Theoretical Material Science:
Electronic Structure Theory at the Computer

Prepared by Hagen-Henrik Kowalski,
based on an exercise by Christian Carbogno
Berlin, June 2017

Some rules on expected documentation from this exercise.

The computational exercises are intended as "hands-on" experience with actual, numerical electronic structure theory. Our main goal is to fill some of the basic concepts with life for real systems.

If you are fast, you may finish the exercise below in the actual exercise class. If not, we ask you to finish as much as you can, and then come back at another point, follow the script, and finish the exercise.

You can perform the computations and hand in your solutions to the exercise in pairs of two. What is expected is a record of the basic data that we ask for (e.g., in table form), rough answers to the questions asked (answers can be short, but should be there, and should indicate that you understood the meaning of your data), and plots, where required. You can just qualitatively draw the plots by hand in your homework or print them on paper or screenshot them and hand-in your homework electronically via email as PDF.

Please hand in all your solutions as usual at the beginning of the exercise in the following week.

Contents

1 Important Notes

- The module "Thermal_expansion.py" contained a minor bug which has been corrected. The patched version can be downloaded from the same link and is also included in the new version of the virtual machine.
- This exercise should be performed using exclusively the generalized gradient approximation (GGA) to density-functional theory. To be precise, please **choose** the **PBEsol** functional. Do not choose the local-density approximation (LDA) by accident.
- Throughout this exercise, use only "light" settings for the species defaults of FHI-aims. This will suffice to demonstrate the principle. In a complete scientific project, one would want to verify the key results with converged "tight" settings, but this is not required here due to the time involved.
- Please closely inspect the example for the control.in file given in Sec. 6.1. It contains several new keywords. You may use it as a template.

2 Background

In this exercise, we will use the structure predictions of exercises 14 and 15 as a basis to investigate the vibrational properties of diamond silicon.

We will use the forces calculated in Kohn-Sham density-functional theory (DFT-GGA) to approximate the potential-energy surface (PES) in a second order Taylor expansion around the equilibrium volume,

$$V^{\mathrm{BO}}(\{\mathbf{R}\}) \approx V^{\mathrm{BO}}(\mathbf{R}_0) + \frac{1}{2} \sum_{\alpha\beta,IJ} \underbrace{\frac{\partial^2 V^{\mathrm{BO}}(\mathbf{R}_0)}{\partial R_I^\alpha \partial R_J^\beta}}_{\Phi_{IJ}^{\alpha\beta}} s_{\alpha,I} s_{\beta,J} \ . \tag{1}$$

Here, the $\Phi_{IJ}^{\alpha\beta}$ are the force constants (or Hessian), $\mathbf{s}_I$ are the displacement vectors, and V^{BO} is the Born-Oppenheimer surface.
We already know from the lecture that, strictly, this "approximation" will be limited in reach as it does not account for anharmonic effects. More accurate methods exist, such as molecular dynamics, but these are usually exceedingly expensive. Accordingly, the methods used in this exercise are frequently employed in the solid state community to date.

As a reminder, we give the five **k**-points that are most relevant for the present exercise below :

- Γ: Located at (0,0,0) in units of the reciprocal lattice.
- L: Located at (0.5,0.5,0.5) in units of the reciprocal lattice of the primitive fcc cell.
- X: Located at (0.0,0.5,0.5) in units of the reciprocal lattice of the primitive fcc cell.
- W: Located at (0.25,0.5,0.75) in units of the reciprocal lattice of the primitive fcc cell.
- K: Located at (0.375,0.375,0.75) in units of the reciprocal lattice of the primitive fcc cell.

3 Important: Necessary scripts and modules

For the exercises 22 and 23, we need a few additional scripts which can be downloaded from the link
https://thcloud.rz-berlin.mpg.de/s/JjHmjwTgvz3910H **modified phonopy for exercise 22**
https://thcloud.rz-berlin.mpg.de/s/SToKOUwH3VPN0gU **modules and script for exercise 23**

Alternatively, you can just re-download and re-install the updated virtual machine instance that already contains these packages in form of tar files from https://thcloud.rz-berlin.mpg.de/s/kElVkeik4aSfMEQ
.

3.1 Classical calculations using phonopy

To complete exercise 22, you need a modified version of the phonopy package. Be careful when using this package. To avoid any conflict between the modified and usual phonopy installed on your machine, it might be useful to name the folder containing the modified phonopy version, e. g. **classical_phonopy**. Accordingly, when you want to use the new installation, just change the path and the python path to:

```
export PATH=/home/theory/classical_phonopy/bin:$PATH
export PYTHONPATH=/home/theory/classical_phonopy/lib/python
```

The installation works in exactly the same way as described in Sec. 4. **IMPORTANT**: when you use the normal phonopy, do not forget to change the paths back to the original ones described in Sec. 4.

3.2 Necessary scripts for exercise 23

The other scripts included in the download are required to perform exercise 23. These are python modules or scripts which require no installation. The file "Calculate.py" is a script that automatically calculates the static energy and the phonon free energy using the "Thermal_expansion.py" and "ASE" modules.
The script "Expansion.py" extracts the temperature dependent volume and thermal expansion. To make use of the scripts, safe them in a new folder, for instance "modules". Now change your python path to

```
export PYTHONPATH=/home/theory/phonopy/lib/python:
/home/theory/modules:/home/theory/modules/ase-3.12.0
```

4 Reminder: Phonopy in the Virtual Machine

To calculate phonons in the harmonic approximation, we will use the phonopy package but before you can get started some other packages have to be installed. Please install pip, h5py, and pyyaml by typing:

```
sudo apt install python-pip
pip install h5py
pip install pyyaml
```

Once these are installed, download phonopy from the link
https://thcloud.rz-berlin.mpg.de/s/6FEiHHM3ftK6F8B.
After downloading, unpack the file and name it simply `phonopy`. Before installing phonopy, you have to add a line to your python path in your bashrc file (this file can only be accessed from the home folder). This can be opened by typing:

```
gedit .bashrc
```

At the end of the bashrc file, please add the lines:

```
export PATH=/home/theory/phonopy/bin:$PATH
export PYTHONPATH=/home/theory/phonopy/lib/python
```

After saving and exiting the bashrc file do not forget to type:

```
source .bashrc
```

Next, change into the folder `phonopy` and install the package by typing:

```
python setup.py install --home=.
```

Now can run the program simply by typing phonopy-FHI-aims into the terminal.
Later in the exercise you will also need some pre-calculated simulations. They can be downloaded from the link
https://thcloud.rz-berlin.mpg.de/s/3PgxVnmhRMT2LqI.
These folders contain force evaluations with a number of atoms that would not be possible on a local machine let alone on a virtual one.

5 Reminder: FHI-aims in the Virtual Machine

As a quick reminder, here is (again) an overview of the most important pieces needed for FHI-aims:

- To run the code, create the necessary input files (`control.in` and `geometry.in`) in a working directory of your choice.
- The calling sequence for FHI-aims is:

```
aims | tee calculation.out
```

As before, although you should not need it, there is a complete manual (pdf) for the FHI-aims code located in
/home/theory/fhi-aims.160328_3/FHI-aims.pdf .

In this exercise sheet, we will use a *python* script to visualize and plot band structures and densities of states. This requires additional python libraries that you can install with the command:

```
sudo apt-get install python-matplotlib python-scipy
```

For this purpose, you will be requested to enter the password ("theory") and to acknowledge the installation of additional, required packages (just press "Y" when asked for it). If you encounter any problems, e.g. "E: Could not get lock /var/lib/dpkg/lock", reboot the virtual machine with the command: `sudo shutdown -r now`

After reboot, the commands

```
sudo dpkg --configure -a
sudo apt-get install python-matplotlib python-scipy
```

should do the trick. Alternatively, you can just re-download and re-install the updated virtual machine instance that already contains these packages from https://thcloud.rz-berlin.mpg.de/s/kElVkeik4aSfMEQ . Please note that deleting the old virtual machine will also delete the data that you created with it during the previous exercises.

6 Exercise 22: Everything moves

In the previous exercise, you calculated basic vibrational properties such as the phonon band structure in four different supercells (primitive, first cubic, second cubic, and fourth cubic). For the present exercise, you are supposed to calculate the quantum mechanical and classical phonon free energy and specific heat using the force evaluations in the second cubic supercell from exercise 21.

6.1 Calculating free energies and specific heats using FHI-aims and the phonopy package

To calculate the phonon free energy and specific heat, you only have to add the line

```
phonon free_energy 0 1010 1010 20
```

to the control.in file used for the force evaluation in the second cubic supercell and run phonopy by typing phonopy-FHI-aims into the terminal. This will calculate the phonon free-energy, the internal energy, the specific heat, and the entropy from 0 K to 1010 K for 1010 temperature points on a $20 \times 20 \times 20$ integration grid. An example for a complete control.in file is given below.

```
# Physical settings
   xc          pbesol

# SCF settings
   sc_accuracy_eev   1E-2
   sc_accuracy_rho   1E-4
   sc_accuracy_etot  1E-4
   sc_accuracy_forces  1E-4
   sc_iter_limit     40
# Mixer settings
   mixer pulay
   n_max_pulay 30
   charge_mix_param 0.05

# Occupation settings
   occupation_type gaussian 0.01

# k-grid settings
   k_grid   4 4 4
# Matrix for the construction of the supercell
   phonon supercell -2 2 2 2 -2 2 2 2 -2

# Phonon free energy
   phonon free_energy 0 1010 1010 20
```

The results will be stored in the file "phonopy-FHI-aims-free_energy.dat". This file contains five columns of data, the first being the temperature, the second the free energy, the third the internal energy, the fourth the specific heat, and the fifth the entropy. To plot the free energy or the specific heat as a function of the temperature using xmgrace you can use the command:

```
xmgrace -block filename -bxy column1:column2
```

7 Exercise 23: Quasi-Harmonic approximation

So far we, only investigated the vibrational properties of diamond silicon within the harmonic approximation, but despite its usefulness it has severe limitations. For instance, some physical phenomena such as thermal expansion cannot be described using the harmonic approximation. In this exercise, you will see how thermal expansion can be assessed using the quasi-harmonic approximation.This requires us to inspect how the phonons, i.e., the vibrational band structures and the associated free energies, change with the volume of the crystal. In a nutshell, we will thus repeat the exact same kind of calculations as performed in exercise 21 and 22, but now for different lattice constants. For your convenience, we have provided a script that prepares the required geometry.in and control.in files and runs phonopy and FHI-aims for you.
In the previous exercise sheets, you already learned how to determine the lattice constant of a crystal by finding the minimum of the total energy E_{DFT} by using the Birch-Murnaghan Equation-of-State. There is a caveat, though: In the canonical ensemble, the relevant thermodynamic potential that needs to be minimized is the free energy $F(T, V)$ and not the total energy E_{DFT}. The free energy of a solid is given by the DFT total energy (per unit cell) and the vibrational free energy, which is also calculated per unit cell:

$$F(V, T) = E_{\mathrm{DFT}}(V) + F^{\mathrm{vib}}(T, V) \ . \tag{2}$$

You already calculated the free energy at a given lattice constant in exercise 22. However it is not explicitly volume dependent in the form it is defined there. To account for the volume dependency, you no will have to calculate the free energy for a series of lattice constants so that we can point wise evaluate and minimize equation 2 using the Birch-Murnaghan Equation- of-States. This is what the scripts

```
Calculate.py and Expand.py
```

do. For each lattice constant a, they calculate the static DFT energy and the vibration free energy. If all paths are set properly you can run the first script simply by typing:

```
python Calculate.py
```

This will generate a folder "SiSi" in which all the generated data will be written. The static DFT energy is then stored in the folder "Murn_data" in a file named "Etot.dat" and the vibrational free-energies are saved in the folders that indicate the volume at which they were calculated both within the folder "SiSi". Assuming for example that the free-energy was calculated at a volume of 50Å it will be saved in the folder "Phonon50".
All phonon calculations will be carried out in the first cubic supercell. Consistency regarding the number of **k**-points is ensured by calculating the static energy also in the first cubic supercell.

To calculate the thermal expansion coefficient and the unitcell volume as a function of temperature you have to copy the script "Expand.py" into the folder "SiSi" and run it by typing

```
python Expand.py
```

It is important to keep this naming convention because the script Expand.py exactly searches for the necessary data in these folders.
This script exploits the Birch-Murnaghan Equation-of-State and determines the equilibrium volume in 5 K intervals up to a temperature of 800 K. From these volumes, the thermal expansion coefficient is determined via a finite difference method using:

$$\alpha(T) = \frac{1}{3V}\frac{\partial V}{\partial T} \approx \frac{1}{3V}\frac{V(T+\delta T) - V(T-\delta T)}{2\delta T}. \tag{3}$$

Finally, "Expansion.py" writes the results for the temperature dependent volume in the file "VolumesPhonon.dat" and the thermal expansion in the file "Thermal_expansionPhonon.dat"