically, you should be aware of which surface you want to investigate, and of which direction you want to study the ribbon. So we need to define the new lattice vector system like this,

Choose two **lattice vectors** on the surface we want to study, and choose another vector which is not on this plane.

The slab calculations are base on the surface constructed by vector R'_1, R'_2 .

Note: a11, a12, a13 ..., a23 should be integers, in unit of three lattice vectors

WannierTools SURFACE example



Primitive lattice vectors

$$\vec{R}_1 = (0.0, 0.5, 0.5)$$

$$\vec{R}_2 = (0.5, 0.0, 0.5)$$

$$\vec{R}_3 = (0.5, 0.5, 0.0)$$

Only valid for version V2.2.6 or later

If we want to study the (001) surface marked as read plane. We need to find two **lattice vectors** parallel to this plane. The first one is $R_2 - R_1$, the second one is R_3 .

$$\vec{R}'_1 = \vec{R}_2 - \vec{R}_1$$

$$\vec{R}'_2 = \vec{R}_3$$

Then the SURFACE card should look like this:

```
SURFACE
-1 1 0
0 0 1
```

The vectors in SURFACE is in unit of R_1, R_2, R_3

Q.S.Wu <http://www.wanniertools.com>

KPATH_BULK

This is the k path for bulk band structure calculation.

```
KPATH_BULK          ! k point path
4                    ! number of k line only for bulk band
G 0.00000 0.00000 0.0000 Z 0.00000 0.00000 0.5000
Z 0.00000 0.00000 0.5000 F 0.50000 0.50000 0.0000
F 0.50000 0.50000 0.0000 G 0.00000 0.00000 0.0000
G 0.00000 0.00000 0.0000 L 0.50000 0.00000 0.0000
```

These k points are in unit of the reciprocal lattice constant built by the lattice vector LATTICE CARD. The number of k points is $Nk1$, which is set in NAMELISTS PARAMETERS. There are no default values for this CARD. So you must set some value in the input file when choosing BulkBand_calc=T.

KPOINTS_3D

You can calculate the properties on some kpoints you specified in point mode. e.g. the energy bands

```
KPOINTS_3D
4          ! number of k points
Direct     ! Direct or Cartesian
0.00000 0.00000 0.0000
0.00000 0.00000 0.5000
0.50000 0.50000 0.0000
0.00000 0.00000 0.0000
```

The number of lines below “Direct” should be the same as the number above “Direct”. You could add comments at the end of each line. But you can’t add additional comment lines between the formatted lines.

KPATH_SLAB

This is the k path for slab system.

```
KPATH_SLAB
2          ! number of k line for 2D case
K 0.33 0.67 G 0.0 0.0 ! k path for 2D case
G 0.0 0.0 M 0.5 0.5
```

including the band structure calculation and the surface state calculation. It is necessary to set it when Slab-Band_calc=T or SlabSS_calc=T. Number of k points along the line is Nk1.

KPLANE_SLAB

Define a 2D k space plane for arc plots.

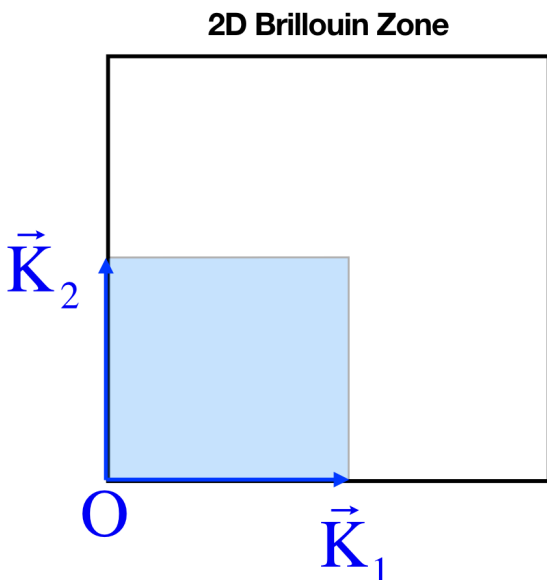
```
KPLANE_SLAB
-0.1 -0.1      ! Original point for 2D k plane
0.2  0.0      ! The first vector to define 2D k plane
0.0  0.2      ! The second vector to define 2D k plane for arc plots
```

The first line is the start point of the plane. The second and third line are the two vectors defining the plane. The number of k points for the 1st and 2nd vector is Nk1 and Nk2 respectively. Set this CARD when SlabArc_calc=T, SlabSpintexture_calc= T. The default values are

```
KPLANE_SLAB
-0.5 -0.5      ! Original point for 2D k plane
1.0  0.0      ! The first vector to define 2D k plane
0.0  1.0      ! The second vector to define 2D k plane for arc plots
```

WannierTools

KPLANE_SLAB example



KPLANE means a slice of the 2D momenta space.
SLAB means the 2D momenta space.

KPLANE_SLAB

0.0 0.0 ! O: the starting point of this slice
0.5 0.0 ! K1: the first vector of this slice
0.0 0.5 ! K2: the second vector of this slice

Q.S.Wu <http://www.wanniertools.com>

KPLANE_BULK

The same set as KPLANE_SLAB CARD, but for 3D case.

```
KPLANE_BULK
-0.50 -0.50 0.00 ! Original point for 3D k plane
1.00 0.00 0.00 ! The first vector to define 3d k space plane
0.00 1.00 0.00 ! The second vector to define 3d k space plane
```

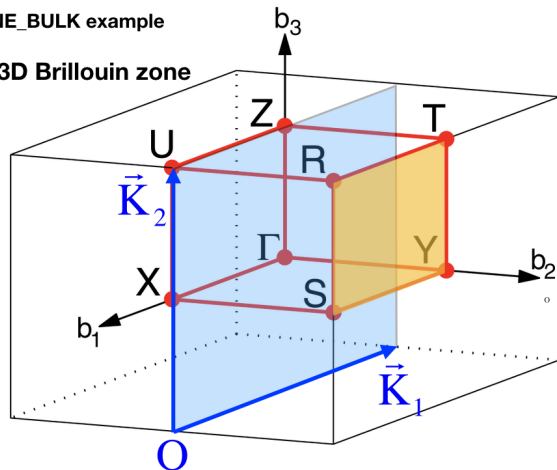
We can use these two vectors to calculate the band gap of a plane in 3D BZ, then we can check whether there are Weyl points or nodal line in that plane. Notice that these vectors is in unit of reciprocal vectors. Set this CARD when BulkGap_plane_calc=T, BerryCurvature_calc=T, wanniercenter_calc=T. Default values are

```
KPLANE_BULK
0.00 0.00 0.00 ! Original point for 3D k plane
1.00 0.00 0.00 ! The first vector to define 3d k space plane
0.00 0.50 0.00 ! The second vector to define 3d k space plane
```

WannierTools

KPLANE_BULK example

3D Brillouin zone



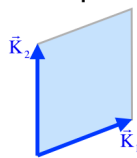
ORC path: Γ -X-S-Y- Γ -Z-U-R-T-Z|Y-T|U-X|S-R

[Setyawan & Curtarolo, DOI: 10.1016/j.commatsci.2010.05.010]

KPLANE means a slice of the 3D momenta space. BULK means the 3D momenta space.

The numbers below are in unit of the reciprocal lattice of the lattice vectors specified in the LATTICE card

Example I



Example I:

KPLANE_BULK

```
0.5 0.0 -0.5 ! O: the starting point of this slice
-1.0 0.0 0.0 ! K1: the first vector of this slice
0.0 0.0 1.0 ! K2: the second vector of this slice
```

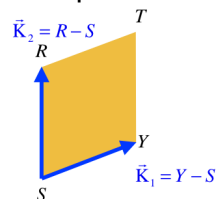
Example II:

KPLANE_BULK

```
0.5 0.5 0.0 ! S: the starting point of this slice
-0.5 0.0 0.0 ! K1: the first vector of this slice
0.0 0.0 0.5 ! K2: the second vector of this slice
```

Q.S.Wu <http://www.wanniertools.com>

Example II



KCUBE_BULK

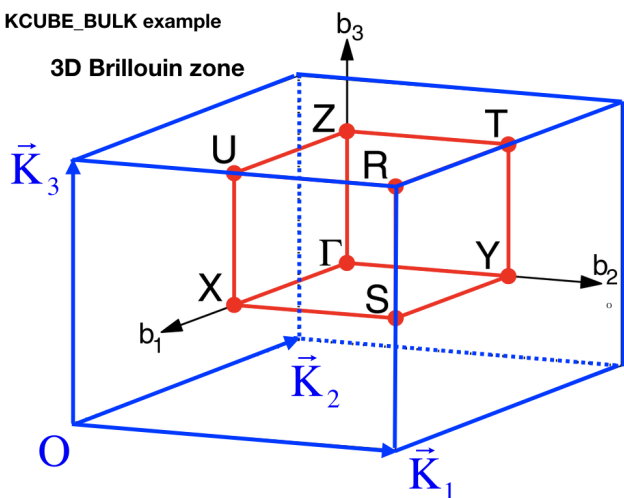
The same set as KPLANE_BULK CARD

```
KCUBE_BULK
-0.50 -0.50 -0.50 ! Original point for 3D k plane
1.00 0.00 0.00 ! The first vector to define 3d k space plane
0.00 1.00 0.00 ! The second vector to define 3d k space plane
0.00 0.00 1.00 ! The third vector to define 3d k cube
```

We add another k vector to construct a k cube. Set this for BulkGap_cube_calc=T. The values list above are default values.

WannierTools

KCUBE_BULK example



ORC path: Γ -X-S-Y- Γ -Z-U-R-T-Z|Y-T|U-X|S-R

[Setyawan & Curtarolo, DOI: 10.1016/j.commatsci.2010.05.010]

KCUBE means a cuboid of the 3D momenta space. BULK means the 3D momenta space.

The numbers below are in unit of the reciprocal lattice of the lattice vectors specified in the LATTICE card

The blue cuboid on the left side is specified as follows:

KCUBE_BULK

0.5 -0.5 -0.5 ! O: the starting point of this slice
 0.0 1.0 0.0 ! K1: the first vector of this cube
 -1.0 0.0 0.0 ! K2: the second vector of this cube
 0.0 0.0 1.0 ! K3: the third vector of this cube

The red cuboid defined by the high symmetry points on the left side is specified as follows:

KCUBE_BULK

0.0 0.0 0.0 ! Gamma : the starting point of this slice
 0.5 0.0 0.0 ! K1: the first vector of this cube
 0.0 0.5 0.0 ! K2: the second vector of this cube
 0.0 0.0 0.5 ! K3: the third vector of this cube

$$\vec{K}1 = X - \Gamma \quad \vec{K}2 = Y - \Gamma \quad \vec{K}3 = Z - \Gamma$$

For example:

$$DOS(\varepsilon) = \frac{1}{(2\pi)^3} \iiint_{Cube} \delta(\varepsilon_{nk} - \varepsilon) dk_x dk_y dk_z$$

Q.S.Wu <http://www.wanniertools.com>

EFFECTIVE_MASS

This card is set for effective mass calculation

```
EFFECTIVE_MASS      ! optional
2                   ! The i'th band to be calculated
0.01                ! k step in unit of (1/Angstrom)
0.0 0.0 0.0         ! k point where the effective mass calculated.
```

SELECTED_ATOMS

This card is useful if you want to get some energy spectrum that only projected on some specific atoms. For example, we can calculate the surface projected spin texture in the bulk system with vacuum.

The example is listed in the example/Bi2Se3-6Qlayers

```
SELECTED_ATOMS
2 ! number groups of selected atoms
6 12 18 24 30 ! top surface's atoms
1 7 13 19 25 ! bottom surface's atoms
```

WANNIER_CENTRES

This card will be usefull for Wilson loop calculations.

```
WANNIER_CENTRES      ! copy from wannier90.wout
Cartesian
-0.000040 -1.194745 6.638646
```

(continues on next page)

(continued from previous page)

0.000038	-1.196699	6.640059
-0.000032	-1.192363	6.640243
-0.000086	-3.583414	2.908040
0.000047	-3.581457	2.906587
-0.000033	-3.585864	2.906443
-0.000001	1.194527	4.773338
0.000003	1.194538	4.773336
-0.000037	1.194536	4.773327
0.000006	-1.194384	1.130261
-0.000018	-1.216986	1.140267
0.000007	-1.172216	1.140684
0.000011	-3.583770	8.416406
-0.000002	-3.561169	8.406398
-0.000007	-3.605960	8.405979
0.000086	-1.194737	6.638626
-0.000047	-1.196693	6.640080
0.000033	-1.192286	6.640223
0.000040	-3.583406	2.908021
-0.000038	-3.581452	2.906608
0.000032	-3.585788	2.906424
0.000001	1.194548	4.773330
-0.000003	1.194537	4.773332
0.000037	1.194539	4.773340
-0.000011	-1.194381	1.130260
0.000002	-1.216981	1.140268
0.000007	-1.172191	1.140687
-0.000006	-3.583766	8.416405
0.000018	-3.561165	8.406400
-0.000007	-3.605935	8.405982

Those centres can be obtained from wannier90.wout by searching “Final state”. The default values for this card are atomic positions.

2.2.2 Special tags for phonon system (under testing)

Now we have one testing version of phonon system, you can write to wuquansheng@gmail.com for testing. There are two steps for using WannierTools for phonon system.

1. Use `phonon_hr.py` to get the tight-binding Hamiltonian from the `FORCE_CONSTANTS` or `FORCE_SETS` which generated with `phonopy`. This part was written by Changming Yue ([yuechangming8 at gmail.com](mailto:yuechangming8@gmail.com)). You can write email to him to get the source. By default the hrfile name of the Hamiltonian is `phonopyTB_hr.dat`. You can change the name of it as whatever you want.

2. After the generation of hrfile. You need another `wt.in` file as the same as the electron system. One difference is that you need to specify **Particle = ‘phonon’** in the **TB_FILE** namelist like

```
&TB_FILE
Hrfile = 'phonopyTB_hr.dat'
Particle = 'phonon'
/
```

LO-TO splitting

The LO-TO splitting can be treated as a perturbation [see phonopy](#).

We need the following necessary CARDS in the **wt.in**. Take **FeSi** as an example

```

ATOM_MASS
2 ! number of types of atom, for FeSi, we have 2
4 4 ! number of atoms for each atom-type Fe4Si4
55.845 28.0855 ! atomic mass for each type of atom

LOTO_DT          ! Dielectric constant tensor
199.480 0 0
0 199.480 0
0 0 199.480

LOTO_BC          ! Born charge tensor for each type of atom
-4.3431500  0.6899200  -0.4140700
-0.4140800  -4.3431600  0.6899300
0.6898900  -0.4140600  -4.3431500
4.3909800  0.2300200  -0.1092900
-0.1093100  4.3909900  0.2300100
0.2300400  -0.1092800  4.3909800

```

LOTO_DT is a 3*3 matrix. LOTO_BC are Number-of-atom-types 3*3 matrices.

2.2.3 Tight-binding model wannier90_hr.dat

This file contains the TB parameters. Usually, it can be generated by **Wannier90**.

Of course, you can generate it from the Slater-Koster method or discretize k.p model onto a cubic lattice. The format should like this

```

written on 8May2016 at 13:57:00
  30
 547
 2  2  1  1  1  1  1  1  1  1  2  2  2  2  2
 1  1  1  2  1  1  1  2  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  4
 2  2  2  2  2  2  2  4  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  2  1
 1  1  1  1  1  1  2  1  1  1  1  1  1  1  1
 2  2  2  2  2  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  2  1  1  1  1
 1  1  1  2  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  2  2  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  2  1  1  1  2  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  2
 1  1  1  1  1  1  1  2  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  2  1  1  1  2  1  1  1  1  1  1  1  1  1
 1  1  2  1  1  1  1  1  1  1  2  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
 1  1  1  1  1  1  1  1  1  1  1  2  2  1

```

(continues on next page)

(continued from previous page)

```

1      1      1      1      1      1      1      1      1      1      1      1      1      1      1
1      1      1      2      1      1      1      1      1      1      1      2      1      1      1
1      1      1      1      1      1      1      1      1      1      1      1      1      1      1
1      1      1      1      1      1      1      1      1      1      1      1      1      1      1
1      1      2      2      2      2      2      1      1      1      1      1      1      1      1
2      1      1      1      1      1      1      1      2      1      1      1      1      1      1
1      1      1      1      1      1      1      1      1      1      1      1      1      1      1
1      1      1      1      1      1      1      1      1      1      1      1      1      1      4
2      2      2      2      2      2      2      4      1      1      1      1      1      1      1
1      1      1      1      1      1      1      1      1      1      1      1      1      1      2
1      1      1      2      1      1      1      2      2      2      2      2      2      1      1
1      1      1      1      1      2      2
-6      2      -3      1      1      -0.000002      0.000003
-6      2      -3      2      1      0.000002      0.000017
-6      2      -3      3      1      -0.000053      0.000002
-6      2      -3      4      1      -0.000031      0.000002
-6      2      -3      5      1      0.000001      -0.000000
-6      2      -3      6      1      -0.000003      0.000002
-6      2      -3      7      1      0.000037      -0.000001
-6      2      -3      8      1      -0.000001      -0.000003
-6      2      -3      9      1      -0.000005      -0.000003
-6      2      -3     10      1      -0.000062      -0.000001
-6      2      -3     11      1      -0.000001      0.000001
-6      2      -3     12      1      -0.000031      0.000002
-6      2      -3     13      1      0.000011      -0.000000
-6      2      -3     14      1      -0.000001      0.000001
-6      2      -3     15      1      0.000003      0.000003
-6      2      -3     16      1      0.000000      -0.000010
-6      2      -3     17      1      -0.000010      -0.000001
-6      2      -3     18      1      -0.000000      -0.000008
-6      2      -3     19      1      0.000000      0.000000
-6      2      -3     20      1      0.000012      -0.000002
.....

```

1. The 1st line is a comment line with any content.
2. The 2nd line is the number of Wannier orbitals, in consideration of spin degeneracy. We call it NUM_WANNS
3. The 3rd line is the number of \mathbf{R} lattice vectors, we call it NRPTS
4. This section is about the degeneracy of \mathbf{R} points. If you generate wannier90_hr.dat by you self, please set it to 1. There are NRPTS number of 1.
5. This section gives the TB parameters. The first three integers are the coordinates or \mathbf{R} vectors in unit of three lattice vectors. The 4th and 5th column are the band index (Row first). The 6th and 7th are complex entities of the Hamiltonian.

2.3 Capabilities of WannierTools

- Bulk band calculation (points mode, line mode and plane mode)
- BulkFS calculation
- BulkFS plane calculation
- Bulk spin texture calculations **New**
- Density state(DOS) calculations

- *Find Nodes calculation*
- *Energy gap calculations (plane and cube mode)*
- *Slab band calculation*
- *Nanowire/nanoribbon band calculation*
- *Surface state ARPES calculation*
- *Surface state QPI calculation*
- *Fermi arc calculation*
- *Spin texture calculation*
- *Berry phase calculation*
- *Berry curvature calculation for 3D bulk case*
- *Berry curvature calculation for slab system* **New**
- *Anomalous Hall conductivity (AHC)*
- *Wannier charge center/Wilson loop calculation*
- *Z2 number for 3D bulk materials*
- *Chern number for 3D bulk materials*
- *Mirror Chern number calculation* **New**
- *Weyl Chirality calculation*
- *Landau level calculations*

2.3.1 Bulk band calculation (points mode, line mode and plane mode)

Points mode

You can calculate the energy bands with the given k points in the KPOINTS_3D *KPOINTS_3D* card.

Input

Typical flags for this mode in the wt.in.

```
&CONTROL
BulkBand_points_calc = T
/

KPOINTS_3D
4                ! number of k points
Direct          ! Direct or Cartesian
0.00000 0.00000 0.0000
0.00000 0.00000 0.5000
0.50000 0.50000 0.0000
0.00000 0.00000 0.0000
```

Output

The outputs for this mode is **bulkek-pointsmode.dat**

Line mode

Calculate bulk energy band for a series k lines. This is the basic calculation after the construction of Wannier functions. You have to compare your Wannier interpolated bands with the DFT bands. Those two bands should match well around the Fermi level.

Input

Typical flags for bulk band calculation in the wt.in.

```
&CONTROL
BulkBand_calc = T
/
&PARAMETERS
Nk1 = 101    ! Number of k points for each k line
/

KPATH_BULK    ! k point path
4              ! number of k lines only for bulk band
G 0.00000 0.00000 0.0000 Z 0.00000 0.00000 0.5000
Z 0.00000 0.00000 0.5000 F 0.50000 0.50000 0.0000
F 0.50000 0.50000 0.0000 G 0.00000 0.00000 0.0000
G 0.00000 0.00000 0.0000 L 0.50000 0.00000 0.0000
```

See *CONTROL*, *PARAMETERS*, *KPATH_BULK*

Output

The outputs for bulk band calculation are **bulkek.dat** and **bulkek.gnu**. You can get the band plot by running

```
gnuplot bulkek.gnu
```

or

```
xmgrace bulkek.dat
```

to get bandstructure plot.

The data structure for **bulkek.dat**

```
0.000000000    -2.673821992  119   80   80  119   80   80  205  138  138   70  40
↪  40   70   40   40   0   0   0   0   0   0   0   0   0   0   0   0
↪  0   0   0
0.016453872    -2.681536808  118   78   78  118   78   78  203  134  134   82  41
↪  41   82   41   41   0   0   0   0   0   0   0   0   0   0   0   0
↪  0   0   0
```

1. The 1st column represents k points for the given kpath (KPATH_BULK)
2. The 2nd column is the energy level

- From the 3rd to the n'th column are the projected weight of the wave function at each k point and each band onto each wannier orbitals. Those weights are normalized to 255 for the color plot convinence.

The subroutine for this feature is `ek_bulk.f90`.

Plane mode

Calculate band structure in a k slice(plane) specified by `KPLANE_BULK` card. The mode is very useful to visualize the Dirac/Weyl cone. You have to set the following tags in `wt.in`

```
&CONTROL
BulkBand_plane_calc = T
/
&PARAMETERS
Nk1 = 51      ! Number of k points along the first vector in KPLANE_BULK
Nk2 = 51      ! Number of k points along the second vector in KPLANE_BULK
/

KPLANE_BULK   ! fractional coordinates
0.00 0.00 0.30 ! Middle point for a k slice(plane) in 3D BZ. Usually, the
↪position of Dirac points.
0.50 0.00 0.00 ! The first vector to define k plane(slice) in 3D BZ
0.00 0.50 0.00 ! The second vector to define k plane(slice) in 3D BZ
```

The output file is **bulkek_plane.dat**, **bulkek_plane-matlab.dat** and **bulkek_plane.gnu**. You can get **bulkek_plane.png** with


```
gnuplot bulkek_plane.gnu
```

The **bulkek_plane-matlab.dat** is in MATLAB data format. You can plot the Dirac cone with matlab.

The format of **bulkek_plane.dat** is as follows:

# kx	ky	kz	k1	
↪k2	k3	E (Numoccupied-1)	E (Numoccupied)	E (Numoccupied+1)
↪E (Numoccupied+2)				
-0.299354337	-0.518496963	0.180167841	-0.518496936	-0.
↪299354384	0.180167841	-1.311721381	-1.311705191	0.
↪588683811	0.588872215			
-0.299354337	-0.504670376	0.180167841	-0.511583643	-0.
↪287380208	0.180167841	-1.294078082	-1.293904952	0.
↪586780093	0.587249790			
...				

Column 1-3rd are k points in cartesian coordinates. Column 4-6th are k points in a rotated cartesian coordinates where the x and y direction are line in the k plane and the z direction is perpendicular to the k plane you specified. Column 7-10th are energies at each k point. Here we only print out 4 energy bands around the fermilevel. It depends on **NumOccupied**. Usually, I choose column 4th and 5th as k coordinates and choose 8 and 9 as energy bands to show the Dirac cone shown below.

 bulkek_plane.png

2.3.2 BulkFS calculation

Bulk Fermi surface calculation.

Input

You should specify the number of k points for each three reciprocal vectors Nk_1 , Nk_2 , Nk_3 in NAMELISTS PARAMETERS

```
&CONTROL
BulkFS_calc = T
/
&PARAMETERS
Nk1 = 101    ! No. of slices for the 1st reciprocal vector
Nk2 = 101    ! No. of slices for the 2nd reciprocal vector
Nk3 = 101    ! No. of slices for the 3rd reciprocal vector
/

KCUBE_BULK
  0.00  0.00  0.00    ! Original point for 3D k plane
  1.00  0.00  0.00    ! The first vector to define 3d k space plane
  0.00  1.00  0.00    ! The second vector to define 3d k space plane
  0.00  0.00  1.00    ! The third vector to define 3d k cube
```

See *CONTROL*, *PARAMETERS*

Output

The outputs for this function are **FS3D.bxsf**. You can plot the FS with *xcrysden* run

```
xcrysden --bxsf FS3D.bxsf
```

to get the plot.

By the way, Bulk band and BulkFS calculations were already implemented in Wannier90 code.

2.3.3 BulkFS plane calculation

Bulk Fermi surface in a fixed k plane specified by KPLANE_BULK

Input

You should specify the number of k points for each three reciprocal vectors Nk_1 , Nk_2 in NAMELISTS PARAMETERS

```
&CONTROL
BulkFS_Plane_calc = T
/
&PARAMETERS
Nk1 = 101    ! No. of slices for the 1st reciprocal vector
Nk2 = 101    ! No. of slices for the 2nd reciprocal vector
/

KPLANE_BULK  ! in fractional coordinates
  0.00  0.00  0.00    ! Original point for 3D k plane
  1.00  0.00  0.00    ! The first vector to define 3d k space plane
  0.00  1.00  0.00    ! The second vector to define 3d k space plane
```

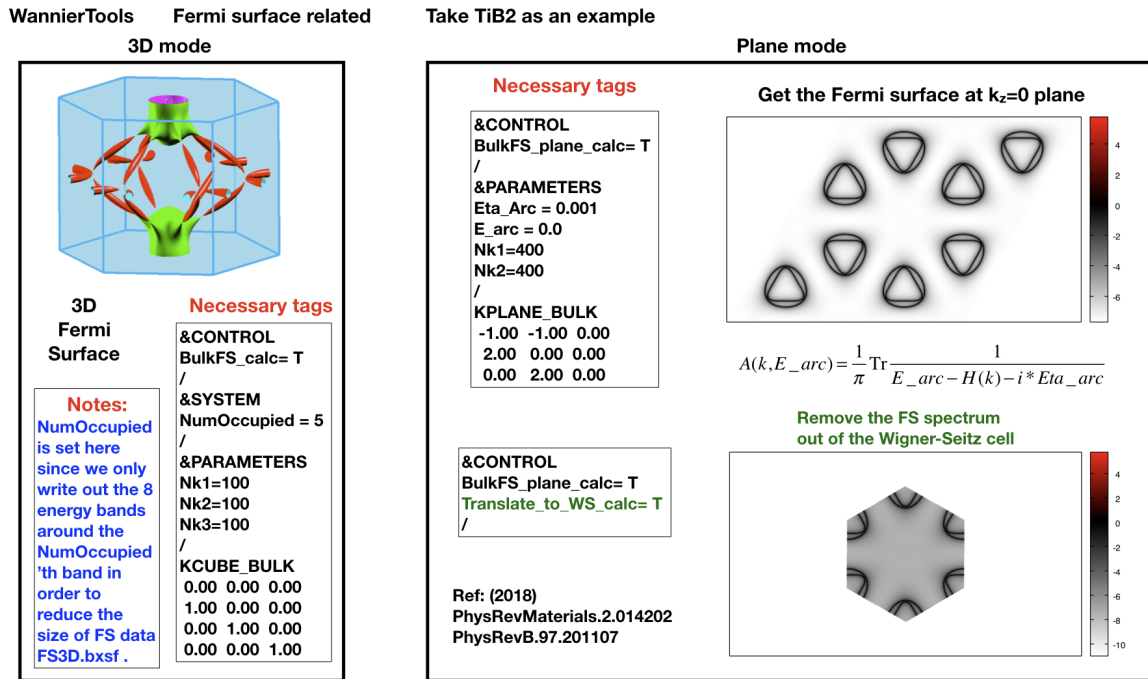
See *CONTROL*, *PARAMETERS*

Output

The outputs for this function are **fs.gnu**, **fs.png**.

gnuplot fs.gnu

to get the plot.



2.3.4 Bulk spin texture calculations

Calculate spin texture for bulk system that with vacuum or without inversion symmetry. For the bulk system with vacuum, you can calculate the surface projected spin texture. This is useful for comparing with the ARPES experiments. if you cut a slab system from a periodic tight binding model, then there is no charge relaxation on the surface which would change the surface state a lot. In this case, you have to do the first-principle calculations for a finite thickness slab system that with vacuum. Then you can construct Wannier functions for this system and use WannierTools to get the iso-energy plot of the Fermi surface (BulkFS_plane_calc=T) and get the surface projected spin texture (Bulkspintext_calc=T).

There is one example in the examples/Bi2Se3-6Qlayers.

2.3.5 Density state(DOS) calculations

Calculation density of state for the bulk system. The typical setup in **wt.in**:

```
&CONTROL
DOS_calc = T
/
&PARAMETERS
OmegaNum = 601      ! number of slices of energy
OmegaMin = -1.0     ! energy range for DOS
OmegaMax = 1.0
```

(continues on next page)

(continued from previous page)

```

Nk1 = 51    ! No. of slices for the 1st reciprocal vector
Nk2 = 51    ! No. of slices for the 2nd reciprocal vector
Nk3 = 51    ! No. of slices for the 3rd reciprocal vector
/

KCUBE_BULK
  0.00  0.00  0.00    ! Original point for 3D k plane
  1.00  0.00  0.00    ! The first vector to define 3d k space plane
  0.00  1.00  0.00    ! The second vector to define 3d k space plane
  0.00  0.00  1.00    ! The third vector to define 3d k cube

```

Outputs are **dos.dat** and **dos.gnu**. **dos.eps** will be obtained with

```
gnuplot dos.gnu
```

2.3.6 Energy gap calculations (plane and cube mode)

We support two modes for energy gap calculations. The formula is $gap(k) = E_{NumOccupied+1}(k) - E_{NumOccpuied}(k)$

Gap_plane mode

Calculate the energy gap for the k points in the KPLANE_BULK. This is useful to show Weyl points and nodal line structure.

Input

Typical input parameters for BulkGap_plane calculation

```

&CONTROL
BulkGap_Plane_calc = T
/
&PARAMETERS
Nk1 = 101    ! No. of slices for the 1st reciprocal vector
Nk2 = 101    ! No. of slices for the 2nd reciprocal vector
/

KPLANE_BULK
  0.00  0.00  0.00    ! Original point for 3D k plane
  1.00  0.00  0.00    ! The first vector to define 3d k space plane(slice)
  0.00  0.50  0.00    ! The second vector to define 3d k space plane(slice)

```

See *CONTROL*, *PARAMETERS*, *KPATH_BULK*

Output

The outputs for Gap_plane mode are **GapPlane.dat**, **GapPlane.gnu**. The **GapPlane.png** will get by

```
gnuplot GapPlane.gnu
```

The head of **GapPlane.dat**

kx	ky	kz	gap	Ev4	Ev3	
Ev2	Ev1	Ec1	Ec2	Ec3		
Ec4	k1	k2	k3			
0.00000000	0.00000000	0.00000000	0.45569845	-0.69109275	-0.	
→ 69109055	-0.29654328	-0.29654073	0.15915772	0.15915871	1.	
→ 24348171	1.24348457	0.00000000	0.00000000	0.00000000		
0.03796028	-0.02191637	0.00548462	0.43770730	-0.77636510	-0.	
→ 77598312	-0.26035113	-0.26027881	0.17742849	0.17771545	1.	
→ 29499437	1.29505298	0.00000000	0.02500000	0.00000000		

- Column 1-3 are the Cartesian coordinates of the k points in the KPLANE_BULK, in unit of $\frac{1}{\text{Angstrom}}$
- Column 4 is the energy gap
- Column 5-12 are the energy value for valence and conduction bands close to the Fermi level. There are 4 conduction bands and 4 valence bands.
- Column 13-15 are the Direct coordinates of the k points in the KPLANE_BULK

Gap_Cube mode

This helps us to find Weyl points and nodal line structure in the 3D BZ.

Input

Typical input parameters for BulkGap_cube calculation

```
&CONTROL
BulkGap_Cube_calc = T
/
&PARAMETERS
Nk1 = 101 ! No. of slices for the 1st reciprocal vector
Nk2 = 101 ! No. of slices for the 2nd reciprocal vector
Nk3 = 101 ! No. of slices for the 3rd reciprocal vector
/
KCUBE_BULK
-0.50 -0.50 -0.50 ! Original point for 3D k plane
1.00 0.00 0.00 ! The 1st vector to define 3d k cube
0.00 1.00 0.00 ! The 2nd vector to define 3d k cube
0.00 0.00 1.00 ! The 3rd vector to define 3d k cube
```

See [CONTROL](#), [PARAMETERS](#), [KCUBE_BULK](#)

Output

The outputs for Gap_plane mode are **GapCube.dat**, **GapCube.gnu**. The **GapCube.png** will get by

```
gnuplot GapCube.gnu
```

The head of **GapCube.dat** are

kx (1/Å)	ky (1/Å)	kz (1/Å)	Energy gap	Ev	
→Ec	k1 (2π/a)	k2 (2π/b)	k3 (2π/c)		
0.00000000	0.87665487	-0.54846229	0.79075142	-0.34827281	0.
→44247861	-0.50000000	-0.50000000	-0.50000000		
0.00000000	0.87665487	-0.51555455	0.86792416	-0.38635069	0.
→48157346	-0.50000000	-0.50000000	-0.45000000		

- Column 1-3 are the Cartesian coordinates of the k points where energy gap is small than Gap_threshold, in unit of $\frac{1}{\text{Angstrom}}$
- Column 4 is the energy gap. Those values are smaller than Gap_threshold, see [PARAMETERS](#)
- Column 5-6 are the energy value for valence and conduction bands close to the Fermi level. There are 4 conduction bands and 4 valence bands.
- Column 7-9 are the Direct coordinates of the k points.

2.3.7 Find Nodes calculation

Beside by using GapCube and GapPlane to find Weyl/Dirac nodes or node lines, we can directly using FindNodes function. $gap(k) = E_{NumOccupied+1}(k) - E_{NumOccupied}(k)$

Input

Typical input parameters for FindNodes_cube calculation

```
&CONTROL
FindNodes_calc = T
/
&PARAMETERS
Nk1 = 8      ! No. of slices for the 1st reciprocal vector
Nk2 = 8      ! No. of slices for the 2nd reciprocal vector
Nk3 = 8      ! No. of slices for the 3rd reciprocal vector
Gap_threshold = 0.0001 ! a value to determine which point should be identified as a
→node
/

KCUBE_BULK
-0.50 -0.50 -0.50  ! Original point for 3D k plane
 1.00  0.00  0.00  ! The 1st vector to define 3d k cube
 0.00  1.00  0.00  ! The 2nd vector to define 3d k cube
 0.00  0.00  1.00  ! The 3rd vector to define 3d k cube
```

Note: Please don't set Nk1, Nk2, Nk3 too large. Otherwise, it will become very time consuming. Usually, 15*15*15 is enough to get converged number of Weyl/Dirac points.

Output

Outputs are **Nodes.dat** and **Nodes.gnu**. Nodes.png will be obtained by

```
gnuplot Nodes.gnu
```

Here are heads of output for WTe2 **Nodes.dat**

# local minimal position and the related energy gap							
#	kx	ky	kz	gap	E	k1	k2
↪	k3						
	0.219436	-0.045611	-0.000001	0.000000	0.056688	0.121432	-0.045363
↪	-0.000003						
	-0.219515	-0.045063	-0.000001	0.000000	0.056461	-0.121476	-0.044818
↪	-0.000002						
	0.220195	-0.038682	-0.000002	0.000000	0.051264	0.121852	-0.038472
↪	-0.000003						
	-0.220183	-0.038936	-0.000001	0.000000	0.051618	-0.121845	-0.038724
↪	-0.000003						
	0.219514	0.045063	0.000001	0.000000	0.056459	0.121475	0.044818
↪	0.000003						
	-0.219434	0.045620	0.000002	0.000000	0.056692	-0.121431	0.045371
↪	0.000004						
	-0.220194	0.038678	0.000000	0.000000	0.051259	-0.121851	0.038468
↪	0.000001						
	0.220181	0.038941	0.000000	0.000000	0.051620	0.121844	0.038729
↪	0.000001						

You will find that there are 8 Weyl points in the BZ as expected.

2.3.8 Weyl Chirality calculation

After you identify the positions of Weyl points, you could use this function to calculate the chirality, which tells you whether a Weyl point is a sink or a source of the Berry Curvature.

Input

Typical input parameters for WeylChirality_calc calculation

```
&CONTROL
WeylChirality_calc = T
/
&PARAMETERS
Nk1 = 41    ! No. of slices for the 1st reciprocal vector, berry phase integration_
↪direction
Nk2 = 21    ! No. of slices for the 2nd reciprocal vector
/

WEYL_CHIRALITY
8           ! Num_Weyls
Cartesian   ! Direct or Cartesian coordinate
0.004       ! Radius of the ball surround a Weyl point
0.219436    -0.045611    -0.000000    ! Positions of Weyl points, No. of lines should_
↪larger than Num_weyls
-0.219515    -0.045063    -0.000000
0.220195    -0.038682    -0.000000
-0.220183    -0.038936    -0.000000
0.219514     0.045063     0.000000
-0.219434     0.045620     0.000000
-0.220194     0.038678     0.000000
0.220181     0.038941     0.000000
```

Output

Outputs are **wanniercenter3D_Weyl.dat** and **wanniercenter3D_Weyl_i.gnu**. **wanniercenter3D_Weyl.png** will be obtained by

```
gnuplot wanniercenter3D_Weyl_i.gnu

for ((i=1; i<9; i++)); do gnuplot wanniercenter3D_Weyl_$i.gnu;done
```

Note: *i* is an integer from 1 to Num_weyls

Here are heads of output for WTe2 **wanniercenter3D_Weyl.dat**

# Chirality	-1	1	1	-1	
→ 1	-1	1	-1		
# k	phase	phase	phase	phase	
→ phase	phase	phase	phase	phase	
0.00000000	0.99970932	0.00005854	0.00004671	0.99975139	0.
→00005851	0.99970861	0.00004736	0.99975087		
0.05000000	0.89229069	0.08696587	0.08941971	0.90855415	0.
→08723118	0.89170870	0.09022452	0.90795187		
0.10000000	0.79659821	0.16589558	0.17112299	0.82248889	0.
→16697194	0.79511289	0.17279423	0.82108022		

The first line shows the chiralities of each Weyl point. The first column is *k* point. From the 2nd to the last column show the Wannier charge center phase. In total, there are Num_weyls columns.

2.3.9 Slab band calculation

Before using iterative green's function to get the surface state spectrum for semi-infinite system. We also can just construct a finite thickness slab system and calculate the band structure for it.

Note: For slab calculations, please read carefully the input card *SURFACE*

Input

```
&CONTROL
SlabBand_calc = T
/
&SYSTEM
NSLAB = 10
/
&PARAMETERS
Nk1 = 101 ! No. of slices for the 1st reciprocal vector
/
KPATH_SLAB
2 ! number of k line for 2D case
K 0.33 0.67 G 0.0 0.0 ! k path for 2D case
G 0.0 0.0 M 0.5 0.5
```

See *CONTROL*, *SYSTEM PARAMETERS*, *KPATH_SLAB*

Output

Outputs are **slabek.dat** and **slabek.gnu**

The heads of **slabek.dat** are

```
0.0000000    -4.9575466    240
0.0508687    -5.0110528    226
0.1017373    -5.0566963    221
0.1526060    -5.0671994    220
...
```

- The 1st column are k points in the KPATH_SLAB
- The 2nd column are energy values.
- The 3rd column represent the surface weight, which is normalized to 255.

The colorfull plot **slabek.png** of the slab energy bands can be obtained by

```
gnuplot slabek.gnu
```

2.3.10 Nanowire/nanoribbon band calculation

Band calculation for wire system. Only one direction is periodic, the other two directions are confined.

Input

You don't have to set the k path, because it only has one direction.

```
&CONTROL
WireBand_calc = T
/
&SYSTEM
NSLAB1 = 4
NSLAB2 = 4
/
&PARAMETERS
Nk1 = 101    ! No. of slices for the 1st reciprocal vector
/
```

See *CONTROL*, *SYSTEM PARAMETERS*

Output

Outputs are **ribbonek.dat** and **ribbonek.gnu**. The data format of **ribbonek.dat** is the same as **slabek.dat**. Get plot **ribbonek.png** with

```
gnuplot ribbonek.gnu
```

2.3.11 Surface state ARPES calculation

One important feature for topological materials is the surface state. The bulk-edge correspondence tells us, if the topological property of the bulk system is nontrivial, then there will be nontrivial states on the surface. Nowadays,

there are several method to detect the surface states. One direct way is the angle resolved photoemission spectroscopy (ARPES). Such spectrum can be obtained by the iterative Green's function.

Note: For slab calculations, please read carefully the input card *SURFACE*

Input

```
&CONTROL
SlabSS_calc = T
/
&PARAMETERS
OmegaNum = 101
OmegaMin = -1.0
OmegaMax = 1.0
Nk1 = 101    ! No. of slices for the 1st reciprocal vector
NP = 2       ! principle layer
/
KPATH_SLAB
2            ! number of k line for 2D case
K 0.33 0.67 G 0.0 0.0 ! k path for 2D case
G 0.0 0.0 M 0.5 0.5
```

See *CONTROL*, *PARAMETERS*, *KPATH_SLAB*

NP : integer valued, Number of principle layers, see details related to iterative green's function. Used if SlabSS_calc=T, SlabArc_calc=T, SlabSpintexture_calc=T. Default value is 2. You need to do a convergence test by setting Np=1, Np=2, Np=3, and check the surface state spectrum. Basically, the value of Np depends on the spread of Wannier functions you constructed. One thing should be mentioned is that the computational time grows cubically of Np.

Output

The output files are **surfdos_l.dat**, **surfdos_r.dat**, **surfdos_l.gnu**, **surfdos_r.gnu**. You could get the the spectral function plots with

```
gnuplot surfdos_l.gnu
gnuplot surfdos_r.gnu
```

_l and **_r** means the top and bottom surface.

2.3.12 Surface state QPI calculation

Settings for this feature are almost the same as *Fermi arc calculation*. Only difference is that you should set

```
# please set SlabQPI_kplane_calc = T from V2.4.2
&CONTROL
SlabQPI_kplane_calc      = T
/
```

Output

There are a lot of outputs for QPI calculation. including **arc.dat_l**, **arc.dat_r**, **arc_l.gnu**, **arc_r.gnu**, **arc.jdat_l**, **arc.jdat_r**, **arc.jsdat_l**, **arc.jsdat_r**, **arc_l_jdos.gnu**, **arc_l_jsdos.gnu**, **arc_r_jdos.gnu**, **arc_r_jsdos.gnu**.

The gnu script with ‘only’ means we only plot the spectrum with the surface contribution, exclude the bulk contribution. (we remove file arc_l_only.gnu after v2.4.1 for the reason of misleading) jdat_l is the QPI data without consideration of spin scattering. jsdat_l is the QPI data in consideration of spin scattering.

2.3.13 Fermi arc calculation

Surface state spectrum at fixed energy level E_{arc} set in NAMELISTS PARAMETERS . Set SlabArc_calc=T, and set Nk1, Nk2, in NAMELISTS PARAMETERS, set k plane in KPLANE_SLAB CARD. Get the plots with “gnuplot arc_l.gnu”, “gnuplot arc_r.gnu”. **_l** and **_r** means the top and bottom surface.

2.3.14 Spin texture calculation

Spin texture calculation at a fixed energy level E_{arc} set in NAMELISTS PARAMETERS . Set Slabspintexture_calc=T, and set Nk1, Nk2, in NAMELISTS PARAMETERS, set k plane in KPLANE_SLAB CARD. Get the plots with “gnuplot spintext_l.gnu”, “gnuplot spintext_r.gnu”.

Note: Here we assume that the tight-binding basis are pure spin up or pure spin down, which means that the spin up and spin down are not mixed in the basis. This could be realized if you don’t do the maximal-localized step by setting num_iter=0 in wannier90.win and select the projectors, disentanglement window properly. If your Wannier functions are the maximal localized ones, then this feature doesn’t work. I suggest you using Wannier90 to get spin-texture which needs information from the first-principle calculations.

2.3.15 Berry phase calculation

Calculate Berry phase of a closed k path in 3D BZ. This is useful in a nodal line system. It is demonstrated that the Berry phase around a closed mirror symmetric k loop is either 0 or π for a mirror protect nodal line system.

In WannierTools, you can specify a k path by a serials k points. Here we take the WC example, which has two nodal lines around K point.

Input

```
&CONTROL
BerryPhase_calc = T
/
&SYSTEM
NumOccupied = 10          ! Number of occupied Wannier orbitals
/
&PARAMETERS
Nk1 = 21      ! No. of slices for the 1st reciprocal vector
/

KPATH_BERRY
11
Direct
0.3    0.333  -0.2
0.3    0.333  -0.1
0.3    0.333  -0.0
0.3    0.333   0.1
0.3    0.333   0.2
```

(continues on next page)

(continued from previous page)

0.33	0.333	0.2
0.33	0.333	0.1
0.33	0.333	0.0
0.33	0.333	-0.1
0.33	0.333	-0.2
0.3	0.333	-0.2

Output

The value of Berry phase can be found in the **WT.out**.

Note: 1. In principle, the Berry phase for around a nodal line should be interger. However, the MLWF-TB model usually is not symmetric. So the value of Berry phase is close to zero or one.

2. The first and the last kpoints in the KPATH_BERRY should be the same to form a loop.

2.3.16 Berry curvature calculation for 3D bulk case

Calculate Berry curvature at a fixed k plane in 3D BZ. Set BerryCurvature_calc=T, and set Nk1, Nk2, in NAMELISTS PARAMETERS, set k plane in KPLANE_BULK CARD. Get the plot with “gnuplot Berrycurvature.gnu”.

please set NumOccuied correctly. It represents the “occuied” wannier bands, not the total number of electrons. In this application, the Berrycurvature is the summation over NumOccupied bands.

A typical input (take ZrTe as an example):

```
&CONTROL
  BerryCurvature_calc=T
/
&SYSTEM
NumOccupied = 8          ! Number of occupied Wannier orbitals
/
&PARAMETERS
Nk1 = 101      ! No. of slices for the 1st reciprocal vector
Nk2 = 101      ! No. of slices for the 2st reciprocal vector
/

KPLANE_BULK
0.00 0.00 0.00 ! Central point for 3D k slice  k3=0
1.00 0.00 0.00 ! The first vector. Integrate along this direction to get WCC
0.00 1.00 0.00 ! WCC along this direction, for Z2, usually half of the reciprocal_
↪lattice vector
```

2.3.17 Berry curvature calculation for slab system

Note: Not well tested.. Use it carefully.

A typical input:

```

&CONTROL
  BerryCurvature_slab_calc=T
/

&SYSTEM
NumOccupied = 8   ! Number of occupied Wannier orbitals of the unit cell
/

&PARAMETERS
Nk1 = 101         ! No. of slices for the 1st reciprocal vector
Nk2 = 101         ! No. of slices for the 2st reciprocal vector
/

KPLANE_SLAB
0.00  0.00        ! Central point for 3D k slice  k3=0
1.00  0.00        ! The first vector. Integrate along this direction to get WCC
0.00  1.00        ! WCC along this direction, for Z2, usually half of the reciprocal_
↪lattice vector

```

2.3.18 Anomalous Hall conductivity (AHC)

Calculate AHC in the energy range [OmegaMin, OmegaMax]. AHC is in unit of (Ohm*cm)⁻¹.

Relevant inputs are

```

&CONTROL
AHC_calc=T
/

&PARAMETERS
OmegaNum = 601    ! number of slices of energy
OmegaMin = -1.0   ! energy range for AHC
OmegaMax = 1.0
Nk1 = 51          ! No. of slices for the 1st reciprocal vector
Nk2 = 51          ! No. of slices for the 2nd reciprocal vector
Nk3 = 51          ! No. of slices for the 3nd reciprocal vector
/

KCUBE_BULK
0.00  0.00  0.00  ! Original point for 3D k plane
1.00  0.00  0.00  ! The first vector to define 3d k space plane
0.00  1.00  0.00  ! The second vector to define 3d k space plane
0.00  0.00  1.00  ! The third vector to define 3d k cube

```

Output is **sigma_ahc.txt**.

2.3.19 Wannier charge center/Wilson loop calculation

Wannier charge center, which is sometimes called Wilson loop can be calculated by set WannierCenter_calc=T and set KPLANE_BULK CARD, set number of k points for two vectors is Nk1, Nk2 in NAMELISTS PARAMETERS. Notice: You should notice that the first vector in KPLANE_BULK CARD is the integration direction, this vector should be equal to one primitive reciprocal lattice vector. If you want to calculate the Z2 number, Please set the second vector to be half of the reciprocal lattice vector. You can get the Wannier charge center along the second k line. See more details In the paper written by Alexey. Soluyanov (2011). If you want to calculate the Chern number, Please set the second vector to be one primitive reciprocal lattice vector.

Note: Important: please set **NumOccupied** correctly. It represents the “occupied” wannier bands, not the total number of electrons. In this application, the Wilson loop is the trace of NumOccupied bands. If you want to study the topology between the 8th and the 9th band, then set NumOccupied=8.

Output

Outputs are **wcc.dat** and **wcc.gnu**, the format of **wcc.dat** is:

#	<i>k</i>	<i>largestgap</i>	<i>sum(wcc(:,ik))</i>	<i>wcc(i, ik) (i=1, NumOccupied)</i>		
0.00000000	0.60940556	0.99998388	0.00000850	0.07701431	0.	
↪07702018	0.19328973	0.19329593	0.28118760	0.28119336	0.	
↪49998615	0.50000060	0.71881052	0.71881646	0.80675987	0.	
↪80676424	0.92297767	0.92298328	0.99993530	0.99994085	0.	
↪99999935						
0.00312500	0.61256609	0.99998716	0.00030351	0.00082300	0.	
↪07688154	0.07709302	0.19117885	0.19525313	0.27952027	0.	
↪28297172	0.49188658	0.50810192	0.71703027	0.72048573	0.	
↪80480135	0.80887751	0.92290159	0.92311931	0.99908212	0.	
↪99967575						
0.00625000	0.61569946	0.99999557	0.00061525	0.00168708	0.	
↪07668131	0.07711296	0.18887026	0.19709893	0.27796336	0.	
↪28485788	0.48373617	0.51625672	0.71514220	0.72204447	0.	
↪80295562	0.81118878	0.92287871	0.92332259	0.99821913	0.	
↪99936414						
.....						

The second column is the position of the largest gap of WCC. It is used for drawing a line to calculate the Z2 number (see A. Soluyanov 2011), From the fourth column to the last column, they are wcc for the occupied bands specified with “NumOccupied”. The third line is the summation of the WCC over all the “occupied” bands. It’s usefull for telling the Chern number.

Example

Take Bi₂Se₃ for example, we calculate the Wilson loop (WCC) at k₃=0 plane, where k₁, k₂, k₃ is in unit of reciprocal lattice vector. The you should set the particular inputs like this

```
&CONTROL
  WannierCenter_calc=T
/
&SYSTEM
NumOccupied = 10          ! Number of occupied Wannier orbitals
/
&PARAMETERS
Nk1 = 41      ! No. of slices for the 1st reciprocal vector
Nk2 = 41      ! No. of slices for the 2st reciprocal vector
/

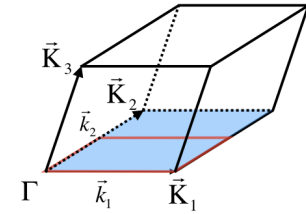
KPLANE_BULK
0.00 0.00 0.00 ! Original point for 3D k slice  k3=0
1.00 0.00 0.00 ! The first vector. Integrate along this direction to get WCC
0.00 0.50 0.00 ! WCC along this direction, for Z2, usually half of the reciprocal_
↪lattice vector
```

For 2D materials stacked along z direction, you could think it as a 3D material with weak coupling along z direction. You can use this function to get the Z2 value at $k_3=0$ plane to characterize the topology.

Use “gnuplot wcc.gnu” to get “wcc.eps” plot.

Here is an example.

$$\text{WannierTools} \quad Wcc_n(\vec{k}_2) = \frac{i}{2\pi} \int_0^{2\pi} d\vec{k}_1 \left\langle u_{n,\vec{k}_1,\vec{k}_2} \left| \partial_{\vec{k}_1} \right| u_{n,\vec{k}_1,\vec{k}_2} \right\rangle$$



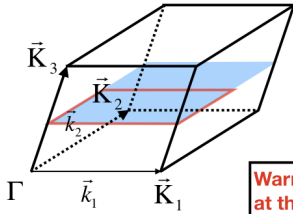
Example 1: WCC at $k_3=0$ plane

Wanniercenter_calc=T helps you to calculate Wilson loop in the time-reversal invariant plane (TRIP). In the left panel, K1, K2, K3 are three reciprocal lattice vectors. The light blue plane is one kind of such TRIP. The Wilson loop (WCC) is calculated numerically in the red rectangle shaped area. The integrate direction is along k_1 , the WCC is calculated along k_2 . So the KBULK_PLANE should be specified as

KBULK_PLANE

0.0 0.0 0.0 ! One corner point of the red rectangle area
1.0 0.0 0.0 ! The integrate direction
0.0 0.5 0.0 ! WCC(k_2), only half of K_2 is need.

All three vectors are in units of the reciprocal lattice vectors.



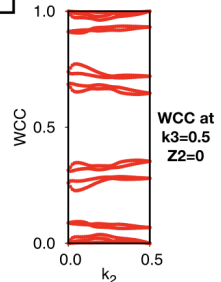
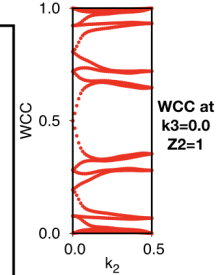
Example 2: WCC at $k_3=0.5$ plane

KBULK_PLANE

0.0 0.0 0.5 ! One corner point of the red rectangle area
1.0 0.0 0.0 ! The integrate direction
0.0 0.5 0.0 ! WCC(k_2), only half of K_2 is need.

Warning: you have to make sure that the gap of all k points at the TRIP should larger than zero. i.e.
Gap(k)= E(NumOccupied+1)- E(NumOccupied) >0
Otherwise, it is meaningless to calculate WCC.

3D TI Bi2Se3



Q.S.Wu <http://www.wanniertools.com>

2.3.20 Mirror Chern number calculation

At present, We can only calculate mirror Chern number for the simplest case (1. There is only one atom per atom's type in the unit cell e.g. ZrTe. 2. $k_z=0$ is the mirror plane we concern). For the more complex case, you can modify the source code by setting the mirror operator properly. Define your own mirror operator based on the atomic like Wannier functions in the symmetry.f90 and change the subroutine wanniercenter_mirror in wanniercenter.f90.

After properly setting of the mirror operator, you can run WannierTools with the basic parameters and the following additional parameters (Here we take ZrTe at $k_z=0$ plane as an example)

```
&CONTROL
  MirrorChern_calc=T
/
&SYSTEM
NumOccupied = 8          ! Number of occupied Wannier orbitals
/
&PARAMETERS
Nk1 = 101    ! No. of slices for the 1st reciprocal vector
Nk2 = 101    ! No. of slices for the 2st reciprocal vector
/

KPLANE_BULK
0.00 0.00 0.00 ! Original point for 3D k slice k3=0
1.00 0.00 0.00 ! The first vector. Integrate along this direction to get WCC,
↳should be a close path
0.00 1.00 0.00 ! WCC along this direction, for Chern, usually one reciprocal
↳lattice vector
```

(continues on next page)

(continued from previous page)

Output

The mirror Chern number can be found in the WT.out. The WCC/Wilson loop is included in the files **wcc-mirrorminus.dat** and **wcc-mirrorplus.dat**. The gnuplot script is **wcc-mirrorchernnumber.gnu**. The format of **wcc-mirrorplus.dat** is:

#	k	sum(wcc(:,ik))	wcc(:, ik)			
0.00000000		0.93401098	0.26748313	0.33122324	0.37761566	0.
→95768895						
0.01000000		0.93458410	0.26776394	0.33149191	0.37747362	0.
→95785463						
0.02000000		0.93515725	0.26806334	0.33205065	0.37717770	0.
→95786557						
0.03000000		0.93572256	0.26838206	0.33288980	0.37673021	0.
→95772050						
...						

The first column is $k=i/Nk_2$ ($i=0, Nk_2$), we take the second vector defined in KPLANE_BULK as unit of 1. The second line is the summation of the WCC over all the “occupied/2” bands. It’s useful for telling the Chern number. From the third column to the last column, they are wcc for the occupied/2 bands specified with “NumOccupied”.

2.3.21 Z2 number for 3D bulk materials

We can get Z2 topological index ($v_0, v_1v_2v_3$) from the z2 calculations of six time reversal invariant planes, i.e. (a) $k_1=0.0$; (b) $k_1=0.5$; (c) $k_2=0.0$; (d) $k_2=0.5$; (e) $k_3=0.0$; (f) $k_3=0.5$; Usually, you can call “Wannier charge center calculation for a plane” six times. Here we packed them up to get another function. You can set the input file like the following.

Input

The necessary tags that you should set in the wt.in

```
&CONTROL
Z2_3D_calc = T
/
&PARAMETERS
NumOccupied = 18 ! No. of occupied wannier bands
Nk1 = 41 ! No. of slices of the k points for WCCs
Nk2 = 41 ! No. of slices of the k points for WCCs
/
```

Output

Outputs are **wanniercenter3D_Z2_1.dat**, **wanniercenter3D_Z2_2.dat**, **wanniercenter3D_Z2_3.dat**, **wanniercenter3D_Z2_4.dat**, **wanniercenter3D_Z2_5.dat**, **wanniercenter3D_Z2_6.dat** and **wanniercenter3D_Z2.gnu**. The z2 value can be found in the WT.out by searching “z2 number for 6 planes”. The WCC (Wilson loop) plots **wanniercenter3D_Z2.eps** can be got with:

```
gnuplot wanniercenter3D_Z2.gnu
```

Note: Important: please set **NumOccpuied** correctly. It represents the “occpuied” wannier bands, not the total number of electrons. In this application, the Wilson loop is the trace of NumOccupied bands. If you want to study the topology between the 8th and the 9th band, then set NumOccupied=8.

About the Z2 index for 3D system.

$$v0 = (z2(ki=0) + z2(ki=0.5)) \bmod 2$$
$$vi = z2(ki=0.5)$$

For the 2D system, if you set the Z axis as the stack axis, please only take the Z2 number at $k_3=0$ plane.

2.3.22 Chern number for 3D bulk materials

Basically, you can calculate the Chern number for a closed manifold, for example, a 2D torus. For this purpose, I would suggest WannierCenter_calc=T in the calculation.

We can get Chern number of six k planes, i.e. $k_1=0.0$; $k_1=0.5$; $k_2=0.0$; $k_2=0.5$; $k_3=0.0$; $k_3=0.5$; where k_1, k_2, k_3 is in fractional units. Usually, you can call “Wannier charge center calculation for a plane” six times. Here we packed them up to get another function. You can set the input file like the following.

Basically, the method used here is very similar to the one used in the Z2 number calculations. We also use the Wilson loop method. However, for Z2 calculation, you only take half of the size of a time reversal invariant slice. For Chern number calculation, you need a closed momentum surface. For example, for the $k_1=0.0$ plane, half of the time reversal invariant slice would be defined like this:

```
k2 is in [0, 0.5]
k3 is in [0, 1]
```

The full closed momentum surface would defined like this

```
k2 is in [0, 1]
k3 is in [0, 1]
```

Input

The necessary tags that you should set in the wt.in

```
&CONTROL
Chern_3D_calc = T
/
&PARAMETERS
NumOccpuied = 18 ! No. of occupied wannier bands
Nk1 = 41 ! No. of slices of the k points for WCCs
Nk2 = 41 ! No. of slices of the k points for WCCs
/
```

Output

Outputs are **wanniercenter3D_Chern.dat** and **wanniercenter3D_Chern.gnu**. The Chern number can be found in the WT.out by searching “Chern number for 6 planes”. The WCC (Wilson loop) plots **wanniercenter3D_Chern.eps**

can be got with:

```
gnuplot wanniercenter3D_Chern.gnu
```

Note: Important: please set **NumOccupied** correctly. It represents the “occupied” wannier bands, not the total number of electrons. In this application, the Wilson loop is the trace of NumOccupied bands. If you want to study the topology between the 8th and the 9th band, then set NumOccupied=8.

For the 2D system, if you set the Z axis as the stack axis, please only take the Chern number at $k_3=0$ plane.

2.3.23 Landau level calculations

This functionality is under testing, not released yet. Developed by QSWu and YFGuan

By applying the uniform magnetic field along one lattice vector, the Landau level spectrum can be calculated by the Peierls substitution.

Here we put one example of Graphene. The input file wt.in is like this

```
&TB_FILE
Hrfile = 'wannier90_hr.dat'
/

!> bulk band structure calculation flag
&CONTROL
BulkBand_calc           = T
Hof_Butt_calc           = T
LandauLevel_k_calc      = T
LandauLevel_wavefunction_calc = F
/

&SYSTEM
NSLAB = 200             ! the size of magnetic supercell
NumOccupied = 1         ! NumOccupied
SOC = 0                 ! soc
E_FERMI = -1.2533       ! e-fermi
/

&PARAMETERS
E_arc = 0.00            ! energy for calculate Fermi Arc
OmegaNum = 201          ! number of eigenvalues to calculate the Landau levels
Nk1 = 100               ! number k points for each line in the kpath_bulk
/

LATTICE
Angstrom
2.1377110  -1.2342080   0.0000000
0.0000000   2.4684160   0.0000000
0.0000000   0.0000000   10.0000000

!> used when you want to study the projections on the orbital for each band
SELECTEDORBITALS
1  ! NumberofSelectedOrbitals without spin degeneracy
1  ! SelectedOrbitals indices without spin degeneracy
```

(continues on next page)

(continued from previous page)

```
ATOM_POSITIONS
2                      ! number of atoms for projectors
Direct                ! Direct or Cartisen coordinate
C 0.333333 0.666667 0.500000 C
C 0.666667 0.333333 0.500000 C

PROJECTORS
1 1                    ! number of projectors
C pz
C pz

SURFACE
0 0 1                 ! magnetic field direction in units of lattice vectors
1 0 0

KPATH_BULK            ! k point path
1                      ! number of k line only for bulk band
M 0.50000 0.00000 0.00000 G 0.00000 0.00000 0.00000

WANNIER_CENTRES
Cartesian
0.712570 1.234209 5.000000
1.425141 -0.000001 5.000000
```

We can calculate the Hofstadter butterfly plot by setting **Hof_Butt_calc = T**. **Nslab** is the size of the supercell. The corresponding magnetic field strength can be found in the output WT.out.

Since the calculation for magnetic supercell is very heavy, we have two versions of eigenvalue solvers. One is direct diagonalization, the other one is the ARPACK solver, which is based on the sparse matrix. So you have to install the ARPACK package, and specify the library in the Makefile. You don't have to choose the solvers. It is automatically chosen according to the matrix dimensionality. If the dimensionality of the Hamiltonian matrix of the magnetic supercell is larger than 1600, WannierTools will call the sparse matrix solver.

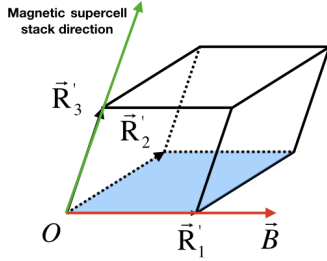
Once the sparse matrix solver is chosen. You have to set **E_arc** and **OmegaNum**, which means you are going to calculate $2 \times \text{OmegaNum} + 1$ Landau energy levels around **E_arc** respect to the Fermi level.

The magnetic field is along the first vector specified in the SURFACE card.

WannierTools Landau level

Tight-binding model: $H = \sum_{\langle i,j \rangle} t_{ij} c_i^\dagger c_j + \text{h.c.}$ $t_{ij} = \langle W_i | H | W_j \rangle$

Peierls substitution: $t_{ij} \rightarrow t_{ij} e^{-i\theta_{ij}}$ $\theta_{ij} = \frac{2\pi}{\phi_0} \int_i^j \mathbf{A} \cdot d\mathbf{l}$



Choose R_1' , R_2' , R_3' as described in the SURFACE card

```
SURFACE
0 0 1 ! Magnetic field
1 0 0
```

The direction of the magnetic field is parallel to the first vector in the SURFACE card.

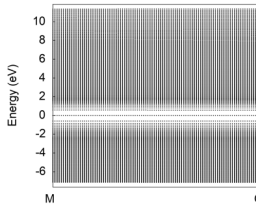
$$\vec{B} // \vec{R}_1'$$

The magnetic supercell stacking direction is along R_3' direction.

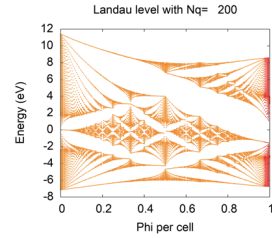
&CONTROL

```
BulkBand_calc = F
Hof_Butt_calc = T ! Calculate Hofstadter butterfly
LandauLevel_k_calc = T ! Calculate Landau level for given kpath
LandauLevel_wavefunction_calc = T
/
```

LandauLevel_k_calc = T
Landau level with Nq= 200



Hof_Butt_calc = T



2.4 Symmetrization of wannier90_hr.dat New

Here is a brief introduction of the symmetrization functionalities. Basically, wannhr_symm is an independent package based on Python2.7 written by Changming Yue (yuechangming8@gmail.com). Although this package is very useful, its requirement is also very restrict.

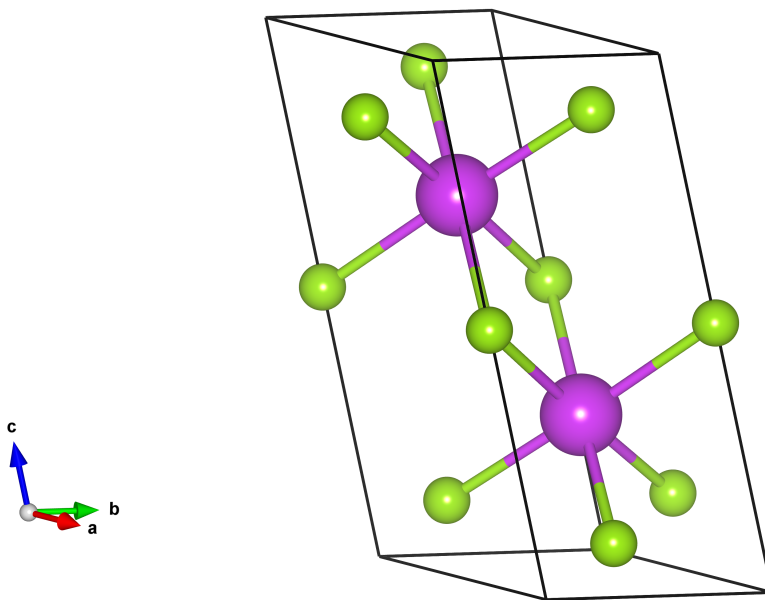
1. The Wannier functions must be atomic like, which means there should be very weak hybridization between orbitals. It's not possible to deal with the sp2, sp3 like wannier orbitals.
2. At present, it can only take care of the s, p, d, t2g, eg orbitals. p orbitals means all three orbitals ordered with pz, px, py. d orbitals are ordered as "dz2", "dxz", "dyz", "dx2-y2", "dxy". Such situation is automatically satisfied with in VASP+Wannier90
3. The file name of tight binding Hamiltonian should be 'wannier90_hr.dat'

Tutorial: learning WannierTools through examples

3.1 Bi₂Se₃ (3D strong TI)

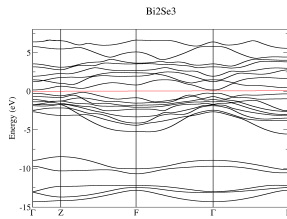
Bi₂Se₃ is a strong topological insulator. The Z₂ topological index is (1, 000). Theoretical and experimental validation can be found in Nature Physics 5, 438-442 (2009) and Nature Physics 5, 398-402 (2009) respectively. Here we show you how to use WannierTools to study strong topological materials. The input file and some related files are included in each distribution.

Here is the primitive unit cell of Bi₂Se₃



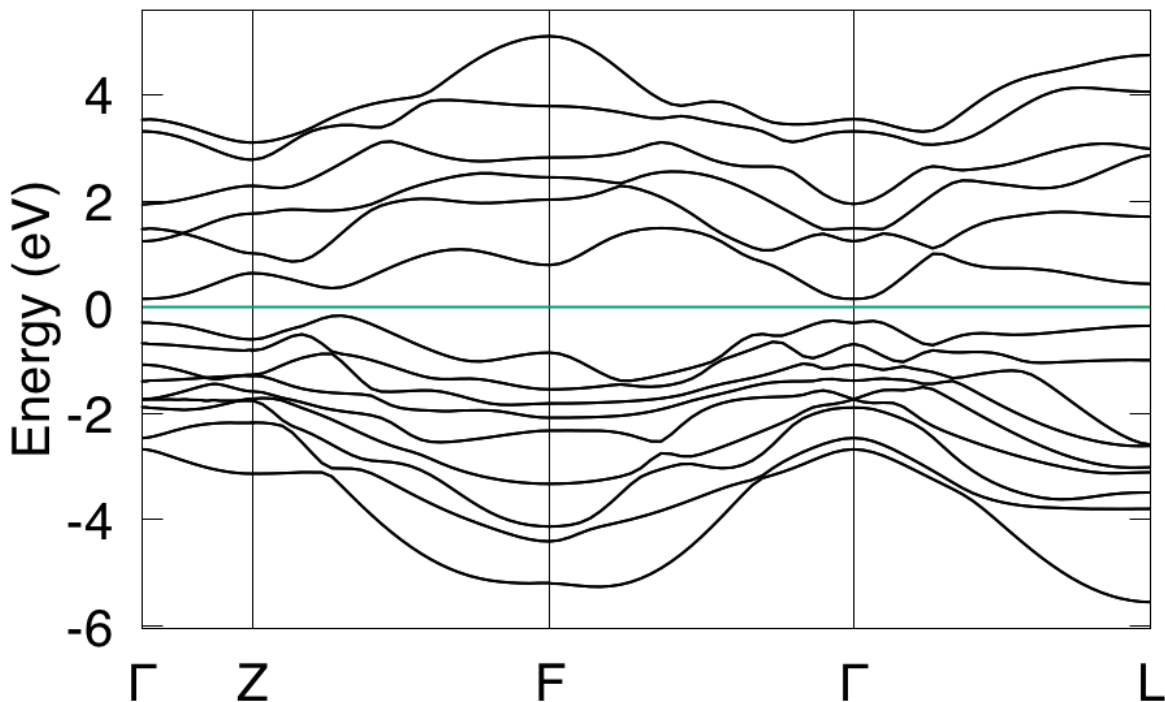
3.1.1 First principle calculation

Firstly, you need to study the electronic structure of Bi₂Se₃ with first-principle software package, like VASP, Wien2k, Abinit, Quantum-espresso et al. In this tutorial, we select VASP. Here is the calculated band structure.



3.1.2 Band structure

Then Wannier90 is applied to construct MLWF tight binding (TB) model (see more details from <http://www.wannier.org>). Here we only tell you that the p orbitals of Bi and Se are selected as the initial projectors for Wannier90. The band structure calculated from the MLWF-TB model is as follows



This band structure can be calculated directly from Wannier90. Also can be calculated from WannierTools. The settings in **WT.in** are

```
&CONTROL
BulkBand_calc      = T
/

&SYSTEM
SOC = 1             ! soc
```

(continues on next page)

(continued from previous page)

```

E_FERMI = 4.4195      ! e-fermi
/

&PARAMETERS
Nk1 = 41              ! number k points  odd number would be better
/

KPATH_BULK           ! k point path
4                    ! number of k line only for bulk band
G 0.00000 0.00000 0.0000 Z 0.00000 0.00000 0.5000
Z 0.00000 0.00000 0.5000 F 0.50000 0.50000 0.0000
F 0.50000 0.50000 0.0000 G 0.00000 0.00000 0.0000
G 0.00000 0.00000 0.0000 L 0.50000 0.00000 0.0000

```

3.1.3 Z2 topological number

From the band structure above, it is clear that it is a insulator. This is also can be checked by calculating the density of state (DOS). In order to identify the topological properties, we have to calculate the Z2 topological number, which is valid for time-reversal invariant system with a continuous full gap in the Brillouin Zone. The Z2 topological number for 3D bulk system can be obtained from the calculation of the Wilson loop (Wannier charge center) for the six time-reversal invariant momentum plane. $k_1=0.0$, $k_1=0.5$; $k_2=0.0$; $k_2=0.5$; $k_3=0.0$, $k_3=0.5$. It can be done using WannierTools with setting in **WT.in**

```

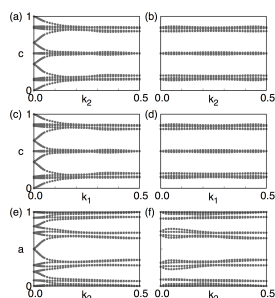
&CONTROL
Z2_3D_calc           = T
/

&SYSTEM
SOC = 1              ! soc
NumOccupied = 18      ! Number of occupied Wannier bands
/

&PARAMETERS
Nk1 = 41              ! number k points  odd number would be better
Nk2 = 41              ! number k points  odd number would be better
/

```

The results are

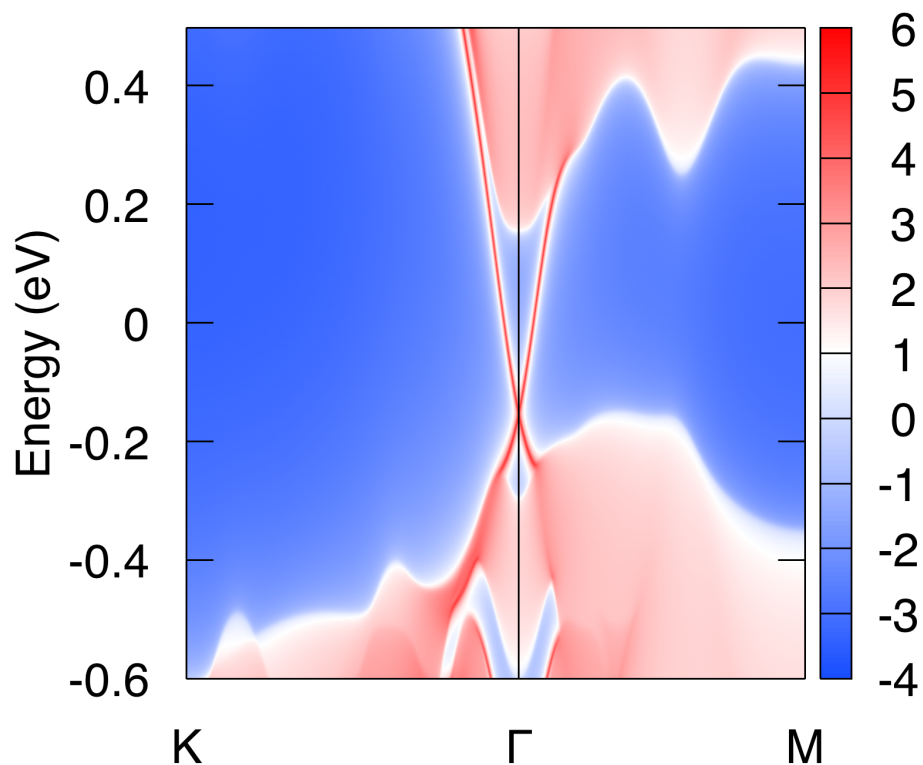


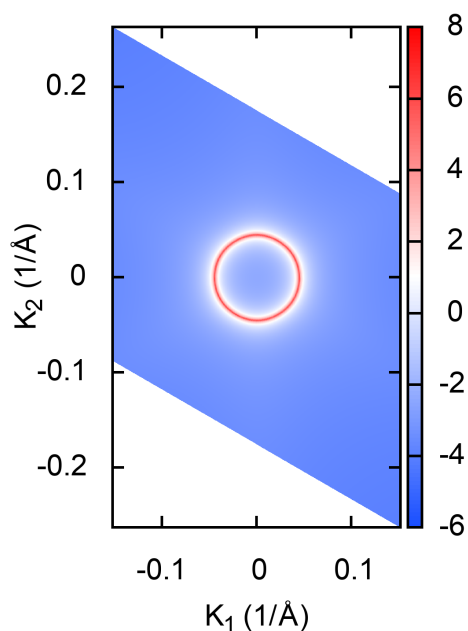
(a) $k_1=0.0$, $z_2=1$; (b) $k_1=0.5$, $z_2=0$; (c) $k_2=0.0$, $z_2=1$; (d) $k_2=0.5$, $z_2=0$; (e) $k_3=0.0$, $z_2=1$; (f) $k_3=0.5$, $z_2=0$;

So the bulk Z2 topological number is (1, 000), which means a strongly topological insulator.

3.1.4 Surface state

The surface states are the correspondence to the non-trivial bulk topology. They are detectable from ARPES experiments. The calculated surface states of Bi₂Se₃ on (0001) surface are





The settings in **WT.in** are

```
&CONTROL
SlabSS_calc      = T
SlabArc_calc     = T
/

&SYSTEM
SOC = 1           ! soc
NumOccupied = 18  ! Number of occupied Wannier bands
E_FERMI = 4.4195  ! e-fermi
/

&PARAMETERS
Eta_Arc = 0.001   ! infinite small value, like brodening
E_arc = 0.0       ! energy for calculate Fermi Arc
OmegaMin = -0.6   ! energy interval
OmegaMax = 0.5    ! energy interval
OmegaNum = 401    ! omega number
Nk1 = 101         ! number k points  odd number would be better
Nk2 = 101         ! number k points  odd number would be better
/

SURFACE          ! See doc for details
1 0 0
0 1 0
0 0 1

KPATH_SLAB
2               ! number of k line for 2D case
K 0.33 0.67 G 0.0 0.0 ! k path for 2D case
G 0.0 0.0 M 0.5 0.5
```

(continues on next page)

(continued from previous page)

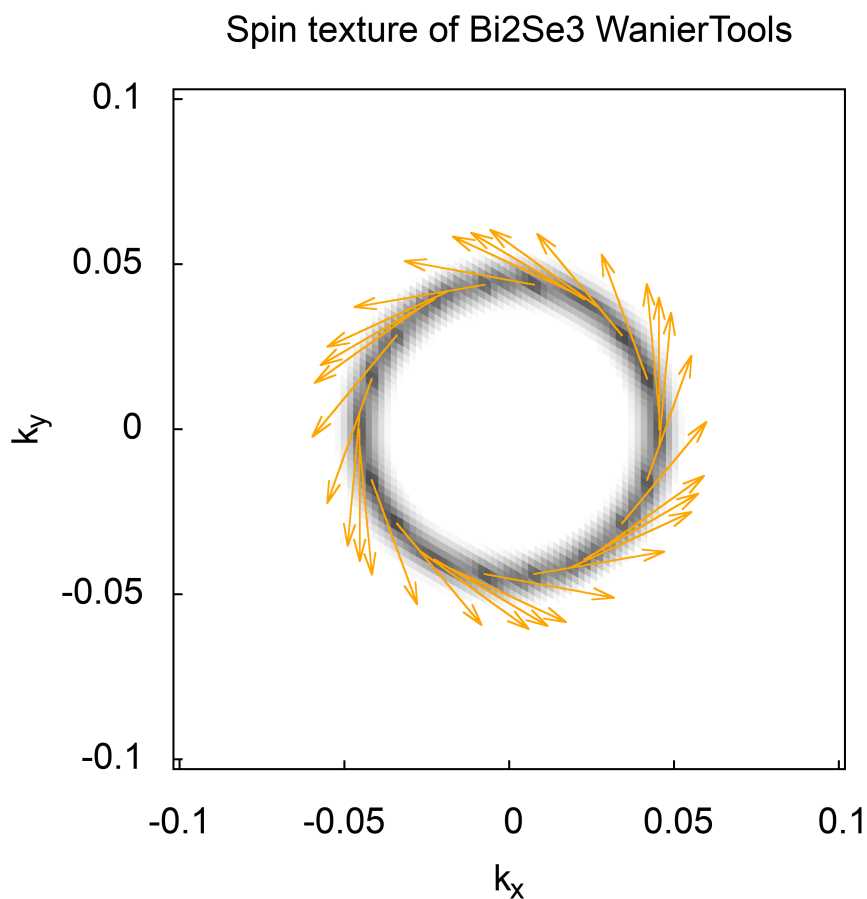
```

KPLANE_SLAB
-0.1 -0.1      ! Original point for 2D k plane
 0.2  0.0      ! The first vector to define 2D k plane
 0.0  0.2      ! The second vector to define 2D k plane  for arc plots

```

3.1.5 Spin texture

Spin orbital coupling (SOC) is a very important to topological insulator. The spin texture of the surface states will form due to SOC. WannierTools can calculate spin texture like this



by setting

```

&CONTROL
SlabSpintexture_calc = F

```

(continues on next page)

(continued from previous page)

```

/

&SYSTEM
SOC = 1                ! soc
E_FERMI = 4.4195        ! e-fermi
/

&PARAMETERS
Eta_Arc = 0.001         ! infinite small value, like brodening
E_arc = 0.0             ! energy for calculate Fermi Arc
Nk1 = 101               ! number k points  odd number would be better
Nk2 = 101               ! number k points  odd number would be better
/

SURFACE                ! See doc for details
  1  0  0
  0  1  0
  0  0  1

KPLANE_SLAB
-0.1 -0.1              ! Original point for 2D k plane
 0.2  0.0              ! The first vector to define 2D k plane
 0.0  0.2              ! The second vector to define 2D k plane  for arc plots

```

3.1.6 Full settings in WT.in of Bi2Se3

```

&TB_FILE
Hrfile = 'wannier90_hr.dat'
/

&CONTROL
BulkBand_calc          = T
BulkFS_calc            = F
BulkGap_cube_calc      = F
BulkGap_plane_calc     = F
SlabBand_calc          = T
WireBand_calc          = F
SlabSS_calc            = T
SlabArc_calc           = T
SlabQPI_calc           = F
Z2_3D_calc             = T
Chern_3D_calc          = F
SlabSpintexture_calc   = F
Wanniercenter_calc     = F
BerryPhase_calc        = F
BerryCurvature_calc    = F
EffectiveMass_calc     = F
/

&SYSTEM
NSLAB = 10             ! for thin film system
NSLAB1= 4              ! nanowire system
NSLAB2= 4              ! nanowire system
NumOccupied = 18       ! NumOccupied

```

(continues on next page)

(continued from previous page)

```

SOC = 1                ! soc
E_FERMI = 4.4195       ! e-fermi
Bx= 0, By= 0, Bz= 0    ! Bx By Bz
surf_onsite= 0.0       ! surf_onsite
/

&PARAMETERS
Eta_Arc = 0.001        ! infinite small value, like brodening
E_arc = 0.0            ! energy for calculate Fermi Arc
OmegaNum = 401         ! omega number
OmegaMin = -0.6        ! energy interval
OmegaMax = 0.5         ! energy interval
Nk1 = 41               ! number k points  odd number would be better
Nk2 = 41               ! number k points  odd number would be better
Nk3 = 21               ! number k points  odd number would be better
NP = 1                 ! number of principle layers
Gap_threshold = 1.0    ! threshold for GapCube output
/

LATTICE
Angstrom
-2.069 -3.583614 0.000000 ! crystal lattice information
 2.069 -3.583614 0.000000
 0.000  2.389075 9.546667

ATOM_POSITIONS
5                      ! number of atoms for projectors
Direct                ! Direct or Cartisen coordinate
Bi 0.3990 0.3990 0.6970
Bi 0.6010 0.6010 0.3030
Se 0 0 0.5
Se 0.2060 0.2060 0.1180
Se 0.7940 0.7940 0.8820

PROJECTORS
3 3 3 3 3            ! number of projectors
Bi px py pz          ! projectors
Bi px py pz
Se px py pz
Se px py pz
Se px py pz

SURFACE              ! See doc for details
1 0 0
0 1 0
0 0 1

KPATH_BULK           ! k point path
4                    ! number of k line only for bulk band
G 0.00000 0.00000 0.0000 Z 0.00000 0.00000 0.5000
Z 0.00000 0.00000 0.5000 F 0.50000 0.50000 0.0000
F 0.50000 0.50000 0.0000 G 0.00000 0.00000 0.0000
G 0.00000 0.00000 0.0000 L 0.50000 0.00000 0.0000

KPATH_SLAB
2                    ! number of k line for 2D case
K 0.33 0.67 G 0.0 0.0 ! k path for 2D case

```

(continues on next page)

(continued from previous page)

```

G 0.0 0.0 M 0.5 0.5

KPLANE_SLAB
-0.1 -0.1      ! Original point for 2D k plane
 0.2  0.0      ! The first vector to define 2D k plane
 0.0  0.2      ! The second vector to define 2D k plane   for arc plots

KPLANE_BULK
 0.00  0.00  0.50  ! Original point for 3D k plane  k3=0.5, bar{a}, along k1
 1.00  0.00  0.00  ! The first vector to define 3d k space plane
 0.00  0.50  0.00  ! The second vector to define 3d k space plane

!> The following 5 matrices are for backup using, will not affect the main input for
↪WannierTools
 0.00  0.00  0.00  ! Original point for 3D k plane  k3=0.0, bar{a}, along k1
 1.00  0.00  0.00  ! The first vector to define 3d k space plane
 0.00  0.50  0.00  ! The second vector to define 3d k space plane

 0.00  0.50  0.00  ! Original point for 3D k plane  k2=0.5, bar{c}, along ka
 0.00  0.00  1.00  ! The first vector to define 3d k space plane
 0.50  0.00  0.00  ! The second vector to define 3d k space plane

 0.00  0.00  0.00  ! Original point for 3D k plane  k2=0, bar{c}, along ka
 0.00  0.00  1.00  ! The first vector to define 3d k space plane
 0.50  0.00  0.00  ! The second vector to define 3d k space plane

 0.50  0.00  0.00  ! Original point for 3D k plane  k1=0.5, bar{c}, along kb
 0.00  0.00  1.00  ! The first vector to define 3d k space plane
 0.00  0.50  0.00  ! The second vector to define 3d k space plane

 0.00  0.00  0.00  ! Original point for 3D k plane  k1=0, bar{c}, along kb
 0.00  0.00  1.00  ! The first vector to define 3d k space plane
 0.00  0.50  0.00  ! The second vector to define 3d k space plane

KCUBE_BULK
-0.50 -0.50 -0.50  ! Original point for 3D k plane
 1.00  0.00  0.00  ! The first vector to define 3d k space plane
 0.00  1.00  0.00  ! The second vector to define 3d k space plane
 0.00  0.00  1.00  ! The third vector to define 3d k cube

EFFECTIVE_MASS      ! optional
2                   ! The i'th band to be calculated
0.01                ! k step in unit of (1/Angstrom)
0.0 0.0 0.0         ! k point where the effective mass calculated.

WANNIER_CENTRES      ! copy from wannier90.wout
Cartesian
-0.000040 -1.194745  6.638646
 0.000038 -1.196699  6.640059
-0.000032 -1.192363  6.640243
-0.000086 -3.583414  2.908040
 0.000047 -3.581457  2.906587
-0.000033 -3.585864  2.906443

```

(continues on next page)

(continued from previous page)

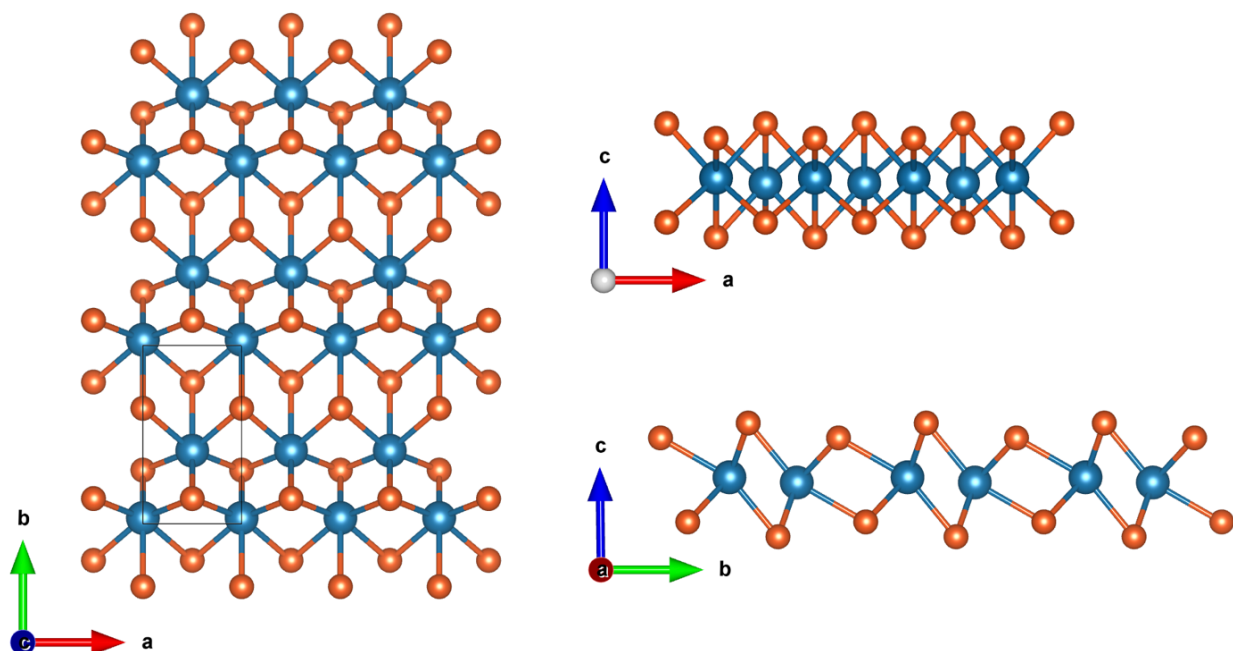
-0.000001	1.194527	4.773338
0.000003	1.194538	4.773336
-0.000037	1.194536	4.773327
0.000006	-1.194384	1.130261
-0.000018	-1.216986	1.140267
0.000007	-1.172216	1.140684
0.000011	-3.583770	8.416406
-0.000002	-3.561169	8.406398
-0.000007	-3.605960	8.405979
0.000086	-1.194737	6.638626
-0.000047	-1.196693	6.640080
0.000033	-1.192286	6.640223
0.000040	-3.583406	2.908021
-0.000038	-3.581452	2.906608
0.000032	-3.585788	2.906424
0.000001	1.194548	4.773330
-0.000003	1.194537	4.773332
0.000037	1.194539	4.773340
-0.000011	-1.194381	1.130260
0.000002	-1.216981	1.140268
0.000007	-1.172191	1.140687
-0.000006	-3.583766	8.416405
0.000018	-3.561165	8.406400
-0.000007	-3.605935	8.405982

3.2 MoS2 (2D QSHE)

WannierTools also works fine with 2D materials. The way we handle it is just like what we simulated in the first principle calculations. A 2D material is just a layer structured 3D material with zero coupling along z direction. The settings of WannierTools for 2D materials are the same of 3D materials. However, you should only care about the properties happen in the $k_z=0$ plane, since the properties are the same for different k_z .

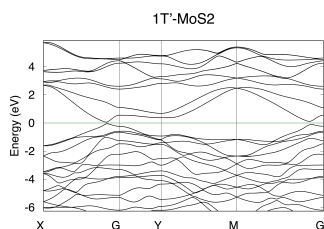
Monolayer square transition metal dichalcogenides (MoS2, MoSe2, MoTe2, WS2, WSe2, and WTe2) was predicted to be robust topological insulators (TIs) with $Z_2=1$. As a good 2D TI, 1T'-MoS2 is taken as an example to test WannierTools.

Here is the primitive unit cell of 1T'-MoS2



3.2.1 Band structure

Firstly, you need to study the electronic structure of MoS₂ with first-principle software package, like VASP, Wien2k, Abinit, Quantum-espreso et al. In this tutorial, we select VASP. Here is the calculated band structure. Then Wannier90 is applied to construct MLWF tight binding (TB) model (see more details from <http://www.wannier.org>). Here we only tell you that the s, d orbitals of Mo and s, p orbitals of S are selected as the initial projectors for Wannier90. The band structure calculated from the MLWF-TB model is as follows



This band structure can be calculated directly from Wannier90. Also can be calculated from WannierTools. The settings in **WT.in** are

```
&CONTROL
BulkBand_calc      = T
/

&SYSTEM
SOC = 1             ! soc
E_FERMI = -3.9151   ! e-fermi
/

&PARAMETERS
Nk1 = 101           ! number k points  odd number would be better
```

(continues on next page)

(continued from previous page)

```

/
KPATH_BULK          ! k point path
4                   ! number of k line only for bulk band
X 0.50000  0.00000  0.00000  G  0.00000  0.00000  0.00000
G 0.00000  0.00000  0.00000  Y  0.00000  0.50000  0.00000
Y 0.00000  0.50000  0.00000  M  0.50000  0.50000  0.00000
M 0.50000  0.50000  0.00000  G  0.00000  0.00000  0.00000

```

3.2.2 Z2 topological number

From the band structure above, it is clear that it is a insulator. This is also can be checked by calculating the density of state (DOS). In order to identify the topological properties, we have to calculate the Z2 topological number, which is valid for time-reversal invariant system with a continuous full gap in the Brillouin Zone. The Z2 topological number for 3D bulk system can be obtained from the calculation of the Wilson loop (Wannier charge center) for the six time-reversal invariant momentum plane. (a) $k_1=0.0$, (b) $k_1=0.5$; (c) $k_2=0.0$; (d) $k_2=0.5$; (e) $k_3=0.0$, (f) $k_3=0.5$. It can be done using WannierTools with setting in **WT.in**

```

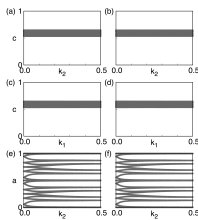
&CONTROL
Z2_3D_calc          = T
/

&SYSTEM
SOC = 1              ! soc
NumOccupied = 36     ! Number of occupied Wannier bands
/

&PARAMETERS
Nk1 = 101            ! number k points  odd number would be better
Nk2 = 41             ! number k points  odd number would be better
/

```

The results are

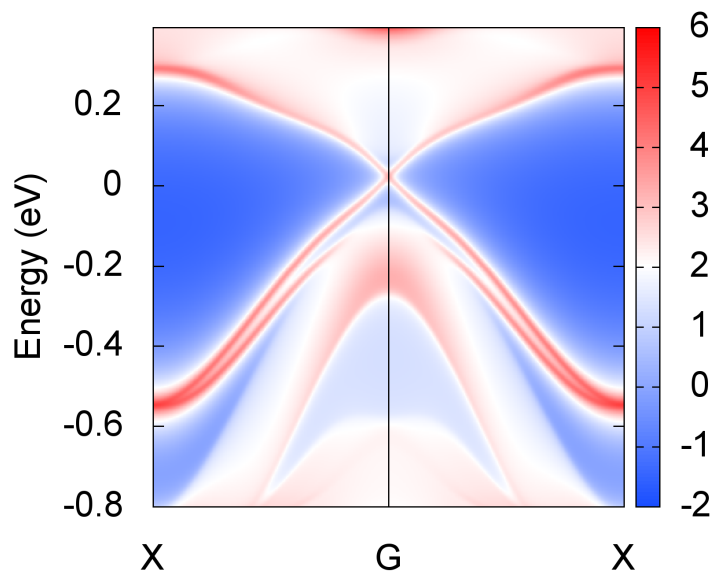


(a) $k_1=0.0$, $z_2=0$; (b) $k_1=0.5$, $z_2=0$; (c) $k_2=0.0$, $z_2=0$; (d) $k_2=0.5$, $z_2=0$; (e) $k_3=0.0$, $z_2=1$; (f) $k_3=0.5$, $z_2=1$;

So the bulk Z2 topological number is (0, 001), which means a weak topological insulator in 3-dimension picture, while a strong TI in 2D picture. For a 2D material, you only need the topological index in figure (e).

3.2.3 Edge state

The edge states of a 2D material is the side surface of a 3D model. The calculated edge states of MoS2 on (100) surface is



The settings in **WT.in** are

```
&CONTROL
SlabSS_calc           = T
/

&SYSTEM
SOC = 1                ! soc
NumOccupied = 36       ! Number of occupied Wannier bands
E_FERMI = -3.9151      ! e-fermi
/

&PARAMETERS
Eta_Arc = 0.001        ! infinite small value, like brodening
E_arc = -0.05          ! energy for calculate Fermi Arc
OmegaMin = -0.8        ! energy interval
OmegaMax = 0.4         ! energy interval
OmegaNum = 401         ! omega number
Nk1 = 101              ! number k points  odd number would be better
Nk2 = 101              ! number k points  odd number would be better
Np = 2                 ! number k points  odd number would be better
/

MILLER_INDICES
1 0 0

!SURFACE                ! See doc for details
0 1 0
0 0 1
1 0 0

KPATH_SLAB
```

(continues on next page)

(continued from previous page)

```

2          ! number of k line for 2D case
X  0.5  0.0 G  0.0  0.0  ! k path for 2D case
G  0.0  0.0 X  0.5  0.0

KPLANE_SLAB
-0.5 -0.5      ! Original point for 2D k plane
 1.0  0.0      ! The first vector to define 2D k plane
 0.0  1.0      ! The second vector to define 2D k plane   for arc plots

```

3.2.4 Full settings in WT.in of 1Tp-MoS2

```

&TB_FILE
Hrfile = 'wannier90_hr.dat'
Package = 'VASP'
/

!> bulk band structure calculation flag
&CONTROL
BulkBand_calc      = T
BulkFS_calc        = F
Z2_3D_calc         = T
DOS_calc           = F
BulkFS_plane_calc  = F
BulkGap_cube_calc  = F
BulkGap_plane_calc = F
SlabBand_calc      = F
WireBand_calc      = F
SlabSS_calc        = T
SlabArc_calc       = T
SlabSpintexture_calc = F
wanniercenter_calc = F
BerryPhase_calc    = F
BerryCurvature_calc = F
/

&SYSTEM
NSLAB = 20
NumOccupied = 36      ! NumOccupied
SOC = 1              ! soc
E_FERMI = -3.9151     ! e-fermi
surf_onsite= 0.0      ! surf_onsite
/

&PARAMETERS
Eta_Arc = 0.001      ! infinite small value, like brodening
E_arc = -0.05         ! energy for calculate Fermi Arc
OmegaNum = 401        ! omega number
OmegaMin = -0.8       ! energy interval
OmegaMax = 0.4        ! energy interval
Nk1 = 101             ! number k points
Nk2 = 101             ! number k points
Nk3 = 3               ! number k points
NP = 2                ! number of principle layers
Gap_threshold = 0.10 ! threshold for GapCube output

```

(continues on next page)

(continued from previous page)

```

/

LATTICE
Angstrom
  3.1770280139589571    0.0000000000000000    0.0000000000000000
  0.0000000000000000    5.7281689431742455   -0.0157118470068534
  0.0000000000000000    0.3756910330601151    30.2151819975086902

ATOM_POSITIONS
6                      ! number of atoms for projectors
Direct                ! Direct or Cartisen coordinate
  Mo 0.5000000049999969 0.4188543132266640 0.5873016275251595
  Mo -0.0000000000000000 0.0233088607917941 0.5925355564141651
  S 0.5000000049999969 0.1358785901303858 0.6473858970202739
  S 0.0000000000000000 0.6405161490437790 0.6337728762151337
  S -0.0000000000000000 0.3062864083475208 0.5324524179435783
  S 0.5000000049999969 0.8016452854598629 0.5460646648816858

PROJECTORS
6 6 4 4 4 4          ! number of projectors
Mo s dxy dyz dxz dx2-y2 dz2 ! projectors
Mo s dxy dyz dxz dx2-y2 dz2 ! projectors
S s px py pz
S s px py pz
S s px py pz
S s px py pz

MILLER_INDEX          ! this is equal to the SURFACE card
1 0 0

!SURFACE              ! MoS2 conventional (010) surface
0 1 0
0 0 1
1 0 0

KPATH_BULK            ! k point path
4                      ! number of k line only for bulk band
  X 0.50000 0.00000 0.00000 G 0.00000 0.00000 0.00000
  G 0.00000 0.00000 0.00000 Y 0.00000 0.50000 0.00000
  Y 0.00000 0.50000 0.00000 M 0.50000 0.50000 0.00000
  M 0.50000 0.50000 0.00000 G 0.00000 0.00000 0.00000

KPATH_SLAB
2                      ! number of k line for 2D case
X 0.5 0.0 G 0.0 0.0 ! k path for 2D case
G 0.0 0.0 X 0.5 0.0

KPLANE_SLAB
-0.5 -0.5            ! Original point for 2D k plane
1.0 0.0              ! The first vector to define 2D k plane
0.0 1.0              ! The second vector to define 2D k plane for arc plots

KPLANE_BULK
-0.50 -0.50 0.00     ! Original point for 3D k plane
1.00 0.00 0.00       ! The first vector to define 3d k space plane
0.00 1.00 0.00       ! The second vector to define 3d k space plane

```

(continues on next page)

(continued from previous page)

KCUBE_BULK

```

-0.50 -0.50 -0.50  ! Original point for 3D k plane
 1.00  0.00  0.00  ! The first vector to define 3d k space plane
 0.00  1.00  0.00  ! The second vector to define 3d k space plane
 0.00  0.00  1.00  ! The third vector to define 3d k cube

```

WANNIER_CENTRES

Cartesian

```

 1.582112  2.389848 17.807782
 1.588504  2.609305 17.734627
 1.588512  2.615066 17.729182
 1.588518  2.627199 17.745843
 1.588498  2.598550 17.729411
 1.588498  2.629729 17.732881
-0.004616  0.781697 17.681791
-0.000012  0.362382 17.901360
 0.000001  0.357923 17.918636
-0.000009  0.350028 17.898957
-0.000005  0.365927 17.908526
-0.000007  0.366124 17.901329
 1.588504  1.034466 19.413821
 1.588523  1.032565 19.619308
 1.588472  1.012794 19.497202
 1.588484  1.034341 19.536688
-0.000085  3.898045 19.004023
-0.000043  3.932819 19.131844
-0.000018  3.905334 19.041137
 0.000028  3.922253 19.122079
-0.000005  1.945318 16.239885
-0.000014  1.949702 16.018513
-0.000042  1.987244 16.143671
-0.000004  1.954899 16.130450
 1.588449  4.785448 16.618078
 1.588486  4.728185 16.486681
 1.588469  4.865486 16.580409
 1.588453  4.810723 16.489254
 1.595220  2.389977 17.807534
 1.588525  2.609305 17.734627
 1.588517  2.615065 17.729183
 1.588510  2.627201 17.745844
 1.588530  2.598543 17.729408
 1.588531  2.629735 17.732878
 0.004808  0.781415 17.681511
 0.000013  0.362387 17.901363
-0.000001  0.357918 17.918638
 0.000009  0.350028 17.898959
 0.000005  0.365927 17.908525
 0.000007  0.366134 17.901324
 1.588524  1.034460 19.413819
 1.588506  1.032552 19.619299
 1.588559  1.012789 19.497215
 1.588545  1.034346 19.536661
 0.000087  3.898051 19.004022
 0.000044  3.932824 19.131846
 0.000018  3.905353 19.041145
-0.000027  3.922253 19.122048

```

(continues on next page)

(continued from previous page)

```

0.000005  1.945317  16.239895
0.000014  1.949708  16.018506
0.000043  1.987240  16.143659
0.000006  1.954899  16.130433
1.588582  4.785446  16.618079
1.588543  4.728170  16.486695
1.588562  4.865497  16.580411
1.588575  4.810739  16.489208

```

3.3 WTe2 (Type II Weyl semimetal)

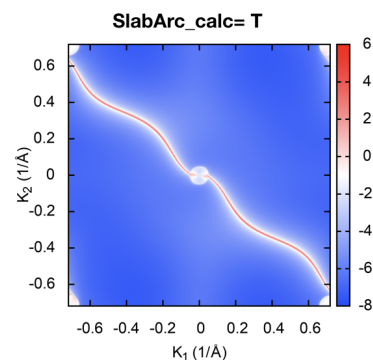
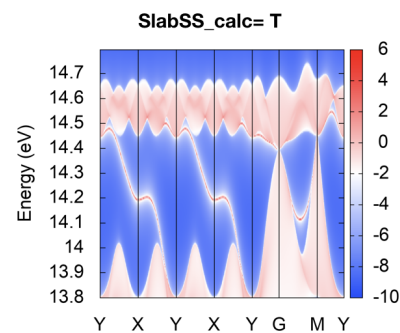
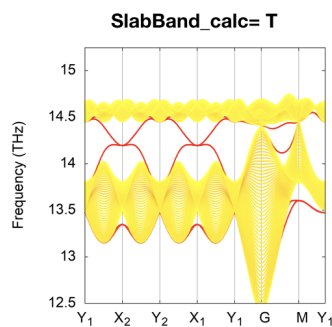
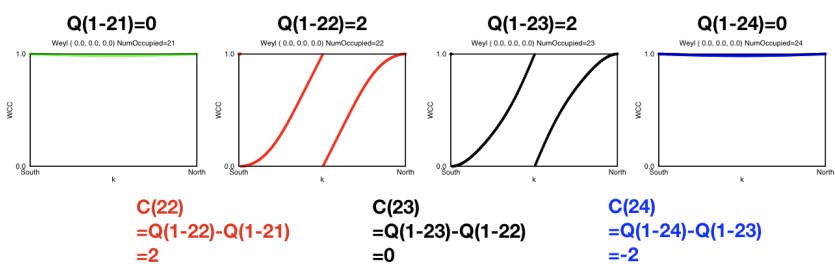
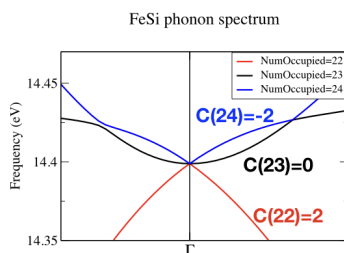
to be continue

3.4 IrF4 (Nodal Chain metals)

to be continue

3.5 FeSi (Weyl point in Phonon system)

FeSi phonon surface states spectrum



This example is from T.T Zhang

QSWU June 27 2018

CHAPTER 4

Citation

Please cite this paper when using WannierTools for your researchs:

```
@article{WU2017,  
title = "WannierTools : An open-source software package for novel topological_  
↪materials",  
journal = "Computer Physics Communications",  
volume = "224",  
pages = "405 - 416",  
year = "2018",  
issn = "0010-4655",  
doi = "https://doi.org/10.1016/j.cpc.2017.09.033",  
url = "http://www.sciencedirect.com/science/article/pii/S0010465517303442",  
author = "QuanSheng Wu and ShengNan Zhang and Hai-Feng Song and Matthias Troyer and_  
↪Alexey A. Soluyanov",  
keywords = "Novel topological materials, Topological number, Surface state, Tight-  
↪binding model"  
}
```


CHAPTER 5

Correspondence

Please report bugs to [wuquansheng at gmail.com](mailto:wuquansheng@gmail.com).

CHAPTER 6

Licence

The WannierTools code is licensed under GPLv3. The licence text in the LICENSE file is included in the root directory of the WannierTools distribution.


Authors and contributions

WannierTools 2.x have been written by :

- QuanSheng Wu (EPFL, Switzerland)
- ShengNan Zhang (EPFL, Switzerland)

Contributors to the code include:

- Changming Yue (IOP, Beijing, China): symmetrization
- Yifei Guan (EPFL, Switzerland): Landau level
- Yi Liu (BNU, Beijing, China): Runge-Kutta integration

 wannier_tools-logo-crop.png

CHAPTER 8

Sponsors

This work was sponsored by the following institutes:



ETH zürich

