BoltzTraP_Tools UserGuide

Hilal BALOUT

2018

1 What is BoltzTraP_Tools?

BoltzTraP_Tools is an interface written using Python 2 language[1]. It allows to parse and plot BoltzTraP [2] output DATA (***.trace, .condtens,.trace_fixdoping, and .condtens_fixdoping***) in greater detail.
For that Numpy[3] and Matplotlib[4] (Pylab) Python packages are needed.
All files can be downloaded from here.

BoltzTraP_Tools includes four folders:

- **src** : includes source files.
- **doc** : includes the user guide and some tutorials.
- **tests** : includes two BoltzTraP examples (output data).
- **scripts** : includes some scripts using BoltzTraP_Tools.

2 What BoltzTraP_Tools can do?

As mentionned before BoltzTraP_Tools can read all ***TRACE*** and ***CONDTENS*** output files.
Therefore, it will be possible to plot the following quantities :

- Energy level : ϵ
- Temperature : T
- Seebeck Coiffecients : S, S_{xx}, S_{yy}, and S_{zz}
- Electrical Conductivity : σ, σ_{xx}, σ_{yy}, and σ_{zz}
- Power Factor : PF, PF_{xx}, PF_{yy}, and PF_{zz}
- Thermal Conductivity : κ, κ_{xx}, κ_{yy}, and κ_{zz}
- Number of Carriers : n
- Hall Coefficient : R_H
- Electronic Specific Heat : c
- Pauli Magnetic : χ

3 BoltzTraP_Tools Functions

BoltzTraP_Tools consists six functions which allow to collect and plot DATA from a file:

1. **Labels_Init :** to initialize units and scale factors for all quantities.
2. **Scaling_DATA :** to set and scale all quantites.
3. **File_Read :** to read from file.
4. **DATA_Process :** to collect DATA for given parameters (Temperature, Energy ...).
5. **Get_DATA :** is a subfunction of **DATA_Process**.
6. **Plot_DATA :** to plot DATA by using Pylab Python Package.

4 Quantites : Units and Scale Factors

The default units of all quantities are listed below :

Quantity	Label	Unit	Scale Factor
Energy level (ϵ)	E	Ry	1.0
Temperature (T)	T	K	1.0
Number of Carriers (n)	N	e/uc	1.0
Density of Stats ($n(\epsilon)$)	DOS	e/uc	1.0
Seebeck Coefficient (S)	S	V/K	1.0
Electrical Conductivity (σ)	Sigma	$1/(\Omega.cm.s)$	1.0
Power Factor ($S^2\sigma$)	PF	$W/(K^2.cm.s)$	1.0
Thermal Conductivity (κ^0)	Kappa	$W/(m.K.s)$	1.0
Hall Coefficient (R_H)	R_H	m^3/C	1.0
Electronic Specific Heat (c)	c	$J/(mol.K)$	1.0
Pauli Magnetic (χ)	chi	m^3/mol	1.0

However, using ***Scaling_DATA*** function, units and scale factors can be modified for all quantities.
With this feature, the unit and the scale factor for every quantitiy must be given manually.

1. First, all quantities informations must be initialized by **Labels_Init()** function.

- As below :

```
labels = Labels_Init()
```

labels will be a dictonary which contains all quantitiy informations such: index, label, unit and scale factor.

2. Then, all quantites can be scaled or all default parameters can be conserved.

   - As below :

```
Scaling_DATA(labels)
"Setting of Units and Scale Factors (y/n) ? >"  n #  Default Set.
"Setting of Units and Scale Factors (y/n) ? >"  y #  Manual Set.
```

3. However, it is possible to scale an individual quantitiy.

   - Example: scaling of Seebeck coefficient of about 10^6 and its unit becomes $\mu V/K$:

```
labels = Labels_Init()
Scaling_DATA(labels)
#   Out:  "Setting of Units and Scale Factors (y/n) ? >"  n #  Default Set.

print labels["S"]
#   Out: [4, ' Seebeck ', ' $S$; ', ' ($V/K$) ', 1.0]

labels["S"][4]= 1e6 # the fourth index corresponds to the quantitiy scale factor.
labels["S"][3]= " ($\mu V/K$) " # the third index corresponds to the quantitiy unit.

print labels["S"]
#   Out: [4, ' Seebeck ', ' $S$; ', ' ($\\mu V/K$) ', 1000000.0]
```

More informations can be found in Scaling_Qunatities Tutorial.

5 Files Type:

5.1 Trace and Condtens Files

Trace and Condtens output files contain all calculated quantites as a function of temperature and energy.

5.2 N-Trace and N-Condtens Files

N-Trace and N-Condtens output files contain all calculated quantities as a function of temperature at fixed doping level.

6 Usage of BoltzTraP_Tools

BoltzTraP_Tools includes output files of two BoltzTraP examples:

- $CoSb_3$
- $LiZnSb$

For using BoltzTraP_Tools a script file can be written or by calling BoltzTraP_Tools functions progressively.

- Script example:

```
from BoltzTraP_Tools import *


labels=Labels_Init()
Scaling_DATA(labels)
Analyse=raw_input("File Extension (Trace, Condtens, N-Trace, or  N-Condtens) ? > ")
File_DATA,Ef=File_Read(Analyse)
DATA_Process(Analyse,File_DATA,Ef,labels)
```

6.1 BoltzTraP_Tools Plot Features

BoltzTraP_Tools can plot any quantity as a function of another one:

1. Parse **Trace** or **N-Trace**; Xplot and Yplot can be :

- *Xplot (E, T, N, DOS, S, Sigma, PF, R_H, Kappa, c or chi)*
- *Yplot (E, T, N, DOS, S, Sigma, PF, R_H, Kappa, c or chi)*

2. Parse **Condtens** or **N-Condtens**; Xplot and Yplot can be:
   - *Xplot (E, T, N, S, Sxx, Syy, Szz, Sigma, Sigmaxx, Sigmayy,Sigmazz, PF, PFxx, PFyy, PFzz, Kappa, Kappaxx, Kappayy or Kappazz)*
   - *Yplot (E, T, N, S, Sxx, Syy, Szz, Sigma, Sigmaxx, Sigmayy,Sigmazz, PF, PFxx, PFyy, PFzz, Kappa, Kappaxx, Kappayy or Kappazz)*

In addition, for **Trace** and **Condtens** files, the plot can be performed at fixed temperature or fixed energy:

- *Parse at fixed Temperature or Energy ? (T/E)*

However, for **N-Trace** and **N-Condtens** files, the plot can be performed at fixed doping level. For all plot the *logscale* of carriers concentration can be chosen,and hence the *electron* or *hole* type:

- *Log Scale for Carrier Concentration ? (y/n)*
- *Plot of electron or hole ?*

Tutorials can be found here

7 Installation

BoltzTraP_Tools sources can be downloaded from the Github Repository.
Once you have a copy of the source, you can install it with:

```
python setup.py install
```

8 Report Bugs

Report bugs at Bugs.
For every bug, it is therefore appropriate to specify the:

- Operating system name and version.
- Detailed steps to reproduce the bug.

References

[1] G. van Rossum et al. The Python programming language, 1991–. URL: http://www.python.org.

[2] Georg K.H. Madsen and David J. Singh. Boltztrap. a code for calculating band-structure dependent quantities. *Computer Physics Communications*, 175(1):67 – 71, 2006.

[3] Stefan van der Walt, S. Chris Colbert, and Gael Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science and Engg.*, 13(2):22–30, March 2011.

[4] J. D. Hunter. Matplotlib, 2004. URL: http://matplotlib.sourceforge.net.