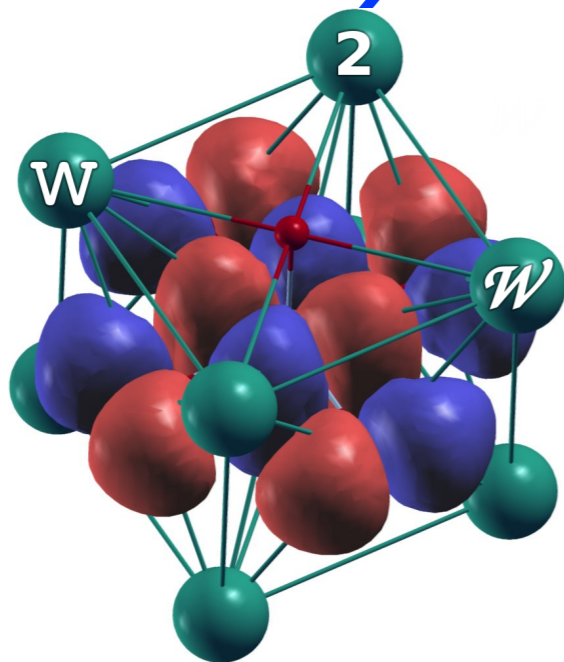


**Wannier90 & Wien2Wannier Tutorial**



**WANNIER90**

## STEP.1 :

1. Run LDA calculation for NiO with Wien2K and get band-structure data (from "task").

Use lattice constant : 7.899055 Bohr & Ni (0.0,0.0,0.0) , O(0.5,0.5,0.5)

2. Plot the band-structure & 'make a plan' (=select states to be wannierized)

3. Prepare "case.fermi" file with Fermi energy  $E_F$  on the first line.

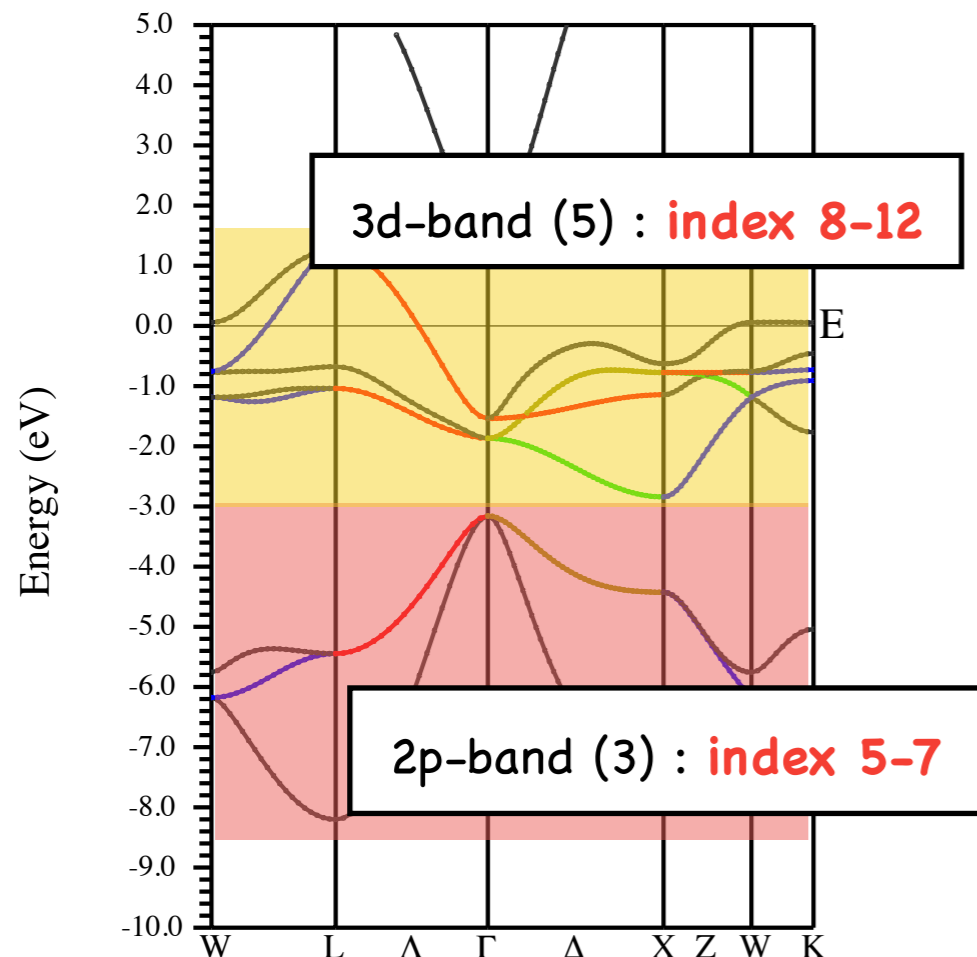
(Use "grep :FER case.scf" command to find  $E_F$  in Ry unit)

4. Make subdir and K-mesh in the whole BZ for wannierization

(Use "prepare\_w2wdir subdir" and "x kgen -fbz" command with no-shift ("0" ) )

## STEP.2 : Write "case.inwf" file

Find the number of target bands (to be wannierized, d-only or d-p ?) and the indices of these bands defined in Wien2K calculation.



Find band index using ...

"case.output1" file

or

"x findbands -emin \*\* -emax \*\* -efermi \*\*" command  
(note : emin and emax in eV, efermi in Ry unit )  
which writes the band index in the energy window to  
"case.outputfind" file

### Example of "case.outputfind"

Bloch bands in the interval			
at all k:	5	12	8
at any k:	5	13	9

# Example of "case.inwf" file (dp-model)

The number of terms used to approximate  $\exp(-ikb)$  (use 3~4)

Min and max of band index

The number of Wannier functions

```

BOTH # AMN, MMN, or BOTH
5 12 # min band, max band
3 8 # LJMAX in exp(ibr) expansion, #WF
2 # 1:d-eg -> 1:dx2-y2
2 2 -2 0.70710678 0.00000000 # iat, l, m, Re(coeff), Im(coeff)
2 2 2 0.70710678 0.00000000
1 # 1:d-eg -> 1:d3z2-r2
2 2 0 1.00000000 0.00000000
2 # 1:d-t2g -> 1:dxy
2 2 -2 0.00000000 0.70710678
2 2 2 0.00000000 -0.70710678
2 # 1:d-t2g -> 1:dxz
2 2 -1 0.70710678 0.00000000
2 2 1 -0.70710678 0.00000000
2 # 1:d-t2g -> 1:dyz
2 2 -1 0.00000000 0.70710678
2 2 1 0.00000000 0.70710678
2 # 5:p -> 5:px
1 1 -1 0.70710678 0.00000000
1 1 1 -0.70710678 0.00000000
2 # 5:p -> 5:py
1 1 -1 0.00000000 0.70710678
1 1 1 0.00000000 0.70710678
1 # 5:p -> 5:pz
1 1 0 1.00000000 0.00000000
    
```

Atom index

$$\begin{aligned}
 |d_{x^2-y^2}\rangle &= \frac{1}{\sqrt{2}}(|d_2\rangle + |d_{-2}\rangle) \\
 |d_{3z^2-r^2}\rangle &= |d_0\rangle \\
 |d_{xy}\rangle &= \frac{-i}{\sqrt{2}}(|d_2\rangle - |d_{-2}\rangle) \\
 |d_{yz}\rangle &= \frac{i}{\sqrt{2}}(|d_1\rangle + |d_{-1}\rangle) \\
 |d_{zx}\rangle &= \frac{-1}{\sqrt{2}}(|d_1\rangle - |d_{-1}\rangle) \\
 |p_x\rangle &= \frac{-1}{\sqrt{2}}(|p_1\rangle - |p_{-1}\rangle) \\
 |p_y\rangle &= \frac{1}{\sqrt{2}}(|p_1\rangle + |p_{-1}\rangle) \\
 |p_z\rangle &= |p_0\rangle
 \end{aligned}$$

E<sub>g</sub>

T<sub>2g</sub>

You can use "init\_w2w" to check the atom index

The number of expansion coefficients

## STEP.3 : Run interface

1. Command `"write_win"` to make "case.win" using "case.inwf" file  
(“case.win” file is input file for wannier90)
2. Command `"x wannier90 -pp"`  
(write a list of nn k-points to "case.nnkp" on the basis on "case.win")
3. Run `"x lapw1"`  
(compute eigen states and vectors for the new k-mesh prepared in STEP.1)
4. Command `"x w2w"` → Output : "case.mmn" and "case.amn"  
(compute overlaps  $M_{mn}$  and initial projections  $A_{mn}$ )

### Example of "case.win" file

```
num_bands      = 8
num_wann       = 8
```

```
!!! Disentanglement parameters !!!
!dis_froz_min  = 7.
!dis_froz_max  = 9.
!dis_mix_ratio = 0.5
```

```
!!! Iterations &c. !!!
```

```
iprint          = 1
num_iter        = 10000
num_print_cycles = 100
conv_window     = 3
!conv_tol       = 1e-10
dis_num_iter    = 10000
!dis_conv_window = 3
!dis_conv_tol   = 1e-10
!restart        = default | wannierise | plot | transport
```

Option for disentanglement

(num\_bands > num\_wann case)

dis\_froz\_min : bottom of the frozen energy window

dis\_froz\_max : top of the frozen energy window

dis\_mix\_ratio : mixing ratio during minimization of  $\Omega_I$

Maximum number of iterations for minimization

"Restart" option

Wannierise : Restart from the beginning of the wannierization

Plot : Go directly to plotting phase

(Use this "restart" option in Q.1 to save time)

```

!!! Post-processing options !!!
write_proj           = .true.
write_xyz            = .true.
translate_home_cell = .true.
hr_plot             = .true.
!fermi_surface_plot = .true.

```

“dist\_cutoff” option :

put “bands\_plot\_mode = cut” and  
enter the largest distance between WFs for which  
the Hamiltonian matrix element is retained.

(use Å unit)

```

!!! Band structure !!!
!!! needs `kpoint_path' block
!bands_plot           = .true.
bands_num_points     = 50
!bands_plot_format   = gnuplot xmgrace
!bands_plot_project  = 1-3
!bands_plot_mode     = s-k | cut [Slater-Koster | truncate Hamiltonian]
!dist_cutoff         = 1.0

```

```
bands_plot = .true.
```

```
begin kpoint_path
```

R	0.50	0.50	0.50	L	0.28	0.28	0.28
L	0.28	0.28	0.28	G	0.00	0.00	0.00
G	0.00	0.00	0.00	D	0.00	0.12	0.12
D	0.00	0.12	0.12	X	0.00	0.25	0.25
X	0.00	0.25	0.25	Z	0.12	0.25	0.38
Z	0.12	0.25	0.38	M	0.25	0.25	0.50
M	0.25	0.25	0.50	S	0.12	0.12	0.25
S	0.12	0.12	0.25	G	0.00	0.00	0.00

```
end kpoint_path
```

## SETP.4 : Run Wannierization

1. Command “**x wannier90**” to make “case.win” on the basis on “case.inwf”  
 (“case.win” file is input file for wannier90)
2. Check the result of wannierization written to “case.wout” file  
(especially, convergence of the spread, which is (usually) smaller than the unit cell)

**Wannierization Fin !!**

## SETP.5 : Analysis and Visualization

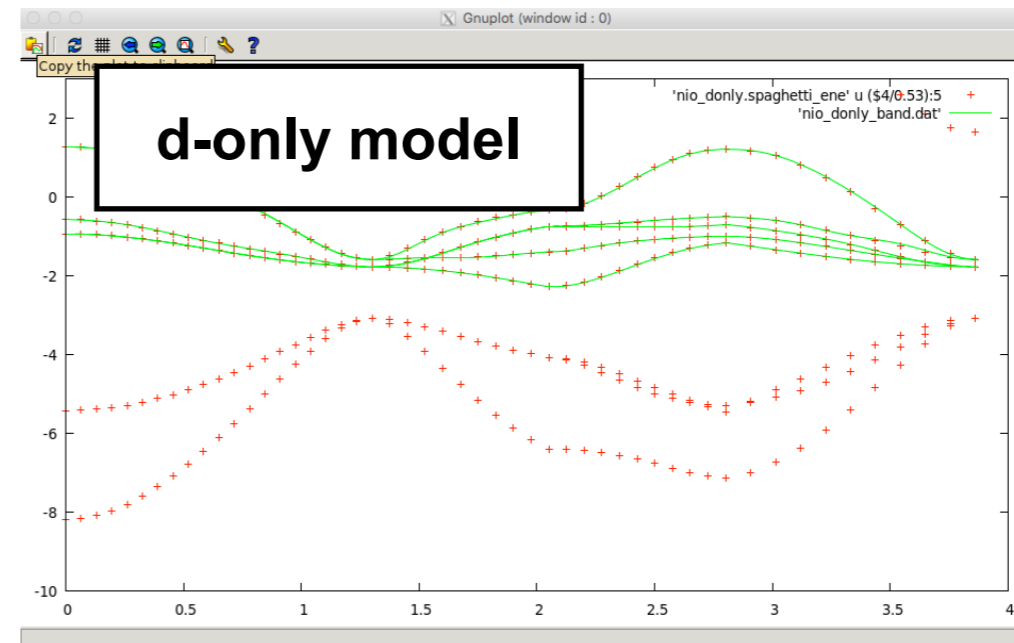
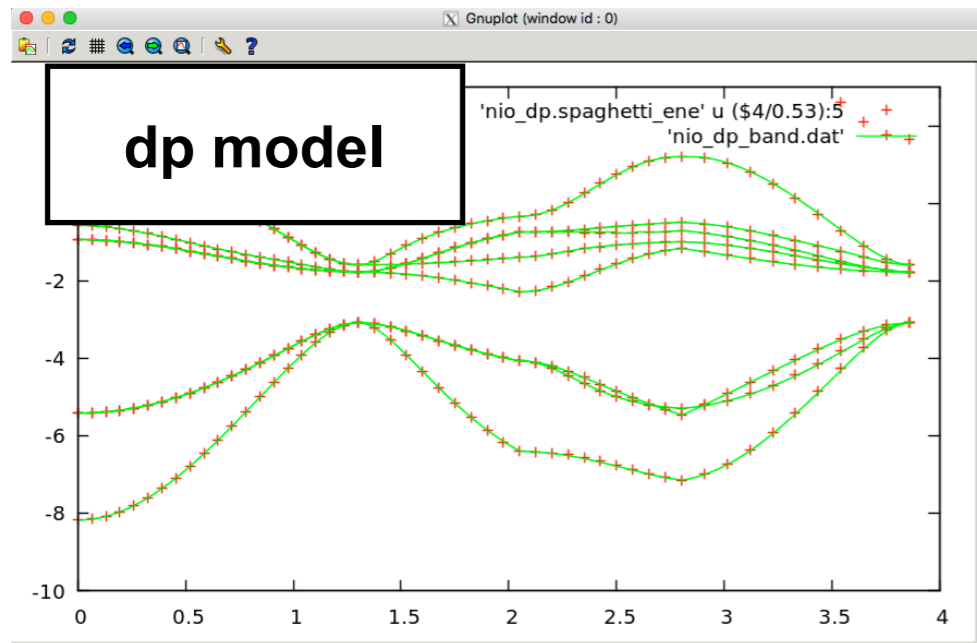
### 1. Compare band-structures

Wannier90 writes a band structure derived from the Wannier-interpolated  $H(k)$  to "case\_band.dat". One can compare it to the one computed in STEP.1 (Wien2K) ("case.spaghetti" file).

**GNU PLOT COMAND** : `p 'case.spaghetti_ene' u ($4/0.53):5, 'case_band.dat' w l`

for conversion from Bohr<sup>-1</sup> to Å<sup>-1</sup> unit

$$1 \text{ Bohr} = 0.5291772083 \text{ \AA}$$



### Question.1

Find the diameters of nn, nnn, ... shells of Ni from "case.outputnn".

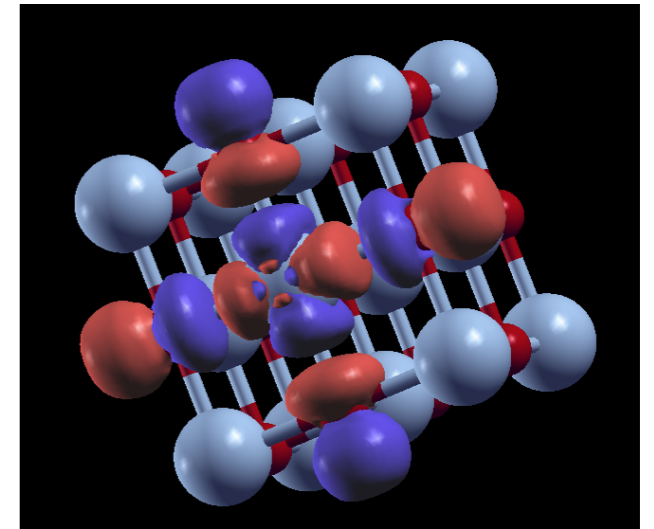
Then, use wannier90 to calculate band-structures with no hopping, nn, nnn ... hopping and compare them with the original band structure.

(use the "dist\_cutoff" parameter and "restart" option to save time.)

## 2. Plot Wannier Functions

1. write `case.inwplot` as follows (see HINTS for details of this input file),

```
3D ORTHO
-1 -1 -1 1
 1 -1 -1 1
-1  1 -1 1
-1 -1  1 1
 40 40 40 0 0 0      # grid points and echo increments
NO
WAN ANG LARGE
1 2 ← "m"-th WF you want to plot
```



2. Command `x wplot -wf -m` (it may take a few minutes depending on your grid points)  
(evaluate m-th WF on the real-space grid and write density to `case_m.psink` )
3. Run `wplot2xsf`  
(converts files to `xsf` files which can be opened by XCrySDen.)
4. Command `xcrysdem --xsf case_m.xsf`  
: Pick `Tools→Date Grid` from the menu and press OK  
(set `isovalue` to 2 and check the `Render +/- isovalue` box )

### Question.2

Plot `xy` and `3z2-r2` wannier functions for d-only and dp models

### Question.3

Find out  $H(R=(0,0,1))$  part in "case\_hr.dat" file and which n-shell it corresponds to.

(use "XCrySDen" and "Modify → # of unit cells drawn" to find out. )

Then, analyze the symmetry of  $H(R=(0,0,1))$  and  $H(R=(0,0,0))$ .

(ex. find out the reason for ZERO values from the viewpoint of symmetry)

### Structure of "case\_hr.dat" file

This file contains the hopping integrals  $\langle m, R | H | n, 0 \rangle$  between the n-th wannier function  $|n, 0\rangle$  in the home unit cell and the m-th wannier function  $|m, R\rangle$  in the unit cell at R.

-6	0	3	1	1	0.000040	0.000000
-6	0	3	2	1	0.000000	-0.000000
-6	0	3	3	1	-0.000000	0.000000
-6	0	3	4	1	0.000047	-0.000000
-6	0	3	5	1	-0.000047	-0.000000
-6	0	3	1	2	0.000000	-0.000000
-6	0	3	2	2	0.000041	-0.000000

$(R_1, R_2, R_3)$

$(m, n)$

Real( $T_{mn}$ )

Im( $T_{mn}$ )

orbital label

where R is given as  $\vec{R} = R_1\vec{a}_1 + R_2\vec{a}_2 + R_3\vec{a}_3$

### Question.4

What is the largest nn and nnn hopping element  $T_{\alpha\beta}$  (values and orbitals) ?



# HINTS

**2.9.12** `character(len=20) :: bands_plot_mode`

To interpolate the band structure along the k-point path, either use the Slater-Koster interpolation scheme or truncate the Hamiltonian matrix in the WF basis. Truncation criteria are provided by `hr_cutoff` and `dist_cutoff`.

The valid options for this parameter are:

- `s-k` (default)
- `cut`

**2.9.42** `real(kind=dp) :: hr_cutoff`

The absolute value of the smallest matrix element of the Hamiltonian in the WF basis. If  $h_{mn}(\mathbf{R}) > \text{hr\_cutoff}$ , then the matrix element  $h_{mn}(\mathbf{R})$  is retained and used in the band structure interpolation (when `bands_plot_mode = cut`) or in the transport calculation. Otherwise it is deemed to be insignificant and is discarded. Units are eV.

The default value is 0.0.

**2.9.43** `real(kind=dp) :: dist_cutoff`

The largest distance between two WFs for which the Hamiltonian matrix element is retained and used in the band interpolation (when `bands_plot_mode = cut`) or in the transport calculation. Units are Å.

The default value is 1000.0.

### 2.9.11 integer :: bands\_plot\_project(:)

If present `wannier90` will compute the contribution of this set of WF to the states at each point of the interpolated band structure. The WF are numbered according to the `seedname.wout` file. The result is written in the `seedname_band.dat` file, and a corresponding gnuplot script to `seedname_band_proj.dat`.

For example, to project on to WFs 2, 6, 7, 8 and 12:

```
bands_plot_project : 2, 6-8, 12
```

### 2.6.7 character(len=20) :: restart

If `restart` is present the code will attempt to restart the calculation from the `seedname.chk` file. The value of the parameter determines the position of the restart

The valid options for this parameter are:

- `default`. Restart from the point at which the check file `seedname.chk` was written
- `wannierise`. Restart from the beginning of the `wannierise` routine
- `plot`. Go directly to the plotting phase
- `transport`. Go directly to the transport routines

# case.inwplot

```
----- top of file: case.inwplot -----
3D ORTHO      # mode O(RTHOGONAL)|N(ON-ORTHOGONAL)
-1 -1 -1 1    #x, y, z, divisor of orig
 0 -1 -1 1    #x, y, z, divisor of x-end
-1  0 -1 1    #x, y, z, divisor of y-end
-1 -1  0 1    #x, y, z, divisor of z-end
 20 20 20 0 0 # grid points and echo increments
NO           # DEP(HASING)|NO (POST-PROCESSING)
WAN ANG LARGE # switch ANG|ATU|AU LARGE|SMALL
1  1         # k-point, band index
----- bottom of file -----
```

Interpretive comments on this file are as follows:

**line 1:** A3, A1

mode flag

MODE	3D	a three-dimensional grid will be specified
	$nD$	with $n = 0, 1, 2$ exist in <code>lapw7</code> , but are untested with <code>wplot</code>
	ANY	read arbitrary grid from <code>case.grid</code> (untested)
flag	O	grid axes will be checked for mutual orthogonality
	N	axes may be non-orthogonal

**line 2:** free format

$ix, iy, iz, idv$       Coordinates of the origin of the grid, where  $x=ix/idv$  etc. in units of the *conventional* lattice vectors

**line 3:** free format

$ix, iy, iz, idv$       Coordinates of the end points of each grid axis

>>> **line 3** must be given for each direction (i.e.,  $n$  times in total for an  $nD$  grid).

**line 6:** free format  $nx, ny, nz; N N N$

$nx, ny, nz$       number of mesh points in each direction; the additional input on this line is unused

**With Spin-orbit interaction**

## STEP.1 :

1. Run LDA calculation (for SrIrO<sub>3</sub> with ideal structure) with Wien2K **WITH spin-orbit interaction**

**case.inso** : input file for spin-orbit interaction

```
WFFIL
4 0 0          lmax,ipr,kpot
-10 2.0       Emin, Emax
  0 0 1          h,k,l (direction of magnetization)
0             number of atoms with RLO
0 0 0 0 0     number of atoms without SO, atomnumbers
```

**run self consistent cycle with -so option**

The screenshot shows the Wien2K GUI with the 'Options' panel on the left and 'Expert options', 'Scratch Directory', and 'Convergence criteria' panels on the right. The 'spinorbit' checkbox in the 'Options' panel is checked and circled in blue. The 'Convergence criteria' panel shows 'Energy' and 'Charge' checked with values of 0.0001 Ry and 0.0001 e, respectively. The 'Type of execution' is set to 'background' and 'E-mail notification' is disabled. At the bottom, there are buttons for 'start SCF cycle', 'Clear entries', and a checkbox for 'only save parameters'.

**Options: (  help)**

- parallel
- optimize positions (MSR1a)
- iterative diag
- iter. after full-diag
- iter.diag (no Hinv)
- vec2pratt with iter.diag
- spinorbit
- spin polarized
- constrain moment to 0
- AFM calc.
- dm
- orbital pot (DFT+U)
- eece (hybrid-DFT for correlated e)
- DFT-D3 (dispersion corrections)
  - hf (full hybrid-DFT, expensive!)
  - diaghf (diagonal-only full hybrid-DFT)
  - non-scf full hybrid-DFT)
  - newklist (full hybrid-DFT with new k-list)
  - redklist (full hybrid-DFT with red. k-list)

no HNS 6

in1new 2

q-limit 0.05

lt-number 4

FSM 0

**Expert options:**

**Scratch Directory:**

**Convergence criteria:**

- Energy: 0.0001 Ry
- Force: 1 mRy/au
- Charge: 0.0001 e

Type of execution: background

E-mail notification  to

start SCF cycle Clear entries  only save parameters

2. Prepare "case.fermiup" & "case.fermidn" file with Fermi energy  $E_F$  on the first line.

(Use "grep :FER case.scf" command to find  $E_F$  in Ry unit)

Note : x findbands -so ... is invoked in the case with spin-orbit interaction

3. Prepare subdir and K-mesh in the whole BZ which is used in wannierization

(Use "prepare\_w2wdir subdir" and "x kgen -fbz" command with no-shift ("0" ) )

## STEP.2 : Write "case.inwfup", "case.inwfdn" and "case.inwf" file

Decide the number of bands to be wannierized (d-only or dip ?) and find the index of these bands defined in Wien2K.

- "case.inwup (dn)" contains initial projections for up-spin channel
- "case.inwf" should contain all projections

### Example of "case.inwfup" file

```
BOTH      # AMN, MMN, or BOTH
35 62     # min band, max band
3 28      # LJMAX in exp(ibr) expansion, #WF
2         # 1:d-eg -> 1:dx2-y2
5 2 -2    0.70710678  0.00000000
5 2 2     0.70710678  0.00000000
1         # 1:d-eg -> 1:d3z2-r2
5 2 0     1.00000000  0.00000000
2         # 1:d-t2g -> 1:dxy
5 2 -2    0.00000000  0.70710678
5 2 2     0.00000000 -0.70710678
2         # 1:d-t2g -> 1:dxz
5 2 -1    0.70710678  0.00000000
5 2 1     -0.70710678 0.00000000
2         # 1:d-t2g -> 1:dyz
5 2 -1    0.00000000  0.70710678
5 2 1     0.00000000  0.70710678
0
```

### Example of "case.inwfdn" file

```
BOTH      # AMN, MMN, or BOTH
35 62     # min band, max band
3 28      # LJMAX in exp(ibr) expansion, #WF
0
0
0
0
0
0
2         # 1:d-eg -> 1:dx2-y2
5 2 -2    0.70710678  0.00000000
5 2 2     0.70710678  0.00000000
1         # 1:d-eg -> 1:d3z2-r2
5 2 0     1.00000000  0.00000000
2         # 1:d-t2g -> 1:dxy
5 2 -2    0.00000000  0.70710678
5 2 2     0.00000000 -0.70710678
2         # 1:d-t2g -> 1:dxz
5 2 -1    0.70710678  0.00000000
```

### STEP.3 : Run interface

1. Command `"write_win"` to make `"case.win"` on the basis on `"case.inwf"`  
(`"case.win"` file is input file for wannier90)
2. Command `"x wannier90 -pp"`  
(write a list of nn k-points to `"case.nnkp"` on the basis on `"case.win"`)
3. Run `"x lapw1 && x lapwso"`  
(compute eigen states and vectors for the new k-mesh prepared in STEP.1)
4. Command ~~`"x w2w -so -up && x w2w -so -dn"`~~  
Command `"w2wc upw2w.def && w2wc dnw2w.def"`  
(`case.inwfup` & `case.inwfdn` should be given in the def files)  
→ Output : `"case.mmnup (dn)"` and `"case.amnup (dn)"`
5. Command `"x w2waddsp"`

### STEP.4 : Run Wannierization

1. Command `"x wannier90"` to make `"case.win"` on the basis on `"case.inwf"`  
(`"case.win"` file is input file for wannier90)
2. Check the result of wannierization written to `"case.wout"` file  
(especially, convergence of the spread, which is (usually) smaller than the unit cell)

## Reference

1. “Wien2wannier: From linearized augmented plane waves to maximally localized Wannier functions”, Jan Kuneš, Ryotaro Arita, Philipp Wissgott, Alessandro Toschi, Hiroaki Ikeda, Karsten Held  
<https://arxiv.org/abs/1004.3934>
2. “WIEN2WANNIER USER’S GUIDE”  
Jan Kuneš, Philipp Wissgott and Elias Assmann  
[http://www.wien2k.at/reg\\_user/textbooks/wien2wannier\\_userguide.pdf](http://www.wien2k.at/reg_user/textbooks/wien2wannier_userguide.pdf)
3. “wannier90: User Guide (version 2.0.1)”  
[http://wannier.org/doc/user\\_guide.pdf](http://wannier.org/doc/user_guide.pdf)



# How to construct rotation matrix in distorted cases ??

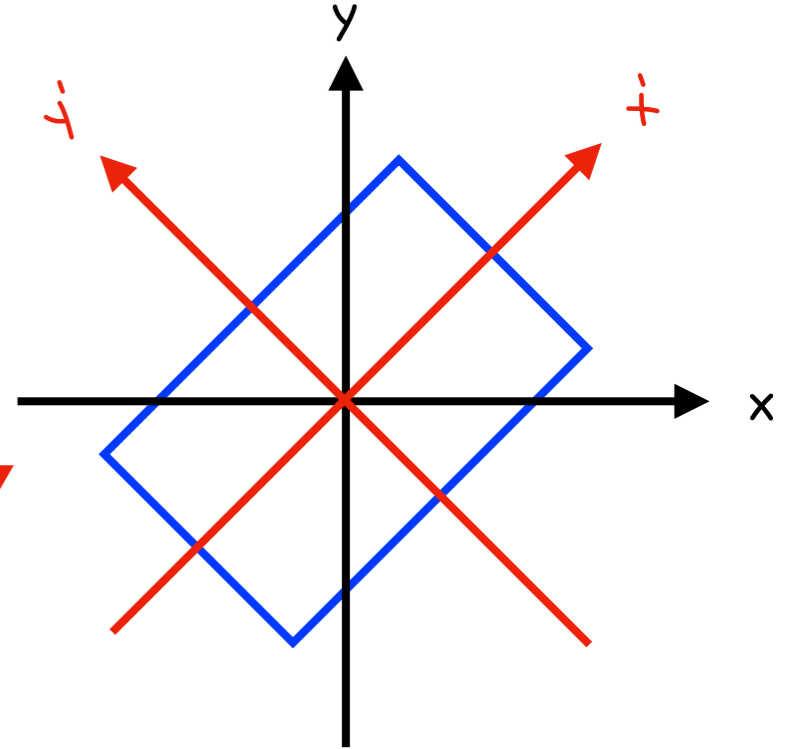
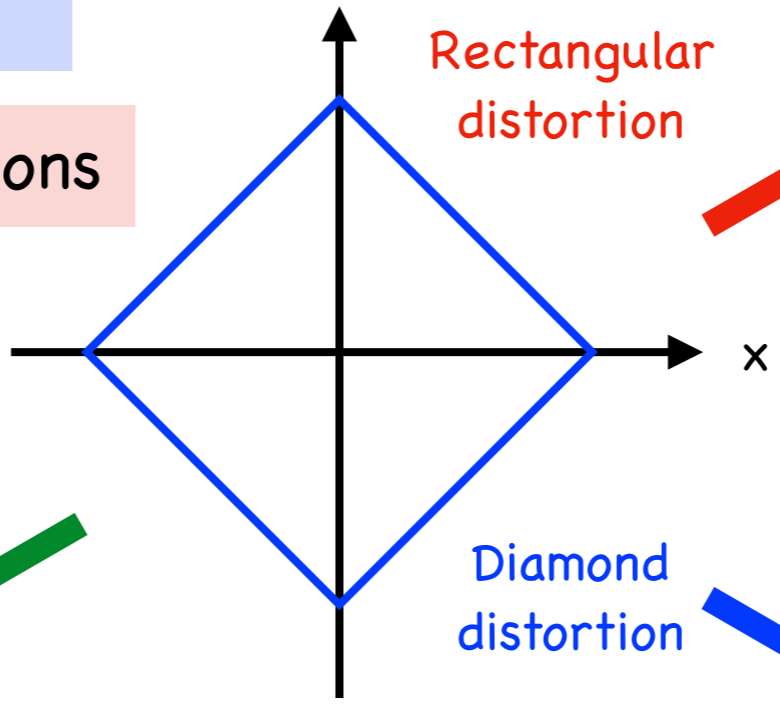
D2h symmetry

Off diag.	_____	b1g (xy)
	_____	a1g (3z <sup>2</sup> -r <sup>2</sup> )
	_____	a1g (x <sup>2</sup> -y <sup>2</sup> )
	_____	b2g (xz)
	_____	b3g (yz)

Rectangular case :  
local axis should be rotated by 180°  
not to create unnecessary off-diagonal CF

## Basis Distortions

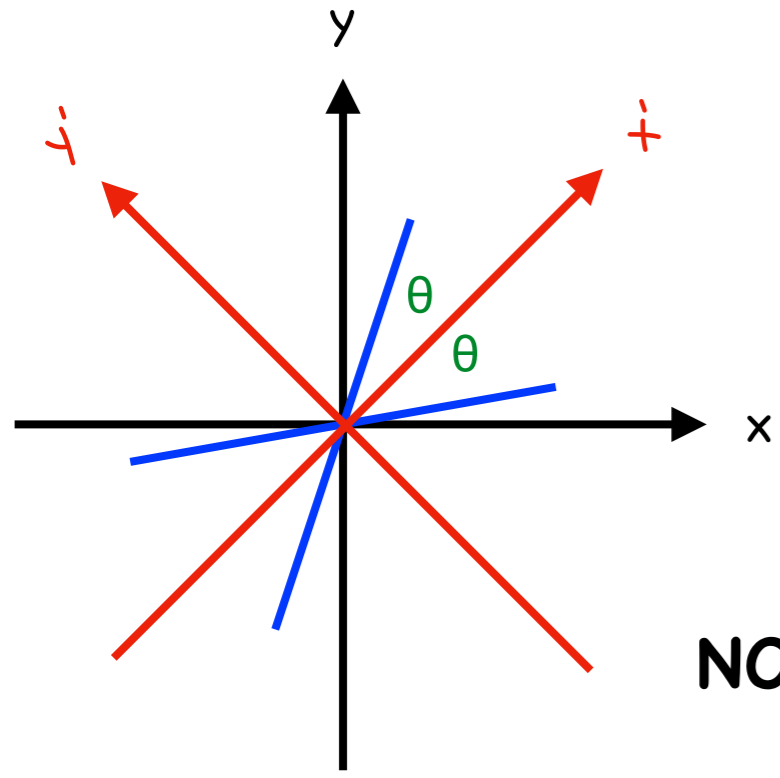
Oh (or D4h)



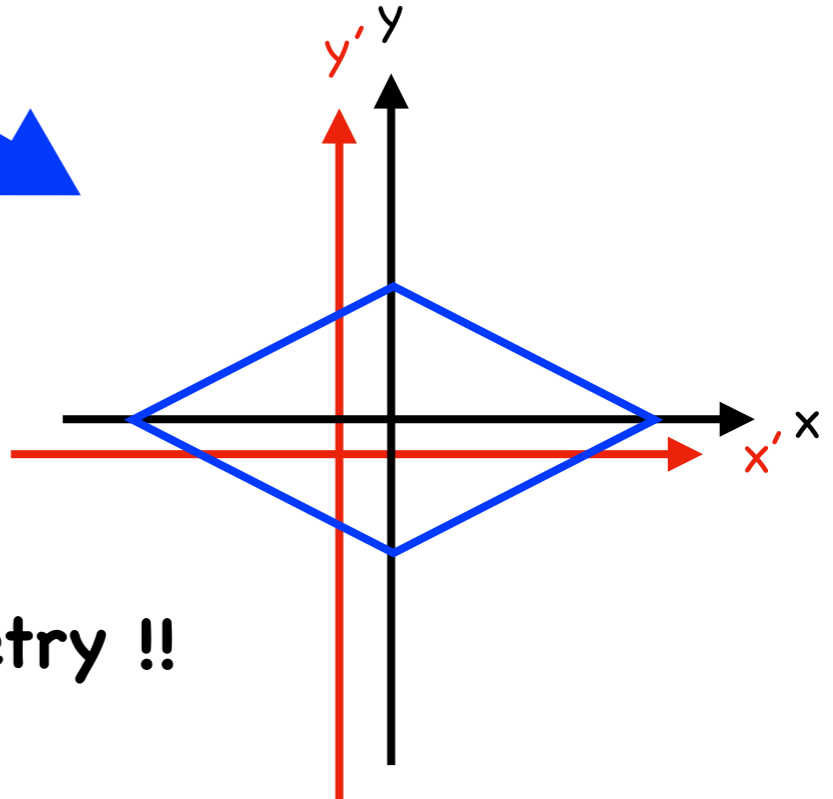
Diamond case :  
one don't have to rotate local axis.

Rectangular distortion =

Rotation (M-O length unchanged)



(Oxygens sit on edges)



**NOTE : They are all D2h symmetry !!**