Quantum ESPRESSO: overview of the code

Davide Romanin

Department of Applied Sience (DISAT)
Politecnico di Torino, Torino

May 14, 2020

QUANTUMESPRESSO

Introduction

Quantum ESPRESSO is an integrated suite of Open-Source computer codes for electronic-structure calculations and materials modeling at the nanoscale. It is based on **density-functional theory**, **plane waves** and **pseudopotentials**.
The latest version can be downloaded from:
https://github.com/QEF/q-e/releases
To compile it, just type inside the folder:

```
$ ./configure
$ make all
```

For this tutorial we will use v.6.2. Feel free to use any other version. Be careful that the input files are written in different ways for different versions of the software (particularly true for old versions). In the hands-on tutorial of the next weeks I will use the 6.5 version (which is not that different from the 6.2).

Electronic structure: self-consistent cycle

The first step is to solve the Kohn-Sham self-consistent cycle. This is done by executing the program pw.x:

```
$ ../qe-6.2/bin/pw.x < scf.in > scf.out
```

where, 'scf.in' is the input file containing all the details of your simulation and 'scf.out' is the output file produced by quantum espresso.

Electronic structure: self-consistent cycle

Input file for pw.x

The input file is for pw.x executable (not only used for scf computations as we will see later). It is structured in a certain number of

- **NAMELISTS**: specify needed values and default values (according to the task), which are read in a specific order while those which are not required are simply ignored;
- **INPUT_CARDS**: provide data that are always needed and which have to be written in a specific order;

All quantities whose dimensions are not explicitly specified are in **RYDBERG ATOMIC UNITS**. Charge is intended as "number charge" (i.e. not multiplied by e). Potentials instead are in energy units (i.e. they are multiplied by e).

Electronic structure: self-consistent cycle

Input file for pw.x

There are three mandatory NAMELISTS:

- **&CONTROL** : input variables that control the flux of the calculation and the amount of I/O;
- **&SYSTEM** : variables that specify the system under study;
- **&ELECTRONS** : variables that control the KS self-consistent computation for electrons;

There are two more NAMELISTS which have to be specified for certain computations:

- **&IONS** : when relaxing the structure allowing ONLY the atoms to move;
- **&CELL** : when relaxing the structure allowing ONLY the cell-shape to remodel itself;

Electronic structure: self-consistent cycle

Input file for pw.x

There are three mandatory INPUT_CARDS:

- **ATOMIC_SPECIES** : input name, mass and pseudopotential for every atomic species in the cell;
- **ATOMIC_POSITIONS** : type and coordinates of each atom in the cell;
- **K_POINTS** : coordinates and weights of the k-points used for the Brillouin zone integration;

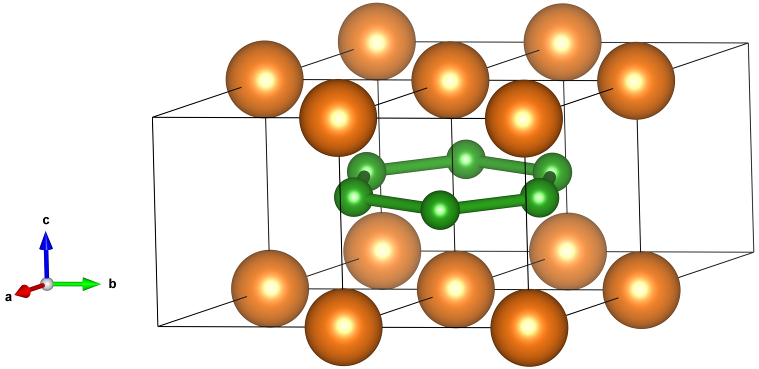
Electronic structure: self-consistent cycle

Input file for pw.x. Example: MgB_2

```
&CONTROL
calculation = 'scf',
prefix='MgB2',
pseudo_dir = './',
outdir = './tmp',
/
&SYSTEM
ibrav= 4,
celldm(1) = 5.808563789,
celldm(3) = 1.145173082,
nat=3,
ntyp= 2,
ecutwfc = 55,
ecutrho = 440,
occupations = 'smearing',
smearing = 'gaussian',
degauss = 0.02
/
&ELECTRONS
conv_thr = 1.0d-9
/
ATOMIC_SPECIES
B 10.811 B_pbe_v1.01.uspp.F.UPF
Mg 24.305 mg_pbe_v1.4.uspp.F.UPF
K_POINTS automatic
16 16 12 0 0 0
ATOMIC_POSITIONS crystal
B 0.666666667 0.333333333 0.50000
B 0.333333333 0.666666667 0.50000
Mg 0.000000000 0.000000000 0.00000
```

Electronic structure: self-consistent cycle

MgB_2 structure



Electronic structure: self-consistent cycle

Input file for pw.x. Example: MgB_2

```
&CONTROL
calculation = 'scf',
prefix='MgB2',
pseudo_dir = './',
outdir = './tmp',
/
```

- *calculation = 'scf'* tells pw.x to perform the self-consistent cycle computation (it can assume also other values, which we will see later);
- *prefix='MgB2'* is the label which will be given to every output file generated by quantum espresso in the output directory;
- *pseudo_dir = './'* is the place where the program can find the pseudopotential files;
- *outdir = './tmp'* is the output directory, if it doesn't exist QE will create it;

Electronic structure: self-consistent cycle

Input file for pw.x. Example: MgB_2

In **&CONTROL** you can also tell the program to print interatomic forces and stresses by adding:

- *tprnfor* = *.true.*, print forces;
- *tstress* = *.true.*, print stresses;

Electronic structure: self-consistent cycle

Input file for pw.x. Example: MgB_2

```
&SYSTEM
ibrav= 4,
celldm(1) = 5.808563789,
celldm(3) = 1.145173082,
nat=3,
ntyp= 2,
ecutwfc = 50,
ecutrho = 500,
occupations = 'smearing',
smearing = 'gaussian',
degauss = 0.02
/
```

- *ibrav=4* indicates the Bravais-lattice index;

- *celldm(i)* is the lattice parameter in the i.th direction given in a.u. ($1 a.u. = 0.529177$Å);
- *nat = 3* is the number of atoms in your cell;
- *ntyp = 2* indicates how many different atoms you have in your cell;
- *ecutwfc=50* indicates the kinetic energy cutoff (in Ry) for wavefunctions;
- *ecutrho=500* indicates the cutoff (in Ry) for charge density and it is usually 8 to 12 times ecutwfc;
- the last three lines are used only for metals and the value of *degauss* is given in Rydberg ($1Ry \approx 13.6$eV)

Electronic structure: self-consistent cycle

Input file for pw.x. Example: MgB_2

Some IBRAV values

- *ibrav=1* for cubic P (sc): celldm(1)=a;
- *ibrav=2* for cubic F (fcc): celldm(1)=a;
- *ibrav=3* for cubic I (bcc): celldm(1)=a;
- *ibrav=4* for Hexagonal and Trigonal P: celldm(1)=a, celldm(3)=c/a;
- *ibrav=6* for Tetragonal P (st): celldm(1)=a, celldm(3)=c/a;
- *ibrav=7* for Tetragonal I (bct): celldm(1)=a, celldm(3)=c/a;
- *ibrav=8* for Orthorombic P: celldm(1)=a, celldm(2)=b/a, celldm(3)=c/a;

Electronic structure: self-consistent cycle

EXTRA: Jellium Doping

If you want to uniformly dope your structure with electrons or holes, you can do this by adding to **&SYSTEM** the following command:

$$\text{tot\_charge} = \pm x$$

where you'll put $+$ if you want to dope with holes (i.e. you're removing electrons) or you put $-$ if you want to dope with electrons (i.e. you're adding electrons). Notice that x is the number of electrons you're adding/removing per unit cell: this means that it can also be a fractional number.

Electronic structure: self-consistent cycle

EXTRA: Jellium Doping

For example, suppose you want to dope uniformly a Si bulk structure with a certain percentage Y of boron (B). Assuming that all the boron contributes to the hole doping, the amount of charge you'll put in your unit cell will be: $X = Y/100$. Therefore, for a Si structure doped with 4% of B: $tot\_charge = +0.04$.

This procedure is called **VIRTUAL CRYSTAL** and it is valid for small percentage of uniformly distributed dopants ($\leqslant 10\%$).
Another way to simulate substitutional doping, insertion doping or vacancies in your system is by directly modifying the types of atoms in your cell. Although this is more precise and represents more realistic situations, it requires (most of the times) the use os **SUPERCELLS** which is more demanding from the computational point of view.

Electronic structure: self-consistent cycle

Input file for pw.x. Example: MgB_2

```
&ELECTRONS
conv_thr = 1.0d-6
/
```

Convergence threshold for selfconsistency: estimated energy error < conv_thr.
Note that conv_thr is extensive, like the total energy.

Electronic structure: self-consistent cycle

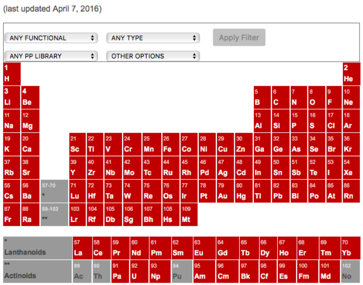
Input file for pw.x. Example: MgB_2

```
ATOMIC_SPECIES
B 10.811 B_pbe_v1.01.uspp.F.UPF
Mg 24.305 mg_pbe_v1.4.uspp.F.UPF
K_POINTS automatic
16 16 12 0 0 0
ATOMIC_POSITIONS crystal
B 0.666666667 0.333333333 0.50000
B 0.333333333 0.666666667 0.50000
Mg 0.000000000 0.000000000 0.00000
```

- in **ATOMIC_SPECIES**, for every atomic species in your cell, you have to specify the element symbol, the mass and the name of the pseudopotential you decided to use;
- in **K_POINTS** the *automatic* keyword automatically generates a uniform grid of k-points according to the Monkhorst-Pack grid you specify (16 16 12) and centered in the k-point you specified (0 0 0);
- in **ATOMIC_POSITION** the keyword *crystal* states that the atomic positions are given in terms of crystal coordinates (i.e. in relative coordinates of the primitive lattice vectors) and have to be declared for every atom in your cell;

Electronic structure: self-consistent cycle

Where to find pseudopotentials


(last updated April 7, 2016)

ANY FUNCTIONAL · ANY TYPE · ANY PP LIBRARY · OTHER OPTIONS

Apply Filter
1 H 2 He
3 Li 4 Be 5 B 6 C 7 N 8 O 9 F 10 Ne
11 Na 12 Mg 13 Al 14 Si 15 P 16 S 17 Cl 18 Ar
19 K 20 Ca 21 Sc 22 Ti 23 V 24 Cr 25 Mn 26 Fe 27 Co 28 Ni 29 Cu 30 Zn 31 Ga 32 Ge 33 As 34 Se 35 Br 36 Kr
37 Rb 38 Sr 39 Y 40 Zr 41 Nb 42 Mo 43 Tc 44 Ru 45 Rh 46 Pd 47 Ag 48 Cd 49 In 50 Sn 51 Sb 52 Te 53 I 54 Xe
55 Cs 56 Ba 57-70 * 71 Lu 72 Hf 73 Ta 74 W 75 Re 76 Os 77 Ir 78 Pt 79 Au 80 Hg 81 Tl 82 Pb 83 Bi 84 Po 85 At 86 Rn
87 Fr 88 Ra 89-102 ** 103 Lr 104 Rf 105 Db 106 Sg 107 Bh 108 Hs 109 Mt
* Lanthanoids 57 La 58 Ce 59 Pr 60 Nd 61 Pm 62 Sm 63 Eu 64 Gd 65 Tb 66 Dy 67 Ho 68 Er 69 Tm 70 Yb
** Actinoids 89 Ac 90 Th 91 Pa 92 U 93 Np 94 Pu 95 Am 96 Cm 97 Bk 98 Cf 99 Es 100 Fm 101 Md 102 No

You can find pseudopotentials at http://www.quantum-espresso.org/pseudopotentials/ Select the element you want, the kind of potential you want to use and save it in the same directory you have specified in the &CONTROL namelist. They can be generated through LDA, GGA or hybrid exchange functionals. There are different kinds of pseudopotentials and they mainly divide into *norm conserving* and *ultra soft*. Moreover they can also be *non relativistic* or *relativistic*.

Another good library for pseudopotentials is the one made by the EPFL, the Standard Solid-State Pseudopotentials (SSSP): *http://materialscloud.org/sssp*

Electronic structure: band structure computation

Electronic Band Structure

The second step consists into computing electronic bands of your material along a selected k-path in the Brillouin zone. This is done by executing again the program pw.x: $ ../qe-6.2/Pw/src/pw.x < bands.in > bands.out

where, 'bands.in' is the input file containing all the details of your simulation and 'bands.out' is the output file produced by quantum espresso.

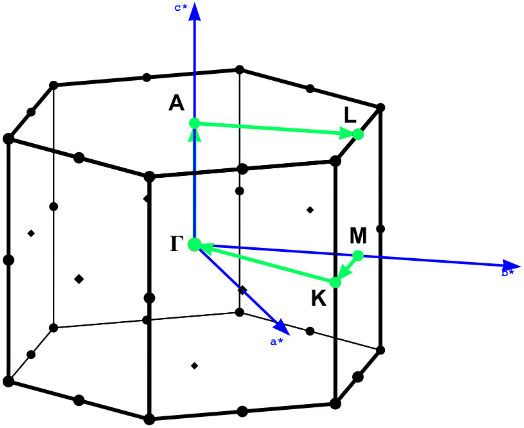
Electronic structure: band structure computation

Input file for pw.x. Example: MgB_2

```
&CONTROL
calculation = 'bands',
prefix='MgB2',
pseudo_dir = './',
outdir = './tmp',
/
&SYSTEM
ibrav= 4,
celldm(1) = 5.808563789,
celldm(3) = 1.145173082,
nat=3,
ntyp= 2,
ecutwfc = 50,
ecutrho = 500,
occupations = 'smearing',
smearing = 'gaussian',
degauss = 0.02
nbnd = 16
/
&ELECTRONS
conv_thr = 1.0d-6
diago_full_acc = .true.
/
ATOMIC_SPECIES
B 10.811 B_pbe_v1.01.uspp.F.UPF
Mg 24.305 mg_pbe_v1.4.uspp.F.UPF
K_POINTS crystal_b
6
0.00000 0.00000 0.00000 100
0.00000 0.50000 0.00000 100
0.33333 0.33333 0.00000 100
0.00000 0.00000 0.00000 100
0.00000 0.00000 0.50000 100
0.00000 0.50000 0.50000 100
ATOMIC_POSITIONS crystal
B 0.666666667 0.333333333 0.50000
B 0.333333333 0.666666667 0.50000
Mg 0.000000000 0.000000000 0.00000
```

Electronic structure: band structure computation

MgB_2 Special k-points



Electronic structure: band structure computation

Input file for pw.x. Example: MgB_2

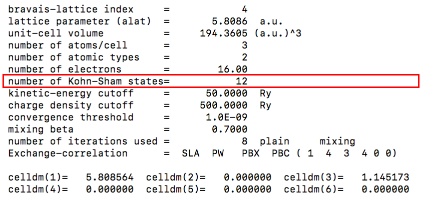
```
&CONTROL
calculation = 'bands',
prefix='MgB2',
pseudo_dir = './',
outdir = './tmp',
/
&SYSTEM
ibrav= 4,
celldm(1) = 5.808563789,
celldm(3) = 1.145173082,
nat=3,
ntyp= 2,
ecutwfc = 50,
ecutrho = 500,
occupations = 'smearing',
smearing = 'gaussian',
degauss = 0.02
nbnd = 16
/
```

- In **&CONTROL** be careful to use the same prefix and the same outdir you used for scf computations;
- In **&SYSTEM** we added *nbnd=16* which states how many bands (valence + conduction) the program has to compute: if you don't specify it, the program will just compute bands up to the highest occupied level;

Electronic structure: band structure computation

Choosing **nbnd**. Example: MgB_2

How can I decide the number of bands in **nbnd**? If we look at the output of the SCF cycle (*scf.out*):



```
bravais-lattice index     =            4
lattice parameter (alat)  =       5.8086  a.u.
unit-cell volume          =     194.3605 (a.u.)^3
number of atoms/cell      =            3
number of atomic types    =            2
number of electrons       =        16.00
number of Kohn-Sham states=           12
kinetic-energy cutoff     =      50.0000  Ry
charge density cutoff     =     500.0000  Ry
convergence threshold     =      1.0E-09
mixing beta               =       0.7000
number of iterations used =            8  plain     mixing
Exchange-correlation      =  SLA  PW   PBX  PBC ( 1  4  3  4 0 0)

celldm(1)=   5.808564  celldm(2)=   0.000000  celldm(3)=   1.145173
celldm(4)=   0.000000  celldm(5)=   0.000000  celldm(6)=   0.000000
```

number of Kohn-Sham states tells you how many bands below the highest occupied level (included) will be automatically computed in the band computation. If you want to show unoccupied bands, add the desired number starting by this value.

Electronic structure: band structure computation

Input file for pw.x. Example: MgB_2

```
&ELECTRONS
conv_thr = 1.0d-6
diago_full_acc = .true.
/
K_POINTS crystal_b
6
0.00000 0.00000 0.00000 100
0.00000 0.50000 0.00000 100
0.33333 0.33333 0.00000 100
0.00000 0.00000 0.00000 100
0.00000 0.00000 0.50000 100
0.00000 0.50000 0.50000 100
```

- In **&ELECTRONS** the line *diago_full_acc = .true.* requires the exact computations of all bands, if you don't put it only the bands up to the highest occupied level are computed exactly while the others are computed using a larger error threshold;

- In **K_POINTS** the flag *crystal_b* says that the k-points are given in crystal coordinates and every couple of points identifies the initial and final point of a line with N intermediate points of the line, where N is the weight of the first point. Then the first line ('6') tells how many special k-points in the Brillouin zone you want and the following lines give the coordinates and weight of these points ($x_{n_{k,i}}$ $y_{n_{k,i}}$ $z_{n_{k,i}}$ N);

Electronic structure: band structure computation

Plotting bands

After that we would like some plottable file (using for example gnuplot or xmgrace). In order to do so we execute:

```
$ ../qe-6.2/bin/bands.x < bande.in > bande.out
```

where, 'bande.in' is the input file containing all the details of your simulation and 'bande.out' is the output file produced by quantum espresso. This program reads the previous electronic structure computation from './tmp' and collect all the informations into a plottable file.

Electronic structure: band structure computation

Input file for bands.x, Example: MgB2

```
&BANDS
outdir ='./tmp',
prefix ='MgB2',
filband='bands_MgB2.d'
&end
```

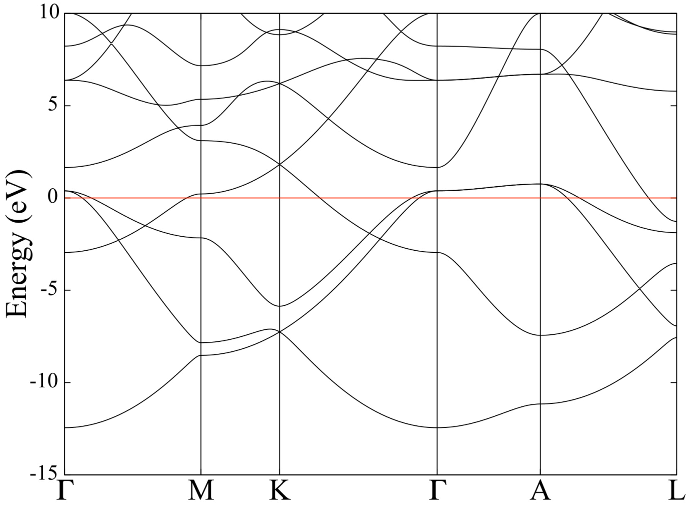
- Be careful to use the same outdir and the same prefix as before;
- The line *filband='bands_MgB2.d* specifies the name of the plottable file you want;

In order to obtain the final image you can use the interactive executable **plotband.x**, which does not require any input or output. Just answer the questions it will provide and at the end you will get a .gnu (or .xmg) file and a .ps file. Otherwise you can directly plot it on gnuplot or xmgrace.

The coordinates for the special k-points in the Brillouin zone can be found in the output of bands.x (*bande.out*).

Electronic structure: band structure computation

MgB_2 band structure



Electronic structure: DOS and Fermi surface

Non-self-consistent cycle

The last step consists into computing the electronic density of states and the Fermi surface of your material . This is done by executing again the program pw.x this time with a non-self-consistent computation:

$ ../qe-6.2/bin/pw.x < nscf.in > nscf.out where, 'nscf.in' is the input file

containing all the details of your simulation and 'nscf.out' is the output file produced by quantum espresso. This kind of computation takes as input charge density and potentials from the self-consistent computation, but reassemble those informations in suitable way for post-processing tasks.

Electronic structure: DOS and Fermi surface

Input file for NSCF computation, Example: MgB2

```
&CONTROL
calculation = 'nscf',
prefix='MgB2',
pseudo_dir = './',
outdir = './tmp',
/
&SYSTEM
ibrav= 4,
celldm(1) = 5.808563789,
celldm(3) = 1.145173082,
nat=3,
ntyp= 2,
ecutwfc = 50,
ecutrho = 500,
occupations = 'tetrahedra',
nbnd = 16
/
&ELECTRONS
conv_thr = 1.0d-6
diago_full_acc = .true.
/
ATOMIC_SPECIES
B 10.811 B_pbe_v1.01.uspp.F.UPF
Mg 24.305 mg_pbe_v1.4.uspp.F.UPF
K_POINTS automatic
32 32 24 0 0 0
ATOMIC_POSITIONS crystal
B 0.666666667 0.333333333 0.50000
B 0.333333333 0.666666667 0.50000
Mg 0.000000000 0.000000000 0.00000
```

Electronic structure: DOS and Fermi surface

Warnings about nscf computation

Two warnings:

- Whether you have a metal or an insulator, this time you always have to put *occupations='tetrahedra'* which is an optimal way of rearranging electrons occupation in the k space;
- For the k-points you have to put high numbers in order to have curves as smooth as possible;

Electronic structure: DOS and Fermi surface

Input files for dos.x and fs.x, Example:MgB2

Then, in order to obtain plottable files you have to run the following programs:

```
$ /qe-6.2/bin/dos.x < dos.in > dos.out
```

```
$ /qe-6.2/bin/fs.x < fermi.in > fermi.out
```

The DOS will be given in a gnuplot or xmgrace plottable file, while the Fermi surface will be given in a xcrysden plottable file '.bxsf'. Moreover you can only obtain Fermi surfaces for **METALS**.

DOS.IN

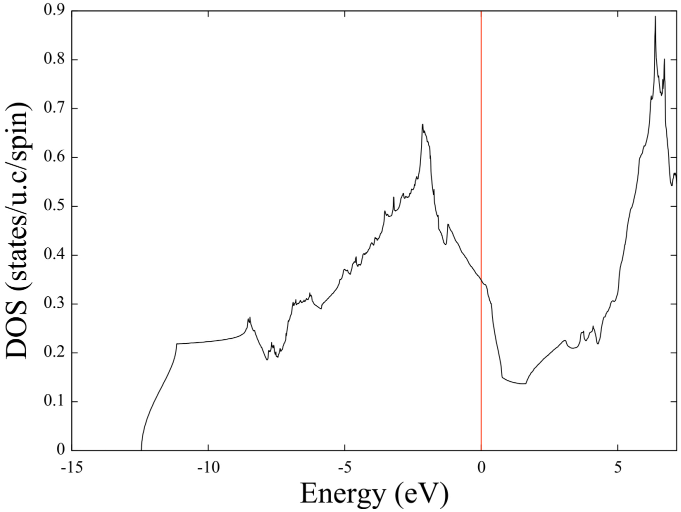
```
&DOS
outdir ='./tmp',
prefix ='MgB2',
&end
```

FERMI.IN

```
&FERMI
outdir ='./tmp',
prefix ='MgB2',
&end
```

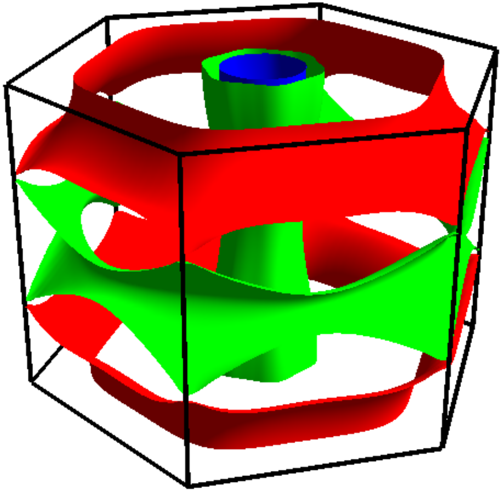
Electronic structure: DOS and Fermi surface

MgB$_2$ density of states



Electronic structure: DOS and Fermi surface

MgB_2 Fermi surface



Phononic properties: phonon dispersion relation and density of states

The first step is to solve again the Kohn-Sham self-consistent cycle. This is done as before:

$ /qe-6.2/bin/pw.x $<$ scf.in $>$ scf.out where 'scf.in' is the same input file containing all the

details of your simulation. Be careful to use an energy threshold smaller than the one used for electronic structure computations (at least $\leq 10^{-9}$).
Then we compute phonons and dynamical matrices on a grid of q-vectors by executing:

$ /qe-6.2/bin/ph.x $<$ ph.in $>$ ph.out where 'ph.in' is the input file containing the instructions

for the computations and 'ph.out' is the output produced by quantum espresso. ph.x will produce the files of the dynamical matrices (prefix.dyn1, prefix.dyn2, ...) on the specified grid of points. The file prefix.dyn0 contains the list of the inequivalent q points.

Phononic properties: phonon dispersion relation and density of states

Input file for ph.x, Example: MgB2

```
&inputph
tr2_ph=1.0d-12,
prefix='MgB2',
ldisp=.true.,
nq1=4, nq2=4, nq3=2,
amass(1) = 10.811,
amass(2) = 24.305,
outdir = './tmp',
fildyn='MgB2.dyn',
trans=.true.
/
```

- In *tr2_ph=1.0d-12* we put the error threshold for phonon computation: be careful to use the a smaller value than that you used for scf computations;
- *ldisp=.true.* tells ph.x to prepare output in order to compute dispersion relations;
- *nqi = 4 or 2* is the number of q points in one direction of the Brillouin zone;
- *trans = .true.* if true, the phonons are computed;

Phononic properties: phonon dispersion relation and density of states

Transformation of the dynamical matrices from **G**- to **R**-space

After that we compute the interatomic force constants (IFC):

$$C(R_l) = \frac{1}{N_q} \sum_n C(q_n) e^{iq_n R_l}$$

where $C(q_n)$ are matrices on a grid of q points NxNxN in reciprocal space (given by ph.x) and $C(R_l)$ are the interatomic force constants in a supercell NxNxN in real space.
We can do this by executing q2r.x:

```
$ /qe-6.2/bin/q2r.x < q2r.in > q2r.out
```

Phononic properties: phonon dispersion relation and density of states

Input file for q2r.x, Example: MgB2

```
&input
fildyn='MgB2.dyn',
flfrc='MgB2.fc',
zasr='simple'
/
```

- *fildyn* is the same as before, tells where to find the dynamical matrices ;
- *flfrc='MgB2.fc'* is the output file containing the interatomic force constants in real space;

Phononic properties: phonon dispersion relation and density of states

Phonon Dispersion Computations

Once the IFC's are calculated, it is possible to calculate phonon frequencies and eigenvalues at any wavevector, by just reconstructing the dynamical matrix. If the grid of q-vectors is dense enough, the "reconstructed" dynamical matrix is very close to what one gets by direct calculation, also for a q-vector that was not in the grid of q used to calculate the IFC's in real space. Let us call this a "Fourier interpolation".
We can do this by executing matdyn.x:

```
$ /qe-6.2/bin/matdyn.x < matdyn.in > matdyn.out
```

Phononic properties: phonon dispersion relation and density of states

Input file for matdyn.x, Example: MgB2

```
&input
amass(1) = 10.811,
amass(2) = 24.305,
flfrc='MgB2.fc',
flfrq='MgB2.freq',
q_in_band_form=.true.
asr = 'simple'
q_in_cryst_coord = .true.
/
6
0.00000 0.00000 0.00000 100
0.00000 0.50000 0.00000 100
0.33333 0.33333 0.00000 100
0.00000 0.00000 0.00000 100
0.00000 0.00000 0.50000 100
0.00000 0.50000 0.50000 100
```

- *flfrc='MgB2.fc'* is the file containing the interatomic force constants in real space;
- *flfrq='MgB2.freq'* is the output file containing phonon frequencies;
- *q_in_band_form=.true.* if .true. the q points are given in band form;
- *q_in_cryst_coord = .true.* if .true. input q points are in crystalline coordinates;
- the last lines specify the number of q points, their coordinates in the Brillouin zone and their respective weights;

Phononic properties: phonon dispersion relation and density of states

Phonon Dispersion Computations

Finally in order to get a plottable file, execute:

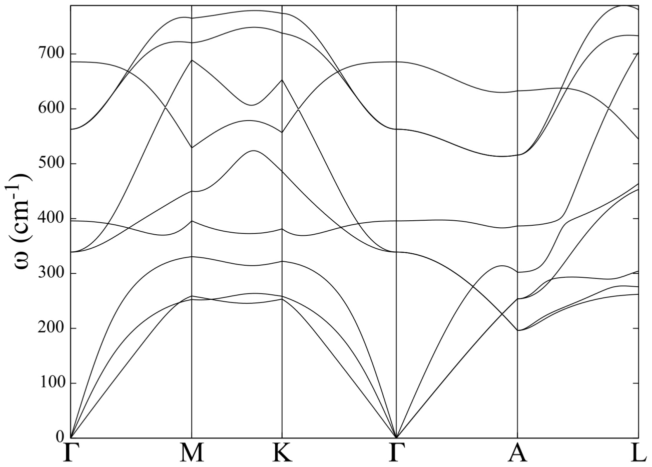
$ /qe-6.2/bin/plotband.x It is an interacting command in which you will have to specify:

```
Input file > DMND.freq
Reading 6 bands at 401 k-points
Range: 0.0000 1491.7566eV Emin, Emax > 0.0000, 1491.7566
high-symmetry point: 0.5000 0.5000 0.5000 x coordinate 0.0000
high-symmetry point: 0.0000 0.0000 0.0000 x coordinate 0.8660
high-symmetry point: 0.5000 0.0000 0.5000 x coordinate 1.5731
high-symmetry point: 0.3750 0.3750 0.7500 x coordinate 2.0408
high-symmetry point: 0.0000 0.0000 0.0000 x coordinate 2.9594
output file (gnuplot/xmgr) > dispersion.gnu
bands in gnuplot/xmgr format written to file dispersion.gnu
```

Stop the process by tapping the 'CTRL+C'.

Phononic properties: phonon dispersion relation and density of states

MgB_2 phonon dispersion relation



Phononic properties: phonon dispersion relation and density of states

Vibrational Density of States

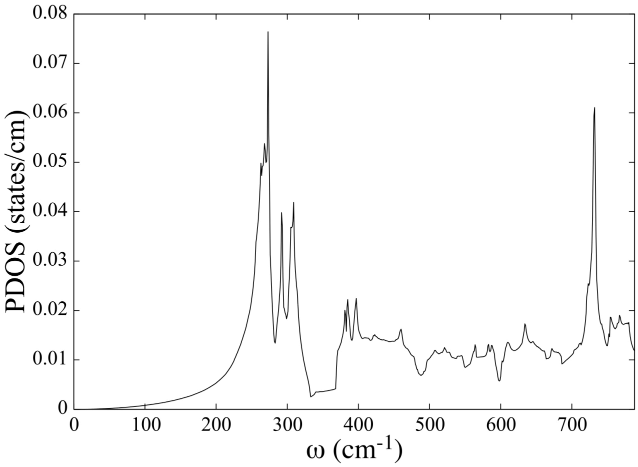
Using again matdyn.x we can also compute the phonon density of states. The input this time is:

```
&input
amass(1)= 12.011,
flfrc='MgB2.fc',
flfrq='MgB2.freq',
dos = .true.,
fldos = 'DMND.dos',
asr = 'simple',
nk1 = 32, nk2 = 32, nk3 = 32
/
```

- *dos = .true.* if .true. calculate phonon Density of States (DOS) using tetrahedra and a uniform q-point grid;
- *fldos = 'DMND.dos'* is the output file containing phonon DOS: be careful that it will be normalized to $3 * N_{at}$, where N_{at} is the number of atoms in the unit cell;
- *nki=32* number of points in the Brillouin zone along direction i;

Phononic properties: phonon dispersion relation and density of states

MgB_2 vibrational density of states



Conversion for Frequencies

Quantum Espresso usually expresses frequencies in cm^{-1}. You can translate them into electronvolts or Hertz by these eqivalences:

$$1\ \mathbf{THz} = 4.1357\ \mathbf{meV} = 33.356\ \mathbf{cm}^{-1}$$

Electron-phonon superconductivity: brief recap

We assume that our materials will show a superconductive phase transition due to electron-phonon interactions. In order to estimate the critical temperature will use Allen-Dynes formula:

$$T_C = \frac{\omega_{log}}{1.2} \exp\left\{ -\frac{1.04(1+\lambda)}{\lambda(1-0.62\mu^*)-\mu^*} \right\} \tag{1}$$

where μ^* is the Morel-Anderson pseudopotential (which will be treated as an external parameter whose typical range is 0.1-0.13) which describes retardation effects in the electron-phonon interaction. The other two parameters are ω_{log}, the logarithmic averaged phonon frequency (which is a more representative frequency of the system than Θ_D), and λ, the adimensional electron-phonon coupling constant. These two parameters will be computed ab-initio thorugh:

$$\omega_{log} = \exp\left\{ \frac{2}{\lambda} \int \frac{d\omega}{\omega} \alpha^2 F(\omega) \log\omega \right\} \tag{2}$$

$$\lambda = 2 \int \frac{d\omega}{\omega} \alpha^2 F(\omega) \tag{3}$$

$\alpha^2 F(\omega)$ is the Eliashberg's spectral function which can be computed by:

$$\begin{aligned}\alpha^2 F(\omega) = & \frac{1}{N_\sigma(E_F) N_q N_k} \sum_{\mathbf{q}\nu} \delta(\hbar\omega - \hbar\omega_{\mathbf{q}\nu}) \cdot \\ & \cdot \sum_{\mathbf{k},n,m} \left| g^{\nu}_{\mathbf{k}n,\mathbf{k}+\mathbf{q}m} \right|^2 \delta(\epsilon_{\mathbf{k}n} - E_F) \delta(\epsilon_{\mathbf{k}+\mathbf{q}m} - E_F)\end{aligned} \tag{4}$$

where n and m are band indices, $N_\sigma(E_F)$ is the electronic density of states at the Fermi energy per spin, N_q and N_k are respectively the number of q-points and k-points used to discretize the Brillouin zone during the computation, ν is the index for vibrational modes, $\omega_{\mathbf{q}\nu}$ is the phonon frequency for momentum $\mathbf{q}$ and mode ν.
The electron-phonon matrix for mode ν, $g^{\nu}_{\mathbf{k}n,\mathbf{k}+\mathbf{q}m}$, describes the strength of the interaction between ions and electrons and it is given by:

$$g^{\nu}_{\mathbf{k}n,\mathbf{k}+\mathbf{q}m} = \frac{1}{\sqrt{2\omega_{\mathbf{q},\nu}}} \langle \mathbf{k}n | \frac{\delta V_{KS}}{\delta \mathbf{e}_{\mathbf{q},\nu}} | \mathbf{k}+\mathbf{q}m \rangle \tag{5}$$

where indices m, n are electronic band indices, $\mathbf{e}_{\mathbf{q},\nu}$ are the phonon eigenvectors or polarization vectors which tells the propagation direction of phonons, V_{KS} is the periodic part of the Kohn-Sham potential computed self consistently and which takes into account the electron-phonon and the electron-electron interactions, $\langle \mathbf{r} | \mathbf{k}n \rangle = u_{\mathbf{k}}(\mathbf{r})/\sqrt{N}$ are the Bloch eigenfunctions and N is the number of cells.

Procedural steps

- Perform a standard SCF computation with a very high number of k-points (at least 32x32x32) and add in the &SYSTEM namelist the option **la2F = .true.** ;
- Perform another SCF computation with the reduced number of k-points that assures you phonon convergence, this time without the option **la2F = .true.** ;
- Perform a phonon + electron-phonon computation, see the **name.ep.in** input below;
- Fourier transform your IFCs with q2r.x as usual, inserting the option **la2F = .true.** ;
- Calculate phonon linewidths with matdyn.x, see the **name.lw.in** input below ;
- Calculate electron-phonon coupling constant and the Eliashberg's spectral function with matdyn.x, see the **name.elh.in** input below ;
- Estimate the superconductive critical temperature with lambda.x, see the **lambda.in** input below;

Procedural steps

Input file for ph.x, MgB2.ep.in

```
&inputph
tr2_ph=1.0d-12,
prefix='MgB2',
ldisp=.true.,
nq1=4, nq2=4, nq3=2,
amass(1) = 10.811,
amass(2) = 24.305,
outdir = './tmp',
fildyn='MgB2.dyn',
fildvscf='MgB2.dvscf',
trans=.true.,
electron_phonon='interpolated',
el_ph_sigma=0.005,
el_ph_nsigma=10,
/
```

- In *el_ph_sigma* we put the value of the smearing used for convergence of electron-phonon computations;
- *el_ph_nsigma* tells the number of convergence steps;
- *fildvscf* saves the functional derivative of the periodic part of Kohn-Sham potential;

Procedural steps

Input file for matdyn.x, MgB2.lw.in

```
&input
amass(1) = 10.811,
amass(2) = 24.305,
flfrc='MgB2.fc',
flfrq='MgB2.freq',
dos = .false.,
la2F = .true.,
asr = 'simple'
q_in_cryst_coord = .true.
/
6
0.00000 0.00000 0.00000 0.0
0.00000 0.50000 0.00000 0.0
0.33333 0.33333 0.00000 0.0
0.00000 0.00000 0.00000 0.0
0.00000 0.00000 0.50000 0.0
0.00000 0.50000 0.50000 0.0
```

- the last lines specify the number of q points, their coordinates in the Brillouin zone and their respective weights which are set to zero this time;

Procedural steps

Input file for matdyn.x, MgB2.elh.in

```
&input
amass(1) = 10.811,
amass(2) = 24.305,
flfrc='MgB2.fc',
flfrq='MgB2.freq',
dos = .false.,
la2F = .true.,
asr = 'simple'
dos = .true.
fildos='phonon.dos',
nk1=10,
nk2=10,
nk3=10,
ndos=50
/
```

Procedural steps

Input file for lambda.x, lambda.in

10 0.12. 1. *! emax (something more than highest phonon mode in THz), degauss, smearing method*
4 *! Number of q-points for which EPC is calculated*
0.00000 0.00000 0.00000 1.0 *! q-points and their weight*
0.33333 0.33333 0.33333 8.0
0.00000 0.66667 0.00000 6.0
0.66667 0.00000 0.66667 12.0
elph_dir/elph.inp_lambda.1 *! elph output file names*
elph_dir/elph.inp_lambda.2 *! in the same order as the q-points before*
elph_dir/elph.inp_lambda.3
elph_dir/elph.inp_lambda.4

0.10 *!* μ^*