User Manual: ACTE -Anisotropic Coefficients of Thermal Expansion

Nicholas A. Pike[1] *and Ole M. Løvvik*[1,2]
[1]***Centre for Materials Science and Nanotechnology***
University of Oslo
NO-0349, Oslo, Norway

[2]***SINTEF Materials Physics***
Forskningsveien 1
NO-0314 Oslo, Norway

Abstract

This algorithm and script provides a first-principle method to calculate the anisotropic coefficients of thermal expansion for any stable material. While the user input for each calculation is minimal, the user must modify a small portion of the script before use. Comparisons for between experimental results and test cases indicate the easy-of-use of this program and its accuracy over a wide range of materials.

Note for all users:

Before using this program, it is suggested that the user be familiar with the first-principles density functional theory package known as VASP [1, 2]. This script will automatically launch calculations using VASP to calculate both ground state and response function properties. Additionally, this script requires the use of the post-processing code known as TDEP [3, 4, 5] to calculate the temperature-dependent properties of our system and to generate canonical configurations allowing for the calculation of the temperature dependent force constants.

Contents

1 Basic Information

1.1 Author Information

Author: Primary development and design of this document, and the related script, come from Nicholas Pike. Ole M. Løvvik was instrumental in formulation of the original idea behind this script. Questions and comments can be addressed to both authors. Bugs and suggested additions should be addressed to Nicholas.

1.2 Email Address

Email: Questions and comments:
OleMartin.Lovvik@sintef.no

Questions, comments, code suggestions, and bugs:
Nicholas.pike@smn.uio.no

1.3 Citations

Citation: While I can not force you to cite my work, it is highly suggested! Please cite (Ref. [6]) the following reference when publishing any calculations using this script.

1. N. A. Pike and O. M. Lovvik "Calculation of the Anisotropic Coefficients of Thermal Expansion: A First-Principles Approach" Computational Materials Science, 167 (2019) 257-263.

2 Accessing and Downloading the Script

Code Access: This script, and related documentation, can be cloned using ssh via: ssh://git@git.code.sintef.no/asmod/acte.git or by cloning from https://github.com/Npikeulg/ACTE.git. Finally, one can also send an email to the developer who will happily share the code with you.

2.1 Acknowledgments

Acknowledgments: We would like to thank the Research Council of Norway through the Frinatek program for funding during development of this software. The computations for the examples given in this manual are performed on resources provided by UNINETT Sigma2 - the National Infrastructure for High-Performance Computing and Data Storage in Norway.

Of course, this script would not work with out the efforts of the many programmers and developers of the VASP and TDEP software packages. Thank you!

2.2 License Information

3 Introduction to Code

3.1 Multicore software

While this script itself does not require a multi-core infrastructure, it does require the use the VASP software package which uses density functional theory to calculate many ground state properties of a quantum system. This software works efficiently on a multi-core infrastructure and therefore, we require that this script be used only on a super-computing infrastructure.

Because of this requirement, there are several parts of the script that need to be modified before running on a supercomputer. These required modifications are all related to submission of jobs on the supercomputer and are outlined below.

3.2 User Modifications of the Script

Required Modifications

Before running this script, the user must modify the script to run on their supercomputer infrastructure. This requires modifications of the following sections of the code:

Batch Script Header

Submission of jobs to the supercomputer requires a specially formatted input file, known internally as a "batch.sh" file. This batch file requires a special header that provides important information to the supercomputer. While several different input parameters can be specified, the user should consult an example submission script from their supercomputer administrators to determine the minimum input parameters required.

VASP Parameters

Since we are undertaking first-principles calculations as part of this program, the user must specify a few input parameters used by VASP that related directly to convergence studies that should be performed before this script is executed. Therefore, modification of the following input parameters is recommended:

- energy cut-off (ecut): Controls the maximum kinetic energy of an electron orbital and effectively cuts off the orbital radius of the electron orbital.
- energy difference (ediff): Controls the convergence of the self-consistent calculations during all ground state VASP calculations.
- kpoint density (kdensity): Parameter indicating how well one samples momentum space and therefore how effective our calculation will be in sampling the interaction between atoms within our unit cell.

Of course, there are many other input parameters for a VASP calculation that are needed for the INCAR file. These input variables are automatically

TDEP Parameters As part of the Tdep calculations, the user can modify any of the following parameters:

- natom_ss: Number of atoms in the supercell calculations. Used during creation of the canonical configurations.
- rc_cut: Second order force constant cut off radius. Using a value of 100 defaults to the maximum radius between a pair of unique atoms.
- tmin: Temperature minimum for the thermal properties.
- tmax: Temperature maximum for the thermal properties.
- tsteps: Number of temperature steps for the calculation of the thermal properties.
- qgrid: Density of the momentum grid in q space for calculations of the thermal properties
- iter_type: Numerical integration method used to calculate the phonon density of states
- n_configs: Number of canonical configurations to generate
- t_configs: Fractional temperature of the canonical configurations. This value is multiplied by the calculated Debye temperature which is calculated by the script as outlined in section 3.5.5.

Additional information on the input parameters for a TDEP calculation can be found at:
http://ollehellman.github.io/

Source Files Paths to both the VASP and TDEP executable must also be specified. Note, the path of the TDEP executable should be pointed to "bin" which is automatically generated when compiling the TDEP software package.

3.3 File Directory Structure

Directory: While the script will generate many of the directories given below, the user must create, download, or generate all of the folders and files that are underlined before starting the calculation.

An example directory structure for the example given in this reference manual is given below.

```
... (root, home, etc)
 ├── pseudos
 │    └── Cu
 │         ├── POTCAR
 │         └── ...
 └── Copper
      ├── POSCAR
      ├── data_extraction
      ├── Relaxation
      ├── Elastic
      ├── free_energies
      └── lattice_*
           ├── configs
           │    └── conf00*
           ├── relaxation2
           └── moldyn
```

3.4 Code execution

Example command line execution of this code, and a brief explanation of the tag, are given below.

--help This tag prints the help menu of the program. This gives access to all of the available execution tags.

--usage Provides an example of how to use the program by providing a general example of how to execute the function of a tag.

--author Provides the author information about this software.

--version Provides a brief summary of the version information and recent updates to the software package.

--email Provides the email address of the primary author where bugs and suggestions can be sent.

--bug Provides an email address and suggestions for what information should be sent in a bug report.

--check_sys A simple script that checks if the file system and pseudopotential files are found on the supercomputer. This function will also check that both the VASP and TDEP executable paths are valid and that the necessary python modules can be loaded.

The following python modules are required by this script and should be installed using the command *pip install –user MODULE_NAME.*

- numpy
  - Necessary for numerical calculations and the construction of arrays and optimization routines
- subprocess
  - Necessary for the interface between python and BASH. This is used many times in the program to execute shell level commands.
- Mendeleev
  - Necessary for accurate values for atomic masses and other atomic properties.

--relaxation To start a calculation of the coefficients of thermal expansion, the user will place a POSCAR file in the correct directory. For example, in the case of copper given in section 4, the file and folder structure should be:

```
... (root, home, etc)
 |__ pseudos
 |    |__ Cu
 |__ Copper
      |__ POSCAR
```

When the user uses the --relaxation tag the program executes the following steps:

1. Creates two folders called “Relaxation” and “Elastic” in the copper folder.
2. Generates a batch.sh script used for execution of the VASP calculations within those folders.
3. Generates the VASP INCAR file for the relaxation and elastic tensor calculations.
4. Generates the KPOINTS file for the relaxation and elastic calcula-

tions.

5. Generates the POTCAR file for all calculations after reading the POSCAR file to determine the elements.
6. Copies the POSCAR file to the "Relaxation" folder.
7. Launches the relaxation calculation with VASP.
    (a) After the calculation is complete, the script will check if the VASP calculation completed successfully. If the calculation completes then the script will move on, if not, it will relaunch the relaxation up to three times.
    (b) After a successful run, the code will extract the structural information to a file called "data_extraction".
8. The script then launches the elastic tensor calculation after moving the necessary files (CONTCAR $\rightarrow$ POSCAR)
9. After the elastic calculation is done, the script extracts the calculated tensor information (Born effective charge tensor, elastic tensor, and dielectric tensor) to the file "data_extraction".
10. Finally, the program will execute the tag --build_cells to launch the construction and calculation of the configurations needed by TDEP.

The file structure after these calculations will be:

```
... (root, home, etc)
 ├── pseudos
 │    └── Cu
 └── Copper
      ├── POSCAR
      ├── data_extraction
      ├── Relaxation
      │    ├── OUTCAR
      │    ├── POSCAR
      │    ├── KPOINTS
      │    ├── POTCAR
      │    ├── INCAR
      │    └── ...
      └── Elastic
           ├── OUTCAR
           ├── POSCAR
           ├── KPOINTS
           ├── POTCAR
           ├── INCAR
           └── ...
```

The user should check the data_extraction file to make sure that all the results of the density functional theory and density functional perturbation theory calculations agree with experiment.

--build_cells The second step of this calculation is by far the most time consuming part of the calculation. After a successful calculation of the structural parameters and elastic tensor, the user should use this tag to launch the calculations that will be used to determine the free energy as a function of temperature and lattice parameter. To do this, the script executes the following steps:

1. The program reads in the relaxed POSCAR file to determine how many different volumes need to be generated to calculate the free energy. From the symmetry of the lattice parameters in the POSCAR file, the program will determine the number of volumes

to generate using both compression and expansion of the unit cell.

2. With the number of new volumes in memory, the program will generate new POSCAR files for each of the volumes.
3. The elastic tensor is read from the file “data_extraction” and used to generate the Debye temperature of the system. This Debye temperature is also written to the data_extraction file.
4. The Born effective charges and dielectric tensor are read from the data_extraction file and printed to a new file “infile.lotosplitting” which is used by the TDEP software package.
5. Two INCAR files are generated. The first is an INCAR file corresponding to relaxation of the unit cell after changes to the cell volume. This relaxation run allows for the movement of the internal positions of the atoms and no change in lattice parameters. The second INCAR file corresponds to a ground state calculation which is used to generate the forces and atomic positions for TDEP.
6. Generation of the KPOINTS file for all stages of this part of the calculation.
7. Generation of two batch.sh files, one for the first relaxation and canonical configuration step, the second for the ground state VASP calculations.
8. Creation of a folder for each of the different volumes and moving the necessary POSCAR, INCAR, POTCAR, and KPOINTS file to the correct folder.
9. Each volume is launched on a supercomputer in parallel.
   (a) The calculations will launch and run the relaxation of the unit cell again. If the relaxation is successful, the calculation moves the necessary files to the folder “moldyn” where the configurations are generated. If not, it aborts.
   (b) If successful, the calculation generates a supercell with the previously specified number of atoms, and creates a set of canonical configurations (as determined previously by the user).
   (c) Folders are created for each configuration, the necessary files are moved, and the calculations are once again launched in parallel for each configuration.

Additionally, one can change the default strain step used by the program. This might be needed for some materials and is thus a provided option. The default strain step is 1% and can be changed with

--strain Modifies the default strain used to generate the different cells used in the free energy calculation. Note, this flag takes a second argument (which is the strain step as a decimal. I.e 1% = 0.01).

Taking copper as an example. The script will determine that 6 volumes are needed, each of these volumes requires 12 configurations. Therefore, the script will launch 6*12= 72 calculations in total!

The file structure at this point should look similar to:

```
... (root, home, etc)
 ├── pseudos
 │    └── Cu
 └── Copper
      ├── POSCAR
      ├── data_extraction
      └── Relaxation
           └── ...
```

```
|
|__ Elastic
|   |__ ...
|__ lattice_0
|__ lattice_1
|__ ⋮
|__ lattice_5
    |__ relaxation2
    |   |__ ...
    |__ moldyn
    |   |__ ...
    |__ config
        |__ conf001
        |__ conf002
        |__ conf003
        |__ ⋮
        |__ conf012
```

--thermal_expansion After successful completion of all the TDEP configurations. The user will use this tag to calculate the free energy, and therefore many different thermal properties of their system. Note: this is the only part of the calculation that takes additional tags, as discussed below.

As part of this calculation, the following steps are executed by the code:

1. The script checks if the user included any additional flags when executing this command. If there are no flags, they are all set to their default values. However, the user can specify the following flags which may improve both the performance and results of this part of the calculation.

   Suggested use:
   --thermal_expansion --bounds --BEC --solver --polyfit 0 2

   --bounds Include bounds during the fitting of the energy vs volume and during the minimization of the free energy as a function of temperature.

   --solver Changes the default least squares solver to solver 2 within TDEP. This is necessary to use whenever the default solver (solver 1) does not find a stable solution. See documentation within TDEP for additional details.

   --BEC Requires that TDEP use the Born effective charge and dielectric tensor when fitting the Born-Oppenheimer energy surface to calculate the inter-atomic force constants. This flag is required whenever the material is semiconducting.

   --polyfit (lower upper print) Uses an eight order polynomial to fit the extracted temperature dependent lattice parameters over an interval from “lower” to “upper” which are multiples of the Debye temperature. The default values, used when lower and upper are not specified, are 0 and 2 times the Debye temperature. The default value of print is False, and in fact need not be included for proper code execution. If print is set to True, then the program will generate an additional file with the lattice parameters generated by the polynomial fit.

2. The script once again uses the original relaxed POSCAR file to determine the symmetry of the system and therefore how many volumes it should look for when calculating the free energy.

3. A folder is created to store the free energy files in a single directory. These files are given a unique name that corresponds to the lattice parameters, internal energy, and volume of the simulated cell.
4. Then, for each volume, the script generates a file to launch the calculation of several TDEP functions necessary to calculate the free energy of the system.
5. Once the free energy is calculated, the free energy file is moved to the recently created folder and the file name is stored in the data_extraction file.
6. Once all the free energy files are extracted, the main function of this code is executed.
   (a) The file data_extraction is read by the program and all information is stored in memory.
   (b) The free energy files for each volume are read in by the script and stored in memory. During this step, the code will check that the calculated free energy makes sense. If, for example, there are unstable phonon modes, TDEP sets the free energy to a fixed value. The script detects this value and relaunches the calculation by generating a new set of canonical configurations generated from the unstable modes. This almost always leads to stable phonon modes in the following calculation.
   (c) After reading in all the free energies, the program finds, for each temperature, the lattice parameters that minimize the free energy of the system using a 4^{th} order polynomial.
   (d) The energy vs volume are fit to the Birch equation of state at each temperature, thus allowing for us to extract the isothermal bulk modulus at each temperature (among other things).
   (e) The coefficients of thermal expansion are then calculated by taking the gradient of the lattice parameters as a function of temperature.
   (f) Next, the specific heat at constant pressure is calculated as a function of temperature.
   (g) Finally, the script will print several output files containing all the calculated and extracted values in this program. Each of these output files starts "out.".

--relauch_tdep Allows for the calculations in --build_cells to be relaunched without generation of new files.

--move_file Usage: --move_file FILENAME NEWLOCATION ORIGINAL

Moves the file called FILENAME to a NEWLOCATION. ORIGINAL is boolean (True or False) and tells the program to keep the original file or delete it.

--copy_file Usage: --copy_file FILENAME NEWLOCATION

Copies the file called FILENAME to a NEWLOCATION.

--make_KPOINTS Usage: --make_KPOINTS KDENSITY

Generates a vasp formatted KPOINTS file with a density specified by KDENSITY.

--launch_calc Usage: --launch_calc FILE FILELOCATION

Launches the the calculation using the FILE (generally batch.sh) on the supercomputer in the directory FILELOCATION.

--outcar	Reads the OUTCAR file generated by VASP and extracts the necessary information depending on what is already found in the data_extraction file.
--vasp_converge	Usage: --vasp_converge CALCTYPE Checks if the VASP calculation converged. The script checks the OUTCAR file for different items depending on CALCTYPE.
--make_folder	Usage: --make_folder FOLDER_NAME Creates a folder in the current directory with the name FOLDER_NAME.
--generate_batch	Usage: --generate_batch BATCH_TYPE FOLDER_LOCATION TAGS Creates a batch.sh file, specified by BATCH_TYPE, in the folder FOLDER_LOCATION. This file is first created and then made to be an executable file. Note that, in some instances, additional tags are required and can be specified by the user.

3.5 Mathematical Background

This script uses first-principle calculations to determine the temperature dependent lattice expansion within an anisotropic material. The mathematical underpinnings of density functional theory and density functional theory are outlined in many other sources. If the user is unfamiliar with density functional theory and density functional perturbation theory the user should consult the following references: [7, 8, 9, 10, 11] and maybe run a few calculations independent of this program as well. Experience helps.

The mathematical underpinnings of TDEP are outlined by Olle Hellman [3, 5, 4] and at http://ollehellman.github.io/. On the website are outlines and examples of the executable for TDEP, many of which are used here. This program is used rather heavily in this program and understanding the basics of this program are desirable before running this script.

For our purposes, the ACTE script calculates the free energy at a number of different unit cell volumes. These volumes are used to determine the temperature-dependent properties. As outlined in the following sections, there are several steps to this calculation that are unique to this script.

3.5.1 Generating Different Volumes

A set of lattice parameters is generated automatically by the script after reading in the relaxed lattice parameters and applying strain to the system. The script applies a series of six lattice parameters in each symmetry unique direction, ranging between one percent compressive and four percent tensile strain. An example of how stress and strain are applied to the lattice parameters are given in Eq. (1) as,

$$a = a_0 \left(0.99 + 0.01s\right) \tag{1}$$

where s = 0, 1, 2 3, 4, 5.

Each of these volumes is launched in parallel by the script in which a VASP calculation determines the ground state energy, forces, and stresses within the system. This information is used by TDEP to calculate the inter-atomic force constants and therefore the phonon density of states.

3.5.2 Free Energy

The temperature dependence of the lattice parameters is determined by finding the lattice parameters, $l_0(T)$, that minimizes the free energy at each temperature. The vibrational free energy (F$_{vib}$) as a function of the lattice parameters, l, and the temperature, T, as

$$F(l,T) = E_0(l) + F_{vib},$$
$$F_{vib} = \int_0^{\omega_{max}} \sum_\lambda \left[g(\omega_\lambda)\frac{\hbar\omega_\lambda}{2} + k_B T g(\omega_\lambda) \ln\left(1 - \exp\left(\frac{\hbar\omega_\lambda}{k_B T}\right)\right) d\omega_\lambda \right] \tag{2}$$

where λ is the phonon mode index. In Eq. (2), both the vibrational density of states ($g(\omega)$) and phonon frequencies depend on l. E_0 is the volume dependent internal energy of the system calculated with VASP and within TDEP.

3.5.3 Free Energy Surface

The optimized lattice parameters $l_0(T)$ are found by fitting the free energy to a polynomial function of the lattice parameters a, b, and c:

$$F(a) = \sum_{i,j} f_{i,j,k} a^i b^j c^k. \tag{3}$$

$f_{i,j,k}$ are the coefficients of the polynomial fit. We have in this study used fourth-order polynomials in the unique lattice parameter. Thus, for copper, only the polynomial is only a function of the "a" lattice parameter.

To determine the minimum of Eq. (3), we use the constrained Broyden-Fletcher-Goldfarb-Shanno minimization method as implemented in the Scipy optimize package [12, 13, 14]. After storing these minimizing parameters they are smoothed using a running average to account for the numerical noise in the free energy calculations.

3.5.4 Coefficients of Thermal Expansion

The coefficient of thermal expansion is then calculated by computing the derivative of the smoothed data [15] as

$$\alpha_a = \frac{d\ \ln(a)}{dT} \tag{4}$$

where the other components of the CTE are found in a similar manner. Note that the results of this calculation using either the zero temperature lattice constant or the temperature-dependent lattice constant, as shown here, are practically identical as assumed by Slack [15].

3.5.5 Debye Temperature

Using the elastic tensor, one can determine both the Debye temperature and isentropic bulk modulus of the system using the following relationships. First, the bulk (B) and shear moduli (G) are found from the components of the elastic tensor, c_{ij} (i,j = 1..6), as [16]

$$
\begin{aligned}
B &= \frac{1}{9}\Big((c_{11}+c_{22}+c_{33}) + 2(c_{12}+c_{13}+c_{23})\Big) \\
G &= \frac{1}{15}\Big((c_{11}+c_{22}+c_{33}) - (c_{12}+c_{13}+c_{23}) \\
&\quad + 3(c_{44}+c_{55}+c_{66})\Big).
\end{aligned} \tag{5}
$$

With the bulk and shear moduli one can then calculate the longitudinal, shear, and average sound velocities (v_l, v_s, and v_a) as [17]

$$
\begin{aligned}
v_l &= \sqrt{\frac{B + 4/3G}{\rho}}, \\
v_s &= \sqrt{\frac{G}{\rho}},
\end{aligned} \tag{6}
$$

and

$$
v_a = \left[\frac{1}{3}\left(\frac{1}{v_l^3} + \frac{2}{v_s^3}\right)\right]^{1/3} \tag{7}
$$

where ρ is the density of the material. Then, from the average velocity, we can calculate the Debye temperature as

$$
\Theta_D = \frac{h}{k_B}\left[\frac{3m}{4\pi}\right]^{1/3} v_a \tag{8}
$$

where h is Planks constant, k_B is Boltzmann constant, and m is the molar mass of the material. Since the Debye temperature is calculated from the elastic tensor, this Debye temperature takes into account the entire vibrational spectra, not just the acoustic modes [17].

4 Example Calculation

As an example of the use of this code, and a possible test case for future users, we will outline the calculation of the coefficients of thermal expansion of copper.

To start the calculation, the POSCAR file, which represents the unit cell of copper, is placed in the appropriate directory. For example, one could use:

```
Cu4
1.0
3.621263 0.000000 0.000000
0.000000 3.621263 0.000000
0.000000 0.000000 3.621263
Cu
4
direct
0.000000 0.000000 0.000000 Cu
0.000000 0.500000 0.500000 Cu
0.500000 0.000000 0.500000 Cu
0.500000 0.500000 0.000000 Cu
```

Using this POSCAR file and the tag --relaxation will lead to the creation of the data_extraction file which will contain the following information:

```
alat 3.633036
blat 3.633036
clat 3.633036
volume 47.950000
TMIN 0
TMAX 1600
TSTEP 800
atpos Cu Cu Cu Cu
natom 4
dielectric 8.255000 0.000000 0.000000
0.000000 8.255000 0.000000
0.000000 0.000000 8.255000
atommul 4
bectensor 1 -0.000000 0.000000 0.000000
-0.000000 0.000000 -0.000000
-0.000000 -0.000000 0.000000
2 -0.000000 0.000000 0.000000
-0.000000 0.000000 -0.000000
0.000000 -0.000000 0.000000
3 0.000000 -0.000000 -0.000000
0.000000 0.000000 0.000000
-0.000000 0.000000 0.000000
4 0.000000 -0.000000 -0.000000
0.000000 0.000000 -0.000000
0.000000 0.000000 -0.000000
elastic 1416.331000 1417.304100 1417.304100 0.000000 0.000000
0.000000
1417.304100 1416.331000 1417.30410 0 0.000000 -0.000000
0.000000
1417.304100 1417.304100 1416.331000 0.000000 -0.000000 -
0.000000
0.000000 0.000000 0.000000 709.427700 0.000000 0.000000
0.000000 -0.000000 -0.000000 0.000000 709.427700 0.000000
0.000000 0.000000 -0.000000 0.000000 0.000000 709.427700
```

The next step of the calculation uses --build_cells to generate the necessary volumes and to calculate the Debye temperature of the system. With the elastic tensor shown above, the Debye temperature and isoentropic bulk modulus are calculated and stored in the data_extraction file. The new information in the data_extraction file is:

```
Debye 322.141152
Bulk Mod 1416.979733
```

After all of the calculations have finished for the different configurations generated by the program, one can use the flag --thermal_expansion to extract the necessary temperature dependent data. Using the following command produces the data that are plotted below.

```
python ../ACTE.py --thermal_expansion --bounds --solver
```

In Fig. 1 we plot the calculated temperature dependent lattice parameter vs temperature and show available experimental data. As shown in

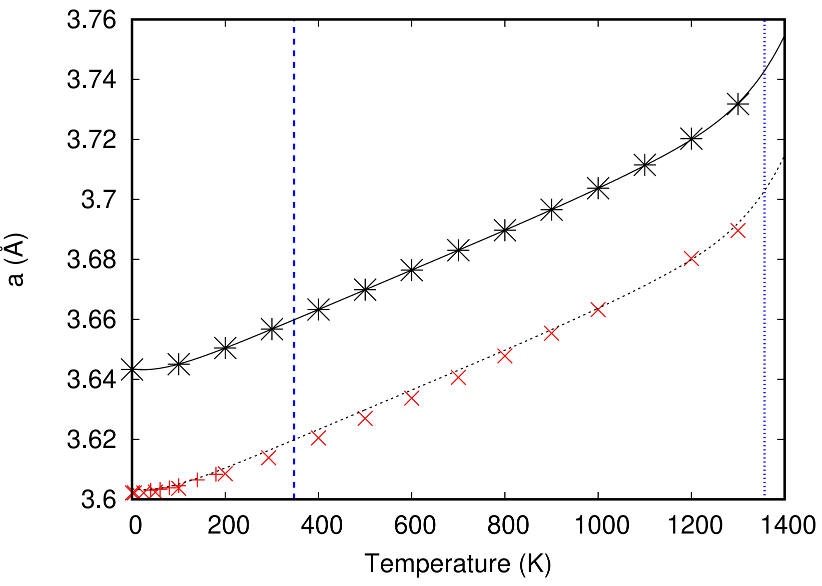

Figure 1: Calculated and experimental temperature dependent lattice parameters for Cu. Our theoretical calculations, shifted by l_{shift}, correspond to the solid line and experimental data for the lattice parameters from Ref. [18] are shown as black plus signs.

the figure, there is good agreement between the calculated temperature dependent lattice parameters and experimental measurement.

In Fig. 2 the coefficient of thermal expansion of copper is plotted against available experimental data. Once again, we find good agreement between our calculated values and experimental results.

Fig. 4 shows the calculated and experimental specific heat of copper as a function of temperature. As is generally assumed, the differences between C_v and C_p are small and below both experimental and theoretical error limits.

Finally, in Fig. 3 we provide the isothermal bulk modulus, and its zero pressure- pressure derivative compared to available experimental data. The points above the isothermal bulk modulus line for copper at room temperature correspond to isoentropic bulk modulus values.

References

[1] G. Kresse and J. Furthmuller. Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Comput. Mat. Sci.*, 6:15, 1996.

[2] G. Kresse and J. Furthmuller. Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. *Phys. Rev. B*, 54:11169, 1996.

[3] O. Hellman, P. Steneteg, I. A. Abriksov, and S. I. Simak. Temperature dependent effective potential method for accurate free energy calculations of solids. *Phys. Rev. B*, 87:10411, 2013.

[4] O. Hellman, I. A. Abriksov, and S. I. Simak. Lattice dynamics of anharmonic solids from first-principles. *Phys. Rev. B*, 84:180301,

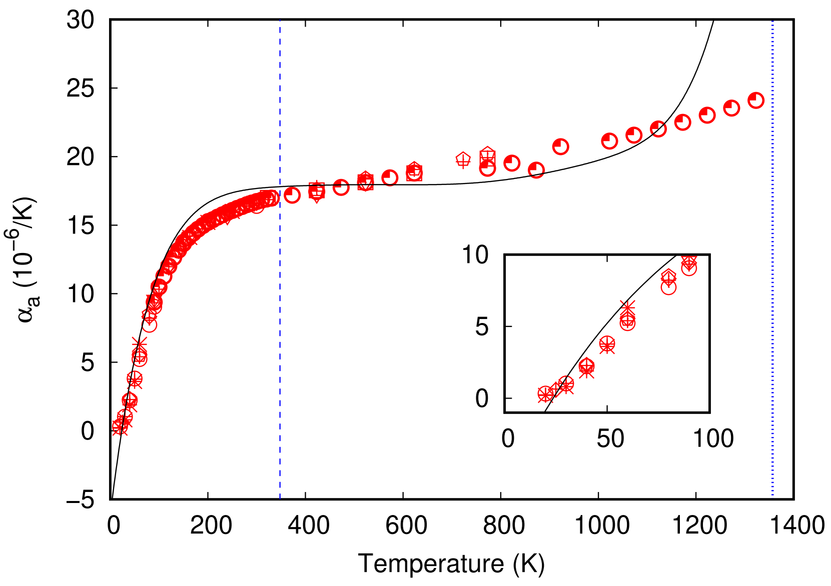

Figure 2: The CTE for Cu from our theoretical calculations is drawn with a solid line and experimental data as red symbols: plus signs (Ref. [19]), crosses (Ref. [20]), asterisks (Ref.[21]), open squares (Ref. [22]), open circles (Ref. [23]), open inverted triangles (Ref. [24]), open diamonds (Ref. [25]), and open pentagons (Ref. [26]).

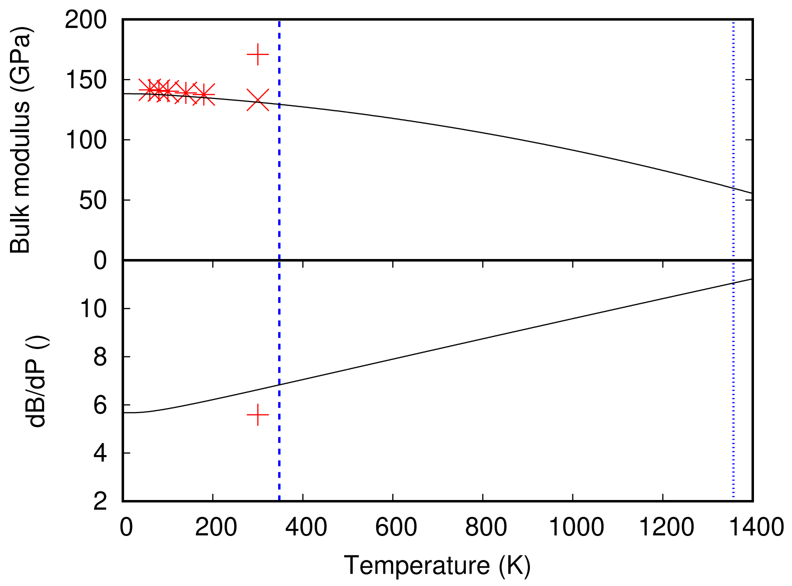

Figure 3: Calculated isothermal bulk modulus, and its pressure derivative, for Cu. Experimental data points for the isothermal bulk modulus are shown in black: plus signs from Ref. [27] for the isoentropic bulk modulus, crosses from Ref. [28], and asterisks from Ref. [18] are for the isothermal bulk modulus. Experimental data for the unit-less pressure derivative at room temperature, shown as a red open square, comes from Ref. [27].

2011.

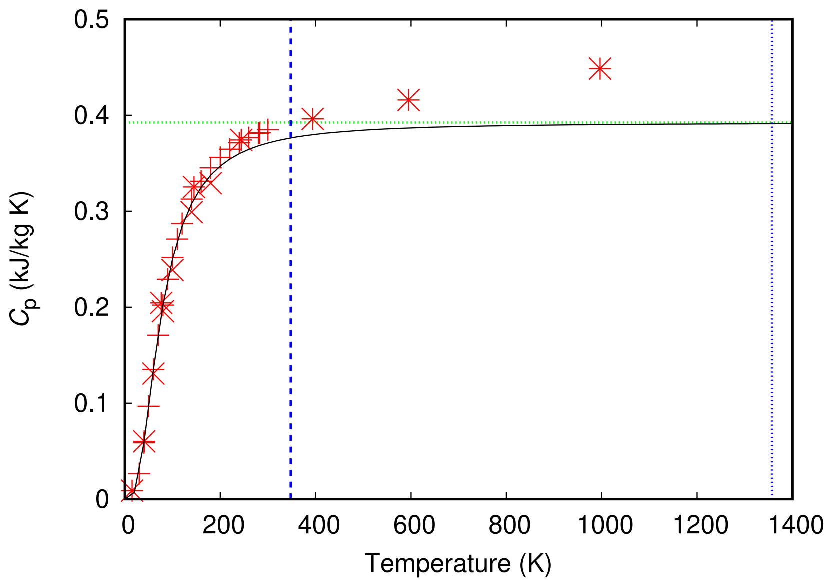

Figure 4: Calculated specific heat C_v and C_p for Cu. Literature data for C_p is given as blue points: plus signs from Ref. [29] and crosses from Ref. [18]. Our calculated values of the specific heats appears as a solid black and blue line, respectively. The log of the difference between C_p and C_v is given as the red line.

[5] O. Hellman and I. A. Abriksov. Temperature-dependent effective third-order interatomic force constants from first-principles. *Phys. Rev. B*, 88:144301, 2013.

[6] N. A. Pike and O. M. Løvvik. Algorithm for the calculation of the thermal expansion of anisotropic materials. *Comp. Mat. Sci*, 167:257–263, 2019.

[7] R. M. Martin. *Electronic Structure: Basic Theory and Practice Methods*. Cambridge University Press, 2004.

[8] X. Gonze. First-principles responses of solids to atomic displacements and homogeneous electric fields: Implementation of a conjugate-gradient algorithm. *Phys. Rev. B*, 55:10337, 1997.

[9] S. Baroni, S. de Gironcoli, A. Dal Corso, and P. Giannozzi. Phonons and related crystal properties from density-functional perturbation theory. *Rev. Mod. Phys.*, 73:515, 2001.

[10] X. Gonze and C. Lee. Dynamical matrices, born effective charges, dielectric permittivity tensors, and, interatomic force constants from density-functional perturbation theory. *Phys. Rev. B*, 55:10355, 1997.

[11] M. J. Verstraete and Z. Zanolli. Density functional perturbation theory. In *Computing Solids: Models, Ab-initio Methods and Supercomputing, Lecture Notes of the 45th Spring School 2014*. Schiften des Forschungszentrums Julich, 2014.

[12] E. Jones, T. Oliphant, P. Peterson, et al. Scipy: Open source scientific tools for python. http://www.scipy.org/, 2001 –. Accessed: 2018-10-30.

[13] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16:1190–1208, 1995.

[14] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. L-bfgs-b: Algorithm 778: L-bfgs-b, fortran routines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software*, 23:550–560, 1997.

[15] G. A. Slack and S. F. Bartram. Thermal expansion of some diamondlike crystals. *J. Appl. Phys.*, 46:89, 1975.

[16] N. A. Pike, B. Van Troeye, A. Dewandre, X. Gonze, and M. J. Verstraete. Vibrational and dielectric properties of the bulk transition metal dichalcogenides. *Phys. Rev. Mat.*, 2:06308, 2018.

[17] T. Jia, G. Chen, and Y. Zhang. Lattice thermal conductivity evaluated using elastic properties. *Phys. Rev. B*, 95:155206, 2017.

[18] J. S. Shah. *Thermal Lattice Expansion of Various Types of Solids.* PhD thesis, University of Missouri-Rolla, 1944.

[19] T. A. Hahn. Thermal expansion of copper from 20 to 800k- standard reference material 736. *J. of Appl. Phys.*, 41:5096, 1970.

[20] H. Adenstedt. Studien zur thermischen ausdehnung fester stoffe in tiefer temperatur (cu, ni, fe, zinkblende, lif, kalkspat, aragonit, nh4cl). *Ann. Physik*, 26:65, 1936.

[21] R. H. Carr, R. D. McCammon, and G. K. White. The thermal expansion of copper at low temperatures. *Proc. Roy. Soc. of London: A, Math. and Phys. Sci.*, 280:72–84, 1964.

[22] H. Esser and H. Eusterbrock. Untersuchung der wärmeausdehnung von einigen metallen und legierungen mit einem verbesserten dilatometer. *Arch. Eisenhuttenw.*, 14:341, 1941.

[23] D. B. Fraser and A. C. H. Hallett. *Can. J. Phys.*, 43:193, 1965.

[24] I. I. Lifanov and N. G. Sherstyukov. Thermal expansion of copper in the temperature range -185 to +300c. *Thermophysical Measurements*, 12:1653, 1968.

[25] T. Rubin, H. W. Altman, and H. L. Johnston. Coefficients of thermal expansion of solids at low temperatures. i. the thermal expansion of copper from 15 to 300k. *J. Am. Chem. Soc.*, 76:5289–5293, 1954.

[26] R. O. Simmons and R. W. Balluffi. Measurement of equilibrium concentrations of vacancies in copper. *Phys. Rev.*, 129:1533, 1963.

[27] P. W. Bridgman. *The physics of high pressure.* Dover Publications, 1931.

[28] R. E. Schmunk and C. S. Smith. Elastic constants of copper-nickel alloys. *Acta Metallurgica*, 8:396–401, 1960.

[29] G. K. White. Thermal expansion of reference materials: Copper, silica, and silicon. *J. Phys. D: Appl. Phys.*, 6:2070, 1973.

Index