Electron-phonon coupling with EPW

Tutorial Fri.1

Hands-on session

Hands-on based on Quantum Espresso 6.5

In this session we will learn how to use the core capabilities of EPW.
First copy the tutorial input files of the exercise:

```
$ git clone https://github.com/wannier-developers/wannier-tutorials.git
$ cd 2020_03_Oxford/3_epw/tuto_epw_core/
```

Electron-phonon interaction in SiC

In this exercise we will examine the electron-phonon interactions in silicon carbide. SiC is a polar material, where the long-range polar coupling results in a $1/|\mathbf{q}|$ divergence of the matrix elements near Γ. Here we will see how to correctly interpolate the matrix elements in this case, and we will calculate the electron linewidths.

▶ Run a self-consistent calculation for SiC:

scf.in

```
--
 &control
    calculation     = 'scf'
    prefix          = 'sic'
    restart_mode    = 'from_scratch'
    wf_collect      = .true.
    pseudo_dir      = './'
    outdir          = './'
 /
 &system
    ibrav           = 2
    celldm(1)       = 8.237
    nat             = 2
    ntyp            = 2
    ecutwfc         = 30.0
 /
 &electrons
    diagonalization = 'david'
    mixing_beta     = 0.7
    conv_thr        = 1.0d-10
 /
ATOMIC_SPECIES
  Si  28.0855     Si.pz-vbc.UPF
  C   12.01078    C.UPF
ATOMIC_POSITIONS alat
  Si  0.00  0.00  0.00
  C   0.25  0.25  0.25
K_POINTS automatic
8 8 8 0 0 0
```

```
$ mpirun -np 2 pw.x  < scf.in > scf.out
```

▶ Run a phonon calculation on a homogeneous $3 \times 3 \times 3$ $\mathbf{q}$-point grid.

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

```
$ mpirun -np 2 ph.x -npool 2 < ph.in > ph.out
```

Note that the output now contains also the Born effective charges Z^* and the electronic dielectric constant ϵ_∞:

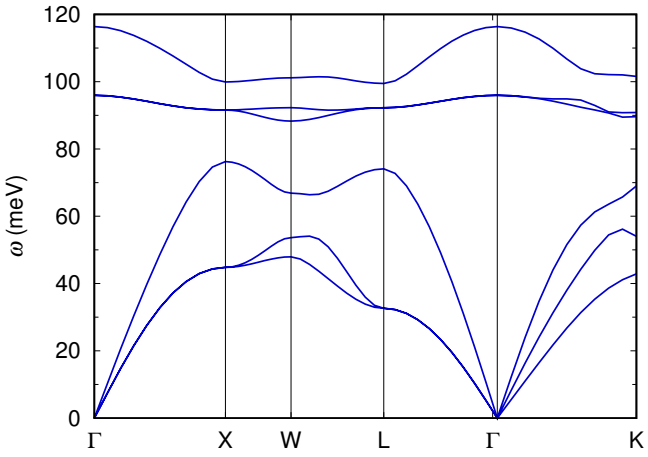
```
     Electric Fields Calculation

     ....

     End of electric fields calculation

          Dielectric constant in cartesian axis

          (       7.214167210       0.000000000       0.000000000 )
     ....

          Effective charges (d Force / dE) in cartesian axis

           atom      1   Si
      Ex  (        2.67035        0.00000        0.00000 )
       ....
```

These quantities are automatically calculated for an insulating system, using the finite response to an electric field, and determine the energy splitting of the TO and LO phonons.

You can plot the phonon dispersion using the `q2r.x`, `matdyn.x` and `plotbands.x` tools with:

```
$ q2r.x < q2r.in > q2r.out
$ matdyn.x < matdyn.in > matdyn.out
$ plotband.x < plotband.in > plotband.out
```

The plot should look like this:



Note that now there are 3 acoustic and 3 optical branches, separated by an energy gap, and that there is the LO-TO splitting at Γ.

▶ Gather the `.dyn`, `.dvscf` and `patterns` files into a `save/` directory using the `pp.py` script:

```
$ python pp.py
```

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

▶ Clean up unnecessary files:

```
$ rm -r _ph0/
```

▶ Re-run the scf calculation (because the earlier phonon calculation might have modified the density files):

```
$ mpirun -np 2 pw.x < scf.in > scf.out
```

▶ Run a non self-consistent calculation on a homogeneous $6 \times 6 \times 6$ $\mathbf{k}$ grid. To do so, first copy the input file for the self-consistent calculation into a new file `nscf.in`; then modify the input variable `calculation` to `nscf`, set the number of bands to `nbnd=4` inside the `&system` block (we will look only at the valence bands here), and delete the last two lines specifying the $\mathbf{k}$-point grid. Then you can use again the script `kmesh.pl` to generate the homogeneous $\mathbf{k}$ grid:

```
$ ./kmesh.pl 6 6 6 >> nscf.in
```

Now your input file should look like this:

nscf.in

```
&control
    calculation     = 'nscf'
    prefix          = 'sic'
    wf_collect      = .true.
    pseudo_dir      = './'
    outdir          = './'
 /
&system
    ibrav           = 2
    celldm(1)       = 8.237
    nat             = 2
```

```
    ntyp            = 2
    ecutwfc         = 30.0
    nbnd            = 4
 /
&electrons
    diagonalization = 'david'
    mixing_beta     = 0.7
    conv_thr        = 1.0d-10
 /
ATOMIC_SPECIES
  Si  28.0855     Si.pz-vbc.UPF
  C   12.01078    C.UPF
ATOMIC_POSITIONS alat
  Si  0.00  0.00  0.00
  C   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
...
```

Now you can run:

```
$ mpirun -np 2 pw.x -npool 2 < nscf.in > 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.

▶ Calculate the interpolated electron-phonon matrix elements along some Brillouin-zone directions (input variable `prtgkk = .true.`).

```
--                                                                    epw1.in
&inputepw
  prefix      = 'sic'
  amass(1)    = 28.0855
  amass(2)    = 12.0107
  outdir      = './'
  dvscf_dir   = './save'

  elph        = .true.
  kmaps       = .false.
  epbwrite    = .true.
  epbread     = .false.
  epwwrite    = .true.
  epwread     = .false.
  lpolar      = .true.

  wannierize  = .true.
  nbndsub     =  4
  nbndskip    =  0
  num_iter    = 300
  proj(1)     = 'Si:sp3'

  elecselfen  = .false.
  phonselfen  = .false.
  a2f         = .false.
  prtgkk      = .true.

  fsthick     = 7.0
  eptemp      = 20
```

```
  degaussw     = 0.05

  filqf        = 'path.dat'
  nkf1         = 1
  nkf2         = 1
  nkf3         = 1

  nk1          = 6
  nk2          = 6
  nk3          = 6
  nq1          = 3
  nq2          = 3
  nq3          = 3
 /
  4
  0.000000000000000E+00   0.000000000000000E+00   0.000000000000000E+00
 -0.333333333333333E+00   0.333333333333333E+00  -0.333333333333333E+00
  0.000000000000000E+00   0.666666666666667E+00   0.000000000000000E+00
  0.666666666666667E+00  -0.555111512312578E-16   0.666666666666667E+00
```

In the `epw1.in` input file, `path.dat` is a high-symmetry path in the BZ. For the definition of input variables you can refer to the EPW website http://epw.org.uk/Documentation/Inputs.
Note also that in order to speed up the calculation, we chose to only print the matrix elements for the initial electronic states at $\mathbf{k} = \Gamma$ (`nkf1=nkf2=nkf3=1`).

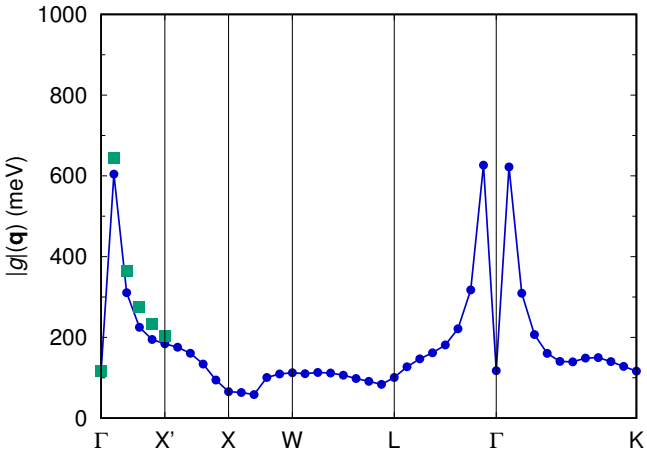
```
$ mpirun -np 2 epw.x -npool 2 < epw1.in > epw1.out
```

Since SiC is a polar material (non-zero Born effective charges), we set `lpolar = .true.` in order to correctly treat the long-range interaction in bulk crystals. The strategy consists in subtracting the long-range component $g^{\mathcal{L}}$ from the full matrix element g before interpolation and adding it back after interpolation (see morning lecture and the following paper Phys. Rev. Lett. **115**, 176401 (2015)). An analogous strategy is implemented to correctly interpolate the dynamical matrix including the long-range dipole-dipole interactions which result in the LO-TO splitting.

The matrix elements for each $\mathbf{q}$ along the BZ path are written in the output, with columns corresponding to each band and phonon mode. An average over degenerate bands and modes has already been performed. If you want to plot the matrix elements for the valence-band top ($n = m = 4$) and the highest-energy phonon branch ($\nu = 6$), for example, you can type:

```
$ grep '4        4        6' epw1.out | awk '{print $7}' >matel.dat
```

and the plot should look similar to:



If you use a denser $\mathbf{q}$-point path, the matrix elements will keep diverging near Γ, in agreement with the direct calculations using `ph.x`.
The matrix elements obtained by direct DPFT calculations are reproduced as green squares. Note that the interpolated ones are not fully accurate, since a larger coarse $\mathbf{q}$-grid is needed in order to accurately interpolate the electron-phonon matrix elements.

Note: if you wish to know how to compute the direct DFPT matrix element, please ask a tutor.

Note 2: if you want to inspect the results of the interpolation when the long-range contribution to the matrix elements is not taken into account, you can run a new calculation using `lpolar=.false.`. Remember to run this test in a new directory, or to run again an `EPW` calculation from scratch using `lpolar=.true.`.

▶ Plot the interpolated electron and phonon band structure and compare them with the results from `pw.x` (you will need to perform a band structure calculation) and from `matdyn.x`, respectively. Note that we are only wannierizing the 4 valence bands.
Hint: You will need to use `filkf` instead of `filqf` for the electron bandstructure in the EPW input file. The code produces a `band.eig` and `phband.freq` file with the electron and phonon bandstructure but you will need to again use `plotband.x` to transform them in a `gnuplot` format.

▶ The Wannier-Fourier interpolation technique is based on the decay properties of the Wannier functions and of the phonon perturbation in real space. To check how each quantity is decaying within the supercell corresponding to our initial $\mathbf{k}$ and $\mathbf{q}$ grids, we can plot the files `decay.H` (Hamiltonian), `decay.D` (dynamical matrix) and `decay.epmat_wanep` (matrix elements; you can plot the data using the first and last column).
From these plots we see clearly that we need a larger supercell (denser $\mathbf{q}$ grid) in order to interpolate more accurately the phonon properties, whereas the electronic part decays well (remember that we are using a $6 \times 6 \times 6$ $\mathbf{k}$ grid, and only a $3 \times 3 \times 3$ $\mathbf{q}$ grid).

▶ We now calculate the linewidths of the valence states in SiC along high-symmetry lines, which correspond to twice the imaginary part of the electron self-energy. To do so, we need to use the input variable `elecselfen = .true.`, the $\mathbf{k}$-point path, and a homogeneous $\mathbf{q}$ grid for the integration. Copy the `EPW` input into a new one:

```
$ cp epw1.in epw2.in
```

and modify it as follows:

epw2.in

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

wannierize  = .false.
...

elecselfen  = .true.
phonselfen  = .false.
a2f         = .false.
prtgkk      = .false.
efermi_read = .true.
fermi_energy= 9.6
...

!filqf       = 'path.dat'
filkf       = 'path.dat'
nqf1        = 20
nqf2        = 20
nqf3        = 20
...
```

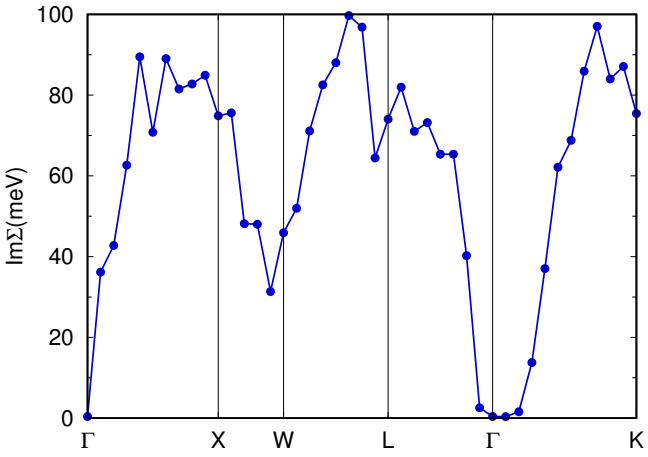
Note that since we use a $\mathbf{k}$ path, the Fermi energy calculated from the fine (interpolated) grid will not be accurate. To overcome this problem, we provide the Fermi energy in the input by using `efermi_read = .true.` and `fermi_energy=9.6` (just above the valence-band top).

In the output you can monitor the progression of the $\mathbf{q}$ integration, before reading the electron self-energy for each $\mathbf{k}$ point:

```
Progression iq (fine) =        100/      8000
Progression iq (fine) =        200/      8000
...
...
Average over degenerate eigenstates is performed
WARNING: only the eigenstates within the Fermi window are meaningful

ik =       1 coord.:    0.0000000   0.0000000   0.0000000
-------------------------------------------------------------------
E(   2 )=  -0.2405 eV   Re[Sigma]=      95.950765 meV Im[Sigma]=       0.392034 meV
E(   3 )=  -0.2405 eV   Re[Sigma]=      95.950765 meV Im[Sigma]=       0.392034 meV
E(   4 )=  -0.2405 eV   Re[Sigma]=      95.950765 meV Im[Sigma]=       0.392034 meV
-------------------------------------------------------------------
...
```

Note that the electron energies are now reported with respect to E_{F}. Moreover, the self-energy for the first valence band is not computed since `fsthick` is 7 eV. To plot the linewidths you can use the file `linewidth.elself` that has been created. For example, ImΣ for the highest valence band should look like:

```
gnuplot> plot "linewidth.elself" u 1:4 every 3::2 w lp
```



If you want to plot the electron lifetimes, these are given by $\tau_{n\mathbf{k}} = \hbar/(2\text{Im}\Sigma_{n\mathbf{k}})$.

Can you understand the behavior of the linewidths along the Brillouin-zone path? It is useful to look also at the electronic band structure.

Note: You can also look at the contribution of each phonon mode by using `iverbosity = 3` in the input. The file `linewidth.elself` will then contain the mode-resolved linewidths. Can you tell which phonon has the largest contribution and why?

▶ Try increasing the $\mathbf{q}$ grid, and also using a random set of $\mathbf{q}$ points: you will see the linewidths are not well converged yet. For polar materials it is indeed more difficult to converge Brillouin-zone integrals, due to the $1/|\mathbf{q}|$ divergence of the Fröhlich matrix elements. Faster convergence can be achieved by using adaptive grids, such as grids generated following a Lorentzian (Cauchy) distribution centered around $\mathbf{q} = 0$.