JDFT + RPA calculations in practice with JDFTx + BGW

The BEAST collaboration

1st Annual BEAST Workshop, 2022

August 16, 2022

U.S. DEPARTMENT OF ENERGY

Office of Science

Award # DE-SC0022247

Moving beyond DFT: calculating RPA energies

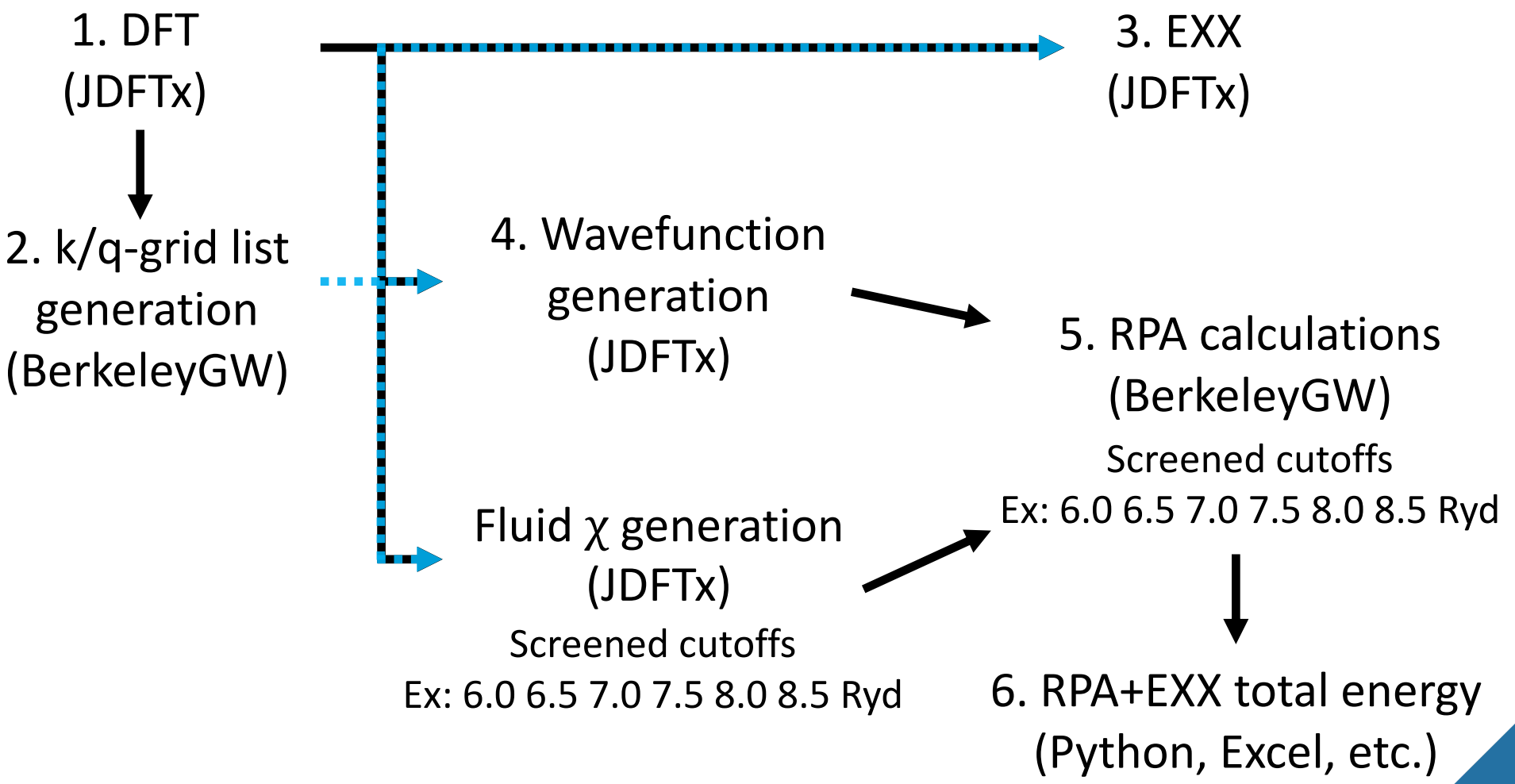
- Replace DFT exchange-correlation energy with exact exchange energy and RPA correlation energy

$$E_{RPA} = E_{DFT} - E_{XC} + E_{XX} + E_{corr,RPA}$$

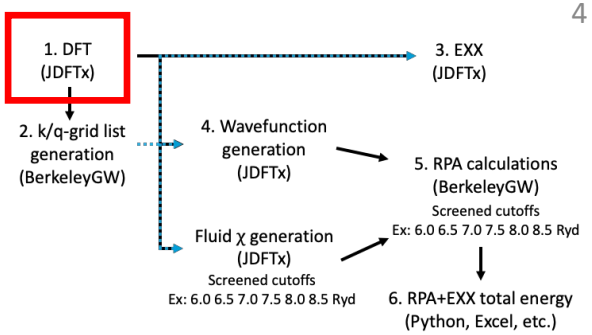
- How do we calculate each energy component?
  - JDFTx → E_{DFT}, E_{XC}, E_{XX}
  - BerkeleyGW → $E_{corr,RPA}$

- Steep scaling for RPA correlation energy
  - BerkeleyGW can scale to thousands of atoms

Practical workflow for RPA total energies



1. Standard DFT calculation with JDFTx

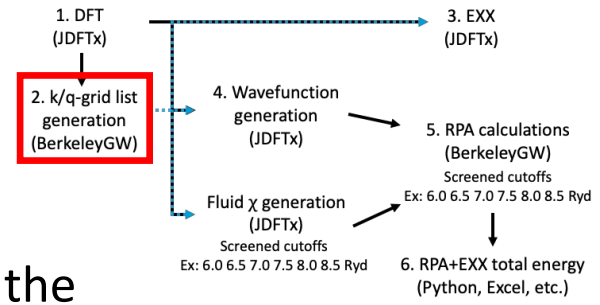


- See tutorials from yesterday for inputs
  - Standard DFT calculation

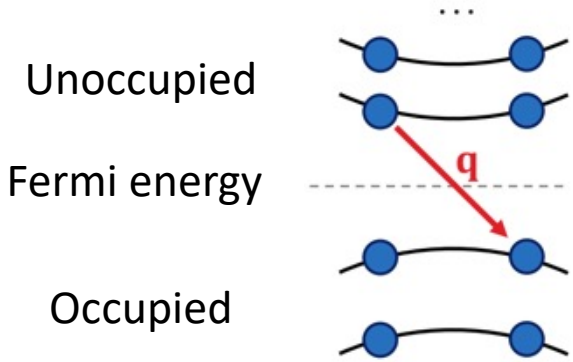
- Pay attention to convergence
  - RPA and EXX calculations may converge slower with respect to planewave cutoff and k-grid density than DFT calculations

- Keep in mind the cost scales quickly with number of electrons and size of supercell!

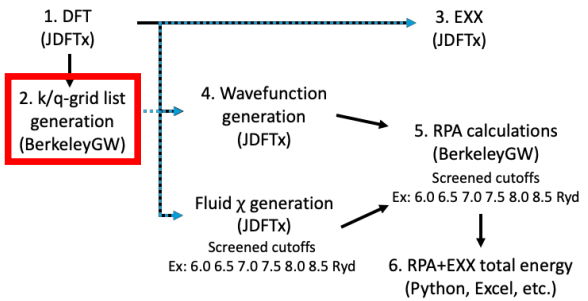
2. k/q-grid generation using BGW utility



- BGW needs an explicit list of points
  - The kgrid.x utility evaluates the symmetries of the supercell to provide this list
  - Convergence is usually best if DFT, EXX, and RPA use same grid
  - See BGW docs for detailed syntax
  - Polarizability and dielectric matrices calculated on q-point grid
    - Regular, Γ-centered grid obtained from all transfer vectors $q = k' - k$



2. k/q-grid generation using BGW utility



kgrid.in

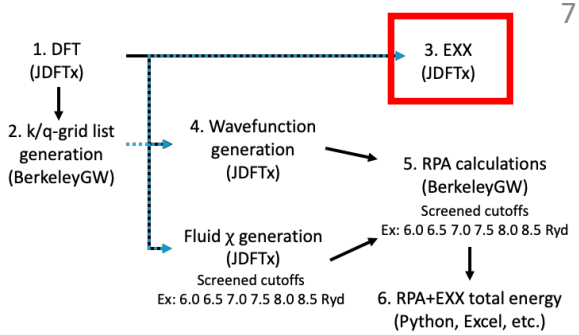
```
4 4 1                          ← Desired folding
0 0 0
0 0 0
5.2219 0.0 0.3199              ← Lattice vectors
2.6011 4.5280 0.3197
0.0 0.0 28.3459
3                              ← Number of atoms
1 1.5646 0.9056 10.0183        ← Ion positions
1 3.8921 2.2528 14.8234
1 6.2166 3.5983 19.6305
24 24 120                      ← FFT grid
.false.
```

kgrid.out

```
K_POINTS crystal
  16
 0.00  0.00  0.00   1.0
 0.00  0.25  0.00   1.0
 0.00  0.50  0.00   1.0
...
```

Need to renormalize weights to 1.0 for JDFTx

3. Exact exchange calculation with JDFTx



- JDFTx interacts with BerkeleyGW via an input file tag
  - bgw-params

bgw-params

Syntax:

```
bgw-params <key1> <value1> <key2> <value2> ...
```

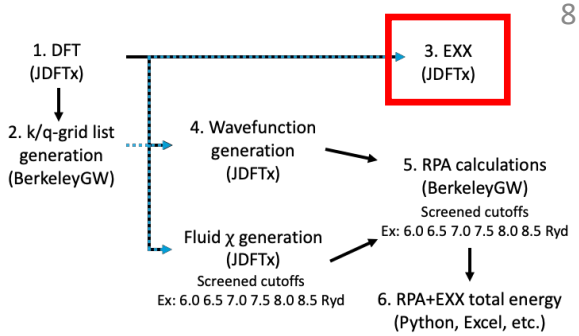
Description:

Control BGW output. Possible keys and value types are:

- blockSize : Block size for ScaLAPACK diagonalization (default: 32)
- clusterSize : Maximum eigenvalue cluster size to allocate extra ScaLAPACK workspace for (default: 10)
- Ecut_rALDA : KE cutoff in hartrees for rALDA polarizability output (default: 0; set non-zero to enable)
- EcutChiFluid : KE cutoff in hartrees for fluid polarizability output (default: 0; set non-zero to enable)
- elecOnly : Whether fluid polarizability output should only include electronic response (default: true)
- freqBroaden_eV: Broadening (imaginary part) of real frequency grid in eV (default: 0.1)
- freqNimag : Number of imaginary frequencies (default: 25)
- freqPlasma : Plasma frequency in Hartrees used in GW imaginary frequency grid (default: 1.), set to zero for RPA frequency grid
- freqReMax_eV : Maximum real frequency in eV (default: 30.)
- freqReStep_eV : Real frequency grid spacing in eV (default: 1.)
- nBandsDense : If non-zero, use a dense ScaLAPACK solver to calculate more bands
- q0 : Zero wavevector replacement to be used for polarizability output (default: (0,0,0))
- rpaExx : Whether to compute RPA-consistent exact-exchange energy (default: no)
- saveVxx : Whether to write exact-exchange matrix elements (default: no)

Any number of these key-value pairs may be specified in any order.

3. Exact exchange calculation with JDFTx

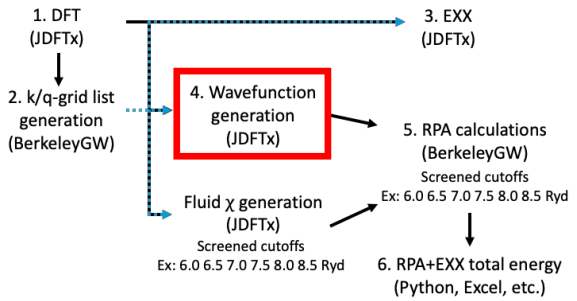


New tags appended to ”in” file

```
fix-electron-density jdft.$VAR
bgw-params rpaExx yes
include kgrid.jdft
symmetries none
```

- Perform fixed electron density calculation starting from converged DFT calculation
- The bgw-params tag tells JDFTx to calculate EXX energy
- kgrid.jdft contains the k/q-point information calculated in Step 2

4. Empty state generation with JDFTx



- JDFTx interacts with BerkeleyGW via an input file tag
  - bgw-params

bgw-params

Syntax:

```
bgw-params <key1> <value1> <key2> <value2> ...
```

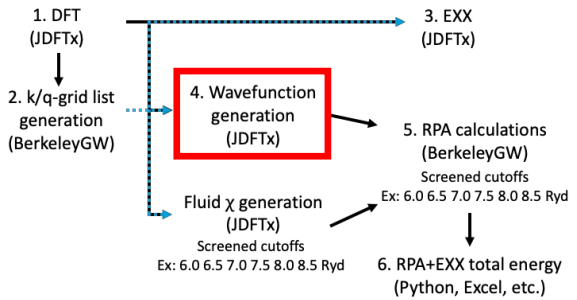
Description:

Control BGW output. Possible keys and value types are:

- blockSize : Block size for ScaLAPACK diagonalization (default: 32)
- clusterSize : Maximum eigenvalue cluster size to allocate extra ScaLAPACK workspace for (default: 10)
- Ecut_rALDA : KE cutoff in hartrees for rALDA polarizability output (default: 0; set non-zero to enable)
- EcutChiFluid : KE cutoff in hartrees for fluid polarizability output (default: 0; set non-zero to enable)
- elecOnly : Whether fluid polarizability output should only include electronic response (default: true)
- freqBroaden_eV: Broadening (imaginary part) of real frequency grid in eV (default: 0.1)
- freqNimag : Number of imaginary frequencies (default: 25)
- freqPlasma : Plasma frequency in Hartrees used in GW imaginary frequency grid (default: 1.), set to zero for RPA frequency grid
- freqReMax_eV : Maximum real frequency in eV (default: 30.)
- freqReStep_eV : Real frequency grid spacing in eV (default: 1.)
- nBandsDense : If non-zero, use a dense ScaLAPACK solver to calculate more bands
- q0 : Zero wavevector replacement to be used for polarizability output (default: (0,0,0))
- rpaExx : Whether to compute RPA-consistent exact-exchange energy (default: no)
- saveVxx : Whether to write exact-exchange matrix elements (default: no)

Any number of these key-value pairs may be specified in any order.

4. Empty state generation with JDFTx

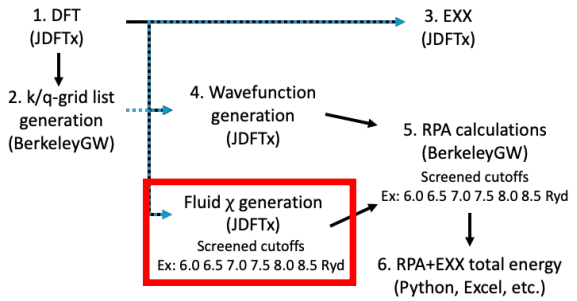


New tags appended to ”in” file

fix-electron-density jdft.$VAR
bgw-params nBandsDense 500 blockSize 96
include kgrid.jdft
symmetries none

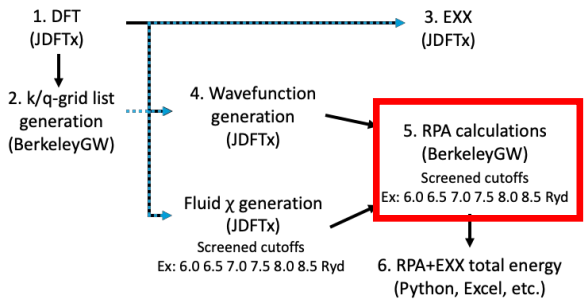
- Perform fixed electron density calculation starting from converged DFT calculation
- The bgw-params tag tells JDFTx to use SCALPACK to do a dense diagonalization and generate 500 bands
- kgrid.jdft contains the k/q-point information calculated in Step 2
- These wavefunctions can be huge! ~1 – 500 GB

Fluid χ generation with JDFTx

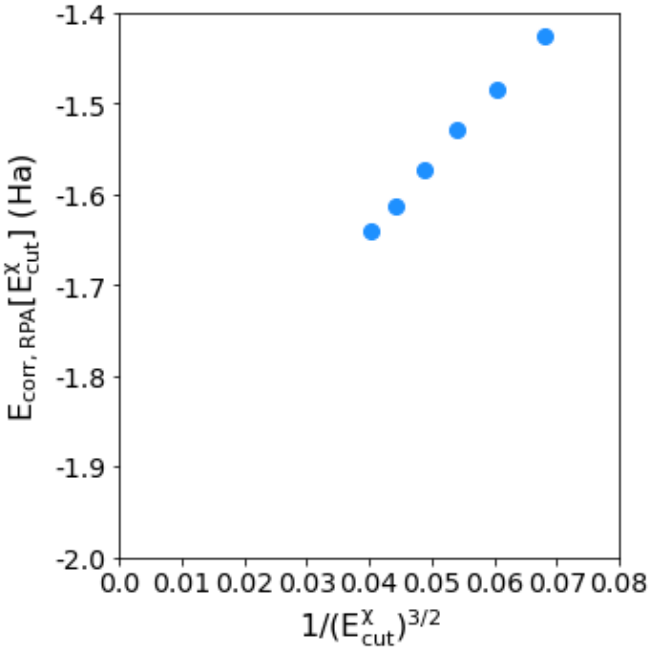


- Will be covered at next year's workshop
- Adjusts the screening environment of the surface
- Uses the bgw-params JDFTx input tag to provide additional details about the fluid polarizability
- Read by BGW and incorporated into the RPA calculation
- Must generate a separate fluid χ file for each screened cutoff used in the RPA calculations

5. RPA correlation energy with BGW



- RPA correlation energies converge slowly with respect to screened cutoff
  - Extrapolate to infinite screened cutoff
  - Each calculation is individually expensive
- The screened cutoff/number of bands change together
  - Screened cutoff defines max cutoff for G-vectors used for χ



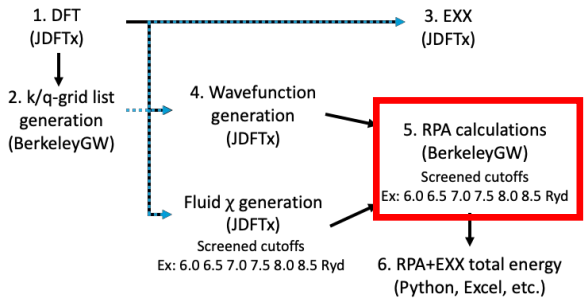
epsilon.inp key tags

cell_slab_truncation
epsilon_cutoff 6.0
number_bands 166
nbasis_subspace 166
number_core_excluded 12

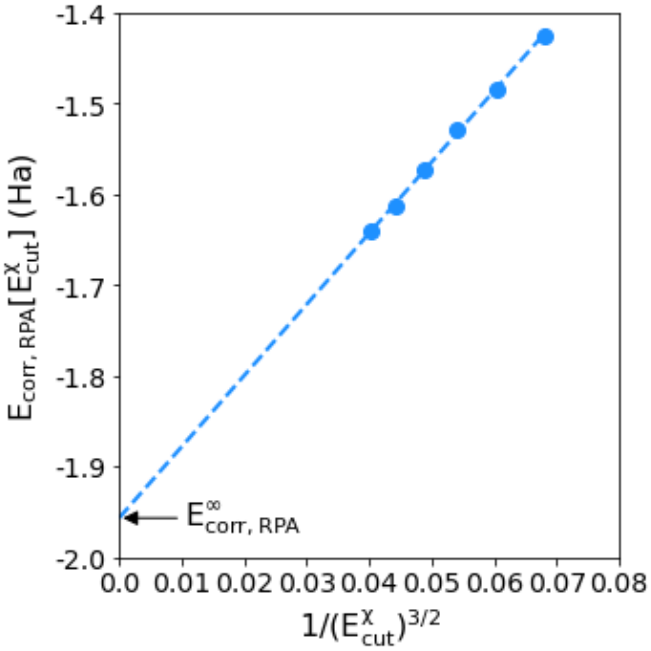
Set same as DFT E_F and broadening

fermi_level 7.5542
fermi_level_absolute
occ_broadening 0.2721

5. Extrapolation of RPA energies



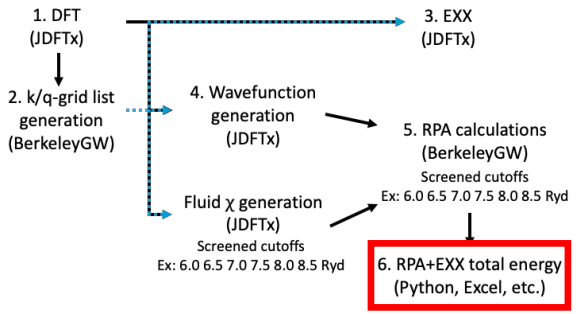
- Can be done with any post-processing analysis tool
  - Python, Excel, MATLAB, etc.
- Extrapolate to infinite screened cutoff (y-axis)
  - 4-8 points for extrapolation is common, depending on size
- Assumes electrons behave as free-electron gas



Arbitrary fitted coefficient

$$E_{corr,RPA}\left[E^{\chi}_{cut}\right] = E^{\infty}_{corr,RPA} + \frac{A}{\left(E^{\chi}_{cut}\right)^{\frac{3}{2}}}$$

6. RPA total energy calculation



$$E_{RPA} = E_{DFT} - E_{XC} + E_{XX} + E_{corr,RPA}$$

- Find each energy component from the ends of the JDFTx/BGW out files
  - JDFTx DFT calculation → E_{DFT}, E_{XC}
  - JDFTx EXX calculation → E_{XX}
  - BerkeleyGW → $E_{corr,RPA}$

- Repeat this entire process for each system needed to calculate an adsorption energy

Let's begin the tutorial!

- Please log-in to Jupyter Hub for Cori like yesterday
- Please copy a new version of the RPA tutorial
- These jobs will not be run interactively like yesterday
  - Use the "sq" command in your terminal to see the status of your jobs after running any "XX_run_job.sh" command