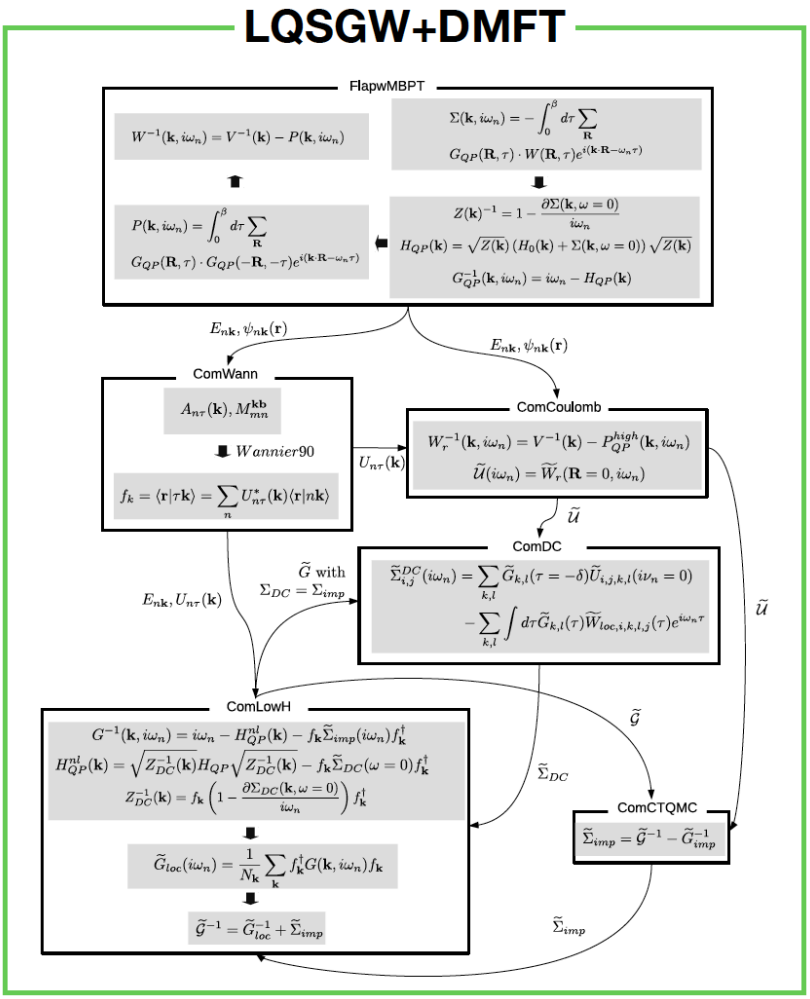
Example: NiO

LQSGW+DMFT in COMSUITE

We will calculate the electronic structure of NiO within LQSGW+DMFT. COMSUITE package for LQSGW+DMFT is composed of six components (softwares) and its work flow is described below.



1. Construction of a quasi-particle Hamiltonian within *ab initio* LQSGW 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. calculation of the bosonic Weiss field within constrained Random Phase approximation(cRPA) and evaluation of Slater's integral associated with the impurity orbitals by **ComCoulomb**
4. calculation of the double-counted self-energy associated with the impurity orbitals within local GW approximation by **ComDC**
5. Wannier interpolation of the mean-field Hamiltonian and solving the DMFT self-consistent equation by **ComLowH** and **ComCTQMC**

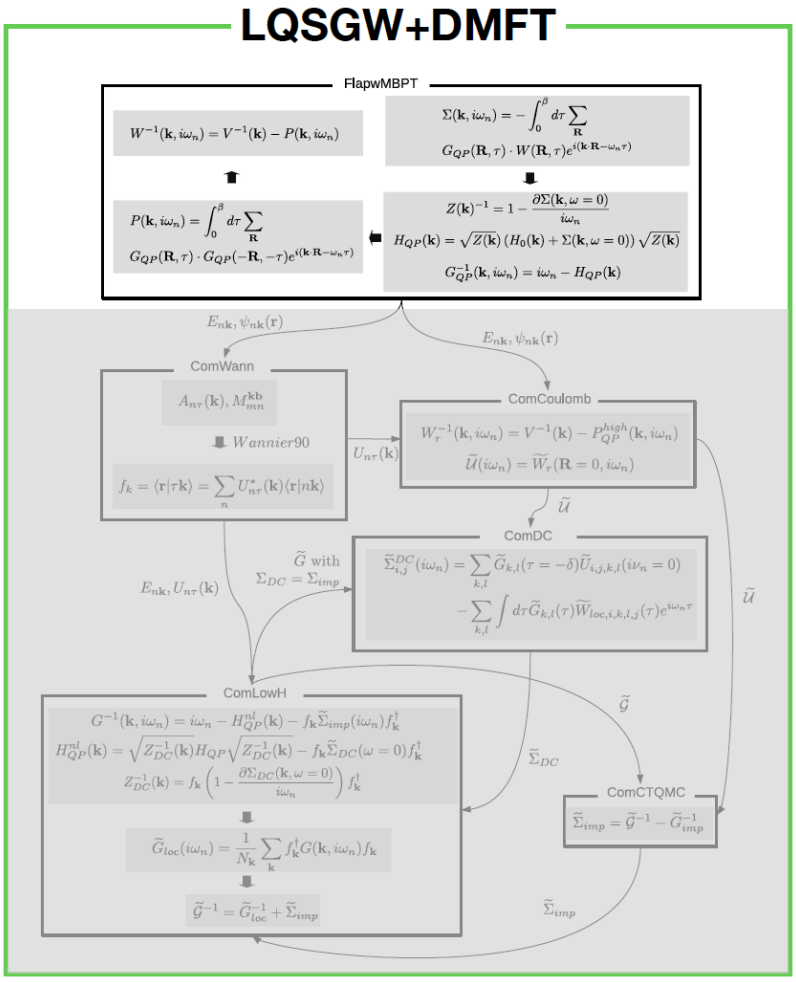
These components (software) along with Wannier90 library are located at install_directory ('install_directory' is described in 'Build and Install' section in Installation of COMSUITE page). To access COMSUITE executables, you should export above bin path in your startup shell script.

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

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

FeSe LQSGW prerun

To run LQSGW+DMFT, the first step is the LQSGW prerun (unshaded part of following figure).



To proceed with the LQSGW calculation, please create a directory named "lqsgw" in your work directory. It is possible to give this directory another name. However, it is important that this name be specified in 'comdmft.ini' (as will be explained in the next section). After doing so, cd to this directory:

```
$ mkdir lqsgw
$ cd lqsgw
```

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':'lqsgw',
      'mpi_prefix':'srun -n 384',
      'nproc_k_flapwmbpt':16,
      'nproc_tau_flapwmbpt': 24
      }

flapwmbpt={
      'cif':'./NiO.cif',
      'dft_mix':0.1,
      'iter_dft': 100,
     'iter_lqsgw': 20,
      '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 "lqsgw".
- mpi_prefix: MPI prefix used for FlapwMBPT dft run. Note that 720 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. For the details on the MPI parallization in FlapwMBPT, please go to FlapwMBPT homepage (https://www.bnl.gov/cmpmsd/flapwmbpt).
- nproc_tau_flapwmbpt: The number of MPI processes associated with the imaginery frequency axis.
- Restart: option to resume dft calculation from a checkpoint. Default: False

In Flapwmbpt

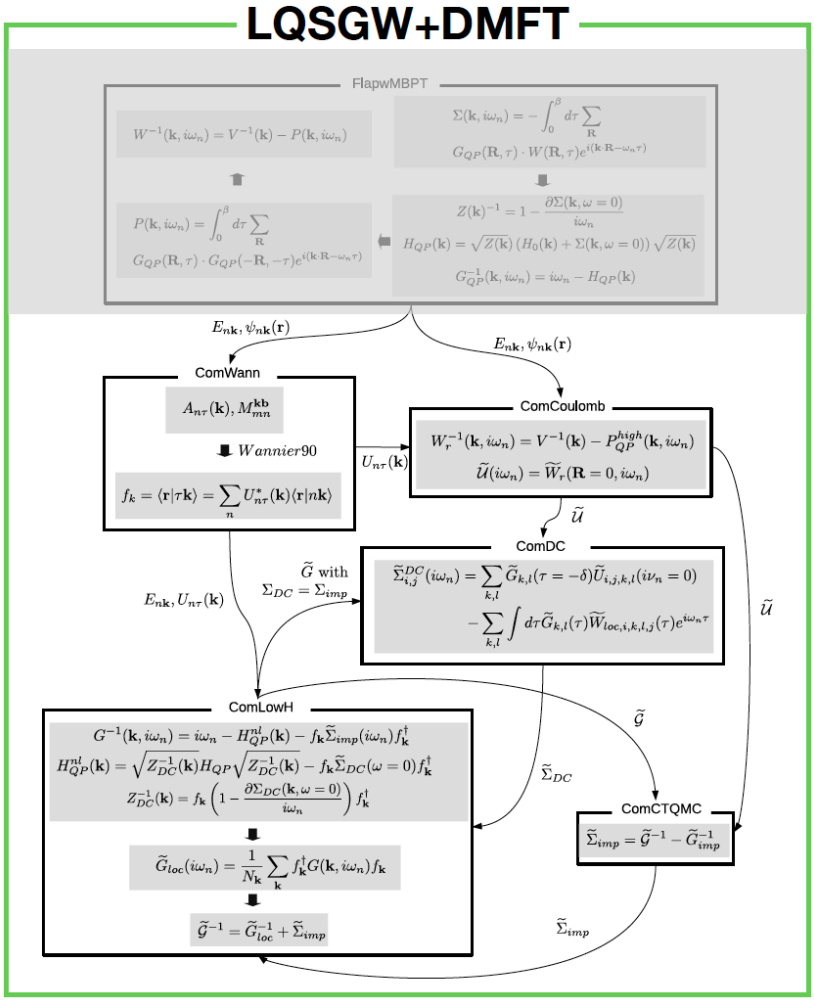
- cif: the path to the cif file contains crystal structure information
- iter_dft: the total number of dft iteration
- iter_lqsgw: the total number of lqsgw iteration
- dft_mix: linear density mixing coefficient.
- rel: relativisity option (0: nonrelativistic, 1: scalar relativistic, 2: fully relativistic)
- magn: magntic order
- kmesh: k point grid

An example of job script using SLURM is

```
#!/bin/bash -l
#SBATCH -J temp
#SBATCH -p regular
#SBATCH -N 12
#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
```

NiO LQSGW+DMFT run

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



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

```
$ cd ..
$ mkdir lqsgw_dmft
$ cd lqsgw_dmft
```

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

```
#!/bin/bash -l
#SBATCH -J temp
#SBATCH -N 12
#SBATCH -e temp.%j.err
#SBATCH -o temp.%j.out
#SBATCH -L SCRATCH
#SBATCH -C haswell
#SBATCH -t 05:00:00
$COMSUITE_BIN/comdmft.py
```

Here 'comdmft.py' is a python script which control LQSGW+DMFT simulation. Based on 'comdmft.ini' input file, 'comdmft.py' generates all necessary input files for individual programs to run and executes jobs.

Input file (comdmft.ini)

In other to perform LQSGW+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 of 'control', 'wan_hmat' and 'imp':

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

wan_hmat={
         'kgrid': [15, 15, 15],
         'froz_win_min': -10.0,
         'froz_win_max': 10.0,
      }

imp={'temperature'          : 300, # temperature (in K)
   '1':
   {
    'impurity_matrix': [ # equivalent orbital index matrix. starting from 1.
      [1,0,0,0,0],
      [0,1,0,0,0],
      [0,0,2,0,0],
      [0,0,0,1,0],
      [0,0,0,0,2]
      ],
   'thermalization_time': 3,
   'measurement_time': 20,
   'green_cutoff':  40,
   'coulomb': 'full',
   }}
```

■ In Control

These fields contain basic parameter which control LQSGW+DMFT run.

- 'initial_lattice_dir' :
  Enter the path which contains LQSGW output such as quasi particle eigenvalue and eigenfunctions. It is the lqsgw prerun folder.

- 'method' :
  Either lda+dmft or lqsgw+dmft. Currently COMSUITE has these two options. Choose 'lqsgw+dmft' for present work (LQSGW+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 ComCoulomb, ComDC, ComLowH, ComWann, and ComCTQMC. If a different MPI prefixs from this prefix is necessary for individual program, use 'mpi_prefix_coulomb', 'mpi_prefix_lowh', 'mpi_prefix_dc', 'mpi_prefix_wannier', and 'mpi_prefix_impurity'. Note that 720 is the number of total MPI processes we want to use for the present case.

You can set any number you want. Here we note that, for ComCoulomb, comdmft.py will redistribute the number of processes on two dimensional MPI grid using MPI_COMM_SPLIT.

- '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 "../lqsgw/coord.xsf"
  Shell index is either "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 LQSGW+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'].

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

- 'mpi_prefix_dc':
  MPI prefix for ComDC. 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 LQSGW 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

- For each distinct impurity problem indexed by the value in control ["impurity_problem_equivalence"],
  - 'impurity_matrix': [
    [1,0,0,0,0],
    [0,1,0,0,0], 1: $d_{xy,}$ $d_{yz,}$ d_{xz}
    [0,0,2,0,0], 2: d_{z^2} $d_{x^2-y^2}$
    [0,0,0,1,0],
    [0,0,0,0,2]
    ],
    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 $|xy\rangle, |yz\rangle, |z^2\rangle, |xz\rangle, |x^2-y^2\rangle$ 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: $|3,-0.5, -2.5\rangle, |3,-0.5, -1.5\rangle, |3,-0.5, -0.5\rangle, |3,-0.5, 0.5\rangle, |3,-0.5, 1.5\rangle, |3,-0.5, 2.5\rangle, |3,0.5, -3.5\rangle, |3,0.5, -2.5\rangle, |3,0.5,-1.5\rangle, |3,0.5, -0.5\rangle, |3,0.5, 0.5\rangle, |3,0.5, 1.5\rangle, |3,0.5, 2.5\rangle, |3,0.5, 3.5\rangle,$

  - '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.

  - 'Coulomb':,
    'full' or 'ising' are available. We construct Coulomb matrix in the following way.

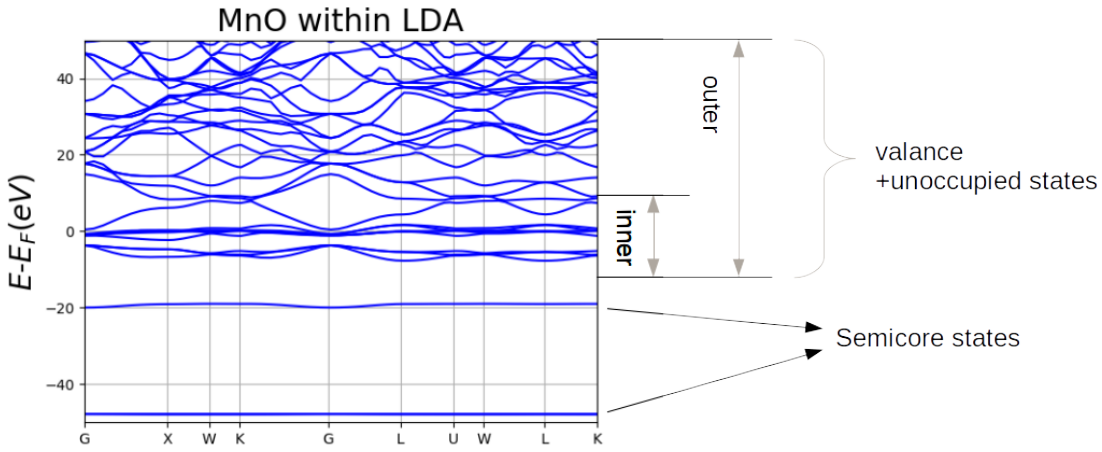
$$U_{m_1,m_2,m_3,m_4} = \sum_{k=0}^{2l,even} \frac{4\pi}{2k+1} F_l^k \sum_{q=-k}^{k} \langle Y_l^{m_1} | Y_k^q Y_l^{m_4} \rangle \langle Y_l^{m_2} Y_k^q | Y_l^{m_3} \rangle$$

If 'full', no additional approximation is considered. If 'ising', only U_{abba} or U_{abab} are non-zero.

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

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 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 LDA band of MnO as an example to illustrate these concepts).

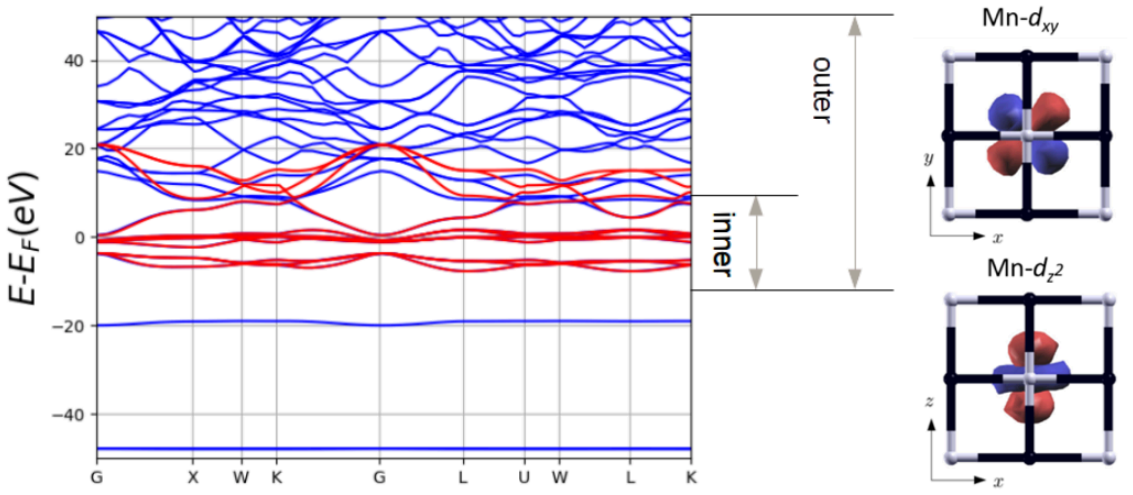


We choose initial trial orbitals $|\tau \mathbf{R} = 0\rangle_t$ using MT orbitals with desired angular momentum character. The radial functions of $|\tau \mathbf{R} = 0\rangle_t$ are chosen in such a way to maximize

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

, 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 (lqsgw_dmft in this example). The list of files is

cmd.log
convergence.log : convergence log file
delta.dat : hybridization function
sig.dat : impurity self-energy
sig_dc.dat : double-counting self-energy
sig_dc_hf.dat : the high-frequency limit of double-counting self-energy
u_slater.dat : Slater-Condon parameterization of bosonic Weiss-field
v_slater.dat : Slater-Condon parameterization of bare coulomb interaction (V_{loc})
w_slater.dat : Slater-Condon parameterization of screened coulomb interaction (W_{loc})

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

■ **convergence.log**

step	i_imp	causality	static_f0	w_sp_min	w_sp_max	mu	std_sig	n_imp	histo_1	histo_2
wannier				0.28543421	1.7773025					
coulomb_1			6.97943595685							
dc_1		good								
delta	1	good				0.019644259897				
impurity_1	1	good					3.37592293015	8.13892	104.203172231	113.442465644
delta	2	good				0.227452938024				
impurity_1	2	good					1.76039597987	8.12925	95.5346308627	93.4049387127
delta	3	good				0.360584306439				
impurity_1	3	good					0.873900782988	8.12575	93.8121360239	90.1533755051
delta	4	good				0.438313820721				

- keeping track of convergence of some quantities at each iteration
- i_imp: The number of solving impurity problem through ComLowH + ComCTQMC + ComDC
- causality: causality of hybridization function / self-energy
- w_sp_min: minimum spreading of the Wannier functions
- w_sp_max: maximum spreading of the Wannier functions
- mu: LQSGW+DMFT chemical potential w.r.t. LQSGW 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 in the impurity orbitals
- histo_1: the first moment of the perturbation order histogram
- histo_2: the second moment of the perturbation order histogram
- ctqmc_sign: CTQMC sign

■ **u_slater.dat** and **w_slater.dat**

Data format in u_slater.dat (also in w_slater.dat):

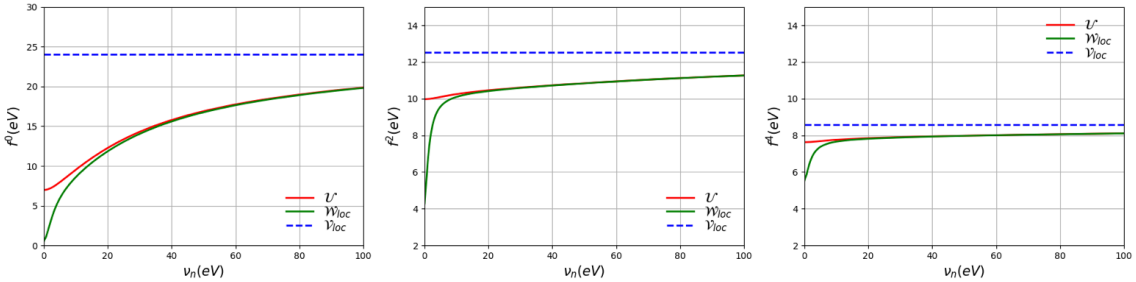
# nu(eV)	1:f0(eV)	1:f2(eV)	1:f4(eV)
0.000000000000	6.979435956849	9.972684910682	7.626659183780
0.162432849384	6.991563460566	9.974254718886	7.627439680636
0.324865698768	6.995381415918	9.974733912820	7.627681553000
0.487298548153	6.996249914058	9.974825995447	7.627732241701
0.649731397537	7.003719509740	9.975764279332	7.628170672937
0.812164246921	7.017060208047	9.977474094269	7.628991575522
0.974597096305	7.033267110201	9.979588473166	7.630054710525
1.137029945690	7.051862355810	9.982051407717	7.631340778350
1.299462795074	7.072740247508	9.984834431675	7.632809228323
1.461895644458	7.095434905356	9.987858550026	7.634389875835

The first column is the bosonic matsubara frequencies and the second, third, and fourth column is Slater's integral F0, F2, and F4, respectively.

Data format of Bare Coulomb interaction in v_slater.dat:

# 1:f0(eV)	1:f2(eV)	1:f4(eV)
24.073782938928	12.539899507718	8.587692445309

Plots of dynamical U using u(v,w)_slater.dat files :



Slater's integrals of partially-screened Coulomb interactions associated with Ni-*d* orbitals are marked by red lines. For comparison, Slater's integrals of bare Coulomb interactions and fully-screened Coulomb interaction are shown by blue dashed lines and green full lines respectively.

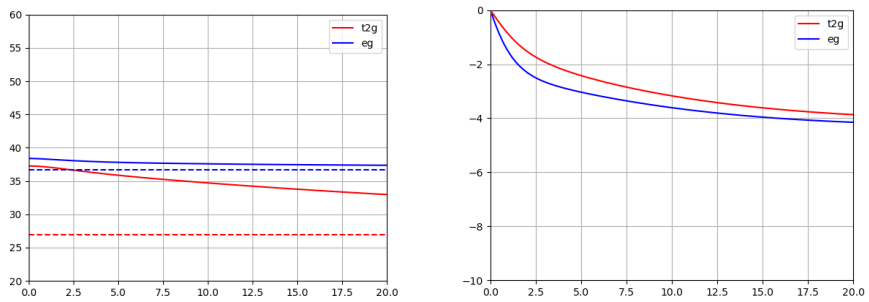
■ sig_dc.dat

'sig_dc.dat' contains impurity self-energies within local GW approximation. The first column lists matsubara frequencies and the next columns are real and imaginary parts of self-energies of Ni-d orbitals.

■ sig_dc_hf.dat

'sig_dc_hf.dat' contains real and imaginary part of the Hartree-Fock contribution to the impurity self-energy within local GW approximation

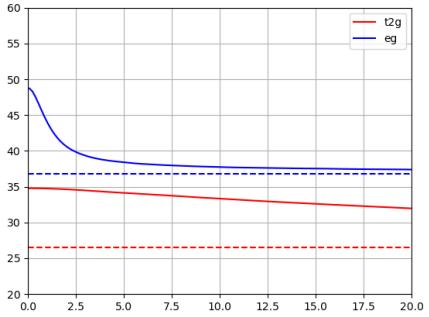
Plots of the real and imaginary parts of local-GW impurity self-energy are as follows

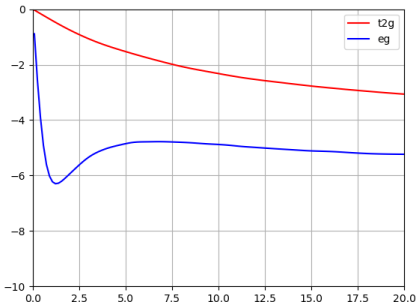


■ **sig.dat**

'sig.dat' contains impurity self-energies obtained from ComCTQMC. The first column lists matsubara frequencies and the next columns are real and imaginary parts of self-energies of Ni-d orbitals.

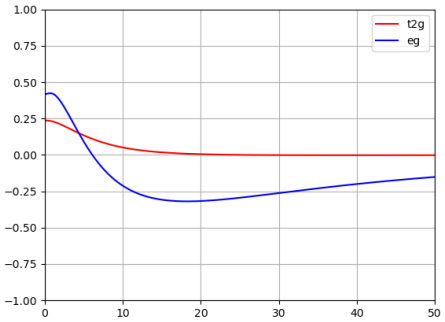
Plots of the real and imaginary parts of impurity self-energy using the sig.dat file:

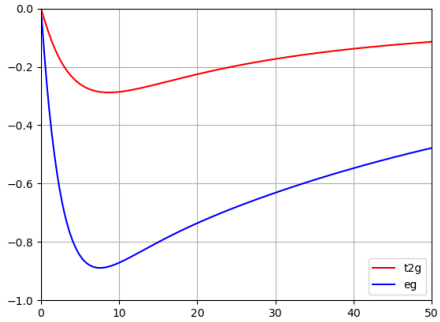




■ **delta.dat**

'delta.dat' is in the same format with 'sig.dat'. Plots of the real and imaginary parts of the hybridization function using the delta.dat file:





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:

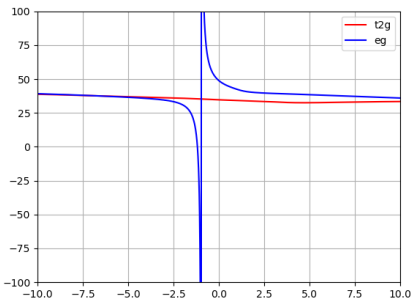
```
$ mkdir maxent
$ cd maxent
```

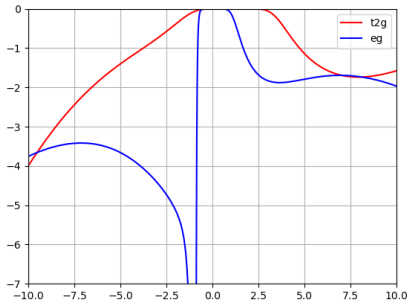
By executing 'maxent_wrapper.py', we can obtain the self-energy on real axis by automatically calling maxent_run.py:

```
$ $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

Plots of the real and imaginary pars of impurity self-energy on the real frequency axis:





LQSGW+DMFT density of states

To obtain the DOS, we must post-process the data. First, create a directory for the DOS calculation in the "lqsgw_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
```

Doing so, you will obtain tdos.dat and pdos.dat.

The format of tdos.dat file is:

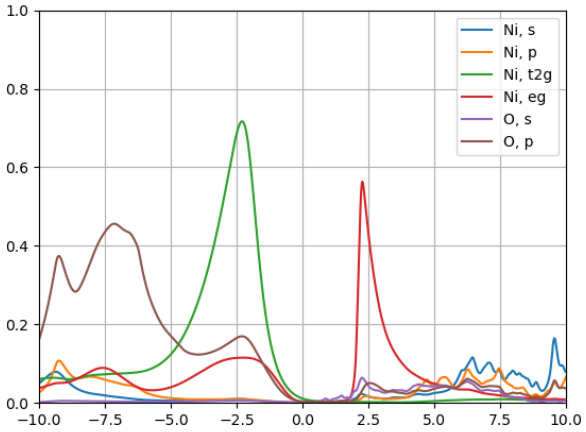
```
#     omega (eV)          DOS (1/eV)
   -70.000000000000     0.000201342724
   -65.733023713200     0.000228882167
   -61.955278682000     0.000258557684
   -58.587128546700     0.000290978926
   -55.565331497900     0.000328380615
   -52.839027896300     0.000379605164
   -50.366851286200     0.000471555161
   -48.114812211200     0.000668833899
   -46.054723948000     0.001093921673
   -44.163014879300     0.001941283676
```

The format of pdos.dat file is :

```
#     omega (eV)            (1,1,-1)            (1,1,0)             (1,1,1)             (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)
   -70.000000000000       0.000007506048      0.000007506046      0.000007506050      0.000006114846      0.000005018605      0.000005018609      0.000005018602      0.000010927273
0.000010927264      0.000007669090      0.000010927268      0.000007669090
   -65.733023713200       0.000008615621      0.000008615619      0.000008615624      0.000006929192      0.000005621753      0.000005621757      0.000005621748      0.000013659821
0.000013659808      0.000008916099      0.000013659814      0.000008916099
   -61.955278682000       0.000009821212      0.000009821208      0.000009821216      0.000007800185      0.000006257663      0.000006257668      0.000006257657      0.000020560307
0.000020560286      0.000010291916      0.000020560296      0.000010291916
   -58.587128546700       0.000011134876      0.000011134871      0.000011134881      0.000008742243      0.000006941787      0.000006941794      0.000006941780      0.000046979894
0.000046979842      0.000011943325      0.000046979866      0.000011943325
   -55.565331497900       0.000012589013      0.000012589006      0.000012589020      0.000009803351      0.000007727704      0.000007727715      0.000007727692      0.000139012572
0.000139012401      0.000014279941      0.000139012481      0.000014279941
```

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

Plot of tdos.dat and pdos.dat:



LGSGW+DMFT quasiparticle bands

To obtain the quasiparticle bands within LQSGW+DMFT, we must post-process the data by executing ComLowH again. First create a directory for the spectral function in “lqsgw_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’

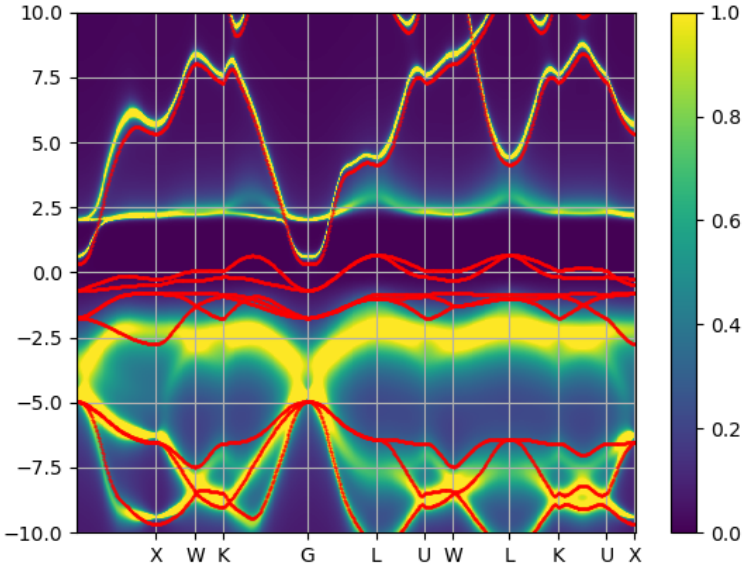
```
control ={
      ‘method’: ‘spectral’,
     ‘mpi_prefix’: ‘srun -n 32’,
    }

postprocessing={
     ‘comsuite_dir’: ‘../’,
     ‘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 wannier_band_qp_interpolated.dat
Here, red lines show lqsgw bandsturctures


10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
X
W
K
G
L
U
W
L
K
U
X
1.0
0.8
0.6
0.4
0.2
0.0