<div style="border:1px solid black; text-align:center">

Resistivity and mobility with EPW

Tutorial Wed.5

Hands-on session

</div>

Exercise 1

In this example we are going to calculate the electric resistivity of fcc Pb using the Ziman formula. First copy the tutorial files and go in the first exercise:

```
$ wget http://epw.org.uk/uploads/School2018/Wed.5.Ponce.tar
$ tar -xvf Wed.5.Ponce.tar; cd tuto_Wed5/exercise1
```

▶ Make a self-consistent calculation for Pb. For this we are going to use the same input file as for the Tues.5 hands-on.

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

```
$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/pw.x < scf.in > scf.out
```

▶ Make a phonon calculation on an homogeneous 3x3x3 **q**-point grid or copy the `save` folder from the previous tutorial.

```
--                                                                        ph.in
&inputph
  recover  = .false.
  prefix   = 'lead',
  fildyn   = 'pb.dyn',
  fildvscf = 'dvscf'
  ldisp    = .true.,
  nq1=3,
  nq2=3,
  nq3=3,
  tr2_ph   = 1.0d-12,
 /
```

The important keyword is `fildvscf` as it will tell the code to write to file the change of potential due to ionic displacement $\partial_{\mathbf{q}\nu} V^{\text{scf}}$.

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

```
$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/ph.x < ph.in > ph.out
```

This should take about 5 min to be completed. In the output file, locate the list of 4 irreducible **q** point 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 of them, a `lead.dynX` files containing the dynamical matrix has been produced. The `lead.dvscf` files are located inside the `_ph0` folder.

▶ Gather the `.dyn`, `.dvscf` and `patterns` [1] files into a new `save` directory. This can easily be done using the `pp.py` python script.

`$ python /home/nfs3/smr3191/q-e/EPW/bin/pp.py`

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

▶ Do a non self-consistent calculation on a homogeneous 6x6x6 **uniform and Γ-centered grid between [0,1] in crystal coordinates**.

```
&control                                                                    nscf.in
   calculation  ='nscf',
   prefix       ='lead',
   pseudo_dir   = './',
   outdir       = './',
/
&system
   ibrav        = 2,
   celldm(1)    = 9.2225583816,
   nat          =  1 ,
   ntyp         = 1 ,
   ecutwfc      = 30.0,
   occupations  = 'smearing',
   smearing     ='marzari-vanderbilt',
   degauss      = 0.05,
   nbnd         = 10,
/
&electrons
   mixing_beta = 0.7
   conv_thr    =  1.0d-10
/
ATOMIC_SPECIES
 Pb 207.2   pb_s.UPF
ATOMIC_POSITIONS crystal
Pb  0.000000000   0.000000000   0.000000000
K_POINTS crystal
216
  0.00000000  0.00000000  0.00000000  4.629630e-03
  0.00000000  0.00000000  0.16666667  4.629630e-03
...
```

`$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/pw.x -npool 4 < nscf.in > nscf.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.

▶ Perform an EPW calculation to interpolate the electron-phonon matrix element from a coarse 6x6x6 to a dense 12x12x12 **k**-point grid and from a coarse 3x3x3 to a dense 12x12x12 **q**-point grid.

```
--                                                                           epw.in
&inputepw
  prefix      = 'lead',
  amass(1)    = 207.2
  outdir      = './'
  dvscf_dir   = './save'

  elph        = .true.
```

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

```
    kmaps       = .false.
    epbwrite    = .true.
    epbread     = .false.
    epwwrite    = .true.
    epwread     = .false.

    nbndsub     = 4
    nbndskip    = 5
    wannierize  = .true.
    num_iter    = 300
    dis_win_max = 21
    dis_win_min = -3
    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.

    parallel_k  = .true.
    parallel_q  = .false.

    fsthick     = 1 ! eV
    eptemp      = 0.075 ! K
    degaussw    = 0.1 ! eV

    nkf1        = 20
    nkf2        = 20
    nkf3        = 20
    nqf1        = 12
    nqf2        = 12
    nqf3        = 12

    nk1         = 6
    nk2         = 6
    nk3         = 6
    nq1         = 3
    nq2         = 3
    nq3         = 3
/
4 cartesian
  0.000000000    0.000000000    0.000000000
 -0.333333333    0.333333333   -0.333333333
  0.000000000    0.666666667    0.000000000
  0.666666667   -0.000000000    0.666666667
```

Since we want to compute the resistivity of Pb, we need the isotropic transport spectral function:

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

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

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

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

Note: The list of coarse **q**-point given after `4 cartesian` in the input file should be exactly the same as in the output of the phonon run. The `dvscf_dir = './save'` specify the place where we have placed the `.dyn`, `.dvscf` and `patterns` using the python script.

```
$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/epw.x -npool 4 < epw.in > epw.out &
```

The calculation should take about 4 min to be completed. While the calculation is running, notice in the `epw.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 12x12x12 **k** and **q** grids.
At the end of the calculation, you should get:

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


lambda :    1.7886518
lambda_tr :   1.5037507


Estimated Allen-Dynes Tc


logavg =     0.0002607 l_a2F =    1.7891574
mu =   0.10 Tc =         5.457685619465 K
mu =   0.12 Tc =         5.189033769643 K
mu =   0.14 Tc =         4.919561033910 K
mu =   0.16 Tc =         4.649648721107 K
mu =   0.18 Tc =         4.379726653209 K
mu =   0.20 Tc =         4.110277688307 K
```
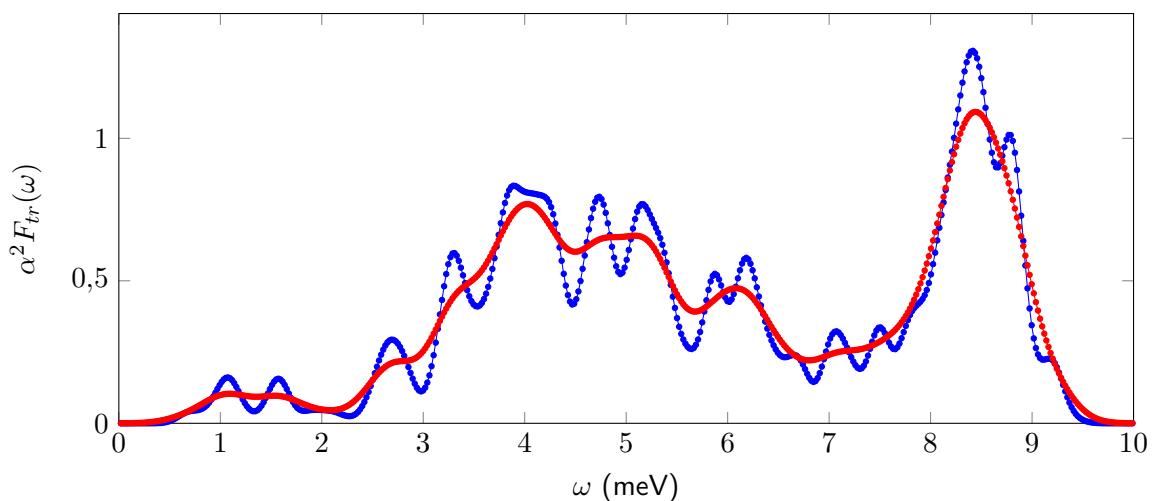
Note that the converged value for λ and λ_{tr} should be around 1.1. In addition the files `lead.a2f.01` and `lead.a2f_tr.01` which contains the spectral function and transport spectral function for different broadening should have been produced.
You should get something similar to this (here shown for two broadening 0.15 meV (blue) and 0.3 meV (red)):

Again this is unconverged. At convergence you should get something closer to [2]:
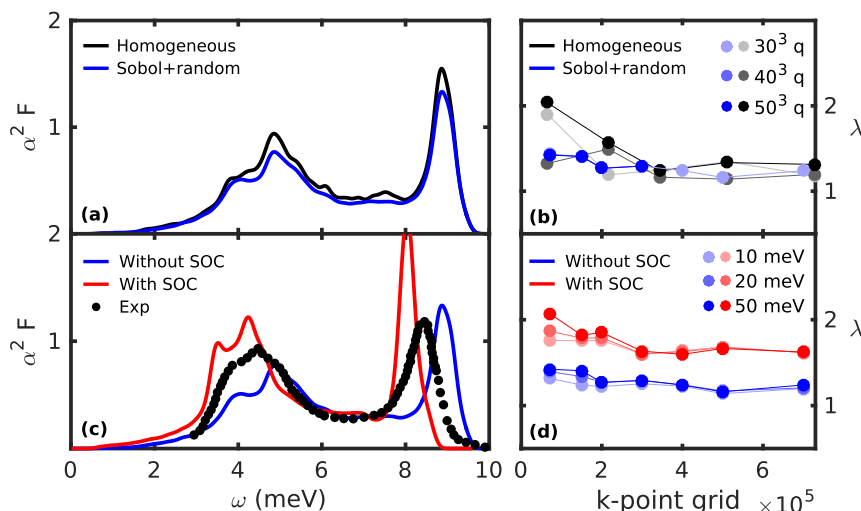
▶ Compute the resistivity of Pb using the Ziman's formula for metals:

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

where n is the number of electron per unit volume and $n(\omega, T)$ is the Bose-Einstein distribution. Usually this means *the number of electrons that contribute to the mobility* and so it is typically 8 (full shell) but not always. This number of electron is hardcoded in `EPW/src/a2f.f90` along with the temperature range (0-1000K). If you want to modify this, you will need to recompile the code.
The resistivity was actually computed during the previous run. A file named `lead.res.01` should have been created.
The file contains the resistivity (in $\mu\text{Ohm}\,\text{cm}$) for various temperature and smearing (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
     10  0.1558457   0.1562785   0.1570210   0.1581165   0.1596702   0.1618586
     20  0.7164545   0.7172950   0.7187385   0.7208710   0.7239029   0.7281869
     30  1.4179942   1.4192187   1.4213233   1.4244371   1.4288760   1.4351688
     40  2.1311356   2.1327492   2.1355236   2.1396310   2.1454923   2.1538130
     50  2.8314408   2.8334483   2.8369004   2.8420124   2.8493106   2.8596765
     60  3.5182289   3.5206328   3.5247670   3.5308897   3.5396324   3.5520532
     70  4.1944317   4.1972334   4.2020520   4.2091886   4.2193800   4.2338606
...
```

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

---

[2]The figure is from S. Poncé *et al.*, Computer Physics Communications **209**, 116 (2016).
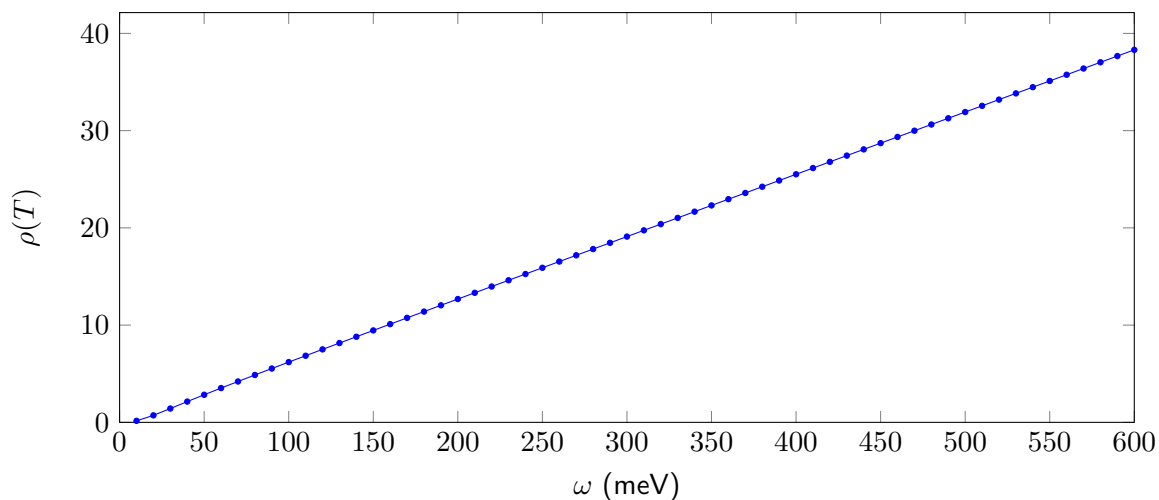
Note that in this case, as it is an integrated quantity, it is not so dependent on smearing. Compare your result with other smearing.

At convergence you should get [3]:
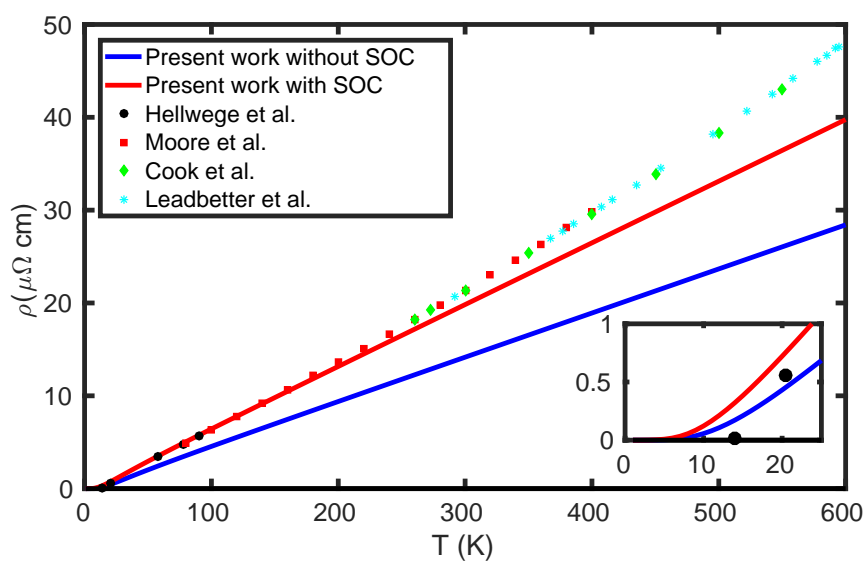
Exercise 2

In this example we are going to calculate the mobility of Si using the Boltzmann transport equation (BTE) in the self-energy relaxation time approximation (SERTA). Additional information can be found at arXiv:1803.05462.

First go to the second exercise:

```
$ cd tuto_Wed5/exercise2
```

▶ Make a self-consistent calculation for Si.

```
&control                                                                scf.in
    calculation    = 'scf'
```

---

[3]The figure is from S. Poncé *et al.*, Computer Physics Communications **209**, 116 (2016).

```
    prefix        = 'si'
    restart_mode  = 'from_scratch'
    wf_collect    = .true.
    pseudo_dir    = './'
    outdir        = './'
 /
 &system
    ibrav         = 2
    celldm(1)     = 10.262
    nat           = 2
    ntyp          = 1
    ecutwfc       = 20
 /
 &electrons
    diagonalization = 'david'
    mixing_beta   = 0.7
    conv_thr      = 1.0d-13
 /
ATOMIC_SPECIES
  Si  28.0855  Si_r.upf
ATOMIC_POSITIONS alat
  Si   0.00  0.00  0.00
  Si   0.25  0.25  0.25
K_POINTS automatic
12 12 12 0 0 0
```

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

```
$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/pw.x < scf.in > scf.out
```

▶ Compute the vibrational properties of Si on a coarse 3×3×3 **q**-point grid.

```
--                                                                         ph.in
&inputph
  prefix  = 'si',
  epsil   = .false.,
  fildyn  = 'si.dyn',
  ldisp   = .true.
  fildvscf = 'dvscf'
  nq1=3,
  nq2=3,
  nq3=3,
  tr2_ph  = 1.0d-12,
 /
```

```
$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/ph.x < ph.in > ph.out &
```

The calculation should take about 15 min on 4 cores. During the run, notice the IBZ q-point grid:

```
    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
```

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

```
        Dielectric constant in cartesian axis

        (      12.982912147      -0.000000000       0.000000000 )
```

```
          (        -0.000000000       12.982912147        -0.000000000 )
          (         0.000000000       -0.000000000        12.982912147 )

        Effective charges (d Force / dE) in cartesian axis

          atom      1   Si
    Ex  (        -0.00778         -0.00000          0.00000 )
    Ey  (        -0.00000         -0.00778          0.00000 )
    Ez  (         0.00000          0.00000         -0.00778 )
          atom      2   Si
    Ex  (        -0.00778         -0.00000         -0.00000 )
    Ey  (        -0.00000         -0.00778          0.00000 )
    Ez  (         0.00000          0.00000         -0.00778 )
```

The experimental dielectric constant in silicon is about 11.9 so that we overestimates it a bit. More accurate values can be obtained with larger **k**-point grids but we expect to overestimate the experimental value due to the DFT band gap problems. Since Si is a non-polar material, the Born effective charges should be 0. Indeed, one can see that we are close to zero. Moreover in non-polar materials the dielectric constant is not used but is a good way to check for convergence. In your calculations you may want to improve this by using a centrosymmetric unit cell.

▶ Impose the crystal acoustic sum rule and create the interatomic force constant (IFC).

```
&input fildyn='si.dyn', zasr='crystal', flfrc='si.fc' /                              q2r.in
```

$ /home/nfs3/smr3191/q-e/bin/q2r.x < q2r.in

This will create a file named `si.fc` that contain the real space IFC.

▶ Like for the first exercise, run the python post-processing to create the `save` folder

$ python /home/nfs3/smr3191/q-e/EPW/bin/pp.py

Note: This time the `pp.py` script will also copy the IFC inside the `save` folder.

▶ Do a non self-consistent calculation on a 6x6x6 uniform and Γ-centered **k**-point grid with crystal coordinates in the interval [0,1[
Such grid can be for example generated using `python` with

```
import numpy as np
N = 6
for ii in np.arange(0,1.0,1.0/N):
  for jj in np.arange(0,1.0,1.0/N):
    for kk in np.arange(0,1.0,1.0/N):
      print ii,' ',jj, ' ',kk, ' ',1.0/(N**3)
```

But in any case the grid is included in the `nscf.in` file for you.

```
 &control                                                                            nscf.in
    calculation    = 'nscf'
    prefix         = 'si'
    restart_mode   = 'from_scratch'
    wf_collect     = .true.
    pseudo_dir     = './'
    outdir         = './'
 /
```

```
 &system
    ibrav         = 2
    celldm(1)     = 10.262
    nat           = 2
    ntyp          = 1
    ecutwfc       = 20
    nbnd          = 12
 /
 &electrons
    diagonalization = 'david'
    mixing_beta   = 0.7
    conv_thr      = 1.0d-10
 /
ATOMIC_SPECIES
  Si  28.0855  Si_r.upf
ATOMIC_POSITIONS alat
  Si  0.00  0.00  0.00
  Si  0.25  0.25  0.25
K_POINTS crystal
216
  0.00000000  0.00000000  0.00000000  4.629630e-03
  0.00000000  0.00000000  0.16666667  4.629630e-03
...
```

```
$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/pw.x -npool 4 < nscf.in > nscf.out
```

▶ Perform an EPW calculation to Fourier-transform the electron-phonon matrix element from a coarse 6x6x6 **k** and 3x3x3 **q**-point grids to real space and then interpolate the electronic and phononic bandstructure along the $L - \Gamma - X$ high symmetry line.

```
--                                                                          epw1.in
&inputepw
  prefix     = 'si'
  amass(1)   = 28.0855
  outdir     = './'

  elph       = .true.
  kmaps      = .false.
  epbwrite   = .true.
  epbread    = .false.
  epwwrite   = .true.
  epwread    = .false.
  etf_mem    = 1


  nbndsub    =  8
  nbndskip   =  0


  lifc       = .true.
  asr_typ    = 'crystal'


  wannierize = .true.
  num_iter   = 1500
  iprint     = 2
  dis_win_max = 18
  dis_froz_max= 8.5
  proj(1)    = 'Si : sp3'
  wdata(1) = 'bands_plot = .true.'
  wdata(2) = 'begin kpoint_path'
  wdata(3) = 'L 0.50 0.00 0.00 G 0.00 0.00 0.00'
  wdata(4) = 'G 0.00 0.00 0.00 X 0.50 0.50 0.00'
  wdata(5) = 'end kpoint_path'
  wdata(6) = 'bands_plot_format = gnuplot'
  wdata(7) = 'guiding_centres = .true.'
  wdata(8) = 'dis_num_iter      = 500'
  wdata(9) = 'num_print_cycles  = 10'
  wdata(10) = 'dis_mix_ratio     = 1.0'
  wdata(11) = 'use_ws_distance = T'
```

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

    parallel_k  = .true.
    parallel_q  = .false.

    fsthick     = 1.2 ! eV
    eptemp      = 1 ! K
    degaussw    = 0.005 ! eV

    dvscf_dir   = './save'

    band_plot   = .true.
    filkf       = './LGX.txt'
    filqf       = './LGX.txt'

    nk1         = 6
    nk2         = 6
    nk3         = 6
    nq1         = 3
    nq2         = 3
    nq3         = 3
/
  4 cartesian
  0.000000000    0.000000000    0.000000000
 -0.333333333    0.333333333   -0.333333333
  0.000000000    0.666666667    0.000000000
  0.666666667   -0.000000000    0.666666667
```

`$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/epw.x -npool 4 < epw1.in > epw1.out &`

The calculation should take about 4 min. In the output, notice the additional information related to the crystal ASR. At the end of the calculation, because of the keyword band_plot = .true., the code should produce a `band.eig` and `phband.freq` files that contains the electronic and phononic bandstructure along a path given by filkf and filqf.
If you want to have files in an easy gnuplot format, you can use the `plotband.x` tool by doing

`$ q-e/bin/plotbands.x`

and follow the instructions. You should check that both bandstructure looks reasonable.

▶ Do a restart calculation (restarting from the `si.epmatwp1` file) a compute the electron and hole mobility of Si. The electron mobility is defined as:

$$\mu_{\mathrm{e},\alpha\beta} = \frac{-e}{n_{\mathrm{e}}\,\Omega} \sum_{n\in\mathrm{CB}} \int \frac{d\mathbf{k}}{\Omega_{\mathrm{BZ}}} \, \frac{\partial f^0_{n\mathbf{k}}}{\partial \varepsilon_{n\mathbf{k}}} v_{n\mathbf{k},\alpha}\, v_{n\mathbf{k},\beta}\, \tau^0_{n\mathbf{k}}, \tag{4}$$

where n_e is the number of electrons, Ω is the volume of the crystalline unit cell, Ω_{BZ} the volume of the first Brillouin zone, $f^0_{n\mathbf{k}}$ is the Fermi-Dirac distribution, $v_{n\mathbf{k},\alpha} = \hbar^{-1}\partial\varepsilon_{n\mathbf{k}}/\partial k_\alpha$ is the band velocity of the single-particle electron eigenvalue $\varepsilon_{n\mathbf{k}}$ and where the relaxation rate is defined by

$$\frac{1}{\tau^0_{n\mathbf{k}}} = 2\mathrm{Im}\Sigma^{\mathsf{FM}}_{n\mathbf{k}} = \frac{2\pi}{\hbar} \sum_{m\nu} \int \frac{d\mathbf{q}}{\Omega_{\mathrm{BZ}}} \, |g_{mn\nu}(\mathbf{k},\mathbf{q})|^2 \left[(1 - f^0_{m\mathbf{k}+\mathbf{q}} + n_{\mathbf{q}\nu})\delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} - \hbar\omega_{\mathbf{q}\nu}) \right.$$
$$\left. + (f^0_{m\mathbf{k}+\mathbf{q}} + n_{\mathbf{q}\nu})\delta(\varepsilon_{n\mathbf{k}} - \varepsilon_{m\mathbf{k}+\mathbf{q}} + \hbar\omega_{\mathbf{q}\nu}) \right]. \tag{5}$$

A similar equation holds for hole mobility.
The input file is as follow (only the difference wrt `epw1.in` are shown):

```
--                                                                          epw2.in
  kmaps        = .true.
  epbwrite     = .false.
  epbread      = .false.
  epwwrite     = .false.
  epwread      = .true.

  scattering   = .true.
  scattering_serta = .true.
  int_mob      = .true.
  carrier      = .true.
  ncarrier     = 1E13

  scissor      =  0.7
  nstemp       =    5
  tempsmin     = 100
  tempsmax     = 500

  restart      = .true.
  restart_freq = 500

  filkf        = './kpt.txt'
  nqf1         = 20
  nqf2         = 20
  nqf3         = 20
```

Notes:

- epwread allows for the restart from the si.epmatwp1 file

- scattering_serta selects a calculation of mobility in the self-energy relaxation time approximation.

- the int_mob alone means intrinsic mobility with only 1 Fermi level. However, if you also have the carrier keyword activated, then you will have 2 Fermi level such that the carrier concentration for the electron and hole is ncarrier (in cm^{-3}).

- scissor makes a scissor shift to correct for the DFT underestimation of the bandgap

- nstemp,tempsmin,tempsmax defines the temperature at which the mobility is evaluated. Note that these input variable supersedes the eptemp variable that has no effect in this context.

- restart and restart_freq will create restart point every (in this case) 500 **q**-points. The value of the mobilities will also be given in the output whenever a restart point is created. You can try breaking the run after a restart point and restart to test this feature.

- To run an IBTE instead of SERTA calculation (not done here as it takes a lot of time), add the keyword iterative_bte = .true. along with scattering_serta = .true. and int_mob.

`$ mpirun -np 4 /home/nfs3/smr3191/q-e/bin/epw.x -npool 4 < 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. Try filling the table below:

Temperature (K)	hole ε_F (eV)	hole μ (cm^2/Vs)	electron ε_F (eV)	electron μ (cm^2/Vs)
100				
200				
300				
400				
500				

At convergence you should get [4]:

---

[4]The figure is from S. Poncé *et al.*, Physical Review B, in press (2018) and can be found on arXiv:1803.05462
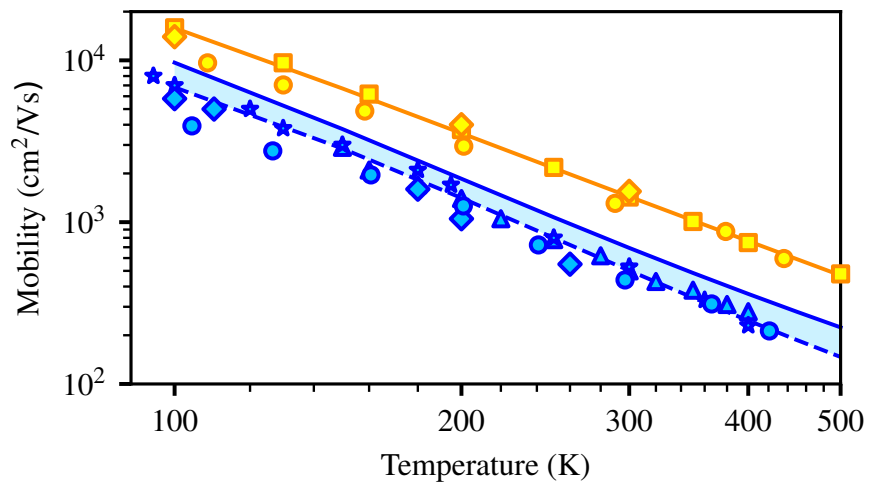
where the room temperature values should be around 700 cm^2/Vs for the hole and 1400 cm^2/Vs for the electron mobility.

▶ Try to increase the fine grids and add a few more temperatures and see if you can get a result closer to convergence.