

---

# **dfttk Documentation**

***Release 0.3.4***

**Yi Wang, Mingqing Liao, Brandon J. Bocklund, Shunli Shang, Long**

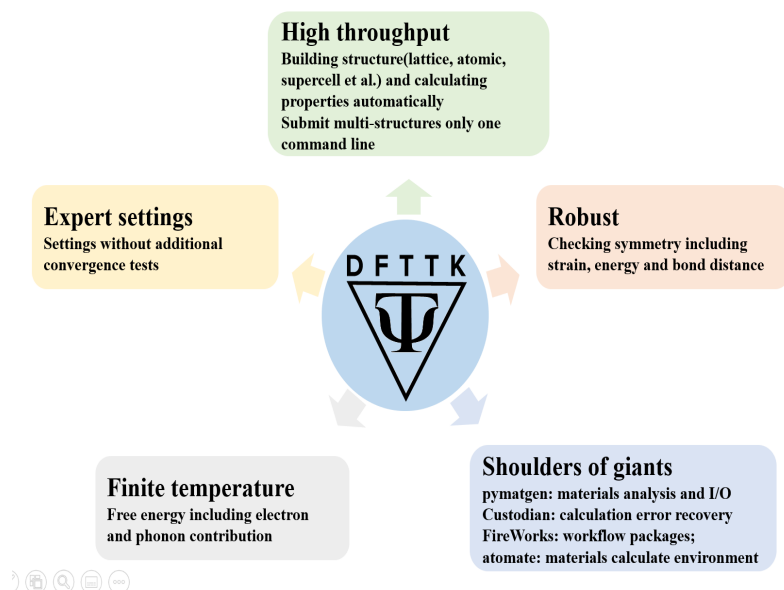
**Oct 14, 2021**



## CONTENTS:

<b>1</b>	<b>Project Goal</b>	<b>3</b>
<b>2</b>	<b>High-throughput calculations</b>	<b>5</b>
<b>3</b>	<b>The main functions of DFTTK</b>	<b>7</b>
<b>4</b>	<b>DFTTK features</b>	<b>9</b>
<b>5</b>	<b>The outline of flow chart of DFTTK</b>	<b>11</b>
<b>6</b>	<b>Robust relaxation scheme of DFTTK</b>	<b>13</b>
6.1	License . . . . .	14
6.2	Installation . . . . .	14
6.2.1	Config MongoDB . . . . .	15
6.2.2	Access MongoDB database from desktop . . . . .	15
6.2.3	YPHON . . . . .	16
6.3	Examples . . . . .	17
6.4	Quick guide to run DFTTK . . . . .	19
6.4.1	Submit dfttk DFT batch job . . . . .	19
6.4.2	Postprocess DFTTK results . . . . .	20
6.5	Configuration of DFTTK . . . . .	21
6.5.1	Content . . . . .	21
6.5.2	On batch job . . . . .	22
6.5.3	Preparation . . . . .	22
6.5.4	Configure all with one command . . . . .	24
6.5.5	Configure separately . . . . .	24
6.5.6	Example . . . . .	25
6.5.7	Validation for configuration . . . . .	26
6.5.8	Help for dfttk config command . . . . .	26
6.6	Settings file . . . . .	27
6.6.1	File name instruction . . . . .	27
6.6.2	Keywords in the file . . . . .	28
6.7	Get Started . . . . .	30
6.7.1	Content . . . . .	30
6.7.2	Run dfttk with dfttk run command . . . . .	31
6.7.3	Run dfttk with python script . . . . .	33
6.7.4	Help for dfttk run command . . . . .	34
6.8	Dfttk postprocessing mechanism . . . . .	34
6.8.1	Flow controls . . . . .	34
6.8.2	Scheme to find equilibrium volume . . . . .	35
6.8.3	Scheme to calculate derivative . . . . .	35

6.8.4	Scheme for LTC . . . . .	35
6.9	More about run DFTTK . . . . .	36
6.9.1	Examples . . . . .	36
6.9.2	Batch run dfttk . . . . .	37
6.9.3	Usage of run module . . . . .	37
6.9.4	Usage of thelec module . . . . .	38
6.9.5	Usage of thfind module . . . . .	39
6.10	Job monitoring . . . . .	41
6.11	Useful help from FireWork . . . . .	41
6.12	Troubleshooting . . . . .	41
6.12.1	Common troubleshooting . . . . .	42
6.12.2	pymatgen 2021 issue . . . . .	43
6.12.3	conda issues . . . . .	43
6.12.4	Troubleshooting Workflow . . . . .	43
6.12.5	Common Errors . . . . .	43
6.13	MongoDB VM hosting . . . . .	44
6.13.1	VM operation . . . . .	45
6.13.2	MongoDB installation/configuration . . . . .	45
6.13.3	MongoDB operation . . . . .	45
6.14	Changelog . . . . .	48
6.14.1	0.3.1 (2021-02-03) . . . . .	48
6.14.2	0.3.0 (2020-12-10) . . . . .	48
6.14.3	September 17, 2020 . . . . .	48
6.14.4	September 3, 2020 . . . . .	48
6.14.5	0.2.2 (2020-08-18) . . . . .	49
6.14.6	0.2 (2020-03-30) . . . . .	50
6.14.7	0.1 (2018-08-28) . . . . .	51
6.15	Contribution Guide . . . . .	51
6.15.1	Style Guidelines . . . . .	51
6.15.2	How to Contribute . . . . .	51
6.16	Releasing DFTTK . . . . .	52
6.16.1	Uploading to PyPI . . . . .	52
6.16.2	Some useful commands are following . . . . .	53
6.17	Recipes . . . . .	53
6.17.1	Construct a series of Debye workflows by substituting into a template structure . . . . .	53
6.17.2	ESPEI datasets from a QHA database . . . . .	55
6.18	FAQ . . . . .	57
6.18.1	What do the references to DFTTK-style or ESPEI-style configurations or occupancies mean? . . . . .	57
6.19	Acknowledgements . . . . .	58
6.20	Indices and tables . . . . .	59





## **PROJECT GOAL**

The goal of DFTTK is to make high-throughput first-principles calculations as simple as possible. The density functional theory (DFT) based software VASP is employed to perform first-principles calculations. In addition thermodynamic properties via the quasiharmonic approach, we proposed that any property, as long as it is dependent on the volume or strain, can be predicted using a quasi-static approach implemented by our group according to (i) the predicted property-volume/strain relationship from first-principles calculations directly and (ii) the volume/strain-temperature relationship of materials from the quasiharmonic approach.

For a given structure and elements, calculate all kinds of thermodynamic properties at finite temperature and pressure by first-principles approach based on density functional theory, including lattice vibration, thermal electron excitation, Seebeck coefficient, Lorenz number, effective charge carrier concentration etc.





## HIGH-THROUGHPUT CALCULATIONS

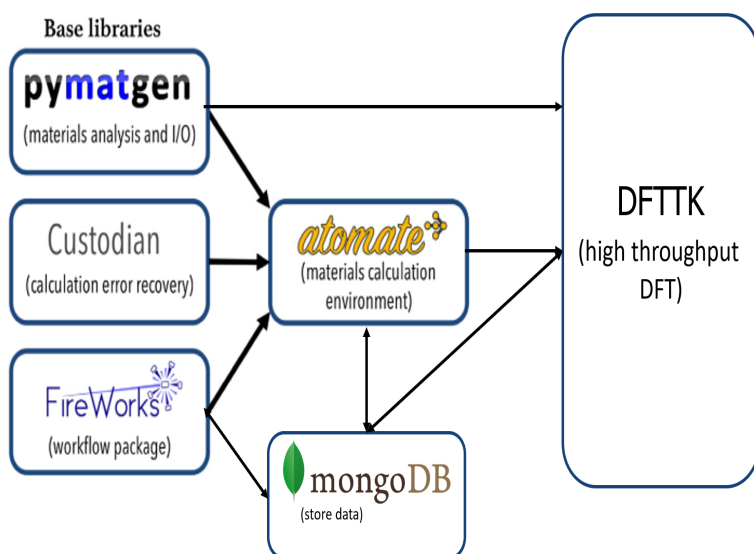
By its definition, the term of “first-principles” represents a philosophy that the prediction is to be based on a basic, fundamental proposition or assumption that cannot be deduced from any other proposition or assumption. This implies that the computational formulations are based on the most fundamental theory of quantum mechanics - Schrödinger equation or density functional theory (DFT) and the inputs to the calculations must be based on well-defined physical constants – the nuclear and electronic charges. In another word, once the atomic species of an assigned material are known, the theory should predict the energy of all possible crystalline structures, without invoking any phenomenological fitting parameters.

However, to perform DFT calculations in reality, it still needs the user to have extensive experiences on a variety of parameter choices and a lot of human handling on numerical or system exceptions. In the last decade, we have been working on solving the problem by integrating our experiences accumulated on high-throughout DFT calculations into a software package named as DFTTK (DFT based toolkits) and opened to the community (<https://www.dfttk.org>).



## THE MAIN FUNCTIONS OF DFTTK

- Structure maker by prototype and elemental substitution;
- Robust 0 K equilibrium volume optimization;
- Robust 0 K energy-volume curve optimization;
- Quasiharmonic phonon calculation;
- Born effective charge calculation;
- Elastic constant calculations.
- MongoDB database management
- Thermodynamic calculations and figure plots





## DFTTK FEATURES

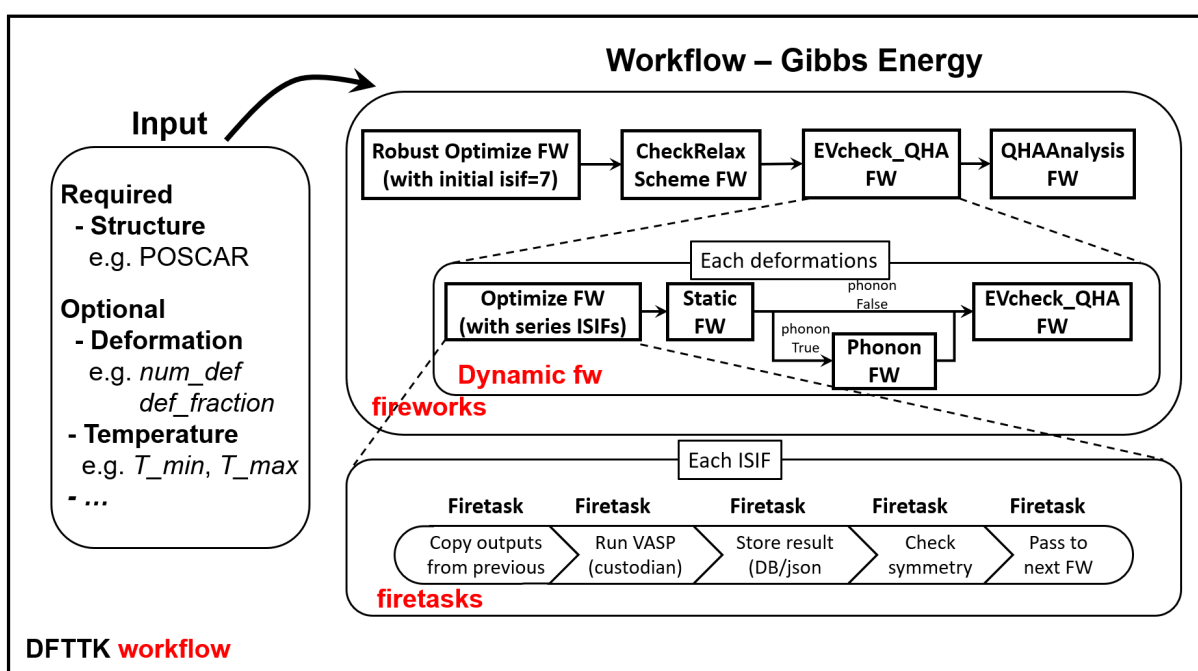
To perform DFT calculation using DFTTK, the user only needs to name the structure file called POSCAR by VASP, either prepared by user or produced by DFTTK by elemental substitution on given prototype. DFTTK is developed on [atomate](#) from the [Materials Project](#) which is built on three open-source Python libraries. The main benefits of [atomate](#) are its flexibility and data management platform, in particular the numerical convergence control and computational exception handling. DFTTK is able to predict properties at finite temperatures by phonon or Debye model for both stoichiometric and solution phases, featured by:

- High-throughput DFT calculation and postprocess;
- Postprocess plenty of data stored in MongoDB with one simple command;
- Compatible with Yphon package and phonopy;
- Can recover data from certain fizzled calculations;
- Can account thermal electron contribution to thermodynamic properties;
- Can calculate thermodynamic properties at 0 K and a few tenth K;
- Can perform doping calculations for semiconductors or thermoelectric materials under rigid band approximation;
- Can account the effect of thermal expansion/temperature on Seebeck coefficient, Lorenz number, thermal carrier concentrations;
- Automatic plot figures for more than 20 thermodynamic properties in the publishable resolution, including atomic volume, free energy, entropy, enthalpy, linear thermal expansion coefficient, isobaric specific heat, constant volume specific heat, lattice only specific heat, bulk modulus, Debye temperature, Seebeck coefficient, Lorenz number, absolute thermal electric force, etc.



## THE OUTLINE OF FLOW CHART OF DFTTK

Provide robust density functional theory workflows for calculating thermodynamic properties in temperature and composition space. The work flow control of DFTTK is given by the following blue print.







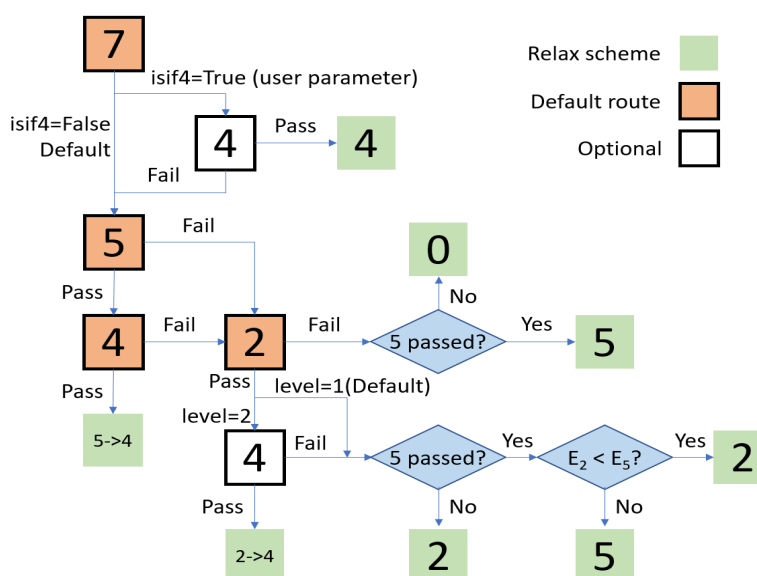
## ROBUST RELAXATION SCHEME OF DFTTK

DFTTK makes it automatic to going through these ISIF options by the workflow shown in the figure in the right

The purpose of DFTTK robust relaxation is to find the lowest energy structure within a given threshold of a given symmetry. This information is especially useful in the case, for example

1. Random solution phase;
2. High temperature phase which is unstable in low temperature;
3. The calculation of energetics for endnumber for CALPHAD modeling.

The solution is to make the best use of **ISIF** control parameter in VASP, which controls:



ISIF	calc. force	re-lax stress tensor	re-lax ions	change cell shape	change cell volume
0	yes	no	yes	no	no
1	yes	trace only	yes	no	no
2	yes	yes	yes	no	no
3	yes	yes	yes	yes	yes
4	yes	yes	yes	yes	no
5	yes	yes	no	yes	no
6	yes	yes	no	yes	yes
7	yes	yes	no	no	yes

## 6.1 License

Copyright (c) 2017 Phases Research Lab

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The MIT License (MIT)

## 6.2 Installation

It is recommended to install DFTTK under the [anaconda](#) environment. Under the linux command prompt (or anaconda powershell prompt for Windows), one can create a preferred directory and then run

- Release version

```
pip install dfttk
```

- Development version

```
git clone https://github.com/PhasesResearchLab/dfttk.git
cd dfttk
pip install -e .
```

- Alpha interanal daily test version

```
git clone https://github.com/yiwang62/dfttk.git
cd dfttk
pip install -e .
```

to get the laste updates for the Alpha version based on one’s current version, run

```
git fetch origin
git checkout 20210405 #20210405 is a branch name which may be change in a time maner
```

mkdir a folder named config wherever you want to followed by copy the file db.json, my\_launchpad.yaml from your MongoDB manager into config/. See the section Config MongoDB

```
dfttk config -all --nodes 1 --ppn 16 --pmem 32gb -aci -M yourcomputer -qt yourbatch -
↪mapi PMG_MAPI_KEY
```

where

vasp\_psp is a place holding your vasp pseudopotentials  
 yourcomputer is your computer name, such as aci-rour, cori-knl, cori-ksw, bridges2, stampede2  
 yourbatch can be pbs, slurm  
 PMG\_MAPI\_KEY can be obtained by: Go to the materials project website, <https://materialsproject.org/>,  
 under the API section, you will easily find you API Keys number.  
 finally, you need to change the account number and queue/partition number in the  
 config/my\_qadapter.yaml file

### 6.2.1 Config MongoDB

DFTTK needs MongoDB to manage DFT calculations and outputs. The users of DFTTK can either buy the commercial MongoDB database service or set up their own MongoDB server.

Ask the MongoDB system manager for two json files: one named db.json and another named my\_launchpad.yaml and save them in a config folder wherever you choose.

db.json used by FireWorks through MongoDB to access the DFTTK output results, templated as follows.

```
{
  "database": "userid-results",
  "collection": "tasks",
  "admin_user": "userid",
  "admin_password": "pass1",
  "readonly_user": "userid-ro",
  "readonly_password": "pass2",
  "host": "146.186.149.69",
  "port": 27018,
  "aliases": {}
}
```

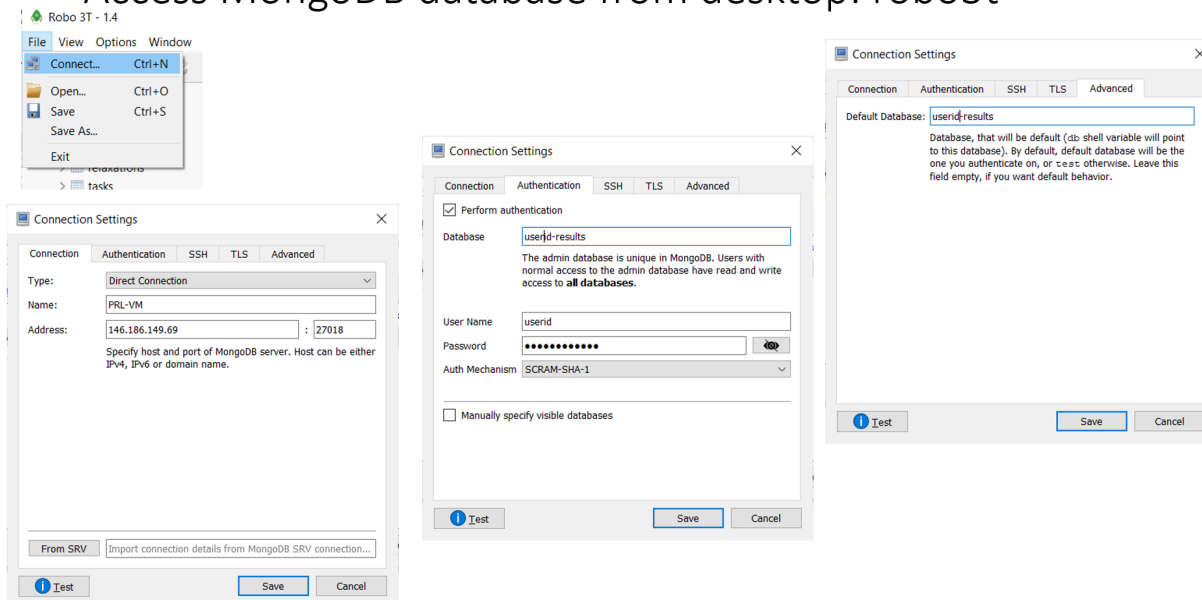
my\_launchpad.yaml used by FireWorks through MongoDB for DFT job managements, templated as follows.

```
host: 146.186.149.69
name: userid-fws
password: pass3
port: 27018
ssl_ca_file: null
strm_lvl: INFO
user_indices: []
username: userid
wf_user_indices: []
```

### 6.2.2 Access MongoDB database from desktop

One can install robo3T from <https://robomongo.org/>. One needs to use the information from the db.json file to setup robo3T connection as indicated below

## Access MongoDB database from desktop: robo3t



#note

1. PSU-VM is a label of your choice to remark the MongoDB connection;
2. 146.186.149.69 is the ip address of the MongoDB server;
3. One needs to replace userid with the one (i.e. the value field of admin\_user) from the db.json file together with the the value field of admin\_password

### 6.2.3 YPHON

To postprocess the finite properties, the Yphon package is required. Yphon can be installed by run

```
cd ~
git clone https://github.com/yiwang62/YphonPackage
#Note: Usually the precompiled binaries should be executable in the common Linux/Unix_
environment. If not, do the following:
```

```
cd YphonPackage/YPHON/YPHON
make
#Note: If errors reported in the compiling stage, insert one line #define R_OK 1 after
#include
```

For csh user: the command search path should be changed by inserting line below into the .cshrc (.tcshrc) file

```
set path = ( . ~/YphonPackage/YPHON/YPHON $BIN_PATH $path)
```

For bsh user: the command search path should be changed by inserting the lines below into the .bash\_profile (.bashrc) file

```
PATH=.:~/YphonPackage/YPHON/YPHON:$BIN_PATH:$PATH
export PATH
```

## 6.3 Examples

The Examples folder is designed to keep the data for user to test the DFTTK package. The two subfolder are:

```
- dir ``Al/`` - contains all data for test run for Al
- dir ``ZrSiO4/`` - contains all data for test run for ZrSiO4
- dir ``ExptData/`` - contains a json file ``ExptData.json`` which saves the
  ↳ experimental thermodynamic data for a collection of materials.
```

Within the Al or the ZrSiO4 subfolder, one see two subfolders

- dir input/ - contain input setup files using Al as the example on E-V, phonon, and thermodynamic property calculations
- Al\_Fm-3m\_225PBE/ - contain outputs by postprocessing data that saved in MongoDB by the above Al example.

For the data within Al\_Fm-3m\_225PBE/

- dir Yphon/ - all data input/output for Yphon, e.x., hessian matrix (superfij.out), calculated phonon dos
- dir figures/ - plots in png format for most of the thermodynamic properties
- file readme - extensive summary of the calculated results in json format
- file fvib\_ele - tablated data containing the calculated thermodynamic properties
- file fvib\_eij - tablated data containing the calculated thermal expansion coefficient tensor
- file record.json - SGTE fitting record for heat capacity, Gibbs energy, enthalpy, and entropy at given temperature range

To run the Example for the VASP calculation, say Al, run

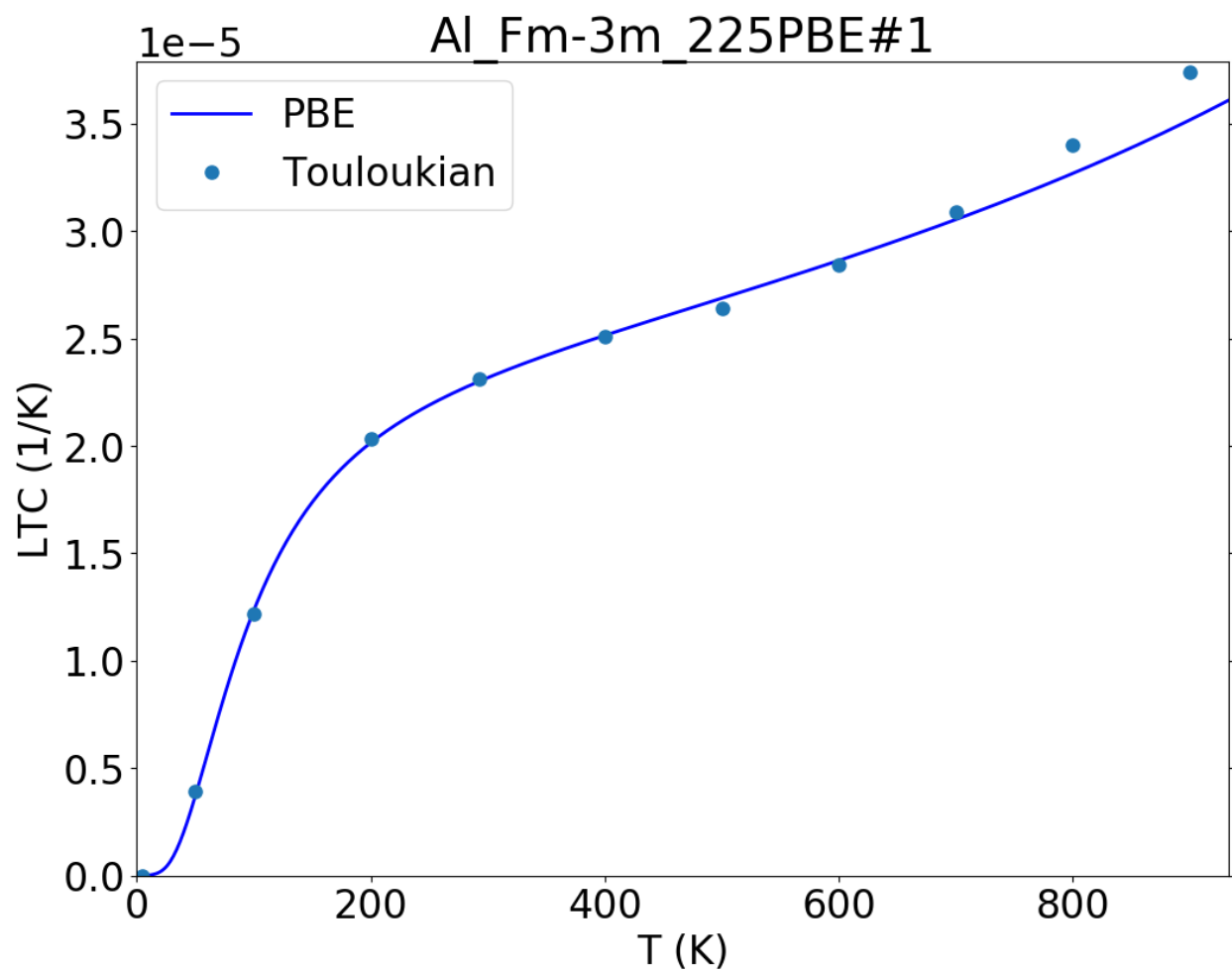
```
cd Al/input
dfttk run -wf robust -f POSCAR.Al -l -m 1
```

To postprocess calculations after the VASP calculation done, run

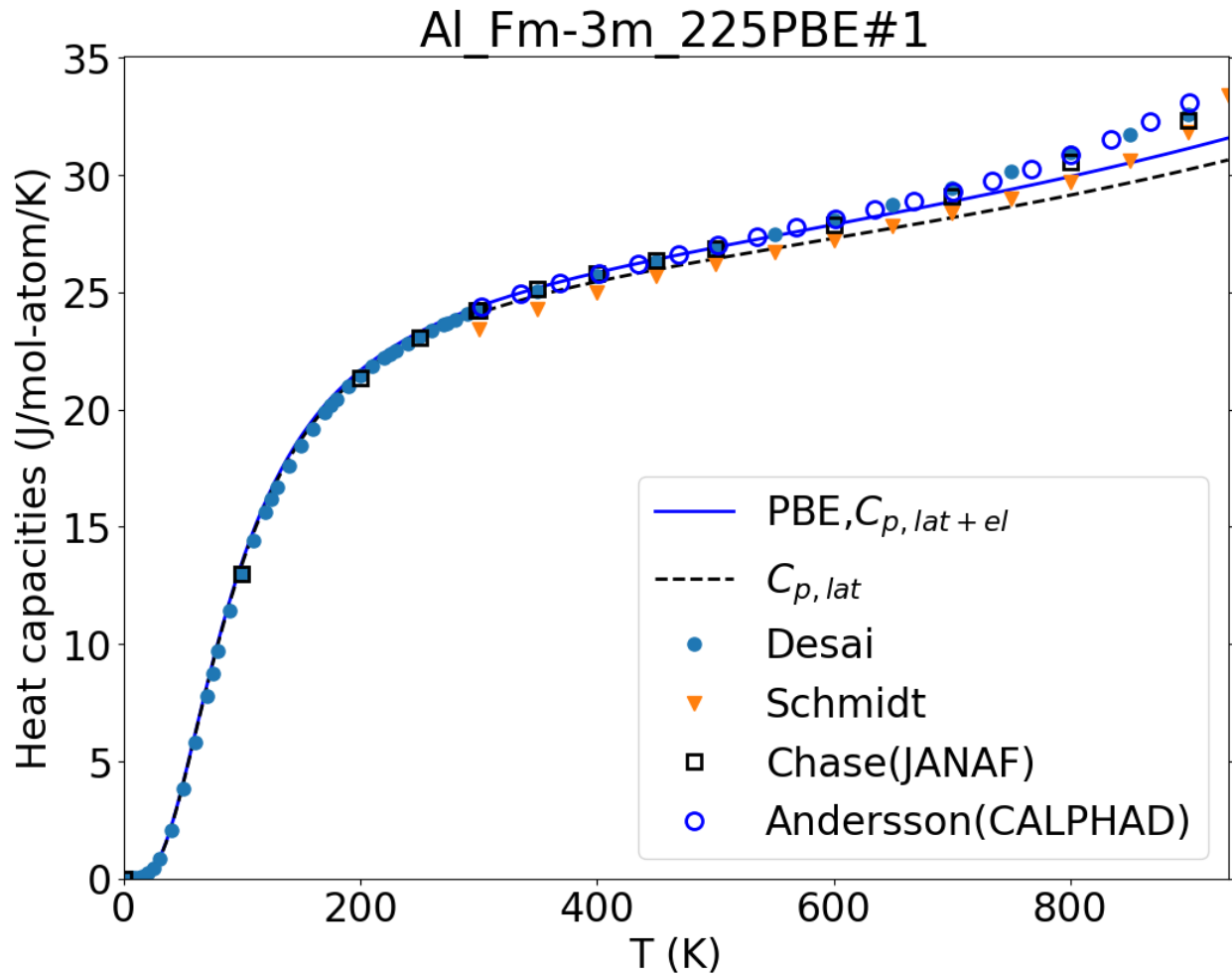
```
cd Al
dfttk thfind -py -td -50 -plot find_or_DFT -eq 4 -smooth -el 1 -renew -get -metatag_
  ↳ 0c1887fa-0cb4-4ce2-9559-7f7909ffa11a -expt ../ExptData/ExptData.json
#note that the key ``0c1887fa-0cb4-4ce2-9559-7f7909ffa11a`` is obtained from the file_
  ↳ ``input/METADATAS.yaml`` automatically produced by the VASP calculation step.
```

The above will produce more than 20 figures stored in the folder “Al\_Fm-3m\_225PBE/figures/” and they can be view them using the linux command display to show the figure, for example

```
display Al_Fm-3m_225PBE/figures/LTC.png #to see the linear thermal expansion coefficient
```



```
display Al_Fm-3m_225PBE/figures/Heat_capacities.png #to see the heat capaticity, and so  
↪ on
```



## 6.4 Quick guide to run DFTTK

### 6.4.1 Submit dfttk DFT batch job

The DFT calculations are managed by the run modules by the command `dfttk run`

- Submit a single calculation

Go to any folder under ytests folder say `cd dfttk/yttests/ex55`, then run

```
dfttk run -f POSCAR.Al -l -m 1
```

## 6.4.2 Postprocess DFTTK results

To postprocess the DFTTK results and get thermodynamic properties.

- Single postprocess

```
dfttk thelec -renew -plot find_or_DFT -metatag metatag.
```

where metatag is the metadata tag automatically produced after one submits the dfttk job, the argument `-renew` instruct thelec to redo the calculations even already done previously, and the arguments `-plot find_or_DFT` instruct thelec to find the label for theoretical curve from the MongoDB database (in the above case, it found PBE) and use DFT if not found. In the above example, one can find out metatag in the value field of tag by `cat dfttk/ytasks/ex55/METADATA.yaml`

The run of the above command will cost a couple of minutes. After the command done, one will see a folder with name like `compound_xxx-xxx` (in the above example, the folder name will be `Al_Fm-3m_225PBE` within which the fcontents are:

folder `figures` - plots in png format for most of the thermodynamic properties;

file `readme` - contain key information in json format;

file `fvib_ele` - text table contains the thermodynamic properties

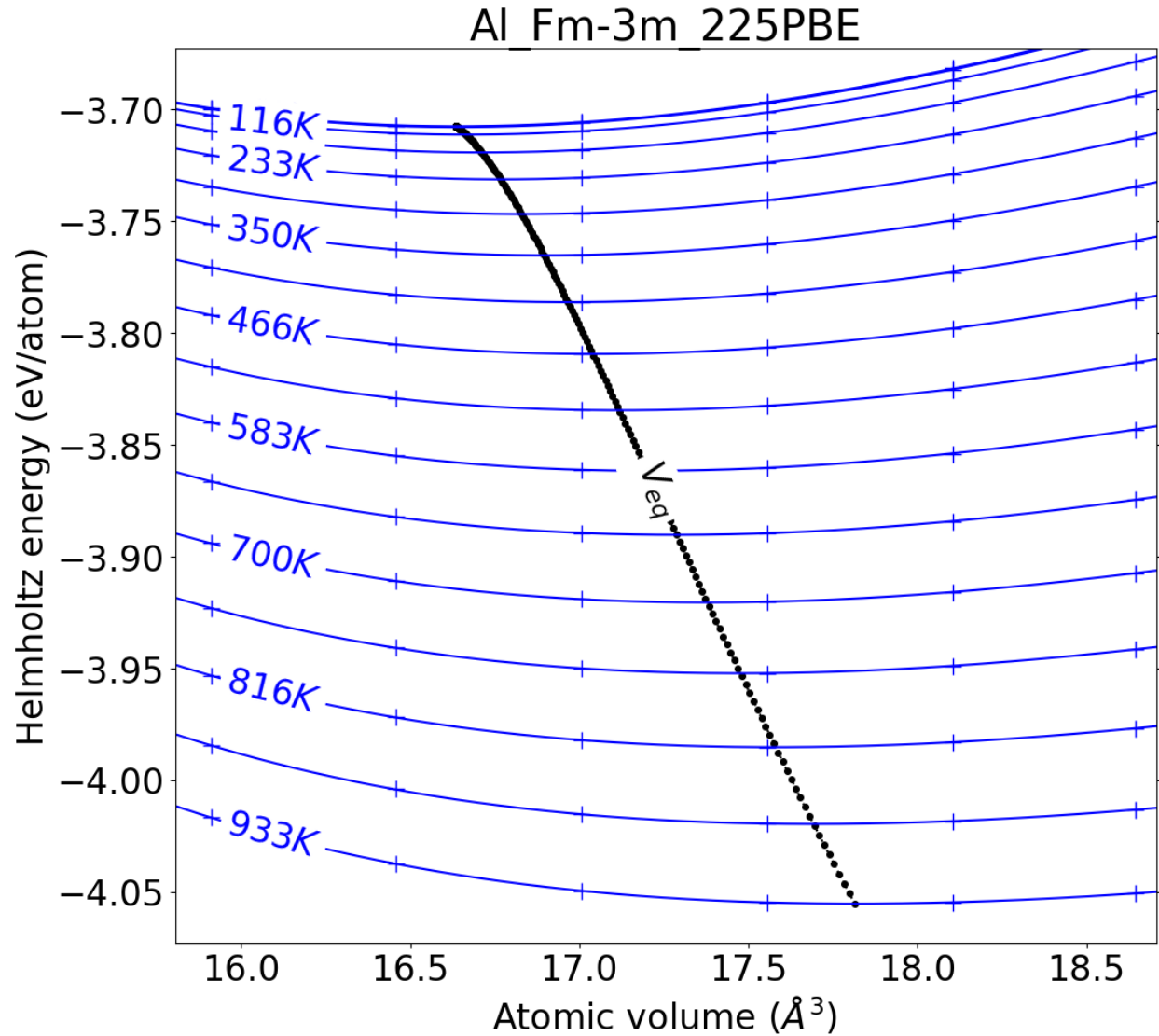
file `record.json` - json file contains Shomate Equation (SGTE) fitting of Cp, G-H298.15, H-H298.15, and S; and

folder `Yphon` - extracted data to run Yphon for further analysis

The figures stored in the figures folder can viewed by the command `display`, e.x, to see the evolution of Helmholtz energy as functions of volume and temperature

```
display Al_Fm-3m_225PBE/figures/Helmholtz_energy.png
```





## 6.5 Configuration of DFTTK

### 6.5.1 Content

- *On batch job*
- *Preparation*
- *Configure all with one command*
- *Configure separately*
  - *Configuration for atomate*
  - *Configuration for pymatgen*
- *Example*
- *Validation for configuration*

- *Help for dfttk config command*
- 

## 6.5.2 On batch job

- [SLURM sbatch instruction](#)
- [SLURM sbatch example](#)

*TO TOP*

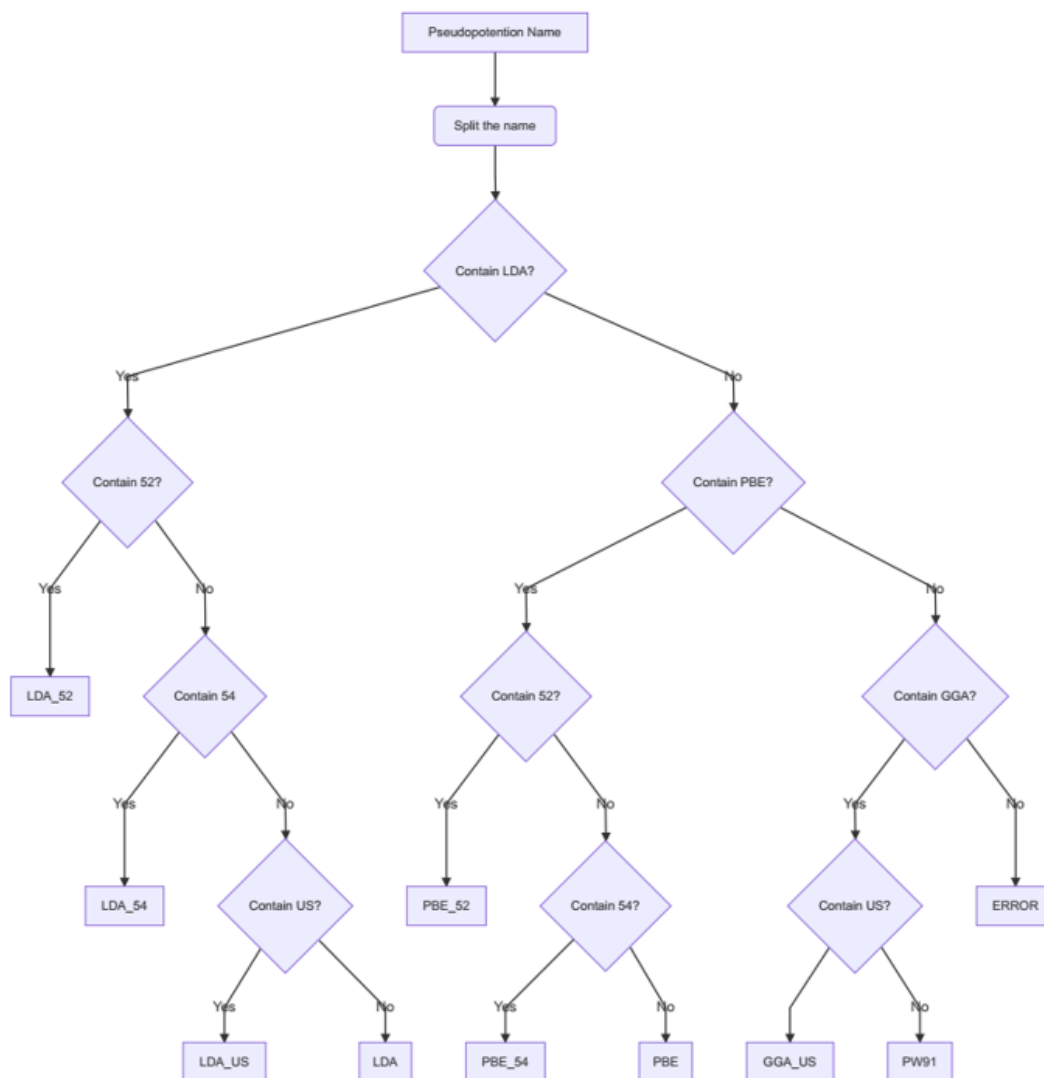
## 6.5.3 Preparation

For configure dfttk, the following file you need to prepare.

```
current_folder
psp                                [specified by -psp]
pseudopotential_content [required if you didnot configurate pymatgen]
...
config                            [specified by -c parameter]
db.json                           [required]
my_launchpad.yaml                 [required]
FW_config.yaml                   [optional]
my_fworker.yaml                  [optional]
my_qadapter.yaml                  [optional]
vaspjob.pbs                       [optional, specified by -q parameter]
```

After prepared the above **required** file, simply run `dfttk config -all` to finish the configuration. If such command successes, you can skip **all** of the reset part of this documents. If not, please reading the following parts.

- **pseudopotential\_content:** The vasp pseudopotential.
  - **PRL Group Notes:** For PRL group, if you use ACI at PSU, you can skip this part, using `-aci` parameter
  - It can be compressed file (`*.tar.gz`) or uncompressed folder. The name is handled as follows:



Parse

PSP Name

**Note: 1.** The split can recognize `.`, `-`, `_`, `+`, `=`, `*` and space .

**\*\*2.\*\*** After split, the name will be a list. The condition e.g. ``contain 52`` is `↪` the elements' level. It means ``52`` should be a elements of the list. But ``US`` `↪` is string level, which means ``US`` only need be a sub-string of the elements `↪` of the list.

**\*\*3.\*\*** For compressed file, it support ``*.tar.gz`` and ``*.tar``

e.g. ``potpaw_PBE``, ``PBE.tar.gz`` and ``potpaw_PBE_5254`` will be recognized as `↪` ``PBE``.

``potpaw_PBE.52`` and ``potpaw_PBE_52`` will be recognized as ``PBE_52``

``potUSPP_LDA`` and ``POT_LDA_US`` will be recognized as ``LDA_US``

After prepare, the file structure should look like as follows:

The original pseudopotential file, please ask those who are in charge of vasp in your group.

- **MAPI\_KEY:** The API key of [Materials Project](#), ref. [API KEY](#)
- **vaspjob.pbs:** The submitting script of your queue system. **Currently, only pbs system is supported**
- **other config files:** At least, you need prepare two files, `db.json` and `my_launchpad.yaml`. The template is shown in the `config` folder.
  - For more details, please ref. [Configure database connections and computing center parameters](#)
- **PRL GROUP NOTES:**
  - For `db.json` and `my_launchpad.yaml` file, please ask **Brandon** for help.
  - `vaspjob.pbs` for ACI can be download from github
  - For **pseudopotential**, two choices: 1. using the pseudopotentials in ACI account (Using `-aci` parameter). 2. Download the pseudopotential from group's box(Group Documents/VASP\_potential note: only PBE\_54 and LDA\_54).

*TO TOP*

---

### 6.5.4 Configure all with one command

- `dfttk config -all`
- For ACI at PSU user, please use `dfttk config -all -aci`

```
usage: dfttk config [-h] [-all] [-p PATH_TO_STORE_CONFIG] [-a]
                  [-c CONFIG_FOLDER] [-q QUEUE_SCRIPT] [-qt QUEUE_TYPE]
                  [-v VASP_CMD_FLAG] [-mp] [-aci] [-psp VASP_PSP_DIR]
                  [-mapl MAPI_KEY] [-df DEFAULT_FUNCTIONAL]
```

The meaning of the parameters, please ref *Help for dfttk config command* or just run `dfttk config -h`

*TO TOP*

---

### 6.5.5 Configure separately

#### Configuration for atomate

- Config manual, ref [Configure database connections and computing center parameters](#)
- `dfttk config -a`

```
usage: dfttk config -a [-p PATH_TO_STORE_CONFIG] [-c CONFIG_FOLDER] [-q QUEUE_
↪SCRIPT]
                  [-qt QUEUE_TYPE] [-v VASP_CMD_FLAG]
```

The meaning of the parameters, please ref *Help for dfttk config command* or just run `dfttk config -h`

*TO TOP*

---

## Configuration for pymatgen

- Config manual, ref [POTCAR setup in pymatgen](#)
- `dfttk config -mp`

```
usage: dfttk config -mp [-aci] [-p PATH_TO_STORE_CONFIG] [-psp VASP_PSP_DIR] [-mapi ↵
↵ MAPI_KEY]
                        [-df DEFAULT_FUNCTIONAL]
```

The meaning of the parameters, please ref *Help for dfttk config command* or just run `dfttk config -h`  
[TO TOP](#)

### 6.5.6 Example

```
dfttk config -all -p test_config -mapi test_api
```

In test\_config folder

```
test_config
config
  db.json
  FW_config.yaml
  my_fworker.yaml
  my_launchpad.yaml
  my_qadapter.yaml
logs
vasp_psp
  POT_GGA_PAW_PBE_54
  Ac
  ...
  POT_LDA_PAW_54
  Ac
  ...
```

In ~/.bashrc

```
export FW_CONFIG_FILE=/gpfs/scratch/mjl6505/test/dfttk/tutorial/config/test_config/
↵ config/FW_config.yaml
```

in ~/.pmgrc.yaml

```
PMG_DEFAULT_FUNCTIONAL: PBE
PMG_MAPI_KEY: test_api
PMG_VASP_PSP_DIR: /gpfs/scratch/mjl6505/test/dfttk/tutorial/config/test_config/vasp_psp
```

[TO TOP](#)

### 6.5.7 Validation for configuration

- `dfttk config -t`

This command will validate the configuration, and give some error or tips for the incorrect configuration

```
dfttk config -t [{all,pymatgen,atomate,none}]
```

- Currently only support validation for the configuration of pymatgen

*TO TOP*

### 6.5.8 Help for dfttk config command

```
dfttk config -h
```

```
DFTTK version: 0.1+121.g8fddda3.dirty
Copyright Phases Research Lab (https://www.phaseslab.com/)

usage: dfttk config [-h] [-all] [-p PATH_TO_STORE_CONFIG] [-a]
                  [-c CONFIG_FOLDER] [-q QUEUE_SCRIPT] [-qt QUEUE_TYPE]
                  [-v VASP_CMD_FLAG] [-mp] [-aci] [-psp VASP_PSP_DIR]
                  [-mapi MAPI_KEY]
                  [-df {LDA,LDA_52,LDA_54,LDA_US,PBE,PBE_52,PBE_54,PW91,PW91_US,Perdew-
↪Zunger81}]
                  [-t [{all,pymatgen,atomate}]]

optional arguments:
  -h, --help            show this help message and exit
  -all, --all           Configure atomate and pymatgen.
  -p PATH_TO_STORE_CONFIG, --prefix PATH_TO_STORE_CONFIG
                        The folder to store the config files. Default: .
                        (current folder)
  -a, --atomate         Configure atomate.
  -c CONFIG_FOLDER, --config_folder CONFIG_FOLDER
                        The folder containing config files, at least contain
                        db.json and my_launchpad.yaml. Default: '.'
  -q QUEUE_SCRIPT, --queue_script QUEUE_SCRIPT
                        The filename of the script for submitting vasp job. It
                        will search in current folder and sub-folders.
                        Default: vaspjob.pbs
  -qt QUEUE_TYPE, --queue_type QUEUE_TYPE
                        The type of queue system. Note, only pbs is supported
                        now. Default: pbs
  -v VASP_CMD_FLAG, --vasp_cmd_flg VASP_CMD_FLAG
                        The flag to distinguish vasp_cmd to othe commands in
                        queue_script. Default: vasp_std
  -mp, --pymatgen       Configure pymatgen.
  -aci, --aci           Using the pseudopotential on the ACI cluster at PSU.
  -psp VASP_PSP_DIR, --vasp_psp_dir VASP_PSP_DIR
                        The path of pseudopotentials. Default: psp
```

(continues on next page)

(continued from previous page)

```

-mapi MAPI_KEY, --mapi_key MAPI_KEY
    The API key of Materials Projects
-df {LDA,LDA_52,LDA_54,LDA_US,PBE,PBE_52,PBE_54,PW91,PW91_US,Perdew-Zunger81}, --
↪ default_functional {LDA,LDA_52,LDA_54,LDA_US,PBE,PBE_52,PBE_54,PW91,PW91_US,Perdew-
↪ Zunger81}
    The default functional. Default: PBE
-t [{all,pymatgen,atomate}], --test_config [{all,pymatgen,atomate}]
    Test for configurations. Note: currently only support
    for pymatgen.

```

## 6.6 Settings file

The user can change the settings for dfttk calculations by providing a settings file.

### 6.6.1 File name instruction

#### Globale settings

By default, the global settings file is named as `SETTINGS.yaml` or `SETTINGS.json`.

The user can change the name by `-s` parameter in `dfttk run`. e.g. if the user run dfttk by `dfttk run -s SET`, then the files named with `SET.yaml` or `SET.json` is the global settings file.

#### Individual settings

The user can provide individual settings for some structures whoes settings is different with others. The individual settings file should be named with `SETTINGS-FILENAME_WITH_EXT.yaml(json)` or `FILENAME_WITH_EXT-SETTINGS.yaml(json)`

#### Note:

- `FILENAME_WITH_EXT` is the file name of structure with extension.
- `SETTINGS` is case insensitive, both `settings` and `SETTINGS` are OK.

e.g. In the working folder, there are some structure files as follows:

```

Fe3Ni.cif
POSCAR
POSCAR-2

```

Then the following files will be recognized as settings files.

```

SETTINGS-Fe3Ni.cif.yaml
POSCAR-SETTINGS.yaml
settings-POSCAR-2.yaml

```

## **6.6.2 Keywords in the file**

### **Default settings**

The settings file is optional, if the user does not provide the settings file, the default value will be used.

- Common settings



Keywords	Default Value	Comments
magmom	No	The MAGMOM for each atom, e.g. [4, 4, -4, 4]. <i>Note</i> , the length must agree with the number of atoms
metadata	No	The metadata of the calculation. If the user provides it, DFTTK will find the existing calculations in the database. If not, it will generate by uuid4.
isif4	False	If run ISIF=4 following ISIF=7 in RobustOptimizeFW
level	1	Optimize level. If run ISIF=2 after last ISIF=4 in RobustOptimizeFW 1 for run and 2 for not
override_symmetry_tolerances	None	Override the default symmetry tolerance, if None, { 'tol_strain':0.05,'tol_energy':0.025, 'tol_bond':0.10 }
override_default_vasp_params	{ }	Override the default vasp settings The optional keys is 'user_incar_settings', 'user_kpoints_settings', 'user_potcar_functional'. For more details, ref. <a href="https://pymatgen.org/pymatgen.io.vasp.sets.html">https://pymatgen.org/pymatgen.io.vasp.sets.html</a>
modify_incar_params	{ }	Modify the incar settings in the fireworks level. e.g. modify_incar_params = { 'Full relax': { 'incar_update': { "LCHARG":False } }, 'static': { 'incar_update': "LAECHG":True } }
modify_kpoints_params	{ }	Modify the kpoints settings in the fireworks level.
store_volumetric_data	False	Store the volumetric data (True) or not (False)
verbose	False	print(True) or not(False) some informations, for debug
passinitrun	False	Pass init vasp result. <b>It will be dropped in the future</b>
run_isif2	False	If run ISIF=2 before ISIF=4 (True) or not (False). <b>It will be dropped in the future</b>
pass_isif4	False	Whether pass isif=4 calculation. <b>It will be dropped in the future</b>
symmetry_tolerance	0.05	The tolerannce for symmetry. <b>It will be dropped in the future</b>
relax_path	''	The path of relaxiation. <b>It will be dropped in the future</b>

- Phonon settings

Keywords	De- fault Value	Comments
phonon	False	Run phonon (True) or not(False, Debye model)
phonon_supercell_matrix	atoms	The supercell matrix for phonon calculations. It can take the following values: <b>Matrix</b> , e.g. [[2, 0, 0], [0, 2, 0], [0, 0, 2]] <b>String</b> : atom/lattice/volume(the first letter works) Determining the supercell matrix automatically by atoms/lattice/volume ranges
phonon_supercell_matrix_lower	60	The lower boundary for phonon_supercell_matrix(String)
phonon_supercell_matrix_upper	60	The upper boundary for phonon_supercell_matrix(String)
force_phonon	False	Force run phonon (True) or not(False), No matter ISIF=4/stable_tor pass or not
stable_tor	0.01	Stable tolerance (The percentage of negative dos), If the negative part of DOS is larger than this value, DFTTK won't run phonon for this structure.

- QHA settings

Keywords	Default Value	Comments
num_deformations	7	The number of deformations/structures
deformation_fraction	[-0.1, 0.1]	The range of deformation, 0.1 means 10%
eos_tolerance	0.01	The tolerance for eos fitting. If larger than this value, DFTTK will append volumes automatically
t_min	5	The minimum of temperature in QHA process
t_max	2000	The maximum of temperature in QHA process
t_step	5	The step of temperature in QHA process
volume_spacing_min	0.03	Minimum ratio of Volumes spacing. This keyword will be dropped in the future

- Elastic settings

Keywords	Default Value	Comments
strain_states	None	Strain modes, if it is None, it will generated by atomate
stencils	None	The amplitude of the strain modes/states
analysis	True	Analysis (True) or not (False)
sym_reduce	False	Reduce the strain according to the symmetry or not
order	2	The order of the elastic constants
conventional	False	Convert the structure into conventional format or not

## 6.7 Get Started

### 6.7.1 Content

- Run dfttk with `dfttk run` command
  - Run dfttk for a single structure or a single folder
  - Write out the workflow
  - Run dfttk for a folder and its sub-folder
  - Run dfttk with individual settings
  - Submit job to launchpad and queue system

- Run dfttk with python script
- Help for dfttk run command

## 6.7.2 Run dfttk with dfttk run command

```
usage: dfttk run [-h] [-f STRUCTURE_FOLDER] [-mh MATCH_PATTERN] [-s SETTINGS]
               [-r] [-wf WORKFLOW] [-l] [-m [MAX_JOB]] [-o]
```

The help of dfttk run command, please ref. *Help for dfttk run command* or simply run `dfttk run -h`

All the examples is run under current folder (get\_started)

```
get_started
NaCl.cif
NotSupportFile
POSCAR
test_folder
  Fe3Ni.cif
  Fe3Ni-settings.yaml
  POSCAR
  POSCAR-2
  SETTINGS-POSCAR.yaml
  SETTINGS.yaml
sub_folder
  POSCAR
```

### Run dfttk for a single structure or a single folder

-f parameter, specify the structure or folder containing structure

```
dfttk run -f STRUCTURE/STRUCTURE_FOLDER
```

- When the parameter value of -f is a file it run this file; if it is a folder, it will search all supported files in the specified folder.
- It support **POSCAR**, **CONTCAR**, **CHGCAR**, **LOCPOT**, **cif**, **vasprun.xml**, **pymatgens structures**, **mcsqs's structure**, more details, ref [pymatgen.core.structure.IStructure.from\\_file](#)
  - For **POSCAR** or **CONTCAR**, the name should be **\*POSCAR\*** or **\*CONTCAR\*** or **\*.vasp**
  - For **CHGCAR** or **LOCPOT**, the name should be **\*CHGCAR\*** or **\*LOCPOT\***
  - For **cif**, the name should be **\*.cif\*** or **\*.mcif\***
  - For **vasprun.xml**, the name should be **vasprun\*.xml\***
  - For **pymatgen's structure**, the name should be **\*.json** or **\*.yaml**
  - For **atat's structure**, the name should be **\*rndstr.in\*** or **\*lat.in\*** or **\*bestsq\***

TO TOP

## Write out the workflow

-o parameter, write out work flow

- When add -o parameter, the work flow for every structure will be write out
- The file name of the work flow is dfttk\_wf- + filename of the structure + .yaml (e.g. dfttk\_wf-POSCAR.yaml)

```
dfttk run -f POSCAR -o
```

It will write out the work flow in current folder as dfttk\_wf-POSCAR.yaml

- The file will write out in the same folder with corresponding structure.

```
dfttk run -f ./test_folder/POSCAR -o
```

It will write out the work flow as ./test\_folder/dfttk\_wf-POSCAR.yaml

*TO TOP*

## Run dfttk for a folder and its sub-folder

-r parameter, recursive the folder

-mh parameter, specify the match pattern

- Add -r parameter, it will searching the folder (specified by -f parameter) and its sub-folder

```
dfttk run -f . -r
```

It will run the following structure

```
get_started/NaCl.cif
get_started/POSCAR
get_started/test_folder/Fe3Ni.cif
get_started/test_folder/POSCAR
get_started/test_folder/POSCAR-2
get_started/test_folder/sub_folder/POSCAR
```

- Add -mh parameter, it will filter the filename by the value of -mh, the pattern should be placed in quotes (" or """)

```
dfttk run -r -mh '*.cif'
```

It will run the following structure

```
get_started/NaCl.cif
get_started/test_folder/Fe3Ni.cif
```

*TO TOP*

## Run dfttk with individual settings

-s parameter, specify the name for settings, default: SETTINGS

- SETTINGS.yaml or SETTINGS.json: The global settings for current folder
- SETTINGS-\*.yaml(json) or \*-SETTINGS.yaml(json): The individual settings, \* is the name of the structure (without ext)
- Case insensitive. (both SETTINGS and settings are OK)
- The value of -s parameter will replace SETTINGS.
- Examples:

```
dfttk run -f test_folder -s set
```

It will take ./test\_folder/Fe3Ni-SET.yaml as setting file, and ./test\_folder/Fe3Ni-settings.yaml will not taken as setting file.

*TO TOP*

## Submit job to launchpad and queue system

-l parameter, launch to launchpad

-m parameter, launch to queue system and determine the number of jobs

- Add -l parameter, it will submit the job to launchpad
- Add -m N parameter, it will submit the job to queue system. N specifies the number of jobs run at the same time. (Note: This only work when -l parameter is added.)

```
dfttk run -l  
dfttk run -l -m 1  
dfttk run -l -m 2
```

- When -m 1, it will run `qlaunch singleshoot` (fireworks command)
- When -m N (N>1), it will run `qlaunch rapidfire -m N` (fireworks command)
- For more details, please ref. [Fireworks: Launch Rockets through a queue](#)

*TO TOP*

## 6.7.3 Run dfttk with python script

*TO TOP*

---

## 6.7.4 Help for dfttk run command

```
dfttk run -h
```

```
DFTTK version: 0.1+98.ge7aa39c.dirty
Copyright Phases Research Lab (https://www.phaseslab.com/)

usage: dfttk run [-h] [-f STRUCTURE_FOLDER] [-mh MATCH_PATTERN] [-s SETTINGS]
               [-r] [-wf WORKFLOW] [-l] [-m [MAX_JOB]] [-o]

optional arguments:
  -h, --help                show this help message and exit
  -f STRUCTURE_FOLDER, --structure_folder STRUCTURE_FOLDER
                           The folder/file containing the structure, Default: '.'
  -mh MATCH_PATTERN, --match_pattern MATCH_PATTERN
                           The match pattern for structure file, e.g. *POSCAR*,
                           Default: * -- everything except SETTING files, ref. -s
  -s SETTINGS, --setting SETTINGS
                           Specify the name of SETTINGS files (yaml or json file)
                           Default: SETTINGS (case insensitive and without ext)
                           The following filename will be treat as SETTINGS file
                           SETTINGS (global settings in the folder) Start with
                           SETTINGS- (individual settings for struct) End with
                           -SETTINGS (individual settings)
  -r, --recursive           Recursive the path.
  -wf WORKFLOW, --workflow WORKFLOW
                           Specify the workflow to run. Default: get_wf_gibbs
                           (NOTE: currently, only get_wf_gibbs is supported.)
  -l, --launch              Launch the wf to launchpad
  -m [MAX_JOB], --max_job [MAX_JOB]
                           Run the job, only works when -l is specified. Default:
                           0 (Not submit to queue) 1: qlaunch singleshots (single
                           job) N(N>1): qlaunch rapidfire -m N
  -o, --write_out_wf       Write out the workflow
```

*TO TOP*

## 6.8 Dfttk postprocessing mechanism

### 6.8.1 Flow controls

Depending on the given condition, postprocessing can take data from

- The qha-phonon collection
- The qha collection (Debye model)
- The phonon collection (in case of qha-phonon calculations failed)

Then, invoke Yphon to recalculate the phonon properties based on the force constants obtained from the database

## 6.8.2 Scheme to find equilibrium volume

If the data quality is excellent, use central symmetric 7-point difference

If the data quality is very good, fit the free energies use 4-parameter Birch-Murnaghan

If the data quality is good (may use as default), fit the 0-K total energies using 4-parameter Birch-Murnaghan

Fit the finite T part of the free energies by UnivariateSpline

## 6.8.3 Scheme to calculate derivative

1. 7-point symmetrical central difference

$$Deriv = \frac{1}{3} \sum_{i=1}^3 \frac{f(X_{N+i}) - f(X_{N-i})}{X_{N+i} - X_{N-i}}$$

2. Birch-Mannhan Euqations of state fitting

$$Deriv = -\frac{2}{3}bx^{-\frac{5}{3}} - \frac{4}{3}cx^{-\frac{7}{3}} - \frac{6}{3}dx^{-\frac{9}{3}}$$

## 6.8.4 Scheme for LTC

1. By entropy derivative

$$\alpha = \frac{1}{B_T} \frac{\partial S}{\partial V}$$

where  $B_T$  is isothermal bulk modulus

2. By internal energy derivative via Birch-Mannhan fitting

$$\alpha = \frac{1}{B_T T} \left( \frac{\partial U}{\partial V} + P \right)$$

where  $P$  is pressure

3. When the data quality is fair (~20% cases)

Fit the 0-K total energies using 4-parameter Birch-Murnaghan; and

fit the finite T part of the free energies by linear function  $f=a+b*V$

## 6.9 More about run DFTTK

To run DFTTK, the user should the following Linux command style:

```
dfttk 'module' [option]
```

where the available module can be:

module	function
run	DFT job submission. Call VASP to perform 0 K E-V, phonon, elastic moduli, Born effective charge (dielectric tensor), etc
config	Set basic dfttk run environment, such as path to pseudopotential, batch job prototype
db_remove	Remove certain data in MongoDB to save storage cost
thelec	Download data from MongoDB database & postprocess them to get thermodynamic properties, elastic constant, etc
thfind	Check the MongoDB database for DFT results followed by calling the thelec module to get thermodynamic properties when the option <code>-get</code> is given.
EVfind	Find the metadata tags that have 0 K static calculation finished.

### 6.9.1 Examples

- Check run results

To get a brief summary for data in the database, use

```
dfttk thfind -v 1
```

this will report the metadata, number of finished phonon calculations, number of finished static calculations, supercell size used for the phonon calculations, brief phase name; and the following command reports the location where the DFT calculations were submitted

```
dfttk thfind -v 1 -jobpath parentfolder
```

where parentfolder is the parent folder whose subfolders hold the individual DFT calculation job submission data.

- Batch postprocessing results

```
dfttk thfind -get -py -td -50 -plot find_or_DFT -eq 4 -renew -ss 30 -w Pb-Ti-O
```

where `-get` instruct thfind to call thelec module to postprocess the data, `-py` to use Yphon to recalculate the phonon density of states based on the force constants saved in the phonon collection, `-td -50` to use the self-adapted temperature mesh, `-eq 4` to use Birch-Murnaghan fitting to get equilibrium volume, `-ss 30` means only handle results with supercell size larger than 30, `-w elist` means only extract for compounds with all elements in the list of elist. Other enhanced screening conditions can be combinations of the follows

```
-all elist # compounds must contain all elements in the list of elist;
-any elist # compounds contain any elements in the list of elist;
-xall elist # compounds must not contain all elements in the list of elist;
-xany elist # compounds must not contain any elements in the list of elist.
```



## 6.9.2 Batch run dfttk

Depend on the cases, dfttk postprocess may take longer time to finish. If it is the case, it is recommended to submit batch job. When submit batch, make sure compatibilities of non-ascii character by including the following in the job script:

```
export LC_ALL='en_US.utf8' #for bsh;
setenv LC_ALL en_US.utf8 #for csh
```

## 6.9.3 Usage of run module

- dfttk run : call VASP to perform 0 K E-V, phonon, elastic moduli, Born effective charge (dielectric tensor), etc

```
usage: dfttk run [-h] [-f STRUCTURE_FOLDER] [-mh MATCH_PATTERN] [-s SETTINGS]
               [-r] [-wf WORKFLOW] [-ph] [-tag TAG] [-a] [-l] [-m [MAX_JOB]]
               [-o]
```

optional arguments:

```
-h, --help            show this help message and exit
-f STRUCTURE_FOLDER, --structure_folder STRUCTURE_FOLDER
                        The folder/file containing the structure, Default: '.'
-mh MATCH_PATTERN, --match_pattern MATCH_PATTERN
                        The match pattern for structure file, and it should be
                        place in quotes. e.g. '*POSCAR*'. Default: * --
                        everything except SETTING files, ref. -s
-s SETTINGS, --setting SETTINGS
                        Specify the name of SETTINGS files (yaml or json file)
                        Default: SETTINGS (case insensitive and without ext)
                        The following filename will be treat as SETTINGS file
                        SETTINGS (global settings in the folder) Start with
                        SETTINGS- (individual settings for struct) End with
                        -SETTINGS (individual settings)
-r, --recursive       Recursive the path.
-wf WORKFLOW, --workflow WORKFLOW
                        Specify the workflow to run. Default: robust (run
                        get_wf_gibbs_robust workflow) (NOTE: currently, only
                        robust and born are supported.)
-ph, --phonon         Run phonon. This is equivalent with set phonon=True in
                        SETTINGS file
-tag TAG, --tag TAG   Specify the tag for continue mode
-a, --append          Append calculation according to metadata, e.g.
                        appending volumes or phonon
-l, --launch          Launch the wf to launchpad
-m [MAX_JOB], --max_job [MAX_JOB]
                        Run the job, only works when -l is specified. Default:
                        0 (Not submit to queue) 1: qlaunch singleshot (single
                        job) N(N>1): qlaunch rapidfire -m N
-o, --write_out_wf    Write out the workflow
```

## 6.9.4 Usage of thelec module

- **thelec** [Download data from MongoDB database & postprocess them to get] thermodynamic properties, elastic constant, etc
- try `dfttk thelec -h` for command available line options

```
dfttk thelec [-h] [-py] [-T0 [T0]] [-T1 [T1]] [-dT [TD]] [-xdn [XDN]]
             [-xup [XUP]] [-dope [DOPE]] [-ne [NDOSMX]]
             [-natom [NATOM]] [-e [EVERYT]] [-gauss [GAUSSIAN]]
             [-i [DOSCAR]] [-o [OUTF]] [-noel] [-metatag [METATAG]]
             [-qhamode [QHAMODE]] [-pn [PHASENAME]] [-eq [EQMODE]]
             [-el [ELMODE]] [-s] [-plot] [-g] [-expt [EXPT]]
             [-xlim [XLIM]]
```

optional arguments:

```
-h, --help          show this help message and exit
-py, --pyphon       use Yphon to recalculate vibrational properties.
                    Default: False
-T0 [T0], -t0 [T0]  Low temperature limit. Default: 0
-T1 [T1], -t1 [T1]  High temperature limit. Default: 1300
-dT [TD], -td [TD]  Temperature increment. Default: 10
-xdn [XDN], --xdn [XDN]
                    Low band energy limit. Default: -100 (eV)
-xup [XUP], --xup [XUP]
                    High band energy limit. Default: 100
-dope [DOPE], --dope [DOPE]
                    dope level (electrons). Default: -1.e-8 for numerical
                    stability
-ne [NDOSMX], --ndosmx [NDOSMX]
                    new DOS mesh. Default: 10001
-natom [NATOM], --natom [NATOM]
                    number of atoms in the DOSCAR. Default: 1
-e [EVERYT], --everyT [EVERYT]
                    number of temperature points skipped from QHA
                    analysis. Default: 1
-gauss [GAUSSIAN], --gauss [GAUSSIAN]
                    densing number near the Fermi energy. Default: 1000
-i [DOSCAR], --doscar [DOSCAR]
                    DOSCAR filename. Default: DOSCAR
-o [OUTF], -outf [OUTF]
                    output filename for calculated thermoelectric
                    properties. Default: fvib_ele
-noel, -noel        do not consider the thermal electron contribution.
                    Default: False
-metatag [METATAG], -metatag [METATAG]
                    metatag: MongoDB metadata tag field. Default: None
-qhamode [QHAMODE], -qhamode [QHAMODE]
                    quasiharmonic mode: debye, phonon, or yphon. Default:
                    debye
-pn [PHASENAME], -phasename [PHASENAME]
                    assign phase name. Default: None
-eq [EQMODE], --eqmode [EQMODE]
```

(continues on next page)

(continued from previous page)

```

Mode to calculate LTC. 0: Symmetrical Central
differential; 4: 4-parameter BM fitting. 5:
5-parameter BM fitting. Default: 0
-el [ELMODE], --elmode [ELMODE]
Mode to interpolate thermal electronic contribution:
0: interp1d; 1: UnivariateSpline. Default: 0
-s, -smooth
smooth the LTC. Default: False
-plot, -plot
plot the figure. Default: False
-g, --debug
turn on debug mode by reducing the mesh. Default:
False
-expt [EXPT], -expt [EXPT]
json file path for experimental thermodynamic
properties for plot. Default: None
-xlim [XLIM], -xlim [XLIM]
Up temperature limit for plot. Default: None

```

## 6.9.5 Usage of thfind module

- **thfind**: Check the MongoDB database for DFT results followed by calling the ‘thelec’ module to get thermodynamic properties when the option ‘-get’ is given.
- try `dfttk thfind -h` for command available line options

```

dfttk thfind [-h] [-q [QHAMODE]] [-w [WITHIN]] [-all [CONTAINALL]]
             [-any [CONTAINANY]] [-v [NV]] [-ss [SUPERCELLN]] [-get]
             [-py] [-T0 [T0]] [-T1 [T1]] [-dT [TD]] [-xdn [XDN]]
             [-xup [XUP]] [-dope [DOPE]] [-ne [NDOSMX]]
             [-natom [NATOM]] [-e [EVERYT]] [-gauss [GAUSSIAN]]
             [-i [DOSCAR]] [-o [OUTF]] [-noel] [-metatag [METATAG]]
             [-qhamode [QHAMODE]] [-eq [EQMODE]] [-el [ELMODE]] [-s]
             [-plot] [-g] [-expt [EXPT]] [-xlim [XLIM]]

```

optional arguments:

```

-h, --help
show this help message and exit
-q [QHAMODE], --qhamode [QHAMODE]
Collection. 'phonon', 'qha'. Default: 'phonon'
-w [WITHIN], --within [WITHIN]
find calculations within element list Default: None
-all [CONTAINALL], --containall [CONTAINALL]
find calculations must contain all elements in the
list Default: None
-any [CONTAINANY], --containany [CONTAINANY]
find calculations contain any elements in the list
Default: None
-v [NV], --nV [NV]
Return phonon calculations finished for number of
volumes larger or equals to. Default: 6
-ss [SUPERCELLN], --supercellsize [SUPERCELLN]
only return phonon calculation with supercell size
larger than. Default: 0
-get, --get
get the thermodynamic data for all found entries.

```

(continues on next page)

(continued from previous page)

```

Default: False
-py, --pyphon          use Yphon to recalculate vibrational properties.
                        Default: False
-T0 [T0], -t0 [T0]     Low temperature limit. Default: 0
-T1 [T1], -t1 [T1]     High temperature limit. Default: 1300
-dT [TD], -td [TD]     Temperature increment. Default: 10
-xdn [XDN], --xdn [XDN]
                        Low band energy limit. Default: -100 (eV)
-xup [XUP], --xup [XUP]
                        High band energy limit. Default: 100
-dope [DOPE], --dope [DOPE]
                        dope level (electrons). Default: -1.e-8 for numerical
                        stability
-ne [NDOSMX], --ndosmx [NDOSMX]
                        new DOS mesh. Default: 10001
-natom [NATOM], --natom [NATOM]
                        number of atoms in the DOSCAR. Default: 1
-e [EVERYT], --everyT [EVERYT]
                        number of temperature points skipped from QHA
                        analysis. Default: 1
-gauss [GAUSSIAN], --gauss [GAUSSIAN]
                        densing number near the Fermi energy. Default: 1000
-i [DOSCAR], --doscar [DOSCAR]
                        DOSCAR filename. Default: DOSCAR
-o [OUTF], -outf [OUTF]
                        output filename for calculated thermoelectric
                        properties. Default: fvib_ele
-noel, -noel           do not consider the thermal electron contribution.
                        Default: False
-metatag [METATAG], -metatag [METATAG]
                        metatag: MongoDB metadata tag field. Default: None
-qhamode [QHAMODE], -qhamode [QHAMODE]
                        quasiharmonic mode: debye, phonon, or yphon. Default:
                        debye
-eq [EQMODE], --eqmode [EQMODE]
                        Mode to calculate LTC. 0: Symmetrical Central
                        differential; 4: 4-parameter BM fitting. 5:
                        5-parameter BM fitting. Default: 0
-el [ELMODE], --elmode [ELMODE]
                        Mode to interpolate thermal electronic contribution:
                        0: interp1d; 1: UnivariateSpline. Default: 0
-s, -smooth            smooth the LTC. Default: False
-plot, -plot           plot the figure. Default: False
-g, --debug            turn on debug mode by reducing the mesh. Default:
                        False
-expt [EXPT], -expt [EXPT]
                        json file path for experimental thermodynamic
                        properties for plot. Default: None
-xlim [XLIM], -xlim [XLIM]

```

## 6.10 Job monitoring

The following briefes some common `automate` commands for DFT jobs submitted by `dfttk run`.

- Monitor workflows

To check running status of all submitted `dfttk` jobs, use

```
lpad get_wflows
```

To find the running calculations

```
lpad get_fws -s RUNNING
```

To find the jobdir for a job with specific `fw_id` (The number portion of job id reported by `lpad get_wflows`), use `lpad get_launchdir fw_id`

- FIZZLED jobs

FIZZLED means failed. The following command can summarize all FIZZLED jobs:

```
lpad get_fws -s FIZZLED; or  
lpad get_wflows -s FIZZLED
```

One can rerun FIZZLED by

```
lpad rerun_fws -s FIZZLED
```

For more details, see ref. FIZZLED.

To get help for the subcommands, try `lpad get_fws -h` or `lpad get_wflows -s FIZZLED -h`

## 6.11 Useful help from FireWork

Controlling the Worker that executes a Firework

Querying FireWorks and Workflows/Generating Reports

Dealing with Failures and Crashes

Writing Queue Adapters

## 6.12 Troubleshooting

This document covers how to handle jobs that have fizzled. There are three main sections: common troubleshooting, general troubleshooting workflow to find the root cause of issue and specific issues and their fixes.

### 6.12.1 Common troubleshooting

1. In the config stages, one may need properly set up VASP environments, such as

```
module load intel impi vasp
```

2. Make the latest `automate` installed;
3. In `.bashrc` or `.cshrc`, make sure not messed up with your other `atomate` FW config;
4. Make sure the setup in the `.bashrc` file

```
export FW_CONFIG_FILE=~/.dfttk/config/FW_config.yaml
```

or equivalently in the `.cshrc` file

```
setenv FW_CONFIG_FILE /storage/work/y/yuw3/dfttk/config/FW_config.yaml
```

5. git push issue for contributors, see <https://docs.github.com/en/github/authenticating-to-github/adding-a-new-ssh-key-to-your-github-account>
6. for batch run of the postprocessing modules, make sure compatibilities of non-ascii character by:

```
export LC_ALL='en_US.utf8' #for bsh;
setenv LC_ALL en_US.utf8 #for csh
```

7. For phonon calculations, due to certain reasons (such as temperature range too high), one may not see results in the `qha_phonon` or `qha` MongoDB collections. In this case, the subcommand `dfttk thfind` will try to find results from the `phonon` collection and process the data by calling `Yphon`
8. When you are interesting in revising the code, if have job running in the system before your changes, the codes in the batch system might not be updated and the results might be not as you assumed. It takes me two days to figure out this problem. The solution is to kill all the `dfttk` running job and resubmit them. Following are the steps of adding API key number on DFTTK.

9. How to solve the install warning of ‘PMG\_MAPI\_KEY’ is empty.

1. Go to the materials project website, <https://materialsproject.org/>, under the API section, you will easily find you API Keys number.
2. Go to the `.pmgrc.yaml` file

```
vi ~/.pmgrc.yaml
```

3. Add your API key number into your `.pmgrc.yaml` file. For example

```
PMG_MAPI_KEY: #####(your API key number)
```

10. How to find the reason for FIZZLED job.

11. `VasprunXMLValidator` found from the command “`lpad get_fws -i fw_id -d more`”

If you get an error from the `VasprunXMLValidator`, that means that the `vasprun.xml` failed to be parsed and/or validated. Usually that means the VASP job did not run or failed for a reason that was not caught and handled by Custodian. Check the output files in the launch directory and see if there are any errors in the VASP output or `stdout/stderr`.

### 6.12.2 pymatgen 2021 issue

You may meet numpy version issues using pymatgen, reporting:

```
pymatgen 2021.2.16 requires numpy>=1.20.1, but you'll have numpy 1.19.2 which is
↳ incompatible.
```

In such case, please upgrade numpy by:

```
pip install numpy --upgrade
```

### 6.12.3 conda issues

In some cases, such as in the Windows environment, one may meet the error:

```
ModuleNotFoundError: No module named 'ruamel' #106
```

**This is due to conda bug on namespace of ruamel\_yaml vs ruamel.yaml.** One can resolve this by open the Anaconda Powershell Prompt as administrator and reinstall ruamel.yaml by:

```
conda install ruamel.yaml
```

### 6.12.4 Troubleshooting Workflow

**My job has fizzled!** The following steps can help you get information about how your job. You can imagine it as a decision tree. Check one thing before moving on to the next one.

1. Check that the job ran and has raised an exception with a traceback.

Run `lpad get_fws -i <ID> -d more`, replacing <ID> with the integer id of the Firework. Search the output for `_exception`. What you see is the Python exception that was raised when running the Firework.

*TIP::* Searching works well when you pipe the output to less with `lpad get_fws -i <ID> -d more | less` and search using `/`.

*todo::* If you don't see a traceback, that means... (this is the first step, but does this actually happen?)

2. Check the traceback is not a common error.

See the [Common Errors](#) section

### 6.12.5 Common Errors

#### Custodian VasprunXMLValidator failed

In this error, you get a traceback that looks something like:

```
Traceback (most recent call last):
  File "/storage/home/bjb54/.conda/envs/wfs/lib/python3.7/site-packages/custodian/
↳ custodian.py", line 320, in run
    self._run_job(job_n, job)
  File "/storage/home/bjb54/.conda/envs/wfs/lib/python3.7/site-packages/custodian/
↳ custodian.py", line 428, in _run_job
```

(continues on next page)

(continued from previous page)

```

    raise CustodianError(s, True, v)
custodian.custodian.CustodianError: (CustodianError(...), 'Validation failed: <custodian.
↳vasp.validators.VasprunXMLValidator object at 0x2af45b1d3908>')

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "/storage/home/bjb54/.conda/envs/wfs/lib/python3.7/site-packages/fireworks/core/
↳rocket.py", line 262, in run
    m_action = t.run_task(my_spec)
  File "/storage/home/bjb54/.conda/envs/wfs/lib/python3.7/site-packages/atomate/vasp/
↳firetasks/run_calc.py", line 204, in run_task
    c.run()
  File "/storage/home/bjb54/.conda/envs/wfs/lib/python3.7/site-packages/custodian/
↳custodian.py", line 330, in run
    .format(self.total_errors, ex))
RuntimeError: 0 errors reached: (CustodianError(...), 'Validation failed: <custodian.
↳vasp.validators.VasprunXMLValidator object at 0x2af45b1d3908>'). Exited...

```

With the key being that Custodian fails to validate the `vasprun.xml`. After running VASP, Custodian will try to parse the `vasprun.xml` file using `pymatgen`.

There are usually two possible triggers for this failure:

1. VASP failed to run at all (more common) or quit in a way that custodian did not detect (less common)
2. The `vasprun.xml` file could not be parsed by `pymatgen`.

To investigate this, first check that VASP ran (e.g. the `OUTCAR` shows that the run completed successfully). If VASP did not run, find out why and fix that issue. If VASP did run successfully, it was probably an issue parsing the `vasprun.xml` file. Try parsing the `vasprun.xml` file using the `pymatgen.io.vasp.outputs.Vasprun` class. If it throws an error when you try to parse, that's what made Custodian fail and you should fix that.

## 6.13 MongoDB VM hosting

This section is for the dfttk users who want to host their MongoDB by themselves.

MongoDB is one of the most popular document-oriented databases under the banner of NoSQL database. The schema-free implementation of MongoDB eliminates the prerequisites of defining a fixed structure required by the SQL database.

In computing, a virtual machine (VM) is an emulation of a computer system. Virtual machines are based on computer architectures and provide functionality of a physical computer. Their implementations may involve specialized hardware, software, or a combination. See [https://en.wikipedia.org/wiki/Virtual\\_machine](https://en.wikipedia.org/wiki/Virtual_machine)

Our MongoDB databases are currently hosted by [Penn State's VM hosting service](#) that provides cost-effective, reliable VM for departments, colleges, and research units at Penn State University.

In our case, we have changed the default tcp port from 27017 into 27018 due to historical reason



### 6.13.1 VM operation

Connect your VM by ssh (ssh `youruserid@146.186.149.69` in our case). One should use VPN if you have firewall for your system

- To add user to your VM linux system

Run the following command under Linux

```
sudo adduser newuser
usermod -aG sudo newuser #add admin user to your VM
```

Note on adding other users to access the VM. In order to add other users to access the VM from the Morpheus portal you would need to add them to the VM. For the VM itself you would need to add their user ID to the `/etc/security/access.conf` file and if they need sudo access you would need to add their ID to the `/etc/sudoers.d/sudo-users` file as well.

### 6.13.2 MongoDB installation/configuration

These are the **one time setup** steps for MongoDB:

Assuming you are on Ubuntu and using the apt package manager:

```
apt update
apt install mongodb
```

This may be an outdated MongoDB. To follow the best security practices and get the latest features, you should install something more recent. More information about installing and configuring MongoDB 3.6 is available here: <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/>

Additional configuration info for MongoDB 4.4 can be found here: <https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-18-04-source>

More broadly, the [MongoDB security checklist](#) should be followed. Critically, you should **enable access control** and set up an authentication database and credentials for at least one administrator.

Some key things to configure in `/etc/mongodb.conf`:

- Set the path of the database to the correct location. You may need to `sudo chown mongodb /path/to/database` for permissions to work
- Add the public IP of the server to `bind_ip` setting so it reads like `127.0.0.1,XXX.XXX.XXX.XX` for your server's IP `XXX.XXX.XXX.XX`.
- Set a non-standard port if you want one
- Enable authentication (assuming you **enabled authentication** and set up an administrator database and user)

### 6.13.3 MongoDB operation

#### Managing the MongoDB service

**This section assumes that you already have your VM set up and you are managing it under linux environment.**

The MongoDB server is managed by a systemd service that is managed through `systemctl`. Common commands are:

- Check status of mongodb service:

```
sudo systemctl status mongodb
```

- Start mongodb service:

```
sudo systemctl start mongodb
```

- Shut down mongodb service:

```
sudo systemctl stop mongodb
```

- Restart mongodb service (if it's already running):

```
sudo systemctl restart mongodb
```

## Connecting to the MongoDB server console

**This section deals with your MongoDB database management to locally or remotely operate on it.** This is to say you are going to manage your database from your local computer by mongo or mongosh.

#note mongosh by [MongoDB Shell](#) is the quickest way to connect, configure, query, and work with your MongoDB database

- Create admin user for mongodb

These are the **one time setup** steps for MongoDB:

With access control enabled, ensure you have a user with userAdmin or userAdminAnyDatabase role in the admin database. This user can administrate user and roles such as: create users, grant or revoke roles from users, and create or modify customs roles.

For more details on MongoDB user management, see <https://docs.mongodb.com/manual/tutorial/enable-authentication/>

1. Connect to the instance by open another terminal in your VM and connect a mongo shell to the instance:

```
mongod --port 27018
```

after the prompt ">" input:

```
use admin
db.createUser(
  {
    user: "admin",
    pwd: "xxxxxxxx", // xxxxxxxx is the admin password of your choice
    roles: [ { role: "userAdminAnyDatabase", db: "admin" }, "readWriteAnyDatabase" ]
  }
)
```

2. Re-start the MongoDB instance with access control

- a. Shut down the mongod instance
- b. Exit the mongo shell by run the command exit or give an EOF (Ctrl+D)
- c. Start the mongod with access control enabled by
  - adding the security.authorization configuration file setting

```
security:
  authorization: enabled
```

- or If you start the mongod from the command line

```
mongod --auth --port 27018
```

- Create general user

Assuming the service is running and configured with authentication (see above), Connect to your mongoDB as admin user locally by:

```
mongo --port 27018 --authenticationDatabase "admin" -u "admin" -p
```

or remotely by:

```
mongo 146.186.149.69:27018 --authenticationDatabase admin -u <admin username> -p <admin.
↪password>
```

or remotely use mongosh by:

```
mongosh --username <admin username> --password --authenticationDatabase admin --host 146.
↪186.149.69 --port 27018
```

followed by inputting the following lines after the prompt ">":

```
use userid-fws
db.createUser({user: "userid", pwd: "B5nRcUvoCZ92", roles: [{role: "dbOwner", db:
↪"userid-fws"}]})
use userid-results
db.createUser({user: "userid", pwd: "BeFihJ2mrKGm", roles: [{role: "dbOwner", db:
↪"userid-results"}]})
db.createUser({user: "userid-ro", pwd: "QIvaUT9ca6H8", roles: [{role: "read", db:
↪"userid-results"}]})
```

These lines can be produced by dfttk by run a python code named `mongodb_user.py` which can be downloaded from <https://github.com/PhasesResearchLab/dfttk/tree/master/dfttk/scripts> After download the code, one can run it by:

```
python mongodb_user.py
```

The run will prompt the MongoDB system manager to input an userid for the user. After you input userid and hit enter, one gets the above outputs in the screen.

Meanwhile, a file named `db.json` in the JSON format containing something similiar to the following lines which should be sent to the MongoDB user:

```
{
  "database": "userid-results",
  "collection": "tasks",
  "admin_user": "userid",
  "admin_password": "BeFihJ2mrKGm",
  "readonly_user": "userid-ro",
  "readonly_password": "QIvaUT9ca6H8",
  "host": "146.186.149.69",
  "port": 27018,
  "aliases": {}
}
```

The MongoDB user should save this data in a json file named `db.json` under the path `dfttk/config` that created by `dfttk config -mp -aci` command.

- Remove user:

```
db.removeUser(username)
```

## 6.14 Changelog

### 6.14.1 0.3.1 (2021-02-03)

(Contributor: @YiWang, @mx469, @bocklund)

- Change List:
- Made a significant online documentation using readthdocs
- Documentation open online by [www.dfttk.org](http://www.dfttk.org)

### 6.14.2 0.3.0 (2020-12-10)

(Contributor: @YiWang, @mx469, @bocklund)

- Change List:
- formally release dfttk version 0.3.0
- Revised `dfttk/script/run_dfttk.py`
- made workflow `get_wf_gibbs` run on robust with `get_wf_gibbs_robust`
- changed test module from `get_wf_gibbs` into `get_wf_gibbs_robust`
- Add elasticity calculation model
- Add Born effective charge calculations

### 6.14.3 September 17, 2020

- Bug fixes:
- Replaced `curve-fit` by `polyfit` for BM fitting
- Fixed bugs on result checks
- Summarizing installation/run documents for DFTTK (This documents)

### 6.14.4 September 3, 2020

- Added a module `EVfind` per discussion with Shunli and enhanced calculation summary
- E-V, P-V, Stress-V, Strain-V
- Implemented 4 numerical schemes for evaluating LTC
- Enhanced plots
- LO-TO splitting were marked
- Gamma point phonon frequencies
- Prepared ~50 testing structures/compounds

- Made LDA pseudopotential as an option
- Speed up calculation
- Turned on LREAL=Auto from default
- Change into ISPIN=1 for nonmagnetic system from default
- For better relaxation
- Turned on EDIFF=1.e-8 (why it showed 2e-7 in the OUTCAR?)
- Turned on EdiffG=-1.e-3 (testing, could be the reason for poor phonon data quality)
- Data report
- Phonon quality
- LTC quality
- Found phonopy was unable handle Cv below 1 K
- Added experimental data for plotting results
- Working on
- Calculate Born effective charge/dielectric tensors for insulators
- Launch by module EVfind when band gaps found and if not calculated

### 6.14.5 0.2.2 (2020-08-18)

(Contributor: @YiWang, @mx469)

- Change List:
  - added codes in the dfttk/scripts directory:
    - run\_dfttk\_ext.py
    - handle the arguments for the thelec and thfind modules
  - added python code in the dfttk directory:
    - pyfind.py
    - database search engine
    - pythelec.py
    - for compatibility with Yphon
    - generating the majority of thermodynamic properties, such as thermal expansion coefficient, Seebeck coefficients, Lorenz number etc
    - pyphon.py for
      - calculate the phonon contributions to the various thermodynamic properties
  - added python code in the dfttk/analysis directory:
    - database
    - for plot phonon dispersions for all crystalline systems
    - ywutils.py
    - general utils code

- ywplot.py
  - for plots of ~20 different phonon and thermodynamic properties in the png format
  - made Yphon compatible with phonopy
    - added codes in the CRO-soc directory:
      - phonopy2yphon, phonopy2yphon.py
      - convert the phonopy force constant matrix in hdf5 format into superfij.out format used by Yphon
    - changed codes:
      - in the dfttk/scripts directory:
        - run\_dfttk.py
          - added the following lines aimed to handle the arguments for the thelec and thfind modules
- ```
# extension by Yi Wang, finalized on August 4, 2020 #
_____ from dfttk.scripts.run_dfttk_ext import
run_ext_thelec run_ext_thelec(subparsers)
```
- in the dfttk/analysis directory:
    - debye.py is renamed as debye\_ext.py
    - to include the vibrational entropy (S\_vib) and heat capacity (C\_vib) into the “qha” MongoDB collection
    - quasiharmonic.py:
    - copy the S\_vib and C\_vib from the “phonon” collection into the “qha\_phonon” MongoDB collection

### 6.14.6 0.2 (2020-03-30)

New features

(Contributor: @bocklund , @Peng\_Gao, @hitliaomq )

- The relax scheme is optimized. (from ISIF=3 to ISIF=2 followed by ISIF=4) (@Peng\_Gao)
- Change the static workflow to dynamic workflow. (EVcheck\_QHA.py increase the data points atomately if the fitting of initial points is incorrect) (@Peng\_Gao)
- Support run dfttk by command. (Add dfttk run [options]) (@hitliaomq)
- Support configrate dfttk automately. (Add dfttk config [options]) (@hitliaomq)
- Documents' enhance. (@hitliaomq)
- Bug fix. (Including #8 ) (@bocklund, @Peng\_Gao, @hitliaomq)

### 6.14.7 0.1 (2018-08-28)

Initial release. Includes

(Contributor: @bocklund, @mxf469)

- Gibbs workflow for stable structures
- Analysis code and libraries for calculation quasiharmonic Gibbs energies with OK, vibrational and thermal electronic contributions
- Useful utilities for interfacing with structure, calculations and the Materials Project

## 6.15 Contribution Guide


### 6.15.1 Style Guidelines

In general, code style should follow [PEP8](#) and [PEP20](#). Specifics are summarized below:

- Code should be indented using spaces, not tabs. One indentation = 4 spaces
- Lines longer than 100 should be manually wrapped, but prefer readability
- Minimize blank lines: 2 around top level classes functions, 1 in nested functions
- Workflows, Fireworks, and Firetasks should follow the same naming scheme as in atomate
- Include docstrings for classes and functions (see code), add comments where needed
- Function and variable names should be descriptive (not 'x' or 'xx') and all lowercase\_with\_underscores
- Class names should be descriptive CapitalWords

### 6.15.2 How to Contribute

It is recommended to first install [anaconda](#) if one does not have it installed yet.

1. Register an account in [www.github.com](http://www.github.com) or [www.gitlab.com](http://www.gitlab.com), depending on where the package that you want to contribute to is resided.
2. Sign into your web account
3. find the the package that you want to contribute to
4. make a fork by click the  Fork sign on top-right corner of the repository (for instance, see the web site of [DFTTK](#))
5. go back to your web account, you will see the forked repository shown in your account
6. login in your local machine
7. clone the repository to your local machine and go to/create a folder that you want to reside your contribution, then install the development version by

```
git clone https://github.com/PhasesResearchLab/dfttk.git
cd dfttk
pip install -e .
dfttk config -mp -aci #a folder named "config" will be created where running_
↪environmental info saved
```

8. Create a new branch for your addition or changes (`git checkout -b mybranchname`), something like


```
git checkout -b yourhname # "yourhname" is a name that you preferred to use
```

9. Write codes or make changes and commit them to your branch. (#Note from me, try your best not changing the original codes, only focusing expanding codes will save a lot of troubles when merging your changes to the package)

```
git commit -am "your notation for the changes"
```

10. Push your branch to the repository

```
git push #push your changes to github/gitlab
```

11. **\*When and only when your are sure that your changes are completely correct\***, submit a pull request by going to your web account and click the sign of  Pull requests on the top-left side of your repository

After you submit a merge request, other members of the group are able to review your changes and give feedback. Someone with a rank of Master or higher in the project can merge your commits into the master branch.

## 6.16 Releasing DFTTK

When releasing a new version of DFTTK

1. `git pull` to make sure you haven't missed any last-minute commits. **After this point, nothing else is making it into this version.**
2. Ensure that all tests pass locally on develop.
3. `git push` and verify all tests pass on all CI services.
4. Generate a list of commits since the last version with `git --no-pager log --oneline --no-decorate 0.1^..origin/master` Replace 0.1 with the tag of the last public version.
5. Condense the change list into something user-readable. Update and commit `CHANGES.rst` with the release date.``
6. `git tag 0.2 master -m "0.2"` Replace 0.2 with the new version. `git show 0.2` to ensure the correct commit was tagged `git push origin master --tags`
7. The new version is tagged in the repository. Now the public package must be built and distributed.

### 6.16.1 Uploading to PyPI

1. `rm -R dist/*` on Linux/OSX or `del dist/*` on Windows
2. With the commit checked out which was tagged with the new version: `python setup.py sdist`  
**Make sure that the script correctly detected the new version exactly and not a dirty / revised state of the repo.**  
Assuming a correctly configured `.pypirc`:  
`twine upload -u bocklund dist/*`



## 6.16.2 Some useful commands are following

```
git checkout master
#
git pull
git pull upstream master
git --version
git remote set-url --all upstream git@github.com:PhasesResearchLab/dfttk.git
git remote set-url --add upstream git@github.com:PhasesResearchLab/dfttk.git
git remote set-url --delete upstream git@github.com:PhasesResearchLab/dfttk.git
git remote remove upstream
git remote rm upstream
git remote -v
git remote add upstream git@github.com:PhasesResearchLab/dfttk.git
#
git tag -d 0.3.0
git push --delete upstream 0.3.0
git tag 0.3.0 master -m "0.3.0"
git push upstream master --tags
rm dist/*
python setup.py sdist
twine upload -u yiwang62 dist/*
```

## 6.17 Recipes

### 6.17.1 Construct a series of Debye workflows by substituting into a template structure

```
DRY_RUN = True  # Don't submit the workflows
VERBOSE = True  # Turn on printing of substitutions

# Filename of the template structure to use, usually a dilute or SQS structure
TEMPLATE_STRUCTURE_FILENAME = 'structures/FCC_A1.sqs.12Ni-4Fe.POSCAR'

# sublattice configuration of the template structure.
# This will be substituted exactly, this does not need to be sorted,
# however the individual configurations to build should be sorted.
TEMPLATE_SUBLATTICE_CONFIGURATION = [['Fe', 'Ni']]
TEMPLATE_SUBLATTICE_OCCUPANCIES = [[0.25, 0.75]]
SUBLATTICE_SITE_RATIOS = [1.0]

PHASE_NAME = 'FCC_A1'

configurations_to_build = [  # list of sublattice configurations in DFTTK_
    ↪format
    [['Cr', 'Ni']],
    [['Fe', 'Ni']],
    # [['V', 'Ni']], # should not use this because it's out of order, make a_
    ↪second script with the templates flipped
```

(continues on next page)

(continued from previous page)

```

]

# Dictionary of densities for each pure element.
# Not necessary, using the periodic_table in pymatgen instead
#DENSITY_DICT = {
#    'V': 6.313, # bcc
#    'Cr': 7.463, # bcc
#    'Ni': 9.03, # fcc
#    'Ti': 4.58, # hcp
#    'Fe': 8.028 # bcc
#}

```

**# SCRIPT #**

# Should not need to edit below this line.

```

from pymatgen import Structure
from fireworks import LaunchPad
from dfttk import get_wf_gibbs
from dfttk.structure_builders.substitutions import substitute_configuration_with_metadata

workflows = []

temp_struct = Structure.from_file(TEMPLATE_STRUCTURE_FILENAME)

for config in configurations_to_build:
    struct, meta = substitute_configuration_with_metadata(temp_struct, TEMPLATE_
↳SUBLATTICE_CONFIGURATION, config, TEMPLATE_SUBLATTICE_OCCUPANCIES, PHASE_NAME,
↳SUBLATTICE_SITE_RATIOS)
    if VERBOSE:
        print("PHASE: {} CONFIGURATION: {} OCCUPANCIES: {} STRUCTURE: {}".
↳format(PHASE_NAME, config, TEMPLATE_SUBLATTICE_OCCUPANCIES, struct.composition.hill_
↳formula))
        workflows.append(get_wf_gibbs(struct, deformation_fraction=(-0.05,0.10),
↳phonon=False, num_deformations=11, t_max=2000, metadata=meta))

#####
# Load everything in the LaunchPad
#####

if VERBOSE:
    print("{} workflows.".format(len(workflows)))

if DRY_RUN:
    exit()

if VERBOSE:
    print('Adding workflows to LaunchPad')

lpad = LaunchPad.auto_load()

```

(continues on next page)

(continued from previous page)

```
for workflow in workflows:
    lpad.add_wf(workflow)
```

### 6.17.2 ESPEI datasets from a QHA database

The following code snippet will take ESPEI datasets from a QHA database, optionally writing the (nicely named) files to dict. The QHA database requires the following metadata schema:

```
{
  'metadata': {
    'phase_name': 'FCC_A1',
    'tag': 'ed447049-ad67-4090-ba99-378188d3416b',
    'sublattice': {
      'configuration': [['Cr', 'Ni']],
      'occupancies': [[0.03125, 0.96875]]
    },
  }
}
```

```
#####
# EDIT #
#####

phase_name = 'BCC_A2'
configuration_to_find = [['Ni', 'V']]
sublattice_site_ratios = [1.0]
db_username = 'BrandonResultsR0'
db_password = 'piqhg38hap3'
db_uri = 'mongodb://206.189.190.225:27018'
WRITE_FILES = True

temperature_index = 59 # index of 300 K temperature (close to 298 K), found by hand

refstate_tags = {
    'Fe': '4ac77fce-0e43-4c07-8418-4843a2cd5723',
    'Ni': '0059ee69-4a8f-4e86-9895-9b40cf67dd96',
    'Cr': '8cceb186-2796-4488-ba8c-2380c5278f62',
    'V': 'fba46b6b-1699-419f-b5e1-da9533530701',
}

#####
### SCRIPT
#####

from dfttk.analysis.formation_energies import get_formation_energy, get_thermal_props
from dfttk.espei_compat import make_dataset, dfttk_config_to_espei, dfttk_occupancies_to_
↳ espei
from pymatgen import Structure
import numpy as np
```

(continues on next page)

(continued from previous page)

```

from dfttk.utils import recursive_flatten
import json
from pymongo import MongoClient

# create the MongoClient
cli = MongoClient(db_uri)
db = cli.results
db.authenticate(name=db_username, password=db_password)
coll = db.qha

# construct the energies, assumption of same temperature grid
# energies are J/mol-atom
refstate_energies = {}
for el, tag in refstate_tags.items():
    qha_result = coll.find_one({'metadata.tag': tag})
    refstate_energies[el] = get_thermal_props(qha_result)

# calculate all the dataset values
configs      = []
occupancies  = []
hm_values    = []
sm_values    = []
cpm_values   = []

# we'll change the T to the right temperatures later
fixed_conds = {'P': 101325, 'T': 0}
temp_conds  = {'P': 101325, 'T': 0}

for qha_res in coll.find({'metadata.sublattice.configuration': configuration_to_find,
    ↳ 'metadata.phase_name': phase_name}):
    configs.append(qha_res['metadata']['sublattice']['configuration'])
    occupancies.append(qha_res['metadata']['sublattice']['occupancies'])

    tprops = get_thermal_props(qha_res)
    struct = Structure.from_dict(qha_res['structure'])
    hm_form = get_formation_energy(tprops, struct, refstate_energies, 'HM',
    ↳ idx=temperature_index)
    sm_form = get_formation_energy(tprops, struct, refstate_energies, 'SM',
    ↳ idx=temperature_index)
    cpm_form = get_formation_energy(tprops, struct, refstate_energies, 'CPM', thin=10)[:
    ↳ 2]
    fixed_temp = tprops['T'][temperature_index]
    cpm_temps = tprops['T'][:10][:-2]

    hm_values.append(hm_form)
    sm_values.append(sm_form)
    cpm_values.append(cpm_form)

fixed_conds['T'] = fixed_temp.tolist()
temp_conds['T'] = cpm_temps.tolist()

```

(continues on next page)

(continued from previous page)

```

# make the HM, SM, CPM values arrays of the proper shape
hm_values = np.array([[hm_values]])
sm_values = np.array([[sm_values]])
cpm_values = np.array(cpm_values).T[np.newaxis, ...]

if WRITE_FILES:
    # write JSON files
    comps = [c.upper() for c in sorted(recursive_flatten(configuration_to_find))]
    for prop, vals, conds in [('HM_FORM', hm_values, fixed_conds), ('SM_FORM', sm_values,
→ fixed_conds), ('CPM_FORM', cpm_values, temp_conds)]:
        ds = make_dataset(phase_name, prop, sublattice_site_ratios, configs, conds, vals,
→ occupancies=occupancies, tag=tag)
        with open('{}-{}-{}-DFTTK.json'.format('-'.join(comps), phase_name, prop), 'w')
→ as fp:
            json.dump(ds, fp)

```

## 6.18 FAQ

### 6.18.1 What do the references to DFTTK-style or ESPEI-style configurations or occupancies mean?

ESPEI is separate software for fitting thermodynamic databases within the CALPHAD method. The CALPHAD method requires descriptions of the formation and mixing energies of phases described by sublattices, as in the compound energy formalism.

DFTTK is an excellent tool for calculating the temperature and composition dependent thermodynamic properties of sublattice configurations within the compound energy formalism.

In both ESPEI and DFTTK, there are three key aspects to describing a phase in the CEF: sublattice site ratios, sublattice configurations, and sublattice occupancies. Sublattice site ratios are a list of integer or fraction ratios of the sublattice. There are no constraints on the order or values. Some valid ones would be:

```

[1] # 1 sublattice
[3.0, 2.0] # 2 sublattices with 3 occurrences of species in sublattice 1 and 2 in
→ sublattice 2
[1, 0.5] # 2 sublattices with 1 occurrence of species in sublattice 1 and 1/2 in
→ sublattice 2
[16, 11] # 2 sublattices with 16 occurrences of species in sublattice 1 and 11 in
→ sublattice 2

```

DFTTK describes a sublattice configuration as a list of sublattices, where each sublattice is also a list of the components occupying that sublattice. Some examples are:

```

[[Fe, Ni]] # 1 sublattice with mixing of Fe and Ni
[[Mg], [Cu]] # 2 sublattices with no mixing in each sublattice.
[[Cr, Fe], [Cr], [Cr, Fe]] # 3 sublattices with mixing in the first and last sublattice

```

The ordering of sublattices themselves have no distinct meaning. As a convention, DFTTK sorts the components in each sublattice by component name, e.g., Cr (starts with C) always comes before Fe (starts with F)). ESPEI requires each sublattice to be ordered in this way.

Sublattice occupancies or occupations describe how a sublattice is filled with each particular element. The shape of these corresponds exactly to the configuration that the occupancy list describes. The sorting of the occupancies therefore is also in the same order of the configuration.

DFTTK includes several tools to help build ESPEI-compatible sublattice descriptions. The only difference between DFTTK and ESPEI configurations and occupancies is that a singly occupied sublattice, e.g. `[[Fe]]` in DFTTK, is not wrapped in the sublattice parenthesis for both the configuration and occupancies. Some examples of ESPEI configuration and occupancy lists are given below:

```
[[FE, NI]], [[0.5, 0.5]] # this is the same as DFTTK, there are no single occupations
[MG, CU], [1.0, 1.0] # each sublattice is singly-occupied and is not wrapped in the
↪ parenthesis
[[CR, FE], CR, [CR, FE]], [[0.5, 0.5], 1, [0.3, 0.7]] # the second sublattice is singly
↪ occupied and is not wrapped
```

As a final note, ESPEI format typically uses all uppercase, consistent with the CALPHAD community, while DFTTK usually uses standard element capitalization. The conversion tools take this into account.

## 6.19 Acknowledgements

This work is so-supported by the following funding resource:

Department of Energy/ Advanced Research Projects Agency - Energy (ARPA-E) with Funding Opportunity No. DE-FOA-0002337, ULTIMATE. Award No.: DE-AR0001435

Department of Energy/ EERE Office of Energy Efficiency and Renewable Energy. Award No.: DE-EE0008456

Department of Energy, Office of Fossil Energy, National Energy Technology Laboratory, Innovative Technology Development to Enhance Fossil Power System Efficiency, Funding Opportunity Number: DE-FOA-0001686. Award number: DE-FE0031553

Department of Energy: Office of Nuclear Energy, Funding Opportunity: DE-FOA-0001772, Award No.: DE-NE0008757

Department of Energy, Basic Energy Sciences; DATA SCIENCE FOR DISCOVERY IN Chemical and Materials Sciences, Award number: DE-SC0020147

Department of Energy, Basic Energy Sciences, Award Number: DE-SC0020145.

DOE-NEUP, Funding Opportunity Announcement DE-FOA-0002128. MS-FC-1: Understanding, Predicting, and Optimizing the Physical Properties, Structure, and Dynamics of Molten Salt, Award number: DE-NE0008945

National Science Foundation (NSF) through Grant No. CMMI-1825538.

Office of Naval Research (ONR) project, Contract no. N00014-17-1-2567

Pennsylvania State University's Institute for CyberScience through the ICS Seed Grant Program

The 2020 Institute for Computational and Data Sciences (ICDS) Seed Grant at the Pennsylvania State University.

First-principles calculations were performed partially on the Roar supercomputer at the Pennsylvania State University's Institute for Computational and Data Sciences (ICDS), partially on the resources of the National Energy Research Scientific Computing Center (NERSC) supported by the U.S. Department of Energy Office of Science User Facility operated under Contract No. DE-AC02-05CH11231, and partially on the resources of the Extreme Science and Engineering Discovery Environment (XSEDE) supported by National Science Foundation with Grant No. ACI-1548562.

## 6.20 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)