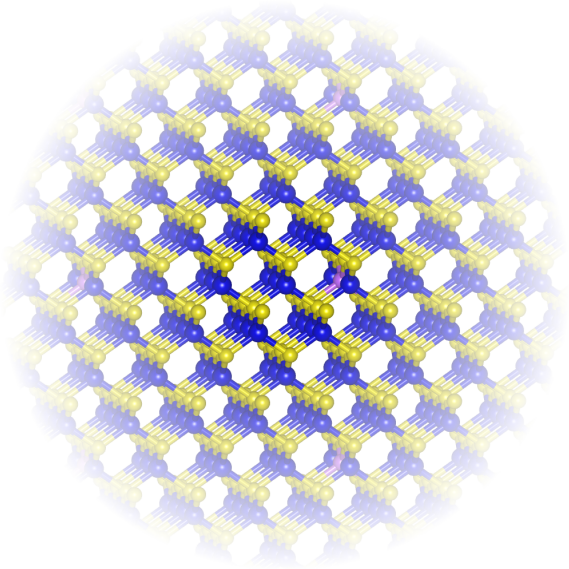
Hands-on Summer School: Electronic Structure Theory for Materials and (Bio)molecules

IPAM, UCLA, Los Angeles
July 21 – August 1, 2014



Tutorial II: Periodic systems
Manuscript for Exercise Problems

Prepared by Franz Knuth, Lydia Nemec, Björn Bieniek and Sergey Levchenko

Fritz-Haber-Institut der Max-Planck-Gesellschaft

Los Angeles, July 23, 2014

Introduction

This tutorial aims to familiarize you with the basic concepts of periodic density functional theory (DFT) calculations and with the settings necessary to run FHI-aims. Before we start working on the first problem, a short overview is provided.

The practice session consists of three parts:

Part I: Basic properties of solids and convergence tests

- Problem I: Generation and visualization of bulk structures
- Problem II: Energy convergence tests
- Problem III: Phase stability and cohesive properties
- Problem IV: Unit cell relaxation
- Problem V: Electronic band structure & density of states

Part II: Basic surface calculations

- Problem VI: Electronic structure of crystal surfaces
- Problem VII: Relaxing surface structures

Part III: Magnetism and collinear spin calculations

- Problem VIII: Magnetic $Ga_3Mn_1As_4$
- Problem IX: Ferromagnetic and anti-ferromagnetic $Ga_{0.75}Mn_{0.25}As$

You should work through all the problems of part I to learn about the basic concepts of periodic systems. Afterwards, you can choose whether you want to proceed with part II or part III depending on your liking. If you have time left after you finished either part II or III, please proceed with the other part.

In the directory `$HANDSON/hands-on-2014-tutorials/tutorial_2/`, you can find all the files necessary for this tutorial. Please copy the `skel/` folder to your own working directory. Dedicated folders have been prepared in the `skel/` directory for each problem. Please use this directory hierarchy, in particular because a few of the directories contain some helpful files. One of the first problems introduces two scripts. The first one prepares and starts FHI-aims calculations on a series of geometries, and the second one post-processes the resulting FHI-aims output. Please try to adapt these scripts to the other problems along the rest of this tutorial.

In case you get stuck with a particular problem, do not hesitate to ask one of the tutors. In any case, it is perfectly fine to skip the rest of a problem and move on to the next. This also applies if your calculation takes significantly longer than the estimated CPU time for the given problem. Any intermediate results required for later problems are provided in the `reference/` folder. If you like, you can also use this folder and compare to your results.

Take your time to read the tasks carefully before proceeding to actually prepare the calculations. Each subtask starts with a short summary (gray box) and gives details and hints afterwards. Also, feel free to consult the supplementary information presented in the Appendices.

Part I: Basic properties of solids and convergence tests

In the first part of this tutorial, we will work on different structural phases of bulk silicon. The correct description of their pressure dependence by Yin and Cohen [1] is one of the early success stories of computational materials science. In this part, we show how to calculate basic properties of solids like lattice constants, cohesive energies, band structures, and density of states.

Please use the basic settings given in Fig. 1 as default for part I of this tutorial (unless specified otherwise).

```
# Physical settings
  xc                  pw-lda
  spin                none
  relativistic        atomic_zora scalar

# SCF settings
  sc_accuracy_rho     1E-4
  sc_accuracy_eev     1E-2
  sc_accuracy_etot    1E-5
  sc_iter_limit       40

# k-grid settings  (to be adjusted)
  k_grid  nkx nky nkz
```

Figure 1: Default physical and computational settings for `control.in` for part I. This file can be found in `skel/problem_1/control_part1.in`.

The Perdew-Wang LDA (`xc pw-lda`) exchange-correlation functional will be used for all calculations. The effect of using different xc functionals has been discussed in "Tutorial 1: Basics of Electronic-Structure Theory". Silicon is known to be non-magnetic, so no explicit spin treatment is needed. The "`relativistic atomic_zora scalar`" setting is not strictly necessary for silicon. The nuclear charge of silicon ($Z = 14$) is still small enough to allow for a non-relativistic treatment. But as the correction is computationally inexpensive, it does not hurt to use it, either. Just be sure to never compare total energies from different relativistic settings. The SCF settings have been discussed in detail in "Tutorial 1: Basics of Electronic-Structure Theory", too. The `k_grid` setting will be discussed in the actual exercises.

Please use the default "light" species settings for Si in:
`$HANDSON/aimsfiles/species_defaults/light/14_Si_default`.

Problem I: Generation and visualization of bulk structures

Our first step towards studying periodic systems with FHI-aims is to construct periodic geometries in the FHI-aims geometry input format (`geometry.in`) and visualize them. As a next step, we learn how to set basic parameters in `control.in` for periodic calculations. Finally, we compare total energies of different Si bulk geometries.

Setting up and visualizing `geometry.in`

- Construct `geometry.in` files for the Si *fcc*, *bcc*, and diamond structures. Use the approximate lattice constants a of 3.8 Å for *fcc*, 3.1 Å for *bcc*, and 5.4 Å for the diamond structure.
- Visualize them (e. g. with Jmol).

To set up a periodic structure in FHI-aims, all three lattice vectors as well as the atomic positions in the unit cell must be specified. The lattice vectors are specified by the keyword `lattice_vector`. There are two ways to specify the atomic positions. As in the cluster case, you can specify absolute Cartesian positions with the keyword `atom`. Alternatively, you can specify the atomic positions in the basis of the lattice vectors, the so called **fractional** coordinates, with the keyword `atom_frac`. The fractional coordinates s_i are dimensionless and the coefficients for the linear combination of the lattice vectors $\mathbf{a}_i$. Written out as a formula, this linear combination reads as follows

$$\mathbf{R} = s_1 \cdot \mathbf{a}_1 + s_2 \cdot \mathbf{a}_2 + s_3 \cdot \mathbf{a}_3, \tag{1}$$

where $\mathbf{R}$ is the Cartesian position of the specified atom.

For example, *fcc* Si with a lattice constant $a = 4$ Å is defined by

```
lattice_vector   0.0   2.0   2.0
lattice_vector   2.0   0.0   2.0
lattice_vector   2.0   2.0   0.0
atom_frac        0.0   0.0   0.0   Si
```

A full set of lattice vectors and atomic positions of primitive unit cells (the unit cell with the smallest volume, that contains only one point of the crystal lattice) for *fcc*, *bcc*, and diamond can be found in Appendix I. In fig. 15 in Appendix I the simple cubic and primitive unitcells of *fcc*, *bcc*, and diamond are shown. Be sure to translate the given atomic positions (which are in Cartesian coordinates) to the right value for either the `atom` or the `atom_frac` keyword. The simplest way to check the `geometry.in` file is to visualize the corresponding geometry. For periodic structures in FHI-aims, we recommend Jmol, an open-source Java viewer for chemical structures in 3D. Information on the program and the source code can be obtained from `http://www.jmol.org`. To visualize a structure given in `geometry.in` with Jmol (Fig. 2), please type

```
jmol geometry.in &
```

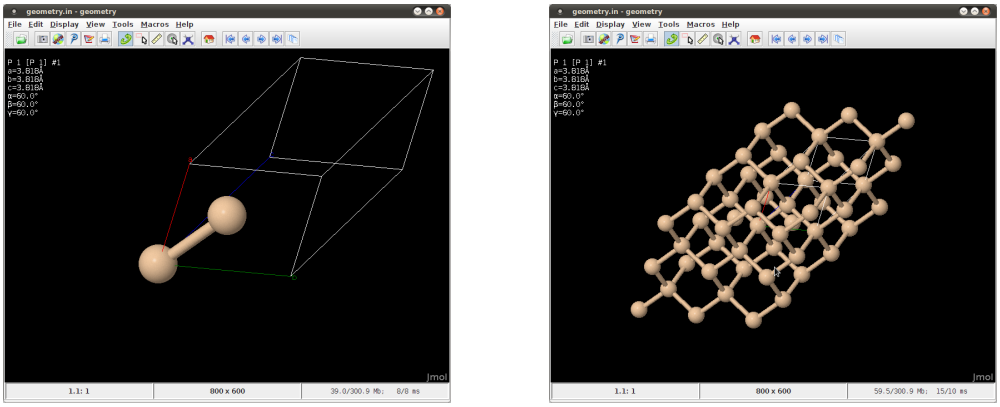


Figure 2: Single unit cell (left) and periodic images (right) of diamond Si in Jmol.

Make sure the lattice vectors in your `geometry.in` are referenced before your atomic positions to visualize the geometry properly with Jmol. To get periodic images, click with the right mouse button inside the Jmol drawing area and choose "Symmetry" → "Reload {444 666 1}".

Setting up `control.in` and running FHI-aims

- Prepare a `control.in` file using 3×3×3 k-points and the settings given in the introduction of part I (see Fig. 1). These settings can be found in `skel/problem_1/control_part1.in`.
- Use the provided `run.sh` script (Fig. 3) to calculate total energies of the different phases as a function of lattice constant a. For this, consider 7 different values of a in steps 0.1 Å around the lattice constants given above for each structure.

[Estimated total CPU time: 2 min]

The `control.in` for periodic calculations looks much the same as for the cluster case (the underlying numerics are the same). There is one important difference, though, as a k-grid for the Brillouin zone integrations must be specified. For example, to specify a 3×3×3 k-grid, the following line must be added to `control.in`:

```
k_grid 3 3 3
```

Note that the k-grid points refer to the reciprocal lattice vectors corresponding to the real-space lattice vectors in `geometry.in`. If there are inequivalent lattice vectors, their order in `geometry.in` determines the ordering of reciprocal lattice vectors in the code. You will see that for a small unit cell this k-point density is *never* enough. But we can try to find out what happens. In the next problem, the k-point settings will be discussed in detail.

In general, you can run FHI-aims just like in the cluster case by

```
mpiexec -n 4 aims.x | tee aims.out
```

which starts a calculation with 4 processors. It is good practice to use a separate directory for every single run of FHI-aims in order to preserve the exact input files along with the output files.[1] In this tutorial, however, most of the calculations can be started using a prepared script which takes care of these things and needs to be adjusted only slightly.

In this exercise, we compare energies of different lattice structures as a function of lattice constant. Each calculation can be prepared and started by hand, in principle, but we strongly suggest to use the shell script `run.sh` provided in `skel/problem_1/02_3x3x3/fcc/` and shown in Fig. 3. This example script calculates the total energy of *fcc* Si with different lattice constants. Please copy this script to dedicated folders for *bcc* and diamond Si (we suggest `problem_1/02_3x3x3/bcc/` and `problem_1/02_3x3x3/diamond/`), and adjust the copies according to the lattice constants given in this subtask and the lattice structures given in Appendix I.

```
#!/bin/bash -l
set -e   # Stop on error
for A in 3.5 3.6 3.7 3.8 3.9 4.0 4.1; do
    echo "Processing lattice constant $A AA."
    mkdir $A
    # Use this construct for simple calculations. As values
    # are replaced verbatim, always put them into "(", ")".
    A2=$(python -c "print ($A)/2.")
    # Write geometry.in
    cat <<EOF >$A/geometry.in
# fcc structure with lattice constant $A AA.
lattice_vector    0.0   $A2   $A2
lattice_vector    $A2   0.0   $A2
lattice_vector    $A2   $A2   0.0
atom_frac         0.0   0.0   0.0  Si
EOF
    # Write control.in
    cp control.in $A/
    # Now run fhi-aims with 4 processors in directory $A
    cd $A
    mpiexec -n 4 aims.x >aims.out
    cd ..
done
```

Figure 3: Example input shell script for running calculations for several lattice constants. This file can be found in `skel/problem_1/02_3x3x2/fcc/run.sh`.

[1] Using a separate directory for each FHI-aims run interactively is slightly simplified by using the `run_aims.sh` script. See `$HANDSON/aimsfiles/utilities/run_aims.sh --help` for help.

```
#!/bin/bash -l
set -e   # Stop on error
echo "# lattice constants in AA, energies in eV" >energies.dat
for A in 3.5 3.6 3.7 3.8 3.9 4.0 4.1; do
    # Check for convergence of calculation.
    if (! grep --quiet "Self-consistency cycle converged." \
            <$A/aims.out) || \
       (! grep --quiet "Have a nice day." \
            <$A/aims.out); then
        echo "`pwd`/$A/aims.out did not converge!"
    fi
    # Get 6th column from the line with "Total energy corr".
    E=$(gawk '/Total energy corr/ {print $6}' $A/aims.out)
    # Write results to data file.
    echo "$A  $E" >>energies.dat
done
```

Figure 4: Example shell script to retrieve the total energy from the FHI-aims output file. This file can be found in `skel/problem_1/02_3x3x2/fcc/postprocess.sh`.

To retrieve the total energies, you should then use the `postprocess.sh` script, which is provided within the same folder and printed in Fig. 4. This script extracts the total energies and writes them to the file `energies.dat`, along with the lattice constants. You will need to adapt this script to the other phases of silicon. In particular, adjust the lattice constants. For the next subtask, it is advantageous to write out the total energy per atom, not per unit cell, which makes a difference for the diamond structure. You can use the expression `Eatom=$(python -c "print ($E)/2.")`, which calculates the energy per atom for the diamond structure and puts it into the bash variable `$Eatom`. You can then write this variable instead of `$E` to the data file.

Plotting total energies

- Plot the total energy per atom of each structure as a function of the lattice constant (e. g. with xmgrace).
- What is the most stable bulk phase of Si according to your results?

After running the code, plot your data (given in `fcc/energies.dat`, `bcc/energies.dat`, and `diamond/energies.dat`) using for example xmgrace by typing:

```
xmgrace -legend load \
  fcc/energies.dat bcc/energies.dat diamond/energies.dat &
```

You should see that, with the current computational settings, the diamond Si phase is unfavorable compared to the other two phases by about 0.1 eV. However, the exper-

imentally most stable phase is the diamond structure. We will show in the next two problems that the too coarse 3×3×3 k-grid is the reason for this disagreement.

Problem II: Energy convergence tests

The results of the last problem were not quite physical. As will be shown later, this is because the convergence was not properly checked. Here, we will explicitly check total energy convergence with respect to the k-grid and to the basis set. In principle, each phase needs to be checked separately. Within this tutorial, however, we split the effort and everyone should only check **one** phase of their own choice.

Convergence with k-grid

- Calculate the total energies for only **one** of the Si phases of Problem I as a function of the lattice constant for k-grids of 8×8×8, 12×12×12, and 16×16×16. Otherwise, use the same computational settings (`control.in`) and the same lattice constants as in Problem I.
- Prepare a plot with all total energies drawn against lattice constant. Add the 3×3×3 results from Problem I, too.
- Which k-grid should be used to achieve convergence within 10 meV?

[Estimated total CPU time: 3 min]

You should dedicate one directory for every series of these calculations. You will find empty folders in `skel/problem_2/`. These calculations should be done exactly as in the last problem but with the appropriate changes to `control.in`. In particular, use the scripts provided for the Problem I (Fig. 3 and Fig. 4).

In the metallic *fcc* and *bcc* phases, the total energy of the 3×3×3 calculation is about 0.3 eV lower than the most accurate (16×16×16) calculation. The larger part of this error is already fixed by the 8×8×8 k-grid, which is still off by about 30 meV. The 12×12×12 grid, on the other hand, is converged within about 5 meV. For the semiconducting diamond Si phase, the total energy of the 3×3×3 calculation is about 0.3 eV too high, but the convergence is already very good for an 8×8×8 k-grid. In general, metals (like Si *fcc* & *bcc*) or small cells require a denser k-grid compared to semiconductors (Si diamond) or large cells.

Increasing the number of k-points does not affect the computation times significantly up to comparably dense k-grids. A total energy calculation with a 12×12×12 grid is not so much more expensive than a 3×3×3 calculation. Only with even denser k-grids, computational times increase noteworthy.

In conclusion, we can use a 12×12×12 k-grid for *fcc* and *bcc* Si as a good compromise of high accuracy and reasonable computational time. For simplicity, we use the same grid also for diamond Si although 8×8×8 would be enough in that case.

Convergence with basis set size

- Calculate the total energies for **your** phase of Si as a function of the lattice constant for the `minimal` and the `tier1` basis sets. Use the same lattice constants and computational settings as in Problem I together with the 12×12×12 k-grid.
- Again, prepare a plot with the total energies. Add the results for the `minimal+spd` basis set (the default for the "light" species settings) from the k-point convergence test above.

[Estimated total CPU time: 2 min]

Use your control.in file from the previous problem for the 12x12x12 k-point grid. In order to change the basis size settings, you should have a look into the species-dependent settings within `control.in`. There, you will find a line starting with "`# "First tier" - ...`". Each line after this defines a basis function which is added to the `minimal` basis. Right now, there is one additional function for each valence function (s and p) as well as a d function to allow the atoms to polarize. This is what we call `minimal+spd` in the context of this tutorial. In quantum chemistry and in particular the Gaussian community, this type of basis set is often called "double zeta (ζ) plus polarization" (DZP).

To run FHI-aims with a `minimal` basis, simply comment out these three lines by prepending a "`#`" character. To run FHI-aims with a full `tier1` basis set, uncomment all four lines following "`# "First tier" - ...`" by removing the initial "`#`" character.

You can see that the `minimal` basis gives completely unphysical results; the energetic minimum is strongly shifted towards larger lattice constants. The `minimal` basis lacks the flexibility to give reasonable geometries. On the other hand, the binding curve does not change drastically from `minimal+spd` to the full `tier1` basis set whereas the computational effort increases significantly by adding the f functions from `minimal+spd` to full `tier1`. You could also check `tier2`. In that case, though, it would be worth using better integration grids (tighter) etc. as well.

While the total energy difference of about 60 meV between `minimal+spd` and `tier1` is still larger than what we were aiming for in the case of the k-grid, we can make use of the fact that the total energy is variational so that a large part of the basis set error actually cancels nicely in energy differences between bounded structures.

Bonus: Effect of Gaussian broadening of the Kohn-Sham occupation numbers

Bonus: Please skip this subtask if you run out of time.

- Calculate the total energies for *fcc* Si as a function of the lattice constant for a Gaussian broadening of $\sigma = 0.1\,\text{eV}$. Use the same lattice constants and computational settings as before with a 12×12×12 k-grid and the `minimal+spd` basis.
- Prepare a plot with the corrected, uncorrected total energies, and the electronic "free energies" for a broadening of $\sigma = 0.1\,\text{eV}$ and the default value of $\sigma = 0.01\,\text{eV}$ from the previous calculations.

[Estimated total CPU time: 1 min]

Use your control.in file from the previous problem for the 12x12x12 k-point grid and `minimal+spd` basis. You can explicitly set the Gaussian broadening to $\sigma = 0.1\,\text{eV}$ by specifying

```
occupation_type gaussian 0.1
```

in `control.in`.

FHI-aims always outputs three different energies. While these energies are all the same for systems with a gap, they differ for metallic systems with finite Gaussian broadening. The "`Total energy uncorrected`" gives the value of the Kohn-Sham energy functional for the final self-consistent electronic structure. However, due to the Gaussian broadening the self-consistency procedure does not minimize this total energy but a "free energy" specified right of "`Electronic free energy`". From these two numbers, FHI-aims backextrapolates to the total energy without broadening and writes the resulting number right of "`Total energy corrected`". For true metals, it is generally best to make use of the correction. For finite systems and in particular for isolated atoms, however, the back-extrapolation is unphysical and should not be used.

For diamond Si, the Gaussian broadening makes no difference at all as long as the broadening σ is small compared to the band gap. The first thing to notice for the metallic phases is that all of these numbers agree with each other within about 2 meV. For the default broadening of $\sigma = 0.01\,\text{eV}$, the energies even agree within 0.1 meV. It can be shown using the variational principle that the total energy always increases and the electronic "free energy" always decreases for finite broadening.

For the following calculations, we will use the default value of $\sigma = 0.01\,\text{eV}$ because there is no benefit in convergence by increasing this parameter for the studied phases of Si. We will stick to the corrected total energy for the periodic systems in this part of the tutorial as it is the most accurate value for metals and makes no difference for semiconductors.

Problem III: Phase stability and cohesive properties

After finding converged computational settings, we now revisit the phase stability of bulk silicon in Problem I. Of course in practice one should always check convergence *first* to avoid false conclusions as in Problem I. We will learn how to compute the basic cohesive properties and study the pressure dependence of phase stability.

Recalculation of $E(a)$ curves

- Calculate the total energy of *fcc*, *bcc*, and diamond Si as a function of lattice constant a. Use the settings from Problem II (k-grid of 12×12×12, `minimal+spd` basis) and the same lattice constants as in Problem I.
- Plot the results as done in Problem I.

[Estimated total CPU time: 2 min]

These tasks can be performed completely along the lines of Problem I. Use your `control.in` file from the previous problem for the 12x12x12 k-point grid and `minimal +spd` basis. The resulting binding curves clearly show that the experimentally observed diamond structure of silicon is most stable in LDA among the crystal structures studied in this tutorial. In the rest of this exercise, we will analyze the results obtained so far.

Cohesive energies and atomic volumes

- Calculate the total energy of a free Si atom as described in the text below.
- Figure out how to calculate the cohesive energies and the atomic volumes for all FHI-aims runs in the first subtask.
- Plot all cohesive energies of all three phases into one plot with the atomic volume on the x axis.

[Estimated total CPU time: <1 min]

For the single atom energy, special care has to be taken. First, the free atom is of course spin polarized, and you should use "`spin collinear`" instead of "`spin none`" as well as properly initialize the magnetization. For the free atom, this can be done by specifying "`default_initial_moment hund`". You can use the `control.in` from the previous task only removing the line specifying the k-grid.

Second, we use a more converged basis. In particular, uncomment all basis functions up to and including "tier 3", increase the confining potential to "`cut_pot 8. 3. 1.`", and turn off basis dependent confining potentials with "`basis_dep_cutoff 0.`"

in the `species` section of `control.in`.[2]

Additionaly, use the "`Total energy uncorrected`" instead of the "`Total energy corrected`" because the entropy correction is not physical for finite systems and in particular for atoms.

The cohesive energy (E_{coh}) of a crystal is the energy per atom needed to separate it into its constituent neutral atoms. E_{coh} is defined as

$$E_{\mathrm{coh}} = -\frac{E_{\mathrm{bulk}} - N E_{\mathrm{atom}}}{N} = -\left[\frac{E_{\mathrm{bulk}}}{N} - E_{\mathrm{atom}}\right], \tag{2}$$

where E_{bulk} is the bulk total energy **per unit cell** and N the number of atoms in the unit cell. E_{atom} is the energy of the isolated atom calculated above.

In order to compare the pressure dependence of phase stabilities, we need to express the lattice constant behavior of all phases on equal footing. One possibility to do so is to express the lattice constant in terms of the volume **per atom**. This atomic volume can be calculated quite easily from the lattice constant a. The simple cubic (super-)cell has the volume $V_{\mathrm{sc}} = a^3$. This number has to be divided by the number of atoms N_{sc} in this cell $V_{\mathrm{atom}} = a^3/N_{\mathrm{sc}}$. Please verify that there are two, four, and eight atoms in the **simple cubic** supercell in the case of the *bcc*, *fcc*, and the diamond structure, respectively. Have a look at fig. 15 in Appendix I.

In summary, a file `energies.dat` containing the lattice constants and total energies per atom can be converted to a file `cohesive.dat` containing atomic volumes V_{atom} and (negative) cohesive energies $-E_{\mathrm{coh}}$ by the script `convert-coh.awk` given in `skel/problem_3/`. You can use the script by typing:

```
convert-coh.awk Nsc=.. Eatom=.. <energies.dat >cohesive.dat
```

where the "`..`" need to be replaced by the corresponding values of N_{sc} and E_{atom}, respectively. Please do not forget to specify the correct path to the script.

After plotting with `xmgrace -legend load */cohesive.dat`, you see that the diamond structure is indeed the energetically most stable phase. But it is considerably more space consuming. This results from the very open structure of the diamond phase compared to the much closer packing of the *bcc* and *fcc* phases. It is plausible that upon high pressure, phases with more compact atomic volumes might be favorable.

[2] In most other finite-basis approaches, it is common to ensure equivalent basis sets for energy differences. The numerical atomic orbitals of FHI-aims, however, treat the free atom in principle exact already with a minimal basis, and no such cancellation of errors is needed. We make use of this fact by using one and the same atomic reference for all basis sizes used for the compounds. This way, basis set convergence can be checked more easily as cohesive energies strictly increase with increasing basis size.

Equation of states and pressure dependence

- Fit the cohesive energy data for the three phases to the Birch-Murnaghan equation of states using the program `murn.py`.
- Determine the lattice constant a, the bulk modulus B_0, and the cohesive energy per atom E_{coh} at equilibrium.
- Compare the above quantities for the diamond phase with the experimental values of $a = 5.430\,\text{Å}$, $B_0 = 98.8\,\text{GPa}$, and $E_{\mathrm{coh}} = 4.63\,\text{eV}$ [2].
- Plot the cohesive energies $E(V)$ with respect to the atomic volume for all three phases.
- *Bonus:* Given this data, can you estimate at what pressure a phase transition would occur? Recall the Maxwell construction (see below).

The equilibrium lattice constant a_0 is an important quantity which we can calculate from our data. In principle, this can be done with a quadratic ansatz for $E(a)$ or $E(V)$. Here, we will discuss and use a thermodynamically motivated and more accurate fitting function, the Birch-Murnaghan equation of states [3, 4]. The energy per atom ($E = -E_{\mathrm{coh}}$) is expressed as a function of the atomic volume ($V = V_{\mathrm{atom}}$)

$$E(V) = E_0 + \frac{B_0 V}{B_0'}\left[\frac{(V_0/V)^{B_0'}}{B_0' - 1} + 1\right] - \frac{B_0 V_0}{B_0' - 1}. \tag{3}$$

The fitting parameters V_0 and E_0 are the equilibrium atomic volume and atomic energy, respectively, B_0 the bulk modulus and B_0' its derivative with respect to pressure. Equation (3) can be derived by assuming a constant pressure derivative B_0'.

The fitting program `murn.py` is part of the FHI-aims distribution. You can get usage information by typing `$HANDSON/aimsfiles/utilities/murn.py --help`. If you have prepared the files `cohesive.dat` using the provided gawk script, you can simply use the script with

```
$HANDSON/aimsfiles/utilities/murn.py cohesive.dat -o fit.dat
```

The program then outputs the parameters V_0, E_0, B_0, and B_0' for the given data set as output. As a quick plausibility check of the fit, you can use the option `-p` to see a plot. The script performs no unit conversions, so the bulk modulus B_0 is given in units of eV/Å^3 because the cohesive energies and atomic volumes were provided in eV and Å^3, respectively. You can use "GNU units" to convert to SI units. For example, use

```
units -v "0.5 eV/angstrom^3" "GPa"
```

to convert 0.5 eV/Å^3 to about 80 GPa. The optimal lattice constant can be calculated from the equilibrium atomic volume V_0 by $a_0 = \sqrt[3]{N_{\mathrm{sc}} V_0}$ with N_{sc} the number of atoms in the cubic unit cell.

Compare the calculated results with experimental reference values given above. *Note:* Exact agreement between DFT and experimental data is not our goal right here – DFT-LDA is an approximation, and we here see how well (or not) it works. It is well known that LDA in general gives only slightly overbound lattice constants and cohesive energies.

After performing the Birch-Murnaghan fit for all three phases, please plot the resulting fitted curves saved in `fit.dat` into one figure. You should get something similar to Fig. 1 in the paper by Yin and Cohen [1].

By exposing the crystal to the right pressure, one can enforce many different atomic volumes smaller than the equilibrium ones. In principle, the most stable phase for a given atomic volume V can simply be deduced from the curve with the lowest $E(V)$. The corresponding pressure can be calculated from the slope of the curve by the simple thermodynamic relation $p = -\partial E/\partial V$.

However, in the regime of about $13\,\text{Å}^3 < V_{\text{atom}} < 18\,\text{Å}^3$ coexistence of a diamond phase at $\sim 18\,\text{Å}^3$ and a *bcc* phase at $\sim 13\,\text{Å}^3$ is favorable (see fig. 5). The fraction of atoms in the two phases then determines the average atomic volume. The resulting average atomic energy is a straight line between the corresponding points which is below both the lines of diamond and *bcc* Si. This is called the Maxwell construction. From the slope of this line (a common tangent), we can deduce a transition pressure of about 14 GPa at which diamond and *bcc* Si could coexist according to our calculations. This is more than 10^5 times the ambient pressure of about 100 kPa. Note that there are additional phases for silicon which have not been calculated here. In fact, the Si β-tin phase is more stable than the *bcc* phase at high pressures. For a more thorough discussion, please refer to [1].

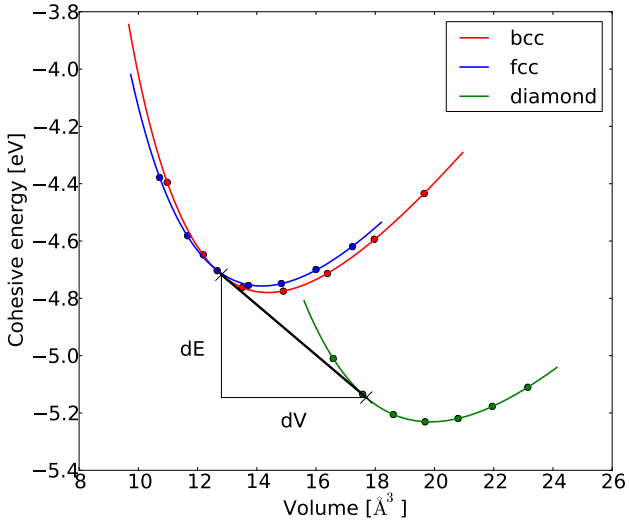


Figure 5: Maxwell construction

Problem IV: Unit cell relaxation

You have learned in "Tutorial 1: Basics of Electronic-Structure Theory" how to optimize the positions of atoms in cluster systems. To obtain optimized periodic structures, not only the atomic positions must be optimized but also the lattice vectors itself. We will discuss in the following how this can be done.

- Fully (atoms & lattice vectors) relax a distorted *bcc* Si structure (see below) with the computational settings you have used before for the Si crystals.
- Analyze the resulting structure (angles between lattice vectors, their length, and atomic positions) and compare it to the equilibrium values of diamond Si obtained from the Birch-Murnaghan fit in the previous problem.

[Estimated total CPU time: 3 min]

The basic idea is to perform a structure relaxation from *bcc* to diamond Si. Since we perform a local structure optimization, we cannot go uphill in the energy landscape or cross energy barriers. Thus, there must be a path from the starting structure to the desired local minimum with no barriers in between. Otherwise, the optimization will get stuck in a different local minimum. In order to cross energy barriers, one can do for example variable cell-shape molecular dynamics, but this is out of the scope of this tutorial.

It is crucial that the number of atoms in the starting unit cell is compatible with the desired structures. Since we want to end up in a diamond structure, we have to start with at least 2 atoms. Hence, our starting unit cell is a cubic *bcc* structure consisting of 2 atoms. Of course, we cannot start with the ideal cubic *bcc* structure because we will be stuck in the local minimum of this high symmetry structure. Even if we distort the starting geometry significantly (atomic positions & lattice vectors), we can easily end up in an unwanted local minimum. Therefore, we will provide a suitable starting geometry, but feel free to try other ones when you are finished with the given one. You can find this geometry in the directory `skel/problem_4/`:

```
lattice_vector   3.1   0.4   0.4
lattice_vector   0.45  3.1   0.4
lattice_vector   0.45  0.45  3.1
atom_frac        0.0   0.0   0.0  Si
atom_frac        0.3   0.3   0.3  Si
```

To perform a full optimization of the crystal structure, you have to add two lines to your `control.in` from the previous problem. The line

```
relax_geometry trm 1E-2
```

requests a structure relaxation for atoms to forces smaller than 10^{-2} eV/Å and with the line

```
relax_unit_cell full
```

the optimization of the lattice vectors is enabled. The convergence criterion will be the same as the one specified for the atoms. Additionally, it is sufficient to calculate the forces with an accuracy of 10^{-3} eV/Å given the desired relaxation accuracy above. Therefore, add the following line, too:

```
sc_accuracy_forces 1E-3
```

The structure optimization with the provided starting geometry should take 14 iterations. If you take a closer look into the FHI-aims output, you will see that not only the atomic forces are calculated but also a quantity called stress tensor. In essence, the stress tensor is a measure of the forces acting on the unit cell itself.

You can have a look at geometries along the relaxation path if you run the script `$HANDSON/aimsfiles/utilities/create_geometry_zip.pl` and specify your output file as an argument of this script. Unzip the resulting file `geometries.zip` if you want to see the individual geometries or use Jmol by typing `jmol -s geometries.spt`.

The resulting geometry is a primitive diamond structure. The angles between the lattice vectors are about 60° and the length of the lattice vectors matches approximately the result from the Birch-Murnaghan fit in the previous problem. The vector connecting the two atoms is $(0.25, 0.25, 0.25)$ in fractional coordinates as it should be for a diamond structure. The convergence criterion for the relaxation is quite loose, a higher accuracy (e.g. 10^{-3} eV/Å) would lead to better results. Increasing the basis set size improves the resulting geometry, too, but you would need to do the same for the Birch-Murnaghan calculation. Ideally, you start a relaxation with "light" settings. Then, you use the resulting geometry (e.g. from the file `geometry.in.next_step`) as an input for a calculation with "tighter" settings.

Problem V: Electronic band structure & density of states

The electronic band structure describes the dispersion relation of electrons inside a solid. The band structure gives information about ranges of energy that electrons are allowed to have or not. Permitted ranges are called bands, forbidden ones band gap. Many properties of a solid can be deduced from its band structure, e.g., if it is a metal, semiconductor, or insulator.

- Calculate the electronic structure of diamond Si using the equilibrium geometries found in Problem III.
- Choose the band structure along the high symmetry lines
  $L \to \Gamma \to X \to W \to K$.
- For the DOS use the energy range of -18 eV to 0 eV, Gaussian broadening of 0.1 eV, a k-grid of $12\times12\times12$, and `dos_kgrid_factors` of 5 for each k-grid direction.

[Estimated total CPU time: 1 min]

Use your `control.in` and `geometry.in` for the equilibrium lattice parameter for diamond Si from Problem III. In order to calculate the band structure, we have to specify the high symmetry points in the reciprocal space to FHI-aims. An example excerpt from `control.in` corresponding to the first part of the suggested path with 50 points per path reads like this:

```
# diamond band structure:
output band  0.5   0.5   0.5    0.0   0.0   0.0    50    L  Gamma
output band  0.0   0.0   0.0    0.0   0.5   0.5    50    Gamma  X
output band  ...
```

In each line, the first three numbers are the coordinates of the starting point in units of the reciprocal lattice vectors. The next three numbers specify the end point. The band structure is then calculated along the path connecting these two points.

The last two entries are not mandatory for the calculation inside FHI-aims, but they provide the label of the specified k-points for the plotting script which you are going to use after the calculation.

Please refer to Appendix II for the location of the other high symmetry points in the Brillouin zone.

The density of states (DOS)

The density of states is one of the basic concepts in solid state physics. Particularly, the DOS around the Fermi level is of interest as it will give you information about the characteristic of your system, for example, whether it is conducting, semiconducting, or insulating. Many material properties depend on the DOS for instance the conductivity.

The number of states n within a given energy interval $(\epsilon_0 - \Delta\epsilon) < \epsilon < (\epsilon_0 + \Delta\epsilon)$ per unit volume V_{cell} is given by

$$n = \int_{\epsilon_0 - \Delta\epsilon}^{\epsilon_0 + \Delta\epsilon} \mathrm{d}\epsilon \, g(\epsilon) \tag{4}$$

where $g(\epsilon)$ is the density of states (DOS). In a free atom or an isolated molecule, the DOS consists of a series of discrete energy levels (δ peaks) and can be written as

$$g(\epsilon) = \sum_i \delta(\epsilon_i - \epsilon). \tag{5}$$

In a periodic system, the single particle energies become k dependent and the DOS continuous. The number of states per energy is averaged over k

$$g(\epsilon) = \frac{1}{V_{\text{BZ}}} \sum_i \int_{\text{BZ}} \mathrm{d}^3k \; \delta(\epsilon_{i,k} - \epsilon). \tag{6}$$

In order to calculate the density of states numerically, we have to replace the integral over the Brillouin zone (BZ) in Eq. (6) by a sum over k-points. In the case of infinite k-points, this replacement is exact. However, to compensate the deficiency of a finite grid, we broaden the $\delta(\epsilon_{k,i} - \epsilon)$ distribution by a Gaussian function with an broadening factor σ,

$$g(\epsilon) = \frac{1}{\sqrt{2\pi}\sigma} \frac{1}{n_k} \sum_i \sum_k \exp\left[-\frac{1}{2} \left(\frac{\epsilon - \epsilon_{k,i}}{\sigma} \right)^2 \right]. \tag{7}$$

Within FHI-aims, we have to specify in the `control.in`:

```
output dos -18. 0. 200  0.1
```

The first two values define the energy window of interest, the first value is the lower energy bound and the second value is the upper energy bound. The third value is an integer specifying at how many energy data points the DOS is evaluated and the last value gives the Gaussian broadening σ. All energies (bounds and broadening) are given in eV.

Small changes in the shape of a peak in the DOS barely affect the total energy. Therefore, a rather coarse k-grid (defined by the keyword `k_grid`) in combination with rather broad choices of σ (given by `occupation_type gaussian`) can be used for the self-consistent field procedure. However, for a well resolved DOS a denser k-grid to determine the sum over k-points in Eq. (7) is necessary. After self-convergence is reached, the DOS is computed using an auxiliary k-grid that is made denser by factors n_1, n_2, n_3, respectively. The factors n_1 to n_3 have to be given in the `control.in` with the keyword:

```
dos_kgrid_factors 5 5 5
```

The density of states is calculated on a denser grid after the SCF cycle. The dimensions of the new k-point grid are $k_1 \times n_1$, $k_2 \times n_2$, $k_3 \times n_3$, where k_i are dimensions of the old k-point grid.

Visualization of band structure and DOS

- Use the python script `aimsplot.py` to visualize the band structure data and the DOS. How large is the LDA band gap?

In order to visualize the band structure, some postprocessing is needed after the FHI-aims run. Fortunately, the script `aimsplot.py` is smart enough to do so as long as the output, `geometry.in`, and `control.in` files are in the same directory. Simply run `$HANDSON/aimsfiles/utilities/aimsplot.py` without any arguments in this directory.

You see a band structure with an indirect band gap of about 0.5 eV. Please note that in contrast to convention, the energy zero is at the Fermi energy and not at the valence band maximum. The resulting LDA band gap is much smaller than the experimental band gap of silicon (1.17 eV [2]). This disagreement is commonly called the "band gap problem" of (semi-)local functions.

You can now chose how you want to proceed. You can *either* do part II which deals with surfaces *or* part III (page 28) which covers magnetism. If you have time left after you finished either part II or III, please proceed with the other part. You are also encouraged to work on unfinished exercises during the extra computer time in the evening.

Part II: Basic surface calculations

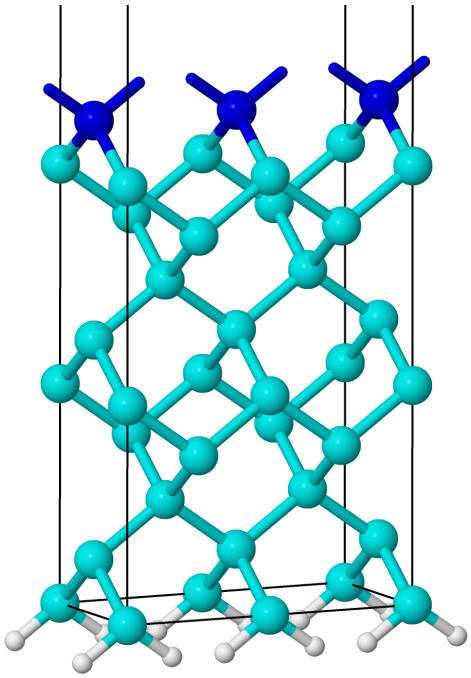


Figure 6: The hydrogen saturated ideal 2×1 Si(001) surface. The cyan (light) atoms correspond to the bulk Si atoms, the blue (dark) atoms are the top Si atoms. The white atoms on the bottom layer are hydrogen atoms.

In the second part of this tutorial, we come to standard techniques needed to describe surfaces. As a physical system, we use the clean Si(001) surface, one of the technological most relevant surfaces. The surface reconstruction (due to dangling silicon

bonds at the surface) at low temperatures were unclear for many years. There was a long discussion and different models were used to understand the differences between various experimental (LEED, STM) and theoretical approaches. Finally, it has been shown by direct evidence in STM by Wolkow [5] that the main surface feature at low temperatures is the asymmetric dimer in a 2×1 reconstructed surface unit cell.

Please use the settings given in Fig. 7 for part II of this tutorial together with the "light" species defaults for silicon. These settings are the same as for part I, thus, if you want, you can take your `control.in` from there and only adjust the k-grid settings. For saturating the bottom layer of the slab, we additionally need the "light" species defaults for hydrogen which you can find in `$HANDSON/aimsfiles/species_defaults/light/01_H_default`. As you can see in Fig. 6, each silicon atom has to be saturated by two hydrogen atoms.

```
# Physical settings
  xc                  pw-lda
  spin                none
  relativistic        atomic_zora scalar

# SCF settings
  sc_accuracy_rho     1E-4
  sc_accuracy_eev     1E-2
  sc_accuracy_etot    1E-5
  sc_iter_limit       40

# k-grid settings
  k_grid    12  24  1
```

Figure 7: Physical and computational settings for `control.in` for part II. This file can be found in `skel/problem_6/control_part2.in`.

Problem VI: Electronic structure of crystal surfaces

Creation and visualization of `geometry.in`

- Use the provided Python script (Appendix III) to construct an ideal bulk-truncated diamond Si(100) surface slab in a 2×1 cell consisting of four Si layers and an additional layer of H atoms to saturate the bottom layer. Use sufficient vacuum between the slabs.
- Adjust the Python script to add four additional layers of Si atoms.
- Visualize the surface slabs.

The geometry of four layers of Si(001)-(1×1) in diamond structure is given in the Appendix I. Please note that by convention the surface is rotated with respect to the bulk structure by 45° around the z axis. The next four layers are just a repetition of these layers shifted by a in z direction. The (2×1) reconstructed surface is constructed by (again) repeating the atomic coordinates, this time shifted by $a/\sqrt{2}$ in x direction and doubling the corresponding lattice vector. Please use the optimized lattice constant obtained in the last problem.

Obviously, a slab always has two surfaces, a top and a bottom one (see Fig. 6). In most cases, one is only interested in one of these surfaces. In order to avoid physical states from the bottom layer within the fundamental gap, the bottom silicon layer is saturated with hydrogen atoms. Additionally, one can argue that the atomic environment of the hydrogen saturated silicon atoms is closer to bulk silicon atoms.

Each bottom layer silicon atom needs two hydrogen atoms placed about 1.5 Å in the direction where the next silicon atoms would have been in the bulk geometry.

For your convenience, we provide a simple python script `write-geom.py` (the source code is given in Appendix III) for the ideal hydrogen saturated four layer slab. The script is provided in `skel/problem_6/01_ideal_2x1_4_layers/`. You should only adjust the lattice constant in line 14. By typing `./write-geom.py >geometry.in`, the script is executed, and the generated structure data is written to the file `geometry.in`.

The eight-layer slab can be created either by hand-editing the resulting `geometry.in` or by editing the python script. You can edit the python script without being an experienced python user. If you read through the code, either on your screen or in Appendix III, you will find in line 17 the variable `n_layer`. This variable gives the number of layers. In line 28, it is used to determine the slab thickness. You have to change the value from "`4`" to "`8`" to ensure you use the same vacuum thickness as you used before.

As a next step, you have to add the atoms of the next four layers. For every layer specify two atoms. This is done by using the `output_atom` command, just like in line 39. The easiest way is to copy paste lines 39–47 and change the number of the layer in the third argument. Run the script and visualize the resulting `geometry.in` file.

To determine a sufficient amount of vacuum between slabs, you would actually need to run several calculations with different vacuum thickness. In FHI-aims vacuum

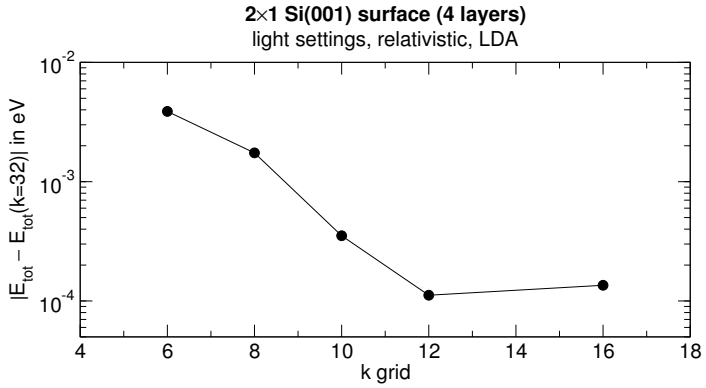


Figure 8: Convergence test for the k-grid of a 4 layer 2×1 Si(001) surface.

is cheap, you can easily use a vacuum of $L_{\text{vac}} = 30\,\text{Å}$ or more without noticeable performance impact. This value is already given in the script by the variable `L_vac` in line 15.

Band structure and DOS calculation

- Prepare a `control.in` according to the specified settings.
- Calculate the density of states (DOS) and the surface band structures for 4 and 8 layer slab along $\bar{\Gamma} \to \bar{J} \to \bar{K}$ (see Appendix II).

[Estimated total CPU time: 1+3 min]

There are two important issues to note for the optimal k-grid for this system. First, there should be no interaction between different periodic images of the slab in z direction. Therefore, only one k-point is needed on that axis. Second, the lattice vector in x direction is twice as large as the lattice vector in y direction. As this gives a shorter periodicity in k_y direction, the number of k-points in the first direction can be half of that in the second direction. We use a well converged k-grid of $12 \times 24 \times 1$ (see Fig. 8).

Similar to part I, for calculating the band structure and DOS, add the following lines in the `control.in` file:

```
# Si 2x1 surface band structure:
output band  0.0   0.0   0.0    0.5   0.0   0.0    50    Gamma  J
output band  0.5   0.0   0.0    0.5   0.5   0.0    50    J  K
```

and for the DOS

```
output dos -18. 0. 200  0.1
dos_kgrid_factors 5 5 1
```

To visualize the band structure, use the script `$HANDSON/aimsfiles/utilities/aimsplot.py` just like before. Simply run the script in the directory that contains the input and output files of FHI-aims.

When plotting the band structures as before, note the differences between the four- and the eight-layer slabs. While the bands within the fundamental band gap, which stem from the surface, hardly change with the number of layers, the bands get about twice as dense in the valence and the conduction bands. These bands are bulk-like; the more layers you add, the more of these bands occur.

Problem VII: Relaxing surface structures

- Create a `geometry.in` with the two top Si atoms of the four-layer geometry in Problem VI perturbed.
- Set up a calculation to perform a structure relaxation of the top three layers of the resulting geometry.
- Specify in your `control.in` a calculation of the DOS and the band structure along $\bar{\Gamma} \to \bar{J} \to \bar{K}$.
- Run the structure relaxation with the above settings.
- Compare this band structure to the ones obtained in the last Problem.

[Estimated total CPU time: 7–21 min]

The DOS and the band structure can be calculated by specifying the same lines in the `control.in` as in the task before.

In order to perform structure relaxations to forces smaller than 10^{-2} eV/Å, add the following line to `control.in`:

```
relax_geometry trm 1E-2
```

Do not try to relax the unit cell, though. Additionally, it is sufficient to calculate the forces with an accuracy of 10^{-3} eV/Å given the desired relaxation accuracy above. Therefore, add the following line, too:

```
sc_accuracy_forces 1E-3
```

During a structure optimization, we like only parts of the structure to relax and the rest kept at a fixed position. This is done by the keyword `constrain_relaxation`, which fixes the position of the last specified atom in `geometry.in`. In the FHI-aims manual, you find different options, but for constraining all coordinates of an atom, we use the flag `.true.`. In the `geometry.in` file, please write the following line right under the atom which you want keep fixed:[3]

```
constrain_relaxation .true.
```

An example excerpt from `geometry.in` reads like this:

```
atom -1.2063524529754976  0.0000000000000000  -0.8530200000000001  H
  constrain_relaxation .true.
atom  1.2063524529754976  0.0000000000000000  -0.8530200000000001  H
  constrain_relaxation .true.
atom ...
```

[3] You can achieve the same result by replacing in `write-geom.py` the expression `output_atom(...)` by `output_constrained_atom(...)`.

As a next step, the two top Si atoms must be perturbed in the xy-plane. The perturbation is best chosen in the order of 0.5 Å. This ensures that the initial geometry is sufficiently far away from the symmetry induced saddle point of the ideal geometry. Also, make sure to clearly break the mirror symmetry among these two silicon atoms, e.g., by moving down one of the two atoms by several tenths of an Å. A possible choice for the top silicon atoms is:

```
atom 2.4148451634531707  0.0000000000000000  3.262  Si
atom 5.2445354903595121  0.0000000000000000  4.062  Si
```

Here, the two top Si atoms have been moved in the x-direction by ±0.5 Å, and one Si atom has been pushed down by 0.8 Å.

The structure optimization with this starting geometry will take about 10 iterations. Feel free to try out your own guess, depending on the starting guess, the structure optimization will take between 10 and 30 iterations and therefore 7 to 21 minutes. As in the previous part, you can have a look at geometries along the relaxation path with the script `$HANDSON/aimsfiles/utilities/create_geometry_zip.pl`. Unzip the resulting file `geometries.zip` if you want to see the individual geometries or, as before, use Jmol by typing `jmol -s geometries.spt`.

You will see that one dangling bond per top silicon atom is saturated by forming a dimer among them. Additionally, this dimer gets asymmetric as experimentally evidenced by Wolkow [5].

If you compare the band structure of the reconstructed and clean Si(001) surface around the Fermi level you find that by means of the asymmetric dimer the surface gets semiconducting and there is a small but clear gap between the valence and the conduction band.

Part III: Magnetism and collinear spin calculations

For a few years, there have been sketches of electronic devices (for instance, transistors) based on the electrons' spin instead of just their charge. The associated field is known as "spintronics" and producing workable devices along these lines would be really beneficial.

Now, nature seems to dictate that for semiconductor devices, semiconductor materials are a plus, and for spin-polarized electrons, ferromagnetic materials are not a bad idea either. It would be good if we had a material that is semiconducting and ferromagnetic at the same time, at room temperature, and that can be grown by conventional semiconductor technology. Then we could realize "spintronics".

One of the most popular candidate materials is $Ga_{1-x}Mn_xAs$, with $x \approx 2\% - 10\%$ in most experiments. This material can be understood as a solid solution. The basic lattice is GaAs (zincblende) but with a fraction of the Ga atoms replaced by Mn.

In this part of the tutorial, you will learn the basic concepts of collinear spin calculations on the basis of GaMnAs. Please use the settings given in Fig. 9 for part III of this tutorial.

```
# Physical settings
  xc                  pw-lda
  spin                collinear
  relativistic        atomic_zora scalar

# SCF settings
  sc_accuracy_rho     1E-4
  sc_accuracy_eev     1E-2
  sc_accuracy_etot    1E-5
  sc_iter_limit       40

# k-grid settings  (to be adjusted)
  k_grid  nkx nky nkz
```

Figure 9: Default physical and computational settings for `control.in` for part III. This file can be found in `skel/problem_8/control_part3.in`.

Problem VIII: Magnetic $Ga_3Mn_1As_4$

Creation and visualization of `geometry.in`

- Create a `geometry.in` for $Ga_3Mn_1As_4$ in the zincblende structure and give Mn an initial magnetic moment of 4.
- Visualize the structure.

We do a supercell calculation, i.e., we assume a particular (and simple) supercell geometry with Ga and As atoms distributed on a lattice. To create a manageable structure, we choose the conventional *fcc* cell formed by four Ga atoms and four As atoms and replace one of them (e.g., the corner) by Mn. This gives us a total composition of $Ga_3Mn_1As_4$, eight atoms. A schematic picture of the structure can be seen in Fig. 10.

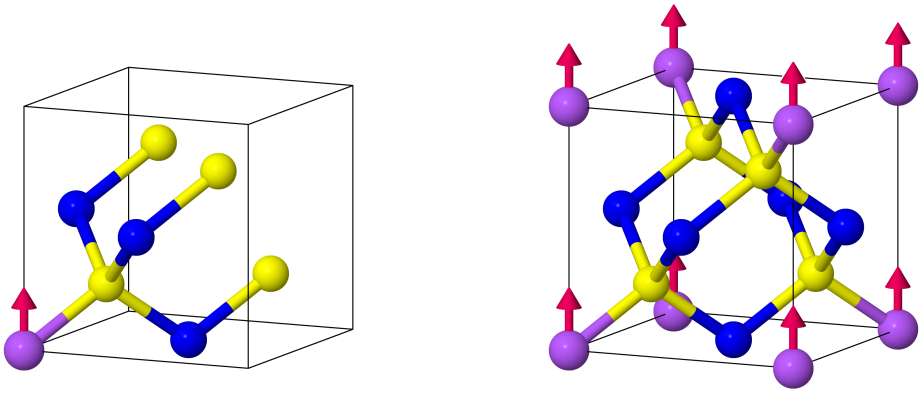


Figure 10: The cubic $Ga_3Mn_1As_4$ cell with 8 atoms. One Ga atom (blue) is replaced by a Mn atom (purple). The As atoms are drawn in yellow. The left figure shows only the 8 atoms that are unique to the cubic cell. The right figure adds the other atoms on the corners and faces only for illustration.

In real life, we would now predict equilibrium geometries (lattice parameters, local atomic positions, etc.) directly from first principles. Since we have limited time, we will instead take lattice parameters from experiment and keep all atoms fixed at their ideal lattice positions.

In a key paper on GaMnAs from Ohno *et al.* [6], the lattice parameter for GaAs is given as $a_{\mathrm{GaAs}} = 5.66\,\text{Å}$ ($x = 0$) and that for *hypothetical* zincblende MnAs $a_{\mathrm{MnAs}} = 5.98\,\text{Å}$ ($x = 1$). The change of the lattice parameter with the solute concentration x is often approximated by Vegard's law. In our case, it relates the lattice parameter of $Ga_{1-x}Mn_xAs$ to the atomic concentration of Mn x in the following way

$$a_{\mathrm{GaMnAs}} = x \cdot a_{\mathrm{MnAs}} + (1 - x) \cdot a_{\mathrm{GaAs}}. \tag{8}$$

Often, Vegard's law is not a great approximation, but in this case and for small concentrations, experiments seem to indicate that it holds, see Fig. 11.

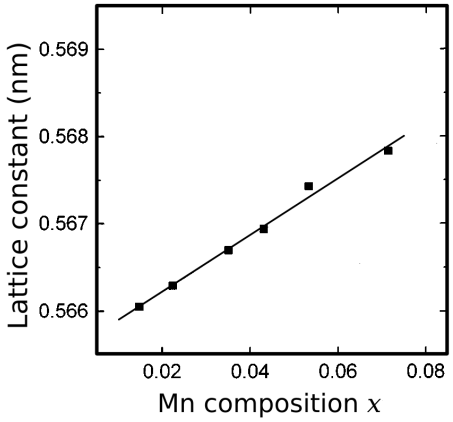


Figure 11: Vegard's law for $Ga_{1-x}Mn_xAs$: The lattice constant a is plotted versus Mn composition x in $Ga_{1-x}Mn_xAs$ at room temperature. Plot taken from [6].

Use Vegard's law (Eq. (8)) to estimate the lattice constant of $Ga_3Mn_1As_4$. (This compound does not have a small manganese concentration, but for sake of simplicity, we use Vegard's law nevertheless.) Build then the appropriate `geometry.in` for the zincblende structure in a cubic unit cell (see Appendix I for the structure information). Remember that you can use the `atom_frac` keyword to specify atomic positions in units of the lattice vectors.

To give the Mn atom an initial magnetic moment, add the following line

```
initial_moment 4
```

immediately after the specification of the Mn atom position.

DOS and projected DOS calculation

- Prepare a `control.in` with settings given below and calculate the DOS as well as projected DOS for this system.
- Plot the total DOS for spin up and down, and the Mn d orbital DOS for spin up and down.

[Estimated total CPU time: 7 min]

Please use the settings given in Fig. 9 and be sure to use "`spin collinear`" in your `control.in`. Use a rather sparse k-grid of 6×6×6 (see Fig. 12) for your calculation and the "light" species defaults for Ga, Mn, As which are located in the directory `$HANDSON/aimsfiles/species_defaults/light/`.

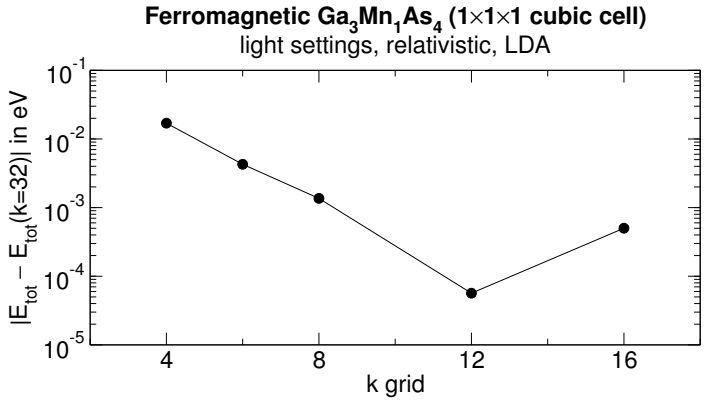


Figure 12: Convergence test for the k-grid of ferromagnetic $Ga_3Mn_1As_4$.

Similar to previous problems, the DOS as well as the projected DOS can be calculated by adding the following lines to your `control.in`:

```
output dos               -20. 0. 200  0.1
output species_proj_dos -20. 0. 200  0.1
dos_kgrid_factors 5 5 5
```

The "projected density of state" is the fraction of the DOS that is attributable to the atomic orbitals of any given atom. The "projected density of states" onto a given atom is not a uniquely defined thing. We here require a concept called "atoms in molecules", and if there is more than one atom, there is *no unique way* to decide whether a given part of the electron density came from the angular momentum channels of one atom or of another.

We here use a concept called a "Mulliken analysis", which simply takes the Kohn-Sham levels and distributes the electrons in each state (at single-particle energy ϵ) into the atoms and angular momentum channels where the associated basis functions came from – a projection. Since basis functions overlap between different atoms, electrons are "evenly assigned" between them. This way of classifying is anything but unique and gives increasingly bad results as the basis set increases in size (because then, electrons on one atom can easily be represented by the basis functions coming from another atom).

With our settings and basis size, the concept still works well on a qualitative level, but again, different schemes could be found.

In any event, based on the keywords above, the code produces files called:

- `KS_DOS_total.dat`
- `Mn_l_proj_dos_spin_up.dat`
- `Mn_l_proj_dos_spin_down.dat`

The total DOS contains information for both spin channels. For the Mn files, this is the occupation of angular momentum channels on the atom for the up and down channels, separately. You are interested in the d channels ($l = 2$), which is the fifth column. Read the header of either file if you are unsure.

You can use the script `aimsplot_Mn_d_orbitals.py` provided in the directory `skel/problem_8/` to plot all these densities. This script is based on the `aimsplot.py` script you used before. It has been modified to plot only the relevant DOS for this exercise. But you can adjust this script if you are interested. Just take a look inside it. The plot shows you which part of the Kohn-Sham density of states is derived from which atomic levels, approximately.

We also note that the `output species_proj_dos [...]` keywords already sum up all the contributions from every atom of a given element, so there is only one file for Ga instead of 3, etc. This is ok as we do not need to see 3 Ga atoms separately here, anyway. But you could if you wanted to.

Electronic & spin configuration with Mulliken analysis

- Find out the electronic configuration a neutral Mn atom and use Hund's rule to predict how the valence electrons are distributed. What magnetic moment do you expect?
- Perform a calculation for the neutral Mn atom together with a Mulliken analysis. Does the calculation give the magnetic moment you expected?
- Compare your results to the ones in the periodic case calculated before.

[Estimated total CPU time: <1 min]

Since we are only interested in a qualitative answer for the neutral Mn atom, you can use the same `control.in` that you used in the last task. Just be sure to remove or comment out all lines that are specific for a periodic calculation, namely the k-grid and the DOS keywords:

```
#k_grid  6 6 6
#output dos              -20.  0.  200  0.1
#output species_proj_dos -20.  0.  200  0.1
#dos_kgrid_factors 5 5 5
```

To request a Mulliken analysis, add the following line to your `control.in`:

```
output mulliken
```

Also, be sure to initialize the Mn atom with a magnetic moment in the `geometry.in` as before. As a value, you can take the magnetic moment you expect according to Hund's rule.

You can find the results of the Mulliken analysis in your output file. Look for the following section:

```
Performing Mulliken charge analysis on all atoms.
Full analysis will be written to separate file 'Mulliken.out'.
Summary of the per-atom charge analysis:
|
|  atom  electrons     charge        l=0        l=1        l=2        l=3
|     1  25.000000   0.000000   7.999856  12.000000   5.000144   0.000000
|
| Total  25.000000   0.000000

Summary of the per-atom spin analysis:
|
|  atom   spin      l=0         l=1         l=2         l=3
|     1    5.000000    0.000144    0.000000    4.999856    0.000000
|
| Total    5.000000
```

This gives you integrated values for the spin and angular momentum occupations of the Mn atom. A detailed list for every single Kohn-Sham level and spin channel is written to the file `Mulliken.out`. As you can see, we have 5 electrons in the d state ($l = 2$), which agrees with the electronic configuration of Mn ($[\mathrm{Ar}]4\mathrm{s}^2 3\mathrm{d}^5$). According to Hund's rule, each of these 5 d electrons is filled into one of the 5 d states in a spin up state, which results in a magnetic moment of 5. We get the same result with our Mulliken analysis. You can see that a Mulliken analysis can help us to get a qualitative picture of the electronic configuration of an atom. However, you should not rely blindly on the results of such an analysis since it is not a unique method as described above.

The summary of the Mulliken analysis for the previous $Ga_3Mn_1As_4$ calculation (done because of the projected DOS) yields a total magnetic moment of 4 for LDA[4] which is one less than for the free atom. In the crystal, some of the Mn d orbitals will hybridize with the As sp orbitals to perform the bonding, the others will provide the magnetic moment.

[4] For HSE06 (a hybrid functional), the magnetic moment will increase to a value of about 4.7.

Problem IX: Ferromagnetic and anti-ferromagnetic $Ga_{0.75}Mn_{0.25}As$

Creation of anti-ferromagnetic (AFM) and ferromagnetic (FM) structure

- Build a supercell with the same stoichiometry as in the previous problem (25% Mn) that supports an anti-ferromagnetic interaction in the (100) direction.

In an anti-ferromagnetic structure, there are atoms with a non-vanishing magnetic moment, but the overall magnetic moment of the system is zero. In the simplest case, the magnetic moments are oriented opposite to each other.

Hence, we need at least two Mn atoms in our supercell for an AFM state. Since we keep the same stoichiometry as before, this results in a supercell containing 16 atoms. We can create such a supercell by taking the primitive unit cell of the zincblende structure (see Appendix I) and double it in every direction. The primitive unit cell has 2 atoms, the resulting 2×2×2 supercell has then 16 atoms as desired. Both unit cells are depicted in Fig. 13.

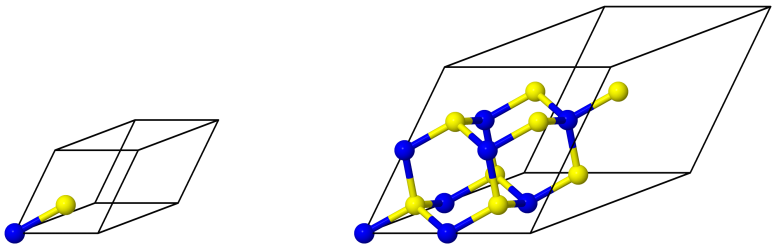


Figure 13: The left figure shows the primitive unit cell of GaAs in the zincblende structure. Doubling this cell in every direction gives the figure on the right – a 2×2×2 supercell that has 16 atoms in total.

As a next step, we need to substitute the Ga atom in the corner and the corresponding Ga atom in the (100) direction with Mn atoms. The Ga atom in the (100) direction has the coordinates $(a, 0, 0)$ with a the lattice constant. Unfortunately, this atom does not lie inside our 2×2×2 supercell. But since we have a periodic crystal, we can find an equivalent atom in our supercell. By adding the lattice vector $(0, a, a)$ of our supercell to the above Ga atom position, we get the coordinates (a, a, a). This equivalent position lies inside our supercell and corresponds to the fractional coordinates $(1/2, 1/2, 1/2)$. Therefore, replacing the Ga atom with this coordinates with a Mn atom results in a magnetic interaction in (100) direction.

For the FM case, we give both Mn atoms "`initial_moment 4`", for the AFM case, one gets "`initial_moment 4`" and the other one "`initial_moment -4`". This will ensure that the final spin state (after self-consistency is reached) is either FM or AFM.

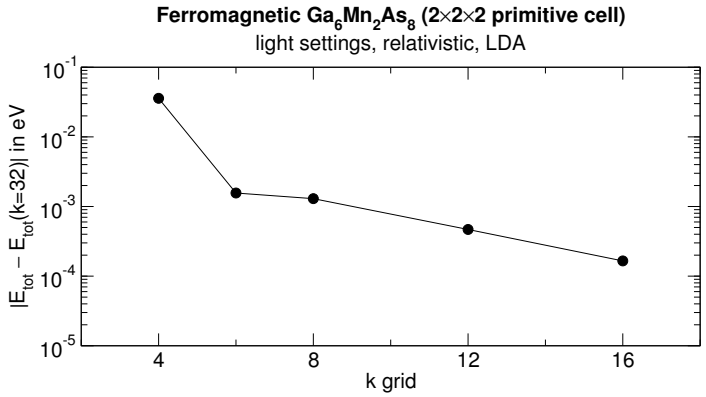


Figure 14: Convergence test for the k-grid of ferromagnetic $Ga_6Mn_2As_8$.

If you struggle to create the appropriate 2×2×2 supercell, the 16 atom supercell of pure GaAs is provided in the file `geometry_16_atoms_GaAs.in` in the directory `skel/problem_9/01_structure_creation/`. The lattice constant is already the right one for $Ga_{0.75}Mn_{0.25}As$. You just have to replace the correct Ga atoms with Mn atoms and assign them the right initial magnetic moment.

For a clearer visualization of the AFM structure, you can *temporarily* rename one of the two Mn atoms in the `geometry.in` to get them drawn in different colors in a visualization program.

Total energy calculation

- Prepare a `control.in` with a 6×6×6 k-grid.
- Start a total energy calculation for your AFM and FM structure.

[Estimated total CPU time: 5+5 min]

This system is tricky because it has states at the Fermi level. Therefore, we need a good k-grid. An acceptable compromise is a 6×6×6 k-grid (see Fig. 14). Now, you can start the calculations for the AFM and FM structure.

Please read on while the calculations are running.

Spin interaction parameter J

- Derive a formula for the spin interaction parameter J from the total energy difference between the AFM and FM state.

It is important to note that in spin-polarized density functional theory as we use it here, there are really only two spin directions, up or down. We can thus calculate configurations that are purely ferromagnetic (all spins point up), purely anti-ferromagnetic (equally many up or down spins on the Mn atoms), or anything in between.

As a result, we can compute for a particular placement of Mn atoms on the lattice different total energies E_{FM} and E_{AFM} as you are doing right now. Since there are only up- and down-spins, the appropriate model to represent our energetics is an Ising-type model. We can postulate:

$$E(\{S_i\}) = -J \sum_{\langle i,j \rangle} S_i \cdot S_j, \tag{9}$$

where we can formally choose $S_i = \pm 1$.

If we let the sum in Eq. (9) run over all possible inequivalent types of pairs, triplets, quadruplets, etc. formed by the Mn atoms on the lattice, with different interaction parameters J_f for each such inequivalent "figure" f, this mapping on an Ising Hamiltonian would even be exact, known as a cluster expansion. The nearest-neighbor form above is just a common truncated interaction Hamiltonian used in the literature.

Note that the spin on the Mn atoms is due to the spin splitting of the d levels, which is much larger than typical experimental temperatures. Hence, the moments located on each Mn atom themselves do not vanish with temperature, only their orientations can change. Our simple Ising model above will thus remain valid even at finite T.

You can then derive an expression for the interaction parameter J using Eq. (9) and the total energy difference $E_{\mathrm{AFM}} - E_{\mathrm{FM}}$ which you are currently calculating. Be sure to consider how many nearest-neighbors are present in your geometry.

Bonus: Dilute magnetic GaMnAs

Bonus: You can do this subtask if you are finished with the derivations from above and your calculations are still running.

- Use the precomputed results for a 64-atom supercell and investigate if the results for the smaller cell prevail.

The precomputed data for the 64-atom cell for the LDA and HSE06 functional can be found in the directory `skel/problem_9/03_dilute_magnetic_GaMnAs/`.

The 8-atom cell from the last problem is not really dilute. Therefore, we have doubled the unit cell size in all directions and computed the properties of a more isolated Mn atom in a 64 atom cell ($Ga_{31/32}Mn_{1/32}As$). Since the computation would be too time-consuming, just look at the results and plot the various densities of states as in the previous problem (total DOS and Mn d orbital DOS) with the script `aimsplot_Mn_d_orbitals.py`.

LDA results are good for total energies, but as you might know, the energy levels are sometimes not particularly good. This can be seen by going to a more sophisticated functional that includes some screened exact exchange in its single-particle equations.

Especially the HSE06 functional is interesting. Since there is actually a reasonable band gap in this functional, you can take the plot of the DOS and see where the top of the total spin-up valence band is. You will see that it contains empty states – holes. If you zoom in at the Fermi energy, you will discover that there are filled Mn d states. In essence, the Mn atom fills up one more of its d orbitals than it has electrons for, and it gets the missing electrons from the top of the valence band. Thus, holes are created.

These holes are what mediates the ferromagnetic interaction between the (otherwise isolated) Mn atoms. The holes are extended, in contrast to the d states of a Mn atom which are localized.

Discussion of results

- Calculate the interaction parameter J.
- Calculate the Curie temperature T_C.

In our 16 atom $Ga_{0.75}Mn_{0.25}As$ geometry, every Mn atom has 6 nearest-neighbors Mn atoms. These 6 inequivalent pairs interact all either anti-ferromagnetic or ferromagnetic depending on the initial magnetic moments.

In the FM case, we get a total magnetic moment (from the line in the FHI-aims output containing `N = N_up - N_down`) of about 8.0. A Mulliken analysis would show that each Mn atom has a moment of +3.81, a Hirshfeld analysis yields +3.75.

In the AFM case, the total magnetic moment vanishes, so you cannot directly see whether there are individual magnetic moments present. However, there is an additional output quantity showing you that non-zero individual moments are present, namely `<|rho_up-rho_dn|>`, which is defined as

$$\big\langle |n_\uparrow - n_\downarrow| \big\rangle = \int \mathrm{d}^3 r \, |n_\uparrow(\boldsymbol{r}) - n_\downarrow(\boldsymbol{r})|. \tag{10}$$

This quantity has a value of about 8.5, which would translate to a magnetic moment of ± 4.25 for the Mn atoms if only the Mn atoms had magnetic moments. A Mulliken analysis would show that the Mn atoms have a moment of ± 3.80, a Hirshfeld analysis yields ± 3.74.

We have 6 nearest-neighbors Mn atom pairs, therefore, we get according to Eq. (9)

$$E_\mathrm{AFM} = 6J \quad \text{and} \quad E_\mathrm{FM} = -6J \tag{11}$$

since $S_i \cdot S_j = -1$ for the AFM case and $+1$ for the FM case. Altogether, we get the following formula for the exchange parameter J

$$J = \frac{E_\mathrm{AFM} - E_\mathrm{FM}}{12}. \tag{12}$$

With the data of your total energy calculations (and taking the "`Total energy uncorrected`"), you should get $J \approx 0.5\,\mathrm{meV}$.

For a simple cubic lattice of magnetic moments and a classical nearest neighbor Ising Hamiltonian (Eq. (9)), Monte Carlo results give for the Curie temperature T_C [7]

$$k_B T_C = J/0.2217. \tag{13}$$

With the above J, we get a T_C of about 20 K for our ferromagnetic $Ga_{0.75}Mn_{0.25}As$ system, which is too low to be of practical importance.

In the previous exercises, you looked at a relatively large concentration of Mn atoms where each atom has multiple nearest neighbors. Of course, it would be better to try to determine J by looking at more dilute structures where we have only isolated Mn atom pairs or at least only connected in one direction.

Moreover, we could have considered other arrangements of Mn atoms in the lattice. Indeed, the FM coupling between Mn atoms is much stronger along the (110) or (220) directions than along the here investigated (100) direction. Thus, we could possibly generate a much higher T_C if we took an arrangement of Mn atoms with (110) or (220) oriented connection vectors. If our theory was exactly right, we could give that structure to an experimentalist and see if it can be actually realized and used for "spintronics".

Appendix I: Structure Information

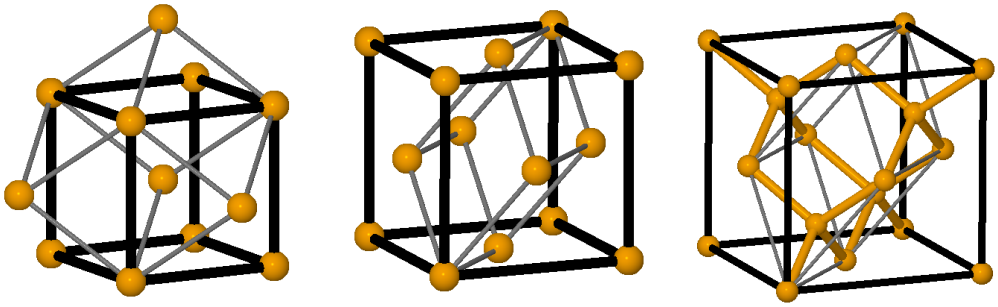


Figure 15: Cubic (black) and primitive (grey) unitcell for bcc, fcc and diamond Si (from left to right).

Crystal structures	atomic coordinates	lattice vectors
fcc	0 0 0	$\begin{matrix} 0 & a/2 & a/2 \\ a/2 & 0 & a/2 \\ a/2 & a/2 & 0 \end{matrix}$
diamond	$\begin{matrix} 0 & 0 & 0 \\ a/4 & a/4 & a/4 \end{matrix}$	$\begin{matrix} 0 & a/2 & a/2 \\ a/2 & 0 & a/2 \\ a/2 & a/2 & 0 \end{matrix}$
bcc	0 0 0	$\begin{matrix} -a/2 & a/2 & a/2 \\ a/2 & -a/2 & a/2 \\ a/2 & a/2 & -a/2 \end{matrix}$
zincblende with atom species A and B (primitive)	A 0 0 0 B $a/4$ $a/4$ $a/4$	$\begin{matrix} 0 & a/2 & a/2 \\ a/2 & 0 & a/2 \\ a/2 & a/2 & 0 \end{matrix}$
zincblende with atom species A and B (cubic)	A 0 0 0 B $a/4$ $a/4$ $a/4$ A $a/2$ $a/2$ 0 B $3a/4$ $3a/4$ $a/4$ A $a/2$ 0 $a/2$ B $3a/4$ $a/4$ $3a/4$ A 0 $a/2$ $a/2$ B $a/4$ $3a/4$ $3a/4$	$\begin{matrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{matrix}$

Table 1: Solids: Atomic coordinates and lattice vectors for different crystal structures.

diamond(001)					
atomic coordinates			lattice vectors		
0	0	0	$a/\sqrt{2}$	0	0
$a/2\sqrt{2}$	0	$a/4$	0	$a/\sqrt{2}$	0
$a/2\sqrt{2}$	$a/2\sqrt{2}$	$a/2$	0	0	L
0	$a/2\sqrt{2}$	$3a/4$			

Table 2: Surfaces: The atomic coordinates of an ideal diamond(001) surface. *Note:* a is the bulk lattice constant and L is the total slab thickness ($L = a + L_{\mathrm{vac}}$ with the vacuum size L_{vac}).

Appendix II: High symmetry k-points

fcc	x_1	x_2	x_3
L	0.5	0.5	0.5
Γ	0	0	0
X	0	0.5	0.5
W	0.25	0.5	0.75
K	0.375	0.375	0.75

fcc(001)	x_1	x_2	x_3
$\bar{\Gamma}$	0.0	0.0	0.0
$\bar{J}$	0.5	0.0	0.0
$\bar{K}$	0.5	0.5	0.0

Table 3: High symmetry points for *fcc*/diamond bulk and (001)surface structures given in units of reciprocal lattice vectors ($\boldsymbol{k} = x_1\boldsymbol{b}_1 + x_2\boldsymbol{b}_2 + x_3\boldsymbol{b}_3$).

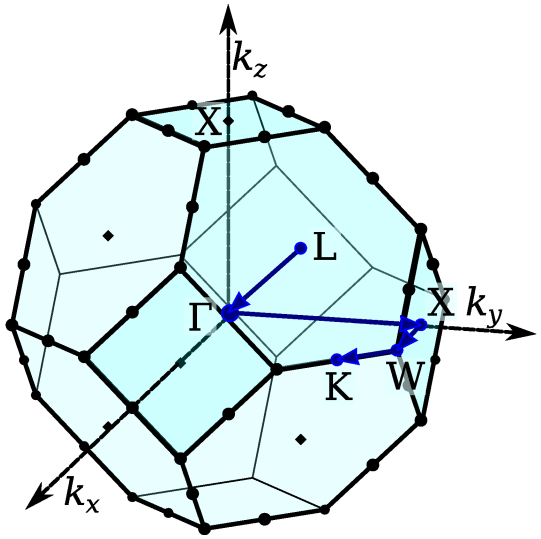


Figure 16: Brillouin zone and high symmetry points for *fcc*/diamond structure.

Appendix III: Si(001) geometry generation python script

```
#!/usr/bin/python

import sys
from math import sqrt

def output_lattice_vector(x, y, z):
    print "lattice_vector  %.16f  %.16f  %.16f" % (x, y, z)
def output_atom(x, y, z, name):
    print "atom %.16f  %.16f  %.16f  %s" % (x, y, z, name)
def output_constrained_atom(x, y, z, name):
    print "atom %.16f  %.16f  %.16f  %s" % (x, y, z, name)
    print "  constrain_relaxation .true."

A = 5.416             # Lattice constant
L_vac = 30.           # Vacuum
A_1x1 = A/sqrt(2.)    # 1x1 surface periodicity
n_layer = 4           # Number of layers in z-direction
Z = A/4.              # Layer separation in z-direction
C = 0.5 * A_1x1       # Row separation in x- and y-direction

# H-saturation is put at this fraction of where the next
# Si atom would have been.
frac_H = 0.63

# (2x1) reconstructed lattice:
output_lattice_vector(2*A_1x1, 0.,    0.)
output_lattice_vector(0.,      A_1x1, 0.)
output_lattice_vector(0.,      0.,    n_layer*Z+L_vac)
print

# Hydrogen saturation
# The next Si would have been at (+/-C, 0., -Z).
output_atom(   -frac_H*C, 0., -frac_H*Z, "H")
output_atom(   +frac_H*C, 0., -frac_H*Z, "H")
# The next Si would have been at (2*C+/-C, 0., -Z).
output_atom(2*C-frac_H*C, 0., -frac_H*Z, "H")
output_atom(2*C+frac_H*C, 0., -frac_H*Z, "H")
# Bottom Si layer
output_atom(0*C, 0., 0*Z, "Si")
output_atom(2*C, 0., 0*Z, "Si")
# Other Si layers
output_atom(0*C, C,  1*Z, "Si")
output_atom(2*C, C,  1*Z, "Si")
output_atom(1*C, C,  2*Z, "Si")
output_atom(3*C, C,  2*Z, "Si")
output_atom(1*C, 0., 3*Z, "Si")
output_atom(3*C, 0., 3*Z, "Si")
```

Figure 17: The python script used in part II (`skel/problem_6/01_ideal_2x1_4_layers/write-geom.py`). The script creates a geometry output (which can be redirected to a file) for the ideal hydrogen saturated 2×1 Si(001) surface with 4 layers.

Acknowledgments

We would like to thank Lydia Nemec for helpful discussions and the testers of this tutorial for their time and feedback.

References

[1] M. T. Yin and M. L. Cohen, *Microscopic Theory of the Phase Transformation and Lattice Dynamics of Si*, Phys. Rev. Lett. **45**, 1004 (1980).

[2] C. Kittel, *Introduction to Solid State Physics* (John Wiley & Sons Inc, 1986), 6th edition.

[3] F. D. Murnaghan, *The compressibility of media under extreme pressures*, Proc. Natl. Acad. Sci. **30**, 244 (1944).

[4] F. Birch, *Finite Elastic Strain of Cubic Crystals*, Phys. Rev. **71**, 809 (1947).

[5] R. A. Wolkow, *Direct observation of an increase in buckled dimers on Si(001) at low temperature*, Phys. Rev. Lett. **68**, 2636 (1992).

[6] H. Ohno, A. Shen, F. Matsukura, A. Oiwa, A. Endo, S. Katsumoto and Y. Iye, *(Ga,Mn)As: A new diluted magnetic semiconductor based on GaAs*, Appl. Phys. Lett. **69**, 363 (1996).

[7] A. M. Ferrenberg and D. P. Landau, *Critical behavior of the three-dimensional Ising model: A high-resolution Monte Carlo study*, Phys. Rev. B **44**, 5081 (1991).