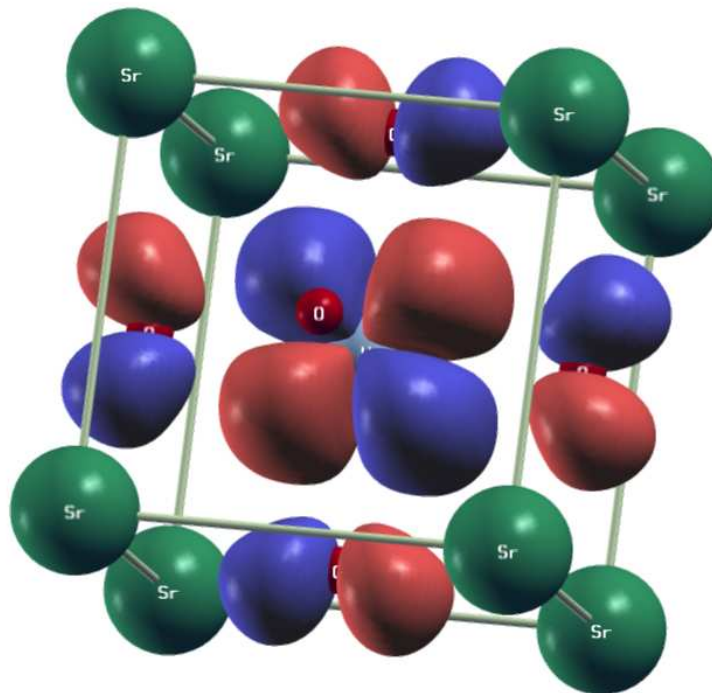


userguide WIEN2WANNIER: From linearized augmented plane waves to maximally localized Wannier functions

Jan Kuneš, Philipp Wissgott

June 20, 2011



1.1 Introduction

In the tool kit for theoretical solid state investigations, algorithms based on the density functional theory (LDA) represent the backbone applications. One of the very popular code packages available is the wien2k program [1, 2]. It is based on the segmentation of the real space in muffin-spheres (around the ionic cores) and an interstitial region. On the former, basis function with more or less atomistic features are employed, while on the latter plain waves are used.

While in wien2k a k-space representation of wave functions is convenient, there are many applications where also a real space picture is desired: Determining transport properties (hopping parameters), visualization and especially, in codes relying on local orbital basis functions such as DMFT [3]. One way to change to a real space representation is a Fourier transform of the Bloch states $\psi_{n\mathbf{k}}(\mathbf{r})$ from the DFT computation

$$w_{m\mathbf{R}}(\mathbf{r}) = \frac{V}{(2\pi)^3} \int_{BZ} dk e^{-i\mathbf{k}\cdot\mathbf{R}} \left(\sum_n U_{nm}^{(\mathbf{k})} \psi_{n\mathbf{k}}(\mathbf{r}) \right). \quad (1.1)$$

where the resulting real space functions $w_{m\mathbf{R}}(\mathbf{r})$ are denoted by Wannier functions. Due to the additional phase factors $U_{nm}^{(\mathbf{k})}$, which all lead to valid Wannier functions, there is generally a ambiguity in the choice of the real space basis set. This can be overcome by the choice of Wannier orbitals $w_{m\mathbf{R}}(\mathbf{r})$ which have a minimal spatial extent (spread) and are denoted by maximally localized Wannier functions (MLWF). A popular program to compute the MLWF for a given set of Bloch functions is wannier90 [4, 5].

Unfortunately, the application of wannier90 to the LAPW basis set from wien2k is not as straightforward as it is for basis consisting entirely of plain waves. In this user guide we describe the package wien2wannier which represents an interface between wien2k and wannier90. Additionally to the main program w2w, we also provide utility programs (FORTRAN, shell scripts and python scripts) to improve the workflow.

The following reference contains the derivations and more examples (it should be cited in case of scientific usage)

J. KUNEŠ, R. ARITA, P. WISSGOTT, A. TOSCHI, H. IKEDA, K. HELD
Wien2wannier: From linearized augmented plane waves to maximally localized Wannier functions, *Comp. Phys. Commun.* **181**, 1888 (2010).

1.2 Quickstart

In this section, the elementary steps of the interface are introduced, a detailed description of all the programs and scripts can be found in Section 1.3.

1.2.1 Preparatory steps

Before running the interface to wannier90, one needs a wien2k computation.

init_lapw the standard command to initialize a wien2k computation.

run_lapw the standard command to run a self-consistent cycle (add problem specific options).

case.klist.band it is useful to obtain a bandstructure before starting the interface w2w (optional).

The **case.klist.band** file can be prepared e.g. via XCrySDen[6].

xlapw1-band standard command to obtain the eigenvalues along the k-path defined in **case.klist.band**

update_FermiEnergy.sh a utility script to create the file **case.insp** and update the Fermi energy therein (taken from the **scf2** file).

x **spaghetti** the standard command to obtain a bandstructure (files **case.spaghetti.ps** and **case.spaghetti.ene**).

Note that though the final four steps are optional, the information of the bands is important: For the setup of the interface **w2w**, the bands which are to be taken into account have to be specified. Furthermore, the main character (e.g. s/p/d/f on atom 2) of these bands has to be known. It is often useful to compute, additionally to a bandstructure, a partial density of states via **tetra** to extract this information (which later needs to be specified in the input file of the interface **case.w2win**). Proceeding without the bandstructure files is possible, albeit not recommended (the error message from **prepare.w2wdir.sh** can be ignored).

1.2.2 Interface (w2w)

Before the initialization of the interface one may use the script

```
prepare_w2wdir.sh case subdir
```

to create the subdirectory **subdir** with all the required file for the workflow below. Thus, one wien2k calculation can be the starting point of multiple different **w2w** (and consequently wannier90) runs. Alternatively, the steps can be done in the original wien2k directory (note that the original **.vector** file will then be overwritten in the process)

The initialization of the interface can be done either by single commands or by the script

```
init_w2w or init_w2w -sp for spin-polarized cases
```

Initialization (init_w2w)

The script **init_w2w** takes the following steps

ksym for wannier90 a non-symmetrized k-mesh is required. Therefore the struct file is copied to **case.ksym**. The number of symmetry operations in **case.ksym** has to be set to 1 and the first operation has to be the identity (ones in the diagonal, all other values zero).

kgen standard wien2k generation of a k-mesh (with **-so** switch to deactivate the inversion operation). The size of the mesh influences the quality of the localization and visualization of Wannier functions. Remark: The interface was only tested with k-meshes without shift.

find_bands(optional) The energy window to be used for the projection on the Wannier functions is determined by the (wien2k) index of minimal and maximal included Bloch states. By use of **find_bands** this information can be extracted from a previous lapw1 run (that is from **case.output1**), when the corresponding energy window is given by [Emin Emax] (in eV with respect to the Fermi energy). A good strategy is to look at a **spaghetti** or **tetra** output and to choose the energy window of interest.

write_w2win this utility program writes the actual input file for the interface **case.w2win**. The index of Bloch bands (internal wien2k numbering) and the number of target Wannier orbitals have to be specified. Additionally, the initial character of the Wannier functions has to be given in terms of atomic orbitals labeled by the atom where the orbital is centered and by the angular momentum quantum numbers ℓ . The numbering of atoms is again the internal wien2k serial numbering, e.g. in the case of TiC, an initial function located at the C atom with p character requires the input **2 p**. The projective angular momentum quantum numbers m are chosen automatically, but can be changed by direct manipulation of the input file **case.w2win**.

write_win A utility program to write the input file for wannier90 **case.win**. Though **write_win** requires no obligatory input, the produced file can be adapted to specific needs (e.g. setting a frozen energy window for the disentanglement procedure in wannier90).

wannier90.x(preliminary run) wannier90 requires the input data to be given on a special k-mesh which includes information with respect to nearest-neighbor k-points. A preliminary call of wannier90 with the option `-pp` stores this mesh to the file `case.nnkp`.

write_w2wdef Based on wien2k the interface w2w also requires a file where the I/O file information is stored, which is called `w2w.def`.

w2w

After running `init_w2w`, one can proceed:

`xlapw1` before the actual interface run, the eigenvectors and eigenvalues for the new (non-symmetrized) k-mesh have to be computed.

`w2w case` this computes the files `case.mmn`, `case.amn` and `case.eig` for a wannier90 computation (information of the run is stored to `case.w2wout`).

In case the wien2k and the wannier90 bandstructure are to be compared, the utility program

```
shift_energy case
```

is useful at this point, since it harmonizes the Fermi energies between the two programs.

1.2.3 wannier90

If the `case.win` file was already obtained at the initialization, one can directly run wannier90 (the third program of the workflow)

`wannier90.x case` where the output is stored to `case.wout`. The Wannier orbitals should be converged to a spread which is usually smaller than the unit cell of the structure.

1.2.4 Postprocessing

After a successful wannier90 run, this package offers two main post-processing possibilities

Comparing the bandstructures

For a comparison of the initial wien2k bandstructure and the one computed with the basis of Wannier functions in wannier90, the files `case.spaghetti_ene` and `case.band.dat` can be employed. Note that wannier90 does not consider Bohr units in the latter file, so, to plot the data on the same scale, one may have to consider a Bohr to Angstrom conversion, e.g. in gnuplot

```
p 'case_band.dat' u ($1*0.5291):2 w l, 'case.spaghetti_ene' u 4:5
```

Plotting the Wannier functions

`write_wplotdef case` writes the obligatory `wplot.def` file.

`write_wplotin case` specify the 3D domain for the mesh of points in real space where the Wannier function(s) should be evaluated. The vectors are to be written in the basis of the conventional unit cell. Usually the domain should include the center(s) of the Wannier function(s).

prepare_plots.sh case all Wannier functions are plotted. The absolute square values are written to **case.m.psink** and the phases are stored to **case.m.psiarg**, where m is the index of the Wannier orbital. A single orbital m can be computed by issuing **wplot case m**, but then, the corresponding output is stored to **case.psink** and **case.psiarg**, respectively.

xsfAll.sh case converts all **case.m.psink** plus **case.m.psiarg** files in the directory to **case.m.xsf.gz** files which can be opened by XCrySDen.

xcrysdn --xsf case.m.xsf.gz within xcrysdn, open the menu, pick "Tools → Datagrid" and press OK (there should be only one datagrid). In the isosurface controls window choose an appropriate isovalue, e.g. 0.1, and check the "Render +/- isovalue" box.

1.3 Detailed description

1.3.1 Interface(w2w)

This is the main program of the interface which provides the files **case.mmn**, **case.amn** and **case.eig** for wannier90 given the output of a wien2k computation, such as, for example, the **case.vector** file. **W2w** is based on **lapwdm**, since the main task, for the computation of the overlap matrices

$$M_{mn}^{(k,b)} = \langle \psi_{m\mathbf{k}} | e^{-i\mathbf{b}\cdot\mathbf{r}} | \psi_{n\mathbf{k}+\mathbf{b}} \rangle \quad (1.2)$$

that are stored in **case.mmn**, is to determine the basis functions $\psi_{kn}(r)$ in the entire unit cell (using appropriate boundary conditions on the muffin-tin spheres).

Additionally to the usual lapwdm input, a file case.nnkp is required which defines the **b** vectors for each **k** in Eq. (1.2). Wannier90 automatically delivers this file when called with the flag "-pp".

Execution

The program w2w is executed by issuing the command:

w2w case or **w2wsp -up/-dn/-so case** (spin-polarized or spin-polarized with spin-orbit coupling)

Adding the flags "-up" or "-dn" tells w2wsp to use the up or down input files. If the "-so" option is issued, w2w is called for both up and down components sequentially. Afterwards the spin components are automatically added by **combine.spinefiles**.

Note that **w2w** searches the **case.dayfile** for "-c" flags to determine whether to use the real or complex version. A certain version can be invoked directly via

w2wr w2w.def (real) or **w2wc w2w.def**(complex)

Input

A sample input file for **w2w** is given below. It can be generated via **write_w2win**

```
----- top of file: case.w2win -----
BOTH
 21 23      # min band Nmin, max band Nmax
 3 3       # LJMAX max in exp(ibr) expansion, \#Wannier functions
2      #number of entries per orbital (list of projected orbit)
2 2 -2    0.00000000  0.70710677 # index of atom, L, M, coef.(complex)
2 2 2     0.00000000 -0.70710677 # index of atom, L, M, coef.(complex)
```

```

2      #number of entries per orbital (list of projected orbit)
2 2 -1 -0.00650789 -0.70707685 # index of atom, L, M, coef.(complex)
2 2 1 -0.00650789 -0.70707685 # index of atom, L, M, coef.(complex)
2      #number of entries per orbital (list of projected orbit)
2 2 -1 0.70707685 -0.00650789 # index of atom, L, M, coef.(complex)
2 2 1 -0.70707685 0.00650789 # index of atom, L, M, coef.(complex)
----- bottom of file -----

```

Interpretive comments on this file are as follows:

line 1: free format
switch

AMN	Only determine the initial orbital projections case.amn
MMN	Only determine the overlap matrices case.mmn
BOTH	Determine both the initial orbital projections and the overlap matrices

line 2: free format
Nmin Nmax

Nmin	the minimal Bloch band from the vector file to be taken into account, determining the lower edge of the energy window
Nmax	the maximal Bloch band from the vector file to be taken into account, determining the upper edge of the energy window

line 3: free format
LJMAX NPROJ

LJMAX	the number of terms which are used in the spherical harmonics expansion to approximate $\exp(-ikb)$, usually 3-4 is sufficient.
NPROJ	the number of target Wannier functions

line 4: free format

nent	the number of entries for the first initial orbital projection
------	--

line 5: free format
IA L M X1 X2

IA	the atom where the entry is centered
L	the angular momentum index ℓ for this entry
M	the projective angular momentum index m for this entry
X1	the real part of the coefficient of this entry
X2	the complex part of the coefficient of this entry

>>> **line 5 has to be repeated nent times**

>>>the whole section of line 4 + subsequent lines has to be repeated NPROJ times.

1.3.2 Plotting Wannier functions (wplot)

Also contained within the package is a visualization program. It provides the Wannier function projections onto real space $|w(r)|^2$, where r is given on a certain spatial mesh (to be specified below). Wannier90 automatically creates a **case.chk** file which contains (amongst other things) the unitary transformation matrices, which build the Wannier function out of the original wien2K Bloch states.

wplot is based on **lapw7** from wien2k sharing the general structure of the input file. In **lapw7** one decides for a certain Bloch function to be plotted, whereas in **wplot** multiple Bloch bands are contributing to a Wannier function in general. The square $|w(r)|^2$ is stored to the file **case.psink** in the order $((\text{psi}(\text{ix}, \text{iy}, \text{iz}), \text{ix}=1, \text{nx}), \text{iy}=1, \text{ny}), \text{iz}=1, \text{nz})$, whereas the corresponding phases are written to **case.psiarg**. Information about the run of the program can be found in **case.wplotout**.

Execution

The program **wplot** is executed by issuing the command

```
wplot case [idx.wann] or wplotsp -up/-dn case [idx.wann](spin-pol.)
or wplotso -up/-dn case [idx.wann](spin-pol. with spin-orbit coupling)
```

where the optional input **idx.wann** overwrites the index of the Wannier orbital to be plotted defined in the input file **case.wplotin**. Note that **wplot** searches the **case.dayfile** for "-c" flags to determine whether to use the real or complex version. A certain version can be invoked directly via

```
wplotr wplot.def (real) or wplotc wplot.def(complex)
```

Input

A sample input file for **wplot** is given below. It can be generated via **write.wplotin**

```
----- top of file: case.wplotin -----
3D ORTHO      # mode 0(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: format(A4)
switch

MODE Sets the dimension of the r-mesh (only tested for 3D orthogonal meshes)

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 3-5: free format

ix,iy,iz,idv Coordinates of the end points of each grid axis in x,y,z direction, respectively.

line 6: format (3I3,A40)

nx,ny,nz number of spatial mesh points in each direction. Note that the script wplot2xf.py currently assumes that the number of mesh points is the same in all directions. The additional input in this line is unused.

line 6: format(A3)

tool post-processing of the orbital

DEP Each wave function is multiplied by a complex phase factor to align it (as most as possible) along the real axis (the so-called DEPhasing option, untested).

NO No post-processing

line 7: format(A3,1X,A3,1X,A5)

WAN iunit whpsi

WAN a Wannier function is to be plotted (all other switches have been deactivated)

iunit the physical units for the Wannier function output

ANG Å units are used

AU or Atomic units are used

ATU

whpsi the relativistic component to be evaluated

LARGE The large relativistic component for each wave functions is evaluated

SMALL The small relativistic component for each wave functions is evaluated

line 8: free format

iskpt iswann

iskpt this is an unused option

iswann the index of the Wannier function to be plotted. This input is not considered when **wplot** is called with a non-zero additional index option.

1.3.3 Optical conductivity (woptic)

To compute the optical conductivity in the basis of Wannier orbitals the package woptic is provided. It uses the Green function formalism

$$\sigma_{ij}(\Omega) = \frac{2\pi e^2 \hbar}{V} \sum_k \int d\omega \frac{f(\omega) - f(\omega + \Omega)}{\Omega} \text{tr} [A(\omega, k) v_i(k) A(\omega, k) v_j(k)] \quad (1.3)$$

where σ_{ij} is the (i, j) element of the optical conductivity matrix with $i, j = x, y, z$, V denotes the unit cell volume, f the Fermi function, A the generalized spectral function¹, and v_i the group velocity in the direction i . The numerical bottleneck in Eq.(1.3) is the k -summation, since usually a lot of k points are required to obtain converged results. For a speed-up in k -mesh convergence the algorithm `woptic` therefore employs an adaptive refinement of tetrahedral elements which build the k space.

The adaptive algorithm consists of two main programs: `woptic_main`, where the optical conductivity is calculated and `refine_tetra`, where the tetrahedral k -mesh is refined. Though the single programs may be called separately, it is recommended to call `woptic` directly. Note that it is required to run `x optic` at least once before `woptic`.

Execution

The program `woptic` is executed by issuing the command

```
woptic [-i n,-restart m,-peierls] case
```

where the optional input `-i n` defines n as the maximal number of iterations (default:5), `-restart m` triggers a restart at iteration m , and `-peierls` avoids wien2k calls of `optic` for a optical conductivity in the Peierls approximation [7]. There are currently no spin-polarized versions available.

In the standard output one may follow the iterative k -mesh refinement. An important role has the quantity "estimator" which is proportional to the integration error of Eq. (1.3). It is given in arbitrary units and should decrease during the iterations. A good test of convergence is the quantity "sumrules" corresponding to the Ω -integration of σ . After some iterations it should not change dramatically any more.

Input

A sample input file for `woptic` is given below. A template can be found in `$W2WROOT/templates`

```
----- top of file: case.woptin -----
OPT 2 #Mode, matel mode
1 0.1 0.1 40.0 #emax,dw,deltino,beta
12 35 21 23 #nemin_w2k nemax_w2k nemin_w90 nemax_w90
0 #use internal unit cell hopping
1.0 # separation Drude/interband in eV
T #non-interacting mode
# matel mode: 1 ... take dH/dk as matrix elements
#              2 ... take matrix elements from wien2k rotated in Wannier basis
#              3 ... take unchanged matrix elements from wien2k
----- bottom of file -----}
```

Interpretive comments on this file are as follows:

line 1: `format(A3,1X,I1)`
`FLAG matelmode`

`OPT` calculation flag (all other options have been deactivated)

`matel mode` defines the group velocities used in the calculation

¹In multiorbital cases, A is defined via the Greens function matrix G as $A = (G - G^*)/2\pi i$, see e.g. Ref [7]

- 1 take dH/dk as matrix elements where H is in the Wannier basis. This is the so-called Peierls approximation.
- 2 take the dipole matrix elements computed by wien2k's optic in the Wannier basis.
- 3 take the dipole matrix elements computed by wien2k's optic the original diagonal basis. This should give similar results as optic.

line 2: free format

$E_{max}, d\omega, \delta, \beta$

- E_{max} The maximal energy Ω for which $\sigma(\Omega)$ is computed (in eV).
- $d\omega$ The energy increment for which $\sigma(\Omega)$ is computed (in eV).
- δ The broadening added for non-interacting bands (in eV).
- β The inverse temperature used in the computation (in eV^{-1}). The conversion to temperature T in Kelvin is $\beta = 11604/T$.

line 3 free format

$nemin_w2k, nemax_w2k, nemin, nemax$

- $nemin_w2k$ The band index defining the lower edge of the energy window which is included in the calculation.
- $nemax_w2k$ The band index defining the upper edge of the energy window which is included in the calculation.
- $nemin, nemax$ The band numbers defining the energy window of the underlying wannier90 run. For these bands the Hamiltonian in the Wannier basis is used (if matel mode is 1 or 2). Note that $nemin_w2k \leq nemin$ and $nemax_w2k \geq nemax$

line 4: free format

use intercell hopping

- 0/1 in the case if matel mode equals 1 (Peierls approximation) with multiple Wannier centers, one can include an additional term to the group velocities covering hopping within the same unit cell (see Ref.[7] for details). The additional term depends on the interatomic distance which should be stored in **case.distmatrix** (template available in \$W2WROOT/templates).

line 5: free format

- E_{sep} The energy below which optical spectral weight is contributed to the Drude peak (in eV). This value only influences the way the sumrules are written out.

line 6: format(L1)
non-interacting mode

T/F For non-interacting calculations the broadening δ defined above is used for all orbitals. For an interacting calculation a file **Sfreal.dat** with the self energy on real frequencies $\Sigma(\omega)$ is expected (template available in \$W2WROOT/templates). In addition to the interacting orbitals with the self energy $\Sigma(\omega)$ there still can be non-interacting orbitals (with broadening δ) contributing if `nemin_w2k < nemin` or `nemax_w2k > nemax`.

1.3.4 Other utility programs and scripts

convert_Hamiltonian

This is a program to Fourier transform the wannier90 real space Hamiltonian $H(R)$ given in **case_hr.dat** to k space $H(k)$ where the k -list is given by **case.klist**. The usage is

```
convert_Hamiltonian case
```

combine_spinfiles

In calculations including spin-orbit coupling, wannier90 has to be invoked on a single .mmn file which contain the sum of .mmnup and .mmndn files (and correspondingly for amnup and amndn). This FORTRAN program joins the w2w files .mmnup+.mmndn→.mmnso, .amnup+.amndn→.amnso. It is executed via

```
combine_spinfiles case
```

find_bands

This FORTRAN program searches **case.output1** to identify the bands which lie within a certain energy window. The program is executed by issuing the command:

```
find_bands [-up/-dn/-soup/-sodn] case Emin Emax
```

where Emin is the lower and Emax the upper energy to frame the energy window (eV). Emin and Emax are supposed to be given with respect to the Fermi energy (from the **case.scf2** file). The output consists of a list of all k points and the number of eigenvalues, the lower and the upper band index within [Emin Emax] and the corresponding number of bands for the each k point. The options specify which **case.output1** and **case.scf2** is analyzed, e.g. **case.outputso** and **case.scf2up** in the case of "-soup".

init_w2w

This script can be invoked to be guided through the initialization process of the interface w2w. It is executed by

```
init_w2w [-sp]
```

The -sp flag starts the spin-polarized version of the initialization. Note that the script is not intended to be called in case of calculations including spin-orbit coupling.

join_vectorfiles

The current version of w2w does not support k-point parallelization. If the preliminary call of lapw1 is k-point parallel, the .vector and .energy input files have to be merged in order to invoke w2w. The usage of the FORTRAN program for this purpose is

```
join_vectorfiles [-up/-dn/-soup/-sodn] [-c] case m
```

where m is the number of cores which were used in parallelized lapw1. All .vector_* files are merged into one .vector file containing the header of the first file .vector_1 (and correspondingly for the .energy files). For the complex version use the -c flag. If the option "-soup" or "-sodn" is included the .vectorsoup or .vectorsodn are joined (in this case the complex version is automatically employed).

prepare_plots.sh

Useful if all the Wannier functions are to be plotted on the same spatial grid. The script prepare_plots is executed by issuing the command:

```
prepare_plots.sh case
```

where the case.psink and case.psiarg files for the Wannier function idx_wann is stored to case_<idx_wann>.psink and case_<idx_wann>.psiarg, respectively. There is also a spin-polarized

```
prepare_plotssp.sh -up/-dn case
```

and spin-orbit version

```
prepare_plotsso.sh -up/-dn case
```

prepare_w2wdir.sh

A wien2k computation can be the starting point for various runs of the interface (and wannier90). This script creates a new subfolder of a wien2k directory and is invoked by

```
prepare_w2wdir.sh [-c/-sp/-spc] case subdir
```

where subdir is the name of the subdirectory to be created and the options are switches to indicate a complex, a spin-polarized and a complex spin-polarized computation, respectively.

rephase

Often, the plotted Wannier functions show a non-trivial phase. This program reads the psiarg files to determine the most probable phases of all Wannier functions. Then, the input for the interface file case.w2win is rewritten in a way that a subsequent run of w2w and wannier90 leads to real Wannier orbitals (this does not work in all cases). The FORTRAN program rephase is invoked by the command

```
rephase case [-w] [-up/-dn] [-wf=idx_wann]
```

where the option -w means that case.w2win is automatically updated (future wien2wannier runs will then have a higher probability of producing real Wannier orbitals) and the option -w idx_wann indicates that the action is only applied for the Wannier function idx_wann.

shift_energy

Within the file `case.vector` the Fermi energy is not considered, thus, wannier90 in general returns a bandstructure which does not have the same Fermi level as the original one from `spaghetti`. To avoid a mismatch the FORTRAN program `shift_energy` shifts the eigenenergies in `case.eig` by the Fermi energy given in `case.scf2` and is invoked by

```
shift_energy [-up/-dn] case
```

update_FermiEnergy.sh

This script is actually a utility script for wien2k use. It updates the Fermi energy read from `case.scf2` in the input file for `spaghetti`, `case.insp`. It is invoked by

```
update_FermiEnergy.sh [-up/-dn]
```

wplot2xsf.py

This python script was contributed by Nikolaus Frohner. It converts the `case.m.psink` and `case.m.psiarg` file to `xsf` format and is invoked by

```
wplot2xsf.py [--noshift] [--nophase] case m >output.xsf
```

Note that $w(r)$ where $\cos(\phi(w(r)))$ is positive are mapped to $+|w(r)|^2$ and $w(r)$ with negative $\cos(\phi(w(r)))$ are mapped to $-|w(r)|^2$. The spin-polarized version of this script is

```
wplot2xsfsp.py --up/--dn [--noshift] [--nophase] case m  
>output.xsf
```

By default `wplot2xsf.py` shifts the Wannier orbitals by the translation vector from `case.centres.xyz` (`case.centres.xyzup/dn` in case of spin-polarization). This shift is omitted if the “-noshift” option is used. When the option “-nophase” is included only the `psink` file is considered (no sign mapping for the phase).

Note that the current version of `wplot2xsf.py` assumes an equal number of spatial mesh points in all directions.

programs for preparing the input and the def files

There is a series of FORTRAN programs writing the input and definition files for `w2w` and `wplot`. The input files of the two programs can be obtained via

```
write_w2win [-up/-dn] case and write_wplotin [-up/-dn] case
```

where user input is required to specify certain input values. On the other hand, the creation of the definition `w2w.def` and `wplot.def` requires no additional inputs

```
write_w2wdef [-up/-dn/-soup/-sodn] case and  
write_wplotdef [-up/-dn/-soup/-sodn] case
```

There is also a program preparing the input file for wannier90 from various information of other files

```
write_in [-up/-dn] case
```

The flags `-soup` and `-sodn` invoke the version of the programs considering spin-polarized up and dn components in a spin-orbit calculation.

xsfAll.sh

This script converts all Wannier function of the problem to xsf files (runs `wplot2xsf.py` for all Wannier orbitals). This script is invoked by

```
xsfAll.sh case
```

(spin-polarized version: `xsfAllsp.sh -up/-dn case`)

1.3.5 Getting help

Additional information about all programs can be accessed via the help flag, "program -h". Before asking the authors, please take a look at the FAQ section below which may answer the question.

1.3.6 Calculations with spin-orbit coupling

This explains the workflow in case of non-spin-polarized calculations (one has to fake spin-polarization in this case):

- ▶ converge a SO calculation `run_lapw -so ...`
- ▶ `write_w2win -up` and `write_w2win -dn`. Note that the numbering of certain Bloch bands has doubled with respect to a wien2k run without spin-orbit coupling. Also, the total number of Bloch bands and Wannier orbitals has doubled. In particular, the header of the `.w2winup` file has to be the same as `.w2windn` but the initial projections may be different between up and dn. This is because they correspond to up and dn component of the same orbital.
- ▶ prepare k-mesh for w2w calculation(e.g. by copying the `case.struct` file to `case.ksym`, setting the symmetry operation to one, issuing `x kgen -so, write_win -up case` and `wannier90.xsp -up -pp case`)
- ▶ fake a spin-polarized calculation by copying `case.vsp` to `case.vspup` and `case.vspdn` as well as `case.vns` to `case.vnsup` and `case.vnsdn`
- ▶ `x lapw1 -up,x lapw1 -dn` . Now up and dn eigenvalues should be identical.
- ▶ `x lapwso -up`
- ▶ `w2wsp -so case`. Now you should have `.amnup/.amndn`, `.mmnup/.mmndn` and `.eigup/.eigdn` files. Additionally, you should have the SO version of the first two files `case.mmnso` and `case.amnso`, where the spin components are added up.
- ▶ copy `case.scf2` to `case.scf2up` or `case.scf2dn`
- ▶ `shift_energies -up` or `-dn` and copy either `case.eigup` or `case.eigdn` to `case.eigso` (up and dn should be the same anyway).
- ▶ `write_win -up/-dn case` (if not already done)
- ▶ `wannier90.xso case`, output can be found in `case.woutso`.

This explains the workflow in case of spin-polarized calculations(very similar to the previous but no faking this time):

- ▶ converge a SO calculation `runsp_lapw -so ...`
- ▶ `write_w2win -up` and `write_w2win -dn`. Note that the numbering of certain Bloch bands has doubled with respect to a wien2k run without spin-orbit coupling. Also, the total number of Bloch bands and Wannier orbitals has doubled. In particular, the header of the `.w2winup` file has to be the same as `.w2windn` but the initial projections may be different between up and dn. This is because they correspond to up and dn component of the same orbital.
- ▶ prepare k-mesh for w2w calculation(e.g. by copying the `case.struct` file to `case.ksym`, setting the symmetry operation to one, issuing `x kgen -so, write_win -up case` and `wannier90.xsp -up -pp case`)

- ▶ x lapw1 -up,x lapw1 -dn
- ▶ x lapwso -up
- ▶ w2wsp -so case. Now you should have **.amnup/.amndn**, **.mmnup/.mmndn** and **.eigup/.eigdn** files. Additionally, you should have the SO version of the first two files **case.mmnso** and **case.amnso**, where the spin components are added up.
- ▶ shift_energies -up or -dn and copy either **case.eigup** or **case.eigdn** to **case.eigso** (up and dn should be the same anyway).
- ▶ write_win -up/-dn case (if not already done)
- ▶ wannier90.xso case, output can be found in **case.woutso**.

1.4 Installation

Download the package from

```
http://www.wien2k.at/reg_user/unsupported/wien2wannier
```

to a directory of choice, e.g. /opt/wien2wannier(in the following \$W2WROOT). Then,

```
gzip -d wien2wannier.tar.gz  
tar -xvf wien2wannier.tar.gz
```

Choose the proper compile script to match the system in the \$W2WROOT/sys directory, copy it to \$W2WROOT/compile_w2w.sh and run it to compile **w2w**, **wplot** and all FORTRAN utility programs

```
compile_w2w
```

After the compilation, every user should call the

```
link_w2w.sh or link_w2w.csh
```

from the \$W2WROOT directory to link the executables. To bring the changes into effect, restart of the shell is required (or for bash issuing the command

```
source .bashrc
```

from the users home directory).

Note: For the standard visualization method presented in this guide, XCrySDen [6] is needed.

In case of compilation problems: Since wien2wannier is based on wien2K, the easiest way to obtain the correct Makefile is to look into the corresponding wien2K folder. The correspondences are **lapwdm**→**w2w**, **lapw7**→**wplot**. Thus, when experiencing problem with the compilation of **w2w** or **wplot**, copy the essential parts of the Makefiles from these wien2k directories into the Makefiles of the corresponding wien2wannier folders (essential meaning the linking and options block, not the filenames).

The package was tested on wien2K Version WIEN2k_10.1 (Release 7/6/2010) and wannier90 Version 1.2 .

1.4.1 Example

In the directory \$W2WROOT/templates there are three debugging scripts **testcase1.sh**, **testcase2.sh** and **testcase3.sh** which show the workflow for the compound SrVO₃(debug.struct) in the standard case, the spin-polarized case and the spin-polarized case including spin-orbit coupling, respectively. To actually run the scripts, create a directory of choice and copy the corresponding script in that directory. If invoked, it should produce a xcrysdn visualization of a V- t_g orbital if wien2k, wien2wannier, wannier90 and xcrysdn are correctly installed.

1.4.2 Installation of wannier90

In the following, some hints for the wannier90 installation (details can be found in [5]):

- download a recent version of wannier90 from **www.wannier.org**

- ▶ expand the files
- ▶ choose the proper configuration file from the config folder and copy it to the wannier90 main directory renaming in `make.sys`
- ▶ update the library directories (e.g. the location of blas) in `make.sys`
- ▶ type `make`
- ▶ add the executable wannier90.x to your \$PATH, e.g. by adding the wannier90 main directory directly in `.bashrc`

1.5 FAQ

How does one choose the initial projections?

There is no general rule to choose the initial projections which have to be prepared in the `w2win` file. There are, however, some hints which usually help:

- ▶ In the energy interval of interest, identify the main atoms which contribute to the partial DOS. These atoms are usually good centers for the initial projections. The character of these atomic contributions can also be seen via the partial DOS. This usually leaves you with a clear idea which angular momentum quantum number combination is adequate.
- ▶ In problems with higher (e.g. octahedral) symmetry, crystal field splitting often separates originally degenerate manifolds in energy. Then the intuitive choice would be to take m_l combinations for the corresponding crystal field states, e.g. for t_{2g} .
- ▶ A glance at the bandstructure often helps to consider the multiplicity of equivalent WF and to identify certain hybridizations.
- ▶ It often helps to define the projections mutually orthogonal. In this case, the first step in wannier90, the orthogonalization is already taken into account at the level of wien2wannier.

The resulting Wannier orbitals are not localized

Several possible reasons (this list is not complete)

- ▶ mixing up Bohr/Angstrom units in the `.win` file at the unit cell definition.
- ▶ choosing non-optimal initial projections
- ▶ the number of k-points is too small (can be easily checked by increasing the number of k-points)

The bandstructure of wien2k and wannier90 does not match

In case of mismatch in terms of the x axis (k vectors), this usually traces back to a Bohr/Angstrom mismatch of wien2k and wannier90, which can be overcome by simply rescaling the x axis, eg. in gnuplot:

```
p 'case_band.dat' u ($1*0.5291):2 w l, 'case.spaghetti_ene' u 4:5
```

If the x axis is fine and the bands themselves differ strongly, then one might have

- ▶ non-optimal initial projections
- ▶ too few k-points (can be easily checked by increasing the number of k-points)
- ▶ chosen the frozen energy window in wannier90 in a wrong way. This usually happens in cases with disentanglement where wannier90 has difficulties identifying the optimal subspace.

I obtained the error message "Eq. (B1) not satisfied" when starting wannier90

The default tolerance for the k mesh in wannier90 is quite small. Thus, it can happen that the k mesh from wien2k is correct but completeness relations in wannier90 are not fulfilled. There is an easy workaround, since wannier90 provides an input variable `kmesh_tol` which has to be inserted in the `.win` file and set to an appropriate value (eg. 0.0001).

When plotting a Wannier orbital in xcrysden I can see nothing

One possible reason is that the WF centers do not have to be necessarily in the home unit cell at $[0\ 0\ 0]$. In the plotting workflow, the interface programs usually account for this by translating the WF to the home cell. However, the spatial mesh for `wplot` has to be defined with respect to the original centers which come out of wannier90. These centers can be found in the `.wout` file, and are automatically printed out when calling `write_wplotin` in order to write the input for `wplot`. Example: assume the WF is centered at $[-0.5\ -0.5\ -0.5]$ in the basis of conventional unit vectors. Then, appropriate mesh vectors might be

```
-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
```

since this defines a cube centered at $[-0.5\ -0.5\ -0.5]$.

How can I access the Hamiltonian in the basis of Wannier functions?

The Hamiltonian $H(k)$ in the basis of Wannier orbitals is computed in wannier90 internally in the subroutine `hamiltonian_get_hr()` at `hamiltonian.F90` (wannier90 version 1.2).

My Wannier orbitals are not centered at the home unit cell when plotted with xcrysden

Within wannier90 it happens quite frequently that a maximally localized Wannier function is not centered within the home unit cell, which is displayed by default by xcrysden. The interface program `write_win` automatically activates the option `translate_home_cell` in the wannier90 input file. Then, wannier90 should produce a file `case.centres.xyz`, where the vectors of all WF are stored which translate the orbital to the home unit cell. If this file cannot be located by `wplot2xf.py` or the option `-noshift` is activated, no translation is conducted and the WF appear centered at their original position.

There is also a second, more peculiar, reason for the translation problem. wannier90 is very harsh when it comes to translating the centres of WF. If a Wannier orbital is centered only 1% outside of the home unit cell (which may happen easily due to numerical reasons) the WF is translated by a unit vector. In these cases, either manipulation of the `case.centres.xyz` file, or the wannier90 source code can help.

1.6 Acknowledgment

Many thanks to Ryotaro Arita, Alessandro Toschi, Hiroaki Ikeda, Karsten Held, Peter Blaha, Karlheinz Schwarz, Nikolaus Frohner, Philipp Hansmann, Nico Parragh and Giorgio Sangiovanni.

Bibliography

- [1] P. BLAHA, K. SCHWARZ, P. SORATIN, AND S.B. TRICKEY (1990). *Comp. Phys Commun.* **59**, 399 from www.wien2k.at
- [2] P. BLAHA, K. SCHWARZ, G. MADSEN, D. KVASNICKA, J.LUITZ (2009). *Userguide Wien2K*
- [3] K. HELD, *ADV. PHYS.* **56** , 829-926 (2007).
- [4] A. A. MOSTOFI, J.R. YATES, Y.-S. LEE, I. SOUZA, D. VANDERBILT AND N. MARZARI **wannier90** *A Tool for Obtaining Maximally-Localized Wannier Functions*, *Comput. Phys. Commun.* **178**, 685 (2008); <http://arxiv.org/abs/0708.0650>
- [5] A. A. MOSTOFI, J.R. YATES, Y.-S. LEE, I. SOUZA, D. VANDERBILT AND N. MARZARI (2007). **wannier90** *Userguide*
- [6] A. KOKALJ, *COMP. MATER. SCI.*, 2003, VOL. **28**, P. 155. Code available from <http://www.xcrysden.org>
- [7] J.M. TOMCZAK, *SPECTRAL AND OPTICAL PROPERTIES OF CORRELATED MATERIALS, PhD Thesis, L'ECOLE POLYTECHNIQUE, PARIS* (2007)
- [8] J. KUNEŠ, R. ARITA, P. WISSGOTT, A.TOSCHI, H. IKEDA, K. HELD *Wien2wannier: From linearized augmented plane waves to maximally localized Wannier functions*, *Comp. Phys. Commun.* **181**, 1888 (2010).