

ecalj MainDocument

This is a MainDocument of ecaljdoc. All files in ecaljdoc are linked from this file.

- Here we give [GetStarted](#) and [UsageDetailed](#), followed by install and Overview section.
- [Qiita Japanese](#) may be a help, but most of all are here.

Licence

[AGPLv3](#). For publications, we hope to make a citation as;

[1] ecalj package available from <https://github.com/tkotani/ecalj/>.

Install

To install ecalj, look into [install](#), as well as [install for ISSP](#)

Features of ecalj package

1. All electron full-potential PMT method

The PMT method means; a mixed basis method of two kinds of augmented waves, that is, APW+MTO.

In other words, the PMT method= the linearized (APW+MTO) method, which is unique except the [Questaal](#) having the same origin with ecalj. Our recent research shows that very localized MTOs (damping factor $\exp(-\kappa r)$ where $\kappa \sim 1$ a.u), together with APW (cutoff is ≈ 3 Ry) works well to get reasonable convergences. We can perform atomic-position relaxation at GGA/LDA level. Because of including APWs, we can describe the scattering states very well.

The current PMT formulation is given in

[1][KotaniKinoAkai2015, PMT formalism](#)

[2][KotaniKino2013, PMT applied to diatomic molecules](#).

Since we have automatic settings for basis-set parameters, we don't need to be bothered with the parameter settings. Just crystal structures (POSCAR) are needed for calculations.

2. PMT-QSGW method

The PMT-QSGW means

the Quasiparticle self-consistent GW method (QSGW) based on the PMT method.

After converged, we can easily make band plots without the Wannier interpolation. This is because an interpolation scheme of self-energy is internally built in.

We can handle even metals, Fermi surface as well. Since we have implemented ecalj on GPU, we can handle ~40 atoms with four GPUs.

[3][Kotani2014,Formulation of PMT-QSGW method](#)

[4][PMT-QSGW applied to a variety of insulators](#)

[5][Obata GPU implementation](#)

3. Dielectric functions and magnetic susceptibilities

We can calculate GW-related quantities such as dielectric functions, spectrum function of the Green's functions, Magnetic fluctuation, and so on.

4. The Model Hamiltonian with Wannier functions

We can generate the effective model (Maxloc Wannier and effective interaction between Wannier functions).

This is originally from codes by Dr.Miyake, Dr.Sakuma, and Dr.Kino. The cRPA given by Juelich group is implemented. We are now replacing this with a new version MLO (Muffin-Tin-orbital-based localized orbital).

Overview of QSGW

- Band calculations (LDA level) are performed with the program `lmf`. The initial setting file is `ctrl.foobar` (`foobar` is user-defined). Before running `lmf`, it is necessary to run `lmfa`, which is a spherically symmetric atom calculation to determine the initial conditions for the electron density (`lmfa` finishes instantaneously).
- A file `sigm.foobar` is the key for QSGW calculations. The file `sigm.foobar` contains the non-local potential $V_{\text{xc}}^{\text{QSGW}} - V_{\text{xc}}^{\text{LDA}}$. By adding this potential term to the usual LDA calculation performed by `lmf`, we can perform QSGW calculations.
- Thus the problem is how to generate $V_{\text{xc}}^{\text{QSGW}}(\mathbf{r}, \mathbf{r}')$. This is calculated from the self-energy $\Sigma(\mathbf{r}, \mathbf{r}', \omega)$, which is calculated in the GW approximation. Roughly speaking, we obtain $V_{\text{xc}}^{\text{QSGW}}(\mathbf{r}, \mathbf{r}')$ with removing the omega-dependence in $\Sigma(\mathbf{r}, \mathbf{r}', \omega)$.
- Therefore, the calculation of $V_{\text{xc}}^{\text{QSGW}}$ is the major part of the QSGW cycle, and is calculated in a double-structure loop. That is, there is an inner loop of `lmf`, and an outer loop to calculate $V_{\text{xc}}^{\text{QSGW}}$ using the eigenfunctions given by `lmf`. This outer loop can be executed with a python script called `gwsc` (which runs fortran programs). The computational time for QSGW is much longer than that of LDA calculation. As a guideline, it takes about 10 hours for 20 atoms (depending on the number of electrons). We see the QSGW cycle in Figure 1 in <https://arxiv.org/abs/2506.03477>.
- We have [GPU acceleration for QSGW](#). Thus we can handle large systems. With 4 GPU, we can compute systems with 40 atoms per cell with surfaces. (As for `lmf` part, GPUs are not efficiently used yet.)
- As noted, we can perform QSGW virtually without parameter settings by hands. Thus I think `ecalj` is one of the easiest code to perform GW for users. See band database in QSGW at <https://github.com/tkotani/DOSnpSupplement/blob/main/bandpng.md>

(this is a supplement of <https://arxiv.org/abs/2507.19189>). This is away from complete one, but showing the ability of ecalj.

GetStarted

Here we explain DFT/QSGW calculations with ecalj. Then we explain how to make band plots. For simplicity, we treat paramagnetic cases (nsp=1), no 4f, no SOC.

We explain things step by step.

Further details are explained at [UsageDetailed](#).

Step 0. Get POSCAR

We first need POSCAR (crystal structure in VASP format).

You can find samples of POSCAR in ecalj/ecalj_auto/INPUT/testSGA/POSCARALL as

```
cd ecalj
mkdir TEST
cd TEST
mkdir test1
mkdir test2
cat ecalj_auto/INPUT/testSGA/joblist.bk
cp ../ecalj_auto/INPUT/testSGA/POSCARALL/POSCAR.mp-2534 test1
cp ../ecalj_auto/INPUT/testSGA/POSCARALL/POSCAR.mp-8062 test2
```

For example, POSCAR of mp-2534 GaAs is given as:

```
Ga1 As1
1.0
  3.52125300000000002    0.00000000000000000    2.03299699999999999
  1.17375100000000000    3.31986900000000002    2.03299699999999999
  0.00000000000000000    0.00000000000000000    4.06599299999999997
Ga As
1 1
direct
  0.00000000000000000    0.00000000000000000    0.00000000000000000 Ga
  0.25000000000000000    0.25000000000000000    0.25000000000000000 As
```

This is another POSCAR for ba2pdo2cl2 (QSGW results are shown below):

```
POSCAR_ba2pdo2cl2
1.0
-2.06443 2.06443 8.40383
2.06443 -2.06443 8.40383
```

```

2.06443 2.06443 -8.40383
Ba Pd O Cl
2 1 2 2
Cartesian
0.0 0.0 6.5153213224
0.0 0.0 10.2923386776
0.0 0.0 0.0
0.0 2.06443 0.0
2.06443 0.0 0.0
0.0 0.0 3.1625293056
0.0 0.0 13.6451306944

```

If you have cif and like to convert it to **POSCAR**, do

cif2cell foobar.cif -p vasp --vasp-cartesian --vasp-format=5.

Step 1. convert POSCAR to ctrl

Then we convert POSCAR to ctrl by vasp2ctrl.

ctrls is the structure file used in ecalj.

```

vasp2ctrl POSCAR.mp-2534
mv ctrls.POSCAR.mp-2534.vasp2ctrl ctrls.POSCAR.mp-2534
cat ctrls.mp-2534

```

ctrls.mp-2534 contains crystal structure equivalent to POSCAR:

```

cat ctrls.mp-2534
STRUC
      ALAT=1.889726877743552
      PLAT=      3.52125300000      0.00000000000      2.03299700000
                1.17375100000      3.31986900000      2.03299700000
                0.00000000000      0.00000000000      4.06599300000
      NBAS=2
      SITE
      ATOM=Ga POS=      0.00000000000      0.00000000000
0.00000000000
      ATOM=As POS=      1.17375100000      0.82996725000
2.03299675000

```

- MEMO:
 - ctrl2vasp ctrl.mp-2534 can convert back to VASP file. Check this by VESTA. We can use viewvesta (convert and invoke VESTA).
 - many unused files are generated (forget them).

Step 2. Get ctrl from ctrls

ctrl is a basis input file for ecalj. We generate template of ctrl by ctrlgenM1.py.

Minimum explanations are embedded in the generated ctrl file.

Number of k points (nk1 nk2 nk3), APW cutoff (pwemax), nspin, so(spin orbit switch) are only what we need to tweak usually.

When we run lmf, we can add command line option such as -vnspin=2. Then const foobar=1 defined in the ctrl file is overridden (referred with {foobar}). save.* file show which -vfoobar you used.

It is possible to enforce symmetry, antiferro symmetry.

We only need ctrl file in the following calculations (while some tmp* kinds of files are generated).

```
ctrlgenM1.py mp-2534
```

If no problem, you see

```
...  
=== End of ctrlgenM1.py. OK! A template of ctrl file, ctrlgenM1.ctrl.mp-  
2534, is generated.
```

Here `ctrlgenM1.py` internally calls `lmf` and `lmchk`, which generate irrelevant files which are automatically deleted.

'SiteInfo.lmchk and PlatQlat.chk' are explained later on (these are easily reproduced by ctrl).

Then copy as

```
cp ctrlgenM1.ctrl.mp-2534 ctrl.mp-2534
```

and edit `ctrl.foobar` if necessary.

How to edit? Explanations are embedded in ctrl.foobar (please let me know wrong descriptions). Possible points to rewrite in ctrl.foobar:

1. Number of k points (nk1,nk2,nk3).
2. nsp=2 if magnetic
3. SpinOrbitCoupling: so=0 (none), so=1 (LdotS), 2 (LzSz). nsp=2 is required for so=1,2. so=1 does not yet support QSGW. SOC axis can also be freely selected, but currently (0,0,1) default and (1,1,0) are supported (m_augmb1.f90). If you want to set SO=1 in QSGW, currently, run QSGW calculation with so=0 or so=2 to obtain ssg file, then set so=1
4. xcfun (choice of LDA exchange correlation term). Only =1:BH, =2:VWN, =103:PBE-GGA.
5. LDA+U settings (not explained yet).
6. ssg=1.0 (If you choose QSGW80, use ssg=0.8. Effective for QSGW calculations.
 $V^{\text{xc QSGW}} - V^{\text{xc LDA}}$ is stored in a file `sigm.foobar`. We add ssg $\times (V^{\text{xc QSGW}} - V^{\text{xc LDA}})$

xc QSGW}-V^{\rm xc LDA})\$ to the potential in the lmf calculation as long as **sigm.foobar** file is available.

- **lmchk --pr60 foobar** allows you to check the recognized symmetries by **lmf**. Turning off --pr60 or reducing 60 will reduce the verbosity of output.

At this point, you can visually check the following check files.

- SiteInfo.chk
MT radius Atomic positions
- PlatQlat.chk
Primitive lattice vector (plat) Primitive reciprocal lattice vector (qlat)

[Here we explain details of ctrl file.](#)

Hereafter, we only use **ctrl.foobar (**ctrls.foobar** is used hereafter.). We can delete other files.**

Install VEST

It is convenient to see crystal structures with VESTA.

(I installed VESTA-gtk3.tar.bz2 (ver. 3.5.8, built on Aug 11 2022, 23.8MB) on ubuntu 24)

At ecalj/StructureTool/, we have 'viewvesta' command. Try

```
viewvesta ctrl.si
```

to see the structure in VESTA. At /StructureTool, we have converters,

vasp2ctrl and **ctrl2vasp**.

(We have ~/ecalj/GetSyml/README.org. Not need to see this)

Step 3. LDA calculation

1. Run **lmfa** at first. It is for spherical atomic electron densities, contained in the crystals. **lmfa** ends instantaneously.

```
lmfa ctrl.mp-2534
```

gives spherical atom calculation for initialization. **lmfa** calculates spherically symmetric atoms and generates the files required for **lmf** below.

Check **conf** section in the console output as

```
lmfa ctrl.mp-2534 |grep conf
```

. This shows atomic configuration (there are no side effects even if **lmfa** is repeated). The initial condition of electron density for **lmf** is given as the superposition of spherically symmetric atomic

densities given by `lmfa`. In addition, `lmfa` calculations are performed with the logarithmic derivative of the radial wave function at the MT sphere edge fixed (`READP True` in default `ctrlgenM1.py` setting). The derivatives are contained in `atmpnu.*` files. So, `atmpnu.*` are needed for `lmf`.

2. After `lmfa`, we run LDA calculation as:

```
mpirun -np 8 lmf mp-2534 |tee l1mf
```

- `mp-2534` (GaAs) gives 5.75 eV for GaAs, while the experimental value is 5.65 eV.
- `l1mf` contains information of iterations, check eigenvalue and fermi energies, band gap.
- `rst.mp-2534` is generated. Self-consistent charge included.
- You can change lattice constant as $ALAT=1.889726877743552 \times 5.65/5.75$ in `ctrl` file. simple math operators such as `*`, `+`, `-`, `/`, `**` can be possible in `ctrl`.
- Note: `ctrlp` is intermediate file generated by python from `ctrl`. Fortran calls a python code internally. (`ctrl2ctrlp.py` is responsible for the math)
- check `save.mp-2534`. Show history of `lmfa` and `lmf`. one line per iteration. Show your console options. `c,x,i,h`
LDA energy shown two values need to be the same (but slight difference).
Repeat `lmf` stops with two iteration.
- `SiteInfo.lmchk` : Site info
- `PlatQlat.chk` : Lattice info
- `estaticpot.dat` : electrostatic potential of smooth part.

NOTE:

We have deguchi paper <https://sci-hub.tw/https://doi.org/10.7567/JJAP.55.051201>

All calculation is by the default setting in QSGW on the PMT method.

No empty spheres. $EH=-1, EH=-2$, MT radius is -3% untouching.
 $RSMH=RSMH2=R/2$

PROF

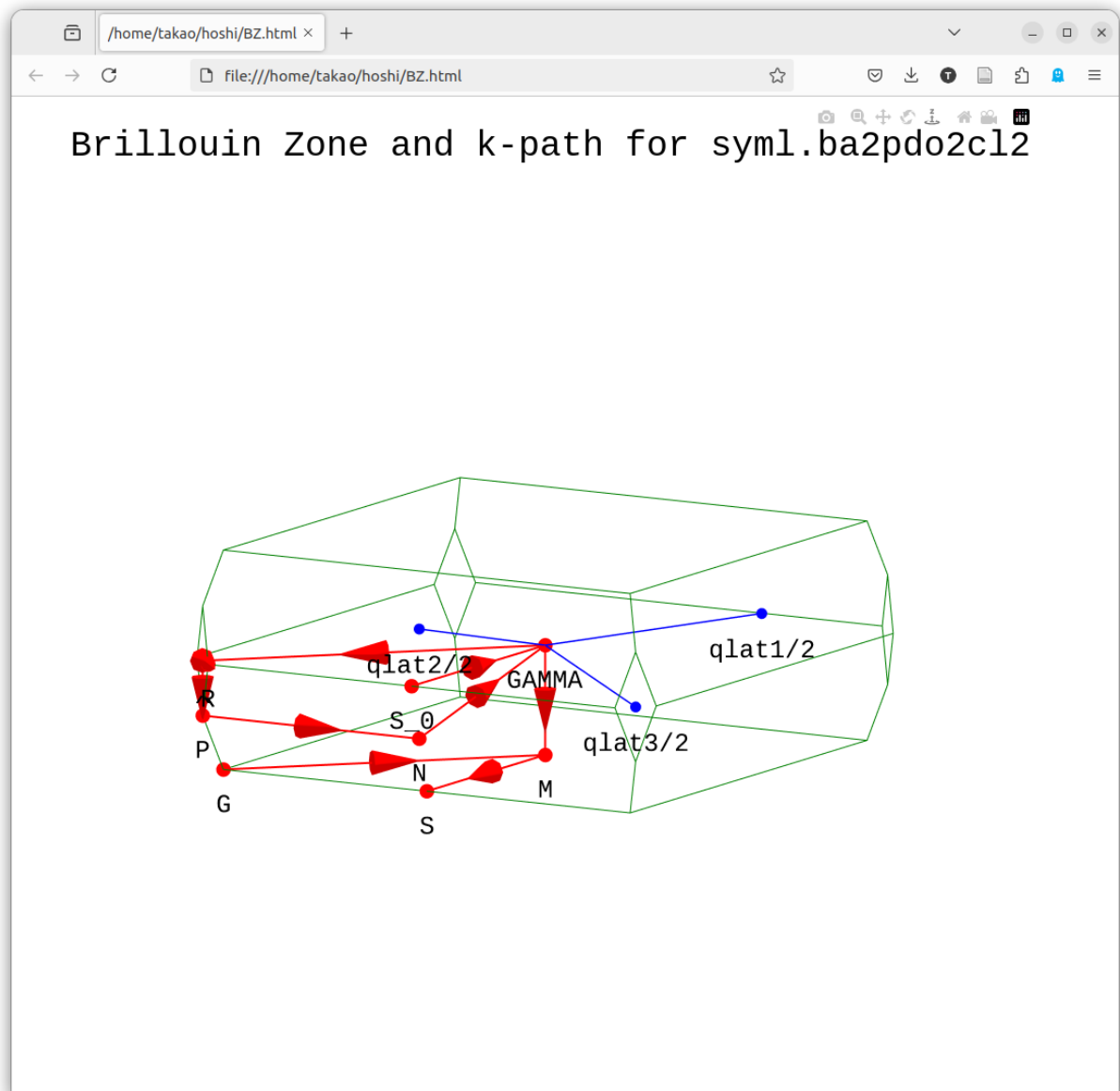
Step 4. Create k-path and BZ for band plot

After the calculation converges, it might be necessary to make a band plot with `job_band` command explain later on. The normality of the calculation of bands can be confirmed by the band plot (for magnetic systems, check the total magnetic moment and the magnetic moment for each site).

Before `job_band`, run `getsym1 gaas`. Install any missing packages with pip. It is on spglib by Togo and seekpath. After finished, view `BZ.html`. It shows the k-path in the BZ as show below for `ba2pdo2cl2`. It is an interactive figure written with plotly, so you can read the coordinate values.

```
getsym1 mp-2534
```

- Samples of BZ.html by getsym1 are seen at <https://ecalj.sakura.ne.jp/BZ/>.



Step 5. band plot

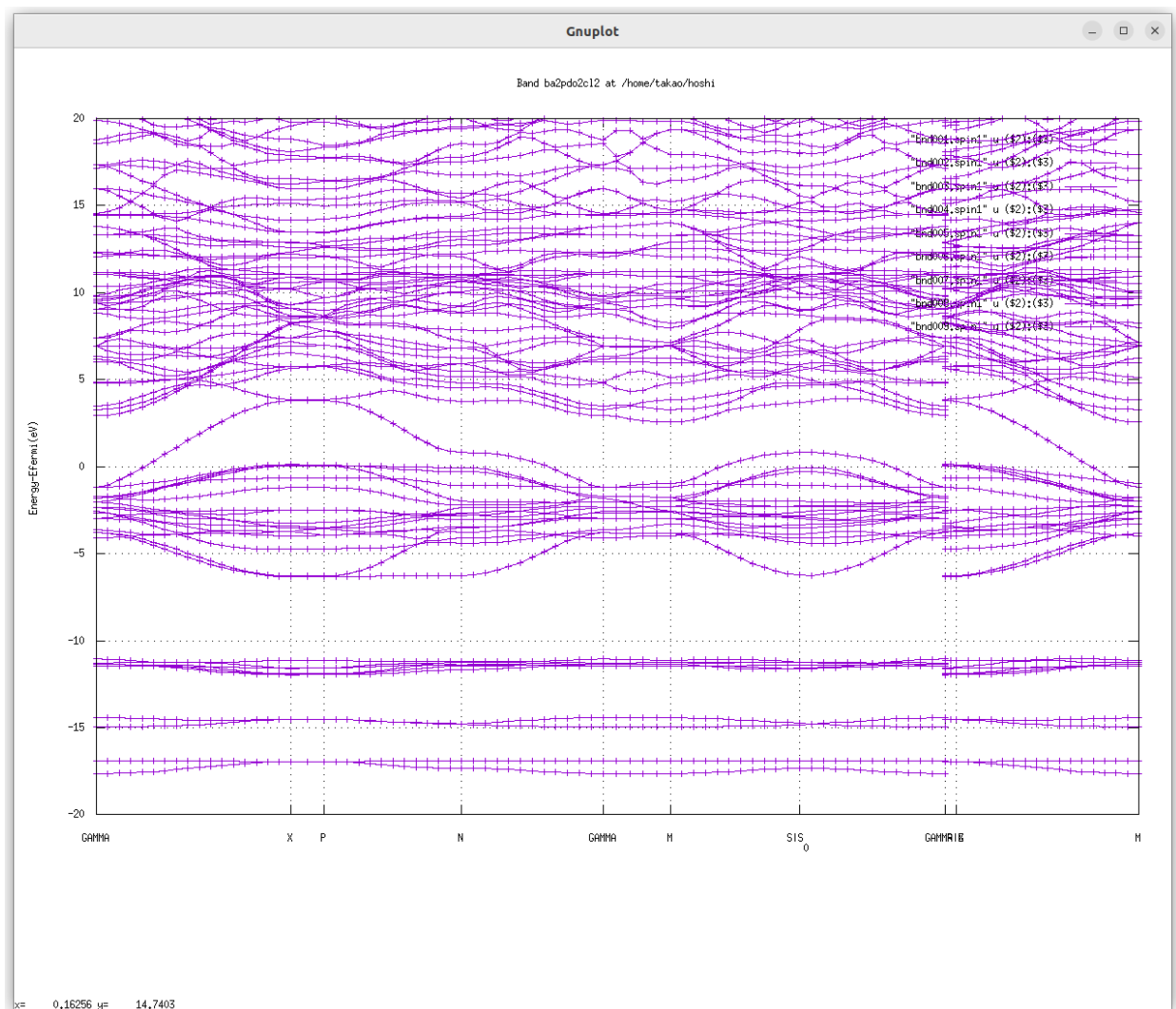
PROF

(this is a case for ba2pdo2cl2)

```
>job_band ctrl.ba2pdo2cl2 -np 8
```

A gnuplot script can be created. Edit it if necessary. If you edit sym1.ba2pdo2cl2 before `job_band`, you can adjust the symmetry line and mesh size.

- The following picture is the LDA bands for the default calculation of ba2pdo2cl2 (the names of the symmetric points can be confirmed with BZ.html. In addition, look into `sym1.foobar`). 0 eV is the Fermi energy. Since this is metallic, we see no band gap.



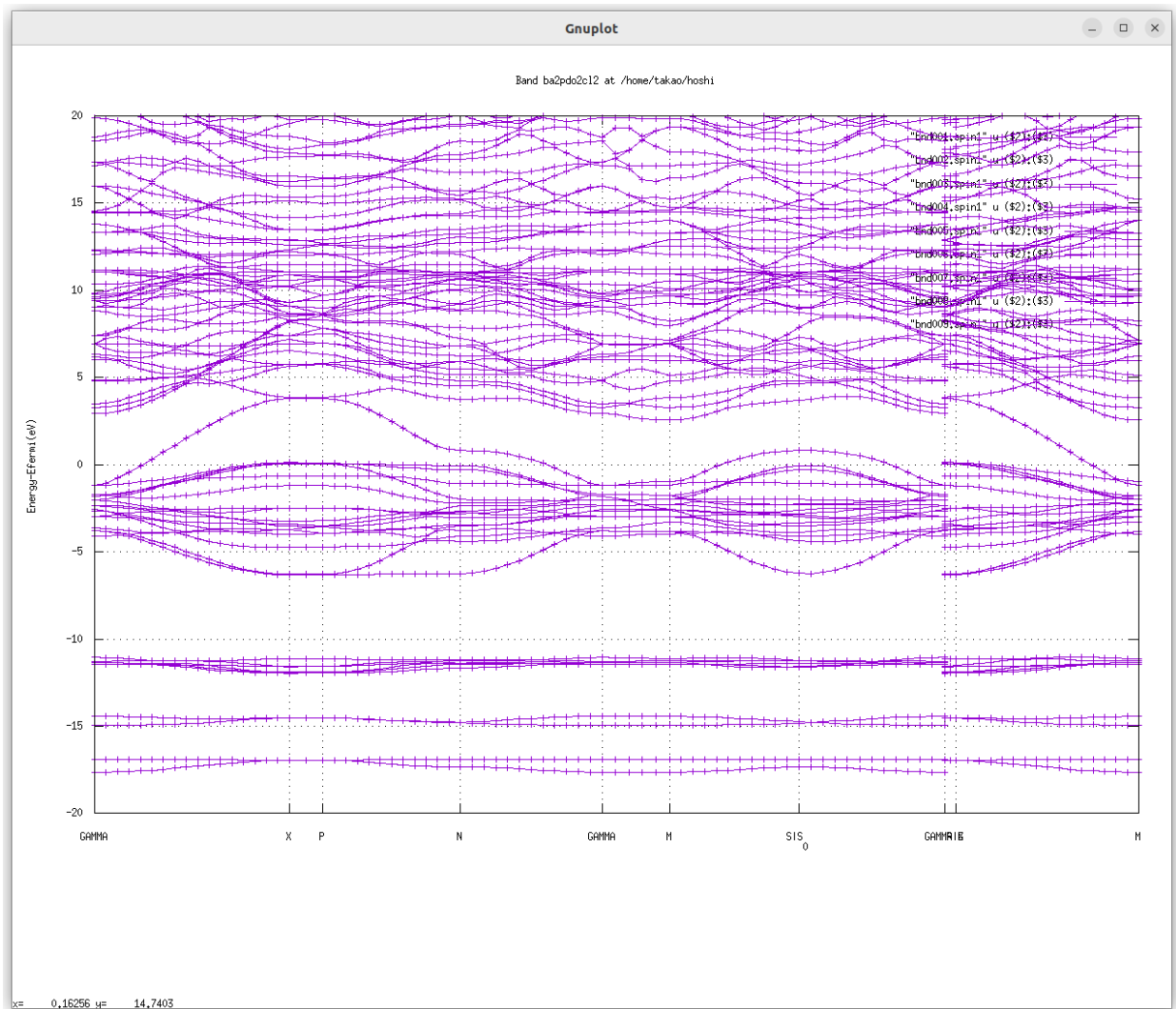
- The defaults are fine except for the k-mesh setting. For example, it is better to increase the k mesh for Fe. In general, for semiconductors, 4 4 4 for Si is a reasonable level, 6 6 6 is a level that can be used for a paper, and 8 8 8 is a level for checking accuracy. For metals such as Fe, 8 8 8 is a reasonable level.

PROF

Here we are talking about band energies.

- In ecalj, the k mesh for **lmf** (ctrl) and the k mesh for GW (n1n2n3 specified in GWinput) can be different. The former has affected little on computational time, but the latter has a large effect (thus we want to reduce **n1n2n3** in GWinput).
- In ecalj's band plot mode, theoretically degenerated bands because of symmetry at the BZ edge are not degenerated. This is because there are limited numbers of APW basis functions, so run the

band plot with pwemax=4, etc. (Temporary solution: We want to automate it).



`job_tdos`, `job_fermisurface`, `job_pdos`

`job_pdos` calculates PDOS, `job_tdos` calculates total DOS, and `job_fermisurface` draws the Fermi surface with Xcrysden.

`job_fermisurface` can be used to draw the shape of the CBM bottom as ellipsoid of Si.

PROF

XXX

Step 6. QSGW calculation

We now run QSGW calculations. qSGW is computationally very expensive. So we recommend you to run smaller systems at first.

For QSGW calculations, we need one additional input file `GWinput`, whose template is generated by `mkGWinput` `GWinput` as

```
mkGWinput ctrl.mp-2534
```

Then copy and edit `GWinput.tmp` to `GWinput`.

In **GWinput**, $n_1 n_2 n_3$ should be smaller than $n_k1\ n_k2\ n_k3$ in ctrl file
in order to reduce computational time (1/2 or 2/3 of ctrl, for example)
If 6x6x6 for Si, it is reasonable. Except k points, not need to modify so much (ask us).
GWinput is explained [here](#). Input system is different from ctrl.

flow of QSGW calculation with the script gwsc

We run the QSGW calculations with gwsc. For semiconductors, several QSGW iterations are fine, close enough to final results.

QSGW is to obtain band structures (or one-body Hamiltonian), the total energy is not yet.

QPU file contains diagonal components of GW calculations.

Note that our **Mixed Produce basis** is a key technology for the GW calculation.

```
gwsc -np NP [--phispinsym] [--gpu] [--mp] nloop extension
```

(--phispinsym is for magnetic materials to keep the same basis for up and down)

Then console outputs of **gwsc** is something like

```
### START gwsc: ITERADD= 1, MPI size= 4, 4 TARGET= si
===== Ititial band structure =====
--> No sigm. LDA caculation for eigenfunctions
0:00:00.226245 mpirun -np 1 /home/takao/bin/lmfa si >llmfa
0:00:00.807062 mpirun -np 4 /home/takao/bin/lmf si >llmf_lda
===== QSGW iteration start iter 1 ===
0:00:03.071054 mpirun -np 1 /home/takao/bin/lmf si --jobgw=0
>llmfgw00
0:00:03.904403 mpirun -np 1 /home/takao/bin/qg4gw --job=1 > lqg4gw
0:00:04.431022 mpirun -np 4 /home/takao/bin/lmf si --jobgw=1
>llmfgw01
0:00:05.918216 mpirun -np 1 /home/takao/bin/heftet --job=1 > leftet
0:00:06.444439 mpirun -np 1 /home/takao/bin/hbasfp0 --job=3 >lbasC
0:00:07.064558 mpirun -np 4 /home/takao/bin/hvccfp0 --job=3 > lvccC
0:00:07.812283 mpirun -np 4 /home/takao/bin/hsfp0_sc --job=3 >lsxC
0:00:08.545956 mpirun -np 1 /home/takao/bin/hbasfp0 --job=0 > lbas
0:00:09.156775 mpirun -np 4 /home/takao/bin/hvccfp0 --job=0 > lvcc
0:00:09.884064 mpirun -np 4 /home/takao/bin/hsfp0_sc --job=1 >lsx
0:00:10.644292 mpirun -np 4 /home/takao/bin/hrcxq > lrcxq
0:00:11.482931 mpirun -np 4 /home/takao/bin/hsfp0_sc --job=2 > lsc
0:00:12.460776 mpirun -np 1 /home/takao/bin/hqpe_sc > lqpe
0:00:13.019735 mpirun -np 4 /home/takao/bin/lmf si >llmf
===== QSGW iteration end iter 1 ===
OK! ===== All calclation finished for gwsc =====
```

The log files of console outputs are `l*`. `C` at the end of the log file means Core-related parts. `lsx` is the exchange self-energy due to cores.

`lsx` is for exchange. `lsc` is correlation. `lvcc` is for Coulomb matrix.

In this calculation we run `gwsc -np 8 1 si`, where 1 is the number of QSGW iteration.

If you repeat `gwsc`, we have additional QSGW iterations on top the previous calculations.

Here is a case of `ba2pdo2cl2`.

Run

```
mkGWinput ba2pdo2cl2
```

to generate `GWinput.tmp`, which is a setting file for QSGW.

After copying this to `GWinput`, you may need to edit `GWinput`.

Minimum thing to edit is the number of `k` points for the self energy (`n1n2n3`).

Compared with `k` points in `ctrl` (`nk1,nk2,nk3`), we use small numbers.

(We often use 1/2 or 2/3 of `k` points given in `ctrl` as `nk1,nk2,nk3`).

There are another setting in `GWinput`. However, we usually do not need to touch things except `n1n2n3` if you treat non-magnetic semiconductors.

Then you can run QSGW calculation with

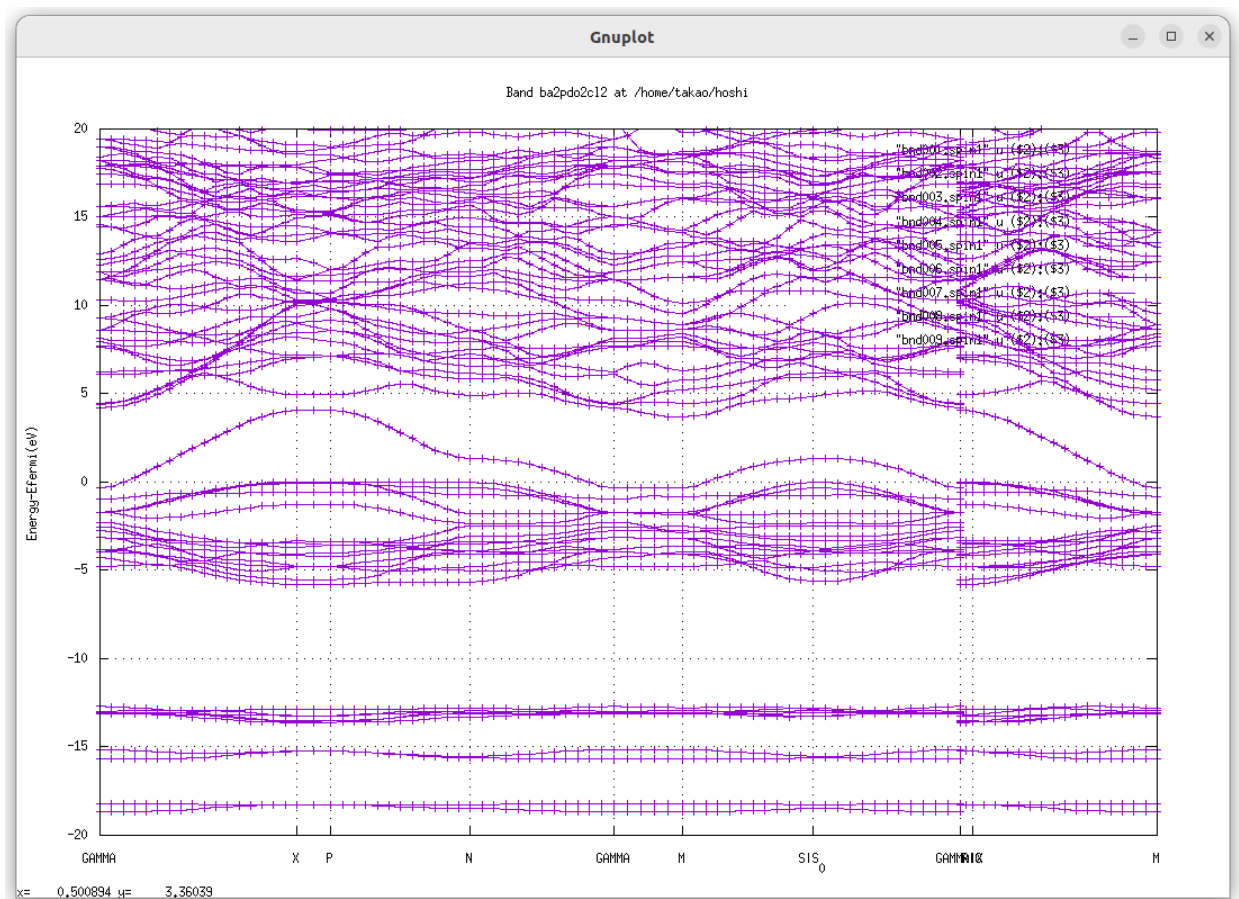
```
gwsc -np 32 1 ba2pdo2cl2
```

. Here 1 means the number of QSGW iterations. QSGW iteration is quite time-consuming. `gwsc` gives minimum help (we need to explain options elsewhere).

The iteration is kept in `rst.foobar`:electron density, `sigm.*:vxcqsgw`.

(Remove these files in addition to `*run` files/directories if you like to start from the beginning).

- It requires 53 minutes to run one iteration of QSGW.
- `job_band ba2pdo2cl2 -np 32` gives the following picture. QSGW one-shot changes band structure around E_f from that in LDA. But still metallic, no band gaps.

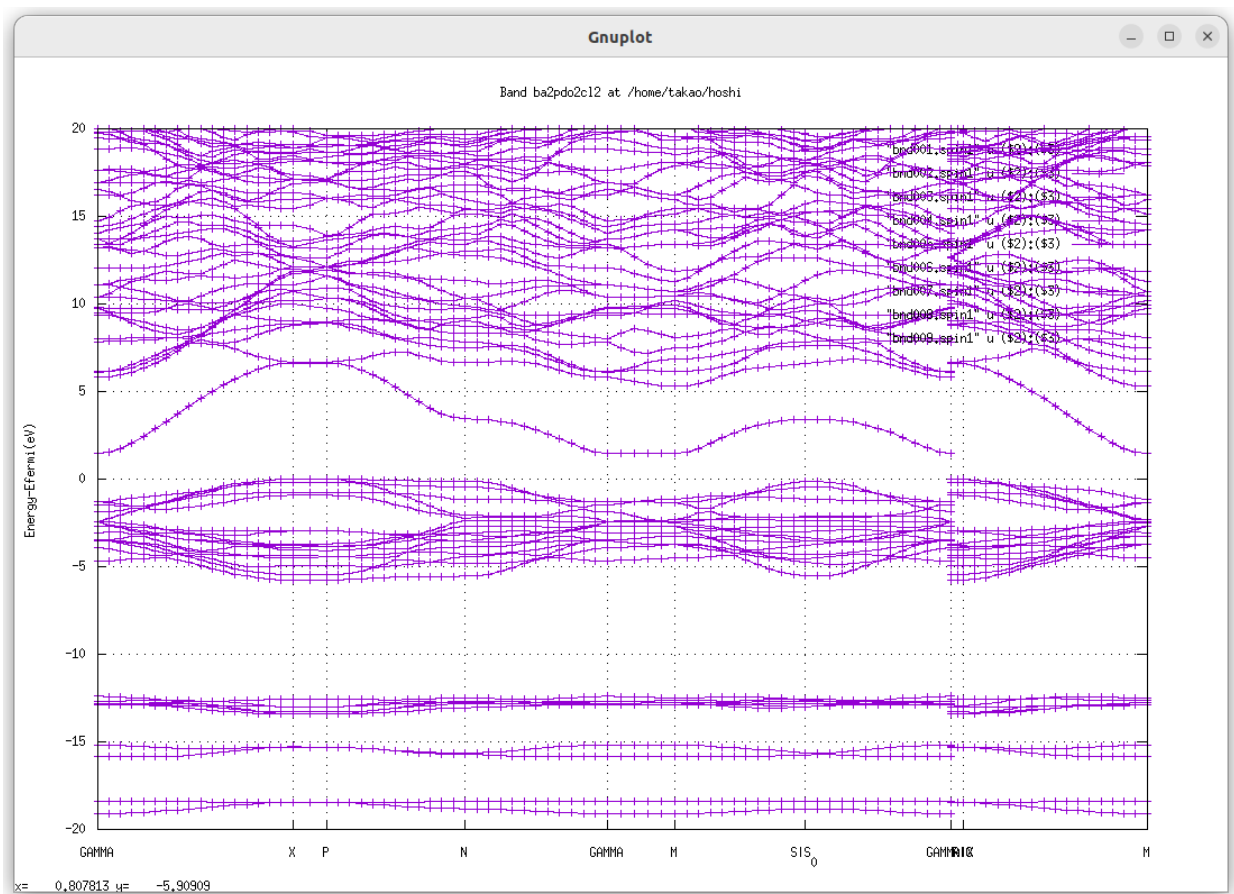


- To continue QSGW iteration, run

```
gwsc -np 32 nx ba2pdo2c12
```

Since you did 1 already. You will have the results of 1+nx QSGQ iteration.

- when we run 8 iterations as for ba2pdo2c12, we had band gap 2.1 eV. We saw band gap after 4th iteration.

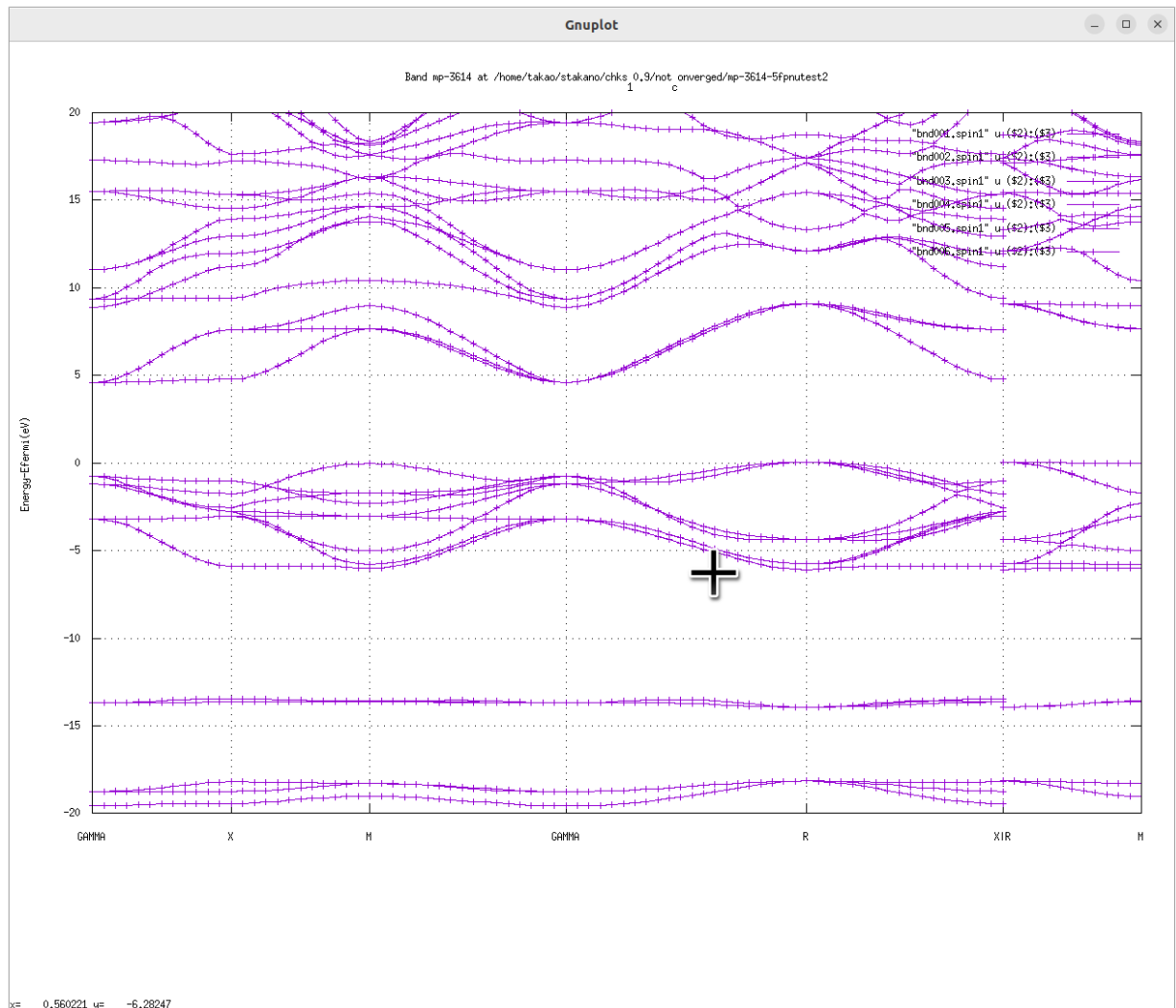


- For comparison with experiments, we recommend to use `ssig=0.8` (set in `ctrl.foobar` file), which is called as QSGW80.
- Spin-orbit coupling. After you obtain `sigm.foobar`, you set `SO=1` (LdotS scheme) and run `lmf`. Then you can include effect of SOC. Since `SO=1` is not implemented in the whole `gwsc` cycle, we have to include SOC just at the end step (We include SOC after we fix `VxcQSGW`).
- If you run

```
gwsc -np 32 5 ba2pdo2cl2 -vssig=0.8
```

, this override `ssig`, which is defined in `ctrl.ba2pdo2cl2`, in `lmf` calculations. (Check it in `save.ba2pdo2cl2`)

- Example of QSGW for KTaO3 (perovskite, mp-3614)



lmchk

lmchk mp-2534

PROF

is to check the crystal symmetry. In addition determine MT radius. and Check the overlap of MTs. Defaults setting is with -3% overlap.(no overlap).

- symmetry
- MT overlap

If you have less symmetry rather than the symmetry of lattice for magnetic systems, you have to set crystal symmetry by hand.

This can be done by adding space group symmetry generator to SYMGRP (instead of find).

We need to pay attention for this point in the case of SOC.

(we have to explain how to read space group and so on.)

How to start over calculations

Remove mix* rst* (mix* is mixing files)

If MT changes, start over from lmfa (remove atm* files)

- As long as converged, no problem.
- If you have 3d spaghetti bands at E_f , need caution.

A mini database for tests.

At `ecalj/MATERIALS/`, you can run `./jobmaterials.py`. It shows a help with a list of materials. It contains samples of simple materials. It performs LDA calculations and generates GWinput for materials.

Run as follows. Then we perform LDA calculations where crystal structures are already contained in a mini database.

```
./job_materials.py
```

gives a help, showing a list of materials. Then

```
./job_materials.py Si
```

performs LDA calculation of Si at `ecalj/MATERIALS/Si/`. '--all' works as well instead of 'Si'.

`job_materials.py` works as follows for given names.

Step 1. Generate `ctrls.*` file for `Materials.ctrls.database`. (names are in `DATASECTION:`)

Step 2. Generate ctrl by `ctrlgenM1.py`

Step 3. Make directory such as `Si/` and run `lmf`, `lmfa`, `mkGWinput`.

- Key input files are
`ctrls.si`, `ctrl.si`
. See sections below. `rst.si` contains self-consistent electron density. Check iterations with the output file `save.si`. The console output of `lmf` is in `llmf`. Not need to know all the console outputs.
- Before QSGW, it is better to confirm the LDA level calculations are fine. In order to do the confirmation, band plot is convenient.
For band plot we need the symmetry line as `sym1.si` which can be generated by

```
getsym1 si
```

Then run

```
job_band si -np 8
```


results band plots in the gnuplot.

UsageDetailed

console output

Console output is now mainly for debug purpose.

But we still need to read some of output data from the console output (we are trying to modify this).

From console output, we can check convergece behavior, band energies, Fermi energies, whether sigm, rst are correctly read in or not.

save.foobar

- Save file record starting history of lmf,lmchk,lmfa. In addition, it give a line of total energies per iteration of lmf. At each line,'i: intermediate, c: converged, x:iteration max without converged'.
- In addition, -vfoobar=xxx is recorded (overriding variables in ctrl).
- Two total energies Kohn-Sham and Harris-Folker is given---both should be virtually the same. But some differences for bigger systems. Take one of them.
- log file
log.foobar generated by lmf,lmchk,lmfa are currently used for debuggging purpose.

Spin polarized case without SOC

To treat magnetic systems, we have to set MMOM (ititial spin magnatic moments)

in addition to set nspin=2 in ctrl.foobar. The spin magnetic moments are specified as the difference of number of electrons between spin channels, isp=1 and isp=2.

For example, we set ctrl.nio for NiO as

```
SITE      ATOM=Niup POS=  .0    .0    .0
          ATOM=Nidn POS= 1.0    1.0    1.0
          ATOM=O   POS=  .5    .5    .5
          ATOM=O   POS= 1.5    1.5    1.5
SPEC
  ATOM=Niup Z=28 R=2.12
    MMOM=0 0 1.2 0
    EH=-1 -1 -1 -1 RSMH=1.06 1.06 1.06 1.06
    EH2=-2 -2 -2 RSMH2=1.06 1.06 1.06
    KMXA={kmtx} LMX=3 LMXA=4 NMCORE=1
  ATOM=Nidn Z=28 R=2.12
    MMOM=0 0 -1.2 0
    EH=-1 -1 -1 -1 RSMH=1.06 1.06 1.06 1.06
    EH2=-2 -2 -2 RSMH2=1.06 1.06 1.06
    KMXA={kmtx} LMX=3 LMXA=4 NMCORE=1
```

Here we have two sites named Niup and Nidn. `MMOM=0 0 1.2 0` means initial spin moment within MTs: that is `MMOM= {s} {p} {d} {f}`, where `{s} {p} {d} {f}` are number of spin moments for each \$l\$ at atomic sites.

Separators can be space or comma. In addition, we set `nspin=2` defined at the `% const` line in ctrl file. Calculated spin moments within MT are at 'true mm' column in the console output, and shown in the `mmom.nio.chk` as

```
# Qtrue      MagMom(up-dn) Rmt    MT
1 8.527587   1.200085  2.120000 Niup
2 8.527604  -1.200081  2.120000 Nidn
3 5.380352  -0.000002  1.700000 0
4 5.380352  -0.000002  1.700000 0
```

Note it is overwritten at every iteration. These are shown in console output as 'true mm' as well. Atomic site index are given in 'Siteinfo.chk'. Total Magnetic Moments are shown as

```
Magnetic moment=      2.241805 !this is a case of bulk Fe
```

orbital moments

When `so=1`, orbital moments within MTs are shown at `IORBTM:` in the console output as

```
IORBTM: orbital moments :
ibas Spec      spin  Moment decomposed by l ...
  1   Pr         1    0.000000  -0.011483  -0.010535  -4.881144
0.000234
  1   Pr         2    0.000000   0.012332   0.003604   0.000373
-0.000090
total orbital moment  1:  -4.886708
  2   N          1    0.000000  -0.004174   0.001584   0.000291
0.000129
  2   N          2    0.000000   0.004622   0.000236   0.000045
0.000006
total orbital moment  2:    0.002738
```

Antiferro symmetry without SOC

We can set Antiferro symmetry (not yet for SOC=1).

We have README_AF.md and samples at `~/ecalj/Samples/AFsymmetry/`

(README_mmtarget.aftest.txt showing the fixed moment method for antiferro symmetry need to be fixed).

Spin-orbit coupling

We have a switch **HAM_SO** in the ctrl file

- For LDA/GGA, set `nspin=2` and `so=1`. Then we can perform calculations including SOC. `so=1` is for soc included (`so=2` is for LzSz mode neglecting LxSz+LySy).
- In the case of semiconductors such as GaAs, we need to include `so=1` to see the band structure at the top of valence.
- Currently, QSGW can not be performed with `so=1`. So we first have to run `gwsc` with `so=0` or `2`. After we get `sigm` file, we run `lmf` with `--vso=1` (`nit=1` can be fine) as a perturbation.
- We can treat only colinear spins. Spin axis is along (0,0,1) as default. We can choose other direction with `SOCAXIS`. See `ecalj/Samples/SOCAXIS`. Not checked completely, but it seems work well.

Band plot with spin orbit coupling.

- method 1: only apply SOC for band plot

```
job_band mp-2534 -np 8 -vso=1 -vnspin=2
```

Caution: when you set `nspin=2`, the size of `rst` is twiced. No way to move it back to `rst` for `nspin=1`. So you may need to keep `rst`.

- method 2. single iteration and `SO=1`

```
mpirun -np 8 lmf -vso=1 -vnspin=2 -vnit=1
```

Then we have revised `rst.foo`. Then run **job_band mp-2534 -np 8 -vso=1 -vnspin=2**.

- method 3. full iteration `SO=1`

```
mpirun -np 8 lmf -vso=1 -vnspin=2 -vnit=1
```

Then run **job_band mp-2534 -np 8 -vso=1 -vnspin=2**

Forces and Atomic position relaxation

See `ecalj/Samples/LaGaO3_relax`.

We have to set **DYN** category. In addition, we can set directions for relaxation. No cell optimizations.

Fermi surface

See a sample at `ecalj/Samples/FermiSurface`

PROCAR mode

See a sample at [ecalj/Samples/MgO_PROCAR](#)

We can generate PROCAR file containing the size of eigenfuncitons**2.

The sample (run job file) generates eps file showing fat band of O2 components.

Run jobprocar. This gives *.eps file which shows Fat band picture.

PROCAR (vasp format) is generated and analysed by a script BandWeight.py.

LDA+U

We have samples

- <https://github.com/tkotani/ecalj/tree/master/Samples/GdNldau>
- [~/ecalj/Samples/ReNcub](#)
- IDU=, UH=, JH= specify parameters for LDA+U. IDU=#,#,#,... specifies which l-channels are to have U and the type of LDA+U implementation.
0 in a particular l-channel means no U is to be applied, 1 or 2 are for particular forms of LDA+U. For example,
IDU= 0 0 0 1 UH= 0 0 0 -0.28 JH=0 0 0 0

BoltTrap

--boltztrap option is to generate files required for boltztrap

Dielectric function

[~/ecalj/Samples/EPS](#)

Dielectric functions for Cu and GaAs. For Cu, we have intraband and interband contributions separately.

See [dielectric fuctnion](#).

Impact ionization rate

[~/ecalj/Samples/IIR](#)

Spin fluctuation

[~/ecalj/Samples/Magnon](#)

now with MaxlocWannier. going to move to MLO

Effective Screening Medium (ESM)

We can apply electric field to slab model. ESM combined with QSGW is quite unique. Ask us.

lmf and ctrl

See [lmf and ctrl](#)

gwsc and GWiput

See [gwsc](#).

For GPU, see [ecaljgpu](#) and [gwsc for GPU for ISSP](#) together.

See [explanation of GWinput](#).

getsym1: automatic symmetry line and BZ for band plot

See [sym1](#)

These citations are required.

1.Y. Hinuma, G. Pizzi, Y. Kumagai, F. Oba, I. Tanaka, Band structure diagram paths based on crystallography, Comp. Mat. Sci. 128, 140 (2017)

2.Cite spglib that is essential for getsym1.

ecalj/Samples/

- MLO: new localized basis
Maximally localized Wannier function and cRPA interaction
- MLWF_samples
Wannier function generator and cRPA
wannier90 method implemented in ecalj and cRPA.
(a cRPA method by Juelich group).
See Samples_MLWF/README.
~/ecalj/README_wannier.md This is going to be replaced with README_MLO.md
- mass_fit_test
Effective mass calculation. See README. probably not maintained\.

ecalj_auto

This is a suit of python script to run thousands of gwsc calculations.

[ecalj_auto](#)

background charge and fractional Z

PROF

[background charge](#)

How to perform paper-quality QSGW calculations with minimum costs.

The accuracy of band gaps can be ~0.1eV or larger for larger band gap materials..

In cases, it is easy, but in cases not so easy. So, it is better to use your own "simple criterion".

"Not stick to convergence so much. Just stick to Reproducibility."

4f and 5f

Caution: For 4f and probably also for 5f systems, some special care is required; just defaults ctrlgenM1 do not work.;

I think this is a little too old--> [HowToSet4f_GdQSGW4.pdf](#)

(I will revise this)

Except 4f systems, use default setting (just change k points).

Papers

It is instructive to reproduce samples in Deguchi paper[ecalj/Document/PAPERandPRESENTATION/deguchi2016.pdf].

We can set up templates for your calculations. Ask us.

We have latest paper at <https://arxiv.org/abs/2506.03477> for GPU version, but show some details of computational steps in ecalj.

In Japanese, pages by Dr.Gomi at <http://gomisai.blog75.fc2.com/blog-entry-675.html> and <https://qiita.com/takaokotani/items/9bdf5f1551000771dc48>.

How to perform paper-quality QSGW calculations with minimum costs.

We expect that the accuracy of band gaps can be $\sim 0.1\text{eV}$ (or larger for larger band gap materials).

In cases, it is easy but in cases not so easy (magnetic metals requires many number of iterations).

So, it is better to use your own "simple criterion".

"Not stick to convergence so much. Just stick to Reproducibility."

LDA calculation

We need to confirm LDA-level of calculations first.

The ctrl file is generated just from ctrl.* (crystal structure file)

For calculation of QSGW, use large enough NKABC, so as to avoid convergence check on them.

QSGW: how to check convergence

QSGW iteration cycle by gwsc contains (1) and (2)

(1) One-body self-consistent calculation

(where we add $\text{sigm} = \text{Sigma} - \text{Vxc}^{\text{LDA}}$ to one-body potential).

to determine one-body Hamiltonian H_0 .

(2) For given H_0 , we calculate sigm file.

PROF

Big iteration cycle of QSGW is made from (1)+(2).

(gwsc script. not run_arg is a subroutine of bash script)

With (1), we have small iteration cycle of one-body calculation with keeping given sigm.

In `save.*`, we see total energy (but not the total energy in the QSGW mode with sigm file), a line per each iteration of (1). A line "c ..." is the final iteration cycle of (1). "x ..." is unconverged (but no problem as long as we finally see "c ...").

The command `"grep '[cx]' 'save.*'"` gives an indicator for going to be converged or not.

Or you can take `"grep gap llmf.*run"` (see it bottom.)

Another way:

`~/ecalj/TestInstall/bin/diffnum QPU.3run QPU.6run`

is to compare two QPU files which contains QP energies.

(note: QP energies shown are calculated just at the begininig of iteration).

For insulator, (I think), comparing band gap for each iteration is good enough to check onvergence. But for metal, it is better to plot energy bands for some of final iterations, and overlapped(`cd QSGW.*run` and run `job_band`).

Another way is `grep rms lqpe*`. This gives rmsdel. Diffence of self-energy (at least we see it is getting smaller for initial first cycles).

How to make 80%QSGW +20% LDA, and SO setting

Note that sigm file contains $V_{xc}^{\text{QSGW}} - V_{xc}^{\text{LDA}}$.

If sigm exists, lmf read it, and run self-consistent calculations with adding sigm to the one-body potential.

See TableII in

<https://iopscience.iop.org/article/10.7567/JJAP.55.051201/pdf>

1. QSGW80(NoSC)

For practical prediction of band structure, such as band gap and so on, it may be better to use 80% QSGW +20% LDA procedure when you make band plot. After, you have rst and sigm files determined self-consistently
Run

```
job_band gaas -np 4 -vssig=0.80
```

(Confirm ssig is defined and cited as `ScaledSigma={ssig}` in the ctrl file). This gives a result of QSGW80nosc in the TableII.

PROF

2. QSGW80(Nosc)+SO

80%QSGW+20%LDA with SO=1 (L.S method).

If you like to include L.S method

```
mpirun lmf gaas -np 4 -vssig=0.80 -vso=1 -vnspin=2
```

This procedure makes self-consistency with keeping the sigm file. This may/(or may not) required. If you expect large obital moment this procedure may be needed.

```
job_band gaas -np 4 -vssig=0.80 -vso=1 -vnspin=2
```

NOTE: nspin=2 is required for so=1. rst and sigm are expanded for nspin=2 (you can not run with nspin=1, after rst and sigm are expanded).

3. QSGW80

With ssg=0.80, you can run QSGW calculaiton in gwsc.

Then you have self-consistent results of QSGW80.

You can simultaneously use the setting so=2 (Lz.Sz scheme).

Be careful for z-direction and setting of SYMOPS (so as to keep the z-axis), for so=2.

If you like to get results of QSGW80+SO, you need to set so=1 after self-consistent of sigm attained.

4. Example of GaAs

Good example to check band gap, and SO splitting at top of valence of Gamma point for ZB structure as GaAs.

Before run it, make sure your ctrl file include variables ssg, so, nspin by

```
>grep ssg ctrl.gaas
>grep so ctrl.gaas
>grep nspin ctrl.gaas
```

to know the variable ssgm is defined and used as

`ScaledSigma={ssg}`, `NSPIN={nspin}`. For `-vso=1` work, you also need to so is defined and `SO={so}` is set in ctrl.

How to do version up? git minimum

Be careful to do version up ecalj. It may cause another problem.

If you are not good at git, make another clone.

Do not mix up with previous version (e.g. where you install)

PROF

```
cd ecalj
git log
This shows what version you use now.
```

```
git diff > gitdiff_backup
This is to save your changes added to the original (to a file git_diff_backup ) for safe.
I recommend you do take git diff >foobar as backup.
git stash also move your changes to stash.
```

```
git checkout -f
CAUTION!!!!: this delete your changes in ecalj/.
This recover files controlled by git to the original which was just downloaded.
```

```
git pull
This takes all new changes. It is safer to use git fetch and gitk --all (git checkout
```



```
FETCH_HEAD -b youbranch) to checkout your local branch.
```

I think it is recommended to use

```
gitk --all
```

and read this document. Difference can be easily taken,
e.g. by

```
git diff d2281:README 81d27:README  
(here d2281 and 81d27 are several digits of the beginning of its version id).
```

```
git show 81d27:README
```

is also useful.