cpw2000 User Guide

Version 5.12, June 2025

José Luís Martins
(Retired from) Departamento de Física, Instituto Superior Técnico,
Lisboa, Portugal

INESC MN, Lisboa, Portugal

`jlmartins@inesc-mn.pt`
`jose.l.martins@tecnico.ulisboa.pt`

Contents

1 Introduction

This is the current version of a old code that calcultes the electronic structure of crystals within the pseudopotential approximation with a plane-wave basis.

The very original code was written by Sverre Froyen at the University of California Berkeley. I do not know what was the first time his version of the code, as there was already an earlier pseudopotential code at Berkeley, I would guess that Physical Review B, 26, 3258 (1983) https://journals.aps.org/prb/abstract/10.1103/PhysRevB.28.3258 already used that version.

After a long conversation with Roberto Car about the seminal Car-Parrinello work, I wrote at Berkeley in Marvin Cohen's group the code for the iterative diagonalization of the plane-wave matrices Physical Review B 37, 6134 (1988).

Since then many people contributed to the code. ADD THEM

2 Installation

2.1 Step 1: Downloading and extracting the archive

The code is available from GitHub

`https://github.com/jlm785/cpw2000`

If you downloaded the `cpw2000-5.x.y.tar.gz` file (where `x` and `y` are the minor version numbers) just extract it,

```
$ tar xzf cpw2000-5.x.y.tar.gz
```

and you will have a `cpw2000-5.x.y` directory

If you cloned the git you already have the relevant code in the main directory.

2.2 Step 2: Generating the documentation

The documentation is in the `cpw2000-5.x.y/Doc` directory. It include this file, and the means to generate a detailed description of the code for developers.

For that detailed description you need doxygen (`https://www.doxygen.nl`) and graphviz (`https://graphviz.org/`), both are available on most distributions. To check that your computer has doxygen and graphviz installed run the commands

```
cpw2000-5.x.y/Doc$ dpkg -s doxygen
```

```
cpw2000-5.x.y/Doc$ dpkg -s graphviz
```

on a Debian based distribution (Ubuntu et al), or use equivalent tools (yum, rpm, dnf zypper,...) or the relevant GUI.

If they are not available just install them

```
cpw2000-5.x.y/Doc$ sudo apt install doxygen
```

```
cpw2000-5.x.y/Doc$ sudo apt install graphviz
```

again for Debian based distros, or use equivalent tools (yum, rpm, dnf zypper,...) or the relevant GUI.

Finally run doxyfile in the doxy directory

```
cpw2000-5.x.y/Doc/doxy$ doxygen Doxyfile
```

If you know what you are doing you can edit `cpw2000-5.x.y/Doc/doxy/Doxyfile`.

Finally it is useful to create a link to the file `cpw2000-5.x.y/Doc/html/index.html`

```
cpw2000-5.x.y/Doc$ ln -s html/index.html documentation.html
```

Opening that link in your browser will allow you to see the documentation for (almost) every file in the code.

2.3 Step 3: Compiling the code

The code has been tested with several Fortran compilers, `ifort` (from Intel oneAPI), `gfortran` (from gnu), `pgfortran` (from Portland), and even LLVM (experimental Intel compiler).

The choice of compiler is in the `cpw2000-5.x.y/Src/make.inc` file. You may edit it for your convenience, but in principle you have to make just two choices. The first is to identify which CPU you are using, as it narrows the choice of possible compilers. This is done by commenting/uncommenting the lines below `# compiler for the job`.

The second is the compiler you want, and is the key decision. Again comment/uncomment the lines below `# Suggestions for compilers`.

This `make.inc` file was adapted from the wannier90 distribution. As compilers may evolve with time, check their documentation and the wannier90 distribution if you run into problems.

For any other modification of the `make.inc` file it is assumed you know what you are doing...

To compile just go to `cpw2000-5.x.y/Src` and type make

```
cpw2000-5.x.y$ cd Src
cpw2000-5.x.y/Src$ make
```

You should get the main library `libcpw_`*compiler*`.a` and a few executables `cpw_`*compiler*`.exe`, etc... where *compiler* is the name of the chosen compiler. As you probably are not interested in comparing compilers, modify the script `copyit` to simplify the names and relocate the executables. In my case I just copy them to `cpw2000-5.x.y` removing the compiler name.

3 First time run of the code

I assume you have a working directory separate from the source. In the following I will call it `cpw2000-5.x.y/WORK`

3.0.1 Files required to run the code

You will need a file with the crystal description and a pseudopotential file for each type of atom in the crystal.

The file with the crystal description is called `cpw.in`. You must have such a file in your working directory. You can find such a file for almost all elements in `cpw2000-5.x.y/Structures/Elements` and files for other structures in the other subdirectories of texttcpw2000-5.x.y/Structures. The format of the crystal description is the same as in the SIESTA code, textttthttps://siesta-project.org/siesta/index.html so you may find descriptions for other crystals in other places.

There is also a tool to obtain a `cpw.in` file by answering a few questions.

You will have to generate the pseudopotentials for each atom, or find someone who has done that for you and have a file in the relevamt format. The name of the file is Xy_POTKB_F.DAT where Xy is the one or two character chemical symbol. This file should be (or at least a link) on your working directory.

The good news is that the code to generate pseudopotentials is available from GitHub

`https://github.com/jlm785/pseudopotential`

and if you run the test on `Validation` after you followed the relevant instructions you will get a pseudopotential for all elements.

The bad news is that some of those pseudopotentials where not extensively tested, and even those that are reported as tested may not be what you want. Just test them in a simple case before you proceed. As pseudopotentials calculations should reproduce first principles calculations in `cpw2000-5.x.y/Structures/Elements` you can find `elk.in` files that you can use to run LAPW all-electron calculations with the elk code, `https://elk.sourceforge.io/`. Testing the pseudopotential will spare you a lot of grief later.

If you want just a "quick and dirty" calculation than you can use the pseudopotentials from `Validation` of the pseudopotential code.

3.1 Running the code

Once you have the required files you run the code with

```
cpw2000-5.x.y/WORK$ ../cpw.exe
```

if you created the appropriate links, or

```
cpw2000-5.x.y/WORK$ ../Src/cpw_compiler.exe
```

otherwise. The code will write to default output, so redirect it with

```
cpw2000-5.x.y/WORK$ ../cpw.exe > output.dat
```

or

```
cpw2000-5.x.y/WORK$ ../cpw.exe | tee output.dat.
```

You will get a `PW_RHO_V.DAT` file with the self-consistent potential. By running

```
cpw2000-5.x.y/WORK$ ../cpw_post_process.exe
```

you will start an interactive analysis of the results.

If both executables ran without errors you are set.

4 The `cpw.in` file

The `cpw.in` syntax is similar to the input file from the SIESTA code. The parser is different, `esdf` instead of `fdf`. The keywords are even the same when possible. Therefore part of the text of this section was pilfered from the SIESTA documentation.

The `cpw.in` contains all the physical data of the system and the parameters of the simulation to be performed.

This file is written in a special format called ESDF, developed by Chris J. Pickard. This format allows data to be given in any order, or to be omitted in favor of default values. It is used in other electronic structure codes such as CASINO and PARSEC. However to be consistent with SIESTA the special character indicating a block structure has been changed.

Here we offer a glimpse of the rules:

- The syntax is a 'data label' followed by its value. Values that are not specified in the datafile are assigned a default value.
- The labels are case insensitive, and characters - _ . in a data label are ignored. Thus, LatticeConstant and lattice_constant represent the same label.
- All text following the # character is taken as comment.
- Logical values can be specified as T, true, .true., yes, F, false, .false., no.
- Character strings should **not** be in apostrophes.
- Real values which represent a physical magnitude must be followed by its units. It is important to include a decimal point in a real number to distinguish it from an integer, in order to prevent ambiguities when mixing the types on the same input line.
- Complex data structures are called blocks and are placed between '%block label' and a '%endblock label' (without the quotes).
- If the same label is specified twice, the first one takes precedence.
- If a label is misspelled it will not be recognized (there is no internal list of "accepted" tags in the program). You can check the actual value used by cpw2000 by looking for the label in the output.

This is an example for silicon:

```
LatticeConstant            10.2629     bohr

%block LatticeVectors
         0.00000000        0.50000000         0.50000000
         0.50000000        0.00000000         0.50000000
         0.50000000        0.50000000         0.00000000
%endblock LatticeVectors

NumberOfSpecies               1

NumberOfAtoms               2

%block Chemical_Species_Label
               1       14   Si
%endblock Chemical_Species_Label

AtomicCoordinatesFormat     Fractional

%block AtomicCoordinatesAndAtomicSpecies
         0.12500000       0.12500000       0.12500000       1     #  Si       1
        -0.12500000      -0.12500000      -0.12500000       1     #  Si       1
```

```
%endblock AtomicCoordinatesAndAtomicSpecies

StructureSource              Experiment

#-------------------------------------------------
# Energy cutoff, bands,  and Brillouin mesh
#-------------------------------------------------

PWEnergyCutoff                  12.0000      hartree

NumberOfEigenStates                  10

%block kgrid_Monkhorst_Pack
              4       0       0       0.500000
              0       4       0       0.500000
              0       0       4       0.500000
%endblock kgrid_Monkhorst_Pack
```

4.1 General stuff

- **SystemLabel**

  A *single* word (max. 20 characters *without blanks*) containing a nickname of the system. Reserved to be used to name output files in the future.

- **PrintingLevel** (integer)

  Defines the detail of the printout. May take the values 1,2, or 3. The higher the value the more details will be printed. Remember that too much information is noise. 1 is recommended for molecular dynamics. 3 for single calculation, or when things seem to go wrong.

4.2 Crystal descriptors

These are the lines that describe the crytal structure.

- **LatticeConstant** (length)

  A physical value, requiring a real number followed by the units. Accepted units are bohr and angstrom. This is just to define the scale of the lattice vectors.

  Default value: One bohr.

- **LatticeVectors** (block)

  The cell vectors are read in units of the lattice constant defined above. They are read as a matrix, each vector being one line.

  Default value: Unit matrix.

  The internal representation of the lattice vectors in the code is by their metric tensor. So the orientation is space is lost! See the section on output.

- **NumberOfSpecies** (integer)

Number of different atomic species in the simulation. It must be the number of lines in the **ChemicalSpeciesLabel** block.

Default value: 1

In case of inconsistency the execution terminates.

- **NumberOfAtoms** (integer)

  Number of atoms in the simulation. It is the number of lines in the block **AtomicCoordinatesAndAtomicSpecies**.

  Default value: 1

  In case of inconsistency the execution terminates.

- **ChemicalSpeciesLabel** (block)

  It specifies the different chemical species that are present, assigning them a number for further identification. cpw2000 recognizes the different atoms by the given atomic number.

  One line for each species. The first number in a line is the species number, it is followed by the atomic number, and then by the chemical symbol. From H to Og all chemical symbols are recognized. There is an extra chemical symbol, ZZ with a number of protons (atomic number) that can be zero to allow adding special pseudopotentials to the crystal.

- **AtomicCoordinatesFormat** (string)

  Character string to specify the format of the atomic positions in input.

  - Fractional
    Atomic positions are given referred to the lattice vectors

- **AtomicCoordinatesAndAtomicSpecies** (block)

  Block specifying the position and species of each atom. One line per atom, with three reals followed by one integer. In total the number of lines indicated by **NumberOfAtoms** must be present.

  The three reals indicate the atomic positions in the format specified by **AtomicCoordinatesFormat** followed by the species of atom on that position, as identified in the **Chemical_Species_Label** block.

  In case of inconsistency the execution terminates.

- **StructureSource** (string)

  Information on the source of the crytal structure.

4.3 Major Self Consistent Field

This are the parameters whose values you would mention in a paper to allow reproducibility.

- **PWEnergyCutoff** (energy)

  Energy cutoff of the plane wave basis set expansion. Real value followed by the energy unit (hartree).

- **NumberOfEigenStates** (integer)

  This parameter indicates the number of eigenstates that are calculated. By choosing a low value the cost of the diagonalization may be reduced by finding fewer eigenstates. However choosing a slightly larger number of active eigenstates than the bare minimum may help converge faster the occupied eigenstates and therefore the overall calculation. Note, that if the electronic temperature is greater than zero then the number of partially occupied states increases, depending on the band gap. The value specified must be greater than the number of occupied states (at least the number of electrons divided by two for a non-spin-polarized calculation) and less than the number of basis functions (which is extremly large for plane waves).

- **kgridMonkhorstPack** (block)

  Specifies the Fourier integration grid, known in the literature as the Monkhorst-Pack grid, for the Brillouin zone integration. It is just the good old Gauss quadrature method with sine/cosine functions chosen for periodic functions.

  Specified as an integer 3×3 matrix and a real vector. At present only the diagonal elements of the matrix are relevant, but this format will alow an extension of the method in the future.

  It has three lines, each with three integers and a real.

$$\begin{array}{llll} m_{11} & m_{12} & m_{13} & \quad d_1 \\ m_{21} & m_{22} & m_{23} & \quad d_2 \\ m_{31} & m_{32} & m_{33} & \quad d_3 \end{array}$$

  In the direction j the Brillouin zone will be divided in m_{jj} sections and a point with a shift of d_j will be selected.

  To use only the Γ point (molecule in a supercell) use

$$\begin{array}{llll} 1 & 0 & 0 & \quad 0.0 \\ 0 & 1 & 0 & \quad 0.0 \\ 0 & 0 & 1 & \quad 0.0 \end{array}$$

  It is usual to have m_{jj} an even number and $d_j = 0.5$. For hexagonal crystals and the conventional axis, it is also usual/convenient to have $m_{11} = m_{22}$ a multiple of three. These are recipes that minimize the number of irreducible points and avoid breaking symmetry.

- **XC.Authors** (string)

  Particular parametrization of the exchange-correlation functional. Options are:

  - CA

    Local density approximation (LDA). Quantum Monte Carlo calculation of the homogeneous electron gas by D. M. Ceperley and B. J. Alder, Phys. Rev. Lett. **45**,566 (1980), as parametrized by J. P. Perdew and A. Zunger, Phys. Rev B **23**, 5075 (1981)

  - PBE

    GGA of J. P. Perdew, K. Burke and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996)

  - TBL

    Meta-GGA of Tran and Blaha. F. Tran and P. Blaha, Phys. Rev. Lett. 102, 226401 (2009)

- **Xc.TBL.C** (real)

  Sets Tran-Blaha constant if a positive value. If it is negative, the original constant is used. A value around 1.09 is usualy a good choice. It can be used to "fine-tune" the band gap in simulations (second-priciples calculations).

- **TypeOfScfDiag** (string)

  Indicates how the SCF was performed.

  - PW
    The full plane-wave basis set. It is the "usual" choice.
  - AO
    Uses the atomic orbitals included in the pseudopotential file. Not available in very old files. It corresponds to a Linear Combination of Atomic Orbitals (LCAO) calculation.
  - AOJC
    Uses the atomic orbitals included in the pseudopotential file, but improves the LCAO wave-functions with a single Jacobian Correction. It is very convenient to explore the Born-Oppenheimer energy surface in optimization and molecular dynamics. However one may want to check the *final* results with a full PW calculation.
  - AOJCPW
    Follows an AOJC calculation with a full PW calculation. Final results are the same as the normal PW calculation, but may be faster, depending on the case.

- **DualApproximation** (boolean)

  Use the dual approximation. It uses a smaller grid for the calculation of the effective potential. Speeds the calculations with a compromise on precision. For molecular dynamics it is very safe. For geometry optimization with small energy cutoffs may not be accurate enough.

- **ElectronicTemperature** (temperature)

  Occupy orbitals with a Fermi-Dirac distribution with that temperature.

  Temperature value and unit (K).

4.4 Minor Self Consistent Field parameters

Other parameters of self-consistency that have less impact, unless "very wrong" choices are made. Probable safe to leave at default values.

- **MaxSCFIterations** (integer)

  Maximum number of self-consistent iterations. If very large, computing time can be wasted in the cases convergence is not achieved. If very small the code can terminate without a result, wasting computer time.

- **ScfTolerance** (real)

  The SCF iterations are deemed converged if the difference between the input aroundoutput values of all the components of the Fourier transform of the effective potential are smaller than this treshold.

- **DiagTolerance** (real)

  The iterative diagonalization is deemed converged if the module of the error vector is smaller than that value.

4.5 Unused Self Consistent Field parameters

Over the years only one option was left in the code.

- **TypeOfPseudoMixing** (string)

  Type of effective potential mixing used to accelerate convergence.

  Broyden 1st (BROYD1) type is the only available option.

- **TypeOfPseudopotential**

  Type of pseudopotential that is used.

  Kleinman-Bylander (PSEUKB) separable pseudos are the only available option

4.6 Molecular dynamics parameters

- **MD.TypeOfRun** (string)

  Choice on how the atoms move. Molecular dynamics or geometry optimization.

  - ONE

    Just one SCF calculation. Atoms do not move.
  - RSTRT

    Restart the molecular dynamics from the last atomic configuration. Use in case it was interrupted by some external reason (power failure). Just keep all the other parameters the same. It will reproduce what a non-interrupted calculation would find in the case of molecular dynamics. For optimization the minimization restarts, so it will not reproduce a non-interrupted calculation, but probably will converge to the same minimum.
  - MICRO

    Microcanonic molecular dynamics. The total/free energy is conserved, depending whether the temperature is zero and the system is an insulator or temperature is finite and not too low in the case of a metal. How low the temperature can go depends on the density of integration k-points.
  - LANG

    Langevin molecular dynamics, in contact with a thermostat with temperature indicated by **MD.TargetTemperature**.
  - VCSLNG

    Langevin molecular dynamics but with a variational cell shape, that is the lattice vectors change with simulation time.
  - LBFSYM

    Minimization of the energy with respect to atomic positions with the LBF algorithm.
  - VCSLBF

    Minimization of the energy with respect to atomic positions and cell shape with the LBF algorithm.
  - EPILBF

    Minimization of the energy with respect to atomic positions and "vertical" dimension of the cell with the LBF algorithm. It models an epitaxial situation. The initial $\vec{a}_1$ and $\vec{a}_2$ of **LatticeVectors** define the epitaxial surface. The cell only contracts and expands in the direction perpendicular to those vectors ($\vec{b}_3$).

- **MD.InitialTemperature** (temperature)

  The initial kinetic energy is set according to the temperature (K). If the initial potential energy is high, the temperature of the system may be quite higher. If unsure, first do a thermalization with Langevin, otherwise the system may break apart. Simulations with **TypeOfScfDiag** as AO or AOJC are great for fast thermalization.

- **MD.NumberOfSteps** (integer)

  Number of steps of the molecular dynamics run or maximum number of optimization steps.

- **MD.LengthTimeStep** (time)

  Time step for molecular dynamics (fs). It too small atoms barely move, if too large the integrator of the molecular dynamics (Verlet) becomes unstable. Instability appears in the non-conservation of energy. Few femtoseconds should be OK, but remember that light atoms move faster (square root of mass scaling). Simulations with **TypeOfScfDiag** as AO or AOJC are great for initial checks.

- **MD.TargetTemperature** (temperature)

  For molecular dynamics with a Langevin thermostat, it is the temperature (K) of the thermal bath.

- **MD.FrictionFracInvTimeStep** (real)

  The friction coefficient for simulations with a Langevin thermostat. It is set with respect to the time step. It gives how many time steps occur until some thermalization is achieved. If small the thermalization is of bad quality, if large it will take a long time to thermalize.

- **MD.TargetPressure** (pressure)

  In a molecular dynamics with variational cell shape, or cell optimiztion, it is the applied pressur (GPa).

- **MD.TargetStress** (block)

  Real 3×3 matrix of an applied stress in addition to the applied pressure (GPa). The matrix should be symmetric, and is in lattice coordinates, so some effort is needed to understand the orientation of the crystal.

- **MD.CellMass** (real)

  Fictitious cell mass for variational cell shape molecular dynamics. It is in units of electron mass.

- **MD.Seed** (integer)

  Seed for the pseudo-random generator for the thermostat and iniial velocity. Same seed will give the same trajectory. Different seeds will give different trajectories allowing trivial parallelization of simulations.

- **MD.UseKeatingCorrections** (boolean)

  For some tetrahedral semiconductors, it provides a correction to LDA that reproduces experimental bond lengths.

- **MD.UseFixedkplusG** (boolean)

  Keep a fixed $\vec{k}+\vec{G}$ expansion in a variational cell shape simulation to avoid the noise of the basis set changing during the simulation. Should only be used near equilibrium. abinit has a neater way of dealing with this noise. Use **MD.CG.FixedkplusGTol** to define when it kicks in.

- **MD.CG.Tolerance** (force)

  The optimization stops when the components of the forces (har/bohr) on all atoms are smaller than this value.

- **MD.CG.StepMax** (length)

  Maximum displacement of atoms to avoid instabilities in the optimization.

- **MD.CG.FixedkplusGTol** (force)

  The fixed $\vec{k} + \vec{G}$ is used *after* the components of the forces (har/bohr) on all atoms are smaller than this value.

4.7 Symmetry

Treatment of symmetry during a molecular dynamics or optimization run.

- **UseSymmetry** (boolean)

  Try to maintain the initial symmetry of the system. It is close to 100% fiable (as close as real number logic allows).

- **SymmTolerance** (real)

  If after a symmetry operation the difference between the positions (in lattice coordinates) is smaller than this value, the atoms are considered superposed.

4.8 Unfolding

If the structure is a supercell of some lattice it may be useful to plot the band structure in the "unfolded" Brillouin zone of the parent structure. The unfolding procedure was developed within an industrial collaboration (rede project), it was used in semiconductors with a parent fcc lattice, so unfolding was only extensively tested in that lattice.

- **Rede.Superlattice** (block)

  The block has three lines with three integers. Each line indicates how each of the supercell lattice vectors relate to the lattice vectors of the parent structure.

  Taking as an example the supercell described in `Structures/Supercells/cpw_Si6Ge6_Si(001).in`, which has 6 Si atomic layers and 6 germanium atomic layers on a (001) surface, the block is

```
%block Rede.Superlattice
          -1     1     0
           0     0     1
           3     3    -3
%endblock Rede.Superlattice
```

the underlying parent fcc structure has lattice vectors $\vec{A}_1 = (0, a/2, a/2)$, $\vec{A}_2 = (a/2, 0, a/2)$ and $\vec{A}_3 = (a/2, a/2, 0)$. The supercell vectors are $\vec{a}_1 = -\vec{A}_1 + \vec{A}_2 = (-a/2, a/2, 0)$, $\vec{a}_2 = \vec{A}_3 = (a/2, a/2, 0)$ and $\vec{a}_3 = 3\vec{A}_1 + 3\vec{A}_2 - 3\vec{A}_3 = (0, 0, 3a)$. Notice that the in-plane vectors have been rotated by 45°, their length is $\sqrt{2}a$ as the lattice constant is that of Si, it is consistent with epitaxy on Si (001). The third vector has a length slightly larger than $3a$ because it was allowed to relax in the presence of the larger Ge atoms.

- Rede.NumberOfLatticePlanes

  In a supercell grown on a surface indicates the number of lattice planes. It is used to help in plots to determine band alignments. In the above example you have six lattice planes (but twelve atomic planes). It must be present for unfolding to work. That is an "hack" to deal with back compatibility of some files, but do not influence the unfolding. This "requirement" should be removed in future releases.

5 Default output

The code writes to the default output, which can be redirected to a file. Depending on the choice of **PrintingLevel** the output will be different. We will use as a first example the file `Validation/Output/pw_ref_01.out`.

5.1 Initial information

On the top of the file we have the information about the version of code, and the date and time it was run.

```
density-functional pseudopotential plane-wave program version 4.99
run on the 19-Mar-21 at 13:30:54
```

notice that care was taken to code back the year 2000 bug in the 1990s.

Continuing with that example we find next some information about how many chemical species, and maximum number of atoms per chemical species the code is using.

```
 The values set by size_mxdtyp_mxdatm_esdf are:

 The value of mxdtyp is:     2
 The value of mxdatm is:        8

 input read from cpw.in
```

notice that the information about which subroutine printed the information is included in this case.

Next information are the lattice vectors,

```
 Primitive Translation Vectors
                      in a.u.                                  in lattice units
a1=  0.125697E+02  0.000000E+00  0.000000E+00          1.225    0.000    0.000
a2= -0.418990E+01  0.118508E+02  0.000000E+00         -0.408    1.155    0.000
a3=  0.000000E+00  0.000000E+00  0.725711E+01          0.000    0.000    0.707
```

preceded by a warning that they were changed,

```
   WARNING:   avec(  3,  3)
   changed from       0.707106780000 to       0.707106781187
```

informing that the code recognized a value very close to $\sqrt{2}/2$ and changed it.
The line

```
      Single geometry calculation
```

informs that **MD.TypeOfRun** was ONE.
To be reproducible the information needed to generate the pseudopotentials is printed

```
    potentials :
    ------------

    Si  ca  nrl  nc
    19-SEP-18 atom 5.804  Improved Troullier - Martinskb-loc= 2
    3s(  2.00)  rc= 1.993p(  2.00)  rc= 1.993d(  0.00)  rc= 1.99
    nql=4000 delql=   0.0100
```

see the user guide of the pseudopotential generation code for details.
Next the other chosen or default parameters are printed

```
    Local Density Approximation (LDA) using
 Ceperley and Alder as parametrized by Perdew and Zunger

 dual aproximation is used

    The energy cutoff for wave-function kinetic energy is      10.000 Hartree

    SCF is converged when the difference in potentials is less then     0.00001000
    Iterative diagonalizationis converged when the error in |h psi - e psi| is less then     0

    The temperature for electron Fermi distribution is      0.000Kelvin

     Plane-wave basis calculation
```

see previous section for the keywords associated.

5.2 Crystal data

In the output the initial crystal data is printed,

```
       CRYSTAL DATA


      1081.02690815      volume
```

```
   real-space metric

        157.99703193     -52.66567731       0.00000000      metric  g11,g12,g13
                         157.99703193       0.00000000      metric      g22,g23
                                           52.66567731      metric          g33

     12.56968703      12.56968703       7.25711219   length 1,2,3  (a.u.)

    109.47122063      90.00000000      90.00000000   angle 12,13,23 (degrees)

        position (lattice coord.)         position (cartesian coord. a.u.)     no. type

      0.50000    0.50000    0.00000     0.72571E+01  0.00000E+00  0.00000E+00   1   Si    posi
     -0.12500    0.62500    0.00000     0.36286E+01  0.76973E+01  0.00000E+00   2   Si    posi
      0.00000    0.00000    0.00000     0.00000E+00  0.00000E+00  0.00000E+00   3   Si    posi
      0.37500    0.12500    0.00000     0.36286E+01 -0.25658E+01  0.00000E+00   4   Si    posi
      0.00000    0.50000    0.50000     0.36286E+01  0.51316E+01  0.36286E+01   5   Si    posi
      0.50000    0.00000    0.50000     0.36286E+01 -0.51316E+01  0.36286E+01   6   Ge    posi
      0.37500    0.62500    0.50000     0.72571E+01  0.25658E+01  0.36286E+01   7   Ge    posi
     -0.12500    0.12500    0.50000     0.00000E+00  0.25658E+01  0.36286E+01   8   Ge    posi
```

in what should be self-explanatory.

Here we encounter a characteristic of the output, in that we have a value `1081.02690815` followed by a "key" `volume`. This allows a quick search for that key in the output. For example

```
cpw2000-5.x.y/WORK$ grep volume ../Validation/Output/pw_ref_05.out
```

will return the volumes encoutered in Langevin a molecular dynamics with variational cell shape

```
        843.03621271      volume
        844.20060174      volume
        847.56015618      volume
        852.95790991      volume
        860.21678359      volume
        869.14551064      volume
        879.54271598      volume
        891.20293979      volume
          ...
       1092.59744351      volume
       1091.08256914      volume
       1089.72581021      volume
       1088.52562153      volume
       1087.47977919      volume
       1086.58556009      volume
       1085.84052430      volume
       1085.24216798      volume
```

in a form that would be trivial to plot (for example with `gnuplot`).

5.3 Crystal symmetry and reciprocal space info

The code recognizes the space group operations, although does not find its conventional name (we will later describe how to find it), it lists the symmetry operations, in lattice coordinates, first the matrix and than the eventual fractional coordinate.

```
    rotation matrices and fractional translations in lattice coordinates

    1      1  0  0      0  1  0     0  0  1         0.0000000000     0.0000000000     0.00000000
    2      0 -1  0     -1  0  0     0  0 -1        -0.5000000000    -0.5000000000     0.00000000
    3      0 -1  0     -1  0  0     0  0  1        -0.5000000000    -0.5000000000     0.00000000
    4      1  0  0      0  1  0     0  0 -1         0.0000000000     0.0000000000     0.00000000
```

we have 4 symmetry operations, the first is the identity and the last is the reflection with respect to the "z" axis. Remember that the code only uses the metric, and therefore the z-axis, which in this case is "obvious" because angles 13 and 23 are 90 degrees. In the general case it may be more complicated.

Next the output informs that there are 13117 G-vectors, but only 1868 are unrelated by spatial or temporal symmetry, and some information about the maximum values of g_i encountered in the expansion $\vec{G} = \sum_{i=1,3} g_i \vec{b}_i$ for the potential and charge densities.

```
    13117 G-vectors are set up in    1868 stars,  kmax =   17  17  10
```

It also informs that of the $4 \times 4 \times 4$ integration grid in reciprocal space, there are 12 unrelated by symmetry, and that it will calculate 24 orbitals

```
    12 k-points generated by program from parameters :
    n =    4    4    4      s =  0.50  0.50  0.50      nb =   24


  Computing time for starting (s):           0.01
```

5.4 The SCF cycle

The first information about the SCF cycle is the FFT grid size and the maximum and minimum values of the local self-consistent potential.

```
      in fft for local potential n =   24   24   16

  max and min of potential (Hartree)    0.1915   -4.8485    0.0000
```

In the first iterations one may encounter warnings of the type

```
   WARNING       WARNING:   After h_kb_dia
   The estimated error in energy has an accuracy
  of      7.3 digits
```

First, 7 digit accuracy is already quite good. To be fast the code does not try to be very accurate in the early SCF iterations, saving computing time. As long as the warnings do not persist into the final iteration, these warnings can be ignored, they are here because they can be helpful if things go wrong...

The information for iteration 8 is

```
  in fft for local potential n =   24   24   16

  max and min of potential (Hartree)    0.1819   -4.7630    0.0000

  the fermi level is at     5.9538 [eV]

 total energy =   -31.8310932988
  computing time for iteration     8          1.890
```

again grep may be used to find out what is happening along the SCF iterations.

5.5 Potential, energies and forces

Once self-consistency is achieved, the information about the reciprocal space expansion of the density and potential are printed,

```
   iteration number  9


    i   k-prot          Ek       den     V(out)    V(in)     delta V   Vionic

    1   0  0  0     0.00000  32.00000  -0.33659
    2  -1  0  0     0.14055   0.01194   0.00052   0.00051   0.00001  -0.01183
                              0.00024   0.00001   0.00001  -0.00000  -0.00000
    3  -1  1  0     0.18740   0.01175   0.00039   0.00039   0.00000  -0.01138
                              0.02455   0.00082   0.00081   0.00001  -0.02276
    4  -1 -1  0     0.37480   0.00032  -0.00005  -0.00005  -0.00000   0.00971
                              0.00000   0.00000   0.00000  -0.00000  -0.00000
    5   0  0  1     0.37480   0.00298   0.00018   0.00018   0.00000  -0.02912
                              0.00000   0.00000  -0.00000   0.00000   0.00000
    6  -2  1  0     0.51535   0.01475   0.00007   0.00007   0.00000   0.00858
                              0.00565   0.00006   0.00005   0.00000   0.00000
    7   0 -1 -1     0.51535  -0.01402  -0.00007  -0.00007  -0.00000  -0.00858
                             -0.00344  -0.00003  -0.00003  -0.00000   0.00000
    8  -2  0  0     0.56220   4.98977   0.03271   0.03271  -0.00000  -0.16014
                             -5.03324  -0.03301  -0.03301  -0.00000   0.15190
    9  -1  1  1     0.56220   4.98124   0.03270   0.03270  -0.00000  -0.16014
                              5.02352   0.03299   0.03299   0.00000  -0.15190
```

after the index, one has the expansion of the G-vector, the kinetic energy associated (half the length squared) the Fourier coefficient of the density, the Fourier coefficients of the input and output XC potential, followed by the difference `delta V` and the coefficients of the ionic pseudopotential. The self-consistency is done by potential mixing (many codes do density mixing) and the self-consistency condition is on `delta V`.

Next we have the details of the total energy

```
Iteration number  9    Energies (Ha)          Changes
---------------------

Alpha Term        =         0.917257
Kinetic  Energy   =        11.949083        0.000022
Local PP Energy   =        -8.515343       -0.000104
Nonlocal Energy   =         5.722961        0.000071
------------------------------------
Harris-Foulkes    =       -31.831093       -0.000000
------------------------------------
Eigenvalue Sum    =         1.020901       -0.000013
HXC  Correction   =         8.135799        0.000001
Hartree  Energy   =         2.166313        0.000017
XC       Energy   =        -9.562418       -0.000005
Ewald    Energy   =       -33.591688
------------------------------------
Total    Energy   =       -31.831093        0.000000
```

The last column has the change with respect to the previous iteration, it is not zero in the contributions, but has six decimals in the variational total energy, meaning nice convergence. it should be noticed that the Harris-Foulkes energy is identical (within the six decimals) with the total energy, another sign of good convergence.

Th total energy is the sum of the five last contributions, the Harris-Foulkes energy is the sum of the four first contributions plus the Ewald energy.

The stress tensor is given in both lattice and cartesian coordinates

```
Contravariant stress tensor  (a.u.)             Cartesian stress (GPa)
 0.000642    0.000195    0.000000        0.239973E+01    0.929093E-12    0.267779E-13
 0.000195    0.000642    0.000000        0.914063E-12    0.256751E+01   -0.116381E-13
 0.000000    0.000000    0.000836        0.254685E-13   -0.104815E-13    0.119894E+01

  0.00006986          2.05539463      pressure (au and GPa)               Total
```

its trace (divided by three) is the pressure.

Final information is the force on the atoms,

```
      Force (Lattice coord.)            Force (Cartesian coord. a.u)       no. type

  0.00053   -0.00062    0.00000    -0.69115E-03 -0.11838E-01  0.74502E-17   1   Si    forc
 -0.00001    0.00001   -0.00000    -0.80366E-16  0.15823E-03 -0.33131E-15   2   Si    forc
  0.00062   -0.00053   -0.00000     0.69115E-03 -0.11838E-01 -0.30332E-15   3   Si    forc
 -0.00058    0.00058    0.00000    -0.57519E-15  0.11843E-01  0.15964E-15   4   Si    forc
 -0.00053    0.00053    0.00000    -0.14862E-15  0.10863E-01  0.88501E-16   5   Si    forc
 -0.00063    0.00063   -0.00000    -0.51521E-15  0.12831E-01 -0.18928E-15   6   Ge    forc
 -0.00032   -0.00091   -0.00000    -0.89184E-02 -0.60653E-02 -0.27596E-15   7   Ge    forc
  0.00091    0.00032   -0.00000     0.89184E-02 -0.60653E-02 -0.51362E-16   8   Ge    forc
```

indicating that we are close but not at equilibrium.

Computing time is given at the very end

```
  Total computing time (s):        47.74    Elapsed time (s):        12.01
```

with the ratio between computing and elapsed time indicating that the system had four cores.

6 Other files from cpw.exe

By design the code tries to write the minimum number of files. With grep one should be able to produce relevant information.

For example

```
cpw2000-5.x.y/WORK$ grep "4 Si position" ../Validation/Output/pw_ref_04.out
```

will produce the trajectory of the 4th atom in the simulation cell

```
    0.37456    0.12804    0.00365     0.36474E+01 -0.25300E+01  0.26210E-01   4   Si    posi
    0.37435    0.13059    0.01401     0.36644E+01 -0.25017E+01  0.10068E+00   4   Si    posi
    0.37460    0.13249    0.02300     0.36800E+01 -0.24848E+01  0.16520E+00   4   Si    posi
    0.37504    0.13449    0.02959     0.36977E+01 -0.24688E+01  0.21256E+00   4   Si    posi
    0.37525    0.13653    0.03398     0.37141E+01 -0.24500E+01  0.24414E+00   4   Si    posi
    0.37561    0.13857    0.03628     0.37314E+01 -0.24328E+01  0.26066E+00   4   Si    posi
    ....
    0.37419    0.12132   -0.02379     0.35960E+01 -0.25952E+01 -0.17094E+00   4   Si    posi
    0.37595    0.11977   -0.02458     0.35975E+01 -0.26292E+01 -0.17655E+00   4   Si    posi
    0.37730    0.11958   -0.02441     0.36060E+01 -0.26451E+01 -0.17536E+00   4   Si    posi
    0.37842    0.11900   -0.02299     0.36098E+01 -0.26624E+01 -0.16519E+00   4   Si    posi
    0.37909    0.11856   -0.02009     0.36116E+01 -0.26738E+01 -0.14429E+00   4   Si    posi
    0.37946    0.11954   -0.01604     0.36213E+01 -0.26676E+01 -0.11524E+00   4   Si    posi
```

allowing the user to plot it.

If the atoms move a `cpw.out` file will be written. With eventual modifications it can be used to continue the calculations with the final geometry. Notice that the default choices are different.

If the atoms move a `RESTART.DAT` file will be present. It may be used to continue the simulation for more steps, or restart if the calculation was interrupted by a power failure.

The most important file will be `PW_RHO_V.DAT`. It is a binary file, so it cannot be modified by accident but that may not allow portability between different operating systems. It contains the information about the crystal structure, charge density and potential.

7 Post processing

Once the self-consistent potential for some structure has been saved to the `PW_RHO_V.DAT` file, one can proceed to calculate many properties witn a post-processing code,

```
cpw2000-5.x.y/WORK$ ../cpw_post_process.exe
```

it is mainly an interactivr code, asking questions and proceeding according to the answers. All the answers are printed back to a file `replay_post.dat`. This way the user will have a record of the answers, and more important, may rerun the post processing, with eventually slight changes.

```
cpw2000-5.x.y/WORK$ cp replay_post.dat replay.dat
```

and then, with optional minor editing of the file,

```
cpw2000-5.x.y/WORK$ ../cpw_post_process.exe < replay.dat
```

The post processing code is in fact several independent subroutines (sub-programs) called from the same top level program.

That program loops at most 100 times (just edit if you want more!) asking what the user wants and reads an integer entered by the user.

If the integer is zero, the program stops. At lower levels of the program a choice of zero would return the execution to the calling level.

If the integer is not in the allowed range, the question is repeated. In lower levels of the program an out of range choice gives the user a second chance. If the user gives again a wrong choice the code returns to the calling level.

Looking at the examples

7.1 Band structures, density of states, optical response, quantum geometric,...

This is the part of the post-processing that has the most options

7.2 Modified tight binding

For a fast dense sampling of the Brillouin zone, a modified tight-binding scheme can be used. It has many similarities with Wannier interpolation, and in fact can be used to write the files required to call WANNIER90.

7.3 Plot of charge densities and potentials

Plots or prepares the plots of real functions electron density, $\rho(\vec{r})$, effective potential, $v_{\text{eff}}(\vec{r})$ and some potential contributions.

It can generate a three dimensional plot in the `".xsf` format of XCrysDen compatible with VESTA, or a two dimensional level plot in a plane chosen by the user.

An interesting capability is to plot along a line these quantities averaged over the perpendicular plane. It is very useful to find band alignments.

7.4 Plot densities of states

The data for the calculation of the density of states and respective plot can be generated with different approaches. This section of post processing reads the file with the data and generates the plot.

Choices of graphics options are separated from the sometimes very heavy calculations of the bands throughout the Brillouin zone.

7.5 Exploration of a k.p model

The first option may store a k.p model in a file. This section allows the exploration of that model.

7.6 Plot dielectric function

Like the case of density of states, the data for the dielectric function is first calculated throughout the Brillouin zone, the choices of graphics options are chosen in this section of the code.

7.7 Plot of the wave-functions

It is similar to the plots of the charge density, but now the function to be plotted is complex, adding an extra choice of real part, imaginary part, or module, plus obviously the choice of which state to plot. Both 2D and 3D options are available.

7.8 Crystalography

This section allows a reveryough examination of the crystal structure.

7.9 Interface to Quantum Espresso

Quantum Espresso has many more developers than the present code. This section writes a QE input file. The pseudopotential code can write the required pseudopotentials, including Troullier-Martins pseudopotentials with spin-orbit (a feature that is not available in the QE toolchain). The consistency between the two codes are at the 4th or 5th decimal place in the total energy.

8 Where to find crystal structures

Many example of `cpw.in` files can be found in the `Structures` directory. Besides (almost) all the elements, there are many files collected over the years.

There is an auxiliary program in the toolbox, `Tools/gen_PW.f90` that can help the user to create a `cpw.in` file by answering some questions.

As mentioned before, the input file was chosen to be very similar to the SIESTA input file, so the user adapt it easily.

Another option is to convert any crystal file in the `".xyz` format which is sufficiently simple to be adapted.

9 Toolbox

In `Tools` directory the user can find a few useful tools.

Most are sufficiently short that the user can easily read and modify the code.

It also has programs that call one of the options of post-processing. They were used to develop those options.

9.1 Equation of state

From a few values of energy as a function of volume (or lattice constant for cubic systems), a least square fit is applied to the Murnaghan or Birch-Murnaghan equations of state. The output is the equilibrium volume, bulk modulus and pressure derivative of the Bulk modulus. If the user has several such data sets, the code will find the transition pressures between those structures.

9.2 Remove gnuplot commands from 2D plot files

The 2D plot files are in the gnuplot format, with plotting directives. To remove the directives one can run `Tools/convert_2Dplot.f90`

9.3 Crystalography from cpw.in

Doing a crystalographic analysis after a self-consistent calculation may be too late. With `Tools/cpwin_geom.f90` the analysis may be done from the `cpw.in` file. Read the crystallography sub-section of the post-processing section to see how it works.

9.4 Coulomb potential

The Coulomb potential for hydrogen can be generate by `Tools/h_pot_kb.f90`

9.5 Graphical interface for analysis of band structure

This is a very useful tool. From a file with the relevant data it generates a very powerful GUI. There is an earlier version `BandInfoUi` based on the `grace` plotting package, which is similar to the most recent package and therefore there is no need to describe in detail.

The most recent package is the `QtBandViewer`. It is a `python` code that uses the `Qt` user interface framework (`https://www.qt.io`).

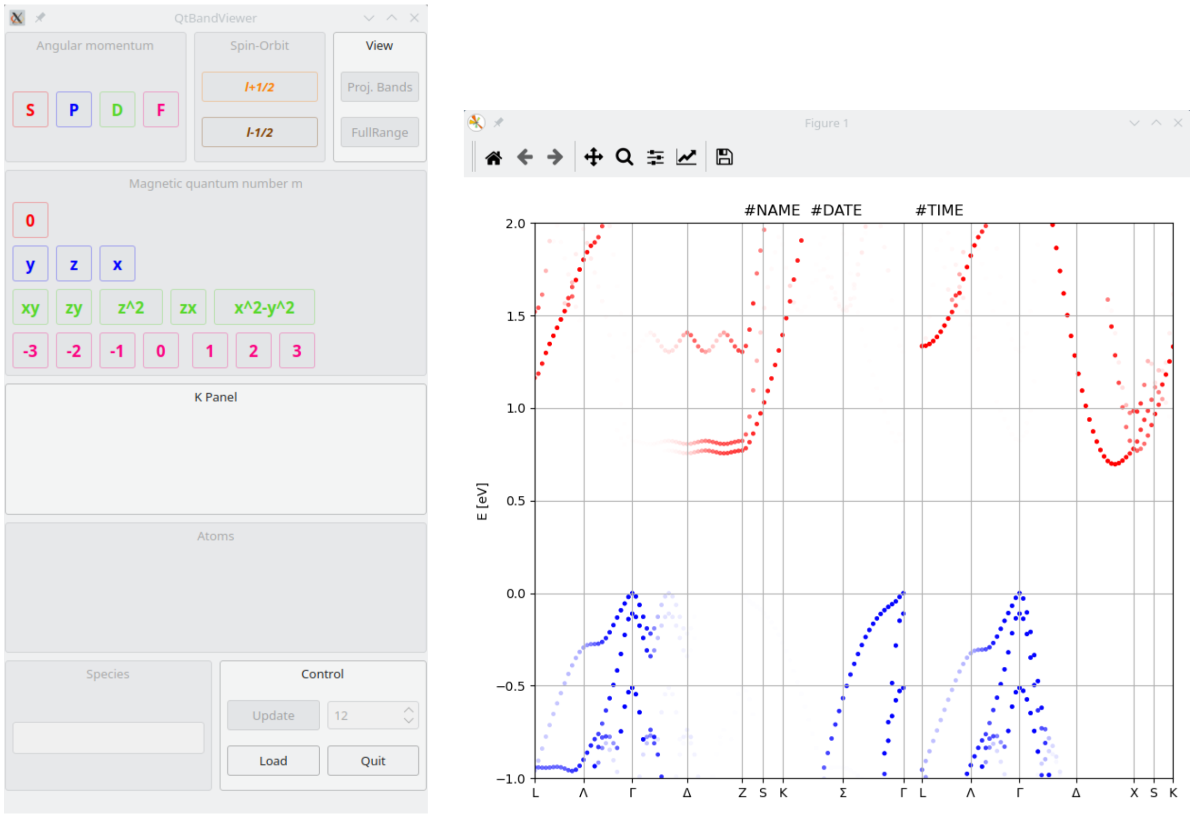
To install follow the instructions on `Tools/QtBandViewer/AAAREADME`. It is strongly recommended to create a python virtual environment under a working directory, and copy files there. By some obscure reasons (the writer of the guide does not use python) the python files should be copied. To use the tool you will need access to a `.bv` file format, by default named `BAND_SO.DAT.bv` or `BAND.DAT.bv` with the band energies and the `BAND_LINES.DAT` file used to generate those files. Those data files can be huge so avoid copying them.

As an example we will use the files generated from the structure file `Structures/Supercells/cpw_Si6Ge6_Si(001).in` with unfolding. The `BAND_LINES.DAT` was copied from `Structures/Supercells/band_lines_Si6Ge6_Si(001).dat`

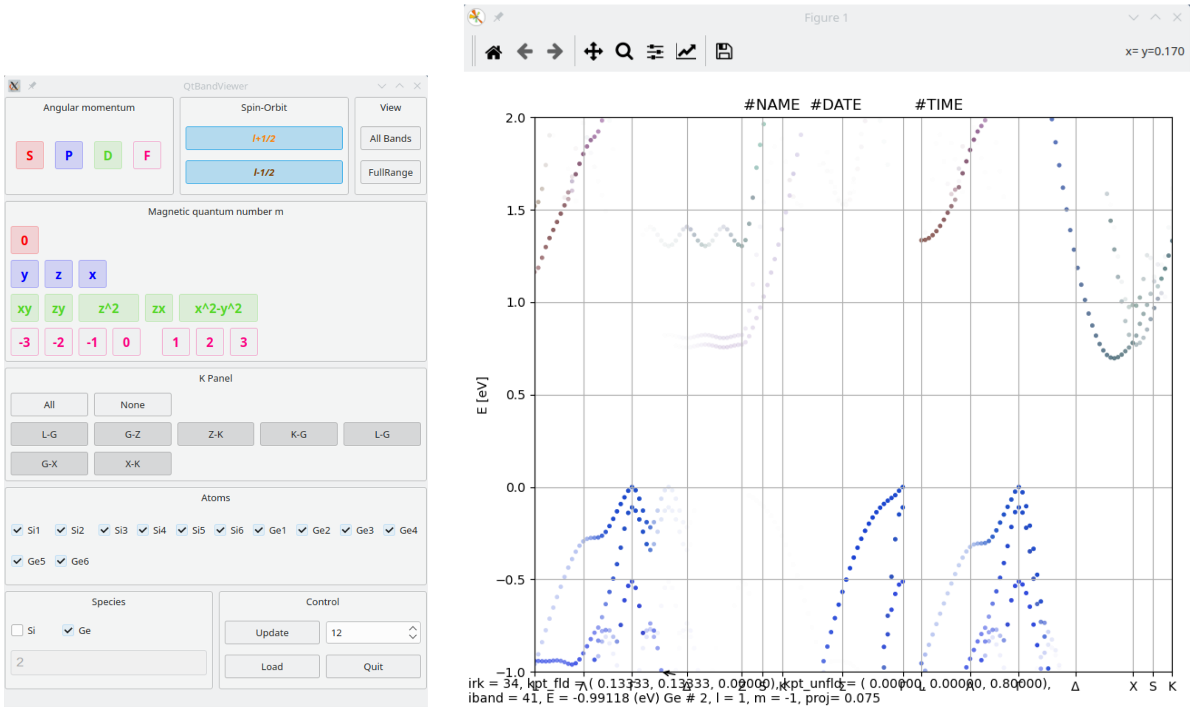
once the code is installed, run it

```
cpw2000-5.x.y/WORK/QtBV$ python3 QtBandViewer.py
```

The initial window is shown in the left of the figure. Press the `Load` button, and a new window shown on the right will display the unfolded band structure. Notice how the bottom of the conduction band appear to be in the Γ–X direction (parallel to the surface) and there are flat bands in the epitaxial growth direction Γ–Z.



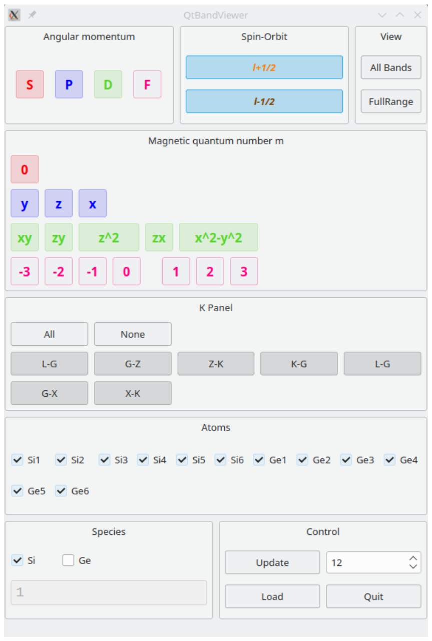
If now we deselect Si leaving only the Ge contributions, and select the s, p and d angular momenta, we see that "dots" at the bottom of the band are still dark, but that the flat band almost disappears. This indicates that the bottom of the conduction band has a contribution from the Ge layers and there is a localized band in the Si layer.



```
irk = 34, kpt_fld = ( 0.13333, 0.13333, 0.00000), kpt_unfld= ( 0.00000, 0.00000, 0.80000),
iband = 41, E = -0.99118 (eV) Ge # 2, l = 1, m = -1, proj= 0.075
```

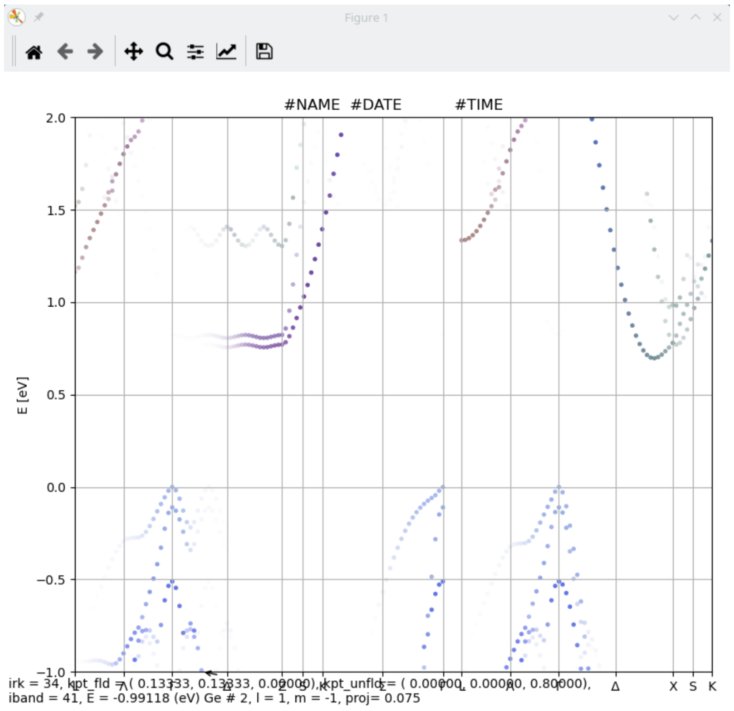
Reselecting Si and deselecting Ge and keeping the s, p and d angular momenta, we confirm that the flat band is indeed mostly Si, the bottom of the conduction band has a Si contribution that is weaker than the Ge contribution. It is also apparent that for the valence band at the unfolded

Γ there is a band below the top and split-off bands that has a stronger Si content. For larger supercell periodicities it would converge to the band alignment between Si and strained Ge.



Si [✓] Ge [ ]

12



```
irk = 34, kpt_fld = ( 0.13333, 0.13333, 0.00000), kpt_unfld = ( 0.00000, 0.00000, 0.80000),
iband = 41, E = -0.99118 (eV) Ge # 2, l = 1, m = -1, proj= 0.075
```