3-31-2021

# GIXStapose: Interactive and Reproducible Scattering Analysis of Simulation Snapshots

Jenny Fothergill
*Boise State University*

# ***GIXStapose:*** *An interactive structure-viewer alongside its simulated diffraction pattern*

BOISE STATE UNIVERSITY

**Authors** Jenny Fothergill, Chris Jones, Eric Jankowski

## INTRO

- GIXStapose enables grazing incidence X-ray scattering (GIXS) patterns to be visualized while interactively rotating chemical structures, especially periodic simulation volumes generated from molecular simulations.
- This functionality is useful for interactively identifying real-space chemical features that correspond to bright diffraction peaks and the rotation matrices that generate them.
- We hope this tool simplifies the process of understanding and creating publication-quality scattering patterns

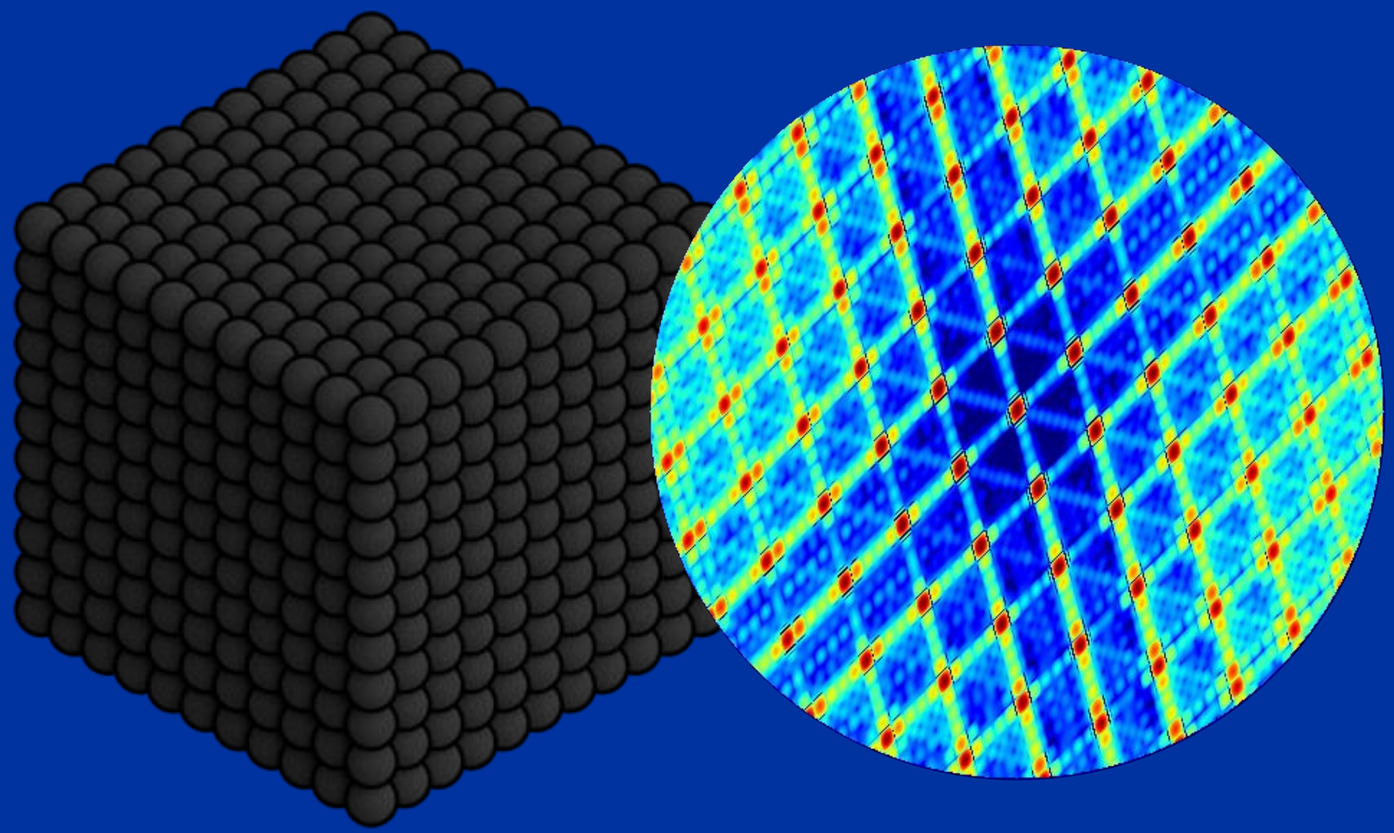
## PROJECT GOALS

1. Create a tool for image creation that aids in reproducibility
2. Collaborate with open-source software community to build on already existing frameworks:
   - The diffractometer package has been added as a module in the Freud analysis suite

## FUTURE WORK

- Use machine learning to automate the detection of bright peaks

# High-quality, reproducible images help users see the connection between morphology and bright diffraction peaks



## How is GIXS calculated?

Scattering patterns are calculated in a sequence of matrix transformations: rotations, shears, and Fourier transforms that are implemented in Numpy.

An in-depth explanation of the diffractometer package is given here.

## How are the pictures made?

The real-space morphology images are created with Fresnel, a ray tracer with a convenient Python API that uses C or CUDA under the hood.
The scattering patterns are displayed using Matplotlib.

## How can I use this tool?

GIXStapose can be used through a GUI, but all the functionality is modular and can be used as part of any simulation workflow.

Check out our example notebook to see a walkthough of creating figures and diffraction patterns from a perfect crystal and messy simulation data.

## Acknowledgements

This project would not be possible without the open-source software packages, Fresnel, MBuild, Matplotlib, and Numpy, and funding from the National Science Foundation (#1835593)