Excercise sheet 1

How to run the calculations

Unless indicated otherwise, the calculations in the exercises will be performed with Quantum Espresso, a mature package of DFT codes that can be used to compute a variety of properties and uses a planewave basis in combination with pseudopotentials. You can either run the exercises on the HPC of the RRZE, using the accounts created a few weeks ago, or on a private computer. The following will lay out how to run the calculations in each of these cases

On the HPC cluster

Unfortunately, the RRZE does not offer a precompiled Quantum Espresso module. I have thus prepared a .zip file with the QE programs, which you can download on StudOn.

First log in to Emmy (see the file “Connecting to the HPC.pdf” on StudOn) and open a command window so that you can run command directly on the HPC.

Now create a folder “bin”, either directly in WinSCP (or a comparable program) or by typing the command

```
mkdir bin
```

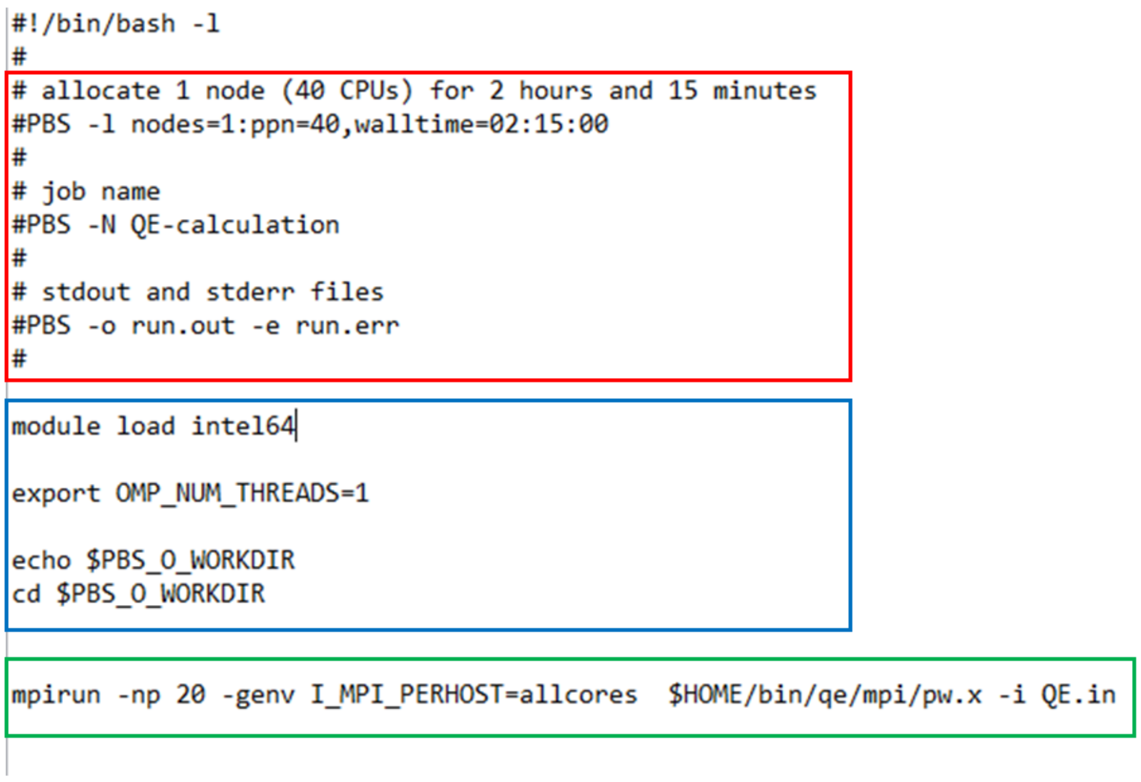
into the command window. In case you have not worked with Linux before, the following page lists a few commands that are useful or essential for working on the HPC: https://www.hostinger.com/tutorials/linux-commands . Now copy the zip file in the newly created “bin” folder. Use the command window to enter into the “bin” folder by typing “cd bin” and uncompress the .zip-file by

“tar -xvzf qe-hpc.zip”. This will unpack the contents of the tar file, after you can delete the file. You should now have a folder “qe” in you “bin” folder. QE has two subfolders, “mpi” and “serial”. “mpi” contains the programs that benefit from parallelization, i.e. from running them on several CPUs simultaneously. “serial” contains programs that are used for post-processing of the results of DFT calculations and are run on a single CPU.

Apart from the quantum espresso package that you just downloaded, there is a number of additional packages and libraries available as modules on the HPC. The command “module avail” will display all available modules on the HPC, such as python or MATLAB, as well as several compilers that can be used to build new software. You can load or unload module X with the commands “module load X” and “module unload X”, respectively. As the downloaded Quantum Espresso programs depend on a number of libraries from such modules, loading modules will be necessary, but can be done in an automatic way. You can display all modules that are currently loaded by “module list”.

So, with the preparation done, how exactly are the calculations run? Like most HPC clusters, the RRZE clusters apply a scheduling software that manages the computational resources. In a nutshell, this means that you prepare a “job” that you want to be executed together with a “submission script” that tells the submission software, what you want to be done and what resources you want.

The following picture shows such a submission script:



```
#!/bin/bash -l
#
# allocate 1 node (40 CPUs) for 2 hours and 15 minutes
#PBS -l nodes=1:ppn=40,walltime=02:15:00
#
# job name
#PBS -N QE-calculation
#
# stdout and stderr files
#PBS -o run.out -e run.err
#

module load intel64

export OMP_NUM_THREADS=1

echo $PBS_O_WORKDIR
cd $PBS_O_WORKDIR

mpirun -np 20 -genv I_MPI_PERHOST=allcores  $HOME/bin/qe/mpi/pw.x -i QE.in
```

Essentially, it consists of three parts:

1. The first part, in a red frame, tells the batch scheduler what computational resources you want to use and for how long. In the above example, we apply for one node with 40 processing units for 2 hours and 15 minutes.

   The indicated walltime is the estimated maximum time that the calculations will run. If the calculation is finished earlier, the slot will be freed again. If the calculation runs longer than the indicated walltime, the scheduler will kill it. We can apply for up to 24 hours.

   As for the number of processing units: You always have to apply for integer numbers of nodes, otherwise the scheduler software will complain. Unless indicated otherwise, one node will be enough for the calculations in this lecture series.
   The number of processing units per node (40 in this case) is given by the cluster you are using and should not be changed, otherwise the scheduler will complain and not run the calculation.

   On Emmy, each node has 40 processing units. You can also use the (smaller) woody cluster. You can log in to that one in a similar way as you logged into Emmy (see PDF on StudOn), just using woody.rrze.fau.de instead of emmy.rrze.fau.de. In this case, each node has 4 processing units.

   The other lines assign a name for the calculation to be run and give names for a file where the scheduler will write all errors it encounters (here “run.err”) and where it writes the output from the programs that are run (here “run.out”). These can be changed freely. “run.out” is particularly important for Quantum Espresso calculations, because it will contain the output of the QE runs.

2. The second block, framed in blue, is more administrative and is used to prepare the actual calculation. Here, the intel64 module is loaded that Quantum Espresso depends on and we tell the cluster to enter the folder where we started the submission script from ($PBS_O_WORKDIR). In later exercises, we will use this block to move into a “scratch” folder and run the calculation there.

3. The third block, framed in green, runs the actual calculation. Here, we run the program “pw.x”, which is located in folder “$HOME/bin/qe/mpi”. “mpirun” is used to run the calculation in parallel, i.e. Quantum Espresso uses 20 CPUs for one single calculation. “QE.in” is an input file that details what pw.x is supposed to calculate and will be explained in a following section.

In the above example, you would prepare an input file QE.in for the quantum espresso calculation and a submission script, let’s call it “j.submit.sh”, in a given folder, let’s call it “calculations”. You would then enter “calculations” and submit “j.submit.sh” to the scheduler software by using the command

“qsub j.submit.sh”

The scheduler software will then try to reserve you the resources you requested and run the calculation as soon as the resources are free. Note that this means that your calculation might be put into a queue, such that it does not run instantly. You can check the status of all your calculations with the command “qstat -f”. This will display all your calculations that are running or waiting in the queue, together with an ID. This ID can be used to cancel running calculations or remove them from the queue. The command for this is “qdel ID”.

Due to the fact that the scheduler will reserve you a slot that fits the resources you requested, it makes sense not requesting significantly more resources than your calculations need. If you request 24 hours, the scheduler will assume that your calculations runs for 24 hours and will plan accordingly, even if your calculation only runs for 15 minutes. Strategy wise, it is thus better to roughly estimate the run time of the calculation (this comes with experience), because the scheduler is more likely to find a 15-30 minutes gap, where it can for your calculations in that to find a 24 hours slot. The same can be said about the number of nodes that are requested.

A final word about file storage space: After login, each user arrives in his/her home directory on the HPC ($HOME). Each user has a data quota of 10 Gb in the home directory, after which we will get warnings per e-mails, asking to delete data. The command ”quota” can be used to see, how much space one occupies currently. As quantum Espresso produces lots of data, it is advisable to only store relevant data in the home folder and either delete or move the other data. Additionally, each user also has access to a second home folder, which is located on the woody cluster and can be reached by typing “$WOODYHOME”. To make life easier, I suggest creating a link from $HOME to $WOODYHOME by typing “ln -s $WOODYHOME woodyhome”.

Linux

If you want to run the exercises from a home computer with a linux system, you probably can install Quantum Espresso from the package manager of your linux distribution, especially if it is fa Debian-based distribution. You can also download pre-compiled binaries from Advanced SoftwareCorp (https://github.com/advancesoftcorp/espresso/releases), but I do not have much experience with them. Alternatively, you can also download Quantum Espresso from their Github page (https://github.com/QEF/q-e/releases) and compile it yourself. Let me know if you need help with that.

After you have acquired Quantum Espresso, prepare an input file, let's call it QE.in here, in a folder, enter the folder and run the calculation with the command

```
$PATH_TO_PW/pw.x <QE.in |tee QE.out
```

or, if you have an mpi-enabled version that can run on several CPUs simultaneously:

```
mpirun -np N $PATH_TO_PW/pw.x <QE.in |tee QE.out
```

In the latter, N is the number of CPUs you want to run Quantum Espresso with.

The "tee" part is optional and will both show the output from the quantum espresso calculation on screen and write it into a file "QE.out". As the name indicates, $PATH_TO_PW is the location of pw.x and the other quantum espresso programs. If you installed it through a package manager, this should be the path "/usr/local/bin", but it depends on your distribution.

The structure of the QE input files will be described in a section below. I will also prepare template input files for each exercise, at least at first.

Windows

On Windows, you have two possibilities for obtaining and running Quantum Espresso:

1. You can download binary packages generated for Windows (https://www.quantum-espresso.org/download, links on the bottom, I would use the AdvanceSoftwareCorp link). These you can install to a location of your choice, let's call the location PATH_TO_PW. There, qe.zip contains the quantum espresso programs, while mpi.zip contains the files necessary to run Quantum Espresso on more than 1 CPU simultaneously. I **strongly** recommend to extract both qe.zip and mpi.zip (if downloaded) to the same directory and then adding this directory to the Windows path (directions can be found here: https://www.computerhope.com/issues/ch000549.htm). You can then run the quantum Espresso calculations without explicitly giving PATH_TO_PW.

   Running Quantum Espresso then requires opening the Windows PowerShell, which can be found by entering "Powershell" in the windows search bar. Use then DOS commands to switch to the hard disk and directory of your choice. A number of essential DOS command can be found here: https://www.computerhope.com/overview.htm

   As for linux, you then need to prepare a QE input file. After this is prepared, you can then run the calculation using the commands

   ```
   pw -i QE.in |tee QE.out
   ```

   or

   ```
   mpiexec -np N pw -i QE.in |tee QE.out
   ```

This is under the assumption that you set the Windows path accordingly, such that the Powershell knows where to find mpiexec.exe and pw.exe. Otherwise you will have to give the full path, for example “PATH_TO_PW\pw”.

2. You can use QuantumMobile, which is a prepared Linux environment that can be run from a VirtualBox under Windows. A warning: While this method is relatively simple, the fact that QuantumMobile runs in a virtual machine eats up more resources than the other method. Calculations might thus be slower than with the other method.

   First download and install VirtualBox (https://www.virtualbox.org/). Also download QuantumMobile from https://www.materialscloud.org/work/quantum-mobile.

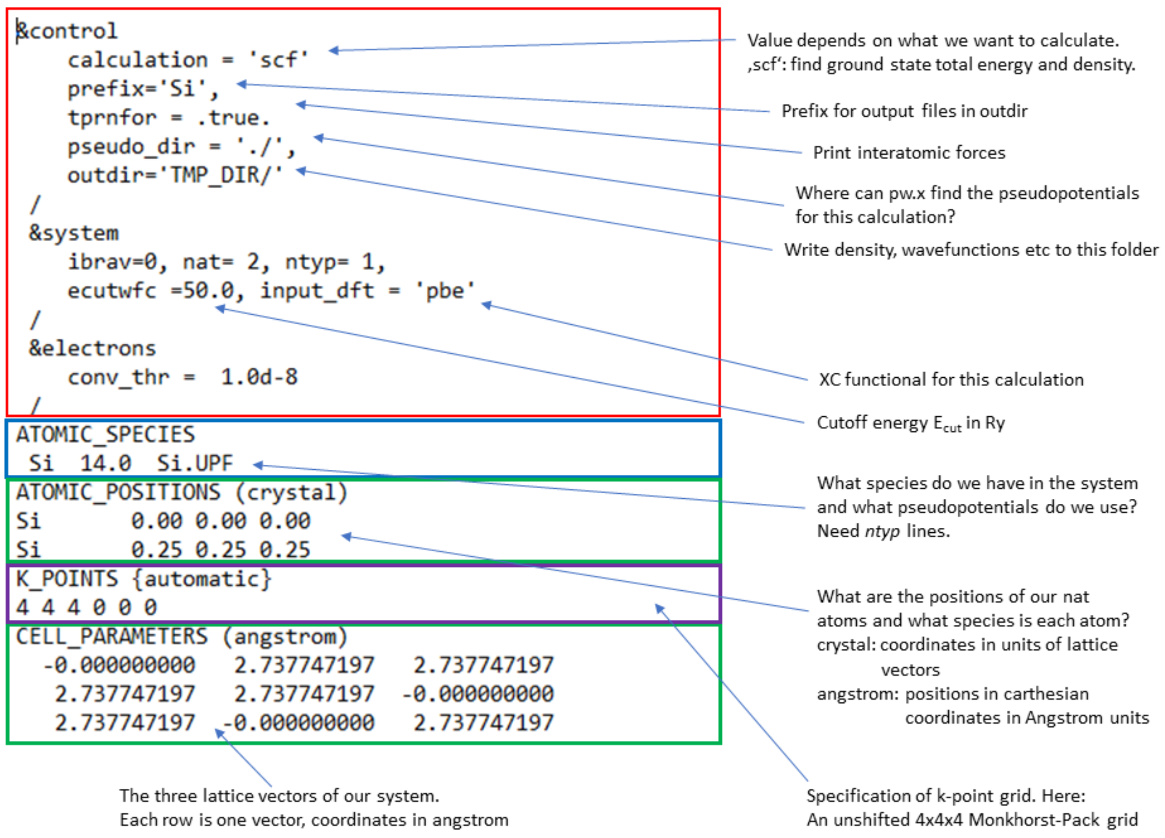
   Now go to the folder where you downloaded that QuantumMobile.ova file to and double-click on the downloaded file. This should open VirtualBox and a window for the import of an applicance (in this case Quantum Mobile). You can change the folder where Quantum Mobile will be “installed” to in the bottom. Clicking “Import” and agreeing to the T&A will install Quantum Mobile. After the import, you will have an entry related to Quantum mobile in the left panel of the VirtualBox window. Double-clicking it will start Quantum Mobile.

   However, there is still no simple way to exchange data between the Virtual Machine and your “host” windows system. You can add a shared folder, a folder on your windows system that the virtual machine can access. To do this, mark the QuantumMobile icon in the left panel of the VirtualBox window and click on the yellow “change” button in the top panel. This opens a window with some settings. Go to “shared folders”. There you can add a shared folder that is automatically linked to the virtual machine whenever you start QuantumMobile. This folder is shown as a symbol “sf_shared” on the desktop of the QuantumMobile environment.

   The black button in the left panel of the quantum mobile window opens a terminal window that can be used to run the calculations in the same way as outlined in the “Linux” part above. The Quantum Espresso files (and those of other installed programs) are located in the folder “/usr/local/bin/”. The top button in the left panel opens a “file manager” that can be used to look at the files and folders stored in the virtual machine. In case you are new to linux, the following page lists a few basic commands that will be useful or even necessary for using the terminal window: https://www.hostinger.com/tutorials/linux-commands

Structure of QE input files

The following picture shows a typical input file that is used to do calculations with Quantum Espresso, more specifically the PWscf module, which is the most important component of Quantum Espresso. It



```
&control
    calculation = 'scf'
    prefix='Si',
    tprnfor = .true.
    pseudo_dir = './',
    outdir='TMP_DIR/'
 /
 &system
    ibrav=0, nat= 2, ntyp= 1,
    ecutwfc =50.0, input_dft = 'pbe'
 /
 &electrons
    conv_thr =  1.0d-8
 /
ATOMIC_SPECIES
 Si  14.0  Si.UPF
ATOMIC_POSITIONS (crystal)
Si       0.00 0.00 0.00
Si       0.25 0.25 0.25
K_POINTS {automatic}
4 4 4 0 0 0
CELL_PARAMETERS (angstrom)
  -0.000000000   2.737747197   2.737747197
   2.737747197   2.737747197  -0.000000000
   2.737747197  -0.000000000   2.737747197
```

Value depends on what we want to calculate.
‚scf': find ground state total energy and density.

Prefix for output files in outdir

Print interatomic forces

Where can pw.x find the pseudopotentials for this calculation?

Write density, wavefunctions etc to this folder

XC functional for this calculation

Cutoff energy E_{cut} in Ry

What species do we have in the system and what pseudopotentials do we use? Need *ntyp* lines.

What are the positions of our nat atoms and what species is each atom?
crystal: coordinates in units of lattice vectors
angstrom: positions in carthesian coordinates in Angstrom units

The three lattice vectors of our system.
Each row is one vector, coordinates in angstrom

Specification of k-point grid. Here:
An unshifted 4x4x4 Monkhorst-Pack grid

is used to obtain the ground state energy and electron density, optimize the atomic structure and lattice vectors, calculate electronic bandstructures etc.

The input file consists of a number of components:

1. The red frame highlights the essentially blocks that are necessary to perform a PWscf calculation.
    - The control block contains keywords that control the type of calculation to be performed and administrative information about where Quantum Espresso can find things and where to write things. I.e. more general information that is not specific to the system to be simulated

    - The system block contains more specific information about the computational parameters to be used in the simulation. These include the cutoff energies for the planewave basis and the exchange-correlation functional, but also control over the possible smearing of band occupations, spin polarization, magnetism, the number of band included in the calculation etc. It also contains information about the number of different species in the calculation (`ntyp`) and the number of atoms (`nat`) and, if known, the Bravais lattice of the system. The keyword `ibrav` together with `celldm` can be used to specify the Bravais lattice of the crystal (https://www.quantum-espresso.org/Doc/INPUT_PW.html#idm217 for more information). I, personally, could never be bothered to actually use `ibrav`, so I set `ibrav=0` (i.e. no information about the Bravais lattice) and instead give the lattice vectors explicitly in the `CELL_PARAMETERS` block.

- The electron block allows additional control about the convergence of the electronic system. The most important keyword is `conv_thr`, which gives the convergence threshold for the ground state total energy in units of Ry for SCF calculations, i.e. the SCF procedure is converged if ΔE_{tot} <conv_thr. It has a different meaning for bandstructure calculations, as we will see. One can also choose the mixing weight for the density mixing or the iterative solver, among other things.

2. The `ATOM_SPECIES` block gives information about the species that are contained in the system, where each line contains the species label, the atomic weight of the species and the name of the pseudopotential file that QE shall use for the atoms of this species. The pseudopotential files need to be in the folder specified by `pseudo_dir`.

3. The green frames highlight blocks that contain probably the most vital information: what is the position of each atom and what is the periodicity of the system? In most cases, it is convenient to give the atomic positions in as fractions of the lattice vectors, $x = n_1\vec{a}_1 + n_2\vec{a}_2 + n_3\vec{a}_3$, where the columns contain n_1, n_2, n_3. Alternatives are specifying the atomic positions in carthesian units, for example in Angstrom, which can be activated by replacing "`ATOMIC_POSITIONS (crystal)`" by "`ATOMIC_POSITIONS (angstrom)`". In a similar way, one can specify coordinates in bohr units.

4. Lastly, the `KPOINTS` block in the purple frame specifies the k-point sampling. For SCF calculations, one usually uses an "automatic" generation of a LxMxN Monkhorst-Pack grid, by giving six number: the first three numbers corresponds to L M N, while the last three numbers indicate the shift: 0 mean no shift in the respective direction in reciprocal space, while 1 indicates a shift by half the distance between grid points in that direction.

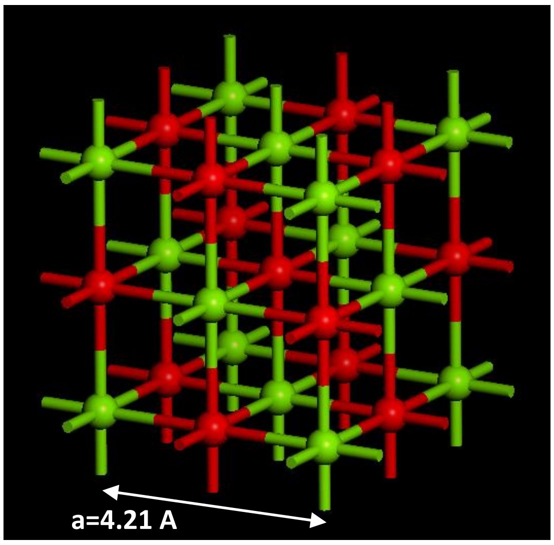
The full documentation of the pw.x input files, including all allowed keywords, can be found at https://www.quantum-espresso.org/Doc/INPUT_PW.html#idm217

Exercise 1: Convergence test on magnesia (MgO)

The following picture shows the conventional unit cell of bulk MgO. The lattice vectors and atomic positions are already included in the input files MgO_abc.in and MgO_d.in in the folder Exercise_1 in the file "Exercise_Sheet_1.zip". The purpose of this exercise is deriving values for the cutoff energy E_{cut} of the electronic wavefunctions, as well as for the discrete k-point sampling of the Brillouin zone, as explained in lecture 6. This will allow for meaningful results from subsequent calculations, for example of the total energy.

(a) We will perform a convergence test of the cutoff energy of MgO for the normconserving pseudopotentials, Mg.nc.UPF and O.nc.UPF, that are included in the .zip file. We will use the change of the total energy with the cutoff energy as a convergence criterion.

First, replace the dummy names in the `ATOMIC_SPECIES` block of the input file MgO_abc.in with the file names of the pseudopotentials. Make sure that the pseudopotentials correctly match the species label in the first column (swapping pseudopotentials between two species should not be a problem here, but can lead to ugly results for other materials. It is also a good practice, to give the proper atomic weights for each species in the second column, even though these values are ignored in most calculation types.



We also have to choose a k-point grid in order to be able to do the convergence test of the cutoff energy. As the effects of cutoff energy and k-point sampling are almost completely independent of each other, we could, in principle, perform the convergence test by only sampling the Gamma point. To be on the safe side, let's use a 3x3x3 unshifted sampling here. To do this, replace the first three values in the K_POINTS block by 3 3 3 and the last three by 0 0 0.

Now, we will systematically increase the cutoff energy by varying the value of `ecutwfc`. The energies are given in units of Ry. Start with a cutoff energy of 40.0 (Ry =544 eV), which is probably too low, but is a good starting point. Enter the value without a unit, else the program will complain with a cryptic error message.

Run the calculation by following the directions given above. A sample submission script j.submit_abc.sh for the HPC is included in the .zip-file. Due to the low cutoff and relatively coarse k-point sampling, the calculations should finish pretty quickly.

Examine the file, where Quantum Espresso wrote its output. After some initial header information, Quantum Espresso reports on the used input, such as computational parameters and geometry. `alat` is a kind of common factor that all atomic positions given in the output are scaled by. Notice that the "number of electrons" in the calculation was 64. This is 16 electrons less than an "all-electron" calculation without the pseudopotential approximation would consider. In fact, the used pseudopotentials merge the effect of the 1s core electrons of Mg and O with the electrostatic potentials of the respective nuclei. The line below "number of electrons" gives the number of Kohn-Sham orbitals per k-point in the calculation. Here, the number is exactly half that of the number of electrons, i.e. we model the systems as an insulator without spin polarization, where exactly $N_e/2$ bands are fully occupied per k-point. This makes sense for a material with a band gap. Obviously, the number of included bands could be chosen larger than $N_e/2$, by using the keyword nbnd in the `system` block, but it would not lead to any advantage for our purposes, as they would remain completely empty. Another line below, you will find the cutoff energy used for the calculation that corresponds to the value of `ecutwfc`. Following this block, Quantum Espresso reports the unit cells vectors of the real-space and the reciprocal space lattices, the atomic structure (in cartesian coordinates) and the species in the calculations. Notice that Quantum Espresso also performed an analysis of the symmetry of the system and found 48 symmetry operations, including inversion symmetry. This shows that MgO is a highly symmetric cubic crystal. Quantum Espresso used these symmetry operations to reduce the k-point grid from 3x3x3=27 points to an irreducible set of 4 k-points, the coordinates and weights of which are reported after the block with the atomic positions. Quantum Espresso then performs a cycle of SCF steps, until the estimated accuracy of the total energy is below the desired threshold `conv_thr` given in the input file. You are then presented with a list of results, in particular the single-particle energies at each k-point included in the calculation, the total energy and the forces acting on each atom. Extract the total energy for a cutoff energy of 40 Ry from the output.

Do similar calculations by increasing the cutoff energy in steps of 5 Ry up to a cutoff energy of 110 Ry. Extract the total energy for each cutoff energy and plot the results. You will see that the change of the total energy saturates for higher cutoff energies. What energy difference between successive cutoff energy steps we take as 'converged' depends on our needs and, of course, the length of the cutoff energy steps. It also depends on the number of atoms in the system, as the total energy is extensive. Let's use an energy difference of 0.001 eV per atom between subsequent cutoff energy steps as a criterion for when the cutoff energy is converged.

For which cutoff energy is this convergence criterion fulfilled for MgO?

(b) We will now do a convergence test for the k-point sampling. For this, we need to choose a cut off energy for the planewave basis. A definitely safe choice to ensure accuracy would be to use the value derived from (a), but this will make the calculations somewhat more expensive than necessary. A good alternative would be choosing a cutoff energy from (a) for which the total energy is not yet converged, but close enough to convergence to ensure a good representation of the electronic wavefunctions, for example somewhere in the range of 70-80 Ry. A choice of the cutoff energy that is too low might lead to problems with solving the Kohn-Sham Equations, for example a slower convergence of the SCF procedure.
With the cutoff energy set, we will now perform calculations with systematically increased densities of the Monkhorst-Pack grids used to sample the Brillouin zone of the system. In general, the aim is to choose k-point grids that lead to a sampling of the BZ that is as uniform as possible, in order to not "favor" one lattice direction over another in terms of accuracy. For the conventional cell of MgO2, all lattice vectors have the same length. It thus makes sense to use samplings with NxNxN k-points, where N is the number of k-points along a given reciprocal lattice vector. These correspond to the first three numbers in the `K_POINTS` block. We will also use unshifted grids for now. This means that the last three numbers in the block, the shifts, will be kept at 0.

Calculate and plot the total energies of MgO for step-wise increase of N from 2 to 12. Use the same convergence criterion as in (a). What k-point sampling should be used to obtain sufficiently converged total energies?

(c) Let's compare the results from (b) to the results from corresponding shifted grids. For this, we still vary N, but will set the three shifts (i.e. the last three numbers) in the `K_POINTS` block to 1. To make the calculations less repetitive, only consider even values of N. Compare the convergence of the total energy and the numbers of irreducible k-points between shifted and unshifted grids. Do both grid kinds leads to the same converged total energy? Would you choose shifted or unshifted k-point grids for efficient calculation of the total energy of MgO?

Side note: Notice the significant increase in efficiency in symmetry reduction of the full k-point grids with increasing N.

(d) So far, we only did calculations with normconserving pseudopotentials. We will now do a convergence test for ultrasoft pseudopotentials as well. In this case, we have an additional parameter to consider. Have another look at the output files from step (a). You will notice in the block reporting the input parameters at the beginning of the file that a second cutoff energy below the kinetic-cutoff energy is mentioned: an energy cutoff for the electron density. This arises from the fact that the Fourier transformation for the electron density is formally given by $n(G) = \sum_k \sum_{G'} c_k^*(G')c_k(G-G')$. This means that one has to include the differences $G - G'$ in the calculation of n(G) as well, as compared to the Fourier transformation of the wavefunctions. The charge density cutoff energy to be used in a calculation can be specified with the keyword `ecutrho` in the system block. If unspecified, it is set by default to a value of 4x `ecutwfc`, which ensures that the charge density has the same accuracy as the wavefunctions. For normconserving pseudopotentials, setting `ecutrho` to anything larger than 4x `ecutwfc` does not (or should) not change the total energy, especially not close to convergence. It can thus be left at the default value.

For ultrasoft pseudopotentials, the situations is different: here, `ecutrho` also controls the grid used for calculations of the augmentation component Q_{ij}, which is independent of the convergence of the wavefunctions. A rule-of-thumb for ultrasoft pseudopotentials is choosing `ecutrho` = 8x `ecutwfc`.

Let's converge. In the first step, store the results from (a), (b) and (c) in a separate folder and replace the filenames of the normconserving pseudopotentials in the input file by those of the ultrasoft pseudopotentials. Also remember to use a coarser grid for the k-point sampling, as in (a)

Now add the keyword `ecutrho` to the system block, ideally right behind the definition of `ecutwfc` (you can also use the input file MgO_d.in). Clearly, we cannot easily converge `ecutwfc` and `ecutrho` simultaneously (as for normconserving psps), as we would have to sample different combinations of `ecutwfc` and `ecutrho`. We will thus first converge `ecutwfc` while setting `ecutrho` = 8x `ecutwfc` and subsequently converging `ecutrho` after a converged value of `ecutwfc` was found.

Calculate and plot the evolution of the total energy by increasing `ecutwfc` from 30 to 100 Ry (again using 5 Ry steps). Use the same convergence criterion as in (a). For what cutoff energy is the total energy converged? Also compare the degree of total energy variation of the considered cutoff energy ranges for ultrasoft and normconserving pseudopotentials.

Now, take the converged value of `ecutwfc` and do a convergence test for `ecutrho` by changing `ecutrho` between 4x `ecutwfc` and 12x `ecutwfc`.

Was the initial rule-of-thumb choice of 8x `ecutwfc` a good idea?

(e) Extra task: Instead of the conventional cell, consider the primitive cell. Use the above picture to derive the lattice vectors and atomic positions of the primitive cell and adjust the input file accordingly. The converged energy cutoffs are the same as those for the conventional cell (you can test!), but the converged k-point sampling should be different, due to the larger size of the corresponding Brillouin zone. Obtain the k-point sampling that is sufficient to yield a converged total energy.

Exercise 2: Convergence test on 2D antimony

In this exercise, we will perform convergence tests on a material of lower dimensionality, antimonene (which is two-dimensional antimony) The input file and pseudopotentials can be found in the .zip file, the structure of antimonene is shown in the figure.

(a) In contrast to the MgO crystal in exercise 1, this material is only periodic in 2 directions. As Quantum Espresso and the planewave basis require periodicity in three directions, we will add large vacuum layer in the z-direction. In this way, the system is formally 3D periodic, while the vacuum layer reduces the residual interaction between our 2D material layer of interest and the periodic images of the layer in the neighboring unit cells in z-direction.

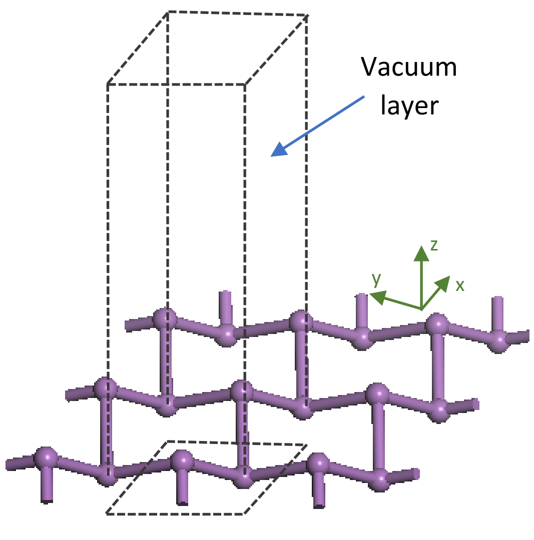
    Our goal is now to find a thickness for this vacuum layer that sufficiently reduces the interlayer interactions to yield converged energies. But that is still not larger than necessary in order to limit the computational effort.

    Systematically increase the length of the lattice vector that points in z- direction from 7.5 Ang to 25 Ang and calculate and plot the total energies. Use the cutoff energy and k-point sampling defined in the input file. You will also notice that the definition of the atomic positions in this input file.

    What length for the lattice vector in z-direction would you choose to calculate a converged total energy?

(b) Finally, we will converge the cutoff energy, but not using the convergence of total energy differences as criterion, but the convergence of the interatomic forces. This is useful for geometry optimizations, which rely on accurate predictions of the interatomic forces.

    Use the converged vacuum thickness from (a) and do a convergence test by increasing `ecutwfc` from 50 to 120 Ry. Due to symmetry constraints, only force components in z-direction and not zero. Plot the evolution of the non-zero component of the interatomic forces and find the cutoff energy for which the difference of interatomic forces between subsequent cutoff energy steps is smaller than 0.00001 Ry/au.


Vacuum layer
z
y
x