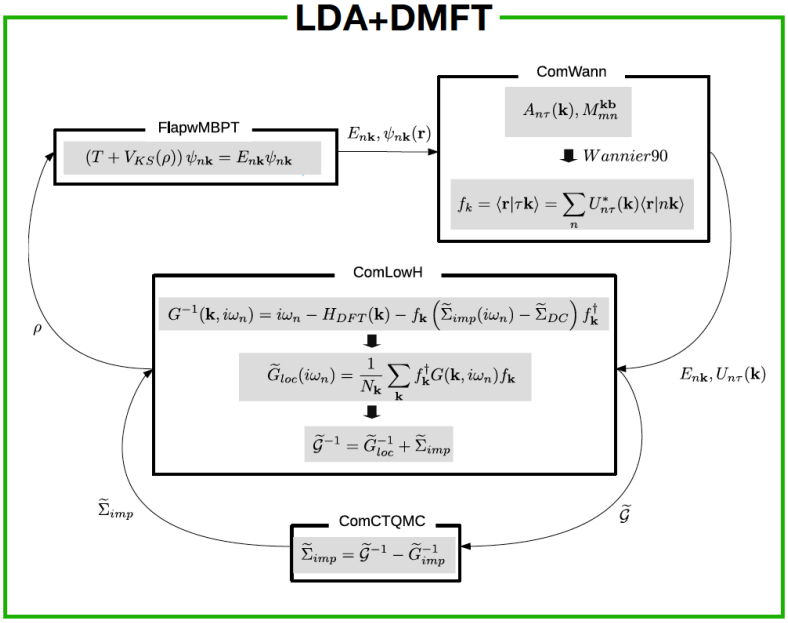
Example: FeSe

LDA+DMFT in COMSUITE

We will calculate the electronic structure of FeSe within LDA+DMFT. COMSUITE package for LDA+DMFT is composed of four components (softwares). Its work flow is described below.



1. Construction of a Kohn-Sham Hamiltonian within LDA by **FlapwMBPT** (see https://doi.org/10.1016/j.cpc.2017.06.012)
2. Construction of the atom-centered local basis set spanning the low energy Hilbert space by **ComWann utilizing Wannier90 package**
3. Wannier interpolation of the mean-field Hamiltonian and solving the DMFT self-consistent equation by **ComLowH** and **ComCTQMC**
4. Updating the electron density

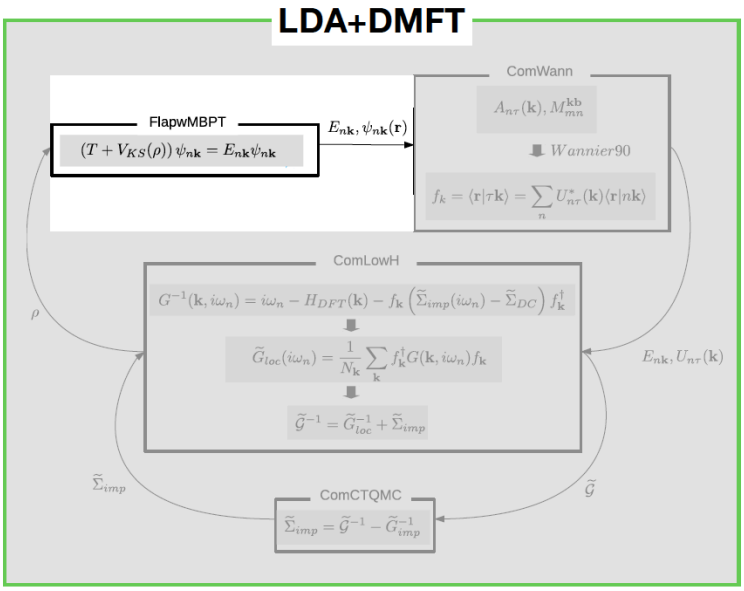
These components (software) along with Wannier90 library are located at install_directory ('install_directory' is described in 'Build and Install' section in Installation of COMSUITE page). To access COMSUITE executables, you should export bin path in your startup shell script.

```
export COMSUITE_BIN=install_directory/bin
```

This tutorial consists of three parts: LDA prerun, LDA+DMFT run, and analysis.

FeSe LDA prerun

To run LDA+DMFT, we start with the LDA prerun (unshaded part of following figure).



To start the LDA calculation, you need to create a directory named "dft" (or a name of your own choice). Note that this name should be specified in 'comdmft.ini' as will be explained in the next section. Having done this, move to the created directory:

```
$ mkdir dft
$ cd dft
```

Create an input file "comdmft.ini". This input file should be written in python dictionary format. All dictionary keys are in small letters. 'comdfmt.ini' is composed of two python dictionaries: 'control' and 'flapwmbpt'.

```
control ={
      'method':'dft',
      'mpi_prefix':'srun -n 32',
      'nproc_k_flapwmbpt':32,
      'restart':False
      }

flapwmbpt={
      'cif':'./FeSe.cif',
      'iter_dft':100,
      'dft_mix':0.1,
      'rel':1,
      'magn':False,
      'kmesh':[6, 6, 4]
            }
```

In Control

These fields contain basic parameters related to MPI setup and methodology.

- method: ab intio methodology. Set to "dft".
- mpi_prefix: MPI prefix used for FlapwMBPT dft run. Note that 32 is the number of total processes we want to use for the present case.
- nproc_k_flapwmbpt: The number of MPI processes associated with the crystal momentum vectors. Set to 32. For the details on the MPI parallization in FlapwMBPT, please go to FlapwMBPT homepage (https://www.bnl.gov/cmpmsd/flapwmbpt).
- Restart: option to resume dft calculation from a checkpoint. Default: False

In Flapwmbpt

- cif: the path to the cif file contains crystal structure information

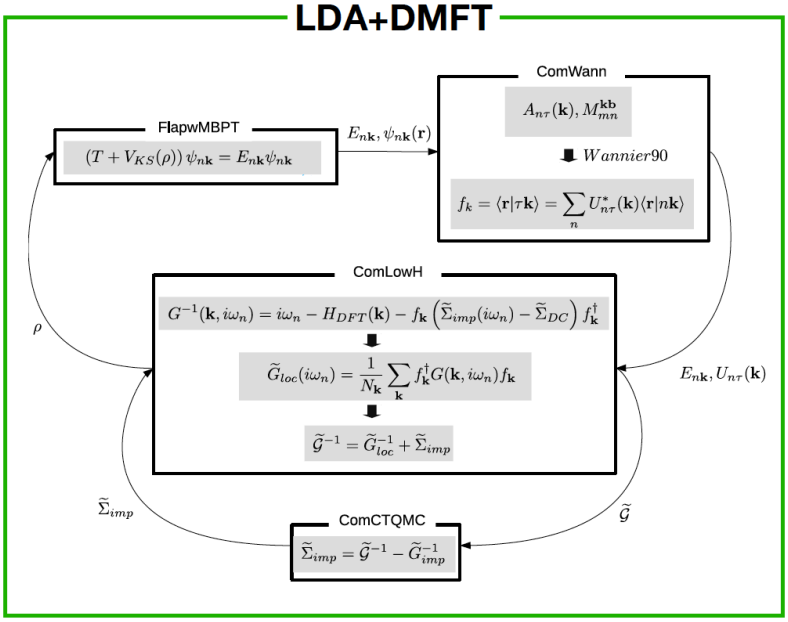
- iter_dft: the total number of dft iteration
- dft_mix: linear density mixing coefficient.
- rel: relativisity option (0: nonrelativistic, 1: scalar relativistic, 2: fully relativistic)
- magn: magntic order
- kmesh: k point grid

The next step is to run LDA by executing rspflapw.exe. An example of job script to run rspflapw.exe using SLURM is

```
#!/bin/bash -l
#SBATCH -J temp
#SBATCH -p debug
#SBATCH -N 1
#SBATCH -e temp.%j.err
#SBATCH -o temp.%j.out
#SBATCH -L SCRATCH
#SBATCH -C haswell
#SBATCH -t 00:30:00
$COMSUITE_BIN/comdmft.py
```

FeSe LDA+DMFT run

Once the prerun is finished successfully, the next step is to run the LDA+DMFT calculation (the entire part of the figure below). To run LDA+DMFT, the input file named "comdmft.ini" is needed (see input file section). The calculation of FeSe within LDA+DMFT reads output data from the LDA prerun. If you specify the prerun path in comdmft.ini (e.g., '../lda' in this case) correctly, it will read necessary data automatically



To run LDA+DMFT, move to your work folder, then create lda_dmft folder (you can name what you want) and move to the folder as follows:

```
$ cd ..
$ mkdir lda_dmft
$ cd lda_dmft
```

Then create comdmft.ini (see input file section) file for LDA+DMFT calculation and execute 'comdmft.py' python file in $COMSUITE_BIN. An example of job script using SLURM is

```
#!/bin/bash -l
#SBATCH -J temp
#SBATCH -q regular
#SBATCH -N 2
```

```
#SBATCH -e temp.%j.err
#SBATCH -o temp.%j.out
#SBATCH -L SCRATCH
#SBATCH -C haswell
#SBATCH -t 12:00:00
$COMSUITE_BIN/comdmft.py
```

Here 'comdmft.py' is a python script which controls LDA+DMFT calculation. Based on the 'comdmft.ini' input file, 'comdmft.py' generates all necessary input files to run individual programs and execute jobs.

Input file (comdmft.ini)

In order to perform LDA+DMFT calculation, we need only a single input file 'comdmft.ini'. This input file should be written in python dictionary format. All dictionary keys are in small letters. 'comdfmt.ini' is composed of three python dictionaries: 'control', 'wan_hmat', and 'imp':

```
control={'initial_lattice_dir'      : '../dft/',
         'method'                   : 'lda+dmft',
       'spin_orbit'                 : False,
         'mpi_prefix': "srun -n 64",
         'impurity_problem':[[1, 'd'],[2, 'd']],
         'impurity_problem_equivalence':[1,1]
       }

wan_hmat={
         'kgrid': [10, 10, 10],
         'froz_win_min': -10.0,
         'froz_win_max': 10.0,
      'local_axis': {1:{'x':[1.0, 1.0, 0.0], 'z':[0.0, 0.0, 1.0]}, 2:{'x':[-1.0, 1.0, 0.0], 'z':[0.0, 0.0, 1.0]}}
      }

imp={'temperature'          : 300, # temperature (in K)
   '1':
   {
    'f0': 5.0,
    'f2': 6.89230769231,
    'f4': 4.30769230769,
    'nominal_n': 6.0,
    'impurity_matrix': [ # equivalent orbital index matrix. starting from 1.
      [1,0,0,0,0],
      [0,2,0,0,0],
      [0,0,3,0,0],
      [0,0,0,2,0],
      [0,0,0,0,4]
      ],
   'thermalization_time': 5,
   'measurement_time': 20,
   'green_cutoff':  10,
   'coulomb': 'full',
   }}
```

■ **In Control**

These fields contain basic parameters to control LDA+DMFT run.

- 'initial_lattice_dir' :
  Enter the path which contains LDA output such as Kohn-Sham eigenvalue and eigenfunctions. It is the LDA prerun folder.

- 'method' :
  Either lda+dmft or lqsgw+dmft. Currently COMSUITE has these two options. Choose 'lda+dmft' for the present work (LDA+DMFT approximation).

- 'spin_orbit':
  Enter True of False. If False, correlated orbitals correspond to cubic spherical harmonics

$$Y_{lm} = \begin{cases} \frac{i}{\sqrt{2}}\left(Y_l^{-|m|} - (-1)^m Y_l^{|m|}\right), & m < 0 \\ Y_l^0, & m = 0 \\ \frac{1}{\sqrt{2}}\left(Y_l^{-|m|} + (-1)^m Y_l^{|m|}\right), & m > 0 \end{cases}$$

  where Y_l^m is a spherical harmonics.
  if True, correlated orbitals chosen at each correlated atom correspond spin-angular functions |l,i,m>

$$\Omega_{l,i=\pm\frac{1}{2},m} = \sum_{s\pm 1/2} C_{i,s}^{l,m} Y_l^{m-s}(\hat{r}) u_s$$

  where u_s is a spinor, and $C_{i,s}^{l,m} = \langle l, m-s, \frac{1}{2}, s | l+i, m \rangle$.

- 'mpi_prefix':
  MPI prefix used for FlapwMBPT, ComLowH, ComWann, and ComCTQMC. If a different prefix is required for individual program, specifiy the number using 'mpi_prefix_lattice','mpi_prefix_lowh', 'mpi_prefix_wannier', and 'mpi_prefix_impurity'. Note that 160 is the number of total processes we want to use for the present case.

- 'impurity_problem':
  A python list to specify correlated orbitals. The first and second indices indicates the atom index and shell type, respectively. Atom index: in the order listed in the "../lda/crystal_structure.xsf". Shell index is either "s", "p", "d" or "f". Here, two Fe-d shells are treated as an impurity problem.

- 'impurity_problem_equivalence':
  Equivalence of each impurity problem. The value is identified by a positive integer starting from 1. If this value is the same, they are equivalent.

- 'restart':
  Enter True or False. If True, it will resume the calculation from the previous LDA+DMFT run. The default value is False.

- 'mpi_prefix_lowh':
  MPI prefix for ComLowH. The default value is the one specified in control['mpi_prefix']

- 'mpi_prefix_impurity':
  MPI prefix for the impurity solver. The default value is the one specified in control['mpi_prefix']

- 'mpi_prefix_wannier':
  MPI prefix for ComWann. The default value is the one specified in control['mpi_prefix'].

- 'sigma_mix_ratio':
  Self-energy linear mixing ratio. You can specify any number within 0.0 – 1.0. The default value is 0.5.

- 'max_iter_num_impurity':
  Maximum iteration for the DMFT self-consistent loop. The default value is 50.

- 'proj_win_min':
  Low-energy cutoff to renormalize the projectors. The default value is the one specified in wan_hmat['dis_win_min']

- 'proj_win_max':
  High-energy cutoff to renormalize the projectors. The default value is the one specified in wan_hmat['dis_win_max']

■ **In wan_hmat**:
These fields define frozen window, disentanglement window, and ab initio calculation from which maximally localized Wannier functions (MLFWs) are constructed.

- 'kgrid':
  Crystal momentum grid for the wannier interpolation of LDA band structure.

- 'froz_win_min':
  Lower boundary of the inner (frozen) window in eV.

- 'froz_win_max':
  Upper boundary of the inner (frozen) window in eV.

- 'dis_win_min':
  Lower boundary of the outer (disentanglement) window in eV. The default value is same with wan_hmat['froz_win_min']

- 'dis_win_max':
  Upper boundary of the outer (disentanglement) window in eV. The default value is wan_hmat['froz_win_max'] +40.0

- 'num_iter':
  The number of minimization step for the wannierization process. (gauge dependent part of total spreading). The default value is 0.

- 'dis_num_iter':
  The number of minimization step for the disentanglement process. (gauge independent part of total spreading). The default value is 100.

- 'local_axis':
  Local cartesian coordinates for each atom. This axis is used to define orientation of wannier functions localized at each atom. For each atom indexed with "atom index", user can provide local x and z axis w.r.t. global cartesian coordinates. If not provided, global cartesian coordinate will be used.

■ **In imp**:
These fields are related with the Monte-Carlo algorithm and sampling of observables.

- 'temperature':
  Simulation temperature in K

- 'F0', 'F2', 'F4':
  The values of Slater integrals in eV. Note that for "f" shells, 'F0', 'F2', 'F4', and 'F6' should be specified.

- 'nominal_n':

  Nominal occupancy associated with the impurity shell. This value is required since we adopt the so-called "nominal double counting" for LDA+DMFT which reads:

$$\tilde{\Sigma}^{DC} = U(N_0 - \frac{1}{2}) - J(\frac{N_0}{2} - \frac{1}{2})$$

  where N_0 is the nominal occupancy specified by 'nominal_n'.

- For each distinct impurity problem indexed by the value in control ["impurity_problem_equivalence"],
  - 'impurity_matrix': [
    [1,0,0,0,0],
    [0,2,0,0,0],
    [0,0,3,0,0],
    [0,0,0,2,0],
    [0,0,0,0,4]

],
Equivalence of the matrix element of the hybridization function and impurity self-energy. Starting from "1", you can set any positive number. If these values are the same, hybridization function and impurity self-energy will be identical for those. If the element in the matrix is zero, then it will not be sampled by the impurity solver. Each column and row correspond to the Wannier orbitals in the following order |xy>, |yz>, |z^2>, |xz>, |x^2-y^2> if control['spin_orbit']==False. If control['spin_orbit']==True, the most rapidly changing index is "m" and the next one is "i". They are sorted in ascending order. For the case of "f" shell, for example, they are ordered as: |3,-0.5, -2.5>, |3,-0.5, -1.5>, |3,-0.5, -0.5>, |3,-0.5, 0.5>, |3,-0.5, 1.5>, |3,-0.5, 2.5>, |3,0.5, -3.5>, |3,0.5, -2.5>, |3,0.5,-1.5>, |3,0.5, -0.5>, |3,0.5, 0.5>, |3,0.5, 1.5>, |3,0.5, 2.5>, |3,0.5, 3.5>,

- 'thermalization_time':
  Wall time for the thermalization in minutes.

- 'measurement_time':
  Wall time for the measurement in minutes.

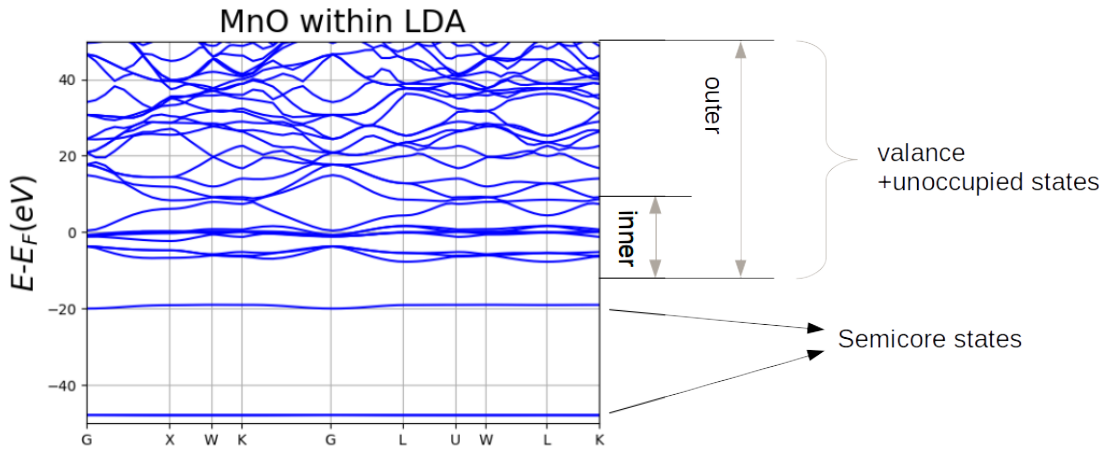
- 'green_cutoff':
  Cutoff-energy in eV to sample green's function and self-energy. The values beyond this energy will be provided by analytical equations.

- 'susceptibility_cutoff':
  Cutoff-energy to sample susceptibility. The default value is 300 eV.

- 'Coulomb': 'full',
  'full' or 'ising' are available. We construct Coulomb matrix in the following way.
  $$U_{m_1,m_2,m_3,m_4} = \sum_{k=0}^{2l,even} \frac{4\pi}{2k+1} F_l^k \sum_{q=-k}^{k} \langle Y_l^{m_1} | Y_k^q Y_l^{m_4} \rangle \langle Y_l^{m_2} Y_k^q | Y_l^{m_3} \rangle$$
  If 'full', no additional approximation is considered. If 'ising', only U_{abba} or U_{abab} are non-zero.

Input file (comdmft.ini)-Important concepts for wan_hmat

COMSUITE uses localized orbitals such as Wannier functions to represent the low-energy Hilbert space. To construct the Wannier functions, the inner (frozen) energy window can be set to range from E_F+'froz_win_min' to E_F+'froz_win_max', and the outer (disentanglement) energy window can range from E_F+'dis_win_min' to E_F+'dis_win_max'; see the figure below (Here we take the MnO case as an example).

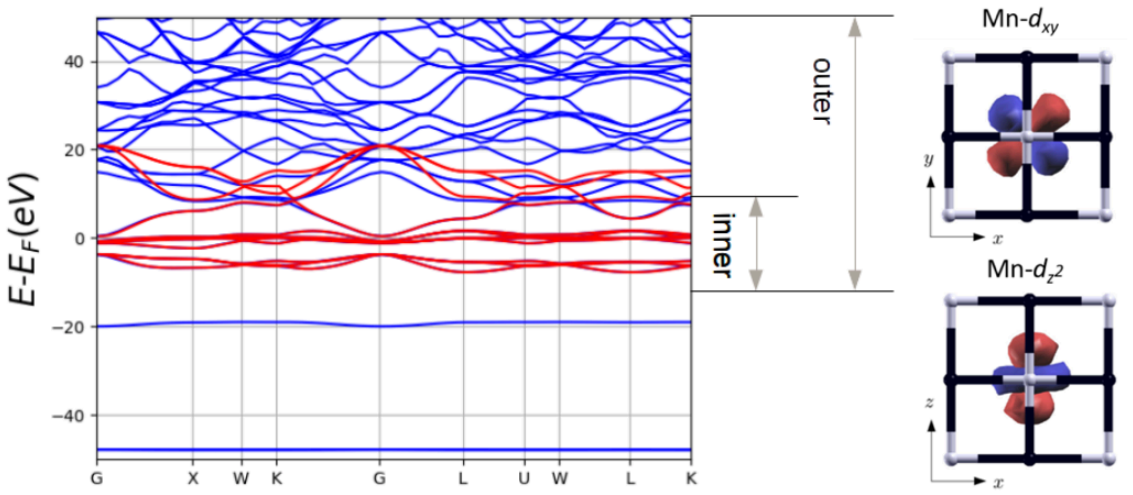


We choose initial trial orbitals $|\tau \mathbf{R} = 0\rangle_t$ using MT orbitals with desired angular momentum character. The radial functions of $|\tau \mathbf{R} = 0\rangle_t$ are chosen in such a way to maximize

$$\frac{1}{N_{\mathbf{k}}} \sum_{n\mathbf{k}}^{E_{min}^{inner} < E_{n\mathbf{k}} < E_{min}^{inner}} |\langle n\mathbf{k} | \tau \mathbf{k} \rangle_t|^2$$

, where $|\tau\mathbf{k}\rangle_t = \frac{1}{N_\mathbf{k}} \sum_\mathbf{R} |\tau\mathbf{R}\rangle_t e^{i\mathbf{k}\cdot\mathbf{R}}$. Among the MT orbitals above, we chosse ones which are larger than 0.15. For correlated orbitals, final wannier functions $|\tau\mathbf{R}=0\rangle_f$ usually satisfy a condition of ${}_f\langle\tau\mathbf{R}=0|\tau\mathbf{R}=0\rangle_t > 0.95$. This means that $|\tau\mathbf{R}=0\rangle_f$ are strongly localized and are regarded as atom-like wavefunctions.

The figure below shows the Wannier functions and interpolated band structure of MnO in comparison with the LDA band structure. The number of bands in the inner window is 10, while the number of bands in the outer window is 29. The number of trial orbitals is 12 (Mn-*s*, Mn-*p*, Mn-*d*, O-*p*).



Output files

COMSUITE places important output files generated from individual programs in the work directory (lda_dmft in this example). The list of files is

cmd.log
convergence.log : convergence log filesig.dat : impurity self-energy
delta.dat : hybridization function

The format of each file and meaning of fields are introduced below. The results of FeSe LDA+DMFT calculation are presented with illustrative plots.

■ convergence.log

step	i_outer	i_latt	i_imp	causality	delta_rho	w_sp_min	w_sp_max	mu	std_sig	n_imp	histo_1	histo_2
wannier	1					0.50677703	2.88818561					
delta	1		1	good				-0.019280458878				
impurity_1	1		1	good					0.287717595956	6.00764	249.733784782	248.017658037
dft	2	1			5.154448e-06							
wannier	2					0.5067774	2.88818893					
delta	2		1	good				0.20448195624				
impurity_1	2		1	good					0.108398183571	6.03352	244.489908363	241.269380924
dft	3	1			0.001663299							
wannier	3					0.50688796	2.88815083					
delta	3		1	good				0.343062784144				
impurity_1	3		1	good					0.0542822907893	6.05166	239.922505082	240.530895385

- i_outer: The scf step number for a given charge density and impurity self-energy
- i_latt: The iteration number for solving Kohn-Sham equation with a given charge density obtained from ComLowH.
- i_imp: The iteration number for solving impurity problem through ComLowH + ComCTQMC
- causality: causality of hybridization function / self-energy
- delta_rho: The norm difference between the current charge density and the one from the previous scf step.
- w_sp_min: minimum spread of the Wannier functions
- w_sp_max: maximum spread of the Wannier functions
- mu: LDA+DMFT chemical potential w.r.t. LDA chemical potential
- std_sig:

$$\sqrt{\frac{\sum_i (\Sigma_i^j(i\omega_n) - \Sigma_i^{j-1}(i\omega_n))^2}{n_\omega n_{orb}}}$$

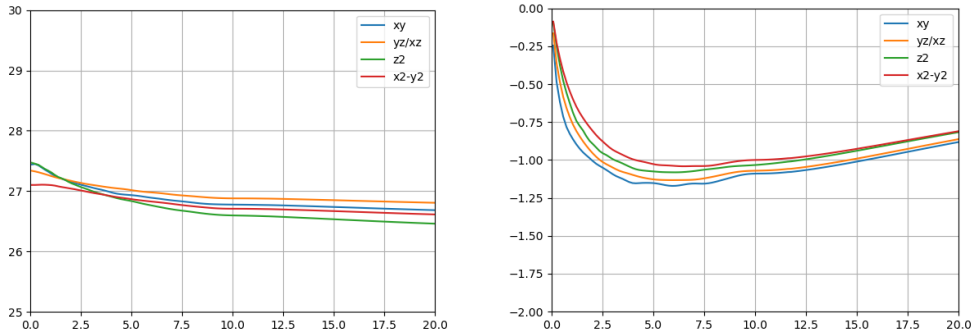
- n_imp: occupation of the impurity orbitals
- histo_1: the first moment of the perturbation order histogram
- histo_2: the second moment of the perturbation order histogram

■ sig.dat

'sig.dat' contains impurity self-energies. The first column lists matsubara frequencies and the next columns are real and imaginary parts of self-energies of Fe d-orbital.

# omega(eV)	Re Sig_{1,1}(eV)	Im Sig_{1,1}(eV)	Re Sig_{1,2}(eV)	Im Sig_{1,2}(eV)	Re Sig_{1,3}(eV)	Im Sig_{1,3}(eV)	Re Sig_{1,4}(eV)	Im Sig_{1,4}(eV)
0.081216424692	27.376953779216	-0.265104324996	27.277842455532	-0.177075773451	27.425998025766	-0.096745581017	27.033042807875	-0.089882158904
0.243649274076	27.377335210723	-0.516027757206	27.263135232851	-0.371165995460	27.409245675634	-0.261729717960	27.035383203165	-0.230922866262
0.406082123461	27.350541973157	-0.651404469022	27.246646297662	-0.496864677447	27.379208055544	-0.386778760189	27.039100192451	-0.336976074057
0.568514972845	27.318305180096	-0.747066591669	27.230117705542	-0.595424684452	27.348947676966	-0.489921729552	27.040624934272	-0.425962173127
0.730947822229	27.283867336136	-0.813375262193	27.213446880534	-0.671093501416	27.316278876584	-0.573829854054	27.038299684794	-0.497785452319
0.893380671613	27.249406344349	-0.860174419442	27.196383033097	-0.731505939544	27.283167710025	-0.642241108661	27.034153928540	-0.558997023966
1.055813520997	27.217499872937	-0.896115537220	27.180008692133	-0.781941528204	27.249932818533	-0.701737027931	27.028489430603	-0.611537107947
1.218246370382	27.189034943431	-0.926890868635	27.164604070524	-0.824741777617	27.216838520564	-0.751693103836	27.020998760780	-0.657759918529
1.380679219766	27.163768431266	-0.954352109219	27.150755271948	-0.861886211823	27.185414567001	-0.792034173701	27.011697700432	-0.698102246840
1.543112069150	27.139979968647	-0.978308633734	27.137567579078	-0.894546386133	27.155791097520	-0.825615737018	27.002172449123	-0.734820401891

Plots of the real and imaginary parts of the self-energies are as follows

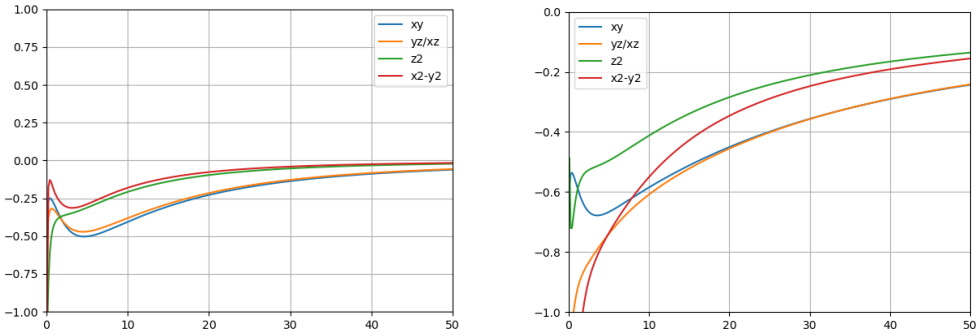


■ delta.dat

'delta.dat' is in the same format of 'sig.dat':

# omega(eV)	Re Sig_{1,1}(eV)	Im Sig_{1,1}(eV)	Re Sig_{1,2}(eV)	Im Sig_{1,2}(eV)	Re Sig_{1,3}(eV)	Im Sig_{1,3}(eV)	Re Sig_{1,4}(eV)	Im Sig_{1,4}(eV)
0.081216424692	-0.164790827849	-0.578253170439	-0.443075833811	-1.072532593788	-1.080496117222	-0.480486426904	-0.933935326792	-3.287178965744
0.243649274076	-0.085975929071	-0.549773510856	-0.248216107607	-1.081679010133	-0.750392962889	-0.711746312504	-0.158153314833	-2.303444685780
0.406082123461	-0.067083525036	-0.545502201394	-0.195149501640	-1.045030341353	-0.573239028216	-0.712120234125	-0.109533071562	-1.838117222560
0.568514972845	-0.069107370488	-0.550932762083	-0.177851525437	-1.007352567113	-0.479545394563	-0.676744763718	-0.129348980065	-1.575849463378
0.730947822229	-0.080363493565	-0.561031721003	-0.175076998543	-0.975652597003	-0.427144505699	-0.642572465938	-0.155879527165	-1.407791078879
0.893380671613	-0.095836308888	-0.572950600252	-0.179547586463	-0.949702618977	-0.395549021259	-0.614562878212	-0.181202705616	-1.290203711398
1.055813520997	-0.113131003935	-0.585297104458	-0.187789380922	-0.928308795299	-0.375319059063	-0.592656137414	-0.203212397537	-1.202523050927

Plots of the real and imaginary parts of hybridization functions are as follows



Analytical Continuation of Self-energy

To obtain the density of states (DOS), we need to perform an analytic continuation of 'sig.dat' to produce the impurity self-energy on the real frequency axis. To do this, we will adopt the maximum entropy (maxent) method. Any publicly available maxent code can be employed. For the purposes of this tutorial, however, we will MQEM code (freely available at

https://github.com/KAIST-ELST/MQEM.jl).

To access maxent code, you should export the path to the executable in your startup shell script.

```
export MQEM =[path to MQEM.jl directory]
```

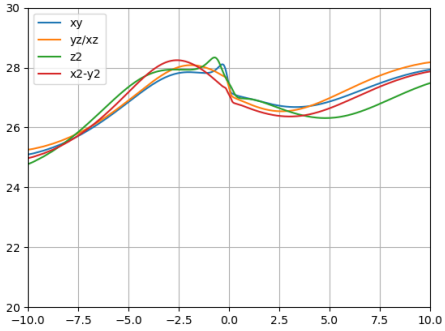
To run the maxent code, move to your working directory, create the maxent directory in the "lda_dmft" directory and then move to it:

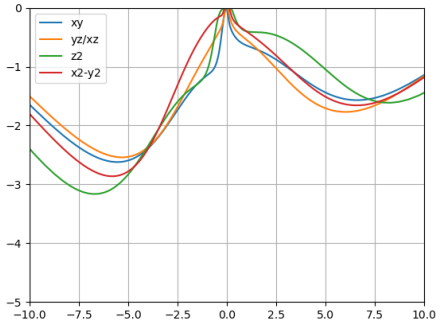
```
$ mkdir maxent
$ cd maxent
```

By executing 'maxent_wrapper.py', we can obtain the self-energy on real axis by automatically calling maxent_run.py:

```
$ $COMSUITE_BIN/mqem_wrapper.py ../sig.dat
```

Plots of the real and imaginary parts of impurity self-energies on the real frequency axis are as follows





LDA+DMFT density of states

To obtain the DOS, we must post-process the data. First, create a directory for the DOS calculation in the "lda_dmft" directory and move to it:

```
$ mkdir realgrid
$ cd realgrid
```

In order to perform DOS calculation, we need only a single input file 'comdmft.ini'

```
control ={
      'method': 'dos',
     'mpi_prefix': 'srun -n 32',
    }

postprocessing={
     'comsuite_dir': '../',
     'self energy': '../maxent/sig_realaxis.dat',
     'kmesh': [15, 15, 15],
     'broadening': 0.1
}
```

Run comdmft.py with job submission script. An example of job script using SLURM is

```
#!/bin/bash -l
#SBATCH -J temp
#SBATCH -q debug
#SBATCH -N 1
#SBATCH -e temp.%j.err
#SBATCH -o temp.%j.out
#SBATCH -L SCRATCH
```

```
#SBATCH -C haswell
#SBATCH -t 00:30:00
$COMSUITE_BIN/comdmft.py
```

Having done so, you will obtain files tdos.dat and pdos.dat.

The format of tdos.dat file is:

# omega (eV)	TDOS (1/eV)
-70.000000000000	0.000195619177
-65.733023713200	0.000220852216
-61.955278682000	0.000247384949
-58.587128546700	0.000275509163
-55.565331497900	0.000305078323
-52.839027896300	0.000336222906
-50.366851286200	0.000368858343
-48.114812211200	0.000403221253
-46.054723948000	0.000439643908
-44.163014879300	0.000479094039

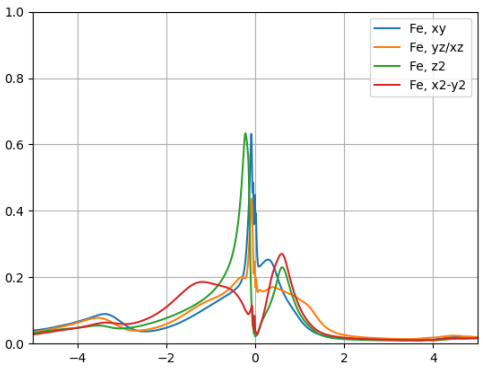
The format of pdos.dat file is:

# omega (eV)	(1,0,0)	(1,1,-1)	(1,1,0)	(1,1,1)	(1,2-2)	(1,2,-1)	(1,2,0)	(1,2,1)
(1,2,2)	(2,0,0)	(2,1,-1)	(2,1,0)	(2,1,1)	(2,2-2)	(2,2,-1)	(2,2,0)	(2,2,1)
(2,2,2)	(3,1,-1)	(3,1,0)	(3,1,1)	(3,2-2)	(3,2,-1)	(3,2,0)	(3,2,1)	(3,2,2)
(4,1,-1)	(4,1,0)	(4,1,1)	(4,2-2)	(4,2,-1)	(4,2,0)	(4,2,1)	(4,2,2)	
-70.000000000000	0.000005939755	0.000005057305	0.000005245784	0.000005057315	0.000006792508	0.000006744313	0.000006834740	0.000006744311
0.000006830156	0.000005939755	0.000005057315	0.000005245805	0.000005057305	0.000006792506	0.000006744311	0.000006834740	0.000006744313
0.000006830156	0.000006469568	0.000006645676	0.000006469557	0.000004628600	0.000004658246	0.000004645276	0.000004658246	0.000004388215
0.000006469545	0.000006645675	0.000006469555	0.000004628616	0.000004658247	0.000004645299	0.000004658247	0.000004388215	
-65.733023713200	0.000006714454	0.000005663130	0.000005884281	0.000005663142	0.000007735440	0.000007693656	0.000007789557	0.000007693654
0.000007779381	0.000006714454	0.000005663142	0.000005884307	0.000005663130	0.000007735438	0.000007693654	0.000007789557	0.000007693656
0.000007779381	0.000007346607	0.000007555772	0.000007346595	0.000005149602	0.000005186301	0.000005168610	0.000005186301	0.000004869603
0.000007346580	0.000007555771	0.000007346592	0.000005149621	0.000005186302	0.000005168637	0.000005186302	0.000004869603	
-61.955278682000	0.000007536253	0.000006297008	0.000006553719	0.000006297022	0.000008740915	0.000008666530	0.000008797795	0.000008666528
0.000008812729	0.000007536253	0.000006297022	0.000006553751	0.000006297008	0.000008740913	0.000008666528	0.000008797795	0.000008666530
0.000008812729	0.000008282317	0.000008528090	0.000008282303	0.000005688591	0.000005733385	0.000005710025	0.000005733385	0.000005365853
0.000008282285	0.000008528090	0.000008282300	0.000005688613	0.000005733386	0.000005710057	0.000005733387	0.000005365853	
-58.587128546700	0.000008405428	0.000006958375	0.000007253559	0.000006958391	0.000009834622	0.000009732416	0.000009891220	0.000009732414
0.000009883403	0.000008405428	0.000006958391	0.000007253596	0.000006958375	0.000009834619	0.000009732414	0.000009891220	0.000009732416
0.000009883403	0.000009277528	0.000009563571	0.000009277510	0.000006244538	0.000006298553	0.000006268481	0.000006298553	0.000005875991
0.000009277490	0.000009563570	0.000009277508	0.000006244563	0.000006298554	0.000006268517	0.000006298555	0.000005875991	
-55.565331497900	0.000009322357	0.000007646758	0.000007983299	0.000007646777	0.000010967621	0.000010864515	0.000011047250	0.000010864513
0.000011046570	0.000009322357	0.000007646777	0.000007983343	0.000007646758	0.000010967618	0.000010864513	0.000011047250	0.000010864515
0.000011046570	0.000010333031	0.000010663112	0.000010333011	0.000006816480	0.000006880890	0.000006842979	0.000006880889	0.000006399075
0.000010332988	0.000010663111	0.000010333008	0.000006816508	0.000006880891	0.000006843021	0.000006880892	0.000006399075	

- (atom index, l, m) if spin_orbit==False and (atom index, I, i,m) if spin_orbit==True.

Plot of pdos.dat:

The total density of states and the projected density of states within charge self-consistent LDA+DMFT.



LDA+DMFT spectral function

To obtain spectral function, we must post-process the data by executing ComLowH again. First create a directory for the spectral function in "lda_dmft" directory and move to it:

```
$ mkdir realaxis
$ cd realaxis
```

In order to perform spectral function calculation, we need only a single input file 'comdmft.ini'

```
control ={
      'method': 'spectral',
     'mpi_prefix': 'srun -n 32',
    }

postprocessing={
     'comsuite_dir': '../',
     'self energy': '../maxent/sig_realaxis.dat',
```

'broadening': 0.01
}

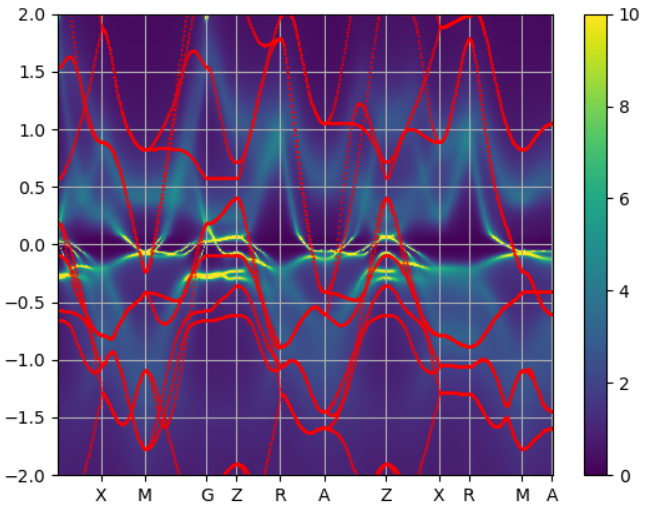
Run comdmft.py with job submission script. An example of job script using SLURM is

```
#!/bin/bash -l
#SBATCH -J temp
#SBATCH -q debug
#SBATCH -N 1
#SBATCH -e temp.%j.err
#SBATCH -o temp.%j.out
#SBATCH -L SCRATCH
#SBATCH -C haswell
#SBATCH -t 00:30:00
$COMSUITE_BIN/comdmft.py
```

Having done so, you will have obtained a file named spectral.dat. The format of this file is

# kpoint	E (eV)	A (1/eV)
1	-70.000000000000	0.000001986280
2	-70.000000000000	0.000001986436
3	-70.000000000000	0.000001986897
4	-70.000000000000	0.000001987642
5	-70.000000000000	0.000001988636
6	-70.000000000000	0.000001989830
7	-70.000000000000	0.000001991164
8	-70.000000000000	0.000001992575
9	-70.000000000000	0.000001993999
10	-70.000000000000	0.000001995379

Plot of spectral.dat file along a high symmetry line in the first Brillouin zone.:



Here the red lines show dft bandstructures.