pynamic-structure-factor manual

Tyler C. Sterling[1,*] and Dmitry Reznik[1,2]

[1] *Department of Physics, University of Colorado at Boulder, Boulder, Colorado 80309, USA*

[2] *Center for Experiments on Quantum Materials,*
University of Colorado at Boulder, Boulder, Colorado 80309, USA

(Dated: June 11, 2021)

Abstract

This document is the manual for the pynamic-structure-factor PSF code.

INTRODUCTION

PSF is primarily intended to calculate the inelastic neutron scattering dynamic structure factors $S(\mathbf{Q}, \omega)$. This quantity is proportional to the measured inelastic neutron scattering intensities in experiments. The intensity from phonons varies from Brillouin zone to Brillouin zone according to the symmetry of the eigenvector, giving information about the polarization of a particular phonon. Importantly, it is possible to go to BZ's where only a few phonons have appereciable intensity. For large unit-cells with many atoms, the 'spectral-energy-density' (SED) method results in closely spaced bands blurred together. The usual method to resolve modes from SED is to project onto *harmonic* eigenvectors. This is undesirable for a few reasons: the harmonic eigenvectors aren't valid at high tempereature, the code to include them is more complicated requiring more user interaction, and broadly speaking, the SED method has no physically measurable analogue. Instead, $S(\mathbf{Q}, \omega)$ is *directly* comparable to experiment and one can use *multizone-fitting* to accurate determine phonon dispersions and linewidths for very complicated unit cells.

PSF code is primarily intended to calculate the inelastic neutron scattering dynamic structure factors $S(\mathbf{Q}, \omega)$ (which are proportional to the measured intensity), but it also can output time-averaged total intensities and Bragg intensities. See appendix (E) in in Dove [**?**] and the first 4 chapters of Squires [**?**] for the formalism.

According the Plancherel's theorem, the time-integrated intensity is required to be equal to the frequency (energy) integrated intensity. Actually my code computes the time average instead of time integral, so the forward time Fourier transform used to get $S(\mathbf{Q}, \omega)$ is normalized to that the *average* of $S(\mathbf{Q}, \omega)$ over all frequencies is equal to the time-averaged intensity.

The process to calculate $S(\mathbf{Q}, \omega)$ (or the time average or Bragg intensity) broadly works as follows:

1. run a molecular dynamics simulation and write the positions over time to a file. it can be either classical, empirical forces (see the LAMMPS scripts in the example dir.) or AIMD. Note that the $\mathbf{Q}$ resolution is determined by the supercell size so that AIMD

is probably not useful to compute $((\mathbf{Q}), \omega)$ colormaps, but will still be useful to look at constant-Q intensities at all commenurate **Q**-points.

2. run the PSF code, reading the trajectories from step 1.

3. profit

The variables in the input file are given below and all of the functionality of the code and the steps above are elucidated in the included examples in the example directory.

DEPENDENCIES

PARALLELISM

My code uses parallism to speed it up. It is only parallelized over **Q** points. e.g. if you want to calculate a $((\mathbf{Q}), \omega)$ colormap with 20 **Q**-points along the path, you could use 10 processors with each working on 2 of the **Q**-points. I used MPI (mpi4py) instead of mulithreading, which may have been a bad choice, because I wanted to learn how to use MPI for other stuff (i.e. DFT calculations). The drawback to MPI is that the data arent shared so you have to copy the trajectory onto each processor which uses alot of memory, which limits the scalability since the trajectories are usually very large (RAM wise). I might switch to threading in the future using multiprocessing lib if I feel like it.

To run the code with mpi4py, first make sure it is installed (easy with conda).

INPUT-FILE

OUTPUT-FILES