ALAMODE Documentation

Release 0.9.0

Terumasa Tadano

June 09, 2014

Contents

CHAPTER 1

Users Guide

1.1 About

1.1.1 What is ALAMODE ?

ALAMODE stands for **A**nharmonic **LA**ttice **MODE**l, which is designed for estimating harmonic and anharmonic properties of lattice vibrations (phonons) in solids. The code is written in C++ and MPI/OpenMP parallelization is implemented.

1.1.2 Features

ALAMODE consists of two main programs **alm** and **anphon** written in C++, and some subsidiary Python scripts.

The program alm can estimate harmonic and anharmonic interatomic force constants (IFCs) based on the supercell approach. The program anphon can compute phonon properties using the estimated IFCs. Currently, it can compute the following quantities:

- Phonon dispersion, Phonon DOS, Atom-projected phonon DOS
- Thermodynamic functions (vibrational entropy, free energy, internal energy)
- Mean square displacement
- Heat capacity at constant volume
- Phonon linewidth due to 3-phonon interactions
- Phonon frequency shift
- Phonon self-energy due to phonon-isotope scatterings
- Lattice thermal conductivity

In addition, a python script may be used to estimate the cumulative thermal conductivity as a function of phonon mean-free-path.

1.1.3 License

If you used ALAMODE, please cite the following article:

> T. Tadano, Y. Gohda, and S. Tsuneyuki, J. Phys.: Condens. Matter **26**, 225402 (2014) [Link].

1.1.4 Acknowledgement

This project is support by a Grant-in-Aid for Scientific Research on Innovative Areas 'Materials Design through Computics: Complex Correlation and Non-Equilibrium Dynamics'. (http://computics-material.jp)

1.1.5 Author

Terumasa Tadano (terumasa.tadano{at}gmail.com)

Current affiliation: Department of Physics, The University of Tokyo, Japan

1.2 Download

The latest stable version of the package ALAMODE can be downloaded from this link.

We will also provide the Git repository of ALAMODE shortly.

1.3 Installation

1.3.1 Requirement

Mandatory requirements

- C++ compiler (Intel compiler is highly recommended.)
- LAPACK library
- MPI library (Either OpenMPI, MPICH2, or IntelMPI)
- Boost C++ library

In addition to the above requirements, users have to get and install a first-principles package (such as VASP, Wien2k, QUANTUM-ESPRESSO, or xTAPP) or another force field package (such as LAMMPS) by themselves in order to compute harmonic and anharmonic force constants. Currently, the package provides some auxiliary tools for the VASP code, which may be useful to make POSCARs for displaced configurations and to extract atomic forces from vasprun.xml files.

Optional requirements

- Python, Numpy, and Matplotlib
- Xcrysden

We provide some small scrips written in Python for visualizing phonon dispersion relations, phonon DOSs, etc. To use these scripts, one need to install the above Python packages. Xcrysden may be useful to visualize a zone-center normal mode.

1.3.2 How to install

1. Download the package from the download page.
2. Change directory to the location of the downloaded file and untar the file as follows:

```
$ tar -xvzf alamode-x.y.z.tar.gz
```

This will create a directory alamode-x.y.z containing the following sub-directories:

- alm/ : Source files for alm (force constant calculation)
- anphon/ : Source files for anphon (phonon calculation)
- external/ : Third-party include files
- include/ : Commonly-used include files
- tools/ : Small auxiliary programs and scripts

3. Edit the Makefiles

   The directories alm/, anphon/, and tools/ contain separate Makefiles. Please modify the Makefiles appropriately by changing variables such as CXX, CXXFLAGS, INCLUDE.

4. Make executables by make command.

1.4 Running ALAMODE

1.4.1 Program alm

Program *alm* estimates harmonic and anharmonic interatomic force constants (IFCs) based on the *supercell approach*.

1. Perform usual SCF calculations for a *primitive celll*

   Before performing phonon calculations, one needs to perform usual self-consistent calculations and check the convergence with respect to the cutoff energy and k point density. After that, please optimize the internal coordinate so that the atomic forces are negligibly small. Optimization of cell parameters may also be necessarily, but please note that phonon properties are relatively sensitive to the cell parameters in polar materials such as perovskites.

2. Decide to size of supercell

   Next, please decide the size of a supercell. Here, one may use a conventional cell. When the primitive cell is fairly large ($a \sim 10$ Å), one may proceed with the primive cell.

3. Prepare an input file for *alm*

   Please make an imput file for *alm*, say alm.in, which should contain `&general`, `&interaction`, `&cutoff`, and `&position` entries. For details of available input variables, please refer to *here*. Once the input file is properly prepared with `MODE = suggest`, necessarily displacement patterns may be generated by executing alm as follows:

   ```
   $ alm alm.in > alm.log
   ```

   This produces the following files containing the pattern of atomic displacements.

   - `PREFIX`.HARMONIC_pattern
   - `PREFIX`.ANHARM?_pattern (If `NORDER` > 1)

4. Perform SCF calculations to generate displacement-force data set

   The next step is to calculate atomic forces for the displaced configurations using a DFT package. Once the atomic forces are calculated, please collect the atomic displacements and atomic forces to separate files, say *disp_all.dat* and *force_all.dat*. The details of file format is described in *this page*.

Note: Atomic displacements and forces should be stored in units of Bohr and Ryd/Bohr, respectively.

5. Estimate IFCs by a least-squares fitting

   In order to perform a fitting, please change the variable `MODE` of the input file *alm.in* to `MODE = fitting`. In addition please add the `&fitting` entry with appropriate `NDATA`, `DFILE`, and `FFILE`. (`DFILE` should be like `DFILE = disp_all.dat`.) Then, IFCs can be estimated by executing

   ```
   $ alm alm.in > alm.log
   ```

   which makes the following two files in the working directory.

   - `PREFIX`.fcs : The list of force constants
   - `PREFIX`.xml : XML file containing necessarily information for subsequent phonon calculations

1.4.2 Program anphon

1. Prepare an input file for *anphon*

   To perform phonon calculations and thermal conductivity calculations, one needs to prepare another input file, say *anphon.in*, for the program *anphon*.

   If one wishes to perform (harmonic) phonon calculations, one should write `MODE = phonons` in the `&general` entry of anphon.in. Please make sure that `FCSXML` variable being set to the XML file generated by *alm*.

   If one wishes to perform thermal conductivity calculations instead of usual phonon calculations, please switch to `MODE = RTA` with appropriate `FCSXML` containing cubic IFCs.

   For details of input variables of anphon, please refer to the *list of input variables for anphon*.

2. Execute *anphon*

   Phonon properties and lattice thermal conductivity can be calculated via executing

   ```
   $ anphon anphon.in > anphon.log
   ```

   or

   ```
   $ mpirun -np NP anphon anphon.in > anphon.log
   ```

   Here, `NP` is the number of MPI threads. If the code is OpenMP enabled, the OpenMP parallelization can also be used by setting the `OMP_NUM_THREADS` variable as

   ```
   $ export OMP_NUM_THREADS=16
   ```

   The number 16 should be modified appropriately for your environment.

   Note: MPI parallelization can accelerate the calculation when `MODE = RTA`. In the current implementation of the code, however, OpenMP parallelization is more efficient.

   When the calculation finishes normally, various files are generated in the working directory.

   - `PREFIX`.bands : Phonon dispersion along designated Brillouin zone paths
   - `PREFIX`.dos : (Atom projected) phonon DOS
   - `PREFIX`.thermo : Thermodynamic functions
   - `PREFIX`.msd : Mean-square displacement of atoms

- ...

The complete list of output files can be found here.

3. Analyze the result

   One can plot the phonon dispersion relation or phonon DOS using gnuplot. Alternatively, one can use a small scripts in the `tools/` directory for visualizing these results. For example,

   ```
   $ ./plotband.py target.bands
   ```

   shows the phonon dispersion relation. Available command line options can be displayed by

   ```
   $ ./plotband.py -h
   ```

   We also provide a similar script for phonon DOS. Another script `analyze_phonons.py` may be useful to analyze the result of thermal conductivity calculations. For example, phonon lifetimes and mean-free-path at 300 K can be extracted by

   ```
   $ ./analyze_phonons.py --calc tau --temp 300 target.result
   ```

   It can also estimate a cumulative thermal conductivity by

   ```
   $ ./analyze_phonons.py --calc kappa_size --temp 300 --length 0:1000:1 target.result
   ```

1.5 Tutorial

Under construction.

1.6 Making input files for *alm*

1.6.1 Format of input files

Each input file should consists of entry fields. Available entry fields are

&general, **&interaction**, **&cutoff**, **&cell**, **&position**, and **&fitting**.

Each entry field starts from the key label **&field** and ends at the terminate character "/". (This is equivalent to Fortran namelist.)

For example, &general entry field of program *alm* should be given like

```
&general
  # Comment line
  PREFIX = prefix
  MODE = fitting
/
```

Multiple entries can be put in a single line. Also, any characters put on the right of sharp ("#") character will be neglected. Therefore, the above example is equivalent to the following:

```
&general
  PREFIX = prefix; MODE = fitting  # Comment line
/
```

Each variable should be written inside the appropriate entry field.

1.6.2 List of input variables

"&general"-field

- **PREFIX**-tag : Job prefix to be used for names of output files

  Default None

  Type String

- **MODE**-tag = fitting | suggest

fitting	Perform fittings to estimate harmonic and anharmonic IFCs. This mode requires appropriate DFILE and FFILE.
suggest	This mode suggests the displacement patterns necessary to estimate harmonic and anharmonic IFCS.

  Default None

  Type String

- **NAT**-tag : Number of atoms in the supercell

  Default None

  Type Integer

- **NKD**-tag : Number of atomic species

  Default None

  Type Integer

- **KD**-tag = Name[1], ... , Name[`NKD`]

  Default None

  Type Array of strings

  Example In the case of GaAs with `NKD = 2`, it should be `KD = Ga As`.

- NSYM-tag = 0 | 1 | nsym

0	The program automatically generates the crystal symmetry operations (rotational and translational parts). When `PRINTSYM = 1`, symmetry operations will be saved in the file "SYMM_INFO".
1	Only the identity operation will be considered.
nsym	"nsym" symmetry operations will be read from "SYMM_INFO" file.

Default 0

Type Integer

- TOLERANCE-tag : Tolerance for finding symmetry operations

  Default 1.0e-8

  Type Double

- PRINTSYM-tag = 0 | 1

0	Symmetry operations won't be saved in "SYMM_INFO"
1	Symmetry operations will be saved in "SYMM_INFO"

  Default 0

  type Integer

"&interaction"-field

- **NORDER**-tag : The order of force constants to be calculated. Anharmonic terms up to $(m+1)$th order will be considered with `NORDER` = m.

  Default None

  Type Integer

  Example `NORDER` should be 1 for harmonic calculations, and 2 to include cubic terms.

- NBODY-tag : Entry for excluding multiple-body interactions from anharmonic force constants

  Default `NBODY` = [2, 3, 4, ..., `NORDER` + 1]

  Type Array of integers

Example If one wishes to exclude three-body interactions from cubic force constants, one should explicitly give `NBODY = 2 2`.

"&cutoff"-field

In this entry field, one needs to specify cutoff radii of interaction for each order in units of Bohr. In the current implementation, cutoff radii should be defined for every possible pairs of atomic elements. For example, the cutoff entry for a harmonic calculation (`NORDER = 1`) of Si (`NKD = 1`) should be like

```
&cutoff
 Si-Si 10.0
/
```

This means that the cutoff radii of 10 a_0 will be used for harmonic Si-Si terms. Please note that the first column should be two character strings, which are contained in the `KD`-tag, connected by a hyphen ('-').

When one wishes to consider cubic terms (`NORDER = 2`), please specify the cutoff radius for cubic terms in the third column as the following:

```
&cutoff
 Si-Si 10.0 5.6 # Pair r_{2} r_{3}
/
```

Instead of giving specific cutoff radii, one can write "None" as follows:

```
&cutoff
 Si-Si None 5.6
/
```

which means that all possible harmonic terms between Si-Si atoms will be included.

Caution: Writing None for anharmonic terms can greatly increase the number of parameters, and hereby increase the computational cost.

When there are more than two atomic elements, please specify the cutoff radii between every possible pairs of atomic elements. In the case of MgO (`NKD = 2`), the cutoff entry should be like

```
&cutoff
 Mg-Mg 8.0
 O-O 8.0
 Mg-O 10.0
/
```

which can equivalently be written by using the wild card ('*') as

```
&cutoff
 *-* 8.0
 Mg-O 10.0 # Overwrite the cutoff radius for Mg-O harmonic interactions
/
```

Important: Cutoff radii specified by an earlier entry will be overwritten by a new entry that comes later.

Once the cutoff radii are properly given, harmonic force constants $\Phi_{i,j}^{\mu,\nu}$ satisfying $r_{ij} \leq r_c^{\mathrm{KD}[i]-\mathrm{KD}[j]}$ will be searched.

In the case of cubic terms, force constants $\Phi_{ijk}^{\mu\nu\lambda}$ satisfying $r_{ij} \leq r_c^{\mathrm{KD}[i]-\mathrm{KD}[j]}$, $r_{ik} \leq r_c^{\mathrm{KD}[i]-\mathrm{KD}[k]}$, and $r_{jk} \leq r_c^{\mathrm{KD}[j]-\mathrm{KD}[k]}$ will be searched and determined by fitting.

"&cell"-field

Please give the cell parameters in this entry in units of Bohr as the following:

```
&cell
 a
 a11 a12 a13
 a21 a22 a23
 a31 a32 a33
/
```

The cell parameters are then given by $\vec{a}_1 = a \times (a_{11}, a_{12}, a_{13})$, $\vec{a}_2 = a \times (a_{21}, a_{22}, a_{23})$, and $\vec{a}_3 = a \times (a_{31}, a_{32}, a_{33})$.

"&position"-field

In this field, one needs to specify the atomic element and fractional coordinate of atoms in the supercell. Each line should be

```
ikd xf[1] xf[2] xf[3]
```

where *ikd* is an integer specifying the atomic element (*ikd* = 1, ..., `NKD`) and *xf[i]* is the fractional coordinate of an atom. There should be `NAT` such lines in the &position entry field.

"&fitting"-field

This field is necessarily when `MODE = fitting`.

- **DFILE**-tag : File name containing atomic displacements in Cartesian coordinate

  Default None

  Type String

  Description The format of `DFILE` can be found *here*

- **FFILE**-tag : File name containing atomic forces in Cartesian coordinate

  Default None

  Type Integer

  Description The format of `FFILE` can be found *here*

- NSTART, NEND-tags : Specifies the range of data to be used for fitting

  Default `NSTART = 1`, `NEND = NDATA`

  Type Integer

Example When one wishes to use the data in the range of [20:30] out of 50 entries, one should set `NSTART = 20` and `NEND = 30`.

- ICONST-tag = 0 | 1 | 2 | 3

0	No constrans
1	Constraints for translational invariance will be imposed between IFCs.
2	In addition to `ICONST = 1`, constraints for rotational invariance will be imposed up to (`NORDER` + 1)th order.
3	In addition to `ICONST = 2`, constraints for rotational invariance between (`NORDER` + 1)th order and (`NORDER` + 2)th order, which are zero, will be considered.

Default 1

Type Integer

- ROTAXIS-tag : Rotation axis used to estimate constraints for rotational invariance. This entry is necessarily when `ICONST > 1`.

  Default None

  Type String

  Example When one wants to consider the rotational invariance around x-axis, one should give `ROTAXIS = x`. If one needs additional constraints for the rotation around y-axis, `ROTAXIS` should be `ROTAXIS = xy`.

- FC2XML-tag : XML file to which the harmonic terms will be fixed upon fitting

  Default None

  Type String

  Description When `FC2XML`-tag is given, harmonic force constants will be fixed to the values stored in the `FC2XML` file. This may be useful for optimizing cubic and higher-order terms without changing the harmonic terms. Please make sure that the number of harmonic terms in the new computational condition be the same as that in the `FC2XML` file.

1.6.3 Format of DFILE and FFILE

The displacement-force data sets obtained by first-principles (or classical force-field) calculations have to be saved to `DFILE` and `FFILE` to estimate IFCs with `MODE = fitting`. In `DFILE`, please explicitly specify the atomic displacements $u_\alpha(\ell\kappa)$ in units of Bohr as follows:

$$\begin{matrix} u_x(1) & u_y(1) & u_z(1) \\ u_x(2) & u_y(2) & u_z(2) \\ & \vdots & \\ u_x(\mathrm{NAT}) & u_y(\mathrm{NAT}) & u_z(\mathrm{NAT}) \end{matrix}$$

When there are `NAT` atoms in the supercell and `NDATA` data sets, there should be `NAT` $\times$ `NDATA` lines in the `DFILE` without blank lines. In `FFILE`, please specify the corresponding atomic forces in units of Ryd/Bohr.

1.7 Making input files for *anphon*

1.7.1 Format of input files

Each input file should consists of entry fields. Available entry fields are

&general, **&cell**, **&analysis**, and **&kpoint**.

The format of the input file is the same as that of *alm* which can be found *here*.

1.7.2 List of input variables

"&general"-field

- **PREFIX**-tag : Job prefix to be used for names of output files

  Default None

  Type String

- **MODE**-tag = phonons | RTA

phonons	Calculate phonon dispersion relations, phonon DOS, Grüneisen parameters etc.
RTA	Calculate phonon lifetimes and lattice thermal conductivity based on the Boltzmann transport equation (BTE) with the relaxation time approximation (RTA).

  Default None

  Type String

- **NKD**-tag : Number of atomic species

  Default None

  Type Integer

- **KD**-tag = Name[1], ... , Name[`NKD`]

  Default None

  Type Array of strings

  Example In the case of GaAs with `NKD = 2`, it should be `KD = Ga As`.

- **MASS**-tag = mass[1], ... , mass[`NKD`]

  Default None

  Type Array of double

  Example In the case of Bi_2Te_3 with `NKD = 2`, `MASS` should be `MASS = 208.98 127.60`.

- **FCSXML**-tag : XML file containing force constants generated by the program *alm*

  Default None

  Type String

- NSYM-tag = 0 | 1 | nsym

0	The program automatically generates the crystal symmetry operations (rotational and translational parts). When `PRINTSYM = 1`, symmetry operations will be saved in the file "SYMM_INFO_PRIM".
1	Only the identity operation will be considered.
nsym	"nsym" symmetry operations will be read from "SYMM_INFO_PRIM" file.

  Default 0

  Type Integer

- TOLERANCE-tag : Tolerance for finding symmetry operations

  Default 1.0e-8

  Type Double

- PRINTSYM-tag = 0 | 1

0	Symmetry operations won't be saved in "SYMM_INFO_PRIM"
1	Symmetry operations will be saved in "SYMM_INFO_PRIM"

  Default 0

  type Integer

- NONALAYTIC-tag = 0 | 1

0	Non-analytic correction is not considered.
1	Non-analytic correction will be considered. Appropriate `NA_SIGMA` and `BORNINFO` should be given.

  Default 0

  Type Integer

- **NA_SIGMA**-tag : Damping factor for the non-analytic term

  Default None

  Type Double

  Description This entry is necessarily when `NONANALYTIC = 1`. The definition of `NA_SIGMA` is described in the formalism section.

- **BORNINFO**-tag : File containing the macroscopic dielectric tensor and Born effective charges for the non-analytic correction

  Default None

  Type String

  Description The details of the file format can be found *here*.

- TMIN, TMAX, DT-tags : Temperature range and its stride in units of Kelvin

  Detaulf `TMIN = 0, TMAX = 1000, DT = 10`

  Type Double

- EMIN, EMAX, DELTA_E-tags : Energy range and its stride in units of kayser (cm^{-1})

  Default `EMIN = 0`, `EMAX = 1000`, `DELTA_E = 10`

  Type Double

- ISMEAR-tag = -1 | 0 | 1

-1	Tetrahedron method
0	Lorentzian smearing with width of `EPSILON`
1	Gaussian smearing with width of `EPSILON`

  Default -1

  Type Integer

  Description `ISMEAR` specifies the method for Brillouin zone integration

---

- EPSILON-tag : Smearing width in units of Kayser (cm^{-1})

  Default 10.0

  Type Double

  Description This variable is neglected when `ISMEAR = -1`

- TRISYM-tag : Flag to use symmetry operations to reduce the number of triples of k kpoints for self-energy calculations

0	Symmetry will not be used
1	Use symmetry to reduce triples of k points

  Default 1

  Type Integer

  Description This variable is used only when `MODE = RTA`.

  Note: `TRISYM = 1` can reduce the computational cost, but phonon linewidth stored to the file `PREFIX`.result needs to be averaged at points of degeneracy. For that purpose, a subsidiary program *analyze_phonons.py** should be used.

- RESTART-tag : Flag to restart the calculation when `MODE = RTA`

0	Calculate from scratch
1	Restart from the existing file

  Default 1 if there is a file named `PREFIX`.result; 0 otherwise

  Type Integer

"&cell"-field

Please specify the cell parameters of the *primitive cell* as:

```
&cell
 a
 a11 a12 a13
 a21 a22 a23
 a31 a32 a33
/
```

The cell parameters are then given by $\vec{a}_1 = a \times (a_{11}, a_{12}, a_{13})$, $\vec{a}_2 = a \times (a_{21}, a_{22}, a_{23})$, and $\vec{a}_3 = a \times (a_{31}, a_{32}, a_{33})$.

Note: The lattice constant a must be consistent with the value used for the program *alm*. For example, if one used $a = 20.4a_0$ for a 2x2x2 supercell of Si, one should use $a = 10.2a_0$ here for the primitive cell.

"&kpoint"-field

This entry field is used to specify the list of k points to be calculated. The first entry **KPMODE** specifies the types of calculation which is followed by detailed entries.

- **KPMODE = 0** : Calculate phonon frequencies at given k points

  For example, if one wishes to calculate phonon frequencies at Gamma (0, 0, 0) and X (0, 1/2, 1/2) of a FCC crystal, the `&kpoint` entry should be written as

  ```
  &kpoint
   0
   0.000 0.000 0.000
   0.000 0.500 0.500
  /
  ```

- **KPMODE = 1** : Band dispersion calculation

  For example, if one wishes to calculate phonon dispersion relations along G-K-X-G-L of a FCC crystal, the `&kpoint` entry should be written as follows:

  ```
  &kpoint
   1
   G 0.000 0.000 0.000  K 0.375 0.375 0.750 51
   K 0.375 0.375 0.750  X 0.500 0.500 1.000 51
   X 0.000 0.500 0.500  G 0.000 0.000 0.000 51
   G 0.000 0.000 0.000  L 0.500 0.500 0.500 51
  /
  ```

  The 1st and 5th columns specify the character of Brillouin zone edges, which are followed by fractional coordinates of each point. The last column indicates the number of sampling points.

- **KPMODE = 2** : Uniform k grid for phonon DOS and thermal conductivity

  In order to perform a calculation with 20x20x20 k grid, the entry should be

  ```
  &kpoint
   2
   20 20 20
  /
  ```

"&analysis"-field

- GRUNEISEN-tag = 0 | 1

0	Grüneisen parameters will not be calculated
1	Grüneisen parameters will be stored

Default 0

Type Integer

Description When `MODE = phonons` and `GRUNEISEN = 1`, Grüneisen parameters will be stored in `PREFIX`.gru (*KPMODE* = 1) or `PREFIX`.gru_all (*KPMODE* = 2).

Note: To compute Grüneisen parameters, cubic force constants must be contained in the `FCSXML` file.

- PRINTEVEC-tag = 0 | 1

0	Do not print phonon eigenvectors
1	Print phonon eigenvectors in the `PREFIX`.evec file

Default 0

Type Integer

- PRINTXSF-tag = 0 | 1

0	Do not save an AXSF file
1	Create an AXSF file `PREFIX`.axsf

Default 0

Type Integer

Description This may be useful to visualize phonon modes at gamma (0, 0, 0) by XCrysDen. The option is valid only when `MODE = phonons`.

- PRINTVEL-tag = 0 | 1

0	Do not print group velocity
1	Store phonon velocities to a file

Default 0

Type Integer

Description When `MODE = phonons` and `PRINTVEL = 1`, group velocities of phonons will be stored in `PREFIX`.phvel (*KPMODE* = 1) or `PREFIX`.phvel_all (*KPMODE* = 2).

- PRINTMSD-tag = 0 | 1

0	Do not print mean-square-displacement (MSD) of atoms
1	Save MSD of atoms to the file `PREFIX`.mds

Default 0

Type Integer

Description This flag is available only when `KPMODE = phonons` and *KPMODE* = 2.

- PDOS-tag = 0 | 1

0	Only the total DOS will be printed in `PREFIX`.dos
1	Atom-projected phonon DOS will be stored in `PREFIX`.dos

Default 0

Type Integer

Description This flag is available only when `KPMODE = phonons` and *KPMODE* = 2.

- TDOS-tag = 0 | 1

0	Do not compute two-phonon DOS
1	Two-phonon DOS will be stored in `PREFIX`.tdos

Default 0

Type Integer

Description This flag is available only when `KPMODE = phonons` and *KPMODE* = 2.

Note: Calculation of two-phonon DOS is computationally expensive.

- ISOTOPE-tag = 0 | 1

0	Do not consider phonon-isotope scatterings
1	Consider phonon-isotope scatterings

Default 0

Type Integer

Description When `MODE = RTA` and `ISOTOPE = 1`, phonon scatterings due to isotopes will be considered perturbatively. `ISOFACT` should be properly given.

- ISOFACT-tag = isofact[1], ... , isofact[`NKD`]

Default 0

Type Array of doubles

Description Isotope factor is a dimensionless value defined by $\sum_i f_i(1-m_i/\bar{m})^2$. Here, f_i is the fraction of ith isotope of an element having mass m_i, and $\bar{m} = \sum_i f_i m_i$ is the average mass, respectively. This quantity is equivalent to g_2 appearing in the original paper by S. Tamura [Phys. Rev. B, 27, 858.].

1.7.3 Format of BORNINFO

When one wants to consider the LO-TO splitting near the Γ point, it is necessarily to set `NONANALYTIC = 1` and provide `BORNINFO` file containing dielectric tensor ϵ^{∞} and Born effective charge Z^*. In `BORNINFO` file, dielectric tensor should be written in first 3 lines which is followed by Born effective charge tensors for each atoms as the following.

$$
\begin{array}{ccc}
\epsilon^{\infty}_{xx} & \epsilon^{\infty}_{xy} & \epsilon^{\infty}_{xz} \\
\epsilon^{\infty}_{yx} & \epsilon^{\infty}_{yy} & \epsilon^{\infty}_{yz} \\
\epsilon^{\infty}_{zx} & \epsilon^{\infty}_{zy} & \epsilon^{\infty}_{zz} \\
Z^*_{1,xx} & Z^*_{1,xy} & Z^*_{1,xz} \\
Z^*_{1,yx} & Z^*_{1,yy} & Z^*_{1,zz} \\
Z^*_{1,zx} & Z^*_{1,zy} & Z^*_{1,zz} \\
 & \vdots & \\
Z^*_{N_p,xx} & Z^*_{N_p,xy} & Z^*_{N_p,xz} \\
Z^*_{N_p,yx} & Z^*_{N_p,yy} & Z^*_{N_p,zz} \\
Z^*_{N_p,zx} & Z^*_{N_p,zy} & Z^*_{N_p,zz}
\end{array}
$$

Here, N_p is the number of atoms contained in the *primitive cell*.

Attention: Please pay attention to the order of Born effective charges.

1.8 Formalism for program alm

1.8.1 Interatomic force constants (IFCs)

The starting point of the computational methodology is to approximate the potential energy of interacting atoms by a Taylor expansion with respect to atomic displacements by

$$
\begin{aligned}
U - U_0 &= \sum_{n=1}^{N} U_n = U_1 + U_2 + U_3 + \cdots, \\
U_n &= \frac{1}{n!} \sum_{\substack{\ell_1\kappa_1,\ldots,\ell_n\kappa_n \\ \mu_1,\ldots,\mu_n}} \Phi_{\mu_1\ldots\mu_n}(\ell_1\kappa_1;\ldots;\ell_n\kappa_n)\, u_{\mu_1}(\ell_1\kappa_1)\cdots u_{\mu_n}(\ell_n\kappa_n).
\end{aligned}
$$

Here, $u_\mu(\ell\kappa)$ is the atomic displacement of κth atom in the ℓth unit cell along μth direction, and $\Phi_{\mu_1\ldots\mu_n}(\ell_1\kappa_1;\ldots;\ell_n\kappa_n)$ is the nth-order interatomic force constant (IFC).

1.8.2 Symmetry relationship between IFCs

The are several relationships between IFCs which may be used to reduce the number of independence IFCs.

- Permutation

  Firstly, IFCs should be invariant under the exchange of triplet (ℓ, κ, μ), e.g.

  $$\Phi_{\mu_1\mu_2\mu_3}(\ell_1\kappa_1;\ell_2\kappa_2;\ell_3\kappa_3) = \Phi_{\mu_1\mu_3\mu_2}(\ell_1\kappa_1;\ell_3\kappa_3;\ell_2\kappa_2) = \ldots.$$

- Periodicity

  Secondly, since IFCs should depend on interatomic distances, they are invariant under a translation in units of lattice vector, namely

$$\Phi_{\mu_1\mu_2\ldots\mu_n}(\ell_1\kappa_1;\ell_2\kappa_2;\ldots;\ell_n\kappa_n) = \Phi_{\mu_1\mu_2\ldots\mu_n}(0\kappa_1;\ell_2-\ell_1\kappa_2;\ldots;\ell_n-\ell_1\kappa_n).$$

- Crystal symmetry

  A crystal symmetry operation maps an atom $\vec{r}(\ell\kappa)$ to another equivalent atom $\vec{r}(LK)$ by rotation and translation. Since the potential energy is invariant under any crystal symmetry operations, IFCs should transform under a symmetry operation as follows:

$$\sum_{\nu_1,\ldots,\nu_n} \Phi_{\nu_1\ldots\nu_n}(L_1K_1;\ldots;L_nK_n)O_{\mu_1\nu_1}\cdots O_{\mu_n\nu_n} = \Phi_{\mu_1\ldots\mu_n}(\ell_1\kappa_1;\ldots;\ell_n\kappa_n), \tag{1.1}$$

  where O is the rotational matrix of the symmetry operation. Let N_s be the number of symmetry operations, there are N_s relationships between IFCs which may be used to find independent IFCs.

Note: In the current implementation of *alm*, independent IFCs are searched in Cartesian coordinate where the matrix element $O_{\mu\nu}$ is 0 or ±1 in all symmetry operations except for those of **hexagonal** (trigonal) lattice. Also, except for hexagonal (trigonal) systems, the product $O_{\mu_1\nu_1}\cdots O_{\mu_n\nu_n}$ in the left hand side of equation (1.1) becomes non-zero only for a specific pair of $\{\nu\}$ (and becomes 0 for all other $\{\nu\}$s). Therefore, let $\{\nu'\}$ be such a pair of $\{\nu\}$, the equation (1.1) can be reduced to

$$\Phi_{\nu'_1\ldots\nu'_n}(L_1K_1;\ldots;L_nK_n) = s\Phi_{\mu_1\ldots\mu_n}(\ell_1\kappa_1;\ldots;\ell_n\kappa_n), \tag{1.2}$$

where $s=\pm1$. The code employs equation (1.2) instead of equation (1.1) to reduce the number of IFCs. If IFCs of the left-hand side and the right-hand side of equation (1.2) are equivalent and the coupling coefficient is $s=-1$, the IFC is removed since it becomes zero. For **hexagonal** (trigonal) systems, there can be symmetry operations where multiple terms in the left-hand side of equation (1.1) become non-zero. For such cases, equation (1.1) is not used to reduce the number of IFCs. Alternatively, the corresponding symmetry relationships are imposed as constraints between IFCs in solving fitting problems.

1.8.3 Constraints between IFCs

Since the potential energy is invariant under rigid translation and rotation, it may be necessarily for IFCs to satisfy corresponding constraints.

The constraints for translational invariance are given by

$$\sum_{\ell_1\kappa_1} \Phi_{\mu_1\mu_2\ldots\mu_n}(\ell_1\kappa_1;\ell_2\kappa_2;\ldots;\ell_n\kappa_n) = 0, \tag{1.3}$$

which should be satisfied for arbitrary pairs of $\ell_2\kappa_2,\ldots,\ell_n\kappa_n$ and $\mu_1,\ldots,\mu_n$. The code *alm* imposes equation (1.3) by default (`ICONST = 1`).

The constraints for rotational invariance are

$$\begin{aligned}\sum_{\ell'\kappa'}&(\Phi_{\mu_1\ldots\mu_n\nu}(\ell_1\kappa_1;\ldots;\ell_n\kappa_n;\ell'\kappa')r_\mu(\ell'\kappa') - \Phi_{\mu_1\ldots\mu_n\mu}(\ell_1\kappa_1;\ldots;\ell_n\kappa_n;\ell'\kappa')r_\nu(\ell'\kappa'))\\ &+\sum_{\lambda=1}^{n}\sum_{\mu'_\lambda}\Phi_{\mu_1\ldots\mu'_\lambda\ldots\mu_n}(\ell_1\kappa_1;\ldots;\ell_\lambda\kappa_\lambda;\ldots;\ell_n\kappa_n)(\delta_{\mu,\mu_\lambda}\delta_{\nu,\mu'_\lambda}-\delta_{\nu,\mu_\lambda}\delta_{\mu,\mu'_\lambda}) = 0,\end{aligned}$$

which must be satisfied for arbitrary pairs of $(\ell_1\kappa_1,\ldots,\ell_n\kappa_n;\mu_1,\ldots,\mu_n;\mu,\nu)$. This is complicated since $(n+1)$th-order IFCs (first line) are related to nth-order IFCs (second line).

For example, the constraints for rotational invariance related to harmonic terms can be found as

$$\sum_{\ell_2\kappa_2}(\Phi_{\mu_1\nu}(\ell_1\kappa_1;\ell_2\kappa_2)r_\mu(\ell_2\kappa_2)-\Phi_{\mu_1\mu}(\ell_1\kappa_1;\ell_2\kappa_2)r_\nu(\ell_2\kappa_2)) + \Phi_\nu(\ell_1\kappa_1)\delta_{\mu,\mu_1}-\Phi_\mu(\ell_1\kappa_1)\delta_{\nu,\mu_1}=0,$$

and

$$\sum_{\ell_3\kappa_3}(\Phi_{\mu_1\mu_2\nu}(\ell_1\kappa_1;\ell_2\kappa_2;\ell_3\kappa_3)r_\mu(\ell_3\kappa_3)-\Phi_{\mu_1\mu_2\mu}(\ell_1\kappa_1;\ell_2\kappa_2;\ell_3\kappa_3)r_\nu(\ell_3\kappa_3)) + \Phi_{\nu\mu_2}(\ell_1\kappa_1;\ell_2\kappa_2)\delta_{\mu,\mu_1}-\Phi_{\mu\mu_2}(\ell_1\kappa_1;\ell_2\kappa_2)\delta_{\nu,\mu_1} + \Phi_{\mu_1\nu}(\ell_1\kappa_1;\ell_2\kappa_2)\delta_{\mu,\mu_2}-\Phi_{\mu_1\mu}(\ell_1\kappa_1;\ell_2\kappa_2)\delta_{\nu,\mu_2}=0.$$

When `NORDER = 1`, equation (1.8.3) will be considered if `ICONST = 2`, whereas equation (1.8.3) will be neglected. To further consider equation (1.8.3), please use `ICONST = 3`, though it may enforce a number of harmonic IFCs to be zero since cubic terms don't exist in harmonic calculations (`NORDER = 1`).

1.9 Formalism for program anphon

1.9.1 Dynamical matrix

The dynamical matrix is given by

$$D_{\mu\nu}(\kappa\kappa';\boldsymbol{q})=\frac{1}{\sqrt{M_\kappa M_{\kappa'}}}\sum_{\ell'}\Phi_{\mu\nu}(\ell\kappa;\ell'\kappa')\exp\left[i\boldsymbol{q}\cdot(\boldsymbol{r}(\ell')-\boldsymbol{r}(\ell))\right], \tag{1.4}$$

where M_κ is the atomic mass of atom κ. By diagonalizing the dynamical matrix, one can obtain $m\ (=3N_\kappa)$ eigenvalues $\omega^2_{\boldsymbol{q}j}$ $(j=1,2,\ldots,m)$ and corresponding eigenvectors $\boldsymbol{e}_{\boldsymbol{q}j}$ for each $\boldsymbol{q}$ point. Here, $\boldsymbol{e}_{\boldsymbol{q}j}$ is a column vector consisting of atomic polarization $e_\mu(\kappa;\boldsymbol{q}j)$. Let $D(\boldsymbol{q})$ denote a matrix form of equation (1.4), the eigenvalues may be written as

$$\omega^2_{\boldsymbol{q}j}=(\boldsymbol{e}^*_{\boldsymbol{q}j})^{\mathrm{T}}D(\boldsymbol{q})\boldsymbol{e}_{\boldsymbol{q}j}. \tag{1.5}$$

Next, we introduce $m\times m$ matrices Λ and W which are defined as $\Lambda(\boldsymbol{q})=\mathrm{diag}(\omega^2_{\boldsymbol{q}1},\ldots,\omega^2_{\boldsymbol{q}m})$ and $W(\boldsymbol{q})=(\boldsymbol{e}_{\boldsymbol{q}1},\ldots,\boldsymbol{e}_{\boldsymbol{q}m})$, respectively. Then, equation (1.5) can be denoted as

$$\Lambda(\boldsymbol{q})=W^\dagger(\boldsymbol{q})D(\boldsymbol{q})W(\boldsymbol{q}).$$

hen one needs to capture the LO-TO splitting near the zone-center by the supercell approach, it is necessarily to add the non-analytic part of the dynamical matrix defined by

$$D^{\mathrm{NA}}_{\mu\nu}(\kappa\kappa';\boldsymbol{q})=\frac{1}{\sqrt{M_\kappa M_{\kappa'}}}\frac{4\pi e^2}{\Omega}\frac{(Z^*_\kappa\boldsymbol{q})_\mu(Z^*_{\kappa'}\boldsymbol{q})_\nu}{\boldsymbol{q}\cdot\epsilon^\infty\boldsymbol{q}},$$

where Ω is the volume of the primitive cell, Z^*_κ is the Born effective charge tensor of atom κ, and ϵ^∞ is the dielectric constant tensor, respectively. In the current implementation of *anphon*, we employ the Parlinski's way [1] to recover the analytic behavior at $\boldsymbol{q}$ points away from the zone-center. Hence, the total dynamical matrix is given by

$$D(\boldsymbol{q})+D^{\mathrm{NA}}(\boldsymbol{q})\exp\left(-q^2/\sigma^2\right),$$

where σ is a damping factor which must be chosen carefully so that the non-analytic contribution becomes negligible at Brillouin zone boundaries. To include the non-analytic term, one needs to set `NONANALYTIC = 1` and give appropriate `BORNINFO` and `NA_SIGMA` tags.

[1] K. Parlinski, Z. Q. Li, and Y. Kawazoe, Phys. Rev. Lett. **81**, 3298 (1998).

1.9.2 Group velocity

The group velocity of phonon mode $\boldsymbol{q}j$ is given by

$$\boldsymbol{v}_{\boldsymbol{q}j} = \frac{\partial \omega_{\boldsymbol{q}j}}{\partial \boldsymbol{q}}.$$

To evaluate the group velocity numerically, we employ a central difference where $\boldsymbol{v}$ may approximately be given by

$$\boldsymbol{v}_{\boldsymbol{q}j} \approx \frac{\omega_{\boldsymbol{q}+\Delta\boldsymbol{q}j} - \omega_{\boldsymbol{q}-\Delta\boldsymbol{q}j}}{2\Delta\boldsymbol{q}}.$$

If one needs to save the group velocities, please turn on the `PRINTVEL`-tag.

1.9.3 Thermodynamics functions

The specific heat at constant volume $C_{\rm v}$, the internal energy U, the vibrational entropy S, and the Helmholtz free energy F of individual harmonic oscillator are given as follows:

$$\begin{aligned}
U &= \frac{1}{N_q}\sum_{\boldsymbol{q},j}\hbar\omega_{\boldsymbol{q}j}\left[\frac{1}{e^{\hbar\omega_{\boldsymbol{q}j}/kT}-1}+\frac{1}{2}\right], \\
C_{\rm v} &= \frac{k}{N_q}\sum_{\boldsymbol{q},j}\left(\frac{\hbar\omega_{\boldsymbol{q}j}}{2kT}\right)^2 \mathrm{cosech}^2\left(\frac{\hbar\omega_{\boldsymbol{q}j}}{2kT}\right), \\
S &= \frac{k}{N_q}\sum_{\boldsymbol{q},j}\left[\frac{\hbar\omega_{\boldsymbol{q}j}}{kT}\frac{1}{e^{\hbar\omega_{\boldsymbol{q}j}/kT}-1} - \log\left(1-e^{-\hbar\omega_{\boldsymbol{q}j}/kT}\right)\right], \\
F &= \frac{1}{N_q}\sum_{\boldsymbol{q},j}\left[\frac{\hbar\omega_{\boldsymbol{q}j}}{2} + kT\log\left(1-e^{-\hbar\omega_{\boldsymbol{q}j}/kT}\right)\right].
\end{aligned}$$

Here, k is the Boltzmann constant. These quantities will be saved in the `PREFIX`.thermo file.

1.9.4 Mean square displacement

The mean square displacement tensor of atom $kappa$ is given by

$$\begin{aligned}
\langle u_\mu(\kappa)u_\nu(\kappa)\rangle = \frac{\hbar}{2M_\kappa N_q}\sum_{\boldsymbol{q},j}\frac{1}{2\omega_{\boldsymbol{q}j}}&\left(e_\mu(\kappa;\boldsymbol{q}j)e^*_\nu(\kappa;\boldsymbol{q}j) + e^*_\mu(\kappa;\boldsymbol{q}j)e_\nu(\kappa;\boldsymbol{q}j)\right) \\
&\times \coth\left(\frac{\hbar\omega_{\boldsymbol{q}j}}{2kT}\right).
\end{aligned} \tag{1.6}$$

When `PRINTMSD` is turned on, the code print the diagonal part of the mean square displacement tensor

$$\left\langle u^2_\mu(\kappa)\right\rangle = \frac{\hbar}{M_\kappa N_q}\sum_{\boldsymbol{q},j}\frac{1}{\omega_{\boldsymbol{q}j}}|e_\mu(\kappa;\boldsymbol{q}j)|^2\left(n_{\boldsymbol{q}j}+\frac{1}{2}\right),$$

where $n_{\boldsymbol{q}j} = 1/(e^{\hbar\omega_{\boldsymbol{q}j}/kT}-1)$ is the Bose-Einstein distribution function.

1.9.5 Phonon DOS

When *KPMODE* = 2, the program *anphon* saves the (one) phonon density of states (DOS) to the file `PREFIX`.dos. The one-phonon DOS is given by

$$\mathrm{DOS}(\omega) = \frac{1}{N_q}\sum_{\boldsymbol{q},j}\delta(\omega-\omega_{\boldsymbol{q}j}).$$

If `PDOS = 1` is given, the program also print the atom-projected phonon DOS which is given by

$$\mathrm{PDOS}(\kappa;\omega) = \frac{1}{N_q}\sum_{\boldsymbol{q},j}|\boldsymbol{e}(\kappa;\boldsymbol{q}j)|^2\delta(\omega-\omega_{\boldsymbol{q}j}).$$

In addition, `TDOS`-tag is available to compute the two-phonon DOS which is defined by

$$\mathrm{DOS2}(\sigma_1\sigma_2;\omega) = \frac{1}{N_q^2}\sum_{\boldsymbol{q}_1,\boldsymbol{q}_2,j_1,j_2}\delta(\omega+\sigma_1\omega_{\boldsymbol{q}_1 j_1}+\sigma_2\omega_{\boldsymbol{q}_2 j_2}),$$

where $\sigma_{1,2} = \pm 1$. Please note that the computation of the two-phonon DOS can be expensive especially when N_q or N_κ is large.

1.9.6 Grüneisen parameter

The mode Grüneisen parameter, defined as $\gamma_{\boldsymbol{q}j} = -\frac{\partial \log \omega_{\boldsymbol{q}j}}{\partial \log V}$, is calculated by

$$\gamma_{\boldsymbol{q}j} = -\frac{(\boldsymbol{e}^*_{\boldsymbol{q}j})^{\mathrm{T}}\delta D(\boldsymbol{q})\boldsymbol{e}_{\boldsymbol{q}j}}{6\omega_{\boldsymbol{q}j}},$$

where $\delta D(\boldsymbol{q})$ is a change in the dynamical matrix due to a volume change δV, which is given by

$$\delta D_{\mu\nu}(\kappa\kappa';\boldsymbol{q}) = \frac{1}{\sqrt{M_\kappa M_{\kappa'}}}\sum_{\ell'}\delta\Phi_{\mu\nu}(\ell\kappa;\ell'\kappa')\exp\left[i\boldsymbol{q}\cdot(\boldsymbol{r}(\ell')-\boldsymbol{r}(\ell))\right], \tag{1.7}$$

$$\delta\Phi_{\mu\nu}(\ell\kappa;\ell'\kappa') = \sum_{\ell'',\kappa'',\lambda}\Phi_{\mu\nu\lambda}(\ell\kappa;\ell'\kappa';\ell''\kappa'')r_\lambda(\ell''\kappa''). \tag{1.8}$$

Please set `GRUNEISEN = 1` and give an appropriate `FCSXML` file containing cubic IFCs to print Grlumulaut_ulneisen parameters.

1.9.7 Anharmonic self-energy

The anharmonic self-energy due to cubic anharmonicity to the lowest order is given by

$$\begin{aligned}\Sigma_{\boldsymbol{q}j}(i\omega_m) = \frac{1}{2}\sum_{\boldsymbol{q}_1,\boldsymbol{q}_2}\sum_{j_1,j_2}|V^{(3)}_{-\boldsymbol{q}j,\boldsymbol{q}_1 j_1,\boldsymbol{q}_2 j_2}|^2 \\ \times\left[\frac{n_1+n_2+1}{i\omega_m+\omega_1+\omega_2}-\frac{n_1+n_2+1}{i\omega_m-\omega_1-\omega_2}+\frac{n_1-n_2}{i\omega_m-\omega_1+\omega_2}-\frac{n_1-n_2}{i\omega_m+\omega_1-\omega_2}\right],\end{aligned}$$

where $i\omega_m$ is the Matsubara frequency. In equation (1.9.7), we simply denoted $\omega_{\boldsymbol{q}_i j_i}$ as ω_i. The matrix element $V^{(3)}$ is given by

$$\begin{aligned}V^{(3)}_{\boldsymbol{q}j,\boldsymbol{q}'j',\boldsymbol{q}''j''} = \left(\frac{\hbar}{2N_q}\right)^{\frac{3}{2}}\frac{1}{\sqrt{\omega_{\boldsymbol{q}n}\omega_{\boldsymbol{q}'j'}\omega_{\boldsymbol{q}''j''}}}\sum_{\ell,\ell',\ell''}\exp\left[\mathrm{i}(\boldsymbol{q}\cdot\boldsymbol{r}(\ell)+\boldsymbol{q}'\cdot\boldsymbol{r}(\ell')+\boldsymbol{q}''\cdot\boldsymbol{r}(\ell''))\right] \\ \times\sum_{\kappa,\kappa',\kappa''}\frac{1}{\sqrt{M_\kappa M_{\kappa'}M_{\kappa''}}}\sum_{\mu,\nu,\lambda}\Phi_{\mu\nu\lambda}(\ell\kappa;\ell'\kappa';\ell''\kappa'')e_\mu(\kappa;\boldsymbol{q}j)e_\nu(\kappa';\boldsymbol{q}'j')e_\lambda(\kappa'';\boldsymbol{q}''j'')\ ,\end{aligned}$$

which becomes zero unless $\boldsymbol{q}+\boldsymbol{q}'+\boldsymbol{q}''$ is a integral multiple of $\boldsymbol{G} = n_1\boldsymbol{b}_1+n_2\boldsymbol{b}_2+n_3\boldsymbol{b}_3$. Phonon linewidth $\Gamma_{\boldsymbol{q}j}$, which is the imaginary part of the phonon self-energy, can be obtained by the analytic continuation to the real axis ($i\omega_m \to \omega + i0^+$) as

$$\begin{aligned}\Gamma^{\mathrm{anh}}_{\boldsymbol{q}j}(\omega) = \frac{\pi}{2}\sum_{\boldsymbol{q}_1,\boldsymbol{q}_2}\sum_{j_1,j_2}|V^{(3)}_{-\boldsymbol{q}j,\boldsymbol{q}_1 j_1,\boldsymbol{q}_2 j_2}|^2 \\ \times\left[-(n_1+n_2+1)\delta(\omega+\omega_1+\omega_2)+(n_1+n_2+1)\delta(\omega-\omega_1-\omega_2)\right. \\ \left.-(n_1-n_2)\delta(\omega-\omega_1+\omega_2)+(n_1-n_2)\delta(\omega+\omega_1-\omega_2)\right].\end{aligned}$$

The computation of equation (1.9.7) is the most expensive part of the thermal conductivity calculations. Therefore, we employ the crystal symmetry to reduce the number of triplet pairs $(\boldsymbol{q}j, \boldsymbol{q}'j', \boldsymbol{q}''j'')$ of $V^{(3)}$ to calculated. To disable the reduction, please set `TRISYM = 0`.

1.9.8 Isotope scattering

The effect of isotope scatterings can be considered by the mass perturbation approach proposed by S. Tamura [2] by the `ISOTOPE`-tag. The corresponding phonon linewidth is given by

$$\Gamma^{\rm iso}_{\boldsymbol{q}j}(\omega) = \frac{\pi}{4N_q}\omega^2_{\boldsymbol{q}j} \sum_{\boldsymbol{q}_1, j_1} \delta(\omega - \omega_{\boldsymbol{q}_1 j_1}) \sum_{\kappa} g_2(\kappa) |\boldsymbol{e}^*(\kappa; \boldsymbol{q}_1 \boldsymbol{j}_1) \cdot \boldsymbol{e}(\kappa; \boldsymbol{q}\boldsymbol{j})|^2,$$

where g_2 is a dimensionless factor given by

$$g_2(\kappa) = \sum_i f_i(\kappa) \left(1 - \frac{m_i(\kappa)}{M_\kappa}\right)^2.$$

Here, f_i is the fraction of ith isotope of an element having mass m_i, and $M_\kappa = \sum_i f_i m_i(\kappa)$ is the average mass, respectively. The g_2 values should be provided by the `ISOFACT`-tag.

1.9.9 Lattice thermal conductivity

The lattice thermal conductivity tensor $\kappa^{\mu\nu}_{\rm ph}(T)$ is estimated within the relaxation-time approximation as

$$\kappa^{\mu\nu}_{\rm ph}(T) = \frac{1}{\Omega N_q} \sum_{\boldsymbol{q}, j} c_{\boldsymbol{q}j}(T) v^{\mu}_{\boldsymbol{q}j} v^{\nu}_{\boldsymbol{q}j} \tau_{\boldsymbol{q}j}(T),$$

where $c_{\boldsymbol{q}j} = \hbar\omega_{\boldsymbol{q}j}\partial n_{\boldsymbol{q}j}/\partial T$ and $\tau_{\boldsymbol{q}j}(T)$ is the phonon lifetime. The phonon lifetime is estimated using the Matthiessen's rule as

$$\tau^{-1}_{\boldsymbol{q}j}(T) = 2(\Gamma^{\rm anh}_{\boldsymbol{q}j}(T) + \Gamma^{\rm iso}_{\boldsymbol{q}j}).$$

The lattice thermal conductivity will be written to the file `PREFIX`.kl.

1.9.10 Delfa function

In order to compute the phonon DOSs and the imaginary part of phonon self-energies, it is necessarily to evaluate the Brillouin-zone integration containing Dirac's delta function. For that purpose, we provide 3 options through the `ISMEAR`-tag.

When `ISMEAR = 0`, the delta function is replaced by the Lorentzian function as

$$\delta(\omega) \approx \frac{1}{\pi}\frac{\epsilon^2}{\omega^2 + \epsilon^2}.$$

When `ISMEAR = 1`, the delta function is replaced by the Gaussian function as

$$\delta(\omega) \approx \frac{1}{\sqrt{\pi}\epsilon} \exp\left(-\omega^2/\epsilon^2\right),$$

which decays faster than the Lorentzian function. For both cases, ϵ should be given by the `EPSILON`-tag, which must be chosen carefully to avoid any unscientific results. ϵ should be small enough to capture detailed phonon structures such as phonon DOS or energy conservation surface related to three-phonon process, but it should be large enough to avoid unscientific oscillations. Choosing appropriate value for ϵ is not a trivial task since it may depend on the phonon structure and the density of $\boldsymbol{q}$ points.

To avoid such issues, the program *anphon* employs the tetrahedron method [3] by default (`ISMEAR = -1`) for numeri-

[2] S. -I. Tamura, Phys. Rev. B **27**, 858 (1983).
[3] P. E. Blöchl, O. Jepsen, and O. K. Andersen, Phys. Rev. B **49**, 1450555 (1994).

cal evaluations of Brillouin zone integration containing $\delta(\omega)$. When the tetrahedron method is used, the `EPSILON`-tag is neglected. We recommend to use the tetrahedron method whenever possible, even though it may slightly increase the computational cost.

CHAPTER 2

Indices and tables

- *genindex*
- *modindex*
- *search*