WIEN2k at the command line

- **Create a struct file**
  - makestruct: *generate it from spacegroup, latt.param, atomic positions*
    - cp init.struct case.struct
  - x cif2struct*:* *generate it from case.cif file (databases, web)*
  - x xyz2struct*:* *generate from POSCAR case.xyz file (or xyz format, see UG)*
    - setrmt [-r 3 or -a Si:2.2,O:1.6] # and copy case.struct_setrmt
- inspect the generated case.struct
  - less case.struct *# does it have the expected number of atoms ?*
- view the structure
  - xcrysden --wien_struct case.struct
  - VESTA $PWD/case.struct
- are the atomic distances as given in the reference, symmetry ok ?
  - x nn *# check distances and angles in* case.outputnn
  - x sgroup *# check case.outputsgroup, eventually accept case.struct_sgroup*

Databases (besides google)

- ***materialsproject**: calculations of „all materials in all possible structures" (what is experiment ?)*
- ***aflow.org**: theoretical data, prototype structures*
- ***nomad-lab.eu**: theoretical data, rather unstructured, depository*
- ***oqmd.org**: open quantum materials database*
- ***computational materials repository** (CMR)*
- ***Open Materials Database***
- ***DAICS** (https://daics.net): Database of Ab Initio Crystal Structures*

- ***ICSD** (Inorganic crystal structure database): experimental data (not free)*
- ***Crystallography Open Database**:*
- ***American Mineralogist Crystal Structure Database***

- ***Bilbao Crystallographic Server** of crystallographic symmetry information*

Initialization (generate input files)

- init_lapw –h
  - -prec X -> set precision 0/1/2/3/0n/1n/2n/3n (default:1; "n" means „no metal" for reduced k-mesh)
  - -m -> manual step-by-step mode (not recommended anymore)
  - -sp -> in batch mode: select spin-polarized calculation
  - -nodstart -> creates new input files, but no case.clmsum (assuming you already have a converged calculation)
  - -nokshift -> produces an unshifted k-mesh (including Gamma)
  - -hdlo -> set HDLOs in lstart (case.in1, automatic with -prec 2/3)
  - -nohdlo -> do not set HDLOs in lstart (case.in1) (overwrites -prec 2/3 setting)
  - -red X -> RMT reduced by X% or with X=Si:2.0,O:1.6 (default: RMT unchanged)
  - -ecut X -> energy separation (or Q/sphere) between core/valence (default: -6.0 Ry or automatic)
  - -rkmax X -> RKMAX (default: automatic)
  - -lvns X -> LVNS_max (default: automatic)
  - -fermit X -> use TEMP with smearing by X Ry (default: TETRA, or TEMP for 2D)
  - -fermits X -> use TEMPS with smearing by X Ry (default: TETRA)
  - -numk X -> use X k-points in full BZ (default: automatic);
    or: 0 NX NY NZ (with unshifted mesh) or: -1 delta-K

initialization

- For spin-polarized case: define initial magnetic moments before init_lapw. Mandatory for antiferromagnets, recommended for calculations with non-magnetic atoms (O)
  - instgen_lapw –ask *and answer „u","d","n" for up/dn/no-moment for each atom.*
  - init_lapw –sp [–prec X …]
- check the output on the screen:
  - *nn distances and RMTs ok ?*
  - *symmetry ok ? Do you have inversion (4x cpu-time, 2x memory) ?*
  - *what are my core/semicore/valence states ? any core leakage ??*
  - *warning about „large sphere" (or even automatic reduction of RMT to 2.3 for –prec2/3)*
  - *Rkmax ?*
  - *k-mesh (total number of IBZ points – important for possible parallelization)*
- prepare possible parallelization (**.machines** file)
  - cp $WIENROOT/SRC_templates/.machines **.** # and adapt it to your needs

scf cycle

- run_lapw -h (runsp runfsm –M x runafm)
  - *-cc LIMIT -> charge convergence LIMIT (0.0001 e)*
  - *-ec LIMIT -> energy convergence LIMIT (0.0001 Ry)*
  - *-fc LIMIT -> force convergence LIMIT (1.0 mRy/a.u.)*
  - *-str LIMIT -> stress convergence LIMIT (.1 GPa)*

    default is -ec 0.0001; multiple convergence tests possible
  - *-i NUMBER -> max. NUMBER (40) of iterations*
  - *-min -> force optimization using MSR1a*
  - *-NI -> do NOT remove case.broyd* (default: rm *.broyd* after 60 s)*
  - *-p -> run in parallel (needs .machine file [speed:name])*
  - *-so -> run SCF including spin-orbit coupling*
  - *-hf -> HF/hybrid-DFT calculation*
  - *-dftd4 -> calculate the dispersion energy with the DFT-D4 method*
  - *-nlvdw -> include a nonlocal van der Waals functional*
  - *-scratch dir -> scratch-dir (for vector files; ./ for PWD; default from $SCRATCH)*
  - *-it -> use iterative diagonalization*

wien2k at the command line

- mkdir case ; cd case
- makestruct or cp ~/Downloads/xxx.cif case.cif ; x cif2struct
- setrmt …
- xcrysden --wien_struct case.struct
- init_lapw –prec 2n ; [edit .machines]
- run_lapw … [–p]
- grep :ENE case.scf # or other parameters of interest for convergence
- save_lapw exp_prec2n or continue
- run_lapw –NI –ec 0.000001 –cc 0.00001 ; save_lapw ….
  - *do properties like DOS, bands, density plots, optics, xspec or*
  - *optimize positions: run_lapw –min …*
- … do something else (e.g. use a different XC functional or volume)
- …
- restore_lapw exp_prec2n (restores case.struct, case.in*, density and pot)
  - *do properties like DOS, bands, density plots, optics, xspec (you have to recreate case.vector using x lapw1)*

monitor scf cycle

- less case.dayfile

```
....
 cycle 18   (Mon Nov  4 08:10:17 CET 2013)  (23/82 to go)

>   lapw0  (08:10:17) 5.997u 0.024s 0:06.03 99.6%      0+0k 0+2920io 0pf+0w
:FORCE convergence: 0 1 0 XCO 2.41 ZCO 2.51 ZCO 1.88 ZCO
>   lapw1            (08:10:24) 84.830u 5.067s 0:46.06 195.1%    0+0k 0+331920io 0pf+0w
>   lapw2            (08:11:10) 50.763u 3.089s 0:30.99 173.7%    0+0k 16+3984io 0pf+0w
>   lcore  (08:11:41) 0.032u 0.001s 0:00.03 100.0%     0+0k 0+472io 0pf+0w
>   mixer  (08:11:41) 0.166u 0.020s 0:00.27 66.6%      0+0k 0+4440io 0pf+0w
:ENERGY convergence:  0 0 .0000026150000000
:CHARGE convergence:  0 0.0000 .0003474
ec cc and fc_conv 1 1 0

   cycle 19            (Mon Nov  4 08:11:42 CET 2013)  (22/81 to go)
```

- top
  - *list the (most cpu-intensive) running programs. updated every 4 seconds.*
  - *shows used cpu-time, number of cores it uses and memory*
  - *quit using „q"*

check convergence

- case.scf contains history of scf cycle (final results at the end)
  - *important quantities are labeled by   :XXXxxx*
  - *most common labels:*
    - :ENE (energy)   :DIS (charge distance [e-])   :FER (EF)   :GAP (only with TETRA) :WAR (something might be not ok)   :INFO (info for experts)   :MMT (total magnetic spin moment)   :FR (max. force for force minim.)   :VZERO (potential in vacuum for surfaces)
    - :EIGxxxx1 (eigenvalues at 1st k-point) :BANxxxxx (band ranges [Ry] around EF)
    - :NTOxxx (new total charge in sphere xxx)   :MMIiii (magn.moment)
      :FGLxxx (force on atom iii)   :POSxxx (position of atom iii)
      :QTLxxx (partial charges)   :CHAxxx (charge and RMT)
      :EPLxxx   :EPHxxx (mean energy + charge in lower and upper valence bands)
      :EFGxxx   :ETAxxx (electric field gradient and eta)
    - :ITE (iteration number) :LAT (lattice param.) :VOL (volume) :POT (xc-functional) :IFFT (FFT-mesh) :RKM (Rkmax + matrix size) :KPT (IBZ k-points) :NEC01 (charge leakage due to core)   :FITxxx (sigma of xc-fit at RMT)
  - *grep :LABEL   case.scf   or   grepline :LABEL   ' *scf '   2*
    - add: alias grep="grep –i" to your .bashrc startup file to make it case insensitive

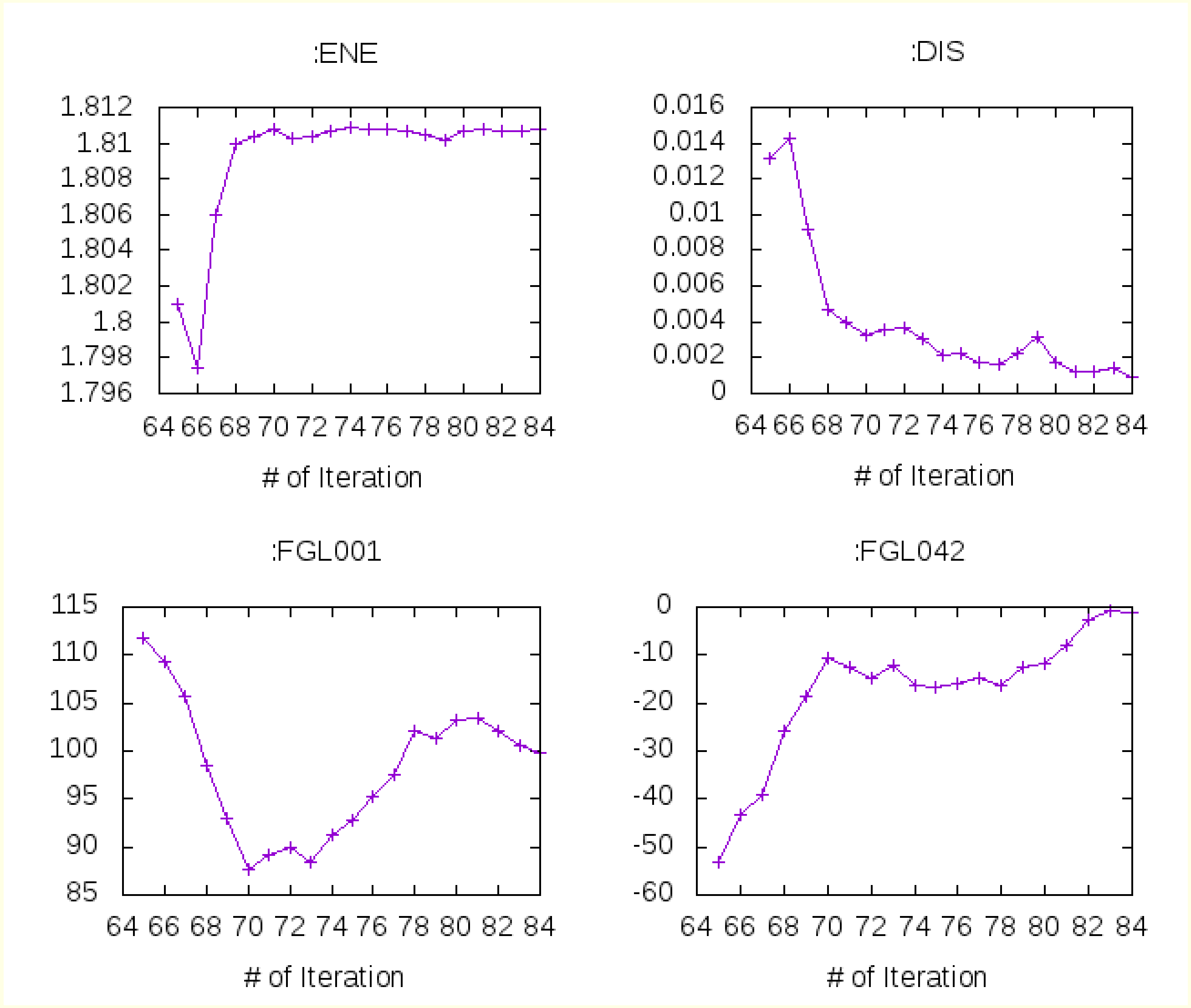
check convergence

- grep :ENE Ni.scf
  ```
  :ENE  : *WARNING** TOTAL ENERGY IN Ry =      -3036.67487071   ← usually uncritical unless
  :ENE  : *WARNING** TOTAL ENERGY IN Ry =      -3036.87575839      at final scf cycle
  :ENE  : *WARNING** TOTAL ENERGY IN Ry =      -3036.79828375
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79779475
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79567887
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79583620
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79596327
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79596900
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79597699
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79597591
  :ENE  : ********** TOTAL ENERGY IN Ry =      -3036.79597570
  ```

- grepline :ENE ´*scf´ 1
  ```
  sp_vol____0.00.scf::ENE  : ********** TOTAL ENERGY IN Ry =     -1707.62757424
  sp_vol___10.00.scf::ENE  : ********** TOTAL ENERGY IN Ry =     -1707.61963274
  sp_vol__-10.00.scf::ENE  : ********** TOTAL ENERGY IN Ry =     -1707.62813765
  sp_vol____5.00.scf::ENE  : ********** TOTAL ENERGY IN Ry =     -1707.62431965
  sp_vol___-5.00.scf::ENE  : ********** TOTAL ENERGY IN Ry =     -1707.62900677
  ```

scfmonitor_lapw (analyse_lapw) [-h]

- scfmonitor –i 20 case.scf :ENE :DIS :FGL001 :FGL042



wien2k at the command line

- mkdir case ; cd case
- makestruct or cp ~/Downloads/xxx.cif case.cif ; x cif2struct
- setrmt …
- xcrysden --wien_struct case.struct
- init_lapw –prec 2n ; [edit .machines]
- run_lapw … [–p]
- grep :ENE case.scf # or other parameters of interest for convergence
- save_lapw exp_prec2n or continue
- run_lapw –NI –ec 0.000001 –cc 0.00001 ; save_lapw ….
  - *do properties like DOS, bands, density plots, optics, xspec or*
  - *optimize positions: run_lapw –min …*
- … do something else (e.g. use a different XC functional or volume)
- …
- restore_lapw exp_prec2n (restores case.struct, case.in*, density and pot)
  - *do properties like DOS, bands, density plots, optics, xspec (you have to recreate case.vector using x lapw1)*

Structural optimization of internal parameters using “**MSR1a**”

- run_lapw –fc 10   # crude convergence
- run_lapw **–min** -fc 1.0 -cc 0.001 -ec 0.0001 [-it -noHinv -p]
- generates case.inM and modifies case.inm and sets „**MSR1a**“

- This runs ONE big scf-calculations optimizing the density and the positions (forces towards zero) simultaneously (may need hundreds of iterations).

- Monitor: :ENE   :FR (av. and max forces, movements) and :APOSxxx

- it continues until all :FR quantities are below „tolf1/2“ (first and third parameter in case.inM) and switches then automatically to MSR1 for a final charge optimization (with fixed positions).
  - *For perfect equillibrium tolf1/2 should be reduced from 2 to 0.5 (0.1 for phonon)*

- quite efficient, **recommended** method, still under development by L.Marks (Northwestern Univ).

Structural optimization of internal parameters using “PORT”

- /home/pblaha/tio2> min_lapw [-p -it -sp] [-j “run -fc 1 -p -it”] [-NI]
  - *performs scf-cycle for fixed positions*
  - *get forces and move atoms along forces (building an approximate Hessian) and writing a new case.struct file*
  - *extrapolate density (case.clmsum)*
  - *perform next scf cycle and loop until forces are below „tolf"*
  - *CONTROL FILES:*
    - .minstop   stop after next structure change
- tio2.inM (generated automatically by “pairhess” at first call of min_lapw)
    - PORT 2.0 0.35 2.0 # NEW1,PORT, MOLD, **tol-force** trust-r **tol-movement**
    - 0.0 1.0 1.0 1.0   # Atom1 (0 will **constrain** a coordinate)
    - 1.0 1.0 1.0 1.0   # Atom2   (NEW1: 1,2,3:delta_i, 4:eta (1=MOLD, damping))
- **monitor minimization in file   case.scf_mini**
  - *contains **last** iteration of each **geometry** step*
  - *each step N is saved as case_N.scf (overwritten with next min_lapw !)*
    - grep :ENE case.scf_mini
    - grep :FGLxxx case.scf_mini   (:POSxxx)

DOS

- x kgen   # optional, but usually you should use a better k-mesh
- x lapw1   # optional Emax in case.in1, create $SCRATCH/case.vector
- x lapw2 -qtl   # create partial charges in   case.qtl
- x tetra   or   # creates default   case.int
- configure_int [–b total 1 tot,d-eg,dt2g 2 tot,s,p] (max 21 cases)
- edit case.int   # select a good E-range for DOS
- x tetra   # create case.dosX, case.dosXev (X=1/2/3)
- dosplot2   # up to 4 PDOS in one plot   or
- import case.dosXev (X=1,2,…) into your favored xy-plotting program (gnuplot, xmgrace,   Excel, Origin)
- x rendos   # optional, to get a renormalized DOS (no interstital). It requires PDOS of all atoms and all „chemical states“ (eg.: Ti-3d,4s,4p; O-2s,2p)
- dosplot2 -ren

electron density plots

- total and even total valence density usually „useless“
- remove semicore states
  - *check out where the semicore states are (case.outputst, case.scf1, case.scf2) and select an EMIN to cut them off*
  - *x lapw2 -emin XX # provided you still have a valid case.vector*
  - *xcrysden --wien_density case.struct*
    - (generates case.in5; x lapw5; create plot)
    - (alternative: x lapw5 (creates default case.in5); edit case.in5; x lapw5)
- difference density: $\rho^{val} - \rho^{atoms}$
  - *x lstart -sigma # create atomic valence densites*
  - *x lapw5 -diff*
  - *xcrysden --wien_renderdensity case.struct*
- for other plotting options (potentials, …):
  - *x lapw5 -h*

bandstructure plots

- create a k-mesh along a certain path in the BZ
  - *cp $WIENROOT/SRC_templates/xxx.klist case.klist_band*
    xxx: simple_cubic/fcc/bcc/hcp *or*
  - *xcrysden --wien_kpath case.struct*
- x lapw1 –band
  - *optional: x lapw2 -band -qtl # for fat bands and/or*
  - *x irrep # for bands with proper crossing*
- x spaghetti # creates case.insp
- edit case.insp: # set proper EF from scf file; energy range; fat bands
- x spaghetti # generates case.spaghetti_ps and case.bands.agr
- gv case.spaghetti_ps or xmgrace case.bands.agr

- After spaghetti you have to rerun x lapw1 before other properties

understanding the many files

- the many WIEN2k programs communicate via files
- Input/output/scf files have endings as the corresponding programs:
  - *case.output1…lapw1; case.in2…lapw2; case.scf0…lapw0*
- lsi # list all input files
- lso; lsc; lss; lsd, lse # output, clm, scf, def, error-files
- ls –als # produces a list with „information"
- which files are created by a program?
  - *x tetra: creates tetra.def and executes tetra tetra.def*
- ls –alsrt # lists the files according the creation date. The last ones are created by tetra (check time stamp)
- „def-files": contain the connection between unit-numbers and filenames. Tells you which files a program reads/writes