DynaPhoPy

Alessandro Serra

September 16, 2024

Contents

GitHub Repo

Un esempio di calcolo completo per il silicio è visibile al link Si-LAMMPS

1 Installazione

Al fine di rendere la gestione delle dipendenze più facile è consigliabile creare un ambiente *conda* apposito nel quale utilizzare i vari tools.

Di seguito la lista di pacchetti necessari:

- Python 2.7.x/3.5 o superiore;
- Matplotlib
- Scipy
- h5py
- pyYAML
- Phonopy 2.0 o superiore (Guida Installazione)
- (opzionale ma consigliato) pyFFTW
- (opzionale nel caso di GPU) Funzioni CUDA (Guida Installazione)

Nella mia esperienza installare pyFFTW tramite *conda* porta a dei problemi su MacOS, consiglio di scaricarlo via *pip*.

Installate le varie dipendenze procediamo con l'istallazione di *DynaPhoPy*:

```
$ pip install dynaphopy
```

2 File Necessari

2.1 File di Struttura POSCAR

L'unico file strettamente necessario (esclusi *force-fields*) è il file di struttura cristallina del materiale di interesse, da scaricare in formato *POSCAR*.

È possibile scaricarlo da un qualunque database, nel mio caso ho utilizzato il sito Materials Project.

2.2 File Input per Dynaphopy

Per prima cosa è necessario creare un file di input per *DynaPhoPy*, in questo stage ci servirà per convertire il file *POSCAR* in un file *lammps-data.*.
Di seguito l'input:

```
STRUCTURE FILE POSCAR
file -POSCAR
```

Per ora il file deve contenere solamente il percorso relativo del *POSCAR*.

2.3 File *lammps-data*

Possiamo ora convertire il file POSCAR in un formato leggibile da lammps con il comando:

```
$ dynaphopy input-file --dim 1 1 1 -c_lammps file.data
```

3 Interfaccia con *PhonoPy*

DynaPhoPy richiede che venga effettuato a priori il calcolo della matrice delle costanti di forza con *PhonoPy*.

3.1 Generazione degli Spostamenti

Partendo dal file data precedentemente creato generiamo gli spostamenti con il comando:

```
$ phonopy --lammps -c file.data -d --dim="N N N"
```

Dove la stringa tra virgolette rappresenta le dimensioni della supercella che si vuole utilizzare per il calcolo.

3.2 Calcolo dei *Force-Sets*

Una volta ottenuti gli spostamenti e i file delle supercelle con il comando precedente è necessario preparare uno script lammps per il calcolo delle forze. Dato che, in generale, gli spostamenti sono più di uno è necessario creare più file, oppure uno script bash che esegua un singolo script per diversi file data di supercella.
Ciò può essere fatto passando una stringa da linea di comando a lammps in modo che ad ogni iterazione esegua in calcolo per una diversa supercella:

```
#!/bin/bash

for iter in $(seq 1 N); do

  echo -e "Starting iteration #${iter}...\n"

  mpirun -np <numero processori> lmp_mpi -in in.forces \
  -var iter ${iter} | tee out.forces-${iter}

done
```

```
echo -e "\nDONE!\n"
```

Dove N rappresenta il numero di file di supercella creati da *PhonoPy*.

Di seguito lo script lammps per calcolare le forze:

```
units           metal

read_data       supercell-00${iter}

mass            1 28.084

pair_style      sw
pair_coeff      * * Si.sw Si

dump            phonopy all custom 1 force.1 id type \
                x y z fx fy fz
dump_modify     phonopy format line "%d %d \
                %15.8f %15.8f %15.8f %15.8f %15.8f %15.8f"

run 0
```

Per creare i *FORCE SETS* si deve eseguire il seguente comando:

```
$ phonopy -f force.*
```

3.3 Calcolo delle *Force Constant*

Una volta calcolati i *FORCE_SETS* è possibile calcolare la matrice delle costanti di forza con il comando:

```
$ phonopy --fc-calc FORCE_SETS --writefc --full-fc -v
```

La flag *–full-fc* consente di calcolare la matrice completa.

4 Calcolo con *DynaPhoPy*

A questo punto possiamo procedere con il calcolo delle bande a temperatura finita.

4.1 Generazione file supercella

I file di supercella generati da *PhonoPy* e *DynaPhoPy* sono leggermente diversi, consiglio quindi di rigenerarlo per evitare incongruenze:

```
$ dynaphopy input-file --dim N N N -c_lammps supercell.data
```

4.2 Run MD

Usando il file di supercella appena generato è necessario eseguire un run MD alla temperatura desiderata:

```
units                   metal

boundary                p p p

box                     tilt large

atom_style              atomic

read_data               si-sc.data

mass                    1 28.084

pair_style              sw
pair_coeff              * * Si.sw Si

variable                t equal ${temp}

neighbor                    0.3 bin

timestep                0.001

thermo_style            custom step etotal temp vol press
thermo                  1000

velocity                all create ${temp} 3627941 dist \
                        gaussian mom yes
velocity                all scale ${temp}

fix                     int all nvt temp ${temp} ${temp} 1.0

run                     10000
dump                    dynaphopy all custom 1 \
                        si${temp}k.lammpstrj vx vy vz
dump_modify             dynaphopy sort id

run                     50000
```

4.3 Aggiornamento Input *DynaPhoPy*

Una volta ottenuta la matrice delle costanti di forza procediamo a completare il file di input per *DynaPhoPy*.

Oltre al percoso del file POSCAR è necessario aggiungere:

- Percorso del file delle costanti di forza;
- Matrice dei vettori di reticolo;
- Matrice di trasformazione per la supercella;
- Percorso in spazio K nel quale rappresetare le bande.

Di seguito un esempio di input completo per il silicio:

```
STRUCTURE FILE POSCAR
./si-POSCAR

FORCE CONSTANTS
./FORCE_CONSTANTS

PRIMITIVE MATRIX
0.0  0.5  0.5
0.5  0.0  0.5
0.5  0.5  0.0

SUPERCELL MATRIX
2  0  0
0  2  0
0  0  2

MESH PHONOPY
40  40  40

BANDS
0.0,    0.0,    0.0          0.5,    0.0,   0.5
0.5,    0.0,    0.5          0.625   0.25   0.625
0.375,  0.375,  0.375        0.0,    0.0,   0.0
0.0,    0.0,    0.0          0.5,    0.5,   0.5
```

4.4 Calcolo delle Bande

A questo punto è possibile eseguire il calcolo da riga di comando, tuttavia è molto macchinoso e scarsamente documentato; ho preferito, invece, sfruttare una sorta di interfaccia grafica, richiamabile con il comando:

```
$ dynaphopy input-file temp.lammpstrj -ts <timestep_MD> -i
```

Successivamente apparirà una schermata del tipo:

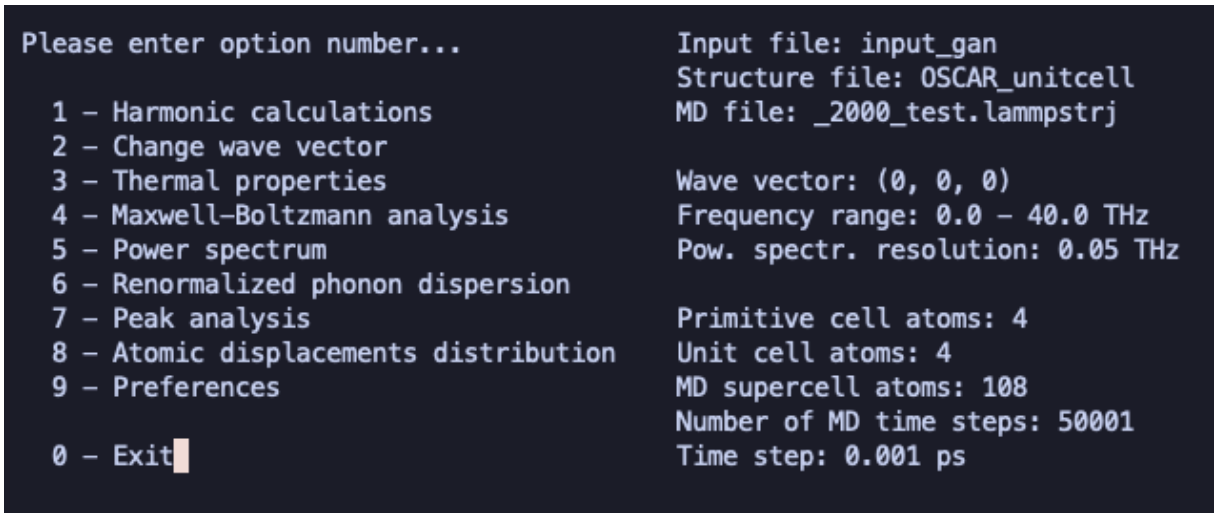

Figure 1: Schermata interattiva di *DynaPhoPy*.

Dalla quale è possibile:

- Calcolare le bande armoniche con la sequenza di numeri 1-3;
- Calcolare le bande a temperatura finita con la sequenza di numeri 6-1;
- Varie altre cose.

4.5 Algoritmo di Fitting

A seconda della temperatura è necessario cambiare algoritmo per il fitting delle bande (sequenza di numeri 9-1):

- Per temperature "basse" l'algoritmo di massima entropia (default) è senza dubbio il più efficiente, mantenendo una buona accuratezza;
- Per temperature "alte" l'errore di fitting è significativo (e talvolta enorme) ed è quindi necessario passare all'interpolazione in serie di Fourier.

Il programma ci consente anche di scegliere quale algoritmo utilizzare per le trasformate di Fourier. Avendo una GPU a disposizione *cuFFT* è senza ombra di dubbio il più rapido, in assenza di GPU *pyFFTW* è più efficiente rispetto a *Numpy*.

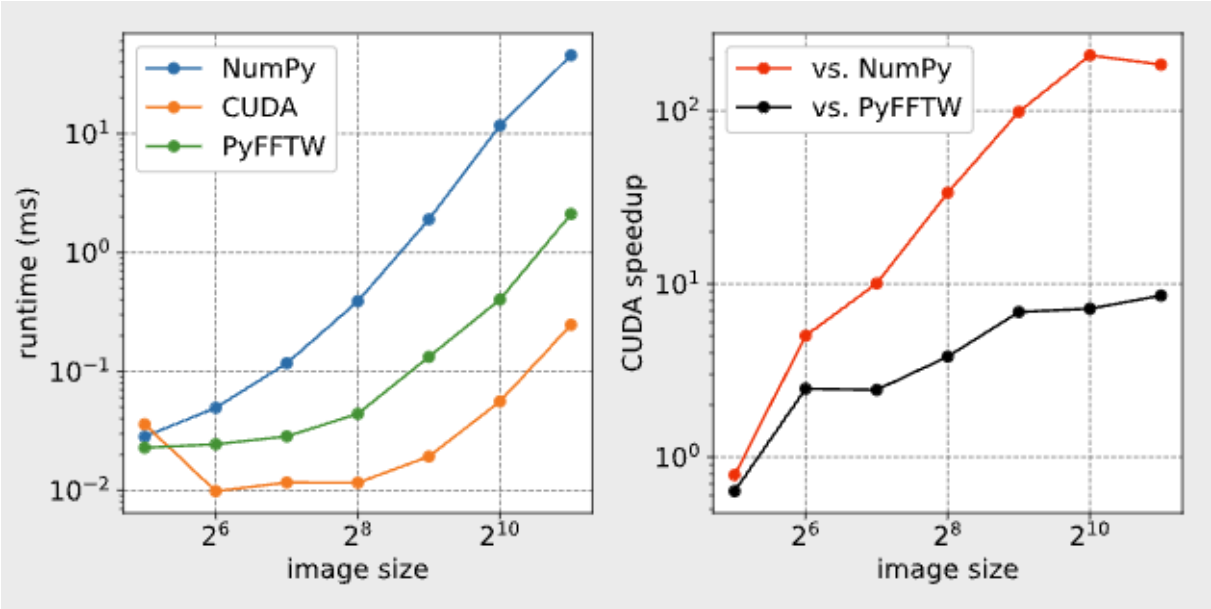

Figure 2: Confronto algoritmi FFT (i dati non sono correlati a questa analisi).

5 Salvare i Dati

DynaPhoPy non presenta un tool apposito per plottare e salvare i dati delle bande in un formato leggibile, dovremmo sfruttare *PhonoPy*.
Per prima cosa è necessario salvare le costanti di forza relative al run MD a temperatura finita con il comando:

```
$ dynaphopy input-file temp.lammpstrj -sfc \
FORCE_CONSTANTS_TEMP --MD_commensurate -psm N -ts <timestep>
```

Dove:

- *-sfc FORCE_CONSTANTS_TEMP* (attenzione a non sovrascrivere file esistenti) salva le nuove costanti di forza;
- *–MD_commensurate* è una flag necessaria in quanto abbiamo eseguito un run MD classico;
- *-psm N* indica l'algorigmo di fitting dei picchi come da numerazione presente nell'interfaccia interattiva (1 M.E.; 2 NumpyFFT; 3 pyFFTW, 4 cuFFT).

Creando un file di configurazione per *PhonoPy*:

```
DIM = N N N
FORCE_CONSTANTS = READ
PRIMITIVE_AXIS = AUTO
BAND = <percorso in spazio K>
MP = I J K (mesh griglia)
```

Dato che *PhonoPy* cerca la stringa "FORCE_CONSTANTS" è necessario rinominare il set di costanti di forza che si vuole utilizzare in tale modo.
Eseguendo il comando:

```
$ phonopy --dim="2 2 2" -c si-poscar band.conf
```

Otteniamo un file plottabile con:

```
$ phonopy-bandplot band.yaml
```

Convertibile nel seguente modo:

```
$ phonopy-bandplot --gnuplot band.yaml > band.dat
```

A questo punto il processo può essere ripetuto per qualunque temperatura e, usando direttamente le costanti di forza generate da *PhonoPy* all'inizio, anche a $0K$.