Resistivity, drift and Hall mobility with EPW

Tutorial for 2022 School on Electron-Phonon Physics from First Principles on June 15th, 2022

Hands-on session

Hands-on based on `Quantum ESPRESSO` (v7.1) and `EPW` v5.5

Introduction

In this tutorial, we will show how to compute the intrinsic electron and hole low-field drift and Hall mobility of the polar cubic semiconductor BN using the linearised iterative Boltzmann transport equation (IBTE) and the self-energy relaxation time approximation (SERTA), with or without external magnetic field. We will also see how to compute the electric resistivity of metals.
You are advised to prepare the following script file, e.g. `run.sh`:

```
--                                                                                    run.sh
#!/bin/bash
#SBATCH -J job.ph # 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 EPSchool2022
#SBATCH -p small
#SBATCH --reservation=EPSchoolDay3

# Launch MPI code...
export PATHQE=/work2/06868/giustino/EP-SCHOOL/q-e

ibrun $PATHQE/bin/pw.x < scf.in | tee scf.out
ibrun $PATHQE/bin/ph.x < ph.in | tee ph.out
```

For the description for all input flags please follow the link:

`https://epwdoc.gitlab.io/source/doc/Inputs.html`

Exercise 1

1.1 Theory

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

$$\mu^{\mathrm{d}}_{\alpha\beta} = \frac{-1}{V^{\mathrm{uc}} n^{\mathrm{c}}} \sum_n \int \frac{\mathrm{d}^3 k}{\Omega^{\mathrm{BZ}}} v_{n\mathbf{k}\alpha} \partial_{E_\beta} f_{n\mathbf{k}} \tag{1}$$

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

$$\begin{aligned} \partial_{E_\beta} f_{n\mathbf{k}} = {} & e v_{n\mathbf{k}\beta} \frac{\partial f^0_{n\mathbf{k}}}{\partial \varepsilon_{n\mathbf{k}}} \tau_{n\mathbf{k}} + \frac{2\pi\tau_{n\mathbf{k}}}{\hbar} \sum_{m\nu} \int \frac{\mathrm{d}^3 q}{\Omega_{\mathrm{BZ}}} |g_{mn\nu}(\mathbf{k},\mathbf{q})|^2 \\ \times & \Big[(n_{\mathbf{q}\nu} + 1 - f^0_{n\mathbf{k}})\delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} + \hbar\omega_{\mathbf{q}\nu}) + (n_{\mathbf{q}\nu} + f^0_{n\mathbf{k}})\delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} - \hbar\omega_{\mathbf{q}\nu})\Big] \partial_{E_\beta} f_{m\mathbf{k}+\mathbf{q}}. \end{aligned} \tag{2}$$

The scattering rate in Eq. (2) is defined as:

$$\tau_{n\mathbf{k}}^{-1} \equiv \frac{2\pi}{\hbar}\sum_{m\nu}\int \frac{d^3q}{\Omega_{\rm BZ}} |g_{mn\nu}(\mathbf{k},\mathbf{q})|^2 \big[(n_{\mathbf{q}\nu}+1-f^0_{m\mathbf{k}+\mathbf{q}}) \\ \times \delta(\varepsilon_{n\mathbf{k}}-\varepsilon_{m\mathbf{k}+\mathbf{q}}-\hbar\omega_{\mathbf{q}\nu}) + (n_{\mathbf{q}\nu}+f^0_{m\mathbf{k}+\mathbf{q}})\delta(\varepsilon_{n\mathbf{k}}-\varepsilon_{m\mathbf{k}+\mathbf{q}}+\hbar\omega_{\mathbf{q}\nu})\big]. \quad (3)$$

A common approximation to Eq. (2) is called the self-energy relaxation time approximation (SERTA) and consists in neglecting the second term in the right-hand of the equation which gives:

$$\mu^{\rm SERTA}_{\alpha\beta} = \frac{-e}{V^{\rm uc}n^{\rm c}} \sum_n \int \frac{{\rm d}^3k}{\Omega^{\rm BZ}} \frac{\partial f^0_{n\mathbf{k}}}{\partial \varepsilon_{n\mathbf{k}}} v_{n\mathbf{k}\alpha} v_{n\mathbf{k}\beta}\tau_{n\mathbf{k}}. \quad (4)$$

The the low-field phonon-limited carrier mobility in the presence of a small finite magnetic field **B** is given by:

$$\mu_{\alpha\beta}(B_\gamma) = \frac{-1}{V^{\rm uc}n^{\rm c}} \sum_n \int \frac{{\rm d}^3k}{\Omega^{\rm BZ}} v_{n\mathbf{k}\alpha}[\partial_{E_\beta} f_{n\mathbf{k}}(B_\gamma) - \partial_{E_\beta} f_{n\mathbf{k}}], \quad (5)$$

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

$$\Big[1 - \frac{e}{\hbar}\tau_{n\mathbf{k}}(\mathbf{v}_{n\mathbf{k}} \times \mathbf{B})\cdot\nabla_{\mathbf{k}}\Big]\partial_{E_\beta} f_{n\mathbf{k}}(B_\gamma) = e v_{n\mathbf{k}\beta}\frac{\partial f^0_{n\mathbf{k}}}{\partial\varepsilon_{n\mathbf{k}}}\tau_{n\mathbf{k}} + \frac{2\pi\tau_{n\mathbf{k}}}{\hbar}\sum_{m\nu}\int\frac{{\rm d}^3q}{\Omega^{\rm BZ}}|g_{mn\nu}(\mathbf{k},\mathbf{q})|^2 \\ \times \Big[(n_{\mathbf{q}\nu}+1-f^0_{n\mathbf{k}})\delta(\varepsilon_{n\mathbf{k}}-\varepsilon_{m\mathbf{k}+\mathbf{q}}+\hbar\omega_{\mathbf{q}\nu}) + (n_{\mathbf{q}\nu}+f^0_{n\mathbf{k}})\delta(\varepsilon_{n\mathbf{k}}-\varepsilon_{m\mathbf{k}+\mathbf{q}}-\hbar\omega_{\mathbf{q}\nu})\Big]\partial_{E_\beta} f_{m\mathbf{k}+\mathbf{q}}(B_\gamma). \quad (6)$$

The Hall factor and Hall mobility are then obtained as:

$$r_{\alpha\beta}(\hat{\mathbf{B}}) \equiv \lim_{\mathbf{B}\to 0}\sum_{\delta\epsilon}\frac{[\mu^{\rm d}_{\alpha\delta}]^{-1}\,\mu_{\delta\epsilon}(\mathbf{B})\,[\mu^{\rm d}_{\epsilon\beta}]^{-1}}{|\mathbf{B}|} \quad (7)$$

$$\mu^{\rm Hall}_{\alpha\beta}(\hat{\mathbf{B}}) = \sum_\gamma \mu^{\rm d}_{\alpha\gamma} r_{\gamma\beta}(\hat{\mathbf{B}}), \quad (8)$$

where $\hat{\mathbf{B}}$ is the direction of the magnetic field. More information can be found in the review Rep. Prog. Phys. **83**, 036501 (2020).

1.2 Preliminary calculations with Quantum Espresso

First download the exercise files:

```
$ cp XXXX/XXX.tar .
$ tar -xvf XXX.tar
$ cd exercise1/
```

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

```
--                                                                    scf.in
 &control
    calculation     = 'scf'
    prefix          = 'bn'
    restart_mode    = 'from_scratch'
    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 < scf.in | tee scf.out
```

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

```
--                                                                    ph.in
&inputph
  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: We have the input variable `epsil=.true.` which computes the macroscopic dielectric constant in non-metallic systems. If you add `.xml` after the name of the dynamical matrix file, it will produce the data in XML format (preferred).
Note 2: The input variable responsible to produce the electron-phonon matrix element is `fildvscf`. Always make sure that this variable is present.

Note 3: Notice the very tight `tr2_ph` threshold parameter on the self-consistent first-order perturbed wavefunction. This is crucial to obtain good vibrational properties.

```
$ ibrun  $PATHQE/bin/ph.x < ph.in | tee ph.out
```

The calculation should take about 5 min on 4 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 with asr applied:
           atom      1   B  Mean Z*:        1.89277
      E*x (        1.89277       -0.00000       -0.00000 )
      E*y (       -0.00000        1.89277       -0.00000 )
      E*z (        0.00000       -0.00000        1.89277 )
           atom      2   N  Mean Z*:       -1.89277
      E*x (       -1.89277        0.00000        0.00000 )
      E*y (        0.00000       -1.89277        0.00000 )
      E*z (       -0.00000        0.00000       -1.89277 )
```

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

Finally, we need to post-process some of the data to make it ready for EPW. To do so, we can use a python script (usually provided in `QE/EPW/bin/pp.py` but copied here for convenience).

▶ Run the python post-processing to create the `save` folder

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

The script will ask you to enter the prefix used for the calculation. In this case enter "bn". The script will create a new folder called "save" that contains the dvscf potential files, pattern files, and dynamical matrices on the IBZ.

1.3 Interpolation of the electron-phonon matrix element in real-space with EPW

▶ Do a non self-consistent calculation on a 4x4x4 uniform and Γ-centered $\mathbf{k}$-point grid with crystal coordinates in the interval [0,1[
Such a grid can be for example generated with the `wannier90` utility with `kmesh.pl 4 4 4`.
The `nscf.in` file is as follow:

```
--                                                                        nscf.in
 &control
    calculation     = 'nscf'
    prefix          = 'bn'
    restart_mode    = 'from_scratch'
    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 < nscf.in | tee nscf.out
```

The reason for the non-self consistent calculation is that EPW needs the wavefunctions on the full BZ on a grid between 0 and 1.

Note: Since we are also interested in electron mobility, we will need the conduction bands. Notice that we added the input `nbnd = 20` in `nscf.in`

▶ Perform an `EPW` calculation to Fourier-transform the electron-phonon matrix element from a coarse 4x4x4 $\mathbf{k}$ and $\mathbf{q}$-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`.

```
--                                                                    epw1.in
&inputepw
  prefix      = 'bn'
  outdir      = './'

  elph        = .true.
  epbwrite    = .true.
  epbread     = .false.
  epwwrite    = .true.
  epwread     = .false.
  etf_mem     =  1
  lpolar      = .true.   ! polar material
  vme         = 'dipole'


  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) = ' L  0.500  0.500  0.500   G  0.000  0.000  0.000 '
  wdata(4) = ' G  0.000  0.000  0.000   X  0.500  0.000  0.500 '
  wdata(5) = ' X  0.500  0.000  0.500   K  0.375  0.375  0.750 '
  wdata(6) = ' K  0.375  0.375  0.750   G  0.000  0.000  0.000 '
  wdata(7) = 'end kpoint_path'
  wdata(8) = 'bands_plot_format = gnuplot'
```

```
  wdata(9) = 'guiding_centres = .true.'
  wdata(10) = 'dis_num_iter      = 5000'
  wdata(11) = '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'

  fsthick     = 100
  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 | tee epw1.out
```

Note: The number of pool `-npool` has to be the same as the total number of core `-np` since **k**-point parallelization is (almost) the only parallelization level allowed. **G**-vector parallelization will be introduced in EPW v6.0.

The calculation should take less than 2 min. Note that the code should have detected the presence of the `quadrupole.fmt` file and correclty 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.

More information can be found in Phys. Rev. Research **3**, 043022 (2021)

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/plotband.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 | tee epw2.out
```

The input file is as follow:

```
--                                                                                                 epw2.in
  &inputepw
  prefix      = 'bn'
  outdir      = './'

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

  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

  nstemp      = 1
  temps       = 300

  restart      = .true.
  restart_step = 1000

  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        = 30
  nkf2        = 30
  nkf3        = 30
  nqf1        = 30
  nqf2        = 30
  nqf3        = 30

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

Notes:

- The value of `fermi_energy` was obtained from the output of the previous calculation `epw1.in`
- `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 recommanded.
- `carrier` and `ncarrier` define the carrier concentration. If `carrier = .true.` then the intrinsic mobility with `ncarrier` concentration (in 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} 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 $\mathbf{q}$-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 (in unit of Tesla). 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.

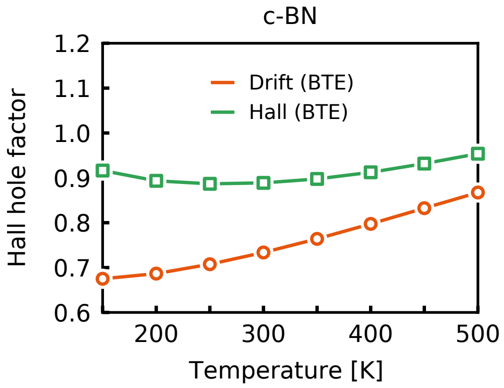
The run should take about 4 min. The fine $\mathbf{k}$ and $\mathbf{q}$ point grids need to be much denser for real calculations. However, we can already get relatively decent results.

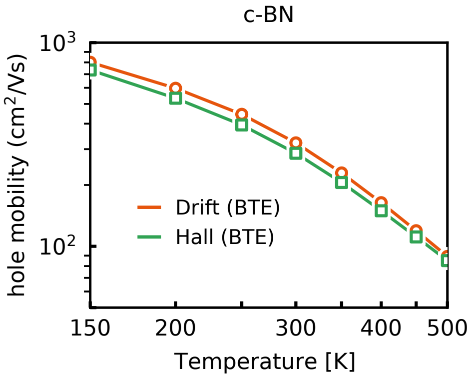
▶ Re-run the code with multiple temperatures (using `nstemp = 4` and `temps = 100, 200, 400, 500`). You should remove the `restart.fmt` file before doing so.

Try filling the table below for the hole mobility:

T (K)	hole ε_F (eV)	drift SERTA μ (cm^2/Vs)	drift BTE μ (cm^2/Vs)	Hall BTE μ (cm^2/Vs)
100				
200				
300				
400				
500				

At convergence you should get [1]:





where the room temperature values with SOC should be around 319 cm^2/Vs for the drift BTE and 281 cm^2/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.

---

[1] The figure is from Phys. Rev. Research **3**, 043022 (2021)

- 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.

Exercise 2

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:

```
$ cd exercise2
```

▶ Make a self-consistent calculation for Pb and a phonon calculation on a homogeneous 3x3x3 $\mathbf{q}$-point grid.
Note: The `ecutwfc` need to be much larger for real calculations.

```
$ ibrun  $PATHQE/bin/pw.x < scf.in | tee scf.out
$ ibrun  $PATHQE/bin/ph.x < ph.in | tee ph.out
```

```
--                                                                scf.in
 &control
    calculation     = 'scf'
    prefix          = 'pb'
    restart_mode    = 'from_scratch'
    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 $\partial_{\mathbf{q}\nu} V^{\mathrm{scf}}$. 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`[2] files into a new save directory. This can easily be done using the `pp.py` python script.

```
$ 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 3x3x3 **uniform and Γ-centered grid between [0,1[in crystal coordinates**.

```
--                                                                      nscf.in
 &control
    calculation     = 'nscf'
    prefix          = 'pb'
    restart_mode    = 'from_scratch'
    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
```

---

[2] The `patterns` file contains the basis in which the `.dvscf` are defined.

```
K_POINTS crystal
27
0.0000000000000000   0.0000000000000000   0.0000000000000000   0.037037037037
0.0000000000000000   0.0000000000000000   0.3333333333333333   0.037037037037
...
```

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

► Perform an EPW calculation to interpolate the electron-phonon matrix element from a coarse 3x3x3 to a dense 18x18x18 **k**-point and **q**-point grids.

```
--                                                                                          epw1.in
&inputepw
  prefix      = 'pb'
  outdir      = './'
  dvscf_dir   = './save'

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

  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'

  elecselfen   = .false.
  phonselfen   = .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:

$$\alpha^2_{\text{tr}}F(\omega) = \frac{1}{2}\sum_{\nu}\int_{\text{BZ}}\frac{d\mathbf{q}}{\Omega_{\text{BZ}}}\omega_{\mathbf{q}\nu}\lambda_{\text{tr},\mathbf{q}\nu}\delta(\omega-\omega_{\mathbf{q}\nu}), \tag{9}$$

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

$$\lambda_{\text{tr},\mathbf{q}\nu} = \frac{1}{N(\varepsilon_F)\omega_{\mathbf{q}\nu}}\sum_{nm}\int_{\text{BZ}}\frac{d\mathbf{k}}{\Omega_{\text{BZ}}}|g_{mn,\nu}(\mathbf{k},\mathbf{q})|^2\delta(\varepsilon_{n\mathbf{k}}-\varepsilon_{\text{F}})\delta(\varepsilon_{m\mathbf{k}+\mathbf{q}}-\varepsilon_{\text{F}})\Big(1-\frac{v_{n\mathbf{k}}\cdot v_{m\mathbf{k}+\mathbf{q}}}{|v_{n\mathbf{k}}|^2}\Big). \tag{10}$$

The calculation of the spectral function in `EPW` is given by the two keywords `phonselfen = .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 δ 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 | tee epw1.out
```

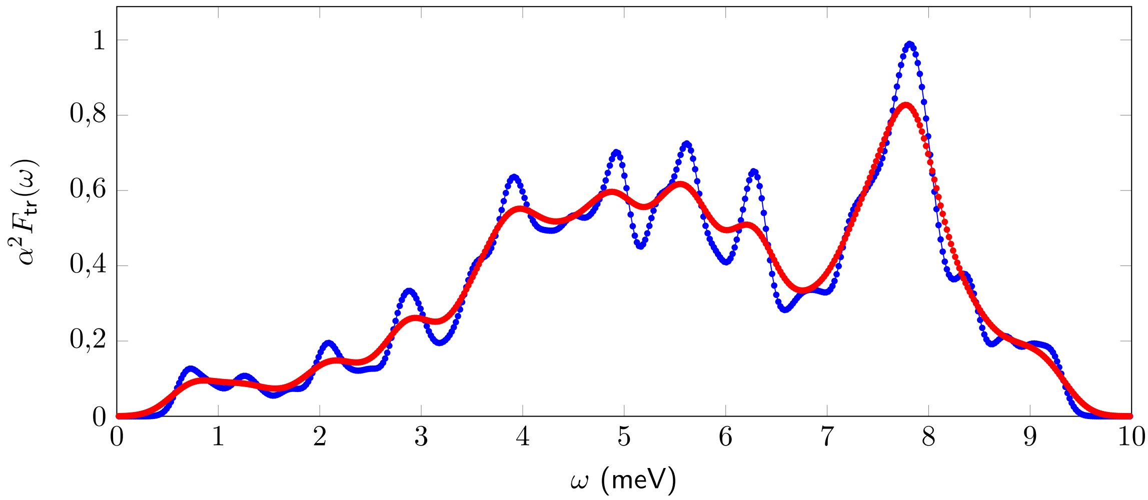
The calculation should take about 6 min to be completed. While the calculation is running, notice in the `epw1.out` the different steps a full `EPW` run goes into. First the Wannierization, then the unfolding into the full 3x3x3 BZ, then the Fourier transform to real space and then finally the interpolation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

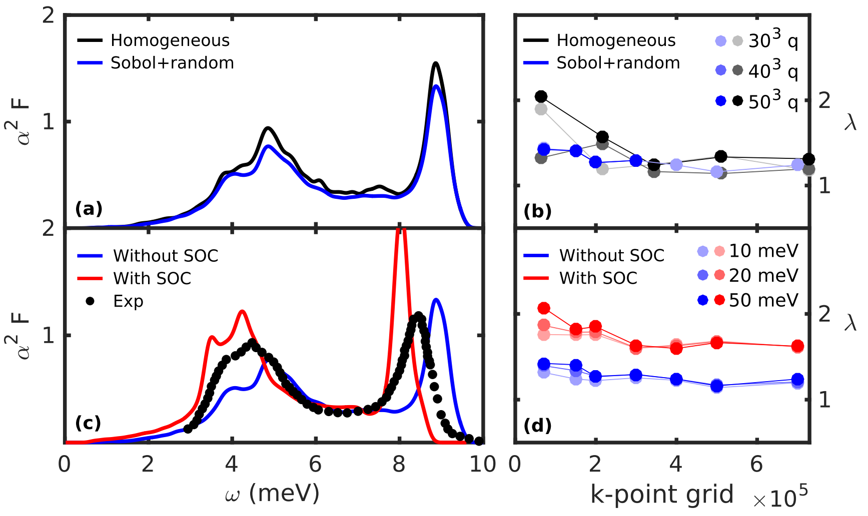
Note that the converged value for λ and $\lambda_{\rm tr}$ should be around 1.1. In addition the files `pb.a2f.01.1.000` and `pb.a2f_tr.01.1.000` which contain the Eliashberg spectral function and transport spectral function for different broadening values should have been produced.

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 [3]:



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

$$\rho(T) = \frac{4\pi m_e}{ne^2 k_B T} \int_0^\infty d\omega\, \hbar\omega\, \alpha^2_{\rm tr} F(\omega)\, n(\omega,T)\big[1+n(\omega,T)\big], \tag{11}$$

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`.

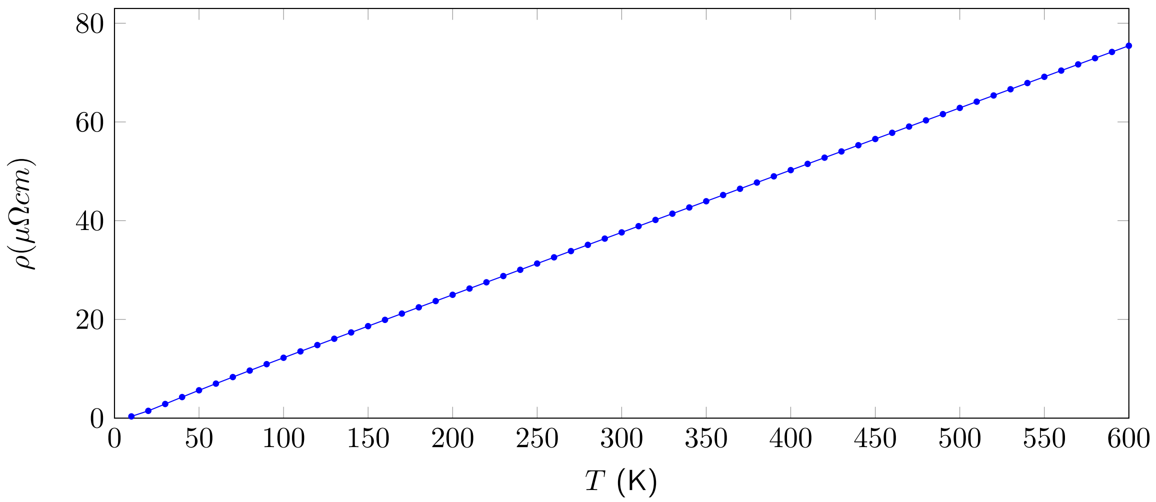
---

[3] The figure is from Comput. Phys. Commun. **209**, 116 (2016).

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 μOhm cm) for various temperatures and smearing values (in meV).

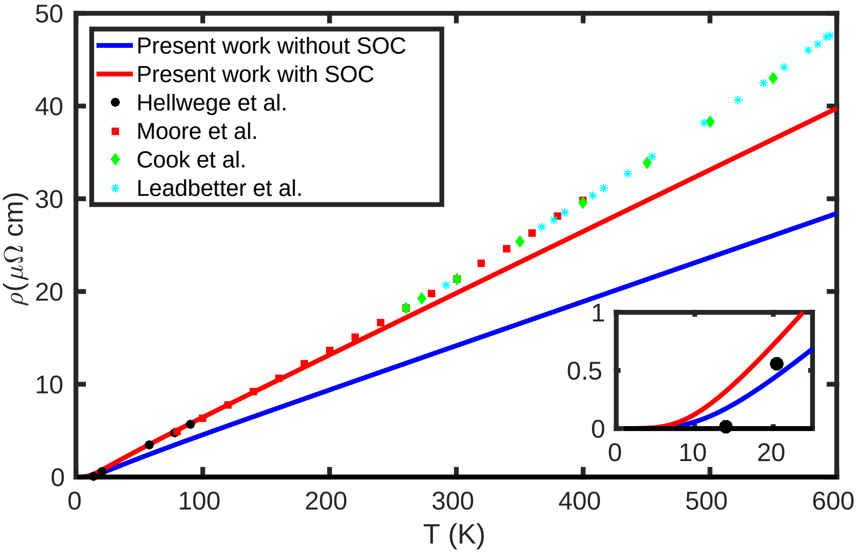
```
# Temperature [K]                Resistivity [micro Ohm cm] for different Phonon smearing (meV)
#           0.0500000    0.1000000    0.1500000    0.2000000    0.2500000    0.3000000    0.3500000    0.4000000    0.4500000    0.5000000
       10    0.3512365    0.3525075    0.3547220    0.3580869    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 [4]:



▶ Now compute the conductivity of Pb using the BTE:

$$\sigma_{\alpha\beta} = \frac{-e}{V_{\text{uc}}} \sum_n \int \frac{\mathrm{d}^3 k}{\Omega_{\text{BZ}}} v^{\alpha}_{n\mathbf{k}} \partial_{E_\beta} f_{n\mathbf{k}} \tag{12}$$

---

[4] The figure is from Comput. Phys. Commun. **209**, 116 (2016).

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

```
--                                                                                                             epw2.in
&inputepw
  prefix      = 'pb'
  outdir      = './'
  dvscf_dir   = './save'

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

  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'

  elecselfen  = .false.
  phonselfen  = .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
```

```
/
```

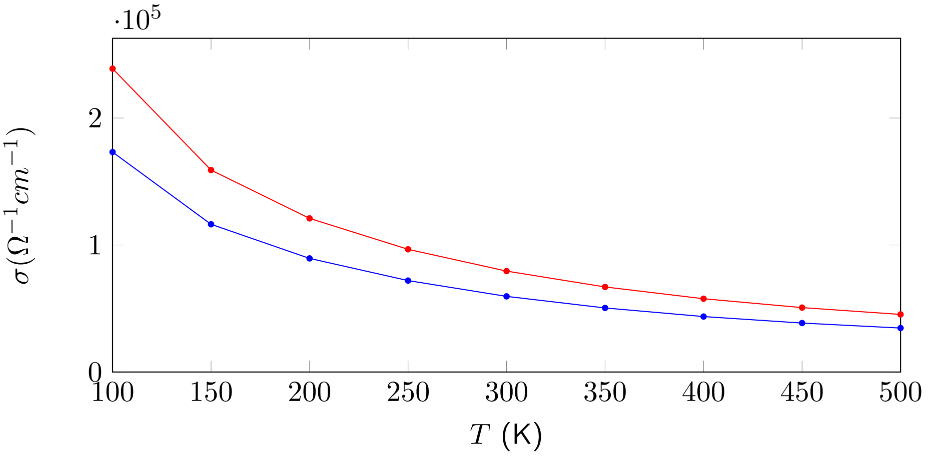
Note 1: In this calculation, we are restarting from the electron-phonon matrix elements written in real space in the `pb.epmatwp` file.

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`.

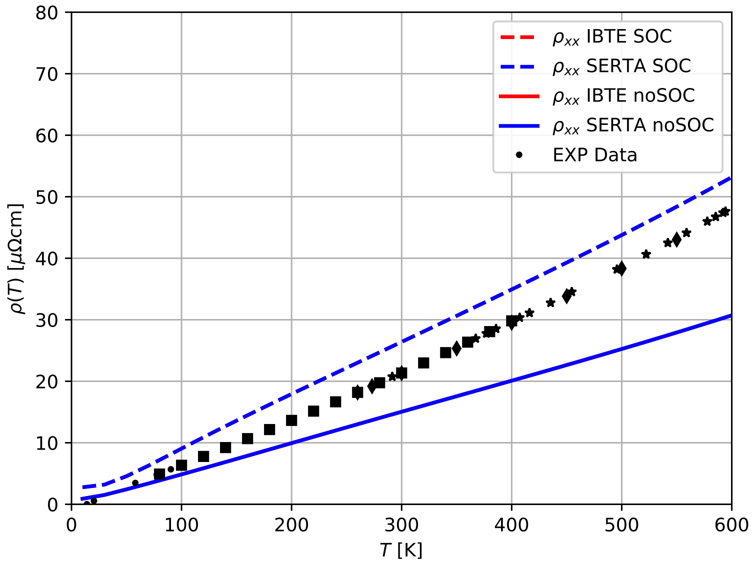
▶ Check the output file to find the minimum and maximum values of smearing reported. You can try changing the fine grids to see how it affects the smearing.

▶ 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.