Tutorial Fri.5 for ZG tree in EPW v.5.5

Temperature-dependent band structures, optical spectra, phonon-induced diffuse scattering, and unfolding principle

Hands-on Session (Friday, 16th June)

In this session we will learn how to use `ZG.x` for generating ZG supercell configurations and how to combine them with standard Density Functional Theory (DFT) calculations for evaluating temperature-dependent properties. You are advised to prepare the following script file, e.g. `script.sh`:

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=56
#SBATCH --time=01:00:00
#SBATCH --account=EPSchool2022
#SBATCH --partition=small
#SBATCH --reservation=EPSchoolDay5
module purge
module load TACC
PATHQE=/work2/06868/giustino/EP-SCHOOL/q-e/ # this is the path to q-e
cd $PWD
```

For each calculation, pass the command at the bottom of the script and submit the job, e.g. `sbatch script.sh`. Also set the following environment variable in your shell:

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

Some routines in PW, PH, and PP directories are required to run these exercises (e.g. `make pw ph pp`). To compile executables in ZG tree go to `$PATHQE/EPW/ZG/src` and type `make`. To use local executables go to `$PATHQE/EPW/ZG/src/local` and type `./compile_ifort.sh`.

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

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

▶ Copy the tutorial tarball from `/work2/06868/giustino/EP-SCHOOL/` and untar. Then, go to `exercise1`, and create directory `workdir`:

```
$ cp /work2/06868/giustino/EP-SCHOOL/Fri.5.Zacharias.tar .
$ tar -xvf Fri.5.Zacharias.tar; cd tutorial/exercise1; mkdir workdir
```

Note: in this tutorial we will show how to obtain all input and output files. The directory `inputs` will be used as a reference or to speed up the process. In case you missed a step, copy the appropriate input file from `inputs` to continue. Most of the temperature-dependent calculations will be performed for $T = 0$ K; one can repeat the steps using a different temperature.

Exercise 1

In this exercise we will generate the ZG configuration of silicon in a $3 \times 3 \times 3$ supercell for the temperature $T = 0$ K and run your first DFT-ZG calculation. In the following, all steps for obtaining the phonons of silicon are provided, but to speed up the process we will skip the first *four* steps.

Note: For generating successfully a ZG configuration you need to make sure that the phonon dispersion is as you would

expect (compare with literature). If there exist negative frequencies (soft modes), the code sets them to positive.

▶ Run a self-consistent calculation for silicon in the `workdir`.
Note: The energy cutoff ecutwfc needed for convergence should be 30 Ry.

```
$  cd workdir; cp ../inputs/si.scf.in .; cp ../inputs/Si.pz-vbc.UPF .
$  ibrun -np 4 $PATHQE/bin/pw.x -nk 4 < si.scf.in > si.scf.out
```

▶ Run a `ph.x` calculation on a homogeneous $4 \times 4 \times 4$ $\mathbf{q}$-point grid using the input:

```
--                                                                    si.ph.in
&inputph
   amass(1) = 28.0855,
   prefix   = 'si'
   outdir   = './'
   ldisp    = .true.
   fildyn   = 'si.dyn'
   tr2_ph   = 1.0d-12
   nq1      = 4, nq2 = 4, nq3 = 4
/
```

```
$ cp ../inputs/si.ph.in .
$ ibrun -np 4 $PATHQE/bin/ph.x -nk 4 < si.ph.in > si.ph.out
```

This will generate 8 **si.dynX** output files containing the dynamical matrix calculated for each irreducible $\mathbf{q}$-point. The list of irreducible $\mathbf{q}$-points is written in the **si.dyn0** file.

▶ Run a `q2r.x` calculation to obtain the interatomic force constants (IFC) file using the input:

```
--                                                                      q2r.in
&input
 fildyn='si.dyn', flfrc = 'si.444.fc'
/
```

```
$  cp ../inputs/q2r.in .
$  ibrun -np 1 $PATHQE/bin/q2r.x < q2r.in > q2r.out
```

This will generate **si.444.fc** which contains the IFCs. As we will see below, this is the only external input file necessary for running `ZG.x`.

▶ Run a `matdyn.x` calculation to check the phonon dispersion:

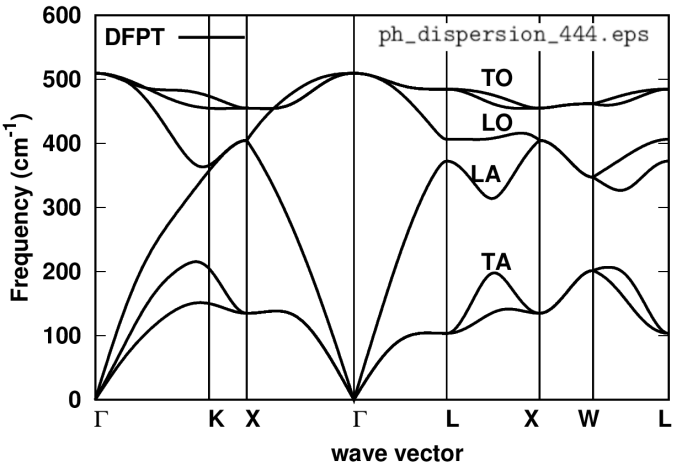
```
--                                                                    matdyn.in
&input
    asr='simple', amass(1)=28.0855,
    flfrc='si.444.fc', fldyn='si.dyn.mat', flfrq='si.freq', fleig='si.dyn.eig',
    q_in_cryst_coord = .false., q_in_band_form = .true.
/
 9
 0.00  0.00  0.00  100
 0.75  0.75  0.00  1
 0.25  1.00  0.25  100
 0.00  1.00  0.00  100
 0.00  0.00  0.00  100
 0.50  0.50  0.50  100
 0.00  1.00  0.00  100
 0.50  1.00  0.00  100
 0.50  0.50  0.50  100
```

```
$  cp ../inputs/matdyn.in .
$  ibrun -np 1 $PATHQE/bin/matdyn.x < matdyn.in > matdyn.out
$  $PATHQE/bin/plotband.x
```

```
Input file >  si.freq
Reading    6 bands at    702 k-points
Range:    0.0000  509.7618eV  Emin, Emax, [firstk, lastk] > 0 600
high-symmetry point:  0.0000 0.0000 0.0000   x coordinate   0.0000
high-symmetry point:  0.7500 0.7500 0.0000   x coordinate   1.0607
...
high-symmetry point:  0.5000 0.5000 0.5000   x coordinate   5.3534
output file (gnuplot/xmgr) > si_ph_dispersion.xmgr
bands in gnuplot/xmgr format written to file si_ph_dispersion.xmgr

output file (ps) >
stopping ...
```

`matdyn.x` will generate **si.freq** which contains the phonon frequencies along the specified path in the Brillouin zone. Then we combine **si.freq** with `plotband.x` to obtain **si_ph_dispersion.xmgr** in gnuplot/xmgr format for the phonon dispersion. To plot the phonon dispersion type:

```
$  cp si_ph_dispersion.xmgr ../gnuplot/.
$  cd ../gnuplot/; gnuplot gp_coms.p; evince ph_dispersion_444.eps
```



Is the phonon dispersion the one you would expect? One can verify the results with literature data.

Once you have obtained the IFC file (e.g. here **si.444.fc**) and verified the correctness of your phonons, you are ready to perform a `ZG.x` calculation. If you have skipped the four steps above, you can copy the IFC file from `inputs` into `workdir`:

```
$ cd ../workdir; cp ../inputs/si.444.fc .
```

▶ Run a ZG calculation using the input:

```
--                                                                                ZG_333.in
&input
    flfrc='si.444.fc',
    asr='simple', amass(1)=28.0855, atm_zg(1) = 'Si',
    flscf = 'si.scf.in'
    T = 0.00,
    dim1 = 3, dim2 = 3, dim3 = 3
    compute_error = .true., synch = .true., error_thresh = 0.02, niters = 30000
    incl_qA = .false.
/
```

Note: The q-point grid (nq1×nq2×nq3) used to obtain the dynamical matrices should not be necessarily the same with the supercell size (dim1×dim2×dim3). In particular, dimX is independent of nqX, since ZG.x takes advantage of Fourier interpolation as implemented in matdyn.x; thus any size of ZG configuration can be generated from **si.444.fc**.

```
$  cp ../inputs/ZG_333.in .
$  ibrun -np 4 $PATHQE/bin/ZG.x -nk 4 < ZG_333.in > ZG_333.out
```

The input format of ZG_333.in is the same as matdyn.in. Lines 3 and 4 contain input variables similar to those in matdyn.in; in line 5 we provide the name of the unit-cell scf file (flscf) used to calculate the phonons. The information therein is read to generate the corresponding supercell scf file (new feature); in lines 6 and 7 we provide the temperature (T) in Kelvin and supercell dimensions (dimX) for which ZG displacements are generated; line 8 contains all flags related to the error minimization (discussed later); in line 8 we specify whether we want to include, or not, q-points in set $\mathcal{A}$ using the flag incl_qA.

The output files from a ZG.x run are: (a) **ZG-configuration_0.00K.dat** and **equil_pos.dat** which contain the ZG (quantum nuclei at 0.00 K) and equilibrium (classical nuclei at 0 K) coordinates in angstroms, and (b) **ZG-scf_333_0.00K.in** and **equil-scf_333.in** which are the corresponding scf files for performing calculations in a $3 \times 3 \times 3$ supercell.

▶ Run a self-consistent calculation for silicon with the nuclei clamped at their equilibrium and ZG coordinates (for $T = 0.00$ K) in the supercell.

```
$  ibrun -np 54 $PATHQE/bin/pw.x -nk 3 < equil-scf_333.in > equil-scf_333.out
$  ibrun -np 56 $PATHQE/bin/pw.x -nk 8 < ZG-scf_333_0.00K.in > ZG-scf_333_0.00K.out
```

Note: parallelization over 3 and 8 **k**-points.

The file **equil-scf_333.in** is also provided here:

```
&control                                                    equil-scf_333.in
  calculation = 'scf'
  restart_mode = 'from_scratch'
  prefix = 'equil-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
K_POINTS automatic
   2   2   2   0   0   0
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
```

```
ATOMIC_POSITIONS {angstrom}
   Si     0.00000000   0.00000000   0.00000000
   Si    -1.34940189   1.34940189   1.34940189
   Si    -2.69880378   0.00000000   2.69880378
...
```

Note: Cell paramaters, number of atoms, **k**-grid, and atomic coordinates have been modified automatically based on the supercell dimensions. The code will always generate the lattice information in angstroms.

The file **ZG-scf_333_0.00K.in** is also provided here:

```
&control                                                    ZG-scf_333_0.00K.in
  calculation = 'scf'
  restart_mode = 'from_scratch'
  prefix = 'ZG-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
K_POINTS automatic
   2   2   2   0   0   0
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS {angstrom}
   Si       0.02090173     -0.01220213     -0.01392228
   Si      -1.39812907      1.32606198      1.38415986
   Si      -2.68002772      0.00399202      2.64203512
...
```

Note: This file contains the same information with **equil-scf_333.in**, except from the prefix and atomic positions. In general you should also add `nosym = .true.`, since symmetries do not apply after ZG displacements.

While the calculations are running, let us discuss some apsects about the generation of the **ZG-configuration** file. Type:

```
$ grep -3 "Optimum" ZG_333.out | tail -4

 Optimum configuration found !

 Sum of diagonal terms per q-point:     0.419085
 Error and niter index:     0.016917   18079
```

This information is printed during a `ZG.x` run when the optimum configuration is found. Line 3 gives the sum of all diagonal terms per q-point, representing the denominator of Eq. (54) in Ref. [Phys. Rev. Res. 2, 013357 (2020)]. The first entry in line 4 represents the value of the minimization function $E(\{S_{\mathbf{q},\nu}\})$ given by Eq. (54) in Ref. [Phys. Rev. Res. 2, 013357 (2020)]. The integer

in line 4 represents the number of attempts required to achieve $E(\{S_{\mathbf{q},\nu}\})$ smaller than the value specified for error_thresh. If this number exceeds the integer specified by niters flag, then ZG.x will stop without printing the ZG-configuration file. A general rule is: the larger the supercell, the fewer attempts are required to achieve $E(\{S_{\mathbf{q},\nu}\}) <$ error_thresh. The reason is: the order of choosing the unique set of signs, assigned to separate **q**-points, is less important as we approach the thermodynamic limit.

Now type:

```
$ head -3 ZG-configuration_0.00K.dat
```

```
    Temperature is:     0.00  K
 Atomic positions          54
   Si       0.02090173     -0.01220213     -0.01392228
```

Lines 1 and 2 give the temperature for which ZG displacements are generated and the total number of atomic coordinates, respectively. In line 3, we have the first ZG atomic coordinate. It is perfectly reasonable to find different ZG coordinates, since the modes obtained by diagonalizing the dynamical matrix can differ by a phase factor (or a unitary matrix in case of degeneracy) if the processor, or compiler, or libraries have changed. The eigenvalues, of course, should remain the same in all cases. The flag synch = .true. should apply a smooth Berry connection and align the sign of the modes with respect to a reference mode, but the sign of this reference depends on the machine. Degeneracy is not taken into account. The best way to check the validity of your configuration is by comparing the anisotropic mean-squared displacement tensor with the exact values, both obtained at the end of the **ZG_333.out**. To this aim type:

```
$ grep -12 "Anisotropic" ZG_333.out | tail -13
```

```
 Anisotropic mean-squared displacement tensor vs exact values (Ang^2):
Atom:  1
   Si    0.002341   0.002346   0.002308
        Exact values
   Si    0.002352   0.002352   0.002352
Atom:  2
   Si    0.002380   0.002329   0.002405
        Exact values
   Si    0.002352   0.002352   0.002352

 off-diagonal terms
   Si   -0.000004   0.000031  -0.000131
   Si   -0.000451   0.000287  -0.000153
```

Reducing error_thresh brings the anisotropic displacement tensor closer to the exact values. Note that in the previous versions of the code these values were multiplied by $8\pi^2$.

Now check whether the calculations have finished and type:

```
$ grep ! *scf_333*out
```

```
equil-scf_333.out:!    total energy              =    -427.83892824 Ry
ZG-scf_333_0.00K.out:!    total energy              =    -427.72102129 Ry
```

These are the total energies (Kohn-Sham potential energy surfaces) of the equilibrium and ZG structures. What is the difference between the two values and what is the reason of this difference? The difference is $\Delta E_{\rm KS} = 0.11791$ Ry and is due to the vibrational potential energy. To check this type:

```
$ grep -3 "Potent" ZG_333.out

    Total vibrational energy:    0.24020898  Ry

            Potential energy:    0.12010449  Ry

              Kinetic energy:    0.12010449  Ry
```

Our computed $\Delta E_{\rm KS}$ is indeed almost equal to half the total vibrational energy, since a DFT calculation will not account for the vibrational kinetic energy contribution. The value 0.118 Ry deviates from 0.12 Ry, since we exclude from our calculations the q-points belonging in set $\mathcal{A}$. As we use larger supercells this small deviation reduces to zero and $\Delta E_{\rm KS} =$ `Total vibrational energy`$/2$.

For the next exercises we will only need the charge-density files from `equil-si.save` and `ZG-si.save`. Thus remove:

```
$ rm -r *wfc* si.save _ph0/ equil-si.save/*wfc* ZG-si.save/*wfc*
```

Exercise 2

In this exercise we will learn how to calculate the phonon-induced band gap renormalization and temperature-dependent band structures using the example of silicon and the $3 \times 3 \times 3$ ZG supercell. To obtain temperature-dependent band structures we will employ the band structure unfolding (BSU) technique with plane-waves as basis sets. For the theory of BSU please refer to Ref. [Phys. Rev. B 85, 085201 (2012)]. To speed up the process one can skip the following two steps.
First go to the directory `exercise2` and copy the following input files in your `workdir`:

```
$  cd ../../exercise2/; mkdir workdir; cd workdir
$  cp ../inputs/Si.pz-vbc.UPF .; cp ../inputs/si.scf.in .
$  cp ../inputs/si.bands.in .; cp ../inputs/bands.in .
```

Note: in file `si.bands.in` we set `nbnd = 5` to include one empty band. We also sample the Γ-X path using 50 **k**-points in Cartesian coordinates (units of 2 pi/a).

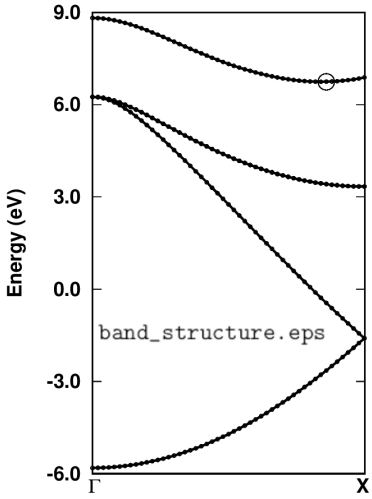
▶ Run a standard band structure calculation in the unit-cell of silicon along Γ-X using 50 **k**-points.

```
$  ibrun -np 4 $PATHQE/bin/pw.x -nk 4 < si.scf.in > si.scf.out
$  ibrun -np 4 $PATHQE/bin/pw.x -nk 4 < si.bands.in > si.bands.out
$  ibrun -np 4 $PATHQE/bin/bands.x -nk 4 < bands.in > bands.out
```

▶ Plot the band structure using the file **bands.dat.gnu**.

```
$  cp bands.dat.gnu ../gnuplot/; cd ../gnuplot/; gnuplot gp_coms.p
$  evince band_structure.eps
```

Can you find the Cartesian k-coordinates in units of \tpiba of the valence band maximum (VBM) and conduction band minimum (CBM)? Inspecting file **bands.dat**, those are:




```
             0.000000  0.000000  0.000000
-5.811     6.253     6.253     6.253     8.817
...
             0.000000  0.840000  0.000000
-2.783    -0.278     3.440     3.440     6.743
```

We can see that the VBM lies at the Γ-point and is threefold degenerated with energy $E_{\mathrm{VBM}} = 6.253$ eV. The CBM lies at $\sim$0.84 Γ-X with energy $E_{\mathrm{CBM}} = 6.743$ eV. Therefore, the DFT-LDA band gap with these settings is $E_{\mathrm{G}} = E_{\mathrm{CBM}} - E_{\mathrm{VBM}} = 0.49$ eV.

▶ Run a non self-consistent calculation for two **K**-points that define the VBM and CBM in the equilibrium and ZG supercell structures.

Note: The selection rule here is to find the reciprocal lattice vector of the supercell $\mathbf{G}$ that maps **K** onto **k**, i.e. $\mathbf{K} = \mathbf{k} - \mathbf{G}$. The trivial choice is $\mathbf{G} = \mathbf{\Gamma}$ and thus set $\mathbf{K} = \mathbf{k}$.

To prepare the calculation proceed as follows:

```
$  cd ../workdir/
$ cp ../../exercise1/workdir/equil-scf_333.in equil-nscf_333.in
$ cp ../../exercise1/workdir/ZG-scf_333_0.00K.in ZG-nscf_333_0.00K.in
$ cp -r ../../exercise1/workdir/equil-si.save/ .
$ cp -r ../../exercise1/workdir/ZG-si.save/ .
```

In `equil-nscf_333.in` and `ZG-nscf_333_0.00K.in` apply the following changes:

1. Set `calculation = 'nscf'`
2. Include one empty band / unit-cell by setting `nbnd = 135` below `ecutwfc = 20.0` flag.
3. Replace the automatic **K**-grid:
   ```
   K_POINTS automatic
      2   2   2   0   0   0
   ```
   with the two **K**-points of the VBM and CBM:
   ```
   K_POINTS crystal
   2
   0.000000 0.000000 0.000000 1
   0.000000 1.260000 1.260000 1
   ```

Remark: Since the coordinates are given in units of the reciprocal lattice parameters, we obtain the K-coordinates by multiplying the k-coordinates of VBM and CBM states with the dimensions of the supercell, i.e. if $\mathbf{k} = [x \;\; y \;\; z]$ then $\mathbf{K} = [m \times x \quad n \times y \quad p \times z]$, where integers m, n, p define an $m \times n \times p$ supercell.

4. For ZG input file add `nosym = .true.` below `nbnd = 135`.

If you did not complete `exercise1` and the steps above, then copy files from `inputs`:

```
$ cd ../workdir/; cp ../inputs/*nscf_333*in .; cp -r ../inputs/*-si.save/ .
```

For example the ZG input file should look like this:

ZG-nscf_333_0.00K.in

```
 &control
  calculation = 'nscf'
  restart_mode = 'from_scratch'
  prefix = 'ZG-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
  nbnd = 135
  nosym = .true.
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
K_POINTS crystal
2
0.000000 0.000000 0.000000 1
0.000000 1.260000 1.260000 1
CELL_PARAMETERS (angstrom)
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS (angstrom)
...
```

```
$  ibrun -n 28 $PATHQE/bin/pw.x -nk 2 < equil-nscf_333.in > equil-nscf_333.out
$  ibrun -n 28 $PATHQE/bin/pw.x -nk 2 < ZG-nscf_333_0.00K.in > ZG-nscf_333_0.00K.out
```

The calculations should take around 5 secs each. When the equilibrium calculation is finished, check that you obtain the VBM and CBM energies you would expect. To this aim type:

```
$ grep highest equil-nscf_333.out
```

```
highest occupied, lowest unoccupied level (ev):     6.2534    6.7435
```

Indeed we obtain the correct VBM and CBM energies. To find the band gap renormalization from the ZG calculation type:

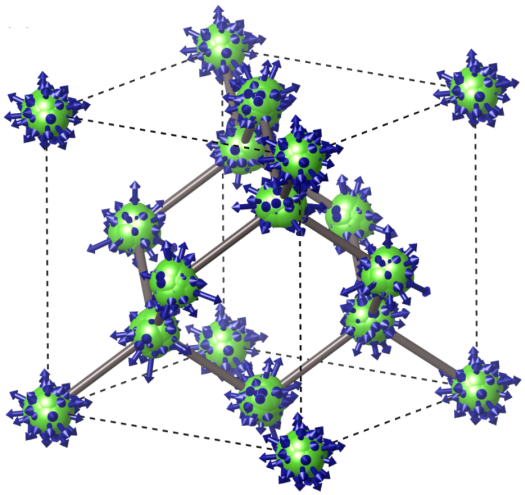
```
$ grep -41 "End of band" ZG-nscf_333_0.00K.out | tail -40

          k = 0.0000 0.0000 0.0000 ( 10849 PWs)   bands (ev):

     ...
     1.1932   1.2366   1.2583   1.2797   1.3641   1.6000   1.6439   1.6632
     1.6700   1.7080   1.7218   1.7378   1.7537   1.7688   1.7839   1.8022
     1.8550   3.6342   3.6870   3.7151   3.7234   3.7284   3.7521   3.7739
     3.7802   3.8081   3.8160   3.8230   3.8692   3.9984   4.0246   4.0419
     4.0451   4.0825   4.0963   4.1372   4.1436   4.1508   4.1664   4.1930
     4.2331   5.0751   5.0903   5.1134   5.1374   5.1660   5.1758   5.2101
     5.2319   5.2590   5.2890   5.3002   5.3327   5.3569   5.3802   5.4029
     5.4291   6.2804   6.2837   6.2884   6.8028   6.8362   6.8761   6.8923
     6.9465   6.9683   7.5885   7.6262   7.6562   7.6802   7.7109   7.7141
     7.7244   7.7621   7.7816   7.7993   7.8251   7.8639   7.9448   7.9779
     7.9945   8.0130   8.0295   8.0543   8.0565   8.0700   8.1321

          k = 0.0000 1.7819 0.0000 ( 10874 PWs)   bands (ev):

     ...
     1.7498   1.7835   1.7983   1.8350   1.8716   1.8784   1.9231   1.9384
     1.9980   2.0134   2.0382   2.0769   2.2691   2.3100   2.3364   2.3594
     2.8271   3.2212   3.2614   3.2838   3.3084   3.3388   3.3709   3.3925
     3.4312   3.4557   3.4624   3.7524   3.7692   3.7872   3.7980   4.0479
     4.0803   4.0988   4.1234   4.1534   4.1894   4.2079   4.2468   4.3466
     4.3542   4.4367   4.4731   4.4850   4.5385   4.5837   4.6246   4.6715
     4.7076   4.7895   4.8152   4.8539   4.8914   5.4894   5.5343   5.5658
     5.6293   5.6546   5.8297   5.8540   6.7226   7.2637   7.3315   7.5498
     7.5821   7.6203   7.6646   7.8126   7.8455   7.8900   7.9016   7.9375
     7.9760   7.9943   8.0392   8.3543   8.3635   8.3798   8.4039   8.4827
     8.5919   8.6417   8.6680   8.6851   8.7069   8.7359   8.7577
```

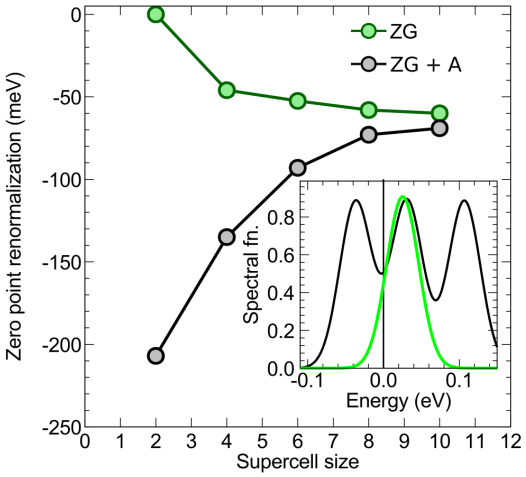
Can you spot the energy of the VBM and CBM? We can see that there is a splitting of the originally threefold degenerated VBM of the order: $\Delta E = 6.2884 - 6.2804 = 8$ meV. This degeneracy breaking results from the ZG displacements in a finite size supercell and should reduce to zero for larger ZG simulation cells. This is a consequence of the harmonic approximation, where the symmetries of the system should be maintained upon thermal averaging (see thermal ellipsoids in the plot below).

We note that to minimize the effect of band degeneracy splitting we exclude the modes in set $\mathcal{A}$. In Ref. [Phys. Rev. Res. 2, 013357 (2020)] we show that these modes dominate VBM splitting (see the spectral function in the figure below), giving $\Delta E > 150$ meV for a $4 \times 4 \times 4$ ZG supercell. Hence, for finite size systems with degenerated band edges, it is suggested to keep `incl_qA = .false.`.

To deal with this finite size supercell artefact, we determine the VBM by taking the average of the three bands: $E_{\rm VBM} = (6.2804 + 6.2837 + 6.2884)/3 = 6.2842$ eV. Thus the band gap at 0K is $E(0\,{\rm K}) = 6.7226 - 6.2842 = 0.4384$ eV, and we can determine a zero-point renormalization $E_{\rm ZPR} = 0.49 - 0.4384 \simeq 52$ meV. This value is in good agreement with literature data (check, for example, Refs. [J. Chem. Phys. 143, 102813 (2015)] and [New J. Phys. 20, 123008 (2018)]).

Note: in general, for obtaining reliable values of the zero-point renormalization, you should always check convergence with respect to the ZG supercell size (see the right part of the figure below).





Before going to the next step remove unnecessary files:

```
$ rm -r *wfc* *save
```

▶ Prepare a band structure unfolding calculation using the $3 \times 3 \times 3$ ZG configuration. Create a new directory and copy the necessary input files:

```
$ mkdir band_structure_unfolding/; cd band_structure_unfolding/
$ cp ../../inputs/bands_unfold.in bands.in; cp ../../inputs/Si.pz-vbc.UPF .
$ cp -r ../../../exercise1/workdir/ZG-si.save/ .
$ cp ../ZG-nscf_333_0.00K.in ZG-bands_333_0.00K.in
```

In `ZG-bands_333_0.00K.in` apply the following modifications:

1. Set `calculation = 'bands'`
2. Replace the following lines containing information about the **K**-points:

   ```
   K_POINTS crystal
   2
   0.000000 0.000000 0.000000 1
   0.000000 1.260000 1.260000 1
   ```

   with

   ```
   K_POINTS crystal_b
   2
   0.000000 0.000000 0.000000 35
   0.000000 1.500000 1.500000 1
   ```

Note: Here we choose to sample the Γ-X path with 36 equally-spaced **K**-points.

If you did not complete the previous step, then copy:

```
$ cp ../../inputs/ZG-bands_333_0.00K.in .
```

The file `ZG-bands_333_0.00K.in` should look like this:

```
&control                                                    ZG-bands_333_0.00K.in
  calculation = 'bands'
  restart_mode = 'from_scratch'
  prefix = 'ZG-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
  nbnd = 135
  nosym = .true.
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
K_POINTS crystal_b
2
0.000000 0.000000 0.000000 35
0.000000 1.500000 1.500000 1
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS {angstrom}
...
```

We also show here the file `bands.in`:

```
--                                                                  bands.in
&bands
 prefix  = 'ZG-si'
 outdir  = './'
 filband = 'bands.dat'
 lsym    = .false.
 dim1    = 3,
 dim2    = 3,
 dim3    = 3
/
```

Note: The input variables in `bands.in` are the same as in a standard `bands.x` calculation. The only difference is the flags `dimX` used to specify the dimensions of the ZG supercell.

▶ Run a band structure unfolding calculation.

```
$  ibrun -n 56 $PATHQE/bin/pw.x -nk 28 < ZG-bands_333_0.00K.in > ZG-bands_333_0.00K.out
$  ibrun -n 28 $PATHQE/bin/bands_unfold.x < bands.in > bands.out
```

Note: Parallelization over **k**-points in `bands_unfold.x` is not implemented.

The bands calculation should take around 50 secs and the unfolding around 1 sec. Now read the outputs of **bands01.dat** and **spectral_weights01.dat**. Can you guess what's the meaning of the spectral weights for each band?

```
$  head -16 bands01.dat
```

```
 &plot nbnd= 135, nks=    36 /
           0.000000  0.000000  0.000000
    -5.835716     -4.569846     -4.546625     -4.475946     -4.455311    ...
    -3.925173     -3.895883     -3.875348     -3.798917     -3.770732    ...
    -2.500872     -2.445933     -2.433439     -2.400825     -2.380339    ...
    -0.601159     -0.556482     -0.554254     -0.515123     -0.510728    ...
     0.778942      0.808380      0.838416      0.865686      0.871446    ...
     1.258270      1.279673      1.364076      1.599987      1.643945    ...
     ...
```

```
$ head -16 spectral_weights01.dat
```

```
 &plot nbnd= 135, nks=    36 /
           0.000000  0.000000  0.000000
   0.989535    0.000643    0.000285    0.000371    0.000089   ...
   0.000601    0.000920    0.000267    0.000044    0.000320   ...
   0.000209    0.000124    0.000160    0.000085    0.000503   ...
   0.000212    0.000211    0.000098    0.000116    0.000302   ...
   0.000173    0.000148    0.000198    0.000304    0.000234   ...
   0.000214    0.000364    0.000205    0.000336    0.000534   ...
   ...
```

Can you spot the weights corresponding to the VBM? What would the weights be if we use the equilibrium $3 \times 3 \times 3$ supercell? In this case, one should obtain the perfect unfolding reproducing the unit cell band structure. The data in the files **bands01.dat** and **spectral_weights01.dat**, represent the band energies, $\varepsilon_{m\mathbf{K}}(T)$, and spectral weights, $P_{m\mathbf{K},\mathbf{k}}(T)$, for all **K**-points, respectively.
Now you have all the ingredients to evaluate the spectral function given by:

$$A_{\mathbf{k}}(\varepsilon, T) = \sum_{m\mathbf{K}} P_{m\mathbf{K},\mathbf{k}}(T)\delta[\varepsilon - \varepsilon_{m\mathbf{K}}(T)] \tag{1}$$

▶ Calculate the spectral function using the energies and spectral weights. To this aim convert first **bands01.dat** and **spectral_weights01.dat** into a different format using `plotband.x` of QE:

```
$ cp ../../inputs/energies.in .
$ cp ../../inputs/spectral_weights.in .
$ $PATHQE/bin/plotband.x spectral_weights01.dat < spectral_weights.in > spw.out
$ $PATHQE/bin/plotband.x bands01.dat < energies.in > energies.out
$ sed -i '/^\s*$/d' spectral_weights.dat # remove empty lines
$ sed -i '/^\s*$/d' energies.dat # remove empty lines
$ paste energies.dat spectral_weights.dat > tmp
$ awk '{print $1,$2,$4}' tmp > energies_weights.dat; rm tmp
```

You have just generated **energies_weights.dat** file in a similar format to a ".gnu" file where the first column has the momentum-axis values, the second the energies, and the third the spectral weights. Now proceed with the calculation of the spectral function using `pp_spctrlfn.x`:

```
$ cp ../../inputs/energies_weights.dat . # if you missed the step above
$ cp ../../inputs/pp_spctrlfn.in .
$ ibrun -n 4 $PATHQE/bin/pp_spctrlfn.x -nk 4 < pp_spctrlfn.in > pp_spctrlfn.out
```

The file pp_spctrlfn.in takes the following input variables:

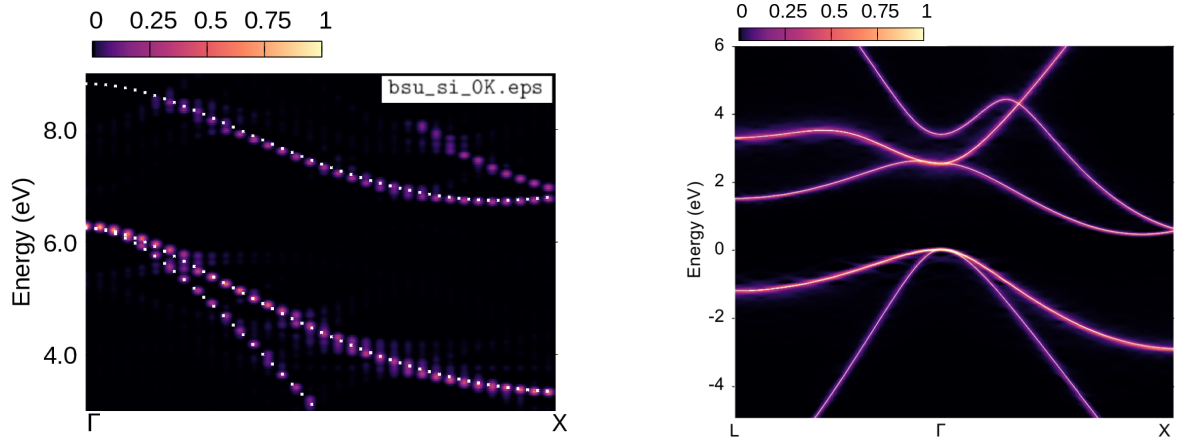
```
--                                                                    pp_spctrlfn.in
&input
  flin = 'energies_weights.dat'
  steps = 4860,
  ksteps = 200,
  esteps = 200,
  kmin = 0,
  kmax = 0.7071,
  emin =  3.00
  emax = 9.00
  flspfn = 'spectral_function.dat'
/
```

Here, (i) steps is the number of rows in the input file flin, (ii) ksteps and esteps define the resolution of the spectral function along the momentum-axis and energy-axis, (iii) (kmin - kmax) and (emin - emax) define the momentum and energy windows, and (iv) in flspfn we provide the name of the output file.

Now plot the spectral function:

```
$ cp spectral_function.dat ../../gnuplot/. ; cd ../../gnuplot/
$ cp ../inputs/bands.dat.gnu .
$ gnuplot gp_pm3d.p; evince bsu_si_0K.eps
```

We also copied bands.dat.gnu for comparing the spectral function with the band structure calculated with the atoms at their equilibrium positions (white lines in the plots below). In the right part of the figure below, we also reproduced the converged calculation from Ref. [Phys. Rev. Res. 2, 013357 (2020)], employing an $8 \times 8 \times 8$ ZG supercell, more **K**-points and empty bands.



In **APPENDIX / Exercise 2** we provide a recipe for performing band structure unfolding for each **K**-point separately. This approach is most effective when larger supercells are employed and pw.x cannot handle all **K**-points in one run.

Exercise 3

In this exercise we will learn how to calculate the phonon-assisted optical spectra using the example of silicon and the $3 \times 3 \times 3$ ZG supercell for $T = 0$ K. To obtain temperature-dependent optical spectra we will evaluate the optical matrix elements in the independent-particle approximation using a routine very similar to `epsilon.x` of QE. We emphasize that this methodology can be extended to higher-level theories of optical absorption, like RPA, BSE, etc ... The present approach of calculating phonon-assisted optical spectra is based on Ref. [Phys. Rev. B 94, 075125 (2016)].

Note: The calculation of phonon-assisted optical spectra using ZG configurations has some drawbacks and advantages when compared to the perturbative methodology implemented in `EPW`.

First go to the directory `exercise3`, create your working directories and copy the input files:

```
$  cd ../../exercise3/; mkdir workdir; cd workdir; mkdir equil; mkdir ZG
$   cd equil
$  cp ../../inputs/equil/K_list.in .
$  cp ../../inputs/equil/epsilon.in .
$  cp -r ../../../exercise1/workdir/equil-si.save/ .
$  cp ../../../exercise2/workdir/equil-nscf_333.in .
```

The file `K_list.in` contains the crystal coordinates of 200 randomly generated **K**-points We use random **K**-points, instead of a uniform grid, in order to speed up convergence. In `equil-nscf_333.in` apply the following changes:

1. Add `nosym = .true.` below `nbnd = 135`.
   Note: although we use the equilibrium structure we set `nosym = .true.` since random **K**-points are employed.

2. Delete the information for the **K**-points:

   ```
   K_POINTS crystal
   2
   0.000000 0.000000 0.000000 1
   0.000000 1.260000 1.260000 1
   ```

   and at the **bottom** of the file add the following:

   ```
   K_POINTS crystal
   30
   ```

3. Copy and paste the first 30 **K**-points from file `K_list.in`.

If you did not complete Exercises 1 and 2, you can copy:

```
$  cp ../../inputs/equil/equil-nscf_333.in .
$  cp -r ../../../exercise2/inputs/equil-si.save/ .
```

The file `equil-nscf_333.in` should look like this:

```
&control                                                    equil-nscf_333.in
  calculation = 'nscf'
  restart_mode = 'from_scratch'
  prefix = 'equil-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
```

```
  ecutwfc =  20.00
  nbnd = 135
  nosym = .true.
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS {angstrom}
   Si     0.00000000   0.00000000   0.00000000
   Si    -1.34940189   1.34940189   1.34940189
   Si    -2.69880378   0.00000000   2.69880378
...
K_POINTS crystal
30
0.236650 0.022946 0.917156 1.0
0.392728 0.655175 0.471114 1.0
...
```

We also show here the file `epsilon.in`:

```
--                                                                    epsilon.in
&inputpp
outdir      = './',
prefix      = 'equil-si',
calculation = 'eps'
/
&energy_grid
smeartype   = 'gauss',
intersmear  = 0.03d0,
wmax        = 4.5d0,
wmin        = 0.2d0,
nw          = 600,
shift       = 0.0d0,
/
```

Note: The input variables in `epsilon.in` are the same as in a standard `epsilon.x` calculation.

▶ Run an optical spectrum calculation for silicon using the $3 \times 3 \times 3$ equilibrium supercell and 30 random **K**-points. The aim is to calculate the imaginary part of the dielectric function as:

$$\epsilon_2(\omega) = \frac{2\pi}{m_{\rm e} N_{\rm e}} \frac{1}{N_{\mathbf{K}}} \frac{\omega_{\rm p}^2}{\omega^2} \sum_{cv\mathbf{K}} |p_{cv\mathbf{K}}|^2 \delta(\varepsilon_{c\mathbf{K}} - \varepsilon_{v\mathbf{K}} - \hbar\omega), \tag{2}$$

where $m_{\rm e}$ is the electron mass, $N_{\rm e}$ is the number of electrons in the crystal unit cell, $\omega_{\rm p}$ is the plasma frequency, $N_{\mathbf{K}}$ is the number of **K**-points, and ω the photon frequency. The sum extends over the **K**-points, the occupied states of energy $\varepsilon_{v\mathbf{K}}$, and the unoccupied states of energy $\varepsilon_{c\mathbf{K}}$. $p_{cv\mathbf{K}}$ denotes the matrix elements of the momentum operator along a particular polarization direction. To calculate the dielectric function we use `epsilon_Gaus.x`, which is a slightly modified routine of `epsilon.x`, that replaces the delta function with a Gaussian instead of a Lorentzian function. We choose to apply Gaussian broadening in order to minimize numerical artefacts at the absorption onset.

```
$  ibrun -n 56 $PATHQE/bin/pw.x -nk 28 < equil-nscf_333.in > equil-nscf_333.out
$  ibrun -n 28 $PATHQE/bin/epsilon_Gaus.x < epsilon.in > epsilon.out
```

The full calculation should take around 40 secs. The output files are (i) `epsi_equil-si.dat` containing $\epsilon_2(\omega)$, (ii) `epsr_equil-si.dat` containing the real part of the dielectric function $\epsilon_1(\omega)$, and (iii) containing the electron energy loss spectrum `eels_equil-si.dat`; all calculated for the equilibrium structure. In each file the first column is the energy grid, and the rest three represent the observable along the three Cartesian directions.

▶ Take the isotropic average of the dielectric function and copy the output file in the `gnuplot` directory using:

```
$  awk '{print $1,($2+$3+$4)/3}' epsi_equil-si.dat > epsi_si_333_equil_30_av.dat
$  cp epsi_si_333_equil_30_av.dat ../../gnuplot/.
```

▶ Run an optical spectra calculation for silicon using the $3 \times 3 \times 3$ ZG supercell and 30 random **K**-points. We will essentially repeat all steps performed for calculating the optical spectra of the $3 \times 3 \times 3$ equilibrium supercell. First go to the `ZG` directory and copy the input files:

```
$  cd ../ZG/
$ cp ../../inputs/ZG/K_list.in .
$ cp ../../inputs/ZG/epsilon.in .
$ cp -r ../../../exercise1/workdir/ZG-si.save/ .
$ cp ../../../exercise2/workdir/ZG-nscf_333_0.00K.in .
```

In `ZG-nscf_333_0.00K.in` apply the following change:

1. Delete the information for the **K**-points:

   ```
   K_POINTS crystal
   2
   0.000000 0.000000 0.000000 1
   0.000000 1.260000 1.260000 1
   ```

   and at the **bottom** of the file add the following:

   ```
   K_POINTS crystal
   30
   ```

2. Copy and paste the first 30 **K**-points from file `K_list.in`.

If you did not complete Exercises 1 and 2, copy `ZG-si.save` and `ZG-nscf_333_0.00K.in` from:

```
$  cp ../../inputs/ZG/ZG-nscf_333_0.00K.in .
$  cp -r ../../../exercise2/inputs/ZG-si.save/ .
```

The file `ZG-nscf_333_0.00K.in` should look like this:

```
&control                                                    ZG-nscf_333_0.00K.in
  calculation = 'nscf'
  restart_mode = 'from_scratch'
  prefix = 'ZG-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
  nbnd = 135
  nosym = .true.
/
&electrons
```

```
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS {angstrom}
   Si       0.02090173     -0.01220213     -0.01392228
   Si      -1.39812907      1.32606198      1.38415986
   Si      -2.68002772      0.00399202      2.64203512
...
K_POINTS crystal
30
  0.236650 0.022946 0.917156 1.0
  0.392728 0.655175 0.471114 1.0
...
```

epsilon.in is the same with the one used before, but with `prefix = 'ZG-si'`.

▶ Now run the following to calculate $\epsilon_2(\omega)$ at 0.00 K:

```
$  ibrun -n 56 $PATHQE/bin/pw.x -nk 28 < ZG-nscf_333_0.00K.in > ZG-nscf_333_0.00K.out
$  ibrun -n 28 $PATHQE/bin/epsilon_Gaus.x < epsilon.in > epsilon.out
```

The full calculation should take around 40 secs. The output files are (i) `epsi_ZG-si.dat` containing $\epsilon_2(\omega)$, (ii) `epsr_ZG-si.dat` containing the real part of the dielectric function $\epsilon_1(\omega)$, and (iii) containing the electron energy loss spectrum `eels_ZG-si.dat`; all calculated for $T = 0.00$ K. In each file the first column is the energy grid, and the rest three represent the temperature-dependent observable along the three Cartesian directions.

▶ Take the isotropic average of the dielectric function and copy the output file in the `gnuplot` directory using:

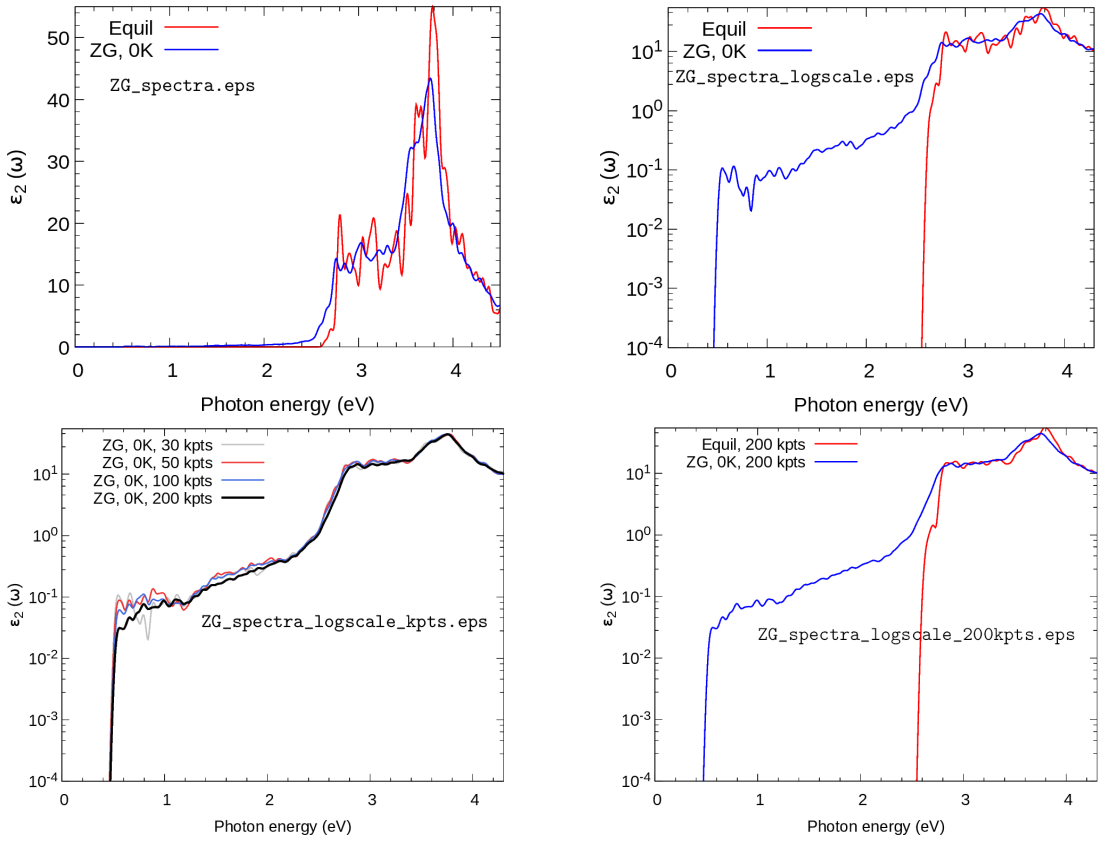
```
$  awk '{print $1,($2+$3+$4)/3}' epsi_ZG-si.dat > epsi_si_333_ZG_30_av.dat
$  cp epsi_si_333_ZG_30_av.dat ../../gnuplot/.
```

The output file containing the data is **epsi_si_333_ZG_30.dat**.

Now copy the output file in `gnuplot` directory and plot ZG and equilibrium spectra using:

```
$  cp epsi_si_333_ZG_30_av.dat ../../gnuplot/.
$  cd ../../gnuplot/; gnuplot gp_coms.p; evince ZG_spectra.eps
$  gnuplot gp_coms_logscale.p; evince ZG_spectra_logscale.eps
```

We also plot $\varepsilon_2(\omega)$ as calculated with more **K**-points to obtain convergence.



What are the differences between the ZG and equilibrium spectra?
Are phonon-assisted transitions captured as expected?

Ref. [Phys. Rev. B 94, 075125 (2016)] reports the convergence of the spectra with respect to the ZG supercell size and the agreement with experimental data for the absorption coefficient.

In **APPENDIX / Exercise 3** we provide a recipe for calculating optical spectra for each **K**-point separately. This approach is most effective when larger supercells are employed and `pw.x` cannot handle all **K**-points at once.

Exercise 4

In this exercise we will discuss the physical meaning of the ZG configuration and learn how to calculate phonon-induced diffuse (or inelastic) scattering patterns using graphene. This is an example of an observable which allows us to computationally reach the thermodynamic limit and assess multi-phonon processes. More details about the theory of phonon-induced inelastic scattering and its connection to the special displacement method can be found in Ref. [Phys. Rev. B 104, 205109 (2021)].

We will evaluate the exact expression for the all-phonon inelastic scattering intensity and compare it with the ZG scattering intensity, i.e. when the scatterers (nuclei) are defined by the ZG displacements. Go to the directory `exercise4`, create your working directories, and copy the input files:

```
$  cd ../../exercise4/; mkdir workdir; cd workdir; mkdir exact; mkdir ZG
$  cd  exact
$  cp ../../inputs/exact/graphene.881.fc .
$  cp ../../inputs/exact/disca.in .
```

Note: The file `graphene.881.fc` contains the IFC of graphene calculated using an $8 \times 8 \times 1$ **q**-grid.

▶ Run a diffuse scattering calculation using `disca.x` and the input:

```
--                                                                                    disca.in
&input
    asr = 'crystal', amass(1) = 12.011, flfrc = 'graphene.881.fc', atm_zg(1) = 'C',
    dim1 = 40, dim2 = 40, dim3 = 1,
    T = 300,
    atmsf_a(1,1) = 0.1361,
    atmsf_a(1,2) = 0.5482,
    atmsf_a(1,3) = 1.2266,
    atmsf_a(1,4) = 0.5971,
    atmsf_a(1,5) = 0.0,
    atmsf_b(1,1) = 0.3731,
    atmsf_b(1,2) = 3.2814,
    atmsf_b(1,3) = 13.0456,
    atmsf_b(1,4) = 41.0202,
    atmsf_b(1,5) = 0.0,
    zero_one_phonon = .true., full_phonon = .true.
    nks1 = 0,  nks2 = 0,  nks3 = 0, nksf1 = 5, nksf2 = 5, nksf3 = 1,
    plane_val = 0.d0, plane_dir = 3
    qstart = 1, qfinal = 40000,
/
&pp_disca
    nrots = 6,
    kres1 = 250,
    kres2 = 250,
    kmin  = -10,
    kmax  = 10,
    col1  = 1,
    col2  = 2,
    Np    = 1600
/
```

```
$ ibrun -np 56 $PATHQE/bin/disca.x -nk 56 < disca.in > disca.out
```

The input format of `disca.in` is the same as `ZG_333.in`. Line 3 contains input variables similar to those in matdyn.in; in line 4 we provide the dimensions `dimX` of the **q**-grid used to sample the Brillouin zone; in line 5 we provide the temperature `T` in Kelvin; lines 6-15 contain the parameters of the atomic scattering factor (`atmsf_a` and `atmsf_b`) from Ref. [Micron 30, 625–648, (1999)]; in line 16 we specify whether we want to compute the one-phonon (`zero_one_phonon`) and/or all-phonon

(full_phonon) structure factor; in line 17 nksX and nksfX define the range of reciprocal lattice vectors (in crystal coordinates) which are used to determine the extent of the calculated scattering vectors (**Q**-points); in line 18 we provide plane_val that defines the plane along which the structure factor is calculated (in units of 2pi/alat) and plane_dir defines the Cartesian direction perpendicular to the plane (where 1 --> x, 2 --> y, and 3 --> z); and in line 19 we specify the range of **Q**-points (from qstart to qfinal) for which the structure factor is calculated.

In the namelist &pp_disca we have: (i) nrots which is the number of rotations to be applied on the all-phonon scattering intensity to obtain the complete map. nrots is based on the space group of each system. This step is important to avoid the extra effort of calculating the scattering intensity for wavevectors connected by rotation symmetry (see also plots at the end of the exercise). (ii) kres1 and kres2 define the resolution of the scattering intensity along the chosen Cartesian axes, (iii) col1 and col2 are integers defining two of the Cartesian directions of the scattering vectors (where 1 --> x, 2 --> y, and 3 --> z), (iv) (kmin - kmax) define the 2D momentum window in reciprocal space, and (v) Np is the number of reduced wavevectors (**q**-points) used to sample each Brillouin zone.

The standard outputs from a disca.x run are: **strf_one-ph_broad.dat** and **strf_all-ph_broad.dat** containing the zero+one-phonon, and all-phonon scattering intensities, respectively, after convolution is applied. These files have three columns: the first two represent the Cartesian coordinates of the scattering vectors and the third column is the scattering intensity.

One can also print for each case the mode resolved and atom resolved scattering intensities by setting mode_resolved = .true. and/or atom_resolved = .true..

For printing the raw data, i.e. before convolution is applied, one can use print_raw_data = .true. to obtain the following data: **Bragg_scattering.dat**, **strf_one-phonon.dat**, **strf_q_nu_one-phonon.dat**, and **strf_all-phonon.dat**, containing the Bragg, mode (or branch) resolved, one-phonon, and all-phonon scattering intensities, respectively. These files have four columns: the first three represent the Cartesian coordinates of the scattering vectors (Q_x, Q_y, and Q_z) and the fourth column is the scattering intensity. disca.x also prints the **Q** vectors in crystal coordinates in **qpts_strf.dat** (to be used for computing the ZG scattering intensity).

Now copy the output file strf_all-ph_broad.dat in the gnuplot directory:

```
$ cp strf_all-ph_broad.dat ../../gnuplot/exact/
```

▶ Run a ZG.x calculation that allows to compute the structure factor. Go to the ZG directory and copy the input files:

```
$ cd ../ZG; cp ../../inputs/ZG/graphene.881.fc .; cp ../../inputs/ZG/ZG_strf.in .
$ cp ../exact/qpts_strf.dat .
```

Note: we also copied qpts_strf.dat containing the **Q**-points generated by disca.x.

The ZG_strf.in should look like this:

```
--                                                                                                    ZG_strf.in
&input
    asr='crystal', amass(1)= 12.011, flfrc= 'graphene.881.fc', atm_zg(1) = 'C'
    dim1 = 40, dim2 = 40, dim3 = 1,
    T = 300,
    synch = .true., compute_error = .false.
    incl_qA = .true.,
    ZG_strf = .true.
/
&strf_ZG
    qpts_strf = 40000
    atmsf_a(1,1) = 0.1361, atmsf_a(1,2) = 0.5482, atmsf_a(1,3) = 1.2266,  atmsf_a(1,4) = 0.5971,
    atmsf_a(1,5) = 0.0,
    atmsf_b(1,1) = 0.3731, atmsf_b(1,2) = 3.2814, atmsf_b(1,3) = 13.0456, atmsf_b(1,4) = 41.0202,
    atmsf_b(1,5) = 0.0,
    nrots = 6,
    kres1 = 250,
    kres2 = 250,
    kmin  = -10,
    kmax  = 10,
    col1  = 1,
    col2  = 2,
    Np    = 1600
/
```

```
$ ibrun -np 56 $PATHQE/bin/ZG.x -nk 56 < ZG_strf.in > ZG_strf.out
```

The input variables in the first 3 lines of ZG_strf.in are the same as in disca.in (but strictly speaking dimX, here, is for the supercell size dimensions); in line 4 we ask the code to synchronize the modes (synch = .true.), but not to compute the minimization function $E(\{S_{\mathbf{q},\nu}\})$, as we have seen in **Exercise 1** (compute_error = .false.). We made this choice to demonstrate that the order of choosing the unique set of signs is not important as we approach the thermodynamic limit; in line 5 we make the choice to include $\mathbf{q}$-points in set $\mathcal{A}$ by setting incl_qA = .true. (at the thermodynamic limit should make no difference); in line 6 we ask the code to compute the structure factor (ZG_strf = .true.).

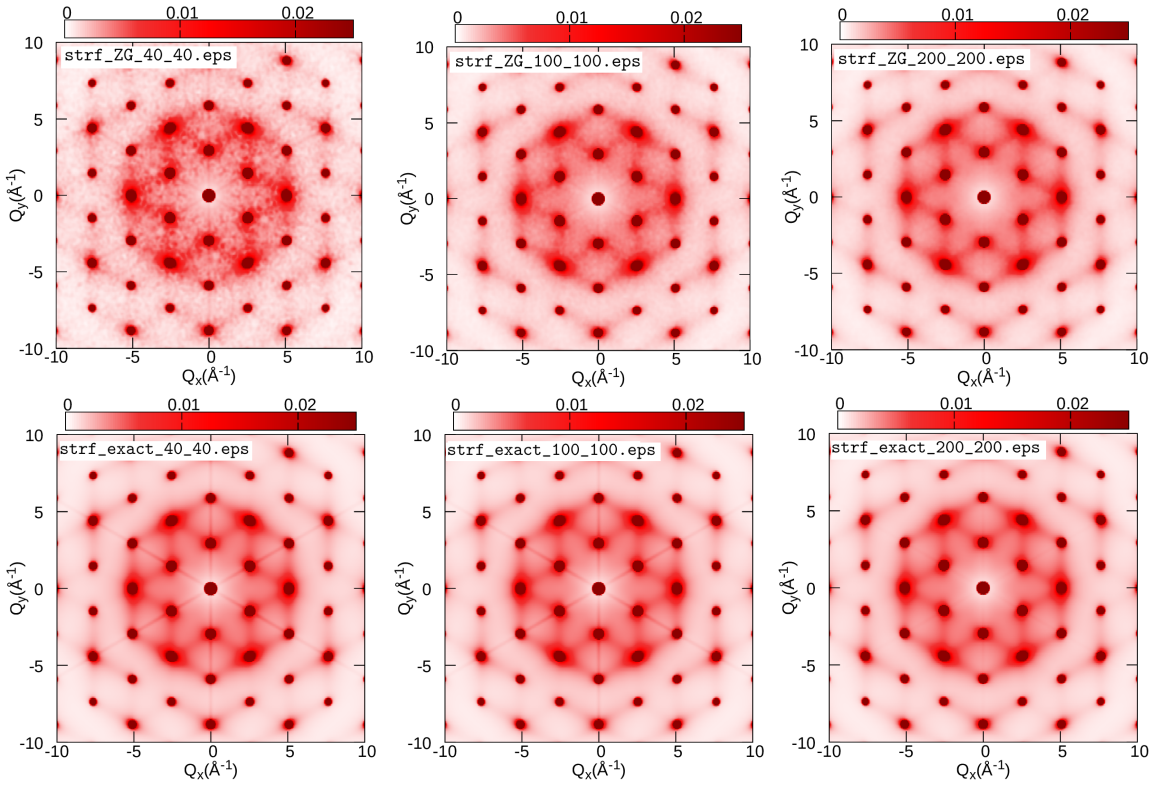
The namelist &strf_ZG is as &pp_disca seen before, but here we include the parameters of the atomic scattering factor (atmsf_a and atmsf_b). We also ask the code to compute the map for 40000 **Q**-points (qpts_strf = 40000) from file qpts_strf.dat.

The calculation should take around 10 secs. The important outputs from this ZG.x run are: **ZG-configuration_300.00K.dat** and **strf_ZG_broad.dat**, containing the collection of ZG atomic coordinates (as before) and the associated scattering intensity, respectively. File **structure_factor_ZG_raw.dat** contains the raw data and has four columns: the first three represent the Cartesian coordinates of the scattering vectors and the fourth column is the scattering intensity.

Now copy the output file strf_ZG_broad.dat in the gnuplot directory and plot your ZG and exact data:

```
$  cp strf_ZG_broad.dat ../../gnuplot/ZG
$  cd ../../gnuplot/ZG; gnuplot gp_pm3d.p; evince strf_ZG_40_40.eps
$  cd ../exact; gnuplot gp_pm3d.p; evince strf_exact_40_40.eps
```

Your results should look like the plots below (left panel). We also add plots of the exact and ZG all-phonon structure maps using a 100×100 and 200×200 **q**-grid for Brillouin sampling What happens to the ZG structure factor map as we approach the thermodynamic limit? Indeed, the choice of the order of signs is no longer important and the function $E(\{S_{\mathbf{q},\nu}\})$ is minimized at the thermodynamic limit. Our calculations here also demonstrate the physical significance of the ZG configuration as the best collection of scatterers that reproduce the phonon-induced inelastic scattering.



Exercise 5

In this exercise we will learn how to perform phonon unfolding. This technique is useful to identify the effect on the phonons in the fundamental Brillouin zone when supercells are employed to accommodate disorder, defects, or any other perturbation that breaks the translational symmetry of the lattice. First, as a proof of concept, we will perform the perfect unfolding of the phonons calculated for a 2×2×2 silicon supercell; then we will show how to calculate the effect on the phonons when a 2×2×2 silicon supercell is doped by a phosphorous (P) atom.

▶ Go to `exercise5`, create `workdir`, and calculate the phonon dispersion of silicon in the unit cell using a 2×2×2 homogeneous $\mathbf{q}$-grid:

```
$  cd workdir; cp -r ../inputs/unit_cell_222_grid/ .; cd unit_cell_222_grid/
$ ibrun -np 4 $PATHQE/bin/pw.x -nk 4 < si.scf.in > si.scf.out
$ ibrun -np 4 $PATHQE/bin/ph.x -nk 4 < si.ph.in > si.ph.out
$ ibrun -np 1 $PATHQE/bin/q2r.x < q2r.in > q2r.out
$ ibrun -np 1 $PATHQE/bin/matdyn.x < matdyn.in > matdyn.out
```

Input files are provided here:

si.scf.in

```
--
 &control
    calculation = 'scf'
    restart_mode = 'from_scratch',
    prefix = 'si',
    pseudo_dir = './',
    outdir='./'
 /
 &system
    ibrav = 2,
    celldm(1) = 10.20,
    nat =  2,
    ntyp = 1,
    ecutwfc = 30.0
 /
 &electrons
    diagonalization='david'
    mixing_mode = 'plain'
    mixing_beta = 0.7
    conv_thr = 1.0d-7
 /
ATOMIC_SPECIES
 Si  28.086  Si.pz-vbc.UPF
K_POINTS automatic
6 6 6 0 0 0
ATOMIC_POSITIONS {crystal}
 Si 0.00 0.00 0.00
 Si 0.25 0.25 0.25
/
```

si.ph.in

```
--
Phonons of Silicon
&inputph
   amass(1)= 28.0855,
   prefix = 'si'
   outdir = './'
   ldisp = .true.
   fildyn = 'si.dyn'
```

```
   tr2_ph = 1.0d-12
   nq1 = 2
   nq2 = 2
   nq3 = 2
/
```

```
--                                                                  q2r.in
&input
 fildyn='si.dyn', flfrc = 'si.222.fc'
/
```

```
--                                                                  matdyn.in
&input
    asr='simple', amass(1)=28.0855,
    flfrc='si.222.fc', fldyn='si.dyn.mat',
    flfrq='si.freq', fleig='si.dyn.eig', q_in_cryst_coord = .false.
    q_in_band_form = .true.
/
 9
 0         0         0      100
 0.75      0.75      0.0    1
 0.2500    1.0000    0.2500 100
 0.000     1         0.000  100
 0         0         0      100
 0.5000    0.5000    0.5000 100
 0         1         0      100
 0.5000    1.0000    0      100
 0.5000    0.5000    0.5000 100
/
```

▶ After the calculation is completed use `plotband.x` of Quantum Espresso to obtain the phonon dispersion data as an a xmgr file.

```
$  $PATHQE/bin/plotband.x
```

```
Input file >  si.freq
Reading    6 bands at    702 k-points
Range:    0.0000  511.6990eV  Emin, Emax, [firstk, lastk] > 0 600
high-symmetry point:  0.0000 0.0000 0.0000   x coordinate   0.0000
high-symmetry point:  0.7500 0.7500 0.0000   x coordinate   1.0607
...
high-symmetry point:  0.5000 0.5000 0.5000   x coordinate   5.3534
output file (gnuplot/xmgr) > si_ph_dispersion.xmgr
bands in gnuplot/xmgr format written to file si_ph_dispersion.xmgr

output file (ps) >
stopping ...
```

▶ Calculate the interatomic force constants of silicon using a $2\times2\times2$ supercell and the Γ-point for the $\mathbf{q}$-grid.

```
$  cd ../; cp -r ../inputs/222_super/ .; cd 222_super
$ ibrun -np 56 $PATHQE/bin/pw.x -nk 4 < si.scf_222.in > si.scf_222.out
$ ibrun -np 56 $PATHQE/bin/ph.x -nk 4 < si.ph_222.in > si.ph_222.out
$ ibrun -np 1 $PATHQE/bin/q2r.x < q2r.in > q2r.out
```

Input files are provided here:

```
--                                                                    si.scf_222.in
 &control
    calculation = 'scf'
    restart_mode = 'from_scratch',
    prefix = 'si',
    pseudo_dir = './',
    outdir='./'
 /
 &system
    ibrav = 2,
    celldm(1) = 20.40,
    nat =  16,
    ntyp = 1,
    ecutwfc = 20.0
 /
 &electrons
    diagonalization='david'
    mixing_mode = 'plain'
    mixing_beta = 0.7
    conv_thr = 1.0d-7
 /
ATOMIC_SPECIES
 Si  28.086  Si.pz-vbc.UPF
K_POINTS automatic
3 3 3 0 0 0
ATOMIC_POSITIONS (crystal)
Si   0.00000000   0.00000000   0.00000000
Si  -0.12500000   0.37500000  -0.12500000
Si   0.50000000   0.00000000   0.00000000
Si   0.37500000   0.37500000  -0.12500000
Si   0.00000000   0.50000000   0.00000000
Si  -0.12500000   0.87500000  -0.12500000
Si   0.50000000   0.50000000   0.00000000
Si   0.37500000   0.87500000  -0.12500000
Si   0.00000000   0.00000000   0.50000000
Si  -0.12500000   0.37500000   0.37500000
Si   0.50000000   0.00000000   0.50000000
Si   0.37500000   0.37500000   0.37500000
Si   0.00000000   0.50000000   0.50000000
Si  -0.12500000   0.87500000   0.37500000
Si   0.50000000   0.50000000   0.50000000
Si   0.37500000   0.87500000   0.37500000
```

```
--                                                                       si.ph.in
Phonons of Silicon
&inputph
   amass(1)= 28.0855,
   prefix = 'si'
   outdir = './'
   ldisp = .true.
   fildyn = 'si.dyn'
   tr2_ph = 1.0d-12
   nq1 = 1
   nq2 = 1
   nq3 = 1
/
```

```
--                                                                        q2r.in
&input
 fildyn='si.dyn', flfrc = 'si.222.fc'
```

```
/
```

▶ Perform phonon unfolding of the phonons of silicon calculated for a 2×2×2 supercell using `ZG.x`

```
$ ibrun -np 56 $PATHQE/bin/ZG.x -nk 56 < ZG_ph_unfold.in > ZG_ph_unfold.out
```

where the input file should look like:

```
--                                                              ZG_ph_unfold.in
&input
    flfrc='si.222.fc',
    asr='simple', amass(1)=28.0855
    q_in_cryst_coord = .false., q_in_band_form = .true.
    ZG_conf = .false., ph_unfold = .true.
/
&phonon_unfold
     dim1 = 2, dim2 = 2, dim3 = 2
     ng1 = 14, ng2 = 14, ng3 = 14
     flfrq='frequencies.dat', flweights='unfold_weights.dat'
/
 9
   0.000000   0.000000   0.000000  74
   1.500000   1.500000   0.000000   1
   0.500000   2.000000   0.500000  24
   0.000000   2.000000   0.000000  70
   0.000000   0.000000   0.000000  60
   1.000000   1.000000   1.000000  60
   0.000000   2.000000   0.000000  34
   1.000000   2.000000   0.000000  50
   1.000000   1.000000   1.000000   1
/
```

Here, lines 3-5 contain standard inputs for a phonon dispersion calculation (as for `matdyn.x`). In line 4, we specify that we would like to use `ZG.x` for phonon unfolding (`ph_unfold = .true.`) and not to generate special displacements (`ZG_conf = .false.`). In the namelist `&phonon_unfold` we have: (i) `dimX` which are essentially the supercell dimensions, (ii) `ngX` define the range of the reciprocal lattice vectors (and thus plane waves) that need to be employed for calculating the spectral weights. Check for convergence by increasing this number; already 14 is a tight value, (iii) `flfrq` is the name of the output file containing the frequencies for each band and $\mathbf{Q}$-point and (iv) `flweights` is the name of the output file containing the spectral weights for each band and $\mathbf{Q}$-point. At the end of the file we provide the high-symmetry path for which we would like to perform phonon unfolding. The input structure of path is in the same spirit with a `matdyn.x` calculation. As for the electron band structure unfolding, the coordinates of the $\mathbf{Q}$-points are obtained by multiplying the $\mathbf{q}$ wavevectors (defining the fundamental Brillouin zone) with the dimensions of the supercell, i.e. if $\mathbf{q} = [x \;\; y \;\; z]$ then $\mathbf{Q} = [\mathrm{dim1} \times x \quad \mathrm{dim2} \times y \quad \mathrm{dim3} \times z]$.

▶ Now we have the phonon frequencies and their spectral weights, we can calculate the phonon spectral function using `pp_spctrlfn.x`. Convert first the raw data in **frequencies.dat** and **unfold_weights.dat** into a different format using `plotband.x` of QE (commands provided in `bands.sh`):

```
$ $PATHQE/bin/plotband.x unfold_weights.dat < spectral_weights.in > spectral_weights.out
$ $PATHQE/bin/plotband.x frequencies.dat < energies.in > energies.out
$ sed -i '/^\s*$/d' spectral_weights.dat # remove empty lines
$ sed -i '/^\s*$/d' energies.dat
$ paste energies.dat spectral_weights.dat > tmp
$ awk '{print $1,$2,$4}' tmp > energies_weights.dat; rm tmp
```

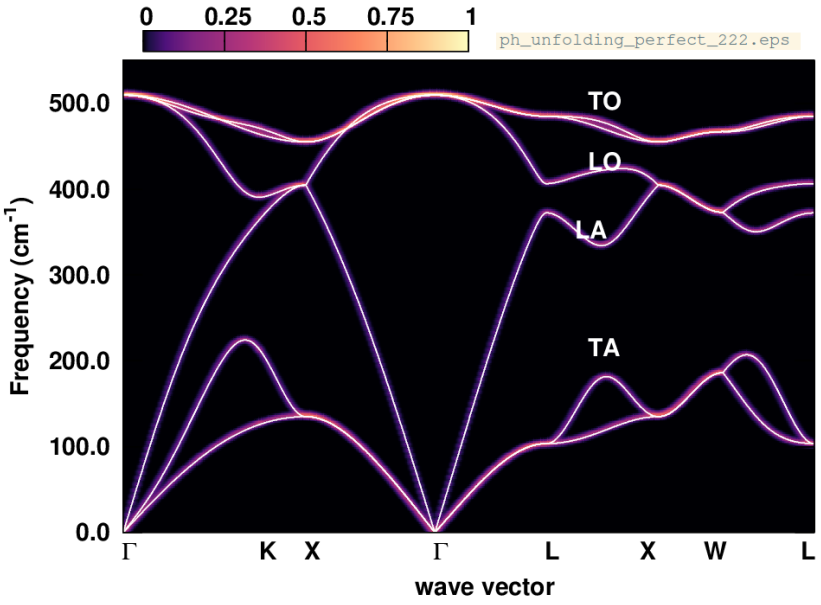
You have just generated **energies_weights.dat** file in a similar format to a ".gnu" file where the first column has the momentum-axis values, the second the phonon frequencies, and the third the spectral weights. Now proceed with the calculation of the spectral function:

```
$ ibrun -n 28 $PATHQE/bin/pp_spctrlfn.x -nk 28 < pp_spctrlfn.in > pp_spctrlfn.out
$ cp spectral_function.dat ../../gnuplot/spectral_function_222.dat
$ cd ../../gnuplot/; gnuplot gp_pm3d_222.p; evince ph_unfolding_perfect_222.eps
```

where the input file should look like:

```
--                                                                    pp_spctrlfn.in
&input
  flin = 'energies_weights.dat'
  steps = 17952,
  ksteps = 300,
  esteps = 300,
  kmin = 0,
  kmax = 5.3534,
  emin =  -10
  emax = 550
  flspfn = 'spectral_function.dat'
/
```

Note: The description of the input flags is available in Exercise 2.



In this plot, the silicon phonon spectral function (color map) overlays with the phonon dispersion calculated in the unit cell, since the supercell is an exact periodic replica of the unit cell. One can also check the raw data in `unfold_weights.dat`, and observe that the weights of the phonon frequencies are either 1 or 0, making an one-to-one correspondence with the frequencies calculated in the unit cell.

▶ Now go to `workdir`, calculate the phonons of P-doped silicon, and perform phonon unfolding:

```
$ cd ../workdir/; cp -r  ../inputs/222_super_P_doped .; cd 222_super_P_doped/
$ ibrun -np 56 $PATHQE/bin/pw.x -nk 4 < si.scf_222.in > si.scf_222.out
$ ibrun -np 56 $PATHQE/bin/ph.x -nk 4 < si.ph_222.in > si.ph_222.out
$ ibrun -np 1 $PATHQE/bin/q2r.x < q2r.in > q2r.out
$ ibrun -np 56 $PATHQE/bin/ZG.x -nk 56 < ZG_ph_unfold.in > ZG_ph_unfold.out
```

Note: Here we replace one Si atom with a P atom and provide the relaxed structure. Due to the excess electron the system becomes metalic.

Here we provide all input files:

```
--                                                                    si.scf_222.in
 &control
    calculation = 'scf'
    restart_mode = 'from_scratch',
    prefix = 'si',
    pseudo_dir = './',
    outdir='./'
 /
 &system
    ibrav = 2,
    celldm(1) = 20.40,
    nat =  16,
    ntyp = 2,
    ecutwfc = 30.0
    occupations='smearing'
    degauss = 0.01
    nosym =.true.
 /
 &electrons
    diagonalization='david'
    mixing_mode = 'plain'
    mixing_beta = 0.7
    conv_thr = 1.0d-7
 /
ATOMIC_SPECIES
 Si  28.086  Si.pz-vbc.UPF
 P  30.974   P.pz-hgh.UPF
K_POINTS automatic
3 3 3 0 0 0
ATOMIC_POSITIONS (crystal)
P             0.0000000002       -0.0000000243        0.0000000412
Si           -0.1261235427        0.3783706126       -0.1261235364
Si            0.4983582216        0.0016417759        0.0016417825
Si            0.3751699490        0.3751699540       -0.1255098692
Si            0.0016417706        0.4983582252        0.0016417800
Si           -0.1261235463        0.8738764598       -0.1261235454
Si            0.4983582134        0.4983582133        0.0016417907
Si            0.3783706477        0.8738764517       -0.1261235465
Si            0.0016417985        0.0016418052        0.4983581945
Si           -0.1255098925        0.3751699883        0.3751699378
Si            0.4983582056        0.0016417944        0.4983582063
Si            0.3751699659        0.3751699770        0.3751699562
Si            0.0016417902        0.4983582115        0.4983582106
Si           -0.1261235587        0.8738764426        0.3783706794
Si            0.4999999969        0.5000000190        0.4999999746
Si            0.3751699807        0.8744900937        0.3751699437
/
```

```
--                                                                    si.ph_222.in
&inputph
   amass(1)= 28.0855,
   amass(2)= 30.974,
   prefix = 'si'
   outdir = './'
   ldisp = .true.
   fildyn = 'si.dyn'
   tr2_ph = 1.0d-12
```

```
   epsil = .false.
   nq1 = 1
   nq2 = 1
   nq3 = 1
/
```

```
--                                                                    q2r.in
&input
 fildyn='si.dyn', flfrc = 'si.222.fc'
/
```

```
--                                                                    ZG_ph_unfold.in
&input
    flfrc='si.222.fc',
    asr='simple', amass(1)=28.0855, amass(2)= 30.974
    q_in_cryst_coord = .false., q_in_band_form = .true.
    ZG_conf = .false.
    ph_unfold = .true.
/
&phonon_unfold
     dim1 = 2, dim2 = 2, dim3 = 2
     ng1 = 14, ng2 = 14, ng3 = 14
     flfrq='frequencies.dat', flweights='unfold_weights.dat'
/
 9
   0.000000   0.000000   0.000000  74
   1.500000   1.500000   0.000000   1
   0.500000   2.000000   0.500000  24
   0.000000   2.000000   0.000000  70
   0.000000   0.000000   0.000000  60
   1.000000   1.000000   1.000000  60
   0.000000   2.000000   0.000000  34
   1.000000   2.000000   0.000000  50
   1.000000   1.000000   1.000000   1
```

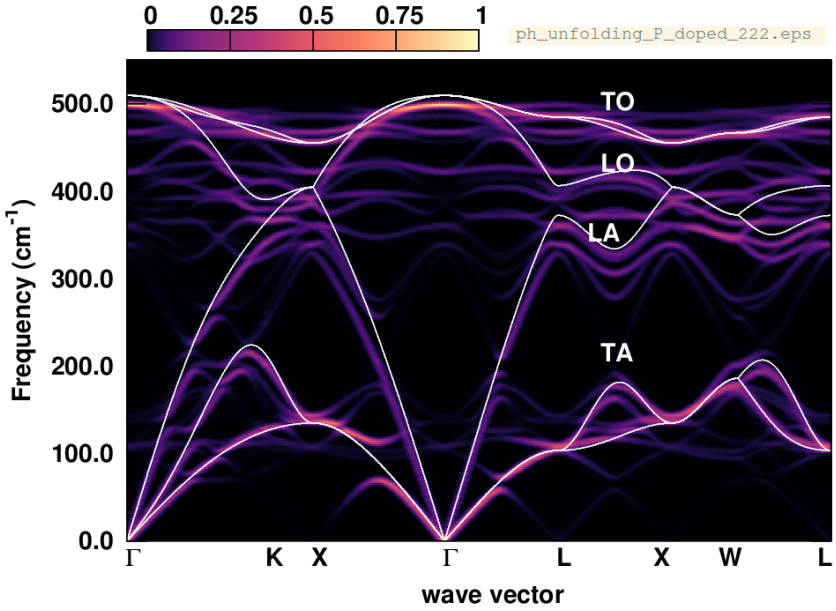
▶ Now we have the phonon frequencies and their spectral weights, we can calculate the phonon spectral function using `pp_spctrlfn.x` as before. Convert first the raw data in **frequencies.dat** and **unfold_weights.dat** into a different format using `plotband.x` of QE (commands provided in `bands.sh`):

```
$ $PATHQE/bin/plotband.x unfold_weights.dat < spectral_weights.in > spectral_weights.out
$ $PATHQE/bin/plotband.x frequencies.dat < energies.in > energies.out
$ sed -i '/^\s*$/d' spectral_weights.dat # remove empty lines
$ sed -i '/^\s*$/d' energies.dat
$ paste energies.dat spectral_weights.dat > tmp
$ awk '{print $1,$2,$4}' tmp > energies_weights.dat; rm tmp
```

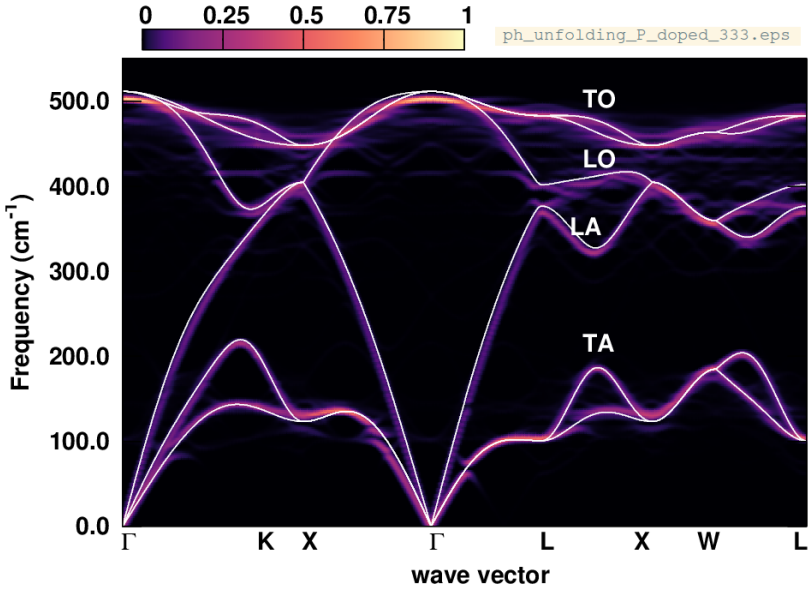
Now proceed with the calculation of the spectral function:

```
$ ibrun -n 28 $PATHQE/bin/pp_spctrlfn.x -nk 28 < pp_spctrlfn.in > pp_spctrlfn.out
$ cp spectral_function.dat ../../gnuplot/spectral_function_222_P_doped.dat
$ cd ../../gnuplot/; gnuplot gp_pm3d_222_P_doped.p; evince ph_unfolding_P_doped_222.eps
```

where the input file `pp_spctrlfn.in` is the same as before.



What are the changes in the phonon dispersion? Below we also provide the phonon spectral function calculated after replacing one Si atom with a P atom in a 3×3×3 supercell. Why are the changes less pronounced now?



APPENDIX / Exercise 2

In this section of the appendix we show how the band structure unfolding calculation can be broken down to one **K**-point at a time. This strategy is useful when the supercell size is large (larger than $4 \times 4 \times 4$) and the calculation requires a large amount of memory and storage space for the wavefunctions.

▶ Go to `workdir/band_structure_unfolding` of Exercise 2 and copy:

```
$ cp -r ../../../exercise1/workdir/ZG-si.save/ .
$ cp ../../inputs/ZG-bands_333_0.00K.in ZG-bands_333_0.00K
$ cp ../../inputs/bands_unfold.in bands.in
```

In `ZG-bands_333_0.00K` apply the following modifications:

1. Set `calculation = 'bands'`
2. Delete the following lines containing information about the **K**-points:

```
K_POINTS crystal_b
2
0.000000 0.000000 0.000000 35
0.000000 1.500000 1.500000 1
```

3. At the bottom of the file add the following (after ZG atomic positions):

```
K_POINTS crystal_b
1
```

The file `ZG-bands_333_0.00K` should look like this:

```
&control                                                    ZG-bands_333_0.00K
  calculation = 'bands'
  restart_mode = 'from_scratch'
  prefix = 'ZG-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
  nbnd = 135
  nosym = .true.
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS {angstrom}
```

```
   Si       0.02090173     -0.01220213     -0.01392228
   Si      -1.39812907      1.32606198      1.38415986
   Si      -2.68002772      0.00399202      2.64203512
...
K_POINTS crystal_b
1
```

▶ Prepare the **K**-point grid for band structure unfolding. To this aim run:

```
$ $PATHQE/EPW/ZG/src/local/kpoints_band_str_unfold.x
```

This script is designed to generate a list of **K**-points that will unfold back to the fundamental Brillouin zone. You will be asked to provide: (i) the number of high-symmetry **k**-points in the Brillouin zone of the unit-cell, (ii) their coordinates, (iii) their position along the $\mathbf{k}$-axis, (iv) the supercell size, and (v) whether you want to print every single **K**-point. For example:

```
 Write number of high-symmetry kpts
2
 Write high-symmetry kpts (3 columns per row)
0.0 0.0 0.0
0.0 0.5 0.5
 Write x-positions of high-sym kpts
0.0
1.0
 Step is (default):  2.8571429E-02
 Supercell size (n * m * p) ?
3 3 3
 kpts to use for Supercell calculation:
   0.000000   0.000000   0.000000  35
   0.000000   1.500000   1.500000   1
 Write every single kpt? (0=no, 1=yes)
1
...
```

Note: for generating more **K**-points along the high-symmetry paths, you can increase the `step` value from the script: `$PATHQE/EPW/ZG/src/local/kpoints_band_str_unfold.f90`.

Pass all 35 **K**-points with their weights into a new file `Kpoints.dat`. Otherwise you can run:

```
$  cp ../../inputs/input_Kpoints.in .
$  $PATHQE/EPW/ZG/src/local/kpoints_band_str_unfold.x \
   < input_Kpoints.in > Kpoints.dat
$  sed -i '1,9d' Kpoints.dat # deletes first 9 info lines from Kpoints.dat
```

▶ Run a band structure unfolding calculation.

```
$ cp ../../inputs/Kpoints.dat . # if you did not complete the previous action
$ cp ../../inputs/merge_files.sh .; ./merge_files.sh
```

Note: The script `merge_files.sh` will generate 35 input files `si.333_ZG_bands_X.in` to be used for a bands calculation for every single **K**-point. This strategy is the most efficient way to perform BSU in large supercells.

Pass the following lines into a script and run:

```
# This script runs a bands calculation with "pw.x" and then performs
# band structure unfolding with "bands_unfold.x".
prefix=ZG-si
```

```
i=1  # initial kpt index to calculate
f=35 # final kpt index to calculate
#
while [ $i -le $f ];do
    mkdir kpt_$i
    cd kpt_$i
#
    EXE=$PATHQE/bin/pw.x
    JNAME=ZG-bands_333_0.00K
#
    cp -r ../"$prefix".save .
    mv ../"$JNAME"_"$i".in .
    ibrun -n 14 $EXE  < "$JNAME"_"$i".in > "$JNAME"_"$i".out
#
    EXE=$PATHQE/bin/bands_unfold.x
    JNAME=bands
#
    cp ../"$JNAME".in .
    sed -i 's/tmp/'$i'/g' $JNAME.in
    ibrun -n 14 $EXE  < "$JNAME".in > "$JNAME".out
#
    rm -r *wfc* "$prefix".save
    cd ../
    i=$((i+1))
done
```

Note: You can also copy the script from `inputs`, i.e. `cp ../../inputs/script_bands.sh .`
This script will generate the output directories **kpt_X**, containing the band energies files (**bands01.dat**) and spectral weights (**spectral_weights01.dat**) for each single **K**-point. The full calculation should take around 5 mins. Meanwhile you can check the progress of the calculation with `ls -lrt` and see how many **kpt_X** files have been generated so far. If **kpt_1** exist, then check the outputs of **bands01.dat** and **spectral_weights01.dat**. The should contain the energies and spectral weights for **K**-point 1 (i.e. Γ here).

▶ Once energies and spectral weights for all **K**-points are calculated type:

```
$ cp ../../inputs/merge_kpts.sh .; ./merge_kpts.sh; cd all_kpts; ls
```

This will generate `all_kpts` directory and the new files **bands01.dat** and **spectral_weights01.dat**, containing the band energies, $\varepsilon_{m\mathbf{K}}(T)$, and spectral weights, $P_{m\mathbf{K},\mathbf{k}}(T)$, for all **K**-points, respectively. Now you have all the ingredients to evaluate the spectral function as in Exercise 2.

▶ Now repeat all steps after Equation 1 in Exercise 2. (Note that you are one directory down, so modify accordingly; you might also need to modify `steps` in `pp_spctrlfn.in` to 4725)

APPENDIX / Exercise 3

In this section of the appendix we show how the calculation of temperature-dependent optical spectra can be broken down to one **K**-point at a time. This strategy is useful when the supercell size is large (larger than $4 \times 4 \times 4$) and the calculation requires a large amount of memory and storage space for the wavefunctions.

▶ First go to the directory `exercise3`, create your working directories and copy the input files:

```
$ cd ../../exercise3/; mkdir workdir; cd workdir; mkdir equil; mkdir ZG
$  cd equil
$ cp ../../inputs/equil/K_list.in .
$ cp ../../inputs/equil/epsilon.in .
$ cp ../../inputs/equil/epsi_av.sh .
$ cp -r ../../../exercise1/workdir/equil-si.save/ .
$ cp ../../../exercise2/workdir/equil-nscf_333.in equil-nscf_333
```

The file `K_list.in` contains the crystal coordinates of 200 randomly generated **K**-points with all weights set to `1.0` (we assign this weight, since we will perform calculations for each **K**-point separately, in a similar spirit with the BSU calculation). We use random **K**-points, instead of a uniform grid, in order to speed up convergence. In `si.333_equil_nscf` apply the following changes:

1. Add `nosym = .true.` below `nbnd = 135`.
2. Delete the information for the **K**-points:

   ```
   K_POINTS crystal
   2
   0.000000 0.000000 0.000000 1
   0.000000 1.260000 1.260000 1
   ```

   and at the **bottom** of the file add the following:

   ```
   K_POINTS crystal
   1
   ```

   Note: although we use the equilibrium structure we set `nosym = .true.` since random **K**-points are employed.

The file `equil-nscf_333` should look like this:

```
&control                                                    equil-nscf_333
  calculation = 'nscf'
  restart_mode = 'from_scratch'
  prefix = 'equil-si'
  pseudo_dir = './'
  outdir = './'
/
&system
  ibrav = 0
  nat =   54
  ntyp =    1
  ecutwfc =  20.00
  nbnd = 135
  nosym = .true.
/
&electrons
  diagonalization = 'david'
  mixing_mode= 'plain'
  mixing_beta = 0.70
  conv_thr =   0.1D-06
/
```

```
ATOMIC_SPECIES
   Si   28.085 Si.pz-vbc.UPF
CELL_PARAMETERS {angstrom}
     -8.09641133      0.00000000      8.09641133
      0.00000000      8.09641133      8.09641133
     -8.09641133      8.09641133      0.00000000
ATOMIC_POSITIONS {angstrom}
   Si     0.00000000   0.00000000   0.00000000
   Si    -1.34940189   1.34940189   1.34940189
   Si    -2.69880378   0.00000000   2.69880378
...
K_POINTS crystal
1
```

▶ Run an optical spectrum calculation for silicon using the $3 \times 3 \times 3$ equilibrium supercell and 30 random **K**-points. In the same spirit with the BSU calculation we calculate $\epsilon_2(\omega)$ for every **K**-point using the following script:

```
prefix=equil-si
i=1  # initial kpt index to calculate
f=30 # final kpt index to calculate

while [ $i -le $f ];do
#
    JNAME=equil-nscf_333
    awk 'NR == '$i'' K_list.in > K_point.txt
    cat $JNAME K_point.txt > "$JNAME"_"$i".in
#
    mkdir kpt_$i
    cd kpt_$i
#
    EXE=$PATHQE/bin/pw.x
#
    cp -r ../"$prefix".save .
    mv ../"$JNAME"_"$i".in .
    ibrun -n 14 $EXE  < "$JNAME"_"$i".in > "$JNAME"_"$i".out
#
    EXE=$PATHQE/bin/epsilon_Gaus.x
    JNAME=epsilon
#
    cp ../"$JNAME".in .
    sed -i 's/tmp/'$i'/g' $JNAME.in
    ibrun -n 14 $EXE  < "$JNAME".in > "$JNAME".out
#
    rm -r *wfc* "$prefix".save
    cd ../
    i=$((i+1))
done
```

Note: You can also copy the script from `inputs`, i.e. `cp ../../inputs/equil/script_equil.sh .`

This script will generate the output directories **kpt_X**, containing $\epsilon_2(\omega)$ in the file **epsi_equil-si.dat** for each single **K**-point. The full calculation should take around 3 mins. Meanwhile you can check the progress of the calculation with `ls -lrt` and see how many **kpt_X** directories have been generated. If **kpt_1** exist, then check the format of **kpt_1/epsi_si_333_equil.dat**. The first column is the energy grid, and the rest three represent $\epsilon_2(\omega)$ along the three Cartesian directions.

▶ Once the calculation is completed run the script:

```
$ ./epsi_av.sh
```

This script takes first the isotropic average of $\epsilon_2(\omega)$ calculated for each **K**-point and then takes the average of $\epsilon_2(\omega)$ over all **K**-points. The output file containing this data is **epsi_si_333_equil_30.dat**.

Note: `epsi_av.sh` takes the average over 30 **K**-points. You can modify this number using the variable `m` in the script.

Now copy the output file in the `gnuplot` directory using:

```
$ cp epsi_si_333_equil_30.dat ../../gnuplot/.
```

▶ Run an optical spectra calculation for silicon using the $3 \times 3 \times 3$ ZG supercell following exactly the same procedure used for the equilibrium structure (as above); make sure now you use the file `ZG-nscf_333_0.00K.in`.