# AICON2 User Manual

AICON2 (Ab Initio Conductivities) is a program which aims to calculate the transport properties of bulk materials quickly and accurately. The first version of AICON provided a robust and highly efficient method (modified Debye-Callaway model) for calculating the lattice thermal conductivity[1]. In this new version, we add functions for calculating electronic transport properties, such as electrical conductivity, Seebeck coefficient *etc.*, which enables AICON2 to do both electron and phonon transport calculations.

For now, the program only relies on the resulting files of VASP and Phonopy for obtaining the electron and phonon information, but it can easily be interfaced to other codes. Please contribute if you make such an interface.

## 1 Getting started

### 1.1 Unpacking

Run the command: unzip AICON-2.0.x.zip
This will unpack several directories

| | |
|---|---|
| Scripts | executable scripts of the program |
| examples | several examples for testing |
| aicon | source codes directory |
| doc | useful documents, including this manual |
| LICENSE | GPL-3.0 license |
| README.md | description of the program |
| setup.py | install script |

### 1.2 Installation

AICON2 is a Python module. AICON2's runtime requirements are Python version 3.5 or higher, and the Python libraries NumPy, SciPy, spglib and pymatgen. If you want to use automatic workflow management tools, atomate and FireWorks should also be installed. All of them can be easily obtained from the Python Package Index (PyPI), using tools such as pip. They may also be bundled with Python distributions aimed at scientists, like Anaconda, and with a number of Linux distributions. Here we recommend to use Anaconda so that dependencies should be resolved automatically.

Install from pip:

```
pip install AICON
```

This will automatically download and install AICON and its dependent python packages.
Or install from source distribution, enter into the package directory and run:

```
python setup.py install
```

you can also specify the installing directory by running like this:

```
python setup.py install --prefix=/your_dir
```

However, this requires the directory you specified is on the python path or else it may report errors.
The installing process should be quick. If there is no error happening during installation and it ends up normally, you may find an executable named **AICON** in directory like *~/anaconda3/bin/*, or the directory you specified.

## 2 Using the program

The workflow of using AICON2 to calculate the electronic transport properties and phonon transport properties are shown in Figure 1 and 2. The user should finish all required first principles calculations before calculating the transport properties. The program can be called either from a command-line interface or from a script. Run the command:

```
AICON -h
```

to see the arguments and their explanations.
The **examples** directory in the package includes **Electron** directory and **Phonon** directory. The **Electron** directory includes four examples: PbTe, $Bi_2Te_3$, SnSe-Pnma and SnSe-Cmcm, and they are analyzed in our paper[2]. The **Phonon** directory includes three examples: diamond, Si, SnSe-Pnma, and they are analyzed in our previous paper[1].

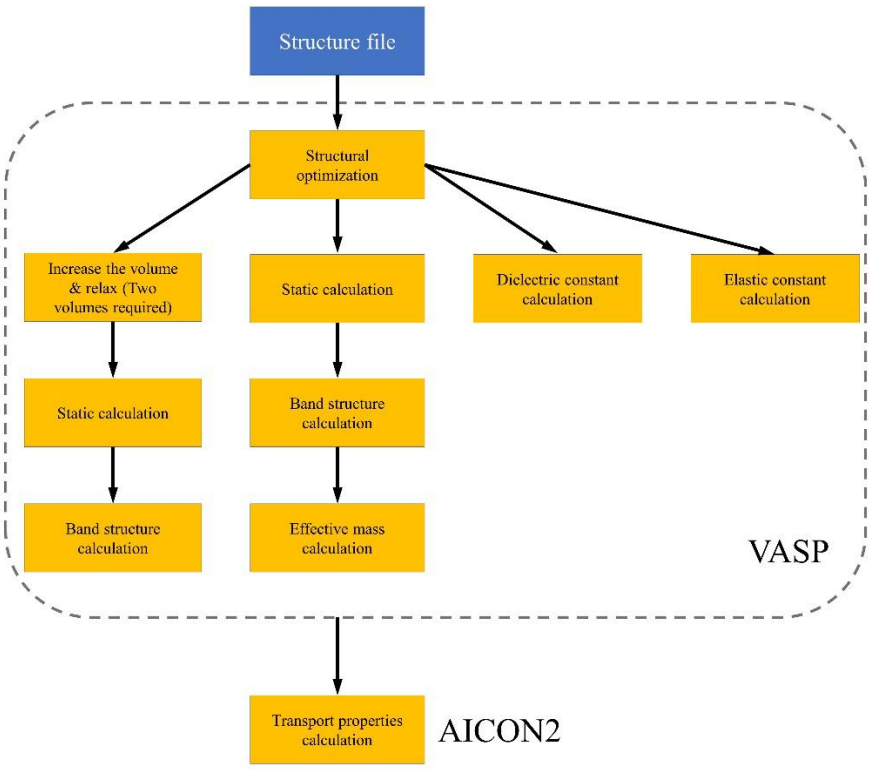


Figure 1. Workflow of the electronic transport properties calculation

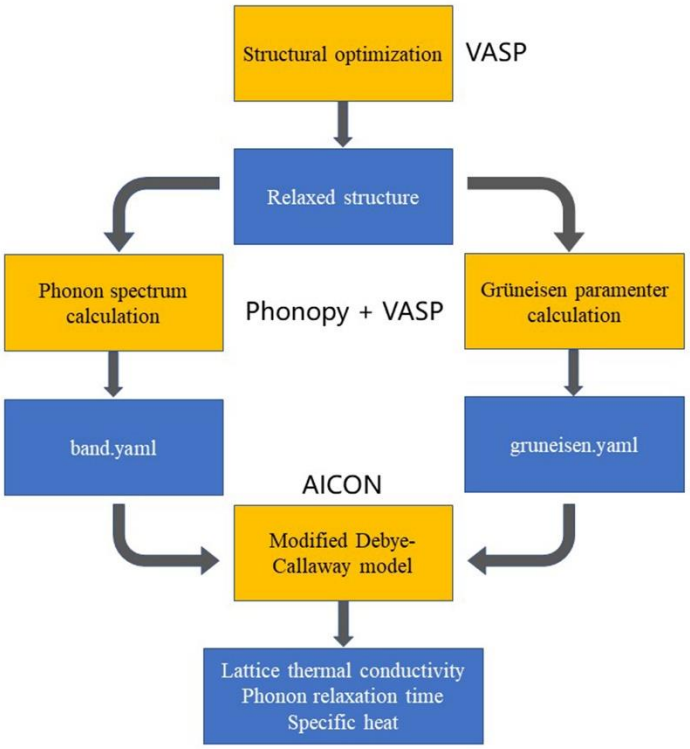


Figure 2. Workflow of the phonon transport properties calculation

2.1 Run an electronic transport calculation example

Enter into */examples/Electron/PbTe* directory, several folders exist in this directory:

| | |
|---|---|
| equi | folder of equilibrium structure's band structure |
| 0.5per | folder of 0.1% larger structure's band structure |
| 1.0per | folder of 0.1% smaller structure's band structure |
| CBM | folder of conduction band minimum calculation |
| VBM | folder of valence band maximum calculation |
| VSB | folder of second valence band calculation |
| dielect | folder of dielectric constant calculation |
| elastic | folder of elastic constant calculation |

These folders contain the results of corresponding VASP calculations. For users' own cases, the users should collect them under the same directory and name each folder **as exact as** we did, or else the program will not find the results. For calculating the electronic transport properties, the program can be run in two modes: 'standard' or 'doping'. The 'standard' mode calculates and outputs the transport coefficients as a function of specified temperature at default chemical potential values. The chemical potential starts from the middle of the gap and goes to the level that is 1 eV higher with a step of 2 meV. The 'doping' mode calculates and outputs the results as a function of specified temperature and carrier concentration. For example, if you want to calculate values at temperature from 300 K to 400 K with a step of 50 K in 'standard' mode, type

```
AICON --elec -m standard --tmin 300 --tmax 400 --tstep 50
```

or

```
AICON --elec -m standard -t 300 350 400
```

If you want to calculate in 'doping' mode, you should also specify the doping concentration, type

```
AICON --elec -m doping -t 300 350 400 -d 1e+18 1e+19
```

The unit of doping concentration is $cm^{-3}$, and the program will calculate both n-type and p-type at specified carrier concentrations. There is a special argument "--noSB", which means **do not** consider the second band's (second valence band VSB or second conduction band CSB) contribution. This argument can be used in case the user only want to include CBM and VBM in the transport calculations.
The program can also be called from a python script. See *aiconforele.py* in *PbTe* directory.
Let's check the results. The key parameters are summarized in *Parameters* file. Each columns is:

| band : | $m_{\parallel}^{*}$ ($m_e$) | $m_{\perp}^{*}$ ($m_e$) | $m_c^{*}$ ($m_e$) | $m_d^{*}$ ($m_e$) | $\Xi$ (eV) | $N$ | $E_g$ (eV) | $c$ (GPa) | $\varepsilon_\infty$ F/m | $\varepsilon_0$ F/m |
|---|---|---|---|---|---|---|---|---|---|---|

Detailed definition of each parameter can be found in our paper[2]. The calculated transport properties are written into several *.xlsx(default) files. This file format can be changed. our program first stores results as pandas.DataFrame object and pandas supports many kinds of file format. Check this site to find a format you want and modify the last several lines in *electron.py* file. If the calculated structure has no CSB or VSB, only CBM.xlsx and VBM.xlsx files exist. If it has, say, CSB, then there are another two files — CSB.xlsx and TCB.xlsx — appearing. TCB.xlsx contains total results from CBM and CSB, while CBM.xlsx and CSB.xlsx contain each band's contribution respectively. Similar for TVB.xlsx and VSB.xlsx. Each column in these files represents:

In 'doping' mode

| Index | |
|---|---|
| Temperature | The specified temperature (K) |
| dope | The user specified carrier concentration ($cm^{-3}$) |
| Concentration | The real concentration used to get chemical potential ($m^{-3}$) |
| Seebeck | The Seebeck coefficient ($V \cdot K^{-1}$) |
| Mobility | The carrier mobility ($m^2 \cdot V^{-1} \cdot s^{-1}$) |
| Elcond | The electrical conductivity ($\Omega^{-1} \cdot m^{-1}$) |
| Lorenz | The Lorenz number ($V^2 \cdot K^2$) |
| Ekappa | The electronic thermal conductivity ($W \cdot m^{-1} \cdot K^{-1}$) |
| Hallcoeff | The Hall coefficient ($m^3 \cdot C^{-1}$) |
| PF | The thermoelectric power factor ($W \cdot m^{-1} \cdot K^{-2}$) |
| TotalRelaxTime | Total relaxation time (s) |
| AcoRelaxTime | Acoustic phonon scattering relaxation time (s) |
| OptRelaxTime | Polar optical phonon scattering relaxation time (s) |
| ImpRelaxTime | Ionic impurity scattering relaxation time (s) |

In 'standard' mode, the third column is "mu" (chemical potential, in eV) instead of "dope". Other columns are the same as 'doping' mode.

2.2 Running a phonon transport calculation example

Suppose you have already gotten the second order force constant file using Phonopy, the next thing is obtaining *band.yaml* file which contains information of phonon frequency and phonon velocity. There is a special requirement of high symmetry path in which phonon frequency is calculated on. This path should be generated by AICON2. There **must** be a structural file in your directory named POSCAR(VASP format unit cell) and then run the command:

```
AICON --phon --highpath
```

it will show some message like this,

```
Please set "BAND" parameter of phonopy as this:

0.3750 0.3750 0.7500  0.0000 0.0000 0.0000  0.5000 0.5000 0.5000  0.6250 0.2500 0.6250  0.0000 0.0000 0.0000  0.5000 0.2500 0.7500
  0.5000 0.0000 0.5000  0.0000 0.0000 0.0000

We notice your structure could have a primitive cell. Please set "PRIMITIVE_AXIS" parameter of phonopy as this:

0.00000000 0.50000000 0.50000000  0.50000000 0.00000000 0.50000000  0.50000000 0.50000000 0.00000000
```

Set the Phonopy input file(such as *band.conf*) as the message said, for diamond:

```
ATOM_NAME = C
DIM = 2 2 2
PRIMITIVE_AXIS = 0.0 0.5 0.5  0.5 0.0 0.5  0.5 0.5 0.0
BAND = 0.3750 0.3750 0.7500  0.0000 0.0000 0.0000  0.5000 0.5000 0.5000  0.6250 0.2500 0.6250  0.0000 0.0000 0.0000  0.5000 0.2500
 0.7500  0.5000 0.0000 0.5000  0.0000 0.0000 0.0000
FORCE_CONSTANTS = READ
GROUP_VELOCITY = .TRUE.
```

Then run the command:

phonopy -c POSCAR-unitcell band.conf

You will get the *band.yaml* file if no error occurs. The next thing is obtaining the *gruneisen.yaml* file. Suppose you have already finished all three needed calculations (one at the equilibrium volume, one at the slightly small volume, another at the slightly large volume. Details about how to calculate Grüneisen parameters can be seen here: http://atztogo.github.io/phonopy/gruneisen.html#phonopy-gruneisen). Then run:

phonopy-gruneisen orig plus minus --dim="2 2 2" --pa="0 1/2 1/2 1/2 0 1/2 1/2 1/2 0" --band="0.3750 0.3750 0.7500 0.0000 0.0000 0.0000 0.5000 0.5000 0.5000 0.6250 0.2500 0.6250 0.0000 0.0000 0.0000 0.5000 0.2500 0.7500 0.5000 0.0000 0.5000 0.0000 0.0000 0.0000" --readfc -c POSCAR-unitcell

Attention! The high symmetry path and primitive cell transform matrix should be exactly the same as in the *band.conf* file.

You will get *gruneisen.yaml* file if no error occurs. Now all input files needed by AICON2 for calculating thermal conductivity have already been obtained. You should put them in the same directory like this:

```
(base) [Tao.Fan@frontend gruneisen]$ ls
band.yaml  gruneisen.yaml  grunsi  minus  orig  plus  POSCAR
```

Then run the command:

AICON --phon --tmin 300 --tmax 1000 --tstep 50
Or
AICON --phon --tmin 300 --tmax 1000 --tstep 50 --scale

The special argument, “--scale”, will enable the program multiply a scaling factor with

the calculated kappa values, so that the final results will have smaller Root Mean Square relative Deviation (RMSrD) compared with results without this factor according to our test. See our new paper about its explaination and the testing data. The factor value we used is 0.60371, and this unmber is obtained by fitting the calculated and experimental data of testing materials. The users can change this value according to their needs (see *Get_kappa* function in *phonon.py*).

The results are written into the file named Kappa.xlsx. Again, the file format can be changed (class method *Output* in *phonon.py*). Each column in these files represents:

| Index | |
|---|---|
| Temp | The specified temperature (K) |
| Kappa | The lattice thermal conductivity ($W \cdot m^{-1} \cdot K^{-1}$) |
| *_N | The Normal process relaxation time (s) |
| *_U | The Umklapp process relaxation time (s) |
| *_ISO | The Isotope defect scattering relaxation time (s) |

Also, this calculation can be executed by a script. See *aiconforpho.py* in *C* directory.

### 2.3 Using automatic workflow management tools

As shown in Figure 1, in order to calculate the electronic transport properties, at least 13 DFT calculations need to be done. Although each of them is relatively simple, managing the workflow becomes an intensive task, which involves checking the results of each step and copying the necessary files from the last step. To alleviate this problem for high-throughput calculation, in AICON2 we also implemented a customized automatic workflow tools based on pymatgen, atomate and FireWorks. The users only need to provide structural files and some necessary settings for each DFT calculation, then the whole workflow can run automatically. Two sample scripts (*createelecworkflow.py* and *createphonworkflow.py*) showing how to use this workflow tools were put in *Electron* and *Phonon* directory, respectively. We strongly encourage users to use this workflow tools. Although the installation and configuration of required packages is a little bit painful, it worth much more once you use it. Besides, it is also convenient to build users' own databases this way.

## 3 Frequently asked questions

### 3.1 How do I know if a compound has CSB or VSB?

A band is recognized as CSB(VSB) if it has the same band index as CBM(VBM) and the energy difference with CBM(VBM) is within 0.2 eV. Once you have finished equilibrium structure's band calculation, you can use tool provided by AICON2 to check if the band has CSB or VSB. The script (*findSB.py*) is in *PbTe* directory. However, we still recommend you to use our automatic workflow tools so that everything will be dealt automatically.

## Reference

[1] T. Fan and A.R. Oganov, AICON: A program for calculating thermal conductivity quickly and accurately, *Computer Physics Communications*, **251** (2020) 107074.
[2] T. Fan and A.R. Oganov, AICON2: A program for calculating transport properties quickly and accurately, *Computer Physics Communications*, **266** (2021) 108027.