Getting your feet wet

OCEAN Tutorial, Session 1

Getting started with OCEAN

- Running on BNL cluster

- Running
- Stages of OCEAN
- First results
- Converging the DFT stage
- Input file(s)
- Converging OCEAN

What is OCEAN?

- OCEAN is a perl script: ocean.pl
  - Parses the input file
  - Bunch of file management throughout
  - Runs some DFT calculations
  - Calculates the screened core-hole potential
  - Runs a number of Bethe-Salpeter equation (BSE) calculations
- OCEAN is a compiled executable: ocean.x
  - Calculates the BSE
  - Also independent particle, RPA*, adiabatic TDLDA*

Simulating materials with OCEAN

- Everything is periodic!
  - Crystalline systems use unit cell
  - Liquids, surfaces, and molecules also possible
    - Cell needs to be “big enough”
    - Capture the disorder
    - Avoid spurious interactions
- Structure is the only **free** parameter
  - Choose system and x-ray
  - Other parameters are for convergence
    - Best BSE answer, not necessarily best agreement

Brief aside on using UNIX variants

- Not always intuitive
- Ask if you aren’t sure!

- To stop the computer from doing things
  - Q quits many programs
  - Control+C kills things (^C)

Detour for the workshop

- We'll be running on BNL local cluster
- Open up a terminal & connect
  - Replace "jtv3" with your username

1. > ssh –Y –A jtv3@ssh01.sdcc.bnl.gov
2. > ssh –Y –A jtv3@icsubmit01

Detour for the workshop

- We'll be running on BNL local cluster
- Open up a terminal & connect
  - Replace "jtv3" with your username
  1. > ssh –Y –A jtv3@ssh01.sdcc.bnl.gov
  2. > ssh –Y –A jtv3@icsubmit01
- Make sure plotting works
  1. > gnuplot
  2. > plot x
- Windows users: checkout mobaxterm, Mac users might need to install X11

Running OCEAN

> /path/to/ocean/ocean.pl inputFile

Running OCEAN: important etymology

Running OCEAN: important etymology

> pwd

/current/path

Running OCEAN: important etymology

> pwd

/current/path

> /path/to/ocean/ocean.pl inputFile

Running OCEAN: important etymology

```
> pwd
/current/path
> /path/to/ocean/ocean.pl inputFile
```

Run Directory

- Where you ran OCEAN from
- All output is relative to here

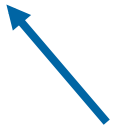
Detour for this workshop

- OCEAN is built and installed
  - /hpcgpfs01/work/workshop/ocean_tutorial/bin
- We want to run on the scratch file system!
  1. > cd /hpcgpfs01/scratch

change directory

Detour for this workshop

- OCEAN is built and installed
  - /hpcgpfs01/work/workshop/ocean_tutorial/bin
- We want to run on the scratch file system!
  1. > cd /hpcgpfs01/scratch
  2. > mkdir [your username]



Make a new directory

Detour for this workshop

- OCEAN is built and installed
  - /hpcgpfs01/work/workshop/ocean_tutorial/bin
- We want to run on the scratch file system!
  1. > cd /hpcgpfs01/scratch
  2. > mkdir [your username]
  3. > cd [your username]

Detour for this workshop

- OCEAN is built and installed
  - /hpcgpfs01/work/workshop/ocean_tutorial/bin
- We want to run on the scratch file system!
  1. > cd /hpcgpfs01/scratch
  2. > mkdir [your username]
  3. > cd [your username]
  4. > pwd

```
jtv3@icsubmit01:/hpcgpfs01/scratch/jtv3> pwd
/hpcgpfs01/scratch/jtv3
jtv3@icsubmit01:/hpcgpfs01/scratch/jtv3>
```

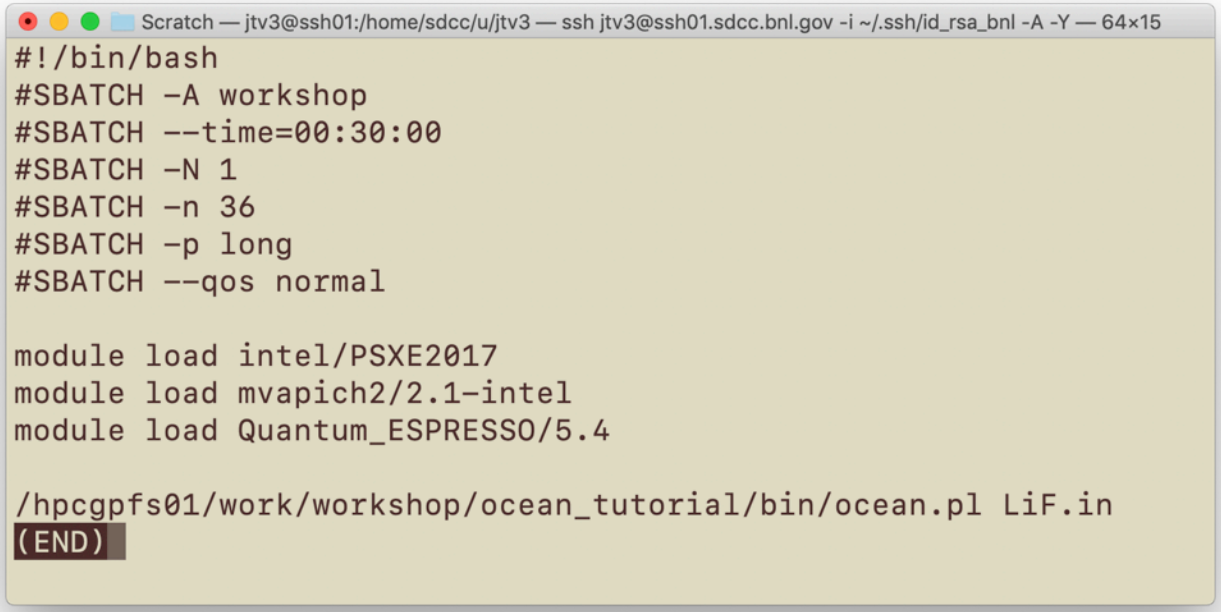
Detour for this workshop

- We want to run on the scratch file system!
  1. > cd /hpcgpfs01/scratch
  2. > mkdir [your username]
  3. > cd [your username]
  4. > pwd
- Copy the tutorial files over
  1. > cp –r /hpcgpfs01/work/workshop/ocean_tutorial/session1 .
  2. > cd session1
  3. > pwd

```
jtv3@icsubmit01:/hpcgpfs01/scratch/jtv3/session1> pwd
/hpcgpfs01/scratch/jtv3/session1
jtv3@icsubmit01:/hpcgpfs01/scratch/jtv3/session1>
```

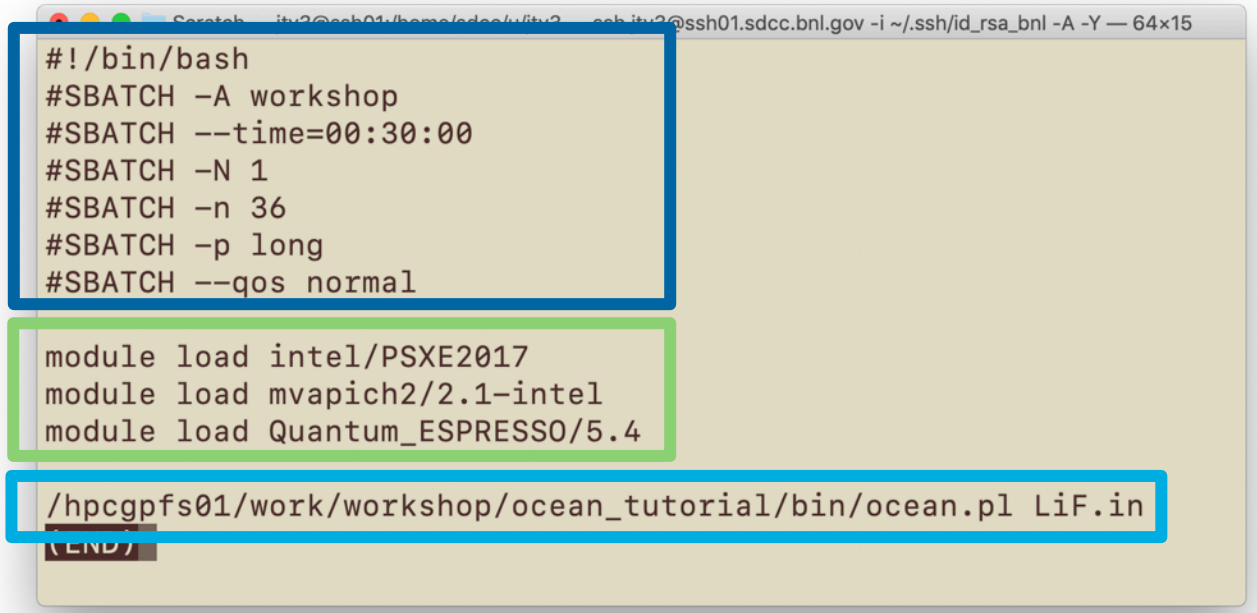
Detour for this workshop

- Look at the file runit
  - ls (lower case L) shows the files in the current directory
  - > less runit
  - Getting around in less
    - Up/down arrow
    - G jumps to bottom
    - F locks to bottom and refreshes
      - Control+c (^c) to stop this
    - q quits



Detour for this workshop

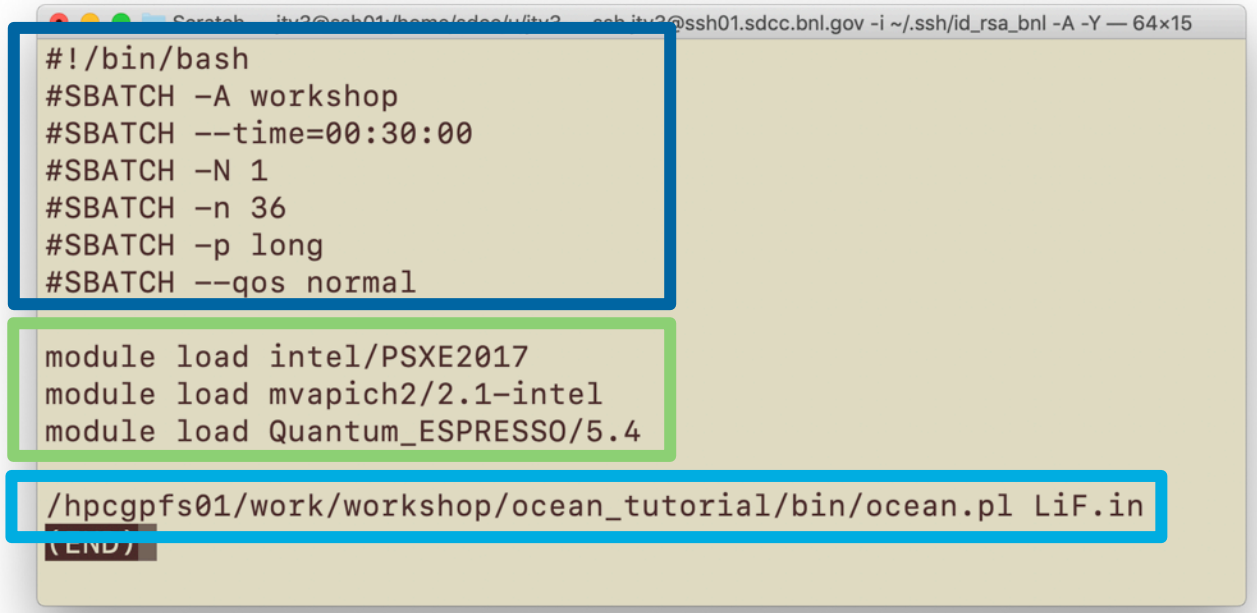
- Look at the file runit
  - ls (lower case L) shows the files in the current directory
  - > less runit
- Cluster-dependent
  - Queue information
  - Load up libraries
- Run OCEAN
  - On your own computer you would just type this line



Detour for this workshop

- To submit our job to the queue
  - > sbatch runit
  - Submitted batch job 369632

- OCEAN writes stuff to the terminal while it runs
  - With the queue this goes into a file “slurm-369632.out”



Detour for this workshop

- OCEAN writes stuff to the terminal while it runs
  - With the queue this goes into a file
  - “slurm-369632.out”
  - > less slurm-369632.out

```
Welcome to OCEAN
#################################################################################
#                 OOO   CC  EEEE  AA  N  N                    #
#                O   O C  C E    A  A NN N                    #
#      **        O   O C    E    A  A NN N         **         #
#     **         O   O C    EE   AAAA N NN          **        #
#    ***    **   O   O C    E    A  A N NN    **    ***       #
#   *****  ***   O   O C  C E    A  A N  N    ***  *****      #
# **************  OOO   CC  EEEE A  A N  N  ************** #
#################################################################################
Version 2.9.0
  commit hash: fb08cb832552a6cb41326f8ecb0d155daefc0033
Run differs
Setup complete, parsing ...
Config read in
```

Detour for this workshop

- OCEAN writes stuff to the terminal while it runs
  - With the queue this goes into a file
  - “slurm-369632.out”
  - > less slurm-369632.out
  - When finished you’ll get “Ocean is done”

```
  qmag =    1.87857835000000      inverse bohrs
1s      corezetaz009
time srun -n 16  /direct/sdcc+u/jtv3/OCEAN_BIN/2.9.0/ocean.x > cm.log
real    0m51.840s
user    0m0.021s
sys     0m0.044s
###############################################################################
Ocean is done
###############################################################################
Waiting for data... (interrupt to abort)
```

First Run: LiF

While that runs …

Stages of OCEAN

1) Parsing (Common)
2) Atomic (OPF)
3) DFT
4) PREP
5) SCREEN
6) BSE (CNBSE/NBSE)

Parsing

- Input file read
- Lots of little files created in Common/

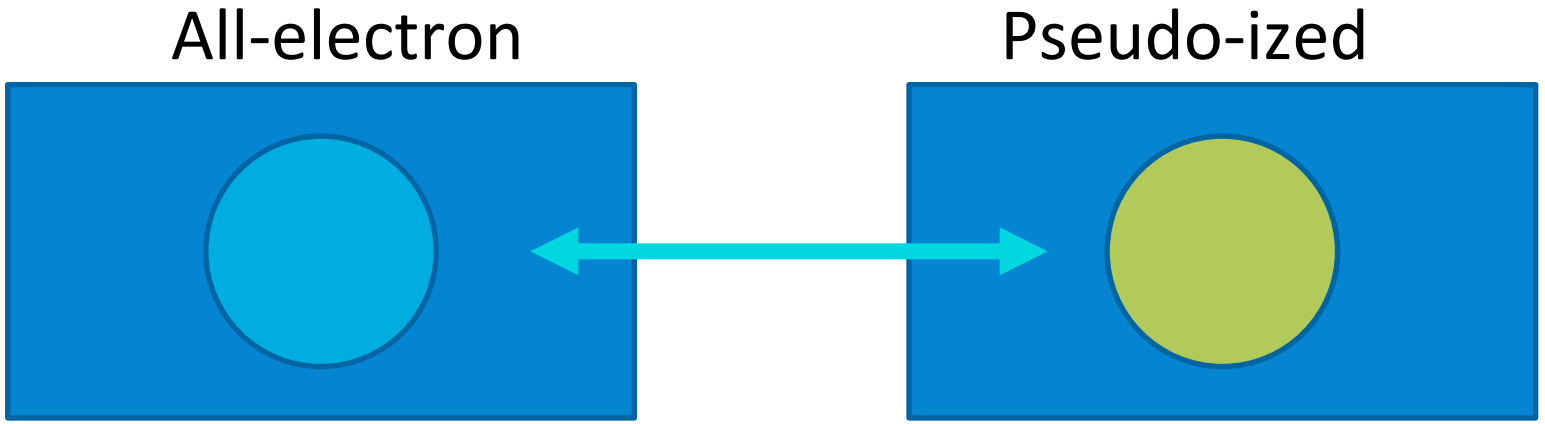
Atomic

- OCEAN uses pseudopotentials
  - Replace nuclei's Coulomb potential Z/r
  - No core electrons
- X-ray spectroscopy involves the core electrons

- Obviously there is a problem here

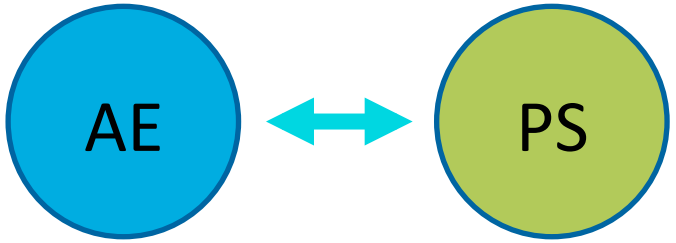
Atomic

- Carry out two atomic calculations
  - All-electron
  - Pseudo-ized
- Outside of cutoff r_c orbitals are the same



Atomic

- Carry out two atomic calculations
  - All-electron
  - Pseudo-ized
- Outside of cutoff r_c orbitals are the same
- Optimal Projector Functions (OPFs)
  - Basis functions to span AE/PS
  - Limited radius (~2 Bohr)
  - Limited energy range (100 eV)
- PAW-like method
  - P. E. Blöchl PRB **50**, 17953 (1994)
  - E. L. Shirley J. Elect. Spect. Relat. Phenom. **110-111**, 305 (2000)



Stages of OCEAN

1) Parsing (Common)
2) Atomic (OPF)
3) DFT
4) PREP
5) SCREEN
6) BSE (CNBSE/NBSE)

Density Functional Theory (DFT)

- OCEAN seeks to be DFT code agnostic
  - QuantumESPRESSO (4.3-6.1, 6.2-6.3*)
    - www.quantum-espresso.org
    - 6.2-6.3: add “-D__OLDXML” to make.inc
  - ABINIT 5.x?-8.10
    - www.abinit.org
- Today we’re using QE

Density Functional Theory (DFT)

- What do we need DFT for?

Density Functional Theory (DFT)

Three steps:

1) Self-Consistent Field (SCF)
   - DFT means H is H[n]
   - Converge density iteratively
   - Only occupied states
2) Non-SCF for BSE orbitals
   - Dense k-point mesh
3) Non-SCF for screening
   - Few k-points, but many bands

Density Functional Theory (DFT)

Log files:

1) Self-Consistent Field (SCF)
   - DFT/scf.out
2) Non-SCF for BSE orbitals
   - DFT/XXXYYYZZZ/nscf.out
   - Directory determined by k-point grid (e.g., 008008008)
3) Non-SCF for screening
   - DFT/SCREEN/nscf.out

PREP

- File management
  - Hides differences between ABINIT & QE
  - New DFT interfaces go here

Stages of OCEAN

1) Parsing (Common)
2) Atomic (OPF)
3) DFT
4) PREP
5) SCREEN
6) BSE (CNBSE/NBSE)

SCREENing

- Electron-hole attraction is screened in the BSE
- Use RPA screening
  - $\chi_0 = iGG$
- Each atomic site will have a unique response χ_0

- We will come back to this

BSE

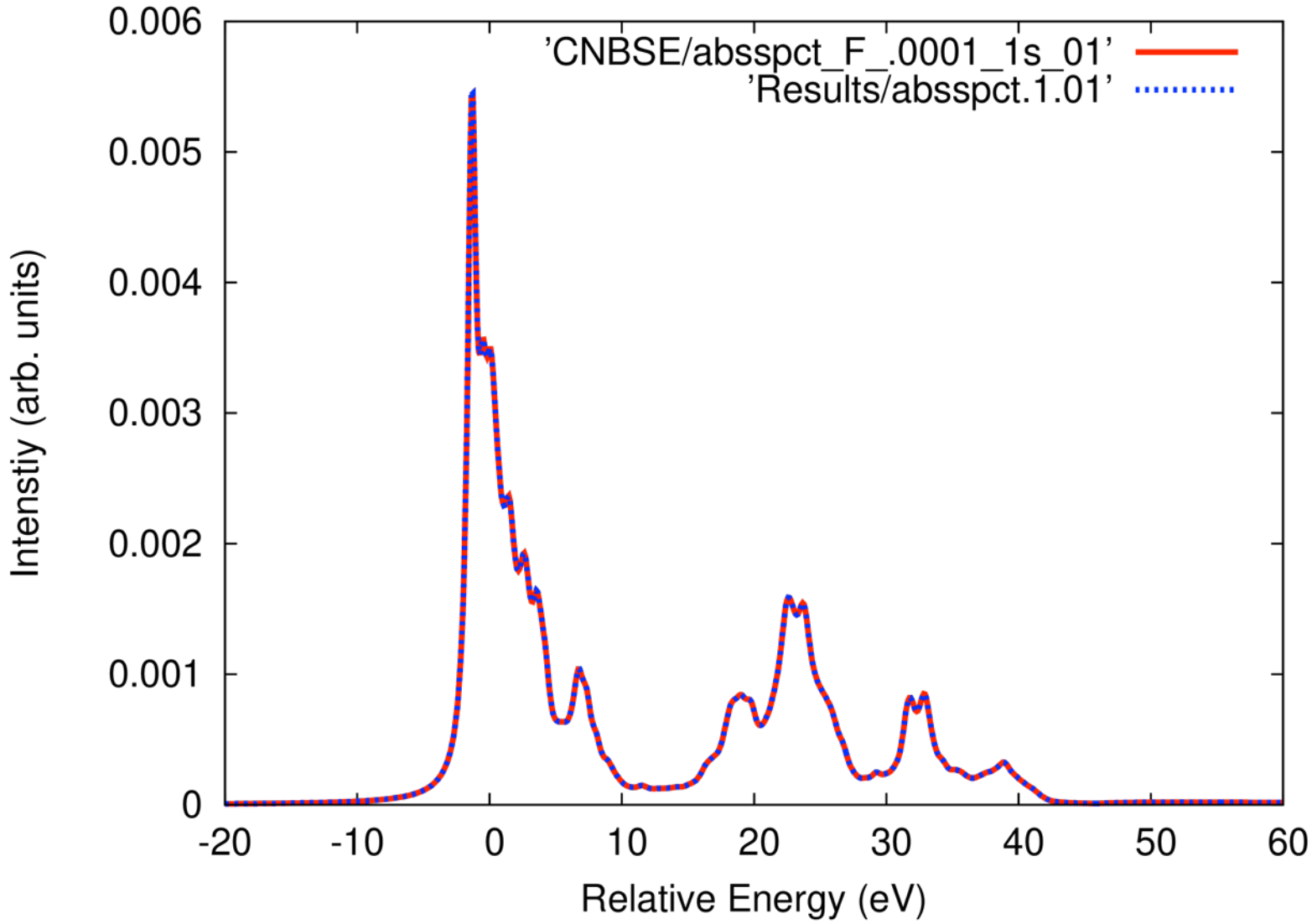
- CNBSE (NBSE for valence)
- Collect info from other sections
- Calculate matrix elements
  - <a|d|v>
- Run BSE code ocean.x

LiF

- Should be finished by now
- The output should end with “OCEAN is done”
- Final spectra are in CNBSE
  - absspct_F_.0001_1s_01

- Open gnuplot
  - plot “CNBSE/absspct_F_.0001_1s_01” w lp
  - replot “Results/absspct.1.01” w l

LiF


'CNBSE/absspct_F_.0001_1s_01'
'Results/absspct.1.01'
Intenstiy (arb. units)
0.006
0.005
0.004
0.003
0.002
0.001
0
-20
-10
0
10
20
30
40
50
60
Relative Energy (eV)

Outline

- Running
- Stages of OCEAN
- First results
- Converging the DFT stage
- Input file(s)
- Converging OCEAN

DFT

- Hamiltonian depends on electron density
- Electron density from solving Hamiltonian

- SCF: Self-consistent field
  - Repeat until density doesn’t change
  - DFT code handles this

SCF convergence

- Need correct structure, spin ordering, etc.

1. ecut: Energy cut-off
   - Discretize calculation using plane waves $e^{iG\cdot r}$
   - Highest energy plane wave set by ecut = ½ G^2
   - Larger = more accurate, but more expensive
2. ngkpt: K-point sampling for the SCF run
   - Integrals over Brillouin zone replaced by sums
   - Larger = more accurate, but more expensive

SCF convergence

- Check the total energy of the SCF calculation
  - If ecut → ∞ and ngkpt → ∞ does the result change?
    - 0.01 eV / atom
  - > less DFT/scf.out and scroll down (hit ‘G’ then use ‘b’ to jump up)

```
!    total energy              =     -49.09072812 Ry
     Harris-Foulkes estimate   =     -49.09072812 Ry
     estimated scf accuracy    <          4.9E-12 Ry
```

  - > grep ‘!’ DFT/scf.out

SCF convergence

- ecut: Set by pseudopotential
  - Most restrictive pseudo in your structure
  - Not by element!
    - Changing pseudo for same element can change cutoff
- ngkpt: unit cell dependent
  - Higher for smaller band gaps
  - Higher for smaller unit cells

SCF convergence

- Start convergence script then examine it
  - > sbatch runConvergence
  - > less runConvergence

- First loop is for ngkpt
  - $2^3 – 8^3$

```
mkdir -p CONV
cd CONV

rm -f k.txt
touch k.txt

# Loop over 2, 3, 4, 6, and 8
for k in  2 3 4 6 8
do
cp ../DFT/scf.in .
echo $k

# Replace the k-point settings in scf.in with the current loop
sed -i "s/3  3  3/$k  $k  $k/" scf.in


# The 2x2x2 grid only has 2 k-points, so I need to set npool down to 2
srun -n 32 $OCEAN_BIN/pw.x -npool 2 -inp scf.in > scf.k${k}.out

grep '!' scf.k${k}.out | awk "{print $k, \$5*13.605 }" >> k.txt

done
```

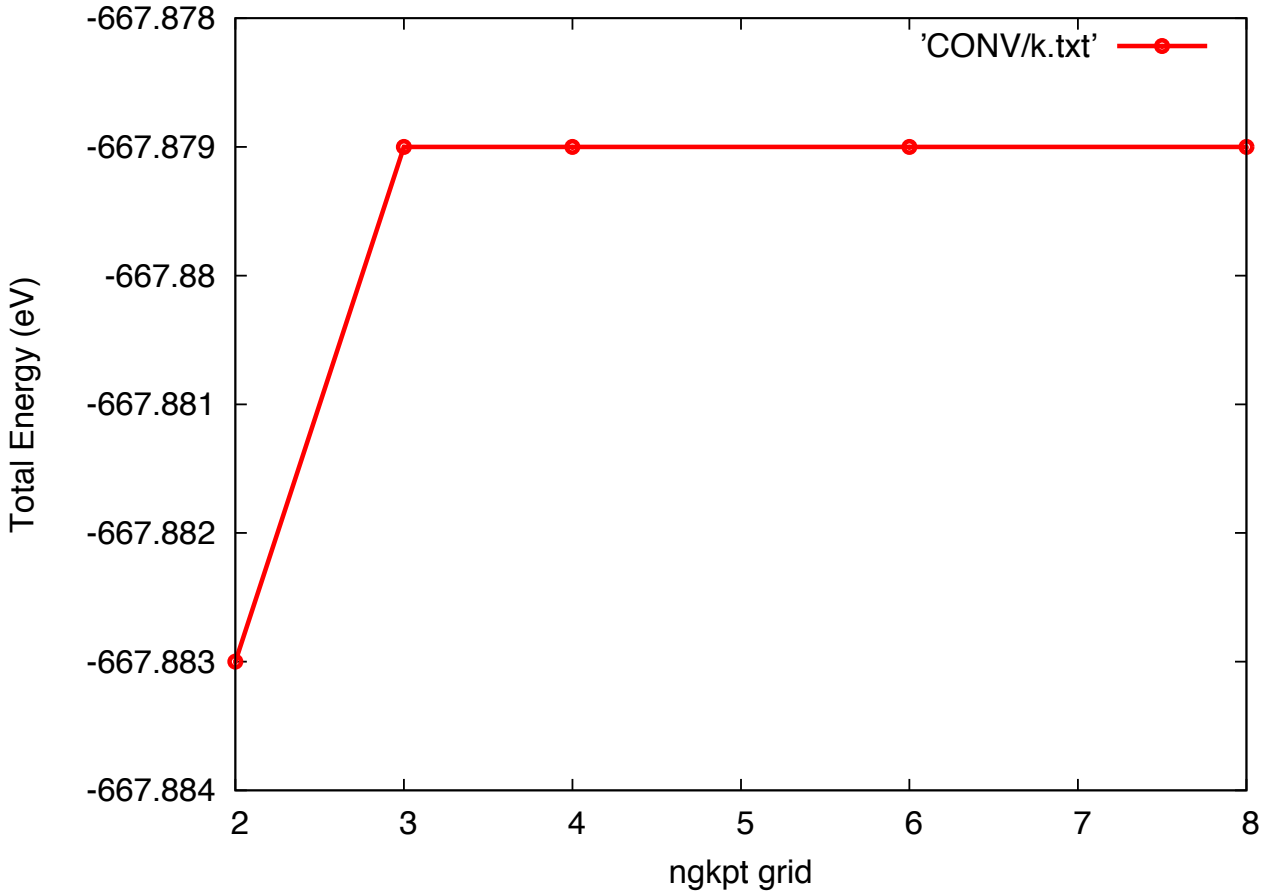
SCF convergence

- Start convergence script then examine it
  - > sbatch runConvergence
  - > less runConvergence
- First loop is for ngkpt
- Second is for ecut
  - 40 Ry – 120 Ry

```
grep '!' scf.k${k}.out | awk "{print $k, \$5*13.605 }" >> k.txt

done


rm -f e.txt
touch e.txt

# now we loop over energy cutoff
for e in 40 60 80 100 120
do
cp ../DFT/scf.in .
echo $e
sed -i "s/ecutwfc = 100/ecutwfc = $e/" scf.in

# I know that a 3x3x3 gives 6 k-points, by looking at the scf.out from the ocean run
# Therefore I change the pools to 6 and the number of processors down to the
# largest even multipule
srun -n 30 $OCEAN_BIN/pw.x -npool 6 -inp scf.in > scf.e${e}.out

grep '!' scf.e${e}.out | awk "{print $e, \$5*13.605 }" >> e.txt

done
```

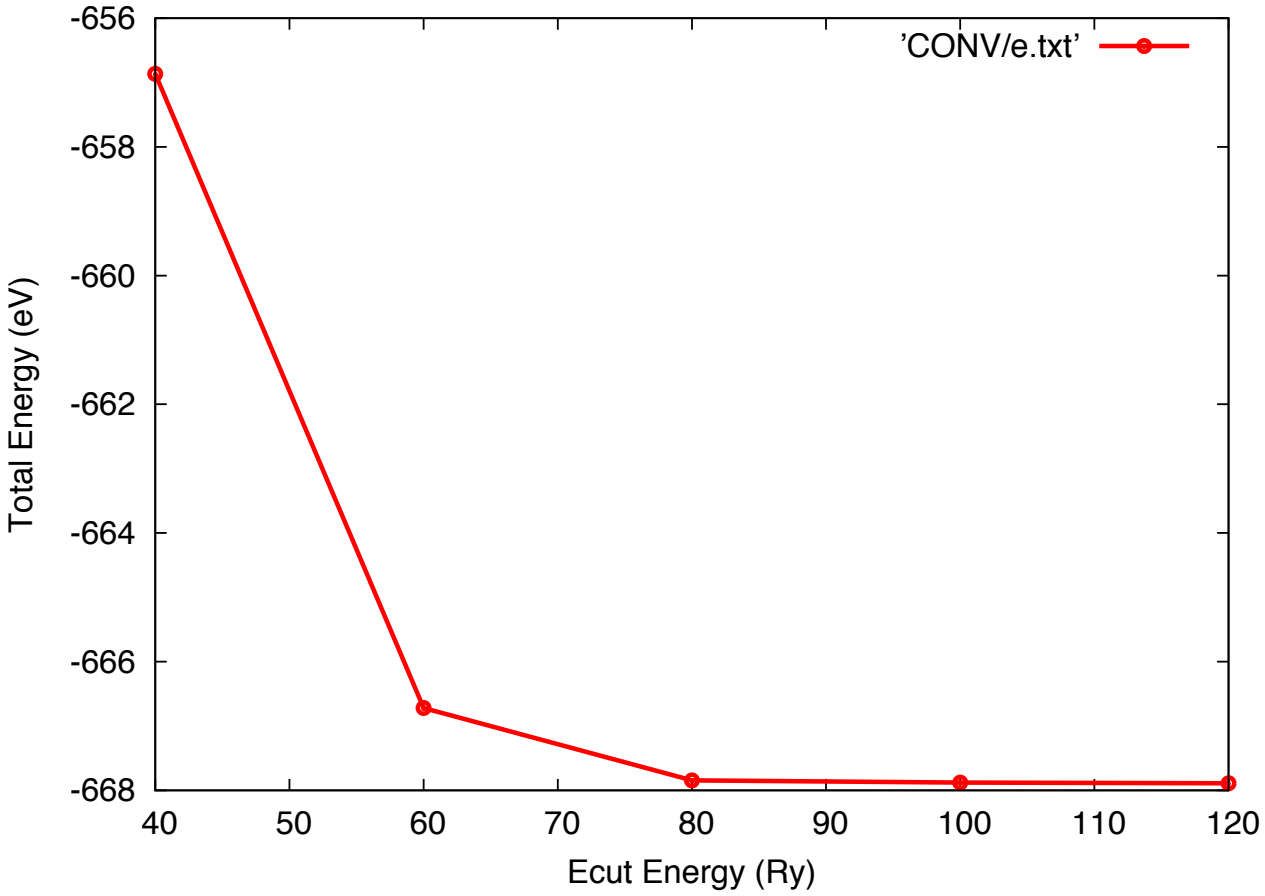
SCF convergence

- Start with ngkpt
  - > gnuplot
  - > p ‘CONV/k.txt’ w lp
- This is ngkpt
  - { 2 2 2 } – { 8 8 8 }
  - 3^3 is converged



SCF convergence

- Now ecut
  - > p ‘CONV/e.txt’ w lp
- Plane-wave energy cutoff
  - 40 Ry – 120 Ry
  - 80 Ry looks good
  - Zoom in, 100 Ry is better



SCF convergence

- Important to check
  - SCF is the foundation
  - DFT/scf.out has important info
    - Forces on atoms
    - Magnetization info when relevant
- Settings in LiF.in are already good!

Outline

- Running
- Stages of OCEAN
- First results
- Converging the DFT stage
- Input file(s)
  - Photon files
  - Main input
  - Pseudopotentials
  - fill & opts
- Convergence

Spectra files: absspct_F_.0001_1s_01

- absspct – absorption
  - xesspct is emission
  - rxsspct is RIXS
- F_ – fluorene atom
  - "_" for single letter elements
  - O_, N_, V_, …
  - Si, Ti, Cu, …
- 1s – the K edge
- 0001 – atom index
  - First fluorene in input
  - Consider amorphous SiO_2
    - 64 formula units
    - Si – 0001 – 0064
    - O – 0001 – 0128
- 01 – photon file index
  - Polarizations
  - Momentum direction & transfer (NRIXS)

Photon files

- photon1 – photon99
  - Automatic detection in run directory
- Completely specifies the electron-photon operator

Photon files: LiF/photon1

```
dipole
cartesian 0 0 1
end
cartesian 0 1 0
end
696
```

- Electron-photon operator
- Polarization direction
- Momentum direction*
- Photon energy*

*Not used for dipole

Photon files: LiF/photon2

quad

cartesian 0 0 1

end

cartesian 0 1 0

end

696

- Electron-photon operator
- Polarization direction
- Momentum direction
- Photon energy*

*Determines momentum magnitude

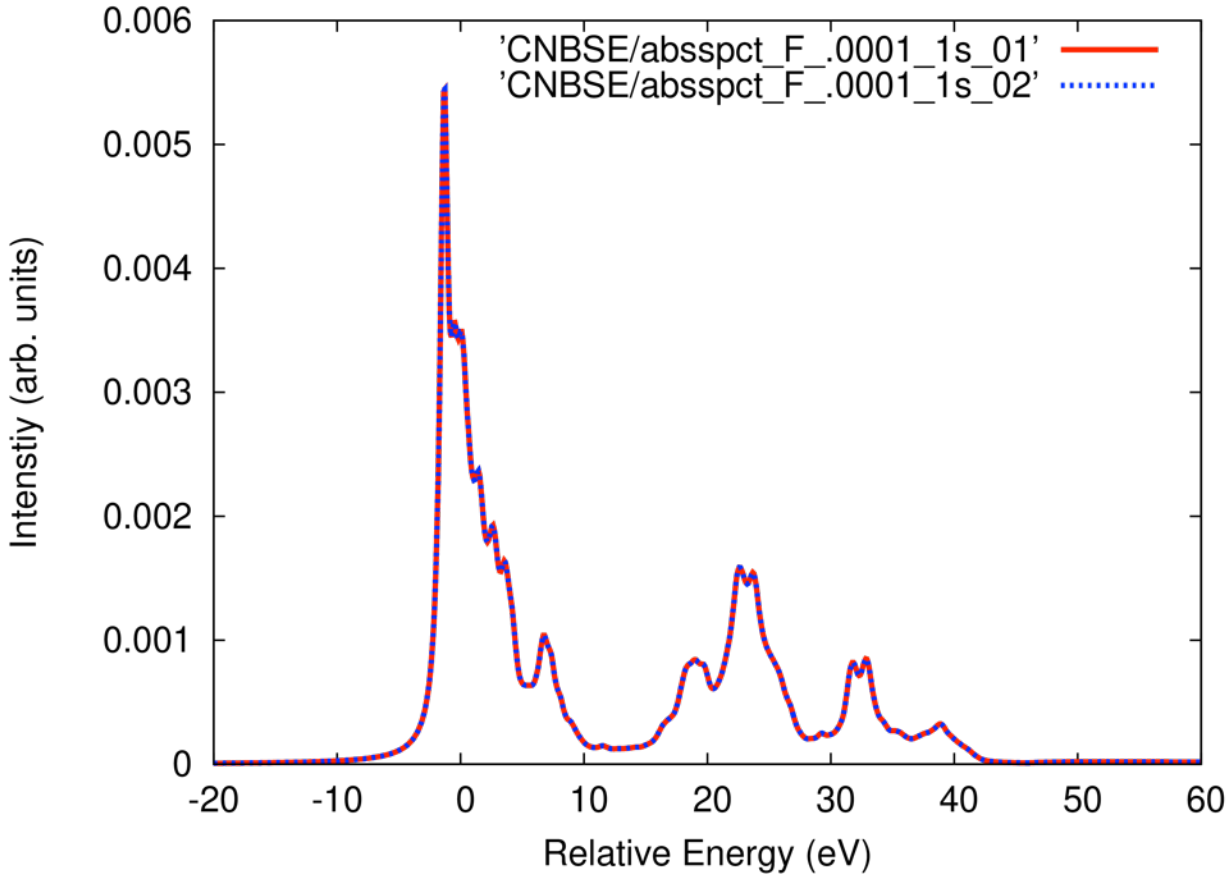
X-ray absorption operator

- $\varepsilon \cdot r + {}^{i}/_{2}(\varepsilon \cdot r)(\mathbf{q} \cdot r) + \cdots$

- (Simplified) dipole selection rules:
  - Δl = ±1, 0
  - No s → s
- Quadrupole
  - Requires large q – higher energy
  - Pre-edge features, e.g., 3d transition metal K edges
  - Averaging over directions is more complicated

LiF: No quadrupole contribution

- Gnuplot (again)
  - plot “CNBSE/absspct_F_.0001_1s_01” w lp
  - replot “CNBSE/absspct_F_.0001_1s_02” w l

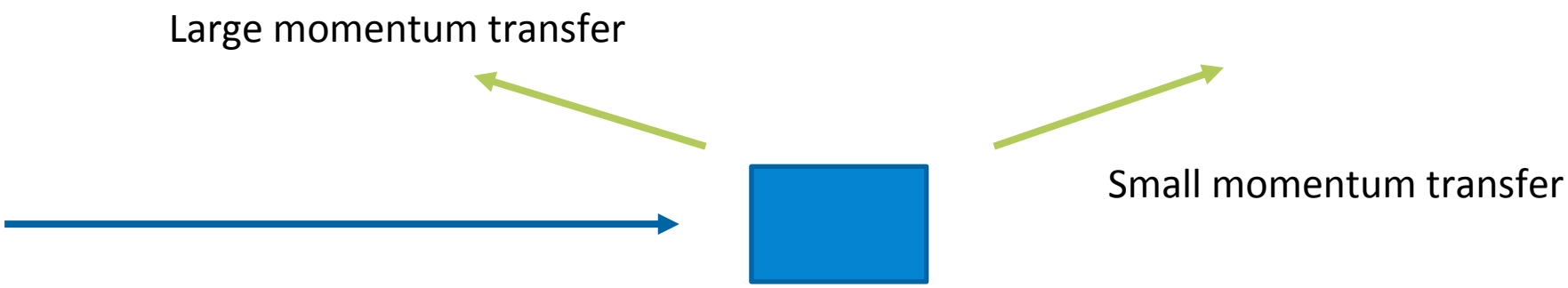


Photon files: qRaman

Non-resonant Inelastic X-ray Scattering (NRIXS)

or X-ray Raman Scattering (XRS)

- Energy and momentum decoupled



NRIXS operator

- Non-resonant Inelastic X-ray Scattering (X-ray Raman Scattering)
- Decouples transferred momentum and energy
  - High momentum at low energy edges
- $e^{i\mathbf{q}\cdot r} = 1 + i\,\mathbf{q}\cdot r - (\mathbf{q}\cdot r)^2/2 + \cdots$
  - For small q this is dipole like
  - Large q allows non-dipole transitions

Photon files: LiF/photon3

```
qRaman
cartesian 0 0 1
end
cartesian 0 1 0
length 3.55 inverseangstrom
end
696
```

- Electron-photon operator
- Polarization direction*

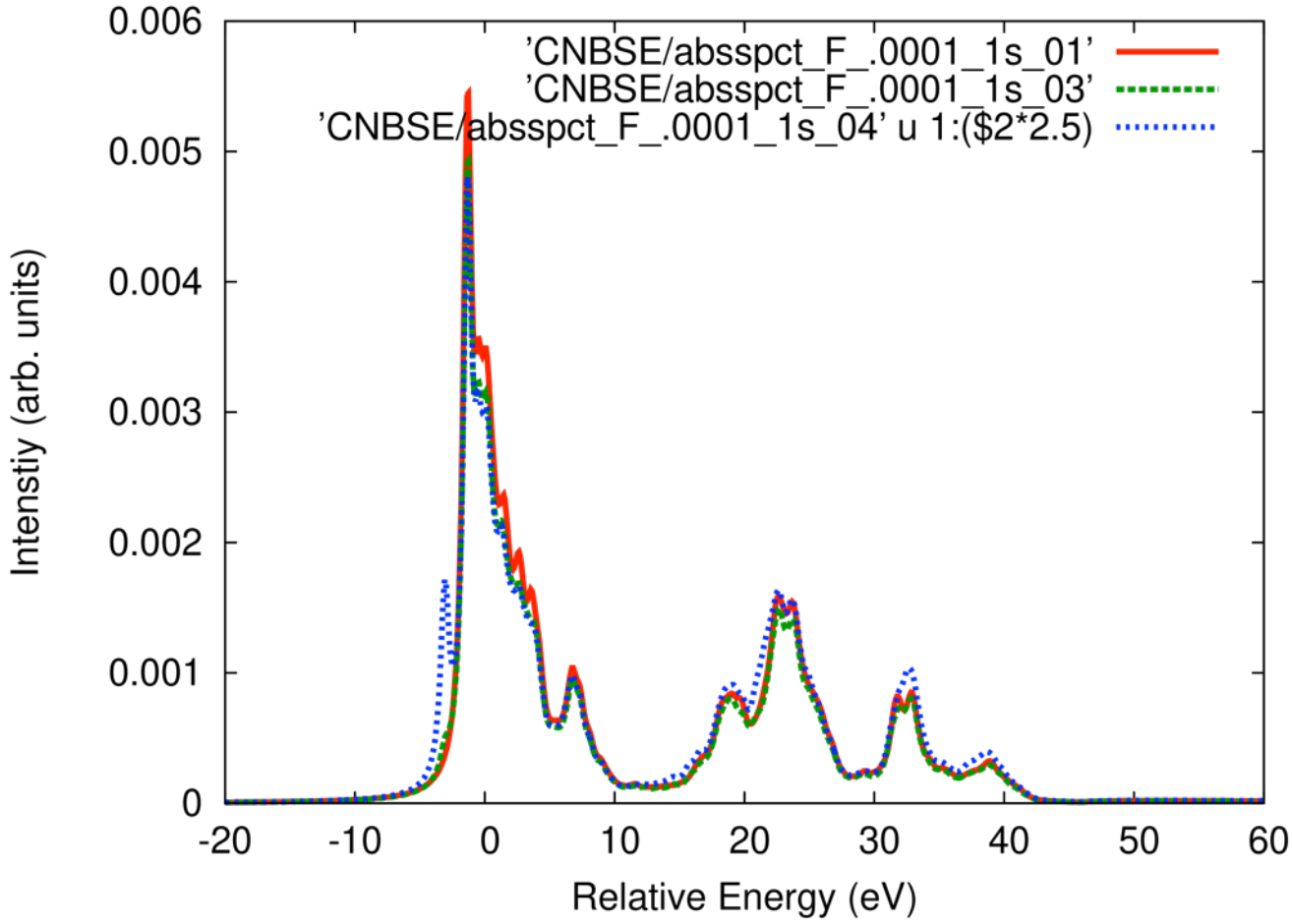
- Momentum direction
- Momentum transferred

- Photon energy*

*Not used for qRaman

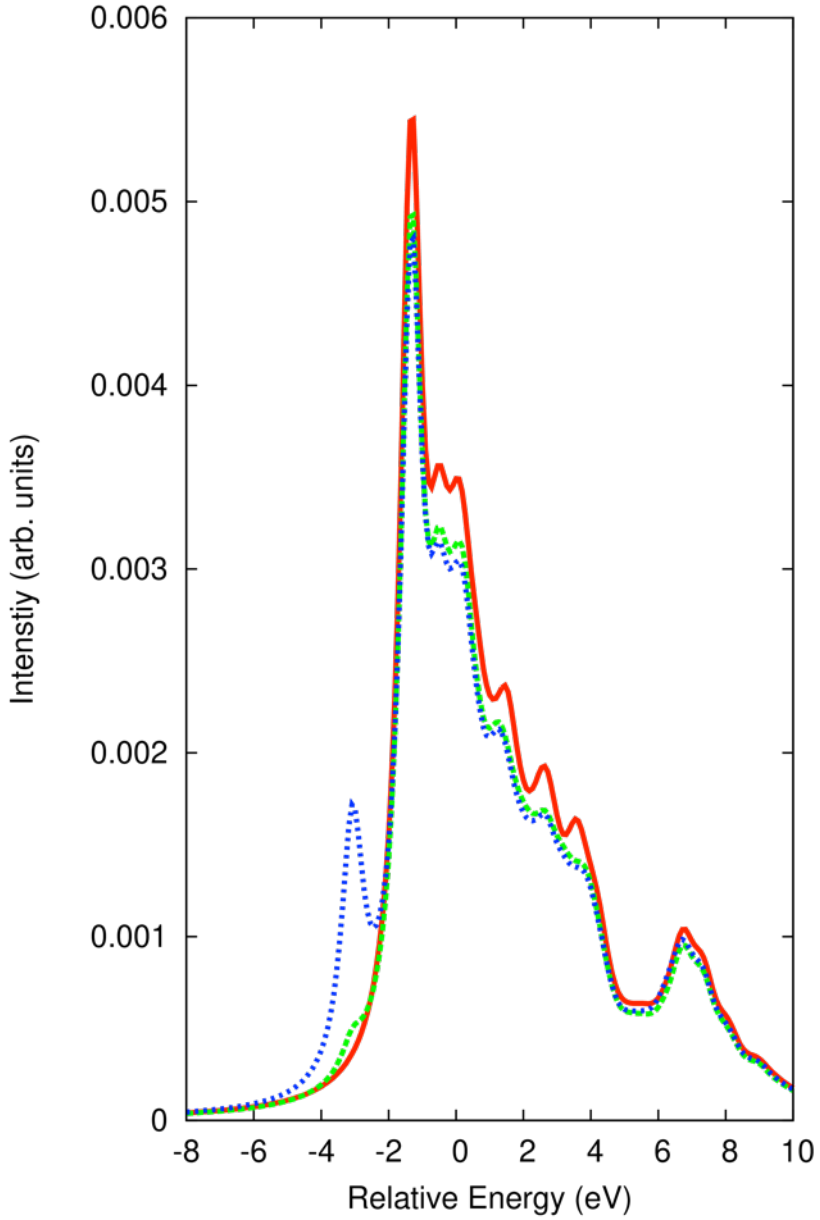
LiF: q-dependence

- plot “CNBSE/absspct_F_.0001_1s_01” w l, “CNBSE/absspct_F_.0001_1s_03” w l
- replot “CNBSE/absspct _F_.0001_1s_04” u 1:($2*3) w l



LiF: q-dependence

- Pre-edge shows strong momentum dependence
- *s*-type exciton is lowest energy excitation
- 1*s* → 3*s* is dipole forbidden



Outline

- Running
- Stages of OCEAN
- First results
- Converging the DFT stage
- Input file(s)
  - Photon files
  - Main input
  - Pseudopotentials
  - fill & opts
- Convergence

Main input: grammar

- # comment out all following text on a line

- Multi-word inputs **require** { }
  - “nkpt { 8 8 8 }” **not** “nkpt 8 8 8”
  - “ecut 100” **or** “ecut { 100 }”

- Separate inputs **must** be separated by a new line

Outline

- Main input
  - Control
  - Structure
  - Pseudopotentials
  - Edge information

Main input: Control

- dft qe
  - Select DFT code
  - qe (QuantumESPRESSO) or abi (ABINIT)
- para_prefix { srun -n 16 }
  - Parallel job launch
  - Common
    - mpirun
    - mpiexec
  - Number of processors should be first number
    - **WRONG**: mpirun --nper-node 8 -n 16

Main input: structure

- Lattice vectors, dielectric constant, elements, and atomic positions
- Adopted the style from ABINIT

Main input: structure

- acell { A B C }
- rprim{ a1 a2 a3
  b1 b2 b3
  c1 c2 c3 }
- Lattice vectors are (A*a1, A*a2, A*a3), etc
- Units are in Bohr

Main input: structure

- diemac 1.92

- Dielectric constant (epsilon infinity)
- ABINIT uses it as a preconditioner (minor)
- Needed for SCREENing (session 2)

Main input: structure

Atoms and locations:

- znucl { 3 9 }
  - Unique atomic numbers in cell
  - (trickier for magnetic systems)
- typat { 1 2 }
  - List of atoms, numbered by znucl
  - 1 → Z=3
  - 2 → Z=9
- xred { 0.5 0.5 0.5
  0.0 0.0 0.0 }
  - Reduced coordinates of atoms
  - Atom 1:
    - Located at ½ ½ ½
    - typat = 1 → Z=3 → Li
  - Atom 2:
    - Located at 0 0 0
    - typat = 2 → Z=9 → F

Outline

- Main input
  - Control
  - Structure
  - Pseudopotentials
  - Edge information

Main input: pseudopotentials

- pp_list {
  03-li.lda.fhi
  09-f.lda.fhi }
  - In the same order as znucl
  - WARNING: code isn’t smart enough to save you
  - By default, pseudos are in the run directory
- ecut 100
  - Planewave energy cut-off in Rydberg
  - Necessary cut-off depends on your pseudopotentials

Main input: edge information

- Each edge selection requires 3 integers: A N L
- Two methods of selecting edges
  1) { -9 1 0 } – Run every Z=9 atom in the cell
  2) { 2 1 0 } – Run the 2nd atom in the cell

- edges { -9 1 0 }
  - A = -9 : run every F atom (identical to { 2 1 0 })
  - N = 1, L = 0: 1s

Main input: edge information

- edges { -9 1 0 }
- opf.fill{ 9 f.fill }
- opf.opts{ 9 f.opts}
  - Z filename
  - Need fill and opts files for element being investigated
- cnbse.broaden 0.4
  - Broadening applied to final spectra in eV
  - Finite corehole lifetime

Outline

- Running
- Stages of OCEAN
- Input file(s)
  - Photon files
  - Main input
  - Pseudopotentials
  - fill & opts
- Convergence

Pseudopotentials

- Core electrons
  - Don’t participate in bonding
  - Don’t change with chemical environment
  - Require large numbers of plane-waves
- Remove them, modify atomic potential
  - Valence electrons now the most-bound allowed states
  - e.g. Si $1s^2 2s^2 2p^6 3s^2 3p^2$ becomes Si* $3s^2 3p^2$
    - Only 4 electrons
    - 3s/3p orbitals have NO nodes

Pseudopotentials

- Need a pseudopotential for each element
- Compatible formats:
  - fhi
  - oncv
- Useful tools:
  - opium
  - oncv
  - fhi2upf.x (QE distribution)

opts & fill files

- Auxiliary files for target element
  - Fluorene 1s needs f.opts & f.fill
  - Titanium 1s and 2p use same ti.opts & ti.fill
- Set once per pseudopotential

opts

- 009
- 1 0 0 0
- scalar rel
- lda
- 2.0 4.5 0.25 0.0
- 2.0 4.5 0.25 0.0

- Z
- Occ. of core shells: s p d f
- Treatment of relativity
- LDA or Hartree-Fock
- Occ. of valence shells: s p d f

fill

- 2
- 0.20 3.00 0.0001
- 3.0
- 0.05 20.

- npower
- Energy window
  - Min starts below lowest occupied
  - Max (both in Hartree)
  - Precision
- Cut-off radius
- Bessel transform controls

fill

- Energy Window: 0.20 3.00 0.0001
  - Solve DFT atom
  - Each angular momentum has a lowest valence energy E
  - Energy window from 0.20 below E to 3.00 Ha.
  - OPFs chosen to span this window

- Cut-off radius: 3.0 Bohr
  - Defines space for OPFs
  - Must be larger than the pseudopotential cutoff

Outline

- Running
- Stages of OCEAN
- First results
- Converging the DFT stage
- Input file(s)
- Convergence
  - Back to main input file

Main input: convergence

1) Bands
2) K-points
3) X-points

Main input: converging bands

So that we aren't waiting!

First, keep a copy of CNBSE

- > mv CNBSE CNBSE.1

Edit LiF.in

- # dft_energy_range 50
- nbands 24

Re-run OCEAN

- > sbatch runit

Main input: converging bands

Specify one of:

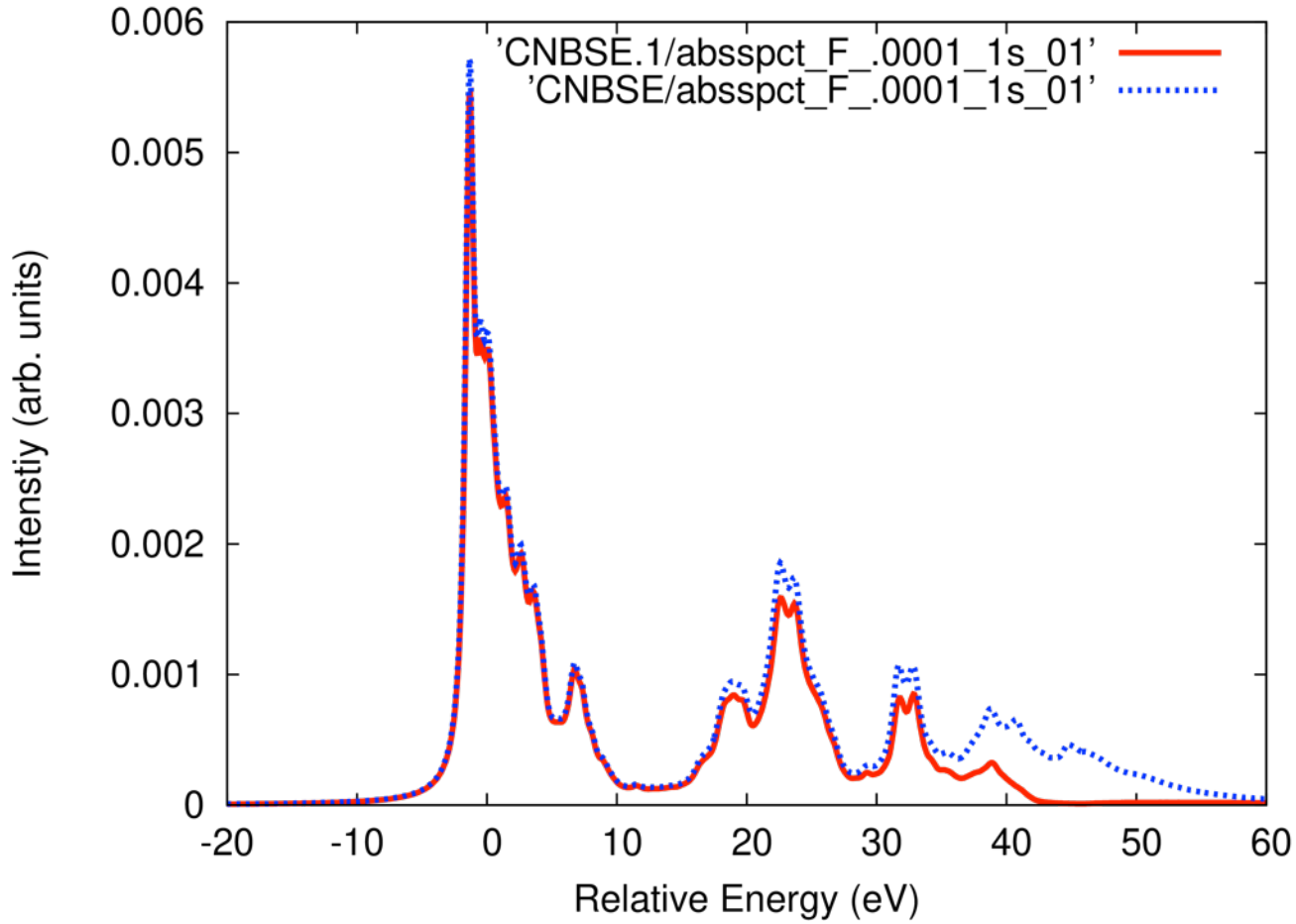
- nbands
- dft_energy_range

Scroll back through the OCEAN output to near the beginning

> " Default requested for nbands. Energy range is 50 eV.
> Default chosen for nbands: 18"

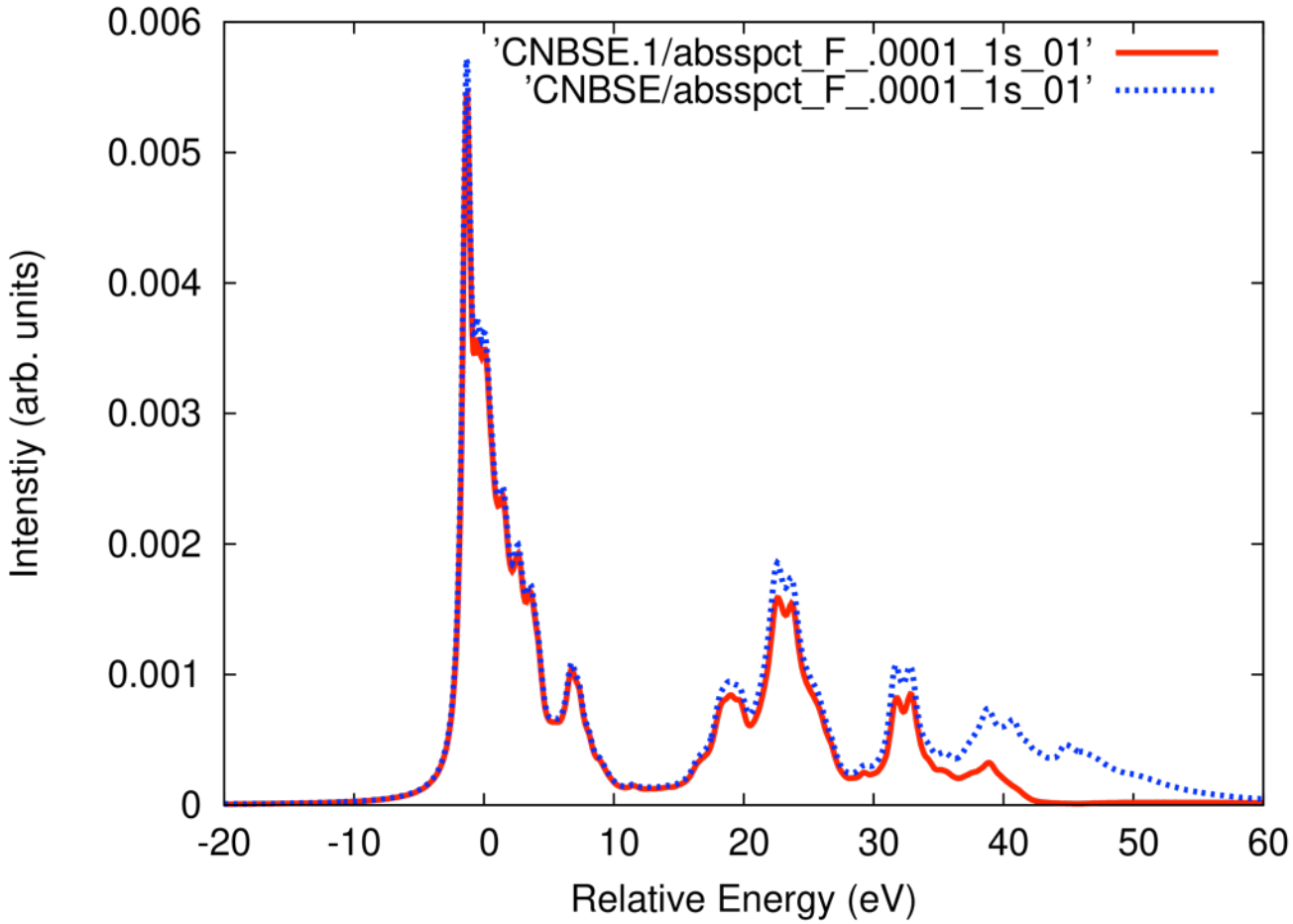
Main input: converging bands

- Gnuplot!
  - plot “CNBSE.1/absspct_F_.0001_1s_01” w l
  - replot “CNBSE/absspct_F_.0001_1s_01” w l



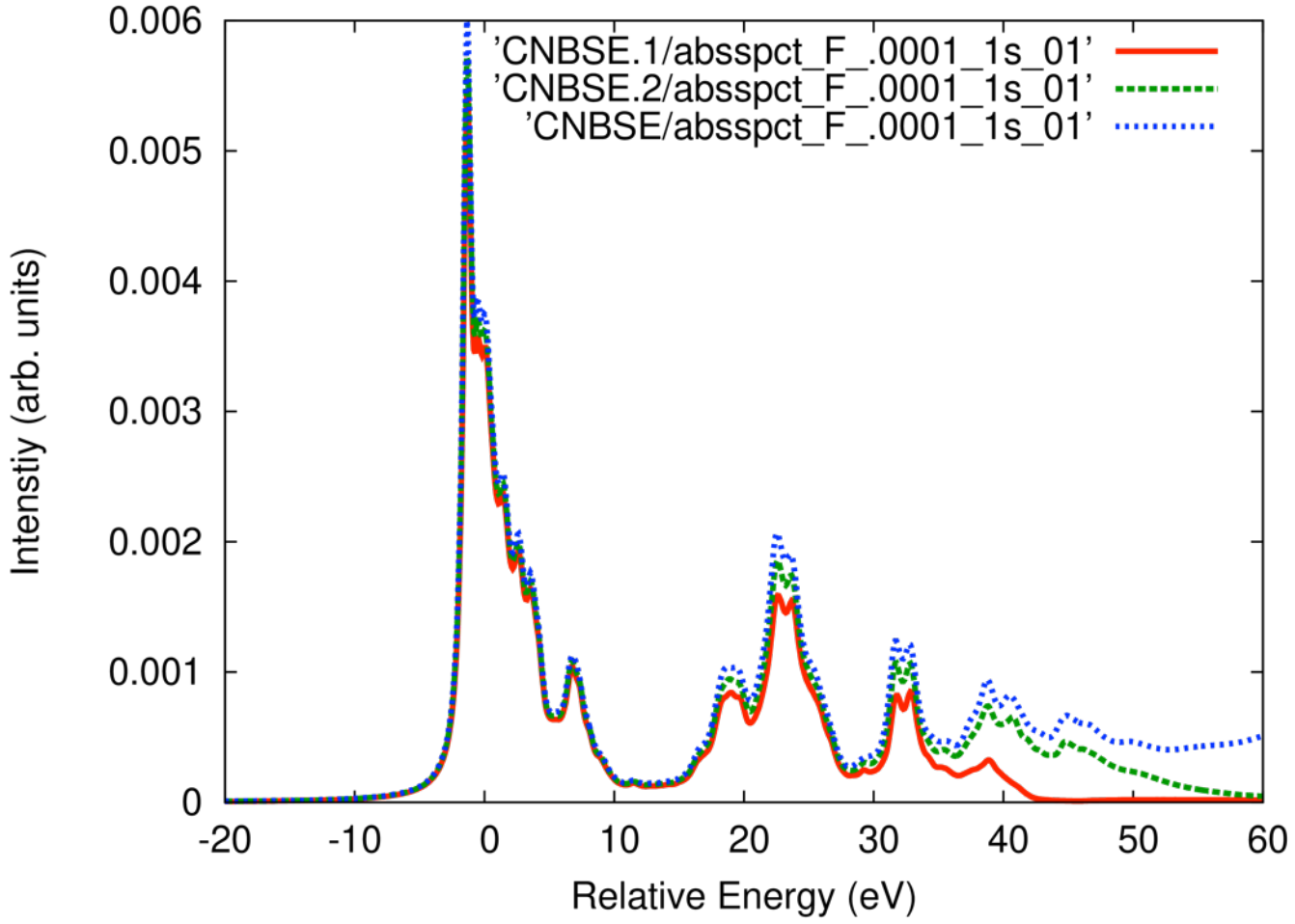
Main input: converging bands

- More bands → Higher energy
- Bands are from ground-state
  - Not basis states of the BSE
  - Even high-energy bands can contribute at the exciton
- nbands **includes** occupied
  - (might change in v3)



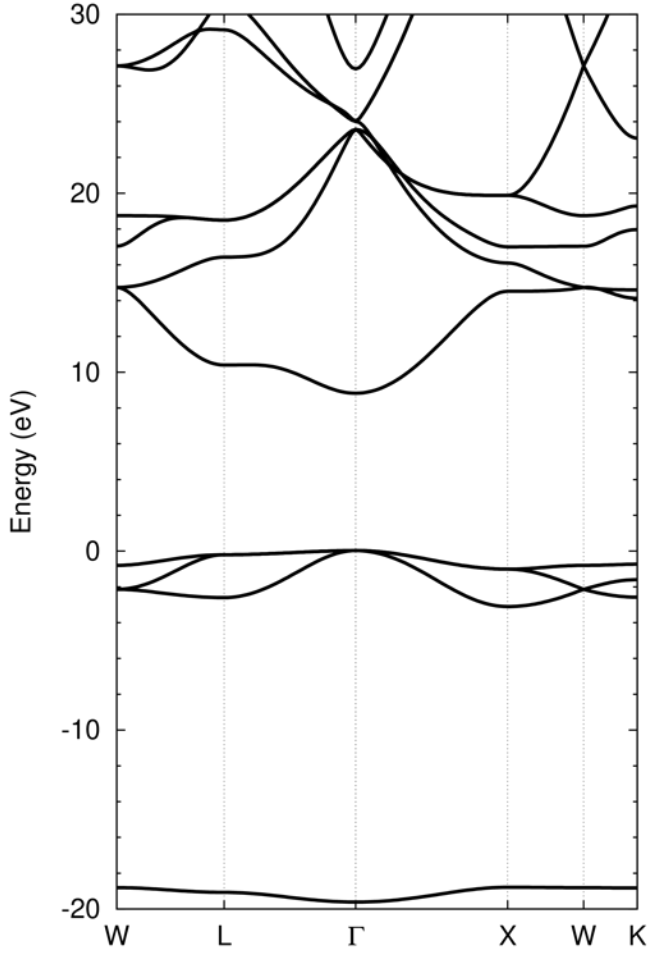
Main input: converging bands

- Change nbands to 40
- Re-run
  - > mv CNBSE CNBSE.2
  - > sbatch runit
- When done
  - > mv CNBSE CNBSE.3
- Good time for questions



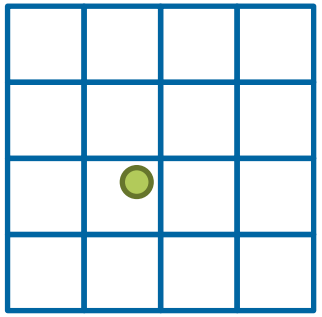
Main input: converging k-points

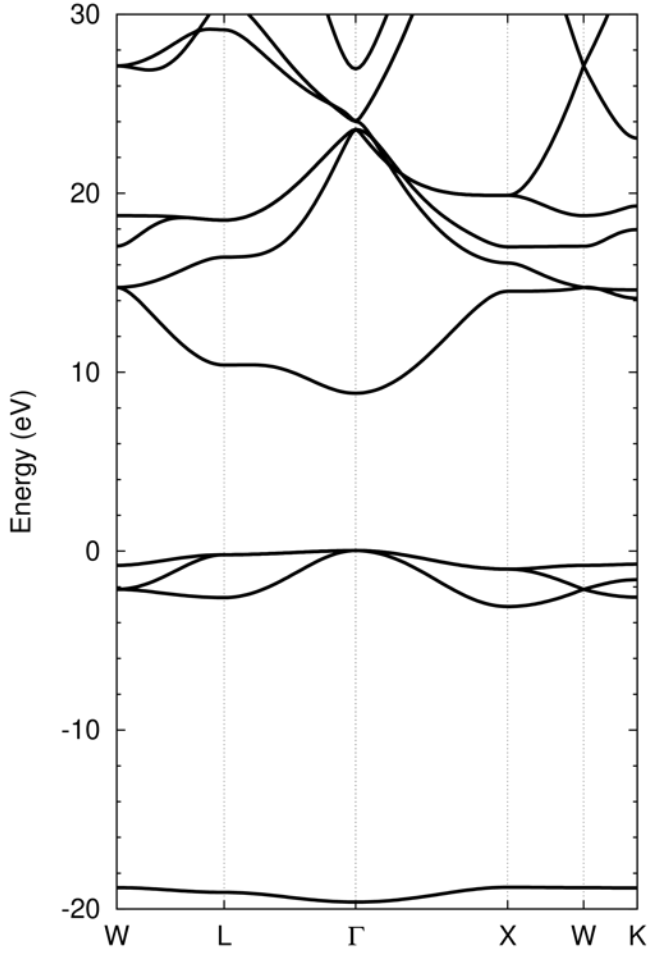
- First: edit LiF.in & re-run OCEAN
  - nbands 24
  - nkpt { 10 10 10 }
- Meanwhile: What does k-point sampling do?
  1) Captures band dispersion



Main input: converging k-points

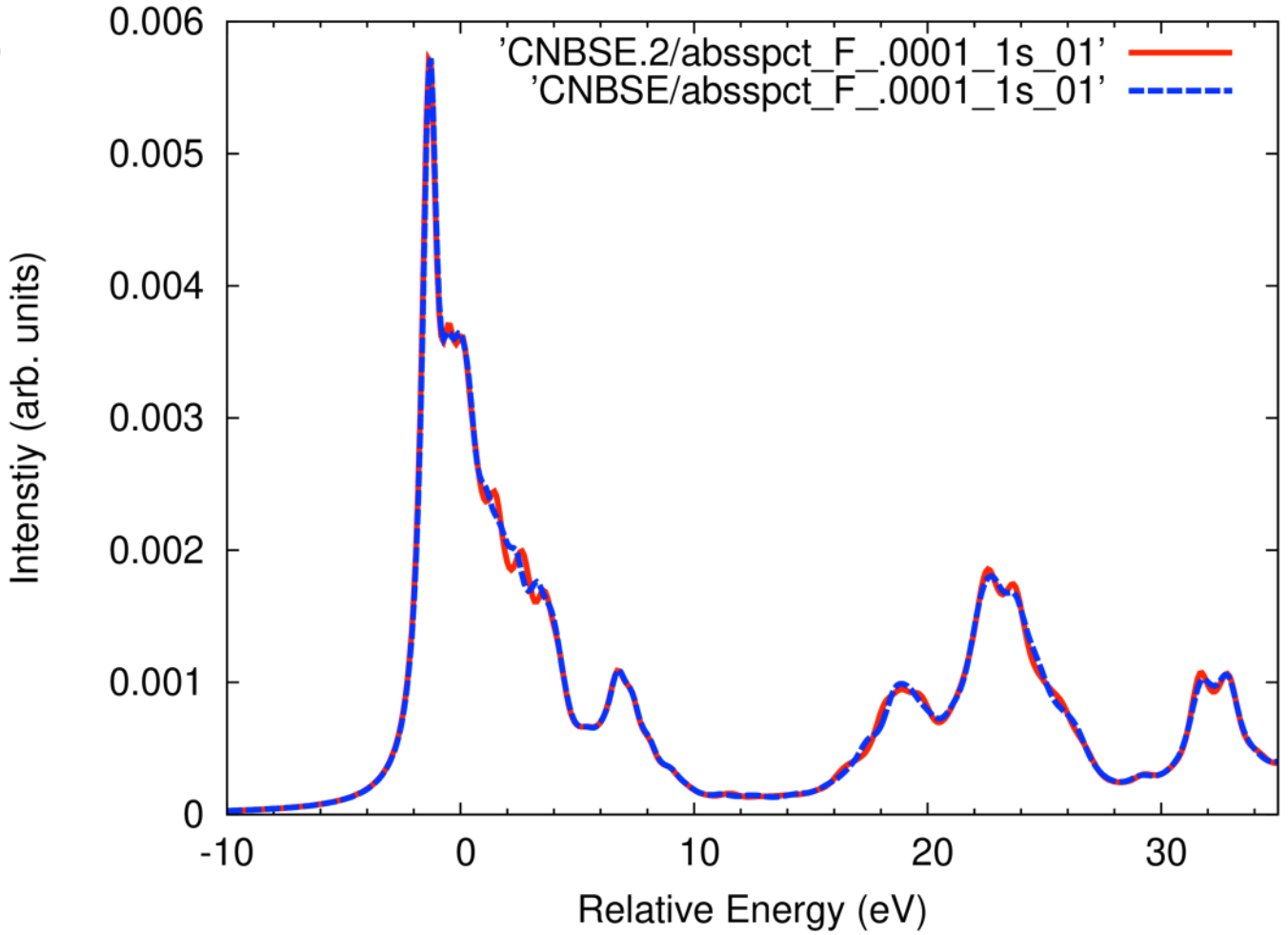
- What does k-point sampling do?
  1) Captures band dispersion
  2) Defines a supercell
     - $\exp\left[i\mathbf{k}\cdot\mathbf{R}\right]$





Main input: converging k-points

- 8^3 (CNBSE.2) vs. 10^3 (CNBSE)



Main input: converging x-points

Look at the top of the last OCEAN log (slurm.??????)

" Defaults requested for xmesh.ipt

Defaults chosen for xmesh.ipt: 4 4 4 "

Keep CNBSE

- > mv CNBSE CNBSE.4

Edit LiF.in,

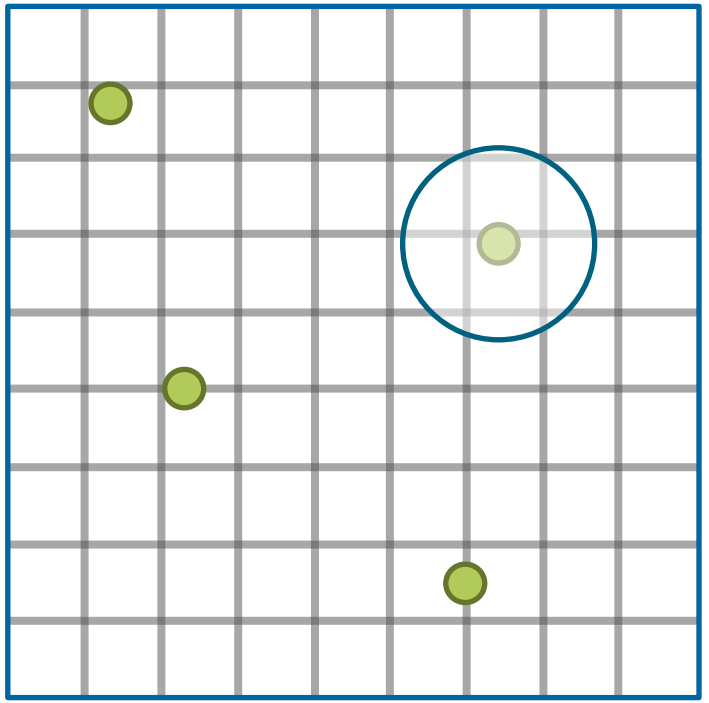
- cnbse.xmesh { 6 6 6 }

Re-run OCEAN

- > sbatch runit

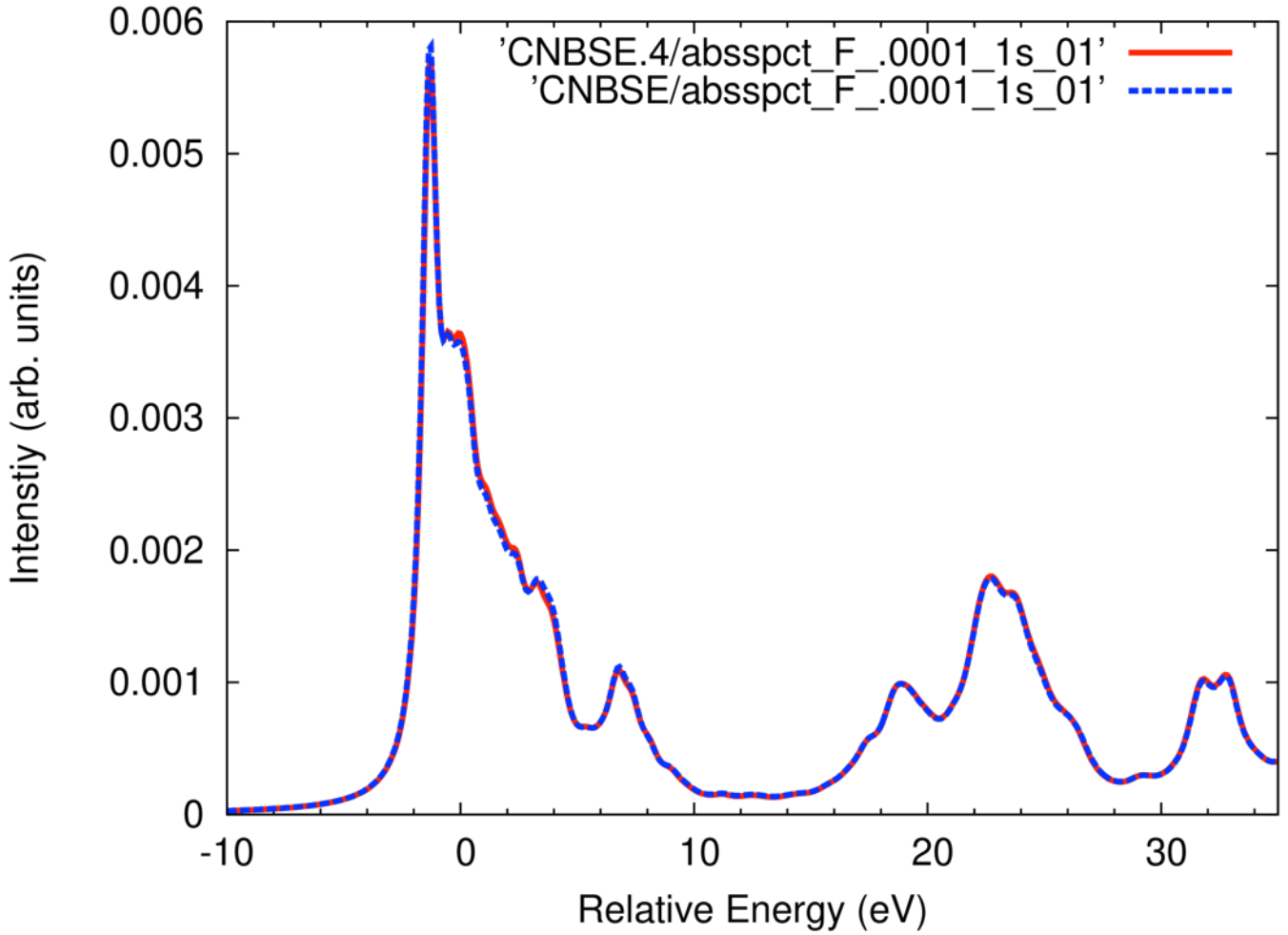
Main input: converging x-points

- X-points are within the unit cell
  - DFT codes tend to use G – the Fourier pair to x
  - More x/G points needed for DFT than for BSE
- OCEAN has two real-space representations
  1) X-points are a regular grid (along lattice vectors)
  2) OPFs provide a local basis around absorbing atom
- Number x-points > number of bands



Main input: converging x-points

- 4^3 vs. 6^3
- LiF has a small unit cell
- 1-2 x-points / Bohr



Convergence; final thoughts

- Both x-point and k-points are FFT grids internally
  - Choose small prime factors: 2, 3, 5
- Only converge what experiment sees
  - Limited energy range
  - Core-hole lifetime broadening
- Number of x-points grow with cell size
  - 1-2 x-points / Bohr
- Number of k-points shrink with cell size
  - Metals require more than insulators