ecalj manual

https://github.com/tkotani/ecalj

August 8, 2025

1 Spectrum function: How to calculate $\langle \mathbf{q}n|\Sigma(\omega)|\mathbf{q}n\rangle$

We have an example at `ecalj/MATERIALS/SiSigma/`, where you can just type `job`. It calls a shell script `gwsigma`, which is just a modification of `gwsc` for spectrum function plotting. If you have `sigm.*`, it will automatically read it as in the case of `gwsc`.

By the script `gwsigma`, we calculate the diagonal elements $\rangle\psi(\mathbf{q},n)|\sigma_{\rm c}(\omega)|\psi(\mathbf{q},n)\langle$. Thus we need to set $\mathbf{q}$ and band index n for which we calcualte. In addition, we need to set resolution of ω.

1. Set `<QPNT>` section. This section is to set the q point, and band index for which we calculate the self energy. In addition, energy mesh for plotting is set.

```
(A) Set q point set


 If you set
----------
*** all q -->1, otherwise 0;  up only -->1, otherwise 0
         1           0
----------
You will have self-energy for all irreducible k points. This may be needed for A(omega).
or
You have to set all q points as
----------
*** q-points, which shoud be in qbz.,See KPNTin1BZ.
         3            <--- number of readin q point
1     0.0000000000000000     0.0000000000000000     0.0000000000000000 <--1st number is
2    -0.5000000000000000     0.5000000000000000     0.5000000000000000
3     0.0000000000000000     0.0000000000000000     1.0000000000000000
----------
 To know allowed q points on regular mesh point, run the command "mkGWIN_lmf2", then
 supply n1,n2,n3. The templete of GWinput.tmp contains all possible q points. Edit it.

 NOTE:Anyq option can allow you to specify any q points by shifted mesh technique.
 (if necessary, but only for some special purpose).
```

(B) Band index set

It is specified by the section

```
----------
*** no. states and band index for calculation.
2
4  5
----------
```

means the self-energy for band index 4 and 5. Just two bands.

If you like to plot self energy from 1 through 8, use

```
*** no. states and band index for calculation.
8
1 2 3 4 5 6 7 8
----------
```

If you need 17 bands for example, it should be 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
in addition to 17 (number of bands at the first line).

(C) energy mesh set

At the bottom of <QPNT> section, we have

```
----------------
*****
 0.01 2.0
----------------
```

Two real number should be supplied.

These are dwplot and omegamaxin, read in hsfp0.m.F by a line

```
  read (ifqpnt,*,err=2038,end=2038) dwplot,omegamaxin
```

dwplot (=0.01 Ry) is mesh for self energy.

omegamaxin=(2.0 Ry) means the range "-2 Ry to 2 Ry" for self-energy plot.

Note that imaginary part of Sigma is given as the comvolution of ImW(omega) and the pol
(esmr in GWinput gives energy smearing of the pole). Resolution for Im W (near omega=0)
I think that the reolution of self-energy is ~ 0.05 eV in the default setting.
This is because {\tt dw} \sim {\tt esmr} \sim 0.05 eV.

2 Run gwsigma. This will run

```
  echo 4| mpirun -np 24 hsfp0,
```

after dielectric funcition is calculated.

Then we have SEComg.UP (DN) files, Look for file handle, ifoutsec,

```
for the file in fpgw/main/hsfp0.m.F to see format for the file.
(not hsfp0.sc.m.F but hsfp0.m.F). Search a line
     open(ifoutsec,file='SEComg'//sss) (around hsfp0.m.F L1052)
 You can find that we use folloing lines to plot SEComg.*.
  ----------------
         write(ifoutsec,"(4i5,3f10.6,3x,f10.6,2x,f16.8,x,3f16.8)")
   &          iw,itq(i),ip,is, q(1:3,ip),  eqx(i,ip,is),
   &          (omega(i,iw)-ef)*rydberg(),  hartree*zsec(iw,i,ip) !,sumimg
   This means we use energy in eV.
   iw:      omega index
   itq(iq): band index specified by <QPNT>
   ip:      k point index specified by <QPNT>
   is:      spin index
   q:       q vector (cartesian in 2pi/alat)
   eqx:     eigenvalue in eV. (I think relative to the Fermi energy)
   (omega(i,iw)-ef)*rydberg():  omega relative to the Fermi energy
   hartree*zsec(iw,i,ip):       Self energy. real and imaginary part.(complex, two value

 You can only repeat echo 4| mpirun -np 24 hsfp0
 when you change setting in <QPNT> section.
```

3 Example. There is an example MATERIALS/SiSigma/

```
plot 'SEComg.UP' u ($9):($10) w l,'' u ($9):($11) w l
can give a plot for Re (Sigma_c(omega)) and Im(Sigma_c).

9th:  energy in eV   (omega(i,iw)-ef)*rydberg()
10th: real part      Re hartree*zsec(iw,i,ip)
11th: imag part      Im hartree*zsec(iw,i,ip)
```

4 To get integrated spectrum function (DOS), we need to superpose all the spectrum function (All q points and all band index). Be careful about the degeneracy (multiplicity) for each q points. You have to build it from SEComg file. To know the multiplicity, search following lines ofkeyword `Multiplicity` in the console output of qg4gw (lqg4gw).

Anyway, consider about "is it worth to do?" To confirm your result, use sum rule (sum of spectrum weight). And pay attention to the relation between real and imag parts (Hilbert transformation).

2 Main stage of gwsc

We can start the main stage of *GW*calculation from these files;
`GWinput DATA4GW_V2 CphiGeig QGpsi QGcou Q0P QIBZ SYMOPS BZDATA HAMindex CLASS`;
these files contains eigenfunctions and so on in the manner of Eq.(17) of [**?**], eigenvalues and other required information. This is the starting point of the GW calculation.

- GWinput : computational conditions.
- DATA4GW_V2 : Crystal structures and so.
- CphiGeig : Eigenvalues and Eigenfunctions
- QGpsi : q and G vector for the eigenfunctions(q means **k** in the previous section),
- QGcou : q and G vector for the Coulomb matrix
- Q0P : q points near q=0 instead of q=0 (offset Gamma points)
- QIBZ : irreducible q points (This is also contained in BZDATA).
- SYMOPS : point group operation
- BZDATA : q points date (and tetrahedron weights if necessary) for BZ integrals.
- HAMindex : Hamiltonian index, all required complex index for Hamiltonian of PMT method. (See the top of subroutine write_hamindex in lm7K/subs/m_hamindex.F. This is also in fpgw/gwsrc/m_hamindex.F. Identical files are in two different directory— it should be avoided in future.)
- CLASS : class information or atomic sites (equivalent sites).

3 Product basis

```
<PRODUCT_BASIS>
 tolerance to remove products due to poor linear-independency
  0.100000D-02 ! =tolopt
```

When the product basis are made, we may have poorly linear independent basis. For example, one in the set $\{f_1, f_2, ..., f_n\}$ would be almost give by a linear-combination of others. We need to make the linear-independent set. Therefore, after calculating the overlap matrix $\langle f_i | f_j \rangle$. We do diagonalization, then we remove eigenvectors corresponding to small eigenvalues than `tolopt`. See the **hbasfp0** command in **gwsc**

```
lcutmx(atom) = maximum l-cutoff for the product basis.
  4  4  4  2  2  4  4
```

For $\phi_1 \times \phi_2$ case, $|l_1 - l_2| \leq l_{tot} \leq |l_1 + l_2|$. So 'lcutmx' changes the maximum cutoff for the l_{tot}. The order is the same as the order of atoms in the **ctrl** file.

```
  atom   l  nnvv  nnc !
    1    0    3    3
    1    1    3    2
    1    2    2    1
```

'atom' means the atom number identified in the **ctrl** file.
'l' is the angular momentum quantum number.
'nnvv' is the number of radial functions (valence) on the augmentation-waves.
'nnc' is the number of radial functions for core.
The latter two ones, 'nnvv' and 'nnc', will be understood more clearly if you see the following ones.

```
atom   l    n  occ unocc  ! Valence(1=yes,0=no)
  1    0    1    1    1   ! 5S_p  -----
  1    0    2    0    0   ! 5S_d
  1    0    3    1    1   ! 4S_l
  1    1    1    1    1   ! 5p_p
  1    1    2    0    0   ! 5p_d
  1    1    3    1    1   ! 4p_l
```

Above options are about the product basis set within MT (Valence).
'atom' and 'l' are explained above. 'nnvv' for 'atom = 1 and l = 0' was '3' so this case we have 3 basis ('n = 1, 2, 3')
'n' is the degree of freedom of the radial function, ϕ. 'n = 1' means ϕ, 'n = 2' means $\dot{\phi}$, and 'n = 3' means kind of $\ddot{\phi}$, which the dot above the letter represents the differentiation with respect to the energy. So 'n = 1 and 2' is related to the linearization of the radial function and 'n = 3' is the local orbital which is restricted in MT. The local orbital can be modified changing 'PZ' in the **ctrl** file. Finally, the number of the basis set which is needed for expanding eigenfunctions is $(l+1)^2 \times n$.
'occ' and 'unocc' mean that we use only ones that checked as '1', in other words we neglects '0' cases for making product basis. Be careful for confusion with name 'occ' and 'unocc'. These don't mean that occupied or unocc. When making product basis, $M = \phi_1 \times \phi_2$, 'occ' corresponds to ϕ_1 and 'unocc' to ϕ_2. For example,

```
atom   l    n  occ unocc  ! Valence(1=yes,0=no)
  1    0    1    1    1   ! 5S_p  -----
  2    3    1    0    1   ! 4f_p
```

If the options are like the above, the product basis will be consists of $(\phi_1 = \phi_{atom=1,l=0}) \times (\phi_2 = \phi_{atom=1,l=0})$, $(\phi_{atom=1,l=0} \times \phi_{atom=2,l=3})$. As you can see, $(\phi_1 = \phi_{atom=2,l=3})$ is skipped.
In the **ctrl** file, 'EH' controls the l part. As for 'EH', (s, p, d, f) are used but **GWinput** file uses (s, p, d, f, g). 'EH' : HEAD part. 'GWinput' : contains TAIL part... need more explanation.

```
atom   l    n  occ unocc  ForX0 ForSxc ! Core (1=yes, 0=no)
  1    0    1    0    0      0    0    ! 1S -----
  1    0    2    0    0      0    0    ! 2S
  1    0    3    0    0      0    0    ! 3S
```

Above options are about the product basis set within MT (Core).
'nnc' for 'atom = 1 and l = 0' was '3' so this case we have 3 basis ('n = 1, 2, 3')
Finally, for the convergence check, we can modify the following three things, (i) tolerance, (ii) lcutmx, and (iii) occ and unoccu.

4 Wannier function

We can generate Wannier functions (maximally localized Wannier Functions or similar) by a script `genMLWF`. It automatically perform cRPA calculation sucessively. (If it is not necessary, insert 'exit' in `genMLWF`, after it performs `lmaxloc2`).

Try to run examples in `ecalj/MATERIALS/Sample_MLWF/`. Read `README` in it. To run the script `genMLWF`, we need to get `GWinput` by editing `GWinput.tmp`. (`mkGWin_lmf2` contains default Wannier section). In addition, we have some settings (energy windows and so on). This is the example of the initial conditions for Cu case. 5 is the number of Wannier function. The most left

one means ϕ index and the right one of it is $\dot{\phi}$ index. They are written in the **@MNLA_CPHI** file.

Then we can run `genMLWF`. After it finished, we can analyze it results. (if you don't need Wannier funciton plot, You can skip a line of wanplot in genMLWF. Then we don't need to set `vis_wan_*` options.)

4.1 lwmatK1 and lwmatK2

If you input the following command

```
>grep Wan lwmatK*
```

You will get the following results. (This case : Cu cases)

```
lwmatK1:  Wannier    1    1   24.644475    0.000000 eV
lwmatK1:  Wannier    1    2   24.644576    0.000000 eV
lwmatK1:  Wannier    1    3   25.471361    0.000000 eV
lwmatK1:  Wannier    1    4   24.644575    0.000000 eV
lwmatK1:  Wannier    1    5   25.470946    0.000000 eV
lwmatK2:  Wannier    1    1    0.000000 eV   -21.263759   -0.000000 eV
lwmatK2:  Wannier    1    2    0.000000 eV   -21.263839    0.000000 eV
lwmatK2:  Wannier    1    3    0.000000 eV   -21.931033   -0.000000 eV
lwmatK2:  Wannier    1    4    0.000000 eV   -21.263839   -0.000000 eV
lwmatK2:  Wannier    1    5    0.000000 eV   -21.930702   -0.000000 eV
```

```
### Wanneir Branch now under developing (imported from T.Miyake's Wannier and H.Kino's).
   A. make at ecalj/fpgw/Wannier/ directory, and do make, and make install.
      (need to check Makefile first). You first have to install fpgw/exec/ in advance.
   B. Samples are at these directories.
      MATERIALS/CuMLWFs (small samples),
      MATERIALS/CuMLWF/
      MATERIALS/CuMLWFs/
      MATERIALS/FeMLWF/
      MATERIALS/NiOMLWF/
      MATERIALS/SrVO3MLWF/
   C. With GWinput and ctrl.*, run
      >genMLWF
      at these directories.
      In GWinput, we supply settings to generate Wannier funcitons. (Sorry,not documentet yet..
   D. After genMLWF, do
      >grep Wan lwmatK*
      then compare these with Result.grepWanlwmatK
      These are onsite effective interactions (diagonal part only shown).
      *.xsf are for plotting the Maximally localized Wannier funcitons.
Anyway, documentaion on Wannier is on the way half.
Time consuming part (and also the advantage) is for effective interaction in RPA.
Look into the shell script genMLWF; you can skip last part if you don't need the effective inte
```

5 How to set local orbitals

As we stated, do "lmfa |grep conf" to check used MTO basis.

We have to set SPEC_ATOM_PZ=?,?,?
(they ordered as PZ=s,p,d,f,...) to set local orbitals.

lmv7 (originally due to ASA in Stuttgart) uses a special terminology
"continious principle quantum number for each l", which is just
relatated to the logalismic derivative of radial funcitons at MT
boundary. It is defined as
 P= principleQuantumNumber + 0.5-1/pi*atan(r* 1/phi dphi/dr),
where phi is the radial function for each l. For example,
 P= n.5 for l=0 of free electron (flat potential) because phi=r^0,
 P= n.25 for l=1 because phi=r^1;
 P= n.147584 for l=2 because phi=r^2; P=, n.102416 n.077979 for l=3,4.
(Integer part can be changed). See Logarithmic Derivative Parameters in
http://titus.phy.qub.ac.uk/packages/LMTO/lmto.html#section2

Its fractioanl part 0.5-atan(1/phi dphi/dr) is closer to unity for
core like orbital, but closer to zero for extended orbitals.

Examples of choice:
Ga p: in this case, choice 0 or choice 2 is recommended.
    We usually use lo for semi-core, or virtually unoccupied level.

   (0)no lo (4p as valence is default treatment without lo.)
      3p core, 4p valence, no lo: default.
      Then we have choice that lo is set to be for 3p,4p,5p.
   (1)3p lo ---> 4p val (when 3p is treated as valence)
      3d semi core, 4d valence
      Set PZ=0,3.9
      (P is not requied to set. *.9 for core like state. It is just an initial condition.)
   (2)5p lo ---> 4p val (PZ>P)
      Set PZ=0,5.5
      5.5 is just simply given by a guess (no method have yet
implemented for
      If 5.2 or something, it may fails
      because of poorness in linear-dependency. We may need to observe
      results should not change so much on the value of PZ.

   (3xxx)4p lo ---> 5p val (we don't use this usually. this is for test purpose)
      4p lo, 5p valence
      Set PZ=0,4.5 P=0,5.5 (In this case, set P= simultaneously).
      (NOTE: zero for s channel is to use defalut numbers for s)

Ga d: (in this case, choice 0 or choice 1 is recommended).
   (0)no lo (3d core, 4d valecne, no lo: default.)

```
          Then we have choice that lo is set to be for 3d,4d,5d.
     (1) 3d lo ---> 4d val  (when 3d is treated as valence)
         Set PZ=0,0,3.9  (P is not required to set)
     (2) 5d lo ---> 4d val  (PZ>P)
         Set PZ=0,0,5.5
     (3xxx) 4d lo ---> 5d val  (this is for test purpose)
         Set PZ=0,0,5.5 P=0,0,4.5
         (NOTE: zero  for s,p are to use defalut numbers )

  %  If you like to read from atm.ga file instead of rst file(if exist).
  %  You have to do lmf --rs=1,1,0,0,1, for example. See lmf --help
  %  Becase rst file keeps the setting of MTO, thus change in ctrl is not
  %  reflected without the above option to lmf.
=================================================================
```

6 Product basis section

This section is to define product basis to expand W and so. Numbers are read by free format read(5,*),thus the numbers should be separated by space. The line number in this section is meaningful (you can not add comment lines).

```
<PRODUCT_BASIS>
 tolerance to remove products due to poor linear-independency
  0.100000D-04 ! =tolopt; larger gives smaller num. of product basis. See lbas and lbasC, which
 lcutmx(atom) = maximum l-cutoff for the product basis.  =4 is required for atoms with valence
  4  3
  atom   l  nnvv  nnc ! nnvv: num. of radial functions (valence) on the augmentation-
waves, nnc: num. for core.
    1    0    2    3
    1    1    2    2
    1    2    3    0
    1    3    2    0
    1    4    2    0
    2    0    2    1
    2    1    2    0
    2    2    2    0
    2    3    2    0
    2    4    2    0
  atom   l    n  occ unocc  ! Valence(1=yes,0=no)
    1    0    1    1    1   ! 4S_p  -----
    1    0    2    1    0   ! 4S_d
    1    1    1    1    1   ! 4P_p
    1    1    2    0    0   ! 4P_d
    1    2    1    1    1   ! 4D_p
    1    2    2    0    0   ! 4D_d
    1    2    3    1    1   ! 3D_l
    1    3    1    0    1   ! 4f_p
    1    3    2    0    0   ! 4f_d
    1    4    1    0    0   ! 5g_p
    1    4    2    0    0   ! 5g_d
    2    0    1    1    1   ! 2S_p  -----
    2    0    2    0    0   ! 2S_d
    2    1    1    1    1   ! 2P_p
    2    1    2    0    0   ! 2P_d
    2    2    1    1    1   ! 3d_p
    2    2    2    0    0   ! 3d_d
    2    3    1    0    1   ! 4f_p
    2    3    2    0    0   ! 4f_d
    2    4    1    0    0   ! 5g_p
    2    4    2    0    0   ! 5g_d
  atom   l    n  occ unocc  ForX0 ForSxc ! Core (1=yes, 0=no)
    1    0    1    0    0      0    0    ! 1S -----
    1    0    2    0    0      0    0    ! 2S
    1    0    3    0    0      0    0    ! 3S
    1    1    1    0    0      0    0    ! 2P
    1    1    2    0    0      0    0    ! 3P
    2    0    1    0    0      0    0    ! 1S -----
</PRODUCT_BASIS>
```

This section is read in the free format in fortran. So, e.g., `0.01` works as same as `0.10000D-01`. The line order is important (you have to keep the order given by GWinput.tmp). Be careful atom atom id—lmf may re-order it and pass it to gw code. Look into LMTO file (generated by **mkGWIN_lmf2**); which contains crystal structure information after such re-ordering by lmf. I used `!` to make clear that things after `!` are comments. But `!` is not meaningful – just the expected numbers of data separated by blank(s) are read for each line from the beginning of lines.

- `0.100000D-02 ! =tolopt` controls a number of Product basis to expand the Coulomb interaction within MTs. `tolopt` is a criterion to remove the poorly linear-independent product basis. Note that the product basis, which is to expand the Coulomb interaction, is different from the basis to expand eigenfunctions. In our experience, `0.100000D-02` (=0.001) is not so bad. If you like to reduce computational time use 0.01 or so, but a little dangerous in cases. With 0.0001, we can check stability on it.

(note: By supplying multiple numbers, we can specify `tolopt` atom by atom. Remember `lmchk` gives atom ID.)

- lcutmx(atom) is the l cutoff of product basis for atoms in the primitive cell (do lmchk for atom id). In the case of Oxygen, we can usually use lcutmx=2 (need check by the difference when you use lcutmx=2 or lcutmx=4). Then the computational time is reduced well.

- (dec2014:`<PBASMAX>` is not checked recently; see `fpgw/main/hbasfp0.m.F` and `fpgw/gwsrc/basnfp.F`.) You can use `<PBASMAX>` section to override this setting. It is given as

```
<PBASMAX>
1  5 5 5 3 3
2  5 5 3 2 3
3  3 3 2 2 2
</PBASMAX>
```

  The first number is for atom index (fixed), and other are product basis for each l channel.

- The integer numbers in 4th line `lcutmx` gives the maximum angular momentum l for the product basis for each atomic site. In our experience, `lcutmx`=4 is required when the semi-core (or valence) $3d$ electrons exist and we want to calculate the QP energies of them.

- Keep a block starting from " atom l nnvv nnc ..." as it originally generated in GWinput.tmp. It just shows that how many kinds of radial functions for cores and valence electrons for each atom and l. `nnvv`=2 in the case of ϕ and $\dot{\phi}$; `nnvv`=3 in the case to add the local orbital in addition.

- There are two blocks after the line "`atom l n occ unocc :Valence(1=yes, 0=no)`' and after "`atom l n occ unocc ForX0 ForSxc ! Core (1=yes, 0=no)`'. These are used to choose atomic basis to construct the product basis. The product basis are generated from the products of two atomic basis.

  GWinput.tmp generated by **mkGWIN_lmf2** contains labels on each orbitals as `4S_p`, `4S_d`, `4P_p`... Here `4S_p` is for ϕ_{4s}; `4S_d` for $\dot{\phi}_{4s}$; `3D_l` for $\phi^{\rm local}_{3d}$. Capital letter just after the principle-quantum number means the orbital is used as 'Head of MTO'; lowercase means just used only as the 'tail of MTO'.

  The switches for columns labeled as `occ` and `unocc`. take 0 (not included) or 1 (included). With the switch, we can construct two groups of orbitals, `occ` and `unocc`. In this sample GWIN_V2 as for atom 1, $\{\phi_{4s}, \dot{\phi}_{4s}, \phi_{4p}, \phi_{4d}, \phi^{\rm local}_{3d}, \phi^{\rm core}_{3s}, \phi^{\rm core}_{3p}\}$ consist the group `occ`, and $\{\phi_{4s}, \phi_{4p}, \phi_{4d}, \phi^{\rm local}_{3d}, \phi_{4f}\}$ consists the group `unocc`. So the any product of combinations $\{\phi_{4s}, \dot{\phi}_{4s}, \phi_{4p}, \phi_{4d}, \phi^{\rm local}_{3d}, \phi^{\rm core}_{3s}, \phi^{\rm core}_{3p}\} \times \{\phi_{4s}, \phi_{4p}, \phi_{4d}, \phi^{\rm local}_{3d}, \phi_{4f}\}$ are included as for the basis of the product basis. As for atom 2, $\{\phi_{2s}, \phi_{2p}, \phi_{3d}\} \times \{\phi_{2s}, \phi_{2p}, \phi_{3d}, \phi_{4f}\}$ are included.

- Core section: (not worth to read, since we currently use no CORE2, `A=B=C=0`.)

  Each line of the last section of `Product BASIS` forms

```
atom   l    n  occ unocc   ForX0 ForSxc :CoreState(1=yes, 0=no)
  1    2    1    A    x      B    C
```

  At first you have to understand the concept of CORE1 and CORE2 in EQ.35 Ref.I. However, in our recent calculations, we do not use "CORE2" generally. So, in such a case, set `A=B=C=0`. And treat shallow cores (above Efermi−2Ry or so) as valence electron by "local orbital method" in lmf.

- Be careful. Current version is inconvenient... Need to repeat mkGWIN_lmf2 to generate GWinput template when you add PZ (local orbital).

  [(Note: you can skip here if you don't use CORE2.)

  Each of `A,x,B,C` takes 0 or 1. There are some possible combination of these switches;

  1. If you take `( A x B C )= (1 0 1 1)`, then the core is included in core2. In other words, this core is treated in the same manner of the valence electron.
  2. If you take `( A x B C )= (0 0 0 0)`, then the core is included in core1. The (exchange only) self-energy related to this core is included in `SEXcore`.
     `C` is the key switch which determine whether it is included in core1 or core2. There could be another option.
  3. If you take `( A x B C )= (1 0 0 1)`. This core is in core2. But it is not included in the calculation of D and W. This core is only included for SEX and SEC calculations.

  These three kinds of choices are reasonable ones but we can consider some another choice. In the following, we show how these switches (`A,B,C`) affect executions called from `gw_lmfh` (essentially as same as `gw_lmf`).

  - `hbasfp0`(mode 3) :Product basis for exchange due to core.
    We include the `C`=0 cores as a part of the product basis as if `A`=1 `x`=0.
  - `hsfp0`(mode 3): exchange mode for core.
    Σ_x only due to the `C`=0 cores are calculated.
  - `hbasfp0` (mode1): Product basis.
    Only see the switch `A` and `x`. The product basis is generated from (occupied $\times$ unoccupied), where `A`=1 core is included as one of the occupied basis.
  - `hsfp0` (mode 1): exchange mode.
    Only see the switch `C`. Σ_x due to valence and due to `C`=1 cores are calculated.
  - `hx0fp` (mode 1): $W - v$ calculation.
    Only see the switch `B`. W is calculates using all the valence and `B`=1 cores.
  - `hsfp0` (mode 2): correlation mode.
    Only see the switch `C`. Σ_c due to valence and due to `C`=1 are calculated.

• After you perform `gw_lmfh` or anything, you find output files `lbas` by `hbasfp0` (mode1), and/or `lbasc` by `hbasfp0` (mode3) for core. These contains important information about how many and how product basis are chosen. E.g. '`grep nbloch lbas`' shows how many product basis are used in the calculations.

7 Main Output Files of GW part

7.1 QPU

This is the main output[1] in human readable format.

An example of one-shot GW by `gw_lmfh`) is (In the case of QSGW, Z ($Z = 1$) is not shown):

```
 ================================================================
  quasiparticle energies MAJORITY
 ================================================================
E_shift=  0.4263273221017709D+00  0.6075150850568627D+00  0.7046628446164018D+00 eV

    q      state SEx   SExcore SEc    vxc   dSE  dSEnoZ  eLDA    eQP  eQPnoZ   eHF  Z   2Z*Simg ReS(elda)
0.0 0.0 0.0 1  -29.56  -1.97  10.40 -20.22 -0.52  -0.90 -19.08 -19.42 -19.71 -30.81 0.58 0.95   -
21.12
0.0 0.0 0.0 2  -30.52  -2.24  10.09 -21.53 -0.70  -1.14 -18.06 -18.58 -18.93 -29.72 0.61 0.96   -
22.66
0.0 0.0 0.0 3  -20.67  -1.87   5.97 -16.85  0.19   0.28  -7.20  -6.83  -6.65 -13.32 0.67 0.66   -
16.57
  ...
```

From the 6h line, we have the eigenvalue data. All of the unit of energy is in eV. We should note that the zero-level of these values `eLDA eQP eQPnoZ` can be changed by hqpe. This `eLDA - E_shift` are the eigenvalues relative to a Fermi energy determined by the smearing method. Detailed value of `eLDA` is in TOTE2.UP. Detailed value of `eLDA- E_shift` is in TOTE.UP.

`q` : $\mathbf{k}$ vector

`state`: Band index n, which is from the lowest eigenvalue (not include cores).

`SEx`: $= = \langle\Psi_{\mathbf{k}n}|\Sigma_{\rm x}^{\rm core2+valence}(\mathbf{r},\mathbf{r}')|\Psi_{\mathbf{k}n}\rangle$

`SExcore`: $= \langle\Psi_{\mathbf{k}n}|\Sigma_{\rm x}^{\rm core1}(\mathbf{r},\mathbf{r}')|\Psi_{\mathbf{k}n}\rangle$

`SEc`: $= \langle\Psi_{\mathbf{k}n}|\Sigma_{\rm c}^{\rm core2+valence}(\mathbf{r},\mathbf{r}',\epsilon_n(\mathbf{k}))|\Psi_{\mathbf{k}n}\rangle$

`vxc`: LDA exchange correlation energy. $\langle\Psi_{\mathbf{k}n}|V_{\rm xc}^{\rm LDA}([n_{\rm total}],\mathbf{r})|\Psi_{\mathbf{k}n}\rangle$

`dSE`: $Z_{n\mathbf{k}}\times$ `dSEnoZ`

`dSEnoZ`: $\langle\Psi_{\mathbf{k}n}|\Sigma_{\rm x}^{\rm core1}(\mathbf{r},\mathbf{r}')+\Sigma_{\rm xc}^{\rm core2+valence}(\mathbf{r},\mathbf{r}',\epsilon_n(\mathbf{k}))|\Psi_{\mathbf{k}n}\rangle - \langle\Psi_{\mathbf{k}n}|V_{\rm xc}^{\rm LDA}([n_{\rm total}],\mathbf{r})|\Psi_{\mathbf{k}n}\rangle$
$=$ `SEx + SExcore + SEc - vxc`

`eLDA`: LDA eigenvalues. $\epsilon_n(\mathbf{k})$

`eQP`: QP energy. $\epsilon_n(\mathbf{k})$+`dSE`

`eQPnoZ`: QP energy without Z. $\epsilon_n(\mathbf{k})$+`dSEnoZ`

`eHF`: HF energy of 1st iteration. $\epsilon_n(\mathbf{k})$+`SEx + SExcore -vxc`

`Z`: Z factor. $Z_{n\mathbf{k}}$

`2Z*Simg`: Quasi-particle life time. $2Z_{n\mathbf{k}}\times{\rm Im}\langle\Psi_{\mathbf{k}n}|\Sigma_{\rm c}^{\rm core2+valence}(\mathbf{r},\mathbf{r}',\epsilon_n(\mathbf{k}))|\Psi_{\mathbf{k}n}\rangle$
(Is this really the usual definition of the life time?—don't believe me)

`ReS(elda)`: ${\rm Re}\langle\Psi_{\mathbf{k}n}|\Sigma_{\rm x}^{\rm core1}(\mathbf{r},\mathbf{r}')+\Sigma_{\rm xc}^{\rm core2+valence}(\mathbf{r},\mathbf{r}',\epsilon_n(\mathbf{k}))|\Psi_{\mathbf{k}n}\rangle$

7.2 XCU

LDA exchange-correlation. Detailed data of above `vxc`.

7.3 SEXU

Exchange part of the self-energy due to valence electrons. Detailed data of above `SEx`.

[1]Note that QPU also implies QPD and so on. U is for up D is for down spins.

7.4 SEXcoreU

Exchange part of the self-energy due to core. Detailed data of above `SExcore`.

7.5 SECU

Correlation part of the self-energy. Detailed data of above `SEc`.

7.6 TOTE.UP (TOTE.DN)

This is a central output. It contains LDA and QP energies. These values are relative to a Fermi energy determined by the smearing method. It contains two kind of QP energies `QP QPnoZ`. The first line contains the Fermi energy in Ry determined by the smearing method. It is also shown in the end of DOSACC.lda.

8 gwsc script and its I/O Files

In `gwsc`, we have a loop of QSGW self-consistency. Look into the `gwsc` script. In each iteration, we perform these fortran programs;

```
NO_MPI=0 #this is used for non-mpi versions of fortran program.
### self-consistent calculation for given Sigma(self-energy) ###
         run_arg '---' $MPI_SIZE $nfpgw   /lmf-MPIK    llmf $TARGET

### Preparation stage ############################################
argin=0;  run_arg $argin $NO_MPI  $nfpgw   /lmfgw      llmfgw00 $TARGET
argin=1;  run_arg $argin $NO_MPI  $nfpgw   /qg4gw      lqg4gw   #Generate requied q+G v
argin=1;  run_arg $argin $MPI_SIZE $nfpgw  /lmfgw-MPIK llmfgw01 $TARGET
          run_arg  '---' $NO_MPI   $nfpgw  /lmf2gw     llmf2gw  #reform data for gw
### Main stage of gw #############################################
argin=0;  run_arg $argin $NO_MPI   $nfpgw  /rdata4gw_v2 lrdata4gw_v2 #prepare files
argin=1;  run_arg $argin $NO_MPI $nfpgw    /heftet      leftet # A file EFERMI for hx0fp0
argin=1;  run_arg $argin $NO_MPI $nfpgw    /hchknw      lchknw # A file NW, containing nw
 ## Core part of the self-energy (exchange only) ##
 argin=3;  run_arg $argin $NO_MPI $nfpgw    /hbasfp0    lbasC # Product basis generation
 argin=3;  run_arg $argin $MPI_SIZE $nfpgw  /hvccfp0    lvccC # Coulomb matrix for lbasC
 argin=3;  run_arg $argin $MPI_SIZE $nfpgw  /hsfp0_sc   lsxC  # Sigma from core1

 ## Valence part of the self-energy Sigma ##
 argin=0;  run_arg $argin $NO_MPI $nfpgw   /hbasfp0     lbas # Product basis generation
 argin=0;  run_arg $argin $MPI_SIZE $nfpgw /hvccfp0     lvcc # Coulomb matrix for lbas
 argin=1;  run_arg $argin $MPI_SIZE $nfpgw /hsfp0_sc    lsx  # Exchange Sigma
 argin=11; run_arg $argin $MPI_SIZE $nfpgw /hx0fp0_sc   lx0 $lx0_para_option #x0 part
 argin=2;  run_arg $argin $MPI_SIZE $nfpgw /hsfp0_sc    lsc  #correlation Sigma
 argin=0;  run_arg $argin $NO_MPI $nfpgw   /hqpe_sc     lqpe #all Sigma are combined.
```

run_arg:

Here a subroutine of bash **run_arg** was used, which is given in ecalj/lm7K; it just invoke a command with the argument `$argin` (this is read by read(*,*) in fortran). In cases with `MPI_SIZE`/ = 0, mpirun is invoked. Console out put go to l* files. For example,

```
argin=2; run_arg $argin $MPI_SIZE $nfpgw /hsfp0_sc  lsc #correlation Sigma
```

invokes `hsfp0_sc` with argument '2' by mpirun with the -np $MPI_SIZE. Console outputs are written into logfiles such as lqpe. `$nfpgw` contains path to the execution binaries.

In the following, We explain input/output files for each fortran program. Note that "`echo 0|lmfgw`" means invoking `lmfgw` with `argin=0`.

8.1 echo 0| lmfgw si

See Sec.??.

8.2 echo 1| qg4gw

This makes **q** points, and **G** vectors for these **q**. (**q** was **k** in previous sections.) Main routine of qg4gw is `fpgw/main/qg4gw.m.F` and calls `fpgw/gwsrc/mkqg.F`

Input files

- GWinput :
- LATTC :
- SYMOPS :

Output files

- QGpsi : (bin) q and G vector for the eigenfunctions.
- QGcou : (bin) q and G vector for the Coulomb matrix
- Q0P : offset-Γ points which are the replacement of the q=0 points. See section??.
- QIBZ : q points in the Irreducible BZ.
- BZDATA : (bin) BZ data for integration (include tetrahedrons if necessary). See e.g. `main/hx0fp0.sc.F` and search "`call read_BZDATA`", which is a readin routine of this file defined in `rwbzdata.F`.
- KPTin1BZ.mkqg.chk : list of q in the 1st BZ for check.
- QBZ : q point in the 1st BZ.
- EPSwklm : Required information for the BZ integration (mainly in order to evaluate the weight in the Γ cell). See Eq.xxx in [?].

8.3 echo 1|lmfgw si

Calculate eigenfunctions, eigenvalues and $\langle\psi|H_{\rm KS}|\psi\rangle$

Input files

- ctrl.si :
- rst.si : (bin) Restart file of the lmf calculation. It contains all information
- sigm.si : (bin) If this exist and
- QGpsi,QGcou,Q0P : :
- NLAindx : :

Output files

- gwa.si : (bin) atomic data
- gwb.si : (bin) band data
- gw1.si : (bin) $\langle\psi|H_{\rm KS}|\psi\rangle$
- gw2.si : (bin) $\langle\psi|H_{\rm KS}-V_{\rm xc}(n_{\rm total})|\psi\rangle$.
- vxc.si,evec.si : (bin) used in hqpe.sc.m.f as "v_xc" and "evec").

vxc.si contains $\langle\psi|V_{\rm xc}(n_{\rm total})|\psi\rangle$ including off-diagonal part. evec.si contains eigenfunctions.

- normchk.si : norm check (only for check) This is like this

```
> head -20 normchk.si
#       IPW          IPW(diag)   Onsite(tot)   Onsite(phi)      Total
      0.436015      0.805123      0.563972      0.562573      0.999988
      0.339134      0.620353      0.660515      0.656881      0.999649
      0.339133      0.620353      0.660516      0.656882      0.999649
      0.339133      0.620353      0.660516      0.656882      0.999649
      0.507738      0.648515      0.492040      0.487673      0.999778
...
```

This check is sometimes important for debugging and to determine the cutoff parameter `QGcut_psi`. The first line (corresponding to 1st band of 1st q point) means that total normalization almost unity = 0.999988 = 0.436015 + 0.563972. Because we expand the MTO by

IPW, the normalization is a bit different from unity, especially for higher bands. You can see that it get closer to unity for larger QGcut_psi, though it does not reach to unity because of some contribution of the higher angular momentum contribution within MT. [Values of `Onsite(phi)` are not correctly shown in the case when you use local orbital.]

Due to historical reason, data in vxc.si and exec.si and others contains duplicated data.

8.4 lmf2gw

All the required information are stored into DATA4GW_V2 and CphiGeig.

Input files

- gwa.si :
- gwb.si :
- gw1.si :
- gw2.si :
- Q0P :
- CLASS :
- NLAindx :

Output files

- DATA4GW_V2 : (bin) Main data for GW calculations.
  I/O of DATA4GW_V2 is controlled by `gwinput.f`, which contains detailed information.
- CphiGeig : (bin) Eigenfunctions for GW calculations.
- VXCFP.chk : Eigenvalue and Vxc check (only used for check)
  It is like this;

```
### LDA exchange correlation ###
#   qvec              ikp iband    eigen        VXC(ntotal)    VXC(nvalence)      eigen(eV)    VXC(ntotal)(eV)  VXC(nvalence)
0.0000  0.0000  0.0000  1  1     -0.68505346     -0.91850436      0.00000000     -9.32070032    -12.49698668      0.00000000
0.0000  0.0000  0.0000  1  2      0.19292662     -0.99853478      0.00000000      2.62492096    -13.58586453      0.00000000
0.0000  0.0000  0.0000  1  3      0.19292763     -0.99853469      0.00000000      2.62493477    -13.58586334      0.00000000
0.0000  0.0000  0.0000  1  4      0.19292777     -0.99853461      0.00000000      2.62493664    -13.58586222      0.00000000
 ...
```

Here `VXC(nvalence)` is not used now. The eigenvalue in `eigen` is in Ry.

— This is the end of the preparation stage. —

From here, the main stage.

8.5 rdata4gw_v2

— Read DATA4GW_V2 and some files, and decompose it into files required in the following *GW*steps. (checked! dec2014)

Input files

- GWinput :
- DATA4GW_V2 :
- CphiGeig :
- QGpsi :
- QGcou :
- Q0P :
- QIBZ :
- SYMOPS : points group operations.

Output files

- hbe.d : data size
- Core_ibas*_l*.chk : core eigenfunctions just for check.
- LMTO : basic date for the crystal.
- EValue : (bin) valence eigen value
- ECORE : core data and core eigenvalues
- CPHI : (bin) Coefficients of eigenfunctions as for the atomic-like argumentation waves in MTs'.
- GEIG : (bin) Coefficients of eigenfunctions as for IPW.
- PHIVC : (bin) All the radial functions.
- @MNLA_CPHI : index set for CPHI. This is not refereed just a check write.
- @MNLA_core : index set for core. This is not refereed just a check write.
- VXCFP : (bin) this is for diagonal elements of $V_{\rm xc}^{\rm LDA}(n_{\rm total})$.
- PPOVLI.* : (bin) Overlap matrix of IPW. xxxxxxxxxx
- PPOVLG.* : (bin) PPOVLG Overlap matrix xxxxxxxxxxoxf IPW. not exactly the the overlap matrix. see around line 500 in rdata4gw.m.f
- PPOVL0 : (bin) xxxxxxxxxxxxx
- HVCCIN : (bin) Required inputs for hvccfp0. Information in this files.
- NQIBZ : q point info. Only used for parallel test mode.
- normchk.dia : Norm check. These numbers should be almost the same as those in normchk.si

These files are input for the following steps. The name of file *foo*U means that it relates to up-spin. We have *foo*D files in the case of spin-polarized calculation with `nspin=2`.

8.6 echo 1|heftet

— Get the Fermi energy EFERMI by tetrahedron method. It is used in **hx0fp0**.

Input files

- EVU :
- BZDATA :
- GWinput :
- ECORE : (dummy)
- SYMOPS : (dummy)
- LMTO :
- hbe.d :

Output files

- EFERMI : contains Fermi energy given by the tetrahedron method. It is used in **hx0fp0** but not in **hsfp0**.
- DOSACC.lda,DOSACC2.lda : They are lists of the all the eigenvalues from the bottom. DOSACC2.lda is a list to show only the un-degenerated eigenvalues. They are just check write. But it is an indicator for you to determine `esmr` in GWinput.

8.7 hchknw

— Calculate the required number of ω points along real axis.
This NW is not essentially used in `gw_lmfh` (but required as a dummy file). Only used in `gw_lmf`.

Input files

- BZDATA :
- GWinput :
- ECORE : (dummy)

• SYMOPS : (dummy)

Output files

• NW : contains number of ω points.

8.8 echo 3|hbasfp0

— Make product basis.

Mode 3 is for the core states. It generate a product basis on each MT suitable to expand to calculate the exchange part due to core. See explanations for the input file of GWinput.

Input files

• LMTO :
• PHIVC :
• GWinput :

Output files

• BASFP* : (bin) Product basis functions
• PPBRD_V2_* : (bin) Radial integrals on each MT, symbolically written as $\int \phi(r)\phi(r)B(r)dr$
• PHIV.chk : Valence radial functions (for check).

8.9 echo 0| hvccfp0

— Calculate the Coulomb matrix in the Mixed basis

Input files

• HVCCIN :
• Q0P :
• BASFP* :

Output files

• VCCFP : The Coulomb matrix expanded in the mixed basis
• Mix0vec : This is used only for dielectric-constant calculation (mode 2 or 3 of **hx0fp0**). This contains the expansion of the plane wave $\exp(i\mathbf{qr})$ in the mixed basis. See Usuda's note.

8.10 echo 3|hsfp0

— Exchange part of the self-energy for the core

Input files

• GWIN_V2,LMTO,ECORE :
• CLASS :
• hbe.d :
• Q0P :
• PPBRD_V2_* :
• CPHI :
• GEIG :
• VCCFP :
• PPOVL :

Output files

• SEXcoreU : The core part of the exchange self-energy for **q** and band index specified in <QPNT>. See **??**.

8.11 echo 0|hsfp0

— Make product basis for the valence part.

8.12 echo 1|hsfp0

— Exchange part of the self-energy for the valence part.

Output files

• XCU : The LDA exchange self-energy for **q** and band index specified in <QPNT>. See **??**.

• SEXU : The valence part of the exchange self-energy for **q** and band index specified in <QPNT>. See **??**.

8.13 echo 11|hx0fp0

— Screened Coulomb interaction W(Sergey mode)

Input files

• GWinput, LMTO, ECORE, EVU :

• NW : dummy

• hbe.d :

• Q0P :

• PPBRD_V2_* :

• CPHI,GEIG :

• PPOVL :

• VCCFP :

• ANFcond : (optional) This file is to specify antiferro condition. This should not exist for other cases. This file should be given by hand.

Output files

• WV.d : size of the dielectric function

• WVR : (bin) $W - v$ in the expansion of mixed basis along the real axis

• WVI : (bin) $W - v$ in the expansion of mixed basis along the imaginary axis

8.14 echo 12|hsfp0

— Correlation part of the self-energy(Sergey mode)

Input files

• GWinput, LMTO, ECORE, SYMOPS : These are readin by `genallcf_v3`.

• CLASS, hbe.d, EVU, Q0P :

• PPBRD_V2_* :

Radial integrals on each MT, symbolically written as $\int \phi(r)\phi(r)B(r)dr$. These are generated by hbasfp0.

• CPHI,GEIG :

• PPOVL :

• WV.d, WVR, WVI :

Output files

• SECU : The correlation part of the self-energy for **q** and band index specified in <QPNT>. See **??**.

8.15 echo 0|hqpe

— Summarize the output

Input files

• SEXcoreU,XCU,SEXU,SECU : See **??**.

Output files

• QPU : The QP energies and related value summary in human interface. See **??**.

• TOTE : The detailed values of the QP energies. See **??**.
• TOTE2 : The detailed values of the QP energy. See **??**. This is used for **bndplot**.

NOTE: For example, if you do `{echo 4$|$hqpe}{hqpe}`, it just shift zero level of QPE, so that 4th line (counted from top) eigenvalue (in QPU) is to be zero.

9 Linear response calculations

With these scripts for linear response calculations, `eps*`, we can calculate **q**-dependent dielectric funciton $\epsilon(\omega, \mathbf{q})$ (and v, W) (and χ for spin fluctuation). But (because of numerical reason), we can not use $\mathbf{q} = 0$ limit. (if $|\mathbf{q}|$ is too small, we have numerical problem, zero divided by zero, because we have not implemented the version to use $\mathbf{q} = 0$.)

- `eps_lmfh`
  Dielectric function epsilon with local field correction. Expensive calculation (we may need to reduce number of wing parts in future...).

- `epsPP_lmfh`
  epsilon without local field correction. $1 - \langle e^{i\mathbf{qr}}|v|e^{i\mathbf{qr}}\rangle\langle e^{i\mathbf{qr}}|\left(\chi^0\right)|e^{i\mathbf{qr}}\rangle$

- `epsPP_lmfh_chipm`

  For spin susceptibility. This essentially calculate non-interacting spin susceptibility. Then it is used for the calculation of full spin susceptibility with `util/calj_*.F` programs (small quick programs). See spin wave paper. See spin susceptibility section Sec.**??**.

- (not maintained now; we will recover this)`eps_lmfh_chipm`

  This gives full non-interacting spin susceptibility. Testing. We have to determine U (Stoner I) for the determination of full spin susceptibility. TDLDA? or so?

- `(This is old mode --- not maintained) epsPP_lmfh_chipm_q`

  For spin susceptibility, spin susceptibility $\langle e^{iqr}|\chi(q,\omega)|e^{iqr}\rangle$ In this script, You have to assign that isp=1 is majority, isp=2 is minority. This is with long wave approximation.

• We use the histogram method (the Hilbert transformation method); we first calculate its imaginary parts with the tetrahedron technique for dielectric functions. Then we get its real part by the Hilbert transformation.
You need to choose `HisBin_dw,HisBin_ratio`. The width of histogram bins are getting larger when omega gets larger. dw is the size of histogram-bin width at omega=0. At omega=omg_c, its width gets twiced.

To plot dielectric function with reasonable resolution, it might be better to set `dw 0.001` and `omg_c 0.1` for example. You may have to choose small enough omega for spin wave mode as 0.001 Ry (Or smaller). omg_c is given like 0.05 Ry or so. But sometimes it can be like 1Ry.
• `epsPP_lmfh` only calculation an matrix element of dielectric funciton for $exp(i\mathbf{qr})$. Thus very faster than `eps_lmfh` mode. It uses a a special product basis set for cases without inversion (problem is in how to expand exp($i\mathbf{qr}$) in the MPB; the product basis is not from phi and phidot, but from spherical Bessel functions).

```
In *_lmfh_* modes( I now use little for *_lmf_* modes), you can use small enough delta.
Use small enough delta (=-1e-8 a.u.) for spin wave modes (also you can use it for
dielectric function and GW).  This is necessary because pole is too smeared
if you use larger delta.
```

9.1 eps_lmfh, epsPP_lmfh: the dielectric functions

You can invoke the script, e.g. as "`eps_lmfh si`".

Specify **q** point in `<QforEPS>` or so. Mesh for ω is specified by `dw, omg_c`.

The obtained data are in EPS*.dat and EPS*.nlfc.dat. EPS*.nlfc.dat contains the result without local-field correction EPS*.dat contains the result with local-field correction (this is generated only for `eps_lmfh`. Both of them contains

q(1:3), ω, Re(ϵ) Im(ϵ), Re($1/\epsilon$), In($1/\epsilon$)
in each line.

9.2 epsPP_lmfh: the dielectric function(No LFC— faster)

You can calculate ϵ without LFC by **epsPP_lmfh**. It is very faster than `eps_lmfh`.

To calculate $\epsilon(\mathbf{q}, \omega)$ without LFC accurately, the best basis set for the expansion of the Coulomb matrix within MT is apparently not the product basis, but the Bessel functions corresponding to the plane waves $\exp(i\mathbf{qr})$. We use such a basis in this mode. However, our experience shows that the changes are little even with the usual product basis (we don't describe this here).

9.3 How to calculate correct dielectric funciton?

(this subsection is essentially OK... but need to clean it up. dec2014)

```
There are prolems to calculate correct epsilon.
At first, we talk about epsPP_lmfh, which is No LFC. Main problem are

-----------------
1.Convergence for number of k point(specified by n1n2n3).
  Roughly speaking, 20x20x20 is required for not-so-bad results for Fe and Ni.
  It is better to do 30x30x30 to see convergence check.
  However, in the case of ZB-MnAs (maybe because of simple structure around Ef),
  it requires less q points.

  figs are for GaAs.
  fig001: n1n2n3 convergence for Chi_RegQbz = on  case.
  fig002: n1n2n3 convergence for Chi_RegQbz = off case.
  (Chi_RegQbz in explained in General section in this manual).

  As you see, k points convergence looks a little better in Chi_RegQbz=off
  (mesh not including gamma). However a little ploblem is that its thereshold around
  0.5eV is too high and slowly changing.

  fig003: Alouanis'(from Arnaud)  vs. ``Chi_RegQbz = on'' vs. ``Chi_RegQbz = off''
  As you see, the threshold of the Red line (20x20x20 Chi_RegQbz=on) and Alouani's
  are almost the same, but the red line is too oscilating at the low energy part.
  On the other hand, ``Chi_RegQbz = off'' in Green broken line is not so satisfactory
  at the low energy part.

  fig.gas_eps_kconf.pdf shows the convergence behevior of epsilon for
```

2.$q \to 0$ convergence (this is related to whether Chi_RegQbz=on or off).
  If you use very small q like q=0.001 is GaAs, it can cause a problem.
  Use q=0.01 or larger (maybe q=0.02 or more is safer).
  Very small q can give numerical error for high-energy region.

  In fig004, we show the high energy tail part of Im $\epsilon(\omega)$ for GaAs case.
  At q=0.01 (this means q= 2*pi/alat * (0 0 0.01)), the imaginary part
  is a little too large . Less than 80eV, q=0.02 gives good results when compared with
  other high q results, though it still has noise above 80eV.
  In fig005, I showed the same results compared with Alouani's (his is up to 40eV).
  Both gives rather good agreements. As you see, q=0.06 or above might be necessary
  to get reasonable convergence for high energy part abouve 40eV.

  We have to be careful for this poorness in high energy part--- it may effect
  low-energy Re[$\epsion$] through KK relation. However this can be very small
  ehough.
  In fig.gas_eps_qconv.jpg, we checked the convergence of eps (\omega=0,q) for q \to 0.
  As you see, it gives convergence, however, q=0.01 is a little out of
  curve---this should be because of the poorness in the high energy part.
  so q=0.02 or q=0.03 is safer, and you can get eps within 1 percenr accuracy.

3. Including Core for dielectric constant is dangerous.
   It can cause very poor results if you include core part in GWinput.
   You need to include core just as valence (with local orbital).

   In fig008, we showed core effects. It starts from \approx 16eV
   (this is core to conduction transition).
   fig007 showd the check about the q point dependence---even with large q,
   it would not change.
   These shows that the core excitation can have larger energy range.
   This is in contrast to the valence case
   (then the most of excitaion is limited to less than 10eV).
   We have to be careful for such high-energy exciation... The LMTO basis might
   be not so good for high energy.

4. basis set.
   Use QpGcut_psi \approx 3.0 a.u. or so (as same as GW calculation).
   In the case of epsPP* mode,
   QpGcut_cou can be very small--- In our codes now,
   ngc>=1 should be for all q vector shown in lqg4gw02 (output of echo 2|qg4gw).
   [In principle, it should be only for the q vector for which we calculate epsilon.
    But there is a technical poorness in our code---
    (maybe) a problem here; the plane-wave part of the eigenfunction generated
    in lmfgw is not correctly passed to lmf2gw when ngc=0].

```
-- eps_lmfh: including LFC ----------------------------------
To include eps with LFC, do eps_lmfh.
But lcutmx=2 seems to be good enough to get 0.5 percent error (maybe better than this).
Test it 10x10x10 or so. (I need to repeat if necessary).
Further you can use smaller QpGcut_cou like 2.2 or so,
with rather smaller product basis (up to p timed d, not including f).

Note: epsPP_lmfh is designed to use good basis to calculate eps
without LFC. This is usually in agreement with what you obtained by eps_lmfh;
however it can give slight difference when you use small product basis.


---Summary --------------------
So in conclusion, I think a best way to do is

1. set q=0.02 [q=2pi/alat(0 0 0.02)] or so for GaAs case.
   If you want to check, do q=0.03 and q=0.06 also.

   ``Chi_RegQbz = off'' is better for matrials like GaAs with direct gap.

2. You can use small QpGcut_cou but all ngc should be one or more.

3. As for the Product basis setting in epsPP* scripts, only
   lcutmx and tolerance (this can be like 0.001 or so) are relevant.to determine eps(omega=0, q
   E.g. set lcutmx=4 or so.

   5. To get eps with LFC, set QpGcut_cut as xxx, and set lcutmx=2 where
   4. Do nk=20 18 16 and take interpolarion

   (occupied sp) \timex (unoccupied spd) are included.
   But correct EPS*.nolfc.d is rather from epsPP_lmfh script.
```

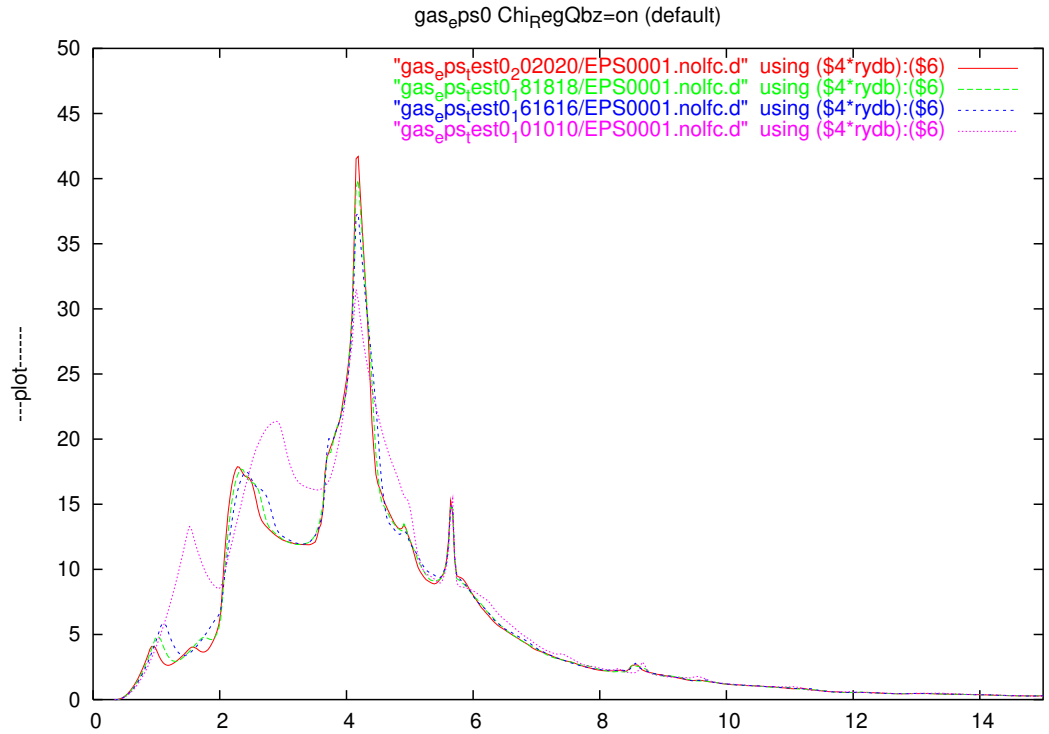

fig001

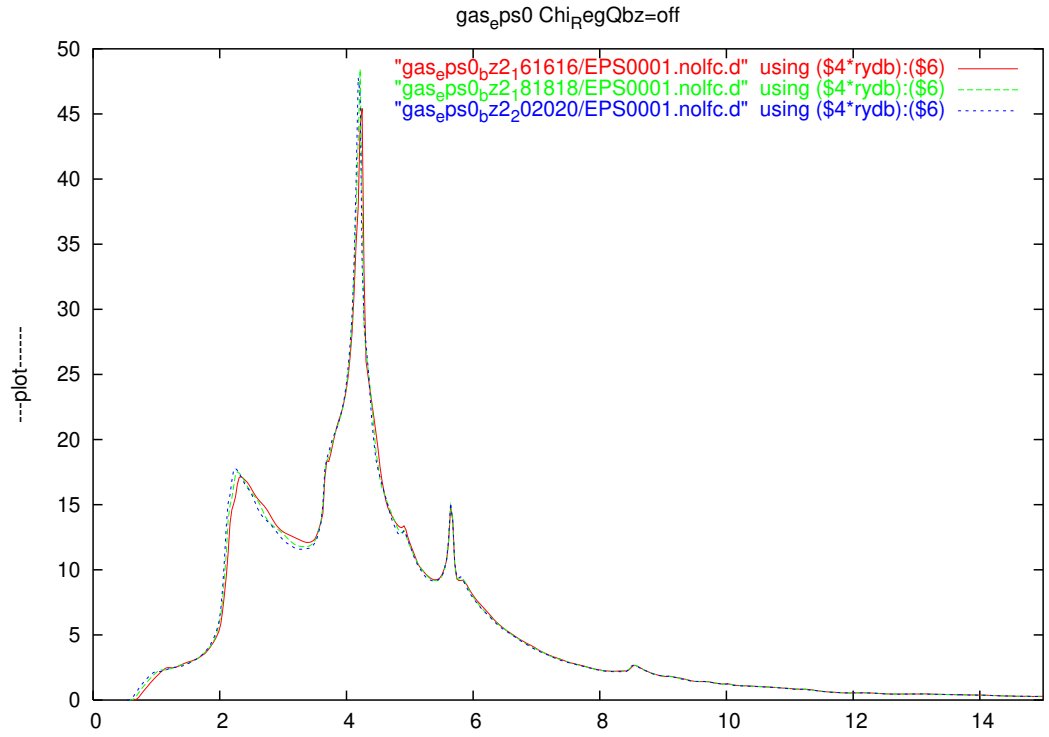

fig002

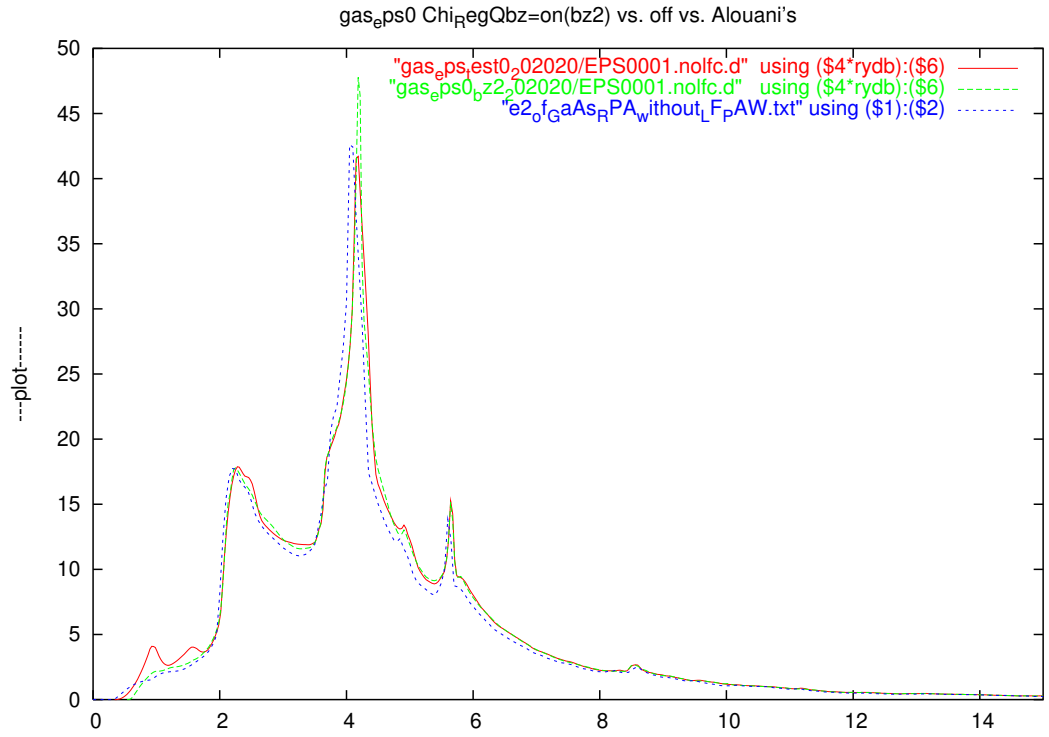

fig003

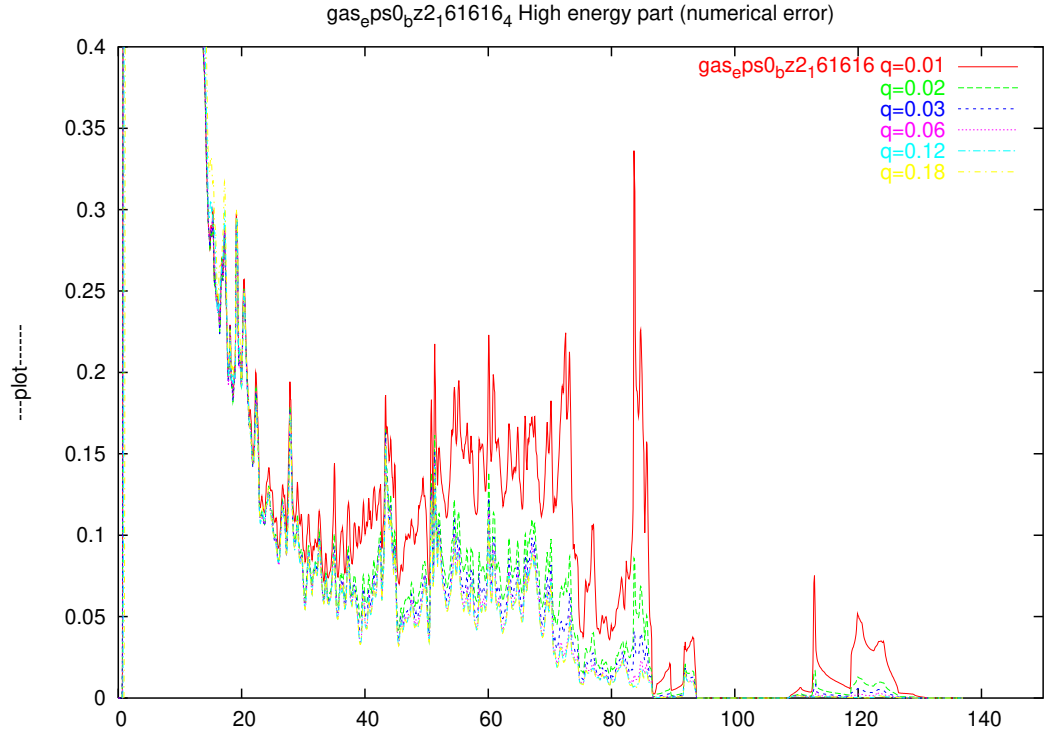

fig004

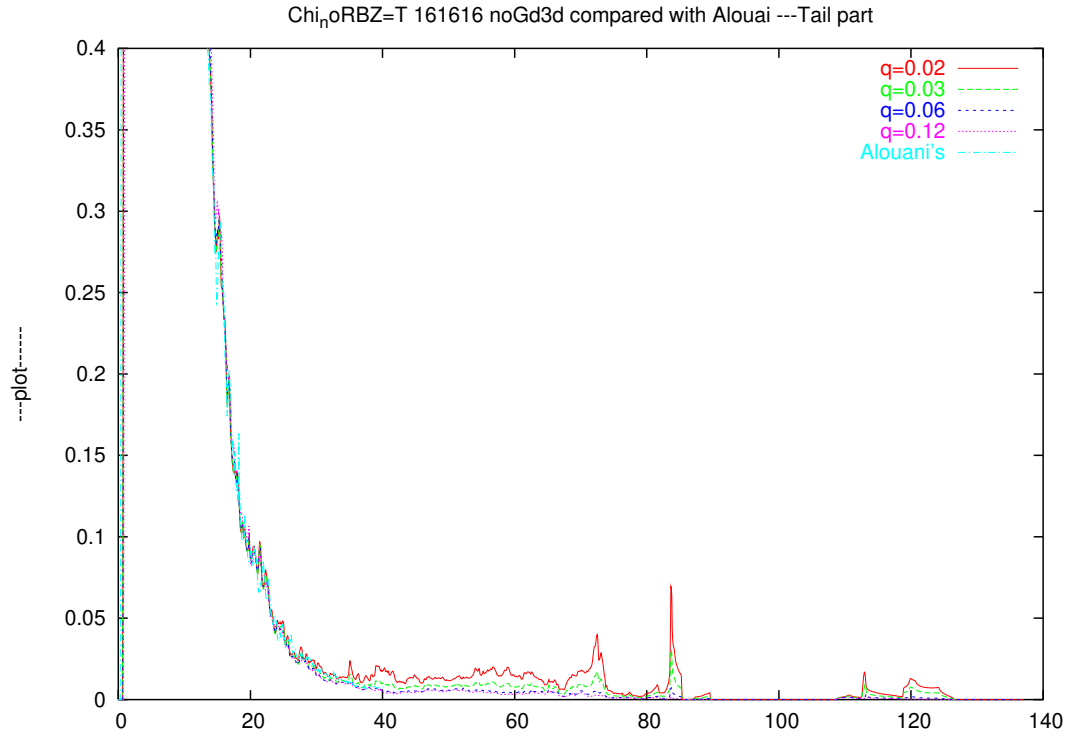

fig005

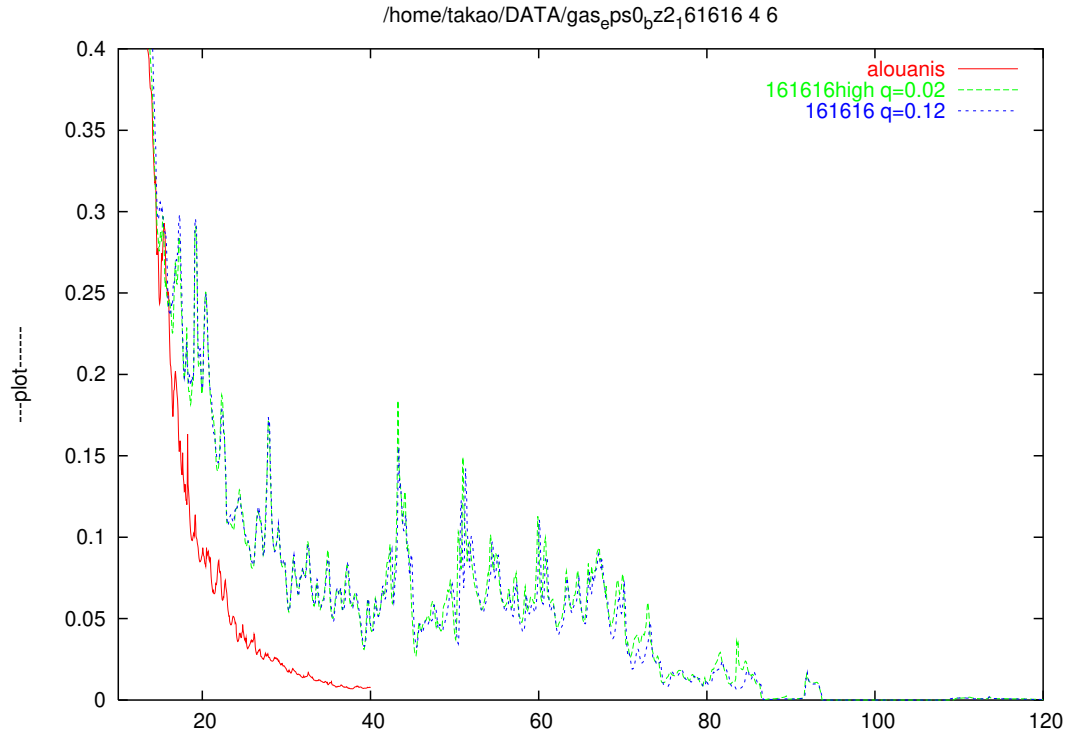

fig007

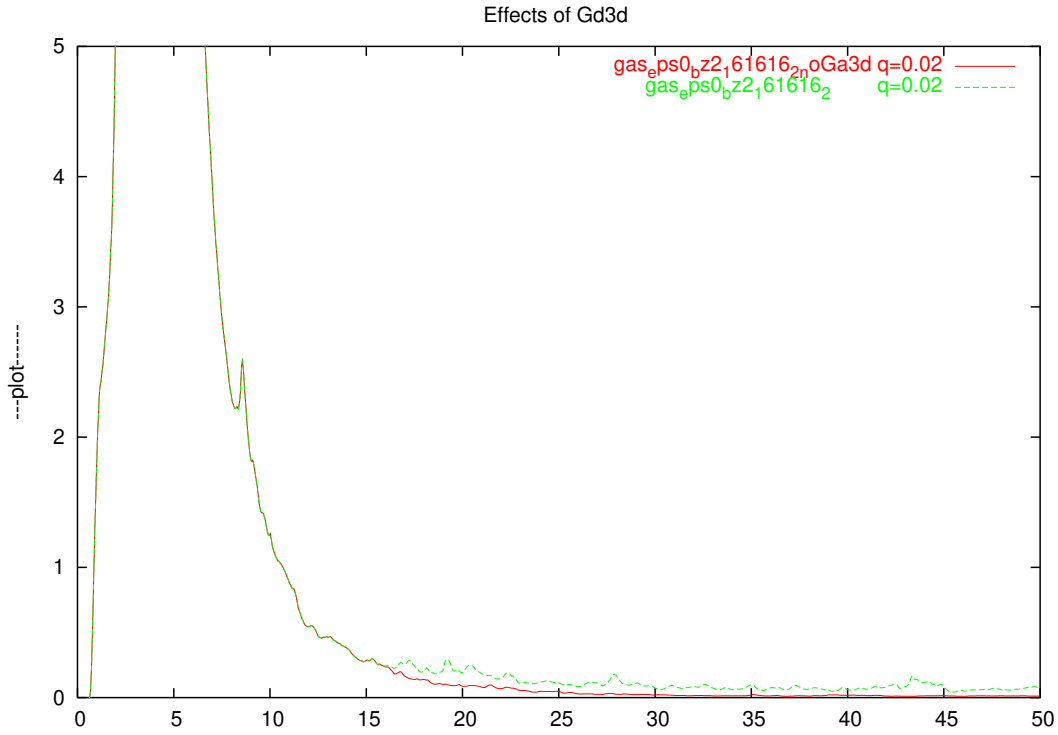
Effects of Gd3d
gas$_e$ps0$_b$z2$_1$61616$_{2n}$oGa3d q=0.02
gas$_e$ps0$_b$z2$_1$61616$_2$ q=0.02
---plot---
0
1
2
3
4
5
0
5
10
15
20
25
30
35
40
45
50

fig008