2022 School on Electron-Phonon Physics from First Principles
13-19 June 2022, Austin, TX

Hands-On Tutorial
Basic calculations with QUANTUM ESPRESSO

In this session we will learn how to use the core capabilities of QUANTUM ESPRESSO.
It may be useful to prepare a script file like the following one:

```
#!/bin/bash
#SBATCH --nodes=1               # Total number of nodes
#SBATCH --ntasks-per-node=56    # Sufficiently descriptive
#SBATCH -t 00:05:00             # Run time (hh:mm:ss)
#SBATCH --account=EPSchool2022
#SBATCH --partition=small
#SBATCH --reservation=EPSchoolDay1
module purge
module load TACC

cd $PWD
```

You are advised to define the following environment variables:

```
PATHSC=/work2/06868/giustino/EP-SCHOOL/      # path to school material
PATHQE=/work2/06868/giustino/EP-SCHOOL/q-e/  # path to Quantum ESPRESSO
export ESPRESSO_PSEUDO=./                    # path to pseudopotential files
export ESPRESSO_TMPDIR=./tmpdir              # path to output data files
```

Download the tutorial input files and go in the first exercise:

```
$ cp $PATHSC/Mon.4.Giannozzi/exercise_Si.tar.gz
$ tar -xzvf exercise_Si.tar.gz
$ cd exercise_Si
```

Note: in this tutorial it will be shown how to obtain all the input files and related files. The folder that you downloaded should be used as a reference or to speed up the process.

Before Starting

The basic step for any advanced calculation is to find the ground state of the system you are interested in. This requires:

0. to provide the correct structure and data in input !!!
1. to find and test a suitable set of pseudopotentials: see the tables and the links listed at `https://www.quantum-espresso.org/pseudopotentials/` and in particular the Standard Solid-State Pseudopotential (SSSP) collection at `https://www.materialscloud.org/discover/sssp`;

2. to perform convergence tests on the plane-wave basis set (energy cutoff) and on the sum of the charge density (k-point grid, Fermi surface treatment for metals);

3. to locate the minimum-energy structure, typically with the code `pw.x`.

For electron-phonon calculations, the next basis step is the calculation of the phonon dispersion with code `ph.x` and with other auxiliary codes. This requires, in addition to the previous steps,

4. to choose a suitable grid of phonon wave-vectors.

In case of trouble,

- have a careful look at input data and error messages;
- consult the documentation, online at `https://www.quantum-espresso.org/documentation/` or in `$PATHQE/PW/Doc/` and `$PATHQE/PHonon/Doc/` subdirectories;
- search the mailing list `users@lists.quantum-espresso.org`, or post a question there (see `https://www.quantum-espresso.org/users-forum/`).

If everything else fails, look into the code!

Exercise 1

In this exercise we will compute selected phonons of **Silicon** in the diamond structure, using LDA norm-conserving pseudopotentials. This is the simplest case one can think of.

▶ Run a self-consistent calculation for Silicon

Note1: This run is fast, parallel execution is not really needed.

Note2: environment variables `ESPRESSO_PSEUDO` and `ESPRESSO_TMPDIR` point respectively to the directory where pseudopotential files are and where output data is written. Can be used as an alternative to input keywords `pseudo_dir` and `outdir`.

Note3: with some parallel libraries, input redirection with "<" may not work properly; "-i"is safer

```
$ ibrun -np 4 $PATHQE/bin/pw.x -i Si.scf.in > si.scf.out
```

```
--                                                                    Si.scf.in
 &control
    calculation     = 'scf'
 /
 &system
    ibrav           = 2
    celldm(1)       = 10.26
    nat             = 2
    ntyp            = 1
    ecutwfc         = 30.0
 /
 &electrons
    diagonalization = 'david'
    mixing_beta     = 0.7
    conv_thr        = 1.0d-10
 /
ATOMIC_SPECIES
  Si  28.0855     Si.pz-vbc.UPF
ATOMIC_POSITIONS alat
  Si  0.00  0.00  0.00
```

```
  Si   0.25   0.25   0.25
K_POINTS automatic
 4 4 4 1 1 1
```

▶ Run a phonon calculation at Γ ($\mathbf{q} = 0$) using the following input:

```
--                                                                    Si.phG.in
&inputph
 fildyn = 'siG.dyn',
 tr2_ph = 1.0d-12
 amass(1)=28.0855
/
0.0 0.0 0.0
```

The keyword tr2_ph is the threshold for phonon self-consistency (sufficient in this case, may be too large in general). The keyword fildyn tells the code where to write the computed dynamical matrix. The keyword amass overwrites (if present) the mass read in the scf calculation.
`$ ibrun -np 4 $PATHQE/bin/ph.x -i Si.phG.in > si.phG.out`

In the output file, locate the list of irreps (irreducible representations, 2 for Si at $\mathbf{q} = 0$) and relative degeneracies (3 and 3, total 6 = 2 * number of atoms). For each irrep, a linear-response calculation is performed. At the end, six frequencies are found:

```
freq (    1) =       0.477968 [THz] =      15.943297 [cm-1]
freq (    2) =       0.477968 [THz] =      15.943297 [cm-1]
freq (    3) =       0.477968 [THz] =      15.943297 [cm-1]
freq (    4) =      15.042327 [THz] =     501.758022 [cm-1]
freq (    5) =      15.042327 [THz] =     501.758022 [cm-1]
freq (    6) =      15.042327 [THz] =     501.758022 [cm-1]
```

Note that the zero-frequency acoustic modes at $\mathbf{q} = 0$ have non-zero frequency! If you want them to be zero, you need to impose the Acoustic Sum Rule (ASR). This can be achieved using code `dynmat.x`.

▶ Impose ASR, save phonon displacements in a plottable form

```
--                                                                 Si.dynmat.in
&input fildyn = 'siG.dyn', asr='simple' /
```

`$ $PATHQE/bin/dynmat.x -i Si.dynmat.in > si.dynmat.out`

Notice keyword asr: acoustic modes have now (almost) zero frequency. File dynmat.asxf contains normal modes in a format that can be visualized using `XCrySDen` (`xcrysden --axsf dynmat.axsf`): they appear as forces on atoms. Do they look reasonable? what did you expect?

▶ Run a phonon calculation at X ($\mathbf{q} = (0, 0, 1)$ in units $2\pi/a_0$) using the following input:
Note: no need to redo the scf calculation, as long as you do not need or want to modify it

```
--                                                                    Si.phX.in
&inputph
 fildyn = 'siX.dyn',
 tr2_ph = 1.0d-12
 amass(1)=28.0855
```

```
/
0.0 0.0 1.0
```

```
$ ibrun -np 4 $PATHQE/bin/ph.x -i Si.phX.in > si.phX.out
```

In the first part of the output, a non-scf calculation is performed. Notice in the output file that there are now 3 irreps of degeneracy 2. At the end, the six frequencies (3 doubly degenerate): The symmetry of the small group of **q** is assumed instead of the crystal symmetry, so the number of **k**-points increases. Note the presence of **k**+**q** vectors intercalated between the **k** vectors.
The linear-response calculation is performed in the second part of the output. Notice that the number and degeneracies of irreps are different from those at Γ: they depend on symmetry, i.e. on the small group of **q**. The file si.dynX contains now three matrices, i.e. the three equivalent X points.

```
freq (    1) =       4.353939 [THz] =     145.231774 [cm-1]
freq (    2) =       4.353939 [THz] =     145.231774 [cm-1]
freq (    3) =      12.013967 [THz] =     400.742810 [cm-1]
freq (    4) =      12.013967 [THz] =     400.742810 [cm-1]
freq (    5) =      13.376804 [THz] =     446.202143 [cm-1]
freq (    6) =      13.376804 [THz] =     446.202143 [cm-1]
```

You may do the same for points L ($\mathbf{q} = (0.5, 0.5, 0.5)$ in units $2\pi/a_0$), or for any other point.

Exercise 2

In this exercise we will compute the entire phonon dispersion for **Silicon**, using Fourier interpolation.

▶ Run a phonon dispersion calculation for a 4x4x4 grid of wave-vectors $\mathbf{q}$.
Note the keyword ldisp instructing the code to make a full dispersion calculation for a nq1*nq2*nq3 grid of wave-vectors.

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

```
$ ibrun -np 4 $PATHQE/bin/ph.x -i Si.ph.in > si.ph.out
```

In the output file, locate the list of irreducible $\mathbf{q}$ points in the Brillouin Zone (IBZ):

```
Dynamical matrices for ( 4, 4, 4)  uniform grid of q-points
(   8 q-points):
  N         xq(1)         xq(2)         xq(3)
  1   0.000000000   0.000000000   0.000000000
  2  -0.250000000   0.250000000  -0.250000000
  3   0.500000000  -0.500000000   0.500000000
```

```
    4   0.000000000   0.500000000   0.000000000
    5   0.750000000  -0.250000000   0.750000000
    6   0.500000000   0.000000000   0.500000000
    7   0.000000000  -1.000000000   0.000000000
    8  -0.500000000  -1.000000000   0.000000000
```

followed by 8 phonon calculations, one per $\mathbf{q}$ point. The list of irreducible $\mathbf{q}$ points is also written in the `si.dyn0` file. If you type `ls`, you can see `si.dynN` files containing the dynamical matrix has been produced for each irreducible $\mathbf{q}$ point.

▶ Compute interatomic force constants (IFC) in real space for **Silicon** with auxiliary code `q2r.x`

Note: keyword fildyn same as in the phonon dispersion calculation.

Si.q2r.in

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

```
$ $PATHQE/bin/q2r.x -i Si.q2r.in > si.q2r.out
```

All dynamical matrices are read, Fourier-transformed, stored in file name specified by keyword flfrc.

Note: before proceeding to serious calculations, it is wise to assess the quality of Fourier interpolation, e.g. by comparing the frequency computed directly with those computed with IFCs at a generic $\mathbf{q}$ vector *not* in the grid used for IFC calculation.

▶ Compute phonon dispersions using real-space IFC's and auxiliary code `matdyn.x`

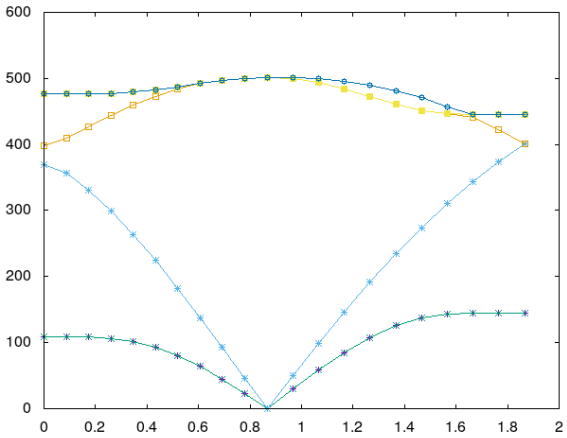
Note: keyword filfrc same as in calculation of IFC's. Keyword q_in_band_form provides a compact way to supply $\mathbf{q}$-vectors for dispersion plotting (below, L-Γ-X) as a list of high-symmetry points plus the number of points in the line between them

Si.phband.in

```
--
&input
flfrc='si.fc',
asr='simple',
flfrq='si.freq',
q_in_band_form=.true.
/
3
0.5 0.5 0.5 10
0.0 0.0 0.0 10
0.0 0.0 1.0  1
```

```
$ $PATHQE/bin/dynmat.x -i Si.phband.in > si.phband.out
```

The file specified in flfrq plus suffix `.gp` contains a list of frequencies (in cm^{-1}) that can be directly plotted using `gnuplot.x`.



▶ Compute phonon DOS, using real-space IFC's and auxiliary code `matdyn.x`

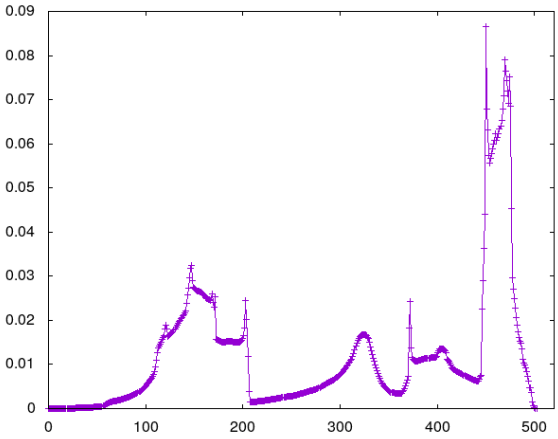
Note: if keyword dos is set to true, one has to specify a uniform wave-vector grid with keywords nk1 nk2 nk3, using the same logic as for Brillouin-Zone sums (Monkhorst-Pack grids).

Si.phdos.in

```
--
&input
flfrc='si.fc'
asr='simple'
dos=.true.
nk1 = 8
nk2 = 8
nk3 = 8
fldos='si.dos'
/
```

```
$ $PATHQE/bin/dynmat.x -i Si.phdos.in > si.phdos.out
```

The file specified in fldos contains the DOS in a format that can immediately be plotted using `gnuplot.x`, or any other plotting program.



Exercise 3

In this exercise we will examine the effect of macroscopic electric fields in polar materials, e.g. **AlAs** in the zincblende structure.

▶ Run a self-consistent calculation for AlAs

Note: here we explicitly set a "prefix" in order to label output data

```
$ cp $PATHSC/Mon.4.Giannozzi/exercise_AlAs.tar.gz
$ tar -xzvf exercise_AsAl.tar.gz
$ cd exercise_AsAl
```

AlAs.scf.in

```
--
 &control
    calculation     = 'scf'
    prefix          = 'alas'
 /
 &system
    ibrav           = 2
    celldm(1)       = 10.60
    nat             = 2
    ntyp            = 2
    ecutwfc         = 30.0
 /
 &electrons
    diagonalization = 'david'
    mixing_beta     = 0.7
    conv_thr        = 1.0d-10
 /
ATOMIC_SPECIES
  Al  26.98       Al.pz-vbc.UPF
  As  74.92       As.pz-bhs.UPF
ATOMIC_POSITIONS alat
  Al  0.00  0.00  0.00
  As  0.25  0.25  0.25
K_POINTS automatic
 4 4 4 1 1 1
```

```
$ ibrun -np 4 $PATHQE/bin/pw.x -i AlAs.scf.in > alas.scf.out
```

▶ Run a phonon calculation at Γ ($\mathbf{q} = 0$) with macroscopic electric fields:

AlAs.phG.in

```
--
&inputph
 prefix='alas',
 fildyn = 'alasG.dyn',
 tr2_ph = 1.0d-12
 amass(1)=26.98
 amass(2)=74.92
 epsil=.true.
/
0.0 0.0 0.0
```

The keyword prefix must be the same as in the scf calculation. The keyword epsil enables the calculation of macroscopic electric fields at $\mathbf{q}$=0. Note that effective charges can be computed in two different but equivalent modes (keywords zue and zeu).

```
$ ibrun -np 4 $PATHQE/bin/ph.x -i AlAs.phG.in > alas.phG.out
```

In the output, notice the three linear-response calculations to an electric field. The dielectric constant and effective charges are calculated and stored into the "alas.dynG" file. Note that *there is no TO-LO*

splitting. The non-analytic term that produces the TO-LO splitting can be added to the dynamical matrix using auxiliary code `dynmat.x`. Also note that the dielectric tensor is the electronic term only (so-called ϵ_∞, not ϵ_0) and is typically overestimated in DFT.

▶ Add non-analytic terms, compute LO-TO splitting

```
--                                                                    AlAs.dynmat.in
  &input fildyn = 'alasG.dyn', asr='simple',
   q(1)=1.0, q(2)=0.0, q(3)=0.0
  /
```

```
$ $PATHQE/bin/dynmat.x -i AlAs.dynmat.in > alas.dynmat.out
```

The code reads the effective charges and the dielectric tensors from the file specified in fildyn and computes the nonanalytical term for the given **q**. The 3-fold degenerate optical modes now exhibit the TO-LO splitting.

Exercise 4

In this exercise we will compute the entire phonon dispersion for **AlAs**, including the LO-TO splitting

▶ Run a phonon dispersion calculation for a 4x4x4 grid of wave-vectors $\mathbf{q}$.

Note the keyword epsil instructing the code to include macroscopic electric fields at **q**=0.

```
--                                                                        AlAs.ph.in
&inputph
 fildyn = 'alas.dyn',
 tr2_ph = 1.0d-12
 ldisp=.true.
 epsil=.true.
 nq1 = 4
 nq2 = 4
 nq3 = 4
/
```

```
$ ibrun -np 4 $PATHQE/bin/ph.x -i AlAs.ph.in > alas.ph.out
```

The calculation proceeds as in Exercise 2, except for **q**=0 where the same additional calculations as in Exercise 3 are performed.

▶ Compute interatomic force constants (IFC) in real space for **AlAs** with auxiliary code `q2r.x`

Everything is like in Exercise 3:

```
  --                                                                      AlAs.q2r.in
  &input
  fildyn='AlAs.dyn', flfrc='alas.fc'
  /
```

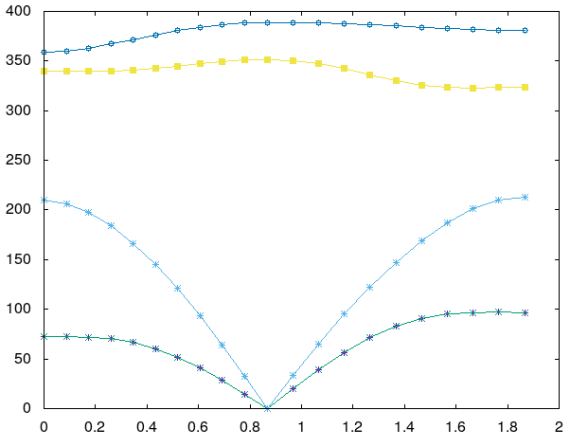
```
$ $PATHQE/bin/q2r.x -i alas.q2r.in > alas.q2r.out
```

▶ Compute phonon dispersions using real-space IFC's and auxiliary code `matdyn.x`

```
  --                                                                   AlAs.phband.in
  &input
```

```
 flfrc='alas.fc', asr='simple', flfrq='alas.freq', q_in_band_form=.true.
 /
 3
0.5 0.5 0.5 10
0.0 0.0 0.0 10
0.0 0.0 1.0  1
```

The file specified in flfrq plus suffix .gp contains a list of frequencies (in cm^{-1}) that can be directly plotted using `gnuplot.x`. Notice the LO-TO splitting: at $\mathbf{q} \neq 0$ it is automatically present in the dynamical matrix, while at $\mathbf{q}$=0 it has to be added via the non-analytical terms in the dynamical matrix




```
$ $PATHQE/bin/dynmat.x -i AlAs.phband.in > alas.phband.out
```

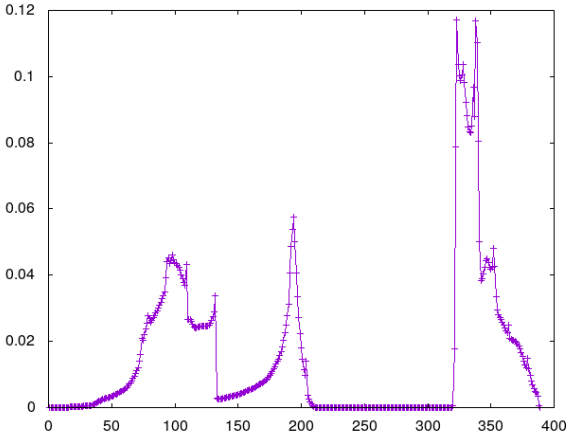
- Compute phonon DOS, using real-space IFC's and auxiliary code `matdyn.x`

AlAs.phdos.in

```
--
&input
flfrc='alas.fc'
asr='simple'
dos=.true.
nk1 = 8
nk2 = 8
nk3 = 8
fldos='alas.dos'
/
```

```
$ $PATHQE/bin/dynmat.x -i AlAs.phdos.in > alas.phdos.out
```

The file specified in fldos contains the DOS in a format that can immediately be plotted using `gnuplot.x`, or any other plotting program.



Exercise 5

In this exercise we will deal with a metal, **Pb**. We will find the optimal treatment of the Fermi surface, then compute the ground state and the phonon spectra.

▶ Find the optimal treatment of the Fermi surface for **Pb**

Note: this requires several calculations at different values of the smearing and with increasingly dense **k**-point grids

```
$ cp $PATHSC/Mon.4.Giannozzi/exercise_Pb.tar.gz
$ tar -xzvf exercise_Pb.tar.gz
$ cd exercise_Pb
```

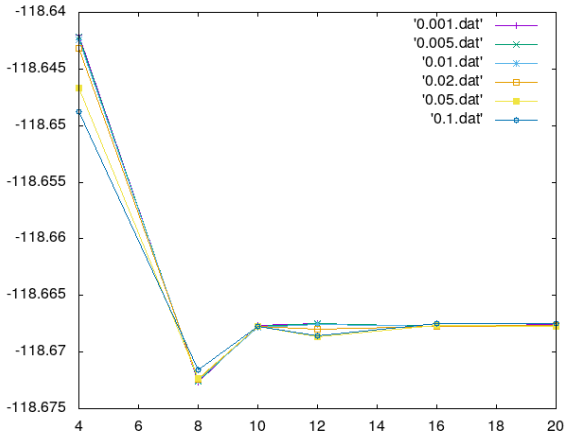
```
--                                                                    pb.scf.in
 &control
    calculation     = 'scf'
    prefix          = 'lead'
 /
 &system
    ibrav           = 2
    celldm(1)       = 9.2225583816
    nat             = 1
    ntyp            = 1
    ecutwfc         = 30.0
    occupations     = 'smearing',
    smearing        = 'marzari-vanderbilt',
    degauss         = ...
 /
 &electrons
    conv_thr        = 1.0d-8
 /
ATOMIC_SPECIES
  Pb  207.2       pb_s.UPF
ATOMIC_POSITIONS alat
  Pb  0.00  0.00  0.00
K_POINTS automatic
 NK NK NK 0 0 0
```

Run the above for a grid of values of degauss and NK, for instance: degauss=0.001, 0.005, 0.01, 0.02, 0.05, 0.1 Ry and NK=4,6,8,10,12,16,20; collect the values of the energy. It is conveniente to use a small script.

```
$ ibrun -np 4 $PATHQE/bin/pw.x -i pb.scf.in > pb.scf.out
$ e=`grep ! pb.scf.out | awk '{print $5}'`
$ echo $e
```

Make a plot of $E(NK)$ for each value of degauss. The converged value of the energy is found for dense grids and small degauss. In practice, it is convenient to use the largest degauss and the less dense **k**-point grid that give an energy very close to the converged one.

For Pb, NK larger than 8-10 and degauss no larger than 0.1 Ry look appropriate here, but you may want to make more careful checks for serious calculations.



▶ Run a phonon dispersion calculation for **Pb**
You may use the previous input for the self-consistent calculation, with NK=12 and degauss=0.05 Ry, and the following for the phonon code:

```
--                                                                    pb.phG.in
&inputph
 prefix='lead',
 fildyn = 'lead.dyn',
 tr2_ph = 1.0d-12
 ldisp=.true.
 nq1=4, nq2=4, nq3=4
/
```

No need to deal with macroscopic electric fields for a metal. Actually, if the keyword epsil is set, the code will complain!

```
ibrun -np 4 $PATHQE/bin/pw.x -i pb.scf.in > pb.scf.out
ibrun -np 4 $PATHQE/bin/ph.x -i pb.ph.in  > pb.ph.out
```

The calculation will proceed as in the previous cases, but at **q**=0 an additional term, "Fermi energy shift", will be computed: see S. de Gironcoli, Phys. Rev **B** 51, 6773R (1995).
▶ Compute interatomic force constants (IFC) in real space for **Pb** with auxiliary code `q2r.x`

```
 --                                                                   pb.q2r.in
 &input
 fildyn='lead.dyn', flfrc='lead.fc'
 /
```

```
$ $PATHQE/bin/q2r.x -i pb.q2r.in > pb.q2r.out
```

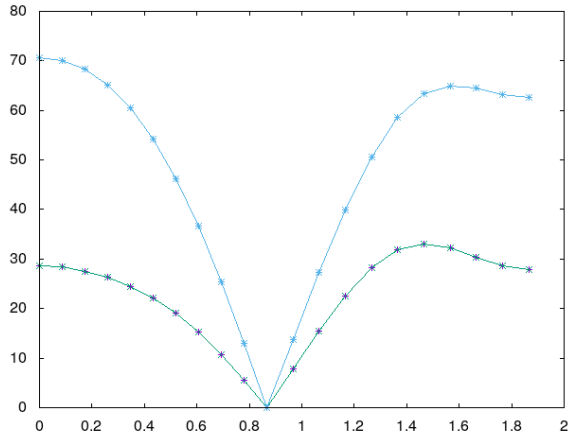
▶ Compute phonon dispersions using real-space IFC's and auxiliary code `matdyn.x`

```
 --                                                                pb.phband.in
 &input
 flfrc='lead.fc', asr='simple', flfrq='lead.freq', q_in_band_form=.true.
 /
 3
0.5 0.5 0.5 10
```

```
0.0 0.0 0.0 10
0.0 0.0 1.0  1
```

```
$ $PATHQE/bin/dynmat.x -i pb.phband.in > pb.phband.out
```

Both the Fermi surface treatment in the self-consistent calculation and the wave-vector grid for Fourier interpolation nq1 nq2 nq3 keywords) have to be carefully chosen.
The phonon dispersions with nq1 nq2 nq3 set to 4 look already reasonable, but if nq1 nq2 nq3 are set to 6 you will notice some differences. If set to 3, results will look rather bad.



▶ Compute phonon DOS, using real-space IFC's and auxiliary code `matdyn.x`

pb.phdos.in

```
--
&input
flfrc='lead.fc'
asr='simple'
dos=.true.
nk1 = 8
nk2 = 8
nk3 = 8
fldos='lead.dos'
/
```

```
$ $PATHQE/bin/dynmat.x -i pb.phdos.in > pb.phdos.out
```

The file specified in fldos contains the DOS in a format that can immediately be plotted using `gnuplot.x`, or any other plotting program.