S. F. Li

Abstract:

Near-field high-energy x-ray diffraction microscopy, or nf-HEDM, is a full field orientation imaging technique. The method of nf-HEDM consists of collecting a set of diffraction images associated with a scan volume. Using the computational reconstruction code, IceNine, the orientation map of the imaged region is reconstructed using the diffraction images. This document serves as a tutorial of nf-HEDM reconstruction, complemented with a complete set of simulated diffraction data and a set of exercises. We hope that this document will serve as a cookbook for nf-HEDM experiments and help debug most of the commonly encountered issues with data processing.

Requirements:

Software: Matlab, C++ compiler, IceNine binary or source code

Reading: Li 2013, Li 2013. Suter 2006, Poulsen 2001. IceNine software manual.

Hardware: Any modern laptop will do for the simulation and simple reconstruction part. Probably a cluster if you want to do the full reconstruction. Get a starter allocation on an NSF cluster here: https://www.xsede.org/using-xsede#allocations

Familiarity of Matlab is expected but not required. We will be writing code in this tutorial. The advanced part of the tutorial will require C++, but they are built for “super users.”

Skills: I assume an advanced undergraduate level of mathematics and the ability to google.

Before Reconstruction:

Chapter 0 - IceNine

Compiling IceNine

Running IceNine

Typical directory structures

Configuration files

Input data

Chapter 1 - Getting an intuition of nf-HEDM:

Before running an nf-HEDM experiment, we should familiarize ourselves with the experimental geometry. To get a sense of the measurement, we can start by running simulations using IceNine with simulated microstructures. The key feature of nf-HEDM (or any spatially resolved rotating x-ray methods) is that spatial extent of the sample is encoded in the diffraction signal. In other words, the diffracting volume geometry is convolved with the diffraction signal. This reduces to simply the projection of the grain shape onto the 2D detector in the case of defect-free crystals. In the case of deformed samples, the situation becomes significantly more complex. To build some intuitions, we will start with some examples of simulated diffractions to determine 1) spatial sensitivity of the diffraction signal as a function of experimental parameters, 2) the expected diffraction patterns as a function of number of grains, 3) the effect of sample deformation to diffraction signals, and 4) a qualitative resolution limits of the reconstruction methods.

1. Spatial sensitivity

As a first step, we should get familiarized with IceNine’s way of specifying detector geometry. As previously mentioned, all configuration or configuration-like files in IceNine are written in parsed text files. In the case of the detector geometry, we have a DesyDet.2Det:

{

JUnitVector 1 0 0

KUnitVector 0 -1 0

BeamCenterJ 1024

BeamCenterK 1024

LabFrameLocation 8.07 0 0

LabFrameOrientation 90 90 0

NumJPixels 2048

NumKPixels 2048

PixelJLength 0.00153

PixelKLength 0.00153

}

# The vertical detector according to the directory naming

{

JUnitVector 1 0 0

KUnitVector 0 -1 0

BeamCenterJ 1065

BeamCenterK 1065

LabFrameLocation 18.07 0 0

LabFrameOrientation 90 90 0

NumJPixels 2048

NumKPixels 2048

PixelJLength 0.0059

PixelKLength 0.0059

}

Here, the two regions delineated by the curly bracelets specify two detectors. Each of the keywords corresponds to a value.

Examples:

1. As a first exercise, generate a set of diffraction images using Random30Micron.mic at LabFrameLocation (aka, L distance vectors) of L = 5, L = 7, L = 10, L = 20, L = 30.
2. Do the same thing for 500Grains\_g7\_sigma.mic. You might want to first look at the difference of the two microstructures:

[snp, sw] = load\_mic( ‘500Grains\_g7\_sigma.mic’);

plot\_mic( snp, sw, 3 0 );

Exercises:

1. Notice that the previous few samples are on axis. What happen when we have something that’s off axis instead? When doing an experiment at the synchtron, it is often useful to be able to “hack” a solution together quickly. We don’t necessarily have the luxury of generating a new microstructure from scratch. Use the microstructures provided above to generate an off axis structure.

[snp, sw] = load\_mic( ‘500Grains\_g7\_sigma.mic’);

snp(:, 1) = snp(:, 1) + x\_offset;

snp(:, 2) = snp(:, 2) + y\_offset;

write\_mic( snp, sw, ‘500Grains\_g7\_sigma\_shifted.mic’);

It’s sometimes more instructive to look at something with less grains. We can select a subsection of the mic file instead.

Flags = snp(:, 1) > 0.2 & snp(:, 1) < 0.3 & snp(:, 2) > 0.2 < snp(:, 2) < 0.3;

Subsection = snp( Flags, : );

write\_mic( snp, sw, ‘500Grains\_g7\_sigma\_shifted.mic’);

Some of the observations to be made:

Experimental Setup:

Reconstruction

Like any reconstruction methods, the quality of the reconstructed, or reduced data depends heavily on the experimental parameters.