

# Resistivity, drift and Hall mobility with EPW

## Tutorial Wed.4

### Hands-on session

Hands-on based on Quantum ESPRESSO development version (v6.7+) and EPW v5.4Beta

In this tutorial, you will learn how to compute electric resistivity in metals and carrier mobility (drift and Hall) in semiconductors.

You can start by going into your scratch folder and copying the tar file which contains all necessary files for this tutorial and extracting it.

```
$ cd $SCRATCH
$ cp /work2/06868/giustino/EPW-SCHOOL/Wed.4.Ponce.tar .
$ tar -xvf Wed.4.Ponce.tar; cd Wed.4.Ponce
```

**Note:** If you are doing the tutorial on your desktop or local machine, the tutorial files can be downloaded from the school webpage <https://epw2021.oden.utexas.edu/74-schedule>.

Inspect the content of the folder:

```
$ ls
exercise1 exercise2 job.sh Wed.4.Ponce.pdf
```

You can see that it contains the two exercises folders, a copy of the pdf of this hands-on and a submission job for Frontera which reads as follow:

```
--                                                                 job.sh
#!/bin/bash
#SBATCH -J myjob           # Job name
#SBATCH -N 1              # Total # of nodes
#SBATCH --ntasks-per-node 8
#SBATCH -t 01:00:00      # Run time (hh:mm:ss)
#SBATCH -A EPW-SCHOOL
#SBATCH -p small
#SBATCH --reservation=EPW-SCHOOL-06-16-2021

module list

export PATHQE=/work2/06868/giustino/EPW-SCHOOL/q-e

# Launch MPI code...
ibrun $PATHQE/bin/pw.x -input scf.in > scf.out
ibrun $PATHQE/bin/ph.x -input ph.in > ph.out
```

**Note:** If you are running on your desktop, you will probably have to use a command such as:

```
$ mpirun -np 8 $PATHQE/bin/pw.x -input scf.in > scf.out
$ mpirun -np 8 $PATHQE/bin/ph.x -input ph.in > ph.out
```

### Exercise 1

In this example we are going to calculate the electric resistivity of fcc Pb using the Ziman formula and the Boltzmann transport equation (BTE).

First go in the first exercise and copy the submission script:

```
$ cd exercise1
$ cp ../job.sh .
```

---

► Make a self-consistent calculation for Pb and a phonon calculation on a homogeneous 3x3x3 q-point grid.

**Note:** The `ecutwfc` need to be much larger for real calculations.

```
$ sbatch job.sh
```

This command will submit the following self-consistent and phonon calculations to the cluster:

```
--                                                                    scf.in
&control
  calculation      = 'scf'
  prefix           = 'pb'
  restart_mode     = 'from_scratch'
  wf_collect       = .true.
  pseudo_dir       = './'
  outdir           = './'
  verbosity        = 'high'
  tprnfor          = .true.
  tstress          = .true.
/
&system
  ibrav            = 2
  celldm(1)        = 9.27
  nat              = 1
  ntyp             = 1
  ecutwfc          = 30
  occupations      = 'smearing'
  smearing         = 'mp'
  degauss          = 0.025
/
&electrons
  diagonalization = 'david'
  mixing_beta     = 0.7
  conv_thr        = 1.0d-12
/
ATOMIC_SPECIES
Pb 207.2 pb_s.UPF
ATOMIC_POSITIONS crystal
Pb 0.0 0.0 0.0
K_POINTS automatic
12 12 12 0 0 0
```

```
--                                                                    ph.in
&inputph
  recover          = .false.
  tr2_ph           = 1.0d-17,
  prefix           = 'pb',
  fildyn           = 'pb.dyn.xml',
  fildvscf         = 'dvscf'
  ldisp            = .true.,
  nq1              = 3,
  nq2              = 3,
  nq3              = 3
/
```

The important keyword is `fildvscf` as it will tell the code to write to file the change of potential due to ionic displacement  $\partial_{q\nu} V^{\text{scf}}$ . The `ldisp` input allows to calculate phonons for a grid of q-points specified by `nq1`, `nq2`, and `nq3`.

**Note 1:** For real calculations the coarse q-point grid should be converged on and is typically 6x6x6 or 8x8x8.

**Note 2:** The `tr2_ph` variable is the threshold on the perturbed wavefunction obtained by solving the Sternheimer equation and should be very small.

**Note 3:** It is recommended to add `XXX.xml` at the end of the `fildyn` as it will force the code to write the output in XML format.

This should take about 5 min to be completed. In the output file, locate the list of 4 irreducible  $q$  points in the Brillouin Zone (IBZ):

```
Dynamical matrices for ( 3, 3, 3) uniform grid of q-points
( 4 q-points):
  N      xq(1)      xq(2)      xq(3)
  1  0.000000000  0.000000000  0.000000000
  2 -0.333333333  0.333333333 -0.333333333
  3  0.000000000  0.666666667  0.000000000
  4  0.666666667 -0.000000000  0.666666667
```

For each  $q$ -point, a `pb.dynX.xml` file containing the dynamical matrix has been produced. The `pb.dvscf` files are located inside the `_ph0` folder.

► Gather the `.dyn`, `.dvscf` and `patterns`<sup>1</sup> files into a new save directory. This can easily be done using the `pp.py` python script.

```
$ export PATHQE=/work2/06868/giustino/EPW-SCHOOL/q-e
```

```
$ python $PATHQE/EPW/bin/pp.py
```

**Note:** Python2 requires to install the future library. Alternatively you can use:

```
$ python3 $PATHQE/EPW/bin/pp.py
```

The script will ask you to provide the prefix of your calculation (here "pb").

► Do a non self-consistent calculation on a homogeneous  $3 \times 3 \times 3$  **uniform and  $\Gamma$ -centered grid between  $[0,1[$  in crystal coordinates.**

```
--
nscf.in
&control
  calculation = 'nscf'
  prefix      = 'pb'
  restart_mode = 'from_scratch'
  wf_collect  = .true.
  pseudo_dir  = './'
  outdir      = './'
  verbosity   = 'high'
/
&system
  ibrav       = 2
  celldm(1)   = 9.27
  nat         = 1
  ntyp        = 1
  ecutwfc     = 30
  occupations = 'smearing'
  smearing    = 'mp'
  degauss     = 0.025
  nbnd        = 10
/
&electrons
  diagonalization = 'david'
  mixing_beta     = 0.7
  conv_thr        = 1.0d-12
/
ATOMIC_SPECIES
Pb 207.2 pb_s.UPF
ATOMIC_POSITIONS crystal
Pb 0.000000000 0.000000000 0.000000000
K_POINTS crystal
27
0.0000000000000000 0.0000000000000000 0.0000000000000000 0.037037037037
0.0000000000000000 0.0000000000000000 0.3333333333333333 0.037037037037
...
```

<sup>1</sup>The `patterns` file contains the basis in which the `.dvscf` are defined.

---

```
$ ibrun $PATHQE/bin/pw.x -input nscf.in > nscf.out
```

**Note:** You will need to update the `job.sh` script to add that line and comment/remove the previous lines. You can then submit the job.

► Perform an EPW calculation to interpolate the electron-phonon matrix element from a coarse  $3\times 3\times 3$  to a dense  $18\times 18\times 18$   $k$ -point and  $q$ -point grids.

```
--                                                                 epw1.in
&inputepw
  prefix      = 'pb'
  amass(1)    = 207.2
  outdir      = './'
  dvscf_dir   = './save'

  elph        = .true.
  epbwrite    = .true.
  epbread     = .false.
  epwwrite    = .true.
  epwread     = .false.
  vme         = 'wannier'

  nbndsub     = 4
  bands_skipped = 'exclude_bands = 1-5'

  wannierize  = .true.
  num_iter    = 300
  dis_win_max = 21
  dis_froz_min= -3
  dis_froz_max= 13.5
  proj(1)     = 'Pb:sp3'
  wdata(1)    = 'bands_plot = .true.'
  wdata(2)    = 'begin kpoint_path'
  wdata(3)    = 'G 0.00 0.00 0.00 X 0.00 0.50 0.50'
  wdata(4)    = 'X 0.00 0.50 0.50 W 0.25 0.50 0.75'
  wdata(5)    = 'W 0.25 0.50 0.75 L 0.50 0.50 0.50'
  wdata(6)    = 'L 0.50 0.50 0.50 K 0.375 0.375 0.75'
  wdata(7)    = 'K 0.375 0.375 0.75 G 0.00 0.00 0.00'
  wdata(8)    = 'G 0.00 0.00 0.00 L 0.50 0.50 0.50'
  wdata(9)    = 'end kpoint_path'
  wdata(10)   = 'bands_plot_format = gnuplot'

  elecselven = .false.
  phonselven = .true.
  a2f        = .true.
  delta_approx = .true.
  nc         = 4.0d0 ! Number of carriers for the Ziman resistivity formula

  fsthick    = 6    ! eV
  temps      = 1    ! K
  degaussw   = 0.1  ! eV
  degaussq   = 0.05 ! meV
  assume_metal = .true.
  ngaussw    = -99  ! we want F-D distribution for metals

  nkf1       = 18
  nkf2       = 18
  nkf3       = 18

  nqf1       = 18
  nqf2       = 18
  nqf3       = 18

  nk1        = 3
  nk2        = 3
  nk3        = 3

  nq1        = 3
  nq2        = 3
```

---

```
nq3      = 3
/
```

---

There are two ways to compute the resistivity of Pb:

- using the phonon self-energy and the Eliashberg transport spectral function in conjunction with the Ziman formula
- using the Boltzmann transport equation

We will do both and start with the first one for which we need the isotropic transport spectral function:

$$\alpha_{\text{tr}}^2 F(\omega) = \frac{1}{2} \sum_{\nu} \int_{\text{BZ}} \frac{d\mathbf{q}}{\Omega_{\text{BZ}}} \omega_{\mathbf{q}\nu} \lambda_{\text{tr},\mathbf{q}\nu} \delta(\omega - \omega_{\mathbf{q}\nu}), \quad (1)$$

where the mode-resolved transport coupling strength is defined by:

$$\lambda_{\text{tr},\mathbf{q}\nu} = \frac{1}{N(\varepsilon_F) \omega_{\mathbf{q}\nu}} \sum_{nm} \int_{\text{BZ}} \frac{d\mathbf{k}}{\Omega_{\text{BZ}}} |g_{mn,\nu}(\mathbf{k}, \mathbf{q})|^2 \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_F) \delta(\varepsilon_{m\mathbf{k}+\mathbf{q}} - \varepsilon_F) \left(1 - \frac{v_{n\mathbf{k}} \cdot v_{m\mathbf{k}+\mathbf{q}}}{|v_{n\mathbf{k}}|^2}\right). \quad (2)$$

The calculation of the spectral function in EPW is given by the two keywords `phonsselfen = .true.` and `a2f = .true.`

**Note 1:** The `dvscf_dir = './save'` specify the place where we have placed the `.dyn`, `.dvscf` and `patterns` using the python script.

**Note 2:** Here we are using the `delta_approx = .true.` to approximate the double  $\delta$  in Eq. (2). In this case the broadening of Dirac deltas is approximated by a Gaussian of widths given by `degaussw = 0.1` and the Dirac delta in Eq. (1) by `degaussq = 0.05`.

**Note 3:** The variable `ngaussw` is used for calculation of the Fermi level and DOS and is a Fermi-Dirac distribution function (input -99) of electronic temperature given by `temps = 1` K such that all the files names will end in `XXX.1.000K`.

```
$ ibrun $PATHQE/bin/epw.x -npool 8 -input epw1.in > epw1.out
```

Since EPW does not yet support G-vector parallelization, we are using **k**-point parallelization. For this reason, you need to specify the `-npool` keyword.

The calculation should take about 6 min to be completed. While the calculation is running, notice in the `epw1.out` the different steps a full EPW run goes into. First the Wannierization, then the unfolding into the full 3x3x3 BZ, then the Fourier transform to real space and then finally the interpolation into the fine 18x18x18 **k** and **q** grids.

At the end of the calculation, you should get:

```
=====
Eliashberg Spectral Function in the Migdal Approximation
=====

lambda :      1.9271341
lambda_tr :    1.4564165

Estimated Allen-Dynes Tc

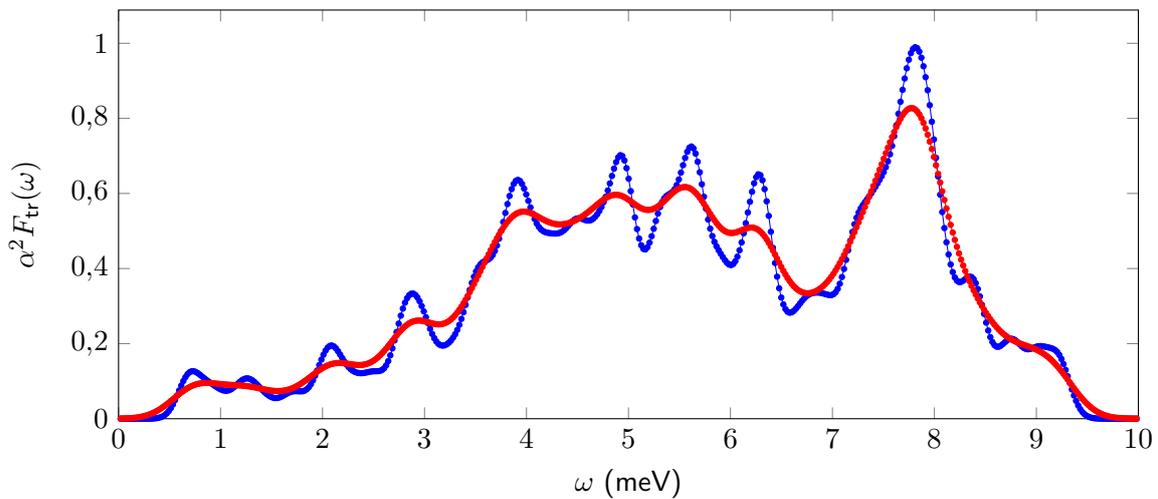
logavg =      0.0002066 l_a2f =      1.9281703
mu =      0.10 Tc =      4.573698490498 K
mu =      0.12 Tc =      4.363659787196 K
```

mu =	0.14	Tc =	4.152675719793	K
mu =	0.16	Tc =	3.941000842563	K
mu =	0.18	Tc =	3.728922987482	K
mu =	0.20	Tc =	3.516766489845	K

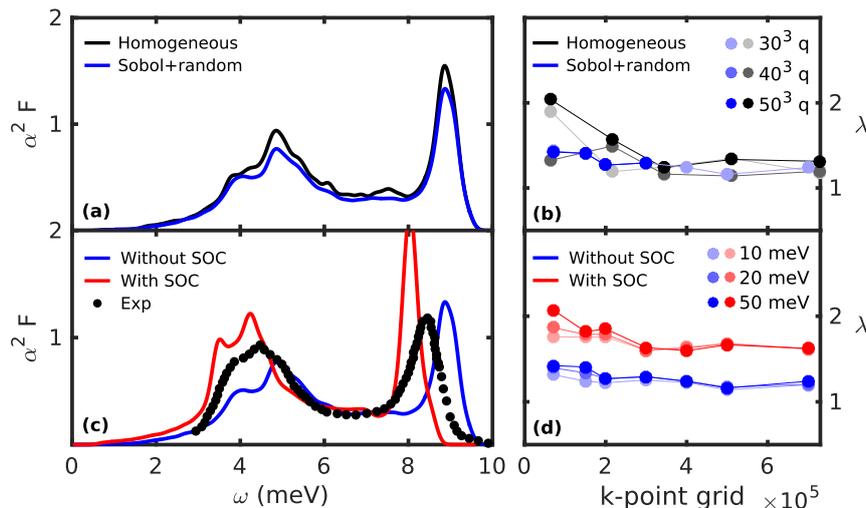
Note that the converged value for  $\lambda$  and  $\lambda_{tr}$  should be around 1.1. In addition the files `pb.a2f.01.1.000` and `pb.a2f_tr.01.1.000` which contain the Eliashberg spectral function and transport spectral function for different broadening values should have been produced.

**Tip:** Look at the end of the `pb.a2f_tr.01.1.000` file to know which column corresponds to which broadening.

You should get something similar to this (here shown for two broadening values 0.15 meV (blue) and 0.3 meV (red)):



Again this is unconverged. At convergence you should get something closer to <sup>2</sup>:



► Compute the resistivity of Pb using the Ziman's formula for metals:

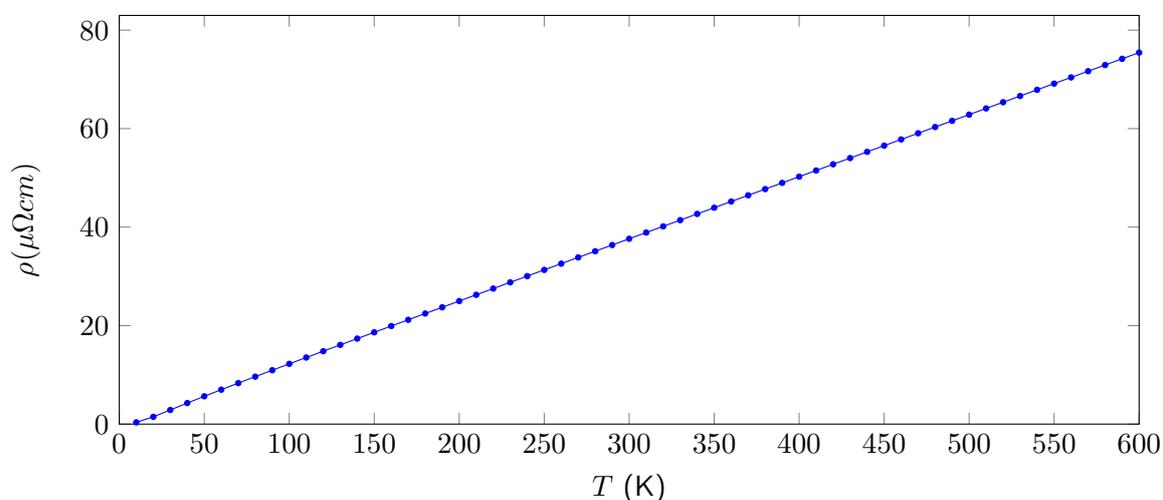
$$\rho(T) = \frac{4\pi m_e}{ne^2 k_B T} \int_0^\infty d\omega \hbar \omega \alpha_{tr}^2 F(\omega) n(\omega, T) [1 + n(\omega, T)], \quad (3)$$

<sup>2</sup>The figure is from [Comput. Phys. Commun. 209, 116 \(2016\)](#).

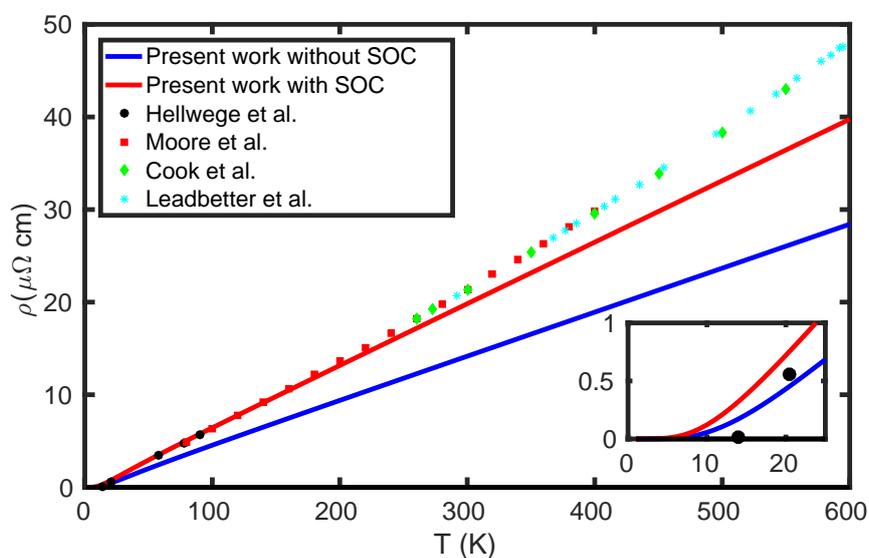
where  $n$  is the number of electrons per unit volume and  $n(\omega, T)$  is the Bose-Einstein distribution. Usually this means *the number of electrons that contribute to the mobility* which is 4.0 (can be fractional) for the case of Pb and given with the input variable `nc = 4.0d0`. The resistivity was actually computed during the previous run. A file named `pb.res.01.1.000` should have been created. The file contains the resistivity (in  $\mu\text{Ohm cm}$ ) for various temperatures and smearing values (in meV).

```
# Temperature [K]          Resistivity [micro Ohm cm] for different Phonon smearing (meV)
# 0.0500000 0.1000000 0.1500000 0.2000000 0.2500000 0.3000000 0.3500000 0.4000000 0.4500000 0.5000000
10 0.3512365 0.3525075 0.3547220 0.3580869 0.3631532 0.3708099 0.3812646 0.3936785 0.4068816 0.4199550
20 1.4725265 1.4750212 1.4793713 1.4859905 1.4959803 1.5111188 1.5318215 1.5564122 1.5825522 1.6084046
30 2.8584829 2.8621542 2.8685611 2.8783246 2.8930970 2.9155452 2.9462931 2.9828262 3.0216362 3.0599667
40 4.2577891 4.2626474 4.2711290 4.2840617 4.3036484 4.3334453 4.3742842 4.4228124 4.4743510 4.5252226
50 5.6291937 5.6352487 5.6458209 5.6619449 5.6863747 5.7235552 5.7745264 5.8350971 5.8994175 5.9628891
...
```

You should get the following graph (for 0.15 meV smearing):



Note that in this case, as it is an integrated quantity, it is not so dependent on smearing. Compare your result with other smearing. At convergence you should get <sup>3</sup>:



<sup>3</sup>The figure is from [Comput. Phys. Commun. 209, 116 \(2016\)](#).

► Now compute the conductivity of Pb using the BTE:

$$\sigma_{\alpha\beta} = \frac{-e}{V_{uc}} \sum_n \int \frac{d^3k}{\Omega_{BZ}} v_{nk}^\alpha \partial_{E_\beta} f_{nk} \quad (4)$$

\$ ibrun \$PATHQE/bin/epw.x -npool 8 -input epw2.in > epw2.out

```

--                                                                 epw2.in
&inputepw
  prefix      = 'pb'
  amass(1)    = 207.2
  outdir      = './'
  dvscf_dir   = './save'

  elph        = .true.
  epwwrite    = .false. ! Restarting
  epwread     = .true.  ! Restarting by reading the pb.epmatwp file
  vme         = 'wannier'

  nbndsub     = 4
  bands_skipped = 'exclude_bands = 1-5'

  wannierize  = .false. ! Restarting.
  num_iter    = 300
  dis_win_max = 21
  dis_froz_min= -3
  dis_froz_max= 13.5
  proj(1)     = 'Pb:sp3'
  wdata(1)    = 'bands_plot = .true.'
  wdata(2)    = 'begin kpoint_path'
  wdata(3)    = 'G 0.00 0.00 0.00 X 0.00 0.50 0.50'
  wdata(4)    = 'X 0.00 0.50 0.50 W 0.25 0.50 0.75'
  wdata(5)    = 'W 0.25 0.50 0.75 L 0.50 0.50 0.50'
  wdata(6)    = 'L 0.50 0.50 0.50 K 0.375 0.375 0.75'
  wdata(7)    = 'K 0.375 0.375 0.75 G 0.00 0.00 0.00'
  wdata(8)    = 'G 0.00 0.00 0.00 L 0.50 0.50 0.50'
  wdata(9)    = 'end kpoint_path'
  wdata(10)   = 'bands_plot_format = gnuplot'

  elecselven = .false.
  phonselven = .false.
  a2f         = .false.

  fsthick     = 0.4 ! eV - we only need states close to Fermi level
  degaussw    = 0.0 ! eV (adaptative smearing)
  assume_metal = .true.
  ngaussw     = -99 ! we want F-D distribution for metals

  int_mob     = .true.
  iterative_bte = .true. ! SERTA and iterative BTE
  scattering   = .true. ! compute scattering rates
  carrier      = .false. ! This is a metal, we do not specify carrier concentration
  mp_mesh_k    = .true. ! Use crystal symmetries
  epmatkqread  = .false. ! Can be used to just perform BTE iterations
  mob_maxiter  = 200 ! Max nb of BTE iterations
  broyden_beta = 0.7 ! Broyden mixing during iterations
  restart      = .true. ! Activate possible restart
  restart_step = 50 ! Write restart points every 50 q-points
  selecqread   = .false.
  nstemp       = 9 ! compute conductivity at 9 temperatures
  temps        = 100 500

  nkf1        = 30
  nkf2        = 30
  nkf3        = 30

  nqf1        = 30
  nqf2        = 30
  nqf3        = 30

```

```

nk1      = 3
nk2      = 3
nk3      = 3

nq1      = 3
nq2      = 3
nq3      = 3
/

```

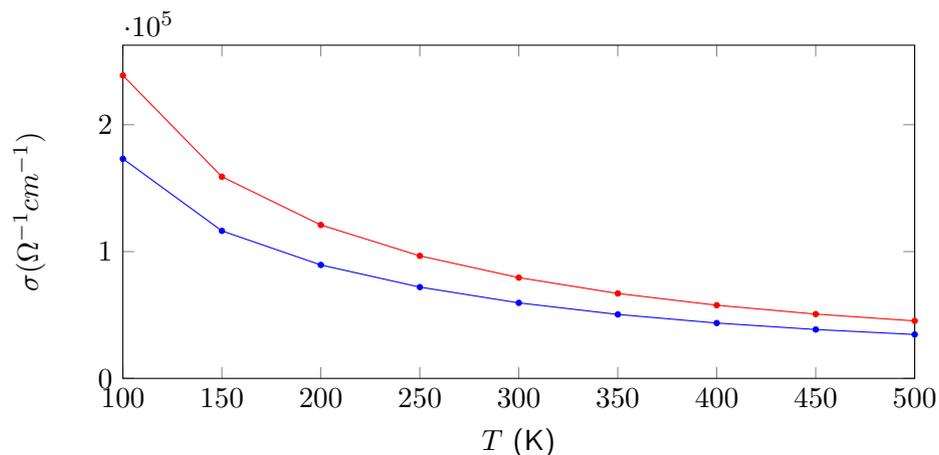
**Note 1:** In this calculation, we are restarting from the electron-phonon matrix elements written in real space in the `pb.epmatwp` file.

**Note 2:** In the case of BTE, `temps` corresponds to the real lattice temperature and `degaussw` is used to approximate the Dirac deltas. In this case it has the value 0.0 eV which means that an adaptative smearing is used. The value of the smearing is therefore band and k-point dependent and depends on the fine grid size: the denser the grids, the smaller the smearing.

**Note 3:** Because of the `restart = .true.` input parameter, if you want to do a clean restart, you need to remove the restart file by doing `rm restart.fmt`.

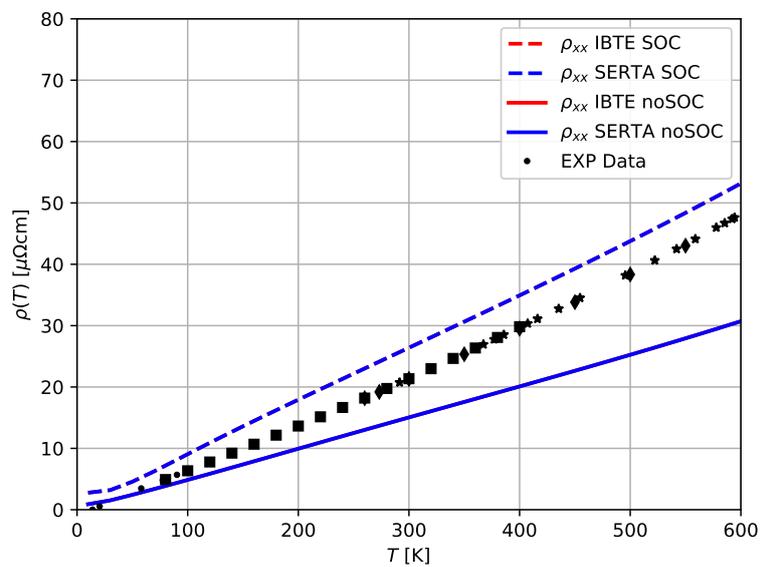
► Check the output file to find the minimum and maximum values of smearing reported. You can try changing the fine grids to see how it affects the smearing.

► Extract the SERTA and BTE conductivity as a function of temperature from the output file. You should get something like this (blue SERTA, red BTE):



**Note:** The resistivity is the inverse of the conductivity, therefore you can also obtain it.

At convergence you should get (BTE and SERTA are almost the same in this case, figure courtesy of Félix Goudreault):



► You can try to include SOC using `noncolin = .true.` and `lspinorb = .true.` in `scf.in` and re-do everything to see the impact. Note that the calculations will be longer and that you need to double the number of Wannier functions and bands.

---

## Exercise 2

In this example we are going to calculate the drift and Hall hole carrier mobility of c-BN. The drift mobility is obtained with:

$$\mu_{\alpha\beta}^d = \frac{-e}{V_{uc}n_c} \sum_n \int \frac{d^3k}{\Omega_{BZ}} v_{n\mathbf{k}}^\alpha \partial_{E_\beta} f_{n\mathbf{k}} \quad (5)$$

where the out of equilibrium occupations are obtained by solving the BTE:

$$\begin{aligned} \partial_{E_\beta} f_{n\mathbf{k}} &= ev_{n\mathbf{k}}^\beta \frac{\partial f_{n\mathbf{k}}^0}{\partial \varepsilon_{n\mathbf{k}}} \tau_{n\mathbf{k}} + \frac{2\pi\tau_{n\mathbf{k}}}{\hbar} \sum_{m\nu} \int \frac{d^3q}{\Omega_{BZ}} |g_{m\nu}(\mathbf{k}, \mathbf{q})|^2 \\ &\times \left[ (n_{\mathbf{q}\nu} + 1 - f_{n\mathbf{k}}^0) \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} + \hbar\omega_{\mathbf{q}\nu}) + (n_{\mathbf{q}\nu} + f_{n\mathbf{k}}^0) \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} - \hbar\omega_{\mathbf{q}\nu}) \right] \partial_{E_\beta} f_{m\mathbf{k}+\mathbf{q}}. \end{aligned} \quad (6)$$

The scattering rate in Eq. (6) is defined as:

$$\begin{aligned} \tau_{n\mathbf{k}}^{-1} &\equiv \frac{2\pi}{\hbar} \sum_{m\nu} \int \frac{d^3q}{\Omega_{BZ}} |g_{m\nu}(\mathbf{k}, \mathbf{q})|^2 [(n_{\mathbf{q}\nu} + 1 - f_{m\mathbf{k}+\mathbf{q}}^0) \\ &\times \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} - \hbar\omega_{\mathbf{q}\nu}) + (n_{\mathbf{q}\nu} + f_{m\mathbf{k}+\mathbf{q}}^0) \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} + \hbar\omega_{\mathbf{q}\nu})]. \end{aligned} \quad (7)$$

The Hall mobility is obtained with:

$$\mu_{\alpha\beta\gamma}^H = \frac{-e}{V_{uc}n_c} \sum_n \int \frac{d^3k}{\Omega_{BZ}} v_{n\mathbf{k}}^\alpha \partial_{E_\beta} f_{n\mathbf{k}}(B_\gamma) \quad (8)$$

again solving the BTE with finite (small) magnetic field:

$$\begin{aligned} \left[ 1 - \frac{e}{\hbar} \tau_{n\mathbf{k}} (\mathbf{v}_{n\mathbf{k}} \times \mathbf{B}) \cdot \nabla_{\mathbf{k}} \right] \partial_{E_\beta} f_{n\mathbf{k}}(B_\gamma) &= ev_{n\mathbf{k}}^\beta \frac{\partial f_{n\mathbf{k}}^0}{\partial \varepsilon_{n\mathbf{k}}} \tau_{n\mathbf{k}} + \frac{2\pi\tau_{n\mathbf{k}}}{\hbar} \sum_{m\nu} \int \frac{d^3q}{\Omega_{BZ}} |g_{m\nu}(\mathbf{k}, \mathbf{q})|^2 \\ &\times \left[ (n_{\mathbf{q}\nu} + 1 - f_{n\mathbf{k}}^0) \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} + \hbar\omega_{\mathbf{q}\nu}) + (n_{\mathbf{q}\nu} + f_{n\mathbf{k}}^0) \delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} - \hbar\omega_{\mathbf{q}\nu}) \right] \partial_{E_\beta} f_{m\mathbf{k}+\mathbf{q}}(B_\gamma). \end{aligned} \quad (9)$$

The Hall factor is obtained as:

$$\mu_{\alpha\beta\gamma}^H = r_{\alpha\beta\gamma}^H \mu_{\alpha\beta}^d \quad (10)$$

$$r_{\alpha\beta\gamma}^H \equiv \sum_{\delta\epsilon} \frac{(\mu_{\alpha\delta}^d)^{-1} \mu_{\delta\epsilon\gamma}^H (\mu_{\epsilon\beta}^d)^{-1}}{B_\gamma}. \quad (11)$$

More information can be found in the review [Rep. Prog. Phys. 83, 036501 \(2020\)](#).

First go to the second exercise:

\$ cd ../exercise2

► Make a self-consistent calculation for c-BN.

```

--
&control
  calculation = 'scf'
  prefix      = 'bn'
  restart_mode = 'from_scratch'
  wf_collect  = .true.
  pseudo_dir  = './'
  outdir      = './'
/
&system
  ibrav       = 2
  celldm(1)   = 6.833
  nat         = 2
  ntyp        = 2
  ecutwfc     = 40
/
&electrons
  diagonalization = 'david'
  mixing_beta     = 0.7
  conv_thr        = 1.0d-13
/
ATOMIC_SPECIES
  B 10.811 B-PBE.upf
  N 14.0067 N-PBE.upf
ATOMIC_POSITIONS {crystal}
  B 0.00 0.00 0.00
  N -0.25 0.75 -0.25
K_POINTS automatic
8 8 8 0 0 0

```

**Note:** In practice the **k**-point grid needs to be fairly large in order to get converged dielectric function and Born effective charges during the following phonon calculation.

```
$ ibrun $PATHQE/bin/pw.x -input scf.in > scf.out
```

► Compute the vibrational properties of c-BN on a coarse 4x4x4 **q**-point grid.

```

--
&inputph
  recover=.false.
  tr2_ph=1.0d-17,
  prefix='bn',
  amass(1)=10.811,
  amass(2)=14.0067,
  outdir='./',
  fildyn='bn.dyn.xml',
  fildvscf='dvscf'
  ldisp=.true.,
  epsil=.true.,
  nq1 = 4,
  nq2 = 4,
  nq3 = 4
/

```

**Note:** With respect to the first exercise, we have added the input variable `epsil=.true.` which computes the macroscopic dielectric constant of the system in non-metallic systems.

```
$ ibrun $PATHQE/bin/ph.x -input ph.in > ph.out
```

The calculation should take about 8 min on 8 cores. During the run, notice the IBZ **q**-point grid:

```

Dynamical matrices for ( 4, 4, 4) uniform grid of q-points
( 8 q-points):

```

---

```

N          xq(1)          xq(2)          xq(3)
1  0.000000000  0.000000000  0.000000000
2 -0.250000000  0.250000000 -0.250000000
3  0.500000000 -0.500000000  0.500000000
4  0.000000000  0.500000000  0.000000000
5  0.750000000 -0.250000000  0.750000000
6  0.500000000  0.000000000  0.500000000
7  0.000000000 -1.000000000  0.000000000
8 -0.500000000 -1.000000000  0.000000000

```

as well as the dielectric function and Born effective charge tensor:

Dielectric constant in cartesian axis

```

( 4.597197252 -0.000000000 0.000000000 )
( -0.000000000 4.597197252 0.000000000 )
( -0.000000000 0.000000000 4.597197252 )

```

Effective charges (d Force / dE) in cartesian axis

```

atom 1 B
Ex ( 1.89177 0.00000 0.00000 )
Ey ( 0.00000 1.89177 -0.00000 )
Ez ( 0.00000 -0.00000 1.89177 )
atom 2 N
Ex ( -1.89376 0.00000 0.00000 )
Ey ( 0.00000 -1.89376 -0.00000 )
Ez ( 0.00000 -0.00000 -1.89376 )

```

The experimental dielectric constant in c-BN is about 4.46. More accurate values can be obtained with larger  $k$ -point grids. c-BN is a polar material and here has a Born effective charge of 1.89 which is very close to theoretical value of 1.91.

► As for the first exercise, run the python post-processing to create the save folder

```
$ python $PATHQE/EPW/bin/pp.py
```

► Do a non self-consistent calculation on a 4x4x4 uniform and  $\Gamma$ -centered  $k$ -point grid with crystal coordinates in the interval  $[0,1[$

Such a grid can be for example generated with:

```
$ $PATHQE/wannier90-3.1.0/utility/kmesh.pl 4 4 4
```

The nscf.in file is as follow:

```

--                                                                 nscf.in
&control
  calculation = 'nscf'
  prefix      = 'bn'
  restart_mode = 'from_scratch'
  wf_collect  = .true.
  pseudo_dir  = './'
  outdir      = './'
/
&system
  ibrav      = 2
  celldm(1) = 6.833
  nat       = 2
  ntyp      = 2
  ecutwfc   = 40
  nbnd      = 20

```

```

/
&electrons
  diagonalization = 'david'
  mixing_beta     = 0.7
  conv_thr        = 1.0d-13
/
ATOMIC_SPECIES
  B 10.811 B-PBE.upf
  N 14.0067 N-PBE.upf
ATOMIC_POSITIONS {crystal}
  B 0.00 0.00 0.00
  N -0.25 0.75 -0.25
K_POINTS crystal
64
  0.00000000 0.00000000 0.00000000 1.562500e-02
  0.00000000 0.00000000 0.25000000 1.562500e-02
  0.00000000 0.00000000 0.50000000 1.562500e-02
...

```

```
$ ibrun $PATHQE/bin/pw.x -input nscf.in > nscf.out
```

► Perform an EPW calculation to Fourier-transform the electron-phonon matrix element from a coarse  $4 \times 4 \times 4$   $k$  and  $q$ -point grids to real space and then interpolate the electronic band structure and phononic dispersion along the  $L - \Gamma - X - K - \Gamma$  high symmetry line by reading the file `LGXKG.txt`.

```

--
&inputepw
  prefix      = 'bn'
  amass(1)    = 10.811,
  amass(2)    = 14.0067,
  outdir      = './'

  elph        = .true.
  epbwrite    = .true.
  epbread     = .false.
  epwwrite    = .true.
  epwread     = .false.
  etf_mem     = 1
  lpolar      = .true.  ! polar material
  vme         = 'wannier'
  use_ws      = .false. ! Gamma centred Wigner-Seitz cell

  nbndsub     = 3
  bands_skipped = 'exclude_bands = 1, 5-20'

  wannierize  = .true.
  num_iter    = 50000
  iprint      = 2
  dis_win_max = 12.0
  dis_win_min = -1.0

  proj(1)     = 'N:p'

  wdata(1) = 'bands_plot = .true.'
  wdata(2) = 'begin kpoint_path'
  wdata(3) = ' G 0.000 0.000 0.000 X 0.500 0.000 0.500 '
  wdata(4) = ' X 0.500 0.000 0.500 U 0.625 0.250 0.625 '
  wdata(5) = ' K 0.375 0.375 0.750 G 0.000 0.000 0.000 '
  wdata(6) = ' G 0.000 0.000 0.000 L 0.500 0.500 0.500 '
  wdata(7) = ' L 0.500 0.500 0.500 W 0.500 0.250 0.750 '
  wdata(8) = ' W 0.500 0.250 0.750 X 0.500 0.000 0.500 '
  wdata(9) = 'end kpoint_path'
  wdata(10) = 'bands_plot_format = gnuplot'
  wdata(11) = 'guiding_centres = .true.'
  wdata(12) = 'dis_num_iter      = 5000'
  wdata(13) = 'num_print_cycles  = 10'
  wdata(12) = 'dis_mix_ratio     = 1.0'

```

```

wdata(13) = 'conv_tol = 1E-12'
wdata(14) = 'conv_window = 4'
wdata(15) = 'use_ws_distance = T'

elecselfen = .false.
phonsselfen = .false.
a2f        = .false.

fsthick    = 100
nstemp     = 1
temps     = 1
degaussw  = 0.001

dvscf_dir = './save'

band_plot = .true.

filkf     = './LGXKG.txt'
filqf     = './LGXKG.txt'

nk1       = 4
nk2       = 4
nk3       = 4
nq1       = 4
nq2       = 4
nq3       = 4
/

```

```
$ ibrun $PATHQE/bin/epw.x -npool 8 -input epw1.in > epw1.out
```

The calculation should take less than 1 min. Note that the code should have detected the presence of the `quadrupole.fmt` file and correctly read the quadrupole tensor. Look in the output for the line `Quadrupole tensor is correctly read:.` In this hands-on we will not cover how to obtain the quadrupole tensor and they are simply given here. There are two ways to obtain them:

- Using perturbation theory. This is implemented in a recent version of the [Abinit](#) software.
- Fitting the perturbed density or the electron-phonon matrix elements in the long wavelength limit obtained by direct DFPT calculations.

At the end of the calculation, because of the keyword `band_plot = .true.`, the code should produce the `band.eig` and `phband.freq` files that contain the electronic band structure and phononic dispersion along a path given in the `filkf` and `filqf` files.

If you want to have files in an easy gnuplot format, you can use the `plotband.x` tool by doing

```
$ $PATHQE/bin/plotbands.x
```

and follow the instructions. You should check that both plots look reasonable.

► Do a restart calculation (restarting from the `bn.epmatwp1` file) and compute the hole mobility of c-BN.

```
$ ibrun $PATHQE/bin/epw.x -npool 8 -input epw2.in > epw2.out
```

The input file is as follow:

```

--
&inputepw
prefix      = 'bn'
amass(1)    = 10.811,
amass(2)    = 14.0067,
--

```

```

outdir      = './'

elph        = .true.
epwwrite    = .false.
epwread     = .true.
etf_mem     = 3      ! generate k-points within fsthick
lpolar      = .true.
vme         = 'wannier'
mp_mesh_k   = .true.
use_ws      = .false.

nbndsub     = 3
bands_skipped = 'exclude_bands = 1, 5-20'

scattering  = .true.
scattering_serta = .true.
int_mob     = .false.
carrier     = .true.
ncarrier    = -1E13
iterative_bte = .true.
epmatkqread = .false.
mob_maxiter = 300
broyden_beta = 1.0
bfieldx     = 0.0d0
bfieldy     = 0.0d0
bfieldz     = 1.0d-10 ! Apply a magnetic field along Cart. z

nstep       = 1
temps       = 300

restart     = .true.
restart_step = 1000
selecqread = .false.

wannierize  = .false.
num_iter    = 50000
iprint      = 2
dis_win_max = 12.0
dis_win_min = -1.0

proj(1)     = 'N:p'

elecselfen  = .false.
phonselfen  = .false.
a2f         = .false.

fsthick     = 0.4 ! 0.3 eV
degaussw    = 0.0

efermi_read = .true
fermi_energy = 11.246840

dvscf_dir   = './save'

nkf1        = 60
nkf2        = 60
nkf3        = 60
nqf1        = 60
nqf2        = 60
nqf3        = 60

nk1         = 4
nk2         = 4
nk3         = 4
nq1         = 4
nq2         = 4
nq3         = 4

```

```

! Note: for reference, if you use ecutwfc = 100, nk1=nk2=nk3 = 10, nq1=nq2=nq3 = 5
!       and nkf1=nkf2=nkf3=nqf1=nqf2=nqf3 = 60 and mob_maxiter = 300, you should get:
!       SERTA mobility = 0.26719E+03 and SERTA Hall factor = 0.486742

```

```

!      BTE mobility   = 0.33361E+03 and BTE Hall factor   = 0.602855
!      and with SOC and the same parameters, you should get:
!      SERTA mobility = 0.25971E+03 and SERTA Hall factor = 0.414480
!      BTE mobility   = 0.32871E+03 and BTE Hall factor   = 0.553747
/

```

**Notes:**

- `epwread` allows for the restart from the `bn.epmatwp1` file
- `int_mob` allows to perform both electron and hole calculations at the same time but is not recommended.
- `carrier` and `ncarrier` define the carrier concentration. If `carrier = .true.` then the intrinsic mobility with `ncarrier` concentration (in  $\text{cm}^{-3}$ ) is computed. If `ncarrier` is positive it will compute the electron mobility and if it is negative it will compute the hole mobility. The resulting mobility should be independent of the choice of carrier concentration in reasonable ranges  $10^{10}$ -  $10^{16}$   $\text{cm}^{-3}$ .
- `iterative_bte` asks for the iterative solution of the BTE in addition to SERTA.
- `nstemp` and `temps` define the lattice temperature at which the mobility is evaluated.
- `restart` and `restart_step` will create restart point every (in this case) 1000 q-points. You can try breaking the run after a restart point and restart to test this feature.
- `bfieldz` adds a (small) finite magnetic field along the Cartesian z direction. This will automatically trigger the calculation of the Hall factor.
- `mob_maxiter` is the maximum number of iterations for the BTE solution.
- `degaussw = 0.0` means that adaptive smearing is used. Positive values give Gaussian smearing.

```
$ ibrun PATH/q-e/bin/epw.x -npool 8 < epw2.in > epw2.out
```

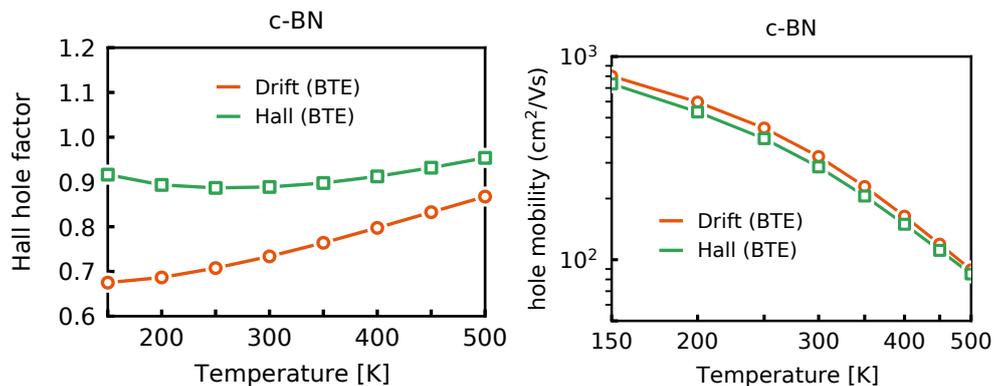
The run should take about 4 min. The fine **k** and **q** point grids need to be much denser for real calculations. However, we can already get relatively decent results.

► Re-run the code with multiple temperatures. You should remove the `restart.fmt` file before doing so.

Try filling the table below for the hole mobility:

T (K)	hole $\varepsilon_F$ (eV)	drift SERTA $\mu$ ( $\text{cm}^2/\text{Vs}$ )	drift BTE $\mu$ ( $\text{cm}^2/\text{Vs}$ )	Hall BTE $\mu$ ( $\text{cm}^2/\text{Vs}$ )
100				
200				
300				
400				
500				

At convergence you should get <sup>4</sup>:



<sup>4</sup>The figure is from [arXiv:2105.04192](https://arxiv.org/abs/2105.04192)

---

where the room temperature values with SOC should be around  $319 \text{ cm}^2/\text{Vs}$  for the drift BTE and  $281 \text{ cm}^2/\text{Vs}$  for the Hall mobility with a Hall factor of 0.88.

- ▶ Try to increase the fine grids and add a few more temperatures and see if you can get a result closer to convergence.
- ▶ Try adding SOC
- ▶ Try removing or renaming the file `quadrupole.fmt` to do the interpolation with dipole only and see the impact on the results.