

Phono3py

Introduction

This GPL software calculates phonon-phonon interactions via the third order force constants. It allows to obtain lattice thermal conductivity, phonon life-time/linewidth, imaginary part of self energy at the lowest order, joint density of states (JDOS) and weighted-JDOS. For details see Phys. Rev. B 91, 094306 (2015) and <http://atzto.github.io/phono3py/index.html>

Load the phono3py/0.9.14-ictce-7.3.5-Python-2.7.9 module

```
$ module load phono3py/0.9.14-ictce-7.3.5-Python-2.7.9
```

Example of calculating thermal conductivity of Si using VASP code.

Calculating force constants

One needs to calculate second order and third order force constants using the diamond structure of silicon stored in POSCAR (the same form as in VASP) using single displacement calculations within supercell.

```
$ cat POSCAR
Si
1.0
5.4335600309153529 0.0000000000000000 0.0000000000000000
0.0000000000000000 5.4335600309153529 0.0000000000000000
0.0000000000000000 0.0000000000000000 5.4335600309153529
Si
8
Direct
0.8750000000000000 0.8750000000000000 0.8750000000000000
0.8750000000000000 0.3750000000000000 0.3750000000000000
0.3750000000000000 0.8750000000000000 0.3750000000000000
0.3750000000000000 0.3750000000000000 0.8750000000000000
0.1250000000000000 0.1250000000000000 0.1250000000000000
0.1250000000000000 0.6250000000000000 0.6250000000000000
0.6250000000000000 0.1250000000000000 0.6250000000000000
0.6250000000000000 0.6250000000000000 0.1250000000000000
```

Generating displacement using 2x2x2 supercell for both second and third order force constants

```
$ phono3py -d --dim="2 2 2" -c POSCAR
```

disp_fc3.yaml	POSCAR-00008	POSCAR-00017	POSCAR-00026	POSCAR-00035	POSCAR-00044	POSCAR-00053
POSCAR	POSCAR-00009	POSCAR-00018	POSCAR-00027	POSCAR-00036	POSCAR-00045	POSCAR-00054
POSCAR-00001	POSCAR-00010	POSCAR-00019	POSCAR-00028	POSCAR-00037	POSCAR-00046	POSCAR-00055
POSCAR-00002	POSCAR-00011	POSCAR-00020	POSCAR-00029	POSCAR-00038	POSCAR-00047	POSCAR-00056
POSCAR-00003	POSCAR-00012	POSCAR-00021	POSCAR-00030	POSCAR-00039	POSCAR-00048	POSCAR-00057
POSCAR-00004	POSCAR-00013	POSCAR-00022	POSCAR-00031	POSCAR-00040	POSCAR-00049	POSCAR-00058
POSCAR-00005	POSCAR-00014	POSCAR-00023	POSCAR-00032	POSCAR-00041	POSCAR-00050	POSCAR-00059
POSCAR-00006	POSCAR-00015	POSCAR-00024	POSCAR-00033	POSCAR-00042	POSCAR-00051	POSCAR-00060
POSCAR-00007	POSCAR-00016	POSCAR-00025	POSCAR-00034	POSCAR-00043	POSCAR-00052	POSCAR-00061

```

$./prepare.sh
$ls
disp-00001  disp-00009  disp-00017  disp-00025  disp-00033  disp-00041  disp-00049  disp-00057
disp-00002  disp-00010  disp-00018  disp-00026  disp-00034  disp-00042  disp-00050  disp-00058
disp-00003  disp-00011  disp-00019  disp-00027  disp-00035  disp-00043  disp-00051  disp-00059
disp-00004  disp-00012  disp-00020  disp-00028  disp-00036  disp-00044  disp-00052  disp-00060
disp-00005  disp-00013  disp-00021  disp-00029  disp-00037  disp-00045  disp-00053  disp-00061
disp-00006  disp-00014  disp-00022  disp-00030  disp-00038  disp-00046  disp-00054  disp-00062
disp-00007  disp-00015  disp-00023  disp-00031  disp-00039  disp-00047  disp-00055  disp-00063
disp-00008  disp-00016  disp-00024  disp-00032  disp-00040  disp-00048  disp-00056  disp-00064

```

```
$ ./submit.sh
```

Once all jobs are finished and `vasprun.xml` is created in each `disp-XXXXX` directory the collection is done by

and `disp_fc2.yaml`, `FORCES_FC2`, `FORCES_FC3` and `disp_fc3.yaml` should appear and put into the hdf format by

resulting in `fc2.hdf5` and `fc3.hdf5`

Thermal conductivity

The phonon lifetime calculations takes some time, however is independent on grid points, so could be splitted:

```
$ phono3py --fc3 --fc2 --dim="2 2 2" --mesh="9 9 9" --sigma 0.1 --wgp
```

Inspecting ir_grid_points.yaml

```
$ grep grid_point ir_grid_points.yaml
num_reduced_ir_grid_points: 35
ir_grid_points: # [address, weight]
- grid_point: 0
- grid_point: 1
- grid_point: 2
- grid_point: 3
- grid_point: 4
- grid_point: 10
- grid_point: 11
- grid_point: 12
- grid_point: 13
- grid_point: 20
- grid_point: 21
- grid_point: 22
- grid_point: 30
- grid_point: 31
- grid_point: 40
- grid_point: 91
- grid_point: 92
- grid_point: 93
- grid_point: 94
- grid_point: 101
- grid_point: 102
- grid_point: 103
- grid_point: 111
- grid_point: 112
- grid_point: 121
- grid_point: 182
- grid_point: 183
- grid_point: 184
- grid_point: 192
- grid_point: 193
- grid_point: 202
- grid_point: 273
- grid_point: 274
```

- grid_point: 283
- grid_point: 364

one finds which grid points needed to be calculated, for instance using following

```
$ phono3py --fc3 --fc2 --dim="2 2 2" --mesh="9 9 9" -c POSCAR --sigma 0.1 --br --write-gamma --g
```

one calculates grid points 0, 1, 2. To automate one can use for instance scripts to submit 5 points in series, see gofree-cond1.sh

```
$ qsub gofree-cond1.sh
```

Finally the thermal conductivity result is produced by grouping single conductivity per grid calculations using

```
$ phono3py --fc3 --fc2 --dim="2 2 2" --mesh="9 9 9" --br --read_gamma
```