manuscript jo5109 for review

JOURNAL OF
APPLIED
CRYSTALLOGRAPHY

Application of signal separation to diffraction image compression and serial crystallography

Jerome Kieffer, Julien Orlans, Nicolas Coquelle, Samuel Debionne, Shibom Basu, Alejandro Homs-PURON, Gianluca Santoni and Daniele De Sanctis

CONFIDENTIAL – NOT TO BE REPRODUCED, QUOTED NOR SHOWN TO OTHERS

SCIENTIFIC MANUSCRIPT

For review only.

Monday 08 July 2024

Category: *computer programs*

Co-editor:

Professor J. Hajdu

Laboratory of Molecular Biophysics, Institute of Cell and Molecular Biology, Uppsala University, Box 596, 75124 Uppsala, Sweden

Email: janos@xray.bmc.uu.se

Submitting author:

Jerome Kieffer

ADA, European Synchrotron Radiation Facility, CS 40220, Grenoble, 38043, France

Email: jerome.kieffer@esrf.fr

Page 1

Application of signal separation to diffraction image compression and serial crystallography

Jérôme Kieffer,[a]* Julien Orlans,[a] Nicolas Coquelle,[a] Samuel Debionne,[a] Shibom Basu,[b] Alejandro Homs,[a] Gianluca Santoni[a] and Daniele De Sanctis[a]

[a]*European Synchrotron Radiation Facility;71, avenue des Martyrs;CS 40220;38043 Grenoble Cedex 9 France*, and [b]*EMBL Grenoble; 71 avenue des Martyrs; CS 90181; 38042 Grenoble Cedex 9; France. E-mail: jerome.kieffer@esrf.fr*

REVIEW DOCUMENT

Abstract

We present here a real-time analysis of diffraction images acquired at high frame-rate (925 Hz) and its application to macromolecular serial crystallography. The software uses a new signal separation algorithm, able to distinguish the amorphous (or powder diffraction) component from the diffraction signal originating from single crystals. It relies on the ability to work efficiently in azimuthal space and derives from the work performed on pyFAI, the fast azimuthal integration library. Two applications are built upon this separation algorithm: a lossy compression algorithm and a peak-picking algorithm; the performances of both is assessed by comparing data quality after reduction with XDS and CrystFEL.

Page 2

1. Introduction

X-ray macromolecular crystallography is one of the most successful methods to determine the atomic structure of biological molecules. The achievable diffraction quality may often be limited by radiation damage. Although cryogenic conditions permit to extend the lifetime of crystals in the X-ray beam and to increase the maximum absorbed dose before inducing damage they may hinder physiologically relevant conformations. This limitation has renovated the interest for room temperature macromolecular crystallography by applying a more drastic approach to overcome the radiation damage problem by collecting data from thousands of small crystals, in what became known as serial crystallography. First developed at X-ray free electron laser sources (XFEL) (Chapman *et al.*, 2011; Boutet *et al.*, 2012) the method is currently applied also at synchrotron sources (Diederichs & Wang, 2017; Gati *et al.*, 2014; Nogly *et al.*, 2015; Owen *et al.*, 2017)

1.1. Serial crystallography using synchrotron sources (SSX)

Serial crystallography consists in exposing thousands of small crystals to the synchrotron beam only once in a serial way. Diffraction is collected with a very high flux density, in order to extract the most information from a single shot. This is in contrast with traditional rotational crystallography, where a complete dataset is collected from a single crystal rotated around one (or several) axis. Those SSX images contain only a fraction of the reciprocal space, as the crystal does not rotate inside the beam. To achieve a complete dataset, thousands of frames have to be collected, individually indexed and then merged. The high flux needed to collect all diffraction signal from a single crystal within a single exposure makes the SSX technique a good candidate to benefit from the 4^{th} generation synchrotron sources, such as the new ESRF-EBS update (Chaize *et al.*, 2018). However, macromolecular crystallography

REVIEW DOCUMENT

beam-lines (MX) have been extremely specialized towards rotational data-collection and thus require modifications to the experimental setup to perfom SSX experiments. The synchrotron serial crystallography beamline (ID29 at the ESRF) has been built to have a dedicated environment to perform SSX experiments with a high flux (using a larger energy bandwidth with a multilayer monochromator), a high-speed chopper (to produce X-ray pulses), several sample delivery methods and a fast detector.

1.2. JUNGFRAU *4M detector*

Macro-molecular crystallography has enormously progressed in the last decades with to the introduction of photon-counting detectors (Broennimann *et al.*, 2006). With their absence of read-out noise and their fast speed, the main limitation of photon-counting detectors is the achievable count-rate, i.e. how fast the electronic of a pixel is able to count arriving photons. Unlike photon-counting detectors such as the Eiger detector (Casanas *et al.*, 2016), the JUNGFRAU detector (Mozzanica *et al.*, 2016) is an integrating detector, thus, it is not limited by the count rate, even under the very intense flux expected when recording Bragg-peaks. To cope with this photons density, every single pixel implements an automatic gain switching mechanism (3 levels) which offers a precision of the order of one third of a keV on the higher gain mode, a precision of the order of one photon in the intermediate level and the ability to cope with thousands of photons in the lower gain mode, every millisecond. Since the JUNGFRAU detector is an integrating detector, dark current and flat-field correction have to be applied: every pixel has three, so called, pedestals and three gain values (one for each gain level). This large number of parameters per pixel combined with the gain-compression makes the pre-processing of raw signal challenging: the signal from a single pixel, initially stored on 16 bits of data, needs to be represented using 32 bits (as floating point or integer value), doubling the required bandwidth and the storage

REVIEW DOCUMENT

size (Leonarski *et al.*, 2020). The ID29 beamline features a JUNGFRAU 4M detector, operating at 1kHz, pace imposed by the chopper and synchronized with the photon bunches from the ESRF. At nominal speed, the detector will produce 20 GBytes of pre-processed data per second, making the data analysis and storage extremely challenging.

1.3. Requirements for online data processing

Serial crystallography is one of the techniques where online data processing is likely to have most impact: millions of images are to be collected but too fast to be saved. We expect only a few percentage of the frames to contain diffraction signal and out of them, a fraction to be indexed, integrated and used to solve the protein structure. Efficient processing of raw images is thus essential for SSX, with 4 levels of increasing complexity:

- image reconstruction with pedestal correction
- *veto*-algorithm: be able to sieve-out images with poor signal (Galchenkova *et al.*, 2024)
- save only pixels with diffraction signal
- precise location of peak position with indexation (Gasparotto *et al.*, 2023)
- real-time integration of diffraction peaks

.

Reconstruction and pedestal correction have been already described in Debionne *et al.* (2022*a*). This document will focus on the subsequent steps: the detection of signal is addressed in section 2, sparse data compression in section 3, and their application to serial crystallography in section 4 before concluding in section 5.

REVIEW DOCUMENT

2. Algorithm for the separation of the amorphous background from the Bragg peaks

2.1. Background scattering

The simplest implementation of Bragg peaks separation is to assume that the background signal originates from scattering of amorphous material (or from an isotropic powder), giving an isotropic signal that ideally presents only smooth variations. Before background subtraction, the raw signal has to be corrected for dark noise and for any systematic anisotropic effects such as polarization corrections. Unlike X-FEL, synchrotron X-ray beam is better characterized in energy and shows little to none pulse to pulse variability. All anisotropic correction can be easily modelled and taken into account.

The initial implementation of signal separation in pyFAI (Kieffer & Wright, 2013) was relying on a 2D polar transform followed by a median filter in the azimuthal dimension to calculate the amorphous scattering curve. Although this method has been successfully used for large dataset analysis (Bordet *et al.*, 2021), it presents four major drawbacks:

- The 2D averaging mixes the signal originating from several pixels and blurs the signal.
- Pixel-splitting is needed to leverage the Moiré effect in the 2D averaging, but this further increases the blurring (Fang *et al.*, 2020).
- The 1D curve obtained after the application of the median filter shows sharp jumps from one azimuthal bin to its neighbour.
- The median filter is computationally heavy since it is required to sort out every azimuthal bin.

We improved on this by developing a new, efficient way of performing the azimuthal averaging (including the associated uncertainty propagation).

REVIEW DOCUMENT

2.2. Efficient azimuthal averaging and uncertainties evaluation

2.2.1. Pre-processing: The first step of the analysis consists in applying a pixel-wise correction for dark current and several normalization corrections (Kieffer *et al.*, 2020):

$$I_{cor} = \frac{signal}{norm} = \frac{I_{raw} - I_{dark}}{F \cdot \Omega \cdot P \cdot A \cdot I_0} \quad (1)$$

In equation 1, the numerator (referred as *signal* hereafter) is the subtraction of the dark current I_{dark} from the the detector's raw signal I_{raw}. The denominator (hereafter *norm*) is a normalization factor composed of the product of F: a factor accounting for the flat-field correction, Ω: the solid angle subtended by a given pixel, P: the polarisation correction term and A: the detector's apparent efficiency due to the incidence angle of the photon on the detector plane. For integrating detectors, photons with high energy see longer sensor path with larger incidence angles compared to the normal thickness, thus they have a higher detection probability. Intensity is also normalized by the incoming flux I_0, but being it independent from the pixel position, this correction can be applied when convenient.

2.2.2. Azimuthal averaging: Historically, the azimuthal averaging was implemented using histograms (Hammersley *et al.*, 1996). Since the geometry of the experimental setup is fixed during the acquisition, a look-up table listing all pixels contributing to each azimuthal-bin can be built and used to speed-up calculations (Kieffer & Ashiotis, 2014). The azimuthal transformation being a linear transformation, it can be implemented as a matrix multiplication, with a sparse-matrix representing the transformation and a dense vector containing the flattened view of the diffraction image. The compressed sparse row (CSR) matrix representation is preferred for its efficiency in performing dot products with dense vectors (Toledo, 1997). The coefficients $c_{i,r}$ of the matrix are the fraction of area of a pixel i falling into the radial bin r. In the

REVIEW DOCUMENT

case where pixel splitting is deactivated these coefficients ($c_{i,r}$) are always one (and zero elsewhere) since each pixel contributes to a single bin. The sparse matrix multiplication can be used to sum efficiently values for all pixels belonging to the same bin. The summed signal divided by the summed normalization provides the weight-averaged intensity over all pixels falling in the bin at the distance r from the center, as formalized in equation 2:

$$< I >_r = \frac{\sum\limits_{i \in bin_r} c_{i,r} \cdot signal_i}{\sum\limits_{i \in bin_r} c_{i,r} \cdot norm_i} \tag{2}$$

2.2.3. Uncertainty evaluation from Poisson distribution. Photon counting detectors, such as Eiger detectors, suffer from hardly any error beside the counting uncertainty which is often referred to as Poisson statistics. This statistical law is described by a single parameter λ which is related to the mean μ and standard deviation σ from a normal distribution by: $\lambda = \mu = \sigma^2$. Other sources of noise, like the dark current noise in the case of an integrating detector, superimpose quadratically on the Poisson noise for integrating detectors, as presented in equation 3:

$$var_I = (\sigma_I)^2 = < I_{raw} > + (\sigma_{dark})^2 \tag{3}$$

During the azimuthal integration, the coefficient of the sparse matrix needs to be squared at the numerator when propagating the variance (equation 4) to have uncertainties σ proportional to the fraction of the pixel considered.

$$(\sigma_r(I))^2 = \frac{\sum\limits_{i \in bin_r} c_{i,r}^2 \cdot \sigma_i^2}{\sum\limits_{i \in bin_r} c_{i,r} \cdot norm_i} \tag{4}$$

One should distinguish the *uncertainty of the mean* (sometimes referred to as the standard error of the mean, *sem*), which describes the precision with which the mean is known (and described in Kieffer *et al.* (2020)), from the *uncertainty of the pixel value* (often referred to as standard deviation, *std*) which describes the uncertainty

REVIEW DOCUMENT

with which the pixel value is known. Those two value differ only by the square root of the number of measurements in the case of an arithmetic mean: $sem = std/\sqrt{N}$ with N being the number of pixel contributing to the bin. When considering the weighted average, the previous formula becomes:

$$sem_r = std_r \frac{\sqrt{\sum_{i \in bin_r} c_{i,r}^2 \cdot norm_i^2}}{\sum_{i \in bin_r} c_{i,r} \cdot norm_i} \tag{5}$$

Thus, the more data points are collected, the more precisely the mean value is known but the uncertainty for a given point remains the same. Since this document focuses on the uncertainties of pixel values, the *standard deviation* will systematically be used from here on.

2.2.4. Uncertainty evaluation from the variance in a bin. Unlike photon counting detectors, most detectors do not follow the Poisson distribution and therefore the definition of a relation $\sigma^2 = f(I)$ is not simple, if possible at all. The integrating JUNGFRAU detector has a complex gain switching mechanism (Leonarski *et al.*, 2020) which makes this equation complicated. A generic approach is proposed to measure the variance in every single azimuthal bin:

When considering the diffraction of an isotropic compound (liquid, amorphous or perfect powder), all pixels contributing to the same radial bin should see the same flux of photons (after correction of anisotropy like polarization) and the deviation to their intensities can be used to estimate the uncertainty. This approach is of course limited when considering the signal coming from few large crystalites (where rings becomes spotty) but it provides an upper bound for the uncertainty. Variances (thus standard deviations) are usually obtained in a two steps procedure: one pass to calculate the average value (equation 2) and a second to calculate the deviation to the average (equation 6). This double pass approach can be implemented using sparse matrix

REVIEW DOCUMENT

multiplication. It requires twice the access to each pixel value, and extra storage space, but it is numerically robust (i.e. not prone to numerical-error accumulation).

$$(\sigma_r(I))^2 = \frac{\sum\limits_{i \in bin_r} c_{i,r}^2 \cdot norm_i^2 \cdot (\frac{signal_i}{norm_i} - < I >_r)^2}{\sum\limits_{i \in bin_r} c_{i,r}^2 \cdot norm_i^2} \tag{6}$$

and:

$$(\sigma_r(< I >))^2 = \frac{\sum\limits_{i \in bin_r} c_{i,r}^2 \cdot norm_i^2 \cdot (\frac{signal_i}{norm_i} - < I >_r)^2}{(\sum\limits_{i \in bin_r} c_{i,r} \cdot norm_i)^2} \tag{7}$$

Single pass implementations of variance calculation are faster than double pass ones since they access pixels only once and offer, in addition, the ability to perform parallel reductions (Blelloch, 1996), i.e. work with blocks of pixels. Schubert & Gertz (2018) present a complete review on the topic, which introduces a formalism adapted here for crystallography. Assume the weight for a pixel being $\omega_i = c_i \cdot norm_i$. If P is a partition of the ensemble of pixels falling into a given azimuthal bin, let Ω_P, V_P and VV_P be the sum of weights (Eq 8), the weighted sum of V (Eq. 10) and the weighted sum of deviation squared (Eq 11) over the partition P:

$$\Omega_P = \sum_{i \in P} \omega_i = \sum_{i \in P} c_i \cdot norm_i \tag{8}$$

$$\Omega\Omega_P = \sum_{i \in P} \omega_i^2 = \sum_{i \in P} c_i^2 \cdot norm_i^2 \tag{9}$$

$$V_P = \sum_{i \in P} \omega_i \cdot v_i = \sum_{i \in P} c_i \cdot signal_i \tag{10}$$

$$VV_p = \sum_{i \in P} \omega_i^2 \cdot (v_i - V_P/\Omega_P)^2 \tag{11}$$

The weighted average and associated variances are then expressed as:

$$< I >_P = \frac{V_P}{\Omega_P} = \frac{\sum\limits_{i \in P} c_i \cdot signal_i}{\sum\limits_{i \in P} c_i \cdot norm_i} \tag{12}$$

REVIEW DOCUMENT

$$std^2 = (\sigma_P(I))^2 = \frac{VV_P}{\Omega\Omega_P} \tag{13}$$

$$sem^2 = (\sigma_P(< I >))^2 = \frac{VV_P}{\Omega_P^2} \tag{14}$$

In this formalism, equations 2 and 12 on one side and equations 6 and 13 are actually equivalent. Schubert & Gertz (2018) present the way to perform the union of two sub-partitions A and B of a larger ensemble which opens the doors to parallel reductions, which are especially efficient when implemented on GPU:

$$\Omega_{A\cup B} = \Omega_A + \Omega_B \tag{15}$$

$$V_{A\cup B} = V_A + V_B \tag{16}$$

$$VV_{A\cup b} = VV_A + \omega_b^2(v_b - \frac{V_A}{\Omega_A})(v_b - \frac{V_{A\cup b}}{\Omega_{A\cup b}}) \tag{17}$$

$$VV_{A\cup B} \approx VV_A + VV_B + \frac{\Omega\Omega_B(V_A \cdot \Omega_B - V_B \cdot \Omega_A)^2}{(\Omega_{A\cup B}.\Omega_A.\Omega_B^2)} \tag{18}$$

While equations 15 and 16 are trivial, equation 17 describes the nominator of the variance of an ensemble when adding an extra member b to the A. Unfortunately, the slight difference of formalism between Schubert & Gertz (2018) and this work prevent some simplification to occur and leads to the approximate numerator of the variance (VV) in the case of the union of two ensemble A and B (eq. 18, used in OpenCL reduction [1]. A numerical stability issue can arise from it when V_A or V_B are very small and this issue is addressed by using double-precision arithmetic when implemented on CPU and double-word arithmetic when running on GPU (Joldes *et al.*, 2017).

[1] see https://github.com/silx-kit/pyFAI/blob/main/doc/source/usage/tutorial/Variance/uncertainties.ipynb

REVIEW DOCUMENT

2.2.5. Comparison of uncertainty models: Figure 1 (right hand side) presents the uncertainties (for the pixel value) as calculated from a background frame with pure Poisson noise (displayed on the left hand-side, synthetic data) using the two algorithms previously described: the Poisson model or calculated from the variance in the azimuthal bin. While the two curves show similar amplitude, except in the corner of the detector where very few pixels contribute to each of the azimuthal bin, the variability of the "azimuthal" model is much more important from one bin to the neighboring one.

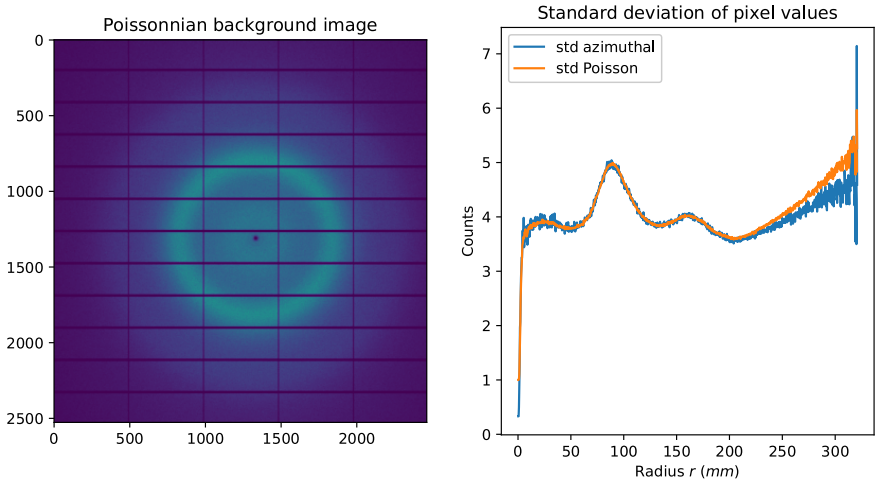

Fig. 1. (a) Simulated diffraction frame with pure azimuthal Poisson noise of a Pilatus 6M detector and (b) uncertainties for pixel intensity as measured with the distance to the mean (azimuthal-model, blue) or from the Poisson model (orange).

2.3. Histogram intensity

The figure 2a presents the diffraction from a single crystal of insulin collected with a Pilatus 6M detector and several curves obtained from azimuthal integration of those data: figure 2b is the azimuthally integrated signal (blue curve) where Bragg peaks are seen as spikes on top of a smooth background. Plot 2c presents the uncertainties measured according to the Poisson distribution (orange curve)or the deviation in the ring (blue curve). The later presents much larger values since Bragg peaks contribute

a lot to the deviation despite they represent few pixels: this highlights the sensitivity of the mean/std to outliers. In the plot 2d are presented histogram of pixel intensity for pixels laying at 87mm and 160mm from the beam center. Each of those histograms is composed of a bell-shaped distribution, centered on the average value with negative outliers tagged with the pixel value -1 (this is specific to the Pilatus detector), and few positive outliers which are usually Bragg peaks. Those histograms in figure 2d have been fitted with a Gaussian curve and both the center (μ) and the width (σ) of the curve match roughly with the average (in 2b) and uncertainties (in 2c).

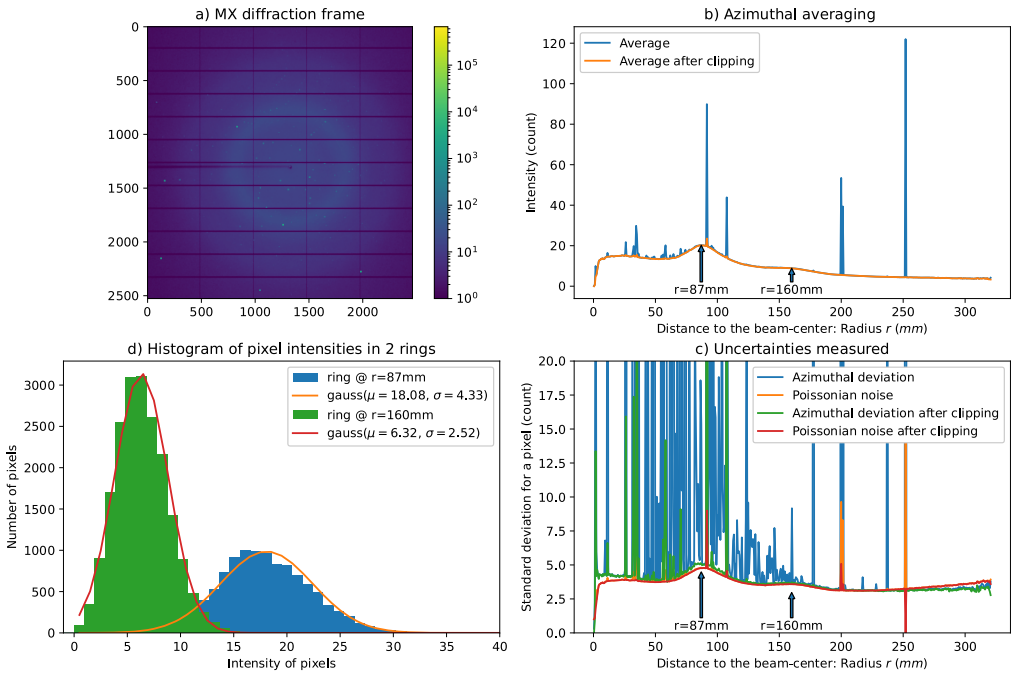

Fig. 2. Single crystal diffraction frame obtained from insulin with a Pilatus 6M (a) with the azimuthally averaged signal (b), before and after clipping data. Uncertainties are presented in (c) when calculated assuming a Poissonian error model (orange, red) or when measuring the deviation within all pixels in a ring (green, blue). Subplot (d) presents the histogram of intensities for two rings at r=87mm and r=160mm from beam center with the distribution fitted as Gaussian curves: $g(x) \propto exp(-\frac{(x-\mu)^2}{2\sigma^2})$.

The core idea of the algorithm for background extraction is to model the distribution of background pixels. Unlike Bayesian statistics (Sivia & Skilling, 2006) where the

cost function is usually tuned to weight less outlier, here, those outliers are simply flagged and discarded. Positive outliers can reasonably be assigned to Bragg peaks and negative outliers to shadows or defective pixels. The distribution is recalculated after discarding pixels for which intensity differ from the average value by more than n times the standard deviation (Eq. 19), where n is called the Signal to Noise Ratio (SNR). This clipping re-centers the distribution of remaining pixels since the mean is sensitive to outlier values:

$$|I - < I >| > n \cdot \sigma(I) \tag{19}$$

The orange plot in figure 2b presents the average after having discarded those outliers, and the orange and green curve of figure 2c are the uncertainties calculated after this clipping. After clipping, the average and uncertainties curves have lost most of their spikes, which means that most Bragg peaks and shadowed pixel have been discarded.

2.4. Sigma-clipping

The *sigma-clipping* algorithm consists in applying this outlier rejection several times. If the initial distribution is mono-modal, this algorithm enforces gradually the data to be sampled symmetrically around the maximum probability, which is likely to look like a normal distribution. If the initial distribution is more complicated (typically multi-modal), the necessary larger standard-deviation will prevent most outlier pixels from being rejected, making it more conservative. The sigma-clipping algorithm takes two parameters: the number of iterations and the rejection cut-off (SNR). Despite the execution time is proportional to the number of iteration of sigma-clipping, iterations should continue until no more outliers are found, so that the background data can be treated assuming a normal distribution. Since the loop exits as soon as no more outliers were discarded at the clipping step, having an arbitrary large number of iteration is not really an issue for the execution time and the number of actual iteration is usually

REVIEW DOCUMENT

few (3 is commonly observed).

2.4.1. Limits of the Poissonian approach: The evaluation of uncertainties based on the variance within a radial shell (azimuthal model) has been developed after numerical artifacts were discovered while performing sigma-clipping with a Poissonian approach. Some azimuthal bins were showing no pixel contribution at all and thus appeared without any mean nor uncertainties, jeopardizing the complete background extraction algorithm. This artifact was directly linked to the usage of Poisson statistics and can be demonstrated with a simple distribution of 2 pixels, having values 1 and 199. The mean of this distribution is 100 and the standard deviation is also close to 100 while the uncertainty derived from a Poissonian law would be close to 10 (i.e. $\sqrt{100}$). With the azimuthal error model, both pixels are a 1σ from the mean while with the Poissonian error-model, pixels are at 10σ. This explains why bins featuring strong Bragg-peaks on top low background got completely emptied from any contributing pixels when sigma-clipping was performed assuming Poissonian noise.

Unlike the Poisson error-model, the uncertainties obtained from the azimuthal-model are resilient to diffraction data coming from several types of samples but show much more variability from one bin to its neighbor (Fig. 1). PyFAI introduces an *hybrid* error-model which uses the azimuthal error-model for the sigma-clipping stage which trims the ensemble of pixels to become mono-modale. The uncertainties are then calculated using a the Poisson error-model on the trimmed ensemble.

2.4.2. Clipping threshold can be automatically calculated based on a variation on Chauvenet's criterion (Maples *et al.*, 2018) where one would accept to discard only a single pixel in a ring with a signal already following a normal law. Thus, the threshold value is adapted to the *size* of the distribution, i.e. the number of pixels in each

REVIEW DOCUMENT

ring (Eq. 20), which can reach several thousands and shrinks with iterations. Typically the numerical value for this cut-off varies from 2 to 4.

$$SNR_{chauv.} = \sqrt{2log(\frac{size}{\sqrt{2\pi}})} \tag{20}$$

The worse case scenario for sigma-clipping corresponds to an initial distribution very far from a normal distribution, as the bimodal distribution seen in the previous paragraph. Another challenging situation occurs close to the detector corners where the background signal is low and the size of the distribution is decreasing. For example this cut-off parameter increases from 2.7 to 3.5 when the size of the ensemble increases from 100 to 1000 elements. So, for a Poissonian detector and a low count rate of one ($\lambda = \mu = \sigma^2 = 1$), any pixel with intensity greater than 4 is discarded with the ensemble of 100, greater than 5 for the ensemble of 1000 pixels.

3. Application to single crystal diffraction image compression

Diffraction images from a single protein crystal exhibit usually an isotropic background on top of which Bragg peaks appear. The sigma-clipping algorithm can be used to select the background level and more importantly the associated uncertainty.

This lossy compression algorithm consists in picking (and storing) pixels which intensity is above the average background value (μ) plus n standard deviation (σ). This cut-off value n (also called SNR_{pick}) controls the amount of data to store and provides also a hint for the compression ratio achievable, assuming a normal distribution has been enforced at the sigma-clipping stage: 16% of the pixel are to be recorded with $n = 1$; 2.3% for $n = 2$ and only 0.13% for $n = 3$ as depicted in figure 3.

REVIEW DOCUMENT

Page 16

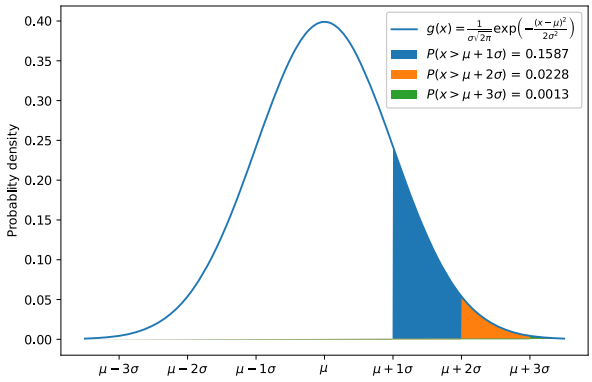

Fig. 3. Normal distribution and probability of having pixels with intensity above certain thresholds.

The storage space is much larger than just the pixel intensity since one needs to store as well its position in detector coordinates. Since diffraction analysis software are performing some kind of noise-level analysis, the background signal has to be regenerated with intensity and noise similar to the original data. Thus, the average intensity per azimuthal bin together with the associated uncertainties needs to be stored.

3.1. Sparsification

The sigma-clipping algorithm was originally written for real-time sparsification of single crystal diffraction data and its integration into the LIMA2 detector control system (Petitdemange *et al.*, 2018) for the JUNGFRAU 4M detector used at ESRF ID29 is described in (Debionne *et al.*, 2022*b*). The real-time constrain imposed to develop code running on GPU since those devices are several times faster than equivalently priced processors. All algorithms were developed in OpenCL (Khronos, 2008) and implemented in the *pyFAI* software package. A command line tool called 'sparsify-Bragg' is available for testing off-line.

REVIEW DOCUMENT

All the pixel coordinates and intensities are stored in a HDF5 container (The HDF Group, 2000-2021) following the NeXus convention (Könnecke *et al.*, 2015), together with a snippet of Python code explaining how to rebuild the dataset. All sparse datasets (averaged and uncertainties curves, pixel coordinates, etc..) are compressed with the bitshuffle-LZ4 (Masui *et al.*, 2015) lossless compression.

3.2. Densification

Since no crystallographic software package can deal with this sparse format, the densification code was developed to regenerate initial frames and the 'densify-Bragg' program was made available as part of the FabIO (Knudsen *et al.*, 2013) software package (version $\geq$0.12). The software constrains for this densification code are very different from the one for sparisfication since this code can be used by the final users. For this reason 'densify-Bragg' was optimized to run on multi-core CPU. Maybe an important consideration is whether, regardless from the file format, it is necessary to reconstruct the background or not. In fact, some crystallographic reduction program like CrysAlisPro (Rigaku Oxford Diffraction, 2015) provide a better result with noiseless background while XDS (Kabsch, 2010), which performs a deep noise analysis, needs to have the noisy background propely restored. Shaded regions are never reconstructed properly and should be masked adequately in the reduction software.

3.3. Performances

The performances for a lossy compression algorithm are to be evaluated along many directions: compression and decompression speeds, compression ratio and degradation of the recorded signal. In the following example we present the sparsification of a Lysozyme (HEWL) dataset obtained using a traditional oscillation data-collection. Data were collected on an Eiger 4M detector (Dectris, 2014), selected for its similarity

REVIEW DOCUMENT

in size and performances with the JUNGFRAU detector. Those data are then densified again to regenerate the data and processed in XDS (Kabsch, 2010). Data quality indicators are finally compared between the original dataset and the one which went through the lossy compression presented here.

3.3.1. Compression ratio After sparsification (picking cut-off: 2σ, error-model: Poissonnian), the dataset still weights 103MB which represents a 15x compression ratio compared to the standard procedure. For conformance with the state of the art, the reference dataset was re-compressed using the bitshuffle-LZ4 algorithm (Masui *et al.*, 2015), for which the 1800 frames weight 1500MB (instead of the 5000GB of the original files compressed in LZ4).

The maximum theoretical compression ratio for 2σ is 22x (figure 3, neglecting the storage of the background data and effects of the lossless compression). To evaluate the effective maximal compression ratio, the dataset was median-filtered along the image stack to produce an image without peaks. A background dataset of 1800 such images sparsifies into a 11MB HDF5 file which represents a compression ratio of 136x ! Indeed, only 19 pixels were saved per frame and the compressed numerical values are mostly the same, which compresses great with bitshuffle-LZ4.

In production conditions, one would like to tune this threshold to the minimal value which guarantees a compression ratio large enough to allow the storage of all data. For ESRF-ID29, where a JUNGFRAU 4M can operate close to 1 kHz, the pedestal+gain pre-processing convert 16-bits integers into 32-bits floating point values, doubling the bandwidth for data saving. The detector outputs the data via 8× 10Gbit/s network links and the storage is performed via a single 25 Gbit/s link, making a minimum compression ratio of 6.4x which is equivalent to a cutoff at 1.4σ (cf. figure 3).

REVIEW DOCUMENT

3.3.2. Compression speed: The compression speed has been measured on the computer designed for online data-reduction of the JUNGFRAU detector (Debionne *et al.*, 2022*b*): an IBM AC922 using two Power9 processors and two (with possibility to upgrade to four) Nvidia Tesla V100 GPU. The sequential execution of the code on the GPU takes about 4 ms to process one image and uses one single CPU-core and a quarter of a GPU. In online condition, multiple parallel processes are expected to achieve much higher frame-rate, since the nominal speed for the JUNGFRAU is 1kHz.

3.3.3. Decompression speed: The decompression of those data should typically be performed on a standard workstation (here running two Intel Xeon Gold 6134 CPU @ 3.20GHz): the reconstruction speed is found to take 30 s for the full dataset, while writing of the densified dataset (with bitshuffle-LZ4 compression) takes 45s. Densification is thus faster than writing on disk. The reading time of the input sparse dataset is negligible (<2s).

3.3.4. Quality of the restored dataset: The densified dataset was processed via XDS and the summary indicator for the quality of the results are compared with the one coming from the reduction of the original dataset. Since those integrator are measured on integral peaks with $I/\sigma > 3$ and the sparsification was performed with a cut-off of 2, those result should be almost unaffected, which is confirmed in the table 1.

Table 1. *Quality indicators after peak integration and averaging using XDS (Kabsch, 2010). Lysozyme (HEWL) dataset provided by Dectris for advertising their Eiger-4M detector (Dectris, 2014).*

Indicator	Initial dataset			Lossy compressed dataset (2σ)		
	2.91Å	2.06Å	all	2.91Å	2.06Å	all
Completeness	98.8	90.8	93.8%	99.8	90.6	93.5%
$R_{obs} = \frac{\sum \lvert I_{h,i} - I_i \rvert}{\sum I_{h,i}}$	9.9	57.3	12.5%	9.2	61.2	11.4%
$R_{expected}$	8.8	73.2	15.0%	8.2	68.7	12.1%
R_{meas} (Diederichs & Karplus, 1997)	10.3	61.2	13.2%	9.6	65.5	12.0%
$CC_{1/2}$ (Karplus & Diederichs, 2012)	99.7	94.0	99.7	99.6	95.4	99.7
$< I/\sigma >$	25.80	5.39	10.52	26.33	4.09	10.17

.

REVIEW DOCUMENT

Those results are very different from the one described in Galchenkova *et al.* (2024) where such lossy data compression technique is described as detrimental for the quality of the reduced data. The main difference between the two approaches is that here, no peak identification is performed, thus no minimum number of contiguous and intense pixels are to be found. So more pixels values get saved, trading disk space for quality of the recorded dataset.

4. Application to serial crystallography

A classical way of pre-processing serial-crystallography data is to shrink the amount of data by sieving-out empty or bad frames, only keeping the frames which deserve processing. This is the role of the "veto-algorithms".

The sigma clipping algorithm provides us with the background (average and deviation) and is used to pick pixels which are likely to be Bragg peaks. For this, several additional checks are performed on the local neighbourhood which is a small square patch (typically 3x3 or 5x5 pixels):

- The considered pixel is the maximum of the local neighborhood.
- Enough of pixels in the local neighborhood satisfying the SNR condition: typically 2 to 5 pixels out of 9 or 25 are intense enough.

For each peak, the pixel coordinate, the precise centroïd, the sum of data and its propagated deviation are recorded and reported. Those peak-position are saved into a HDF5 file (represented figure 4) following the CXI format (Maia, 2012) which can be can be read from CrystFEL (White *et al.*, 2012). CrystFEL is the most popular software to process serial crystallography data, allowing to swap peak-picking algorithms (Zaefferer, 2000; Barty *et al.*, 2014; Hadian-Jazi *et al.*, 2021) and indexing tools (Kabsch, 2010; Powell *et al.*, 2013; Ginn *et al.*, 2016; Gevorkov *et al.*, 2019; Gevorkov *et al.*, 2020). Since indexing is compute intensive, CrystFEL can distributes calculation

REVIEW DOCUMENT

over several cores or nodes.

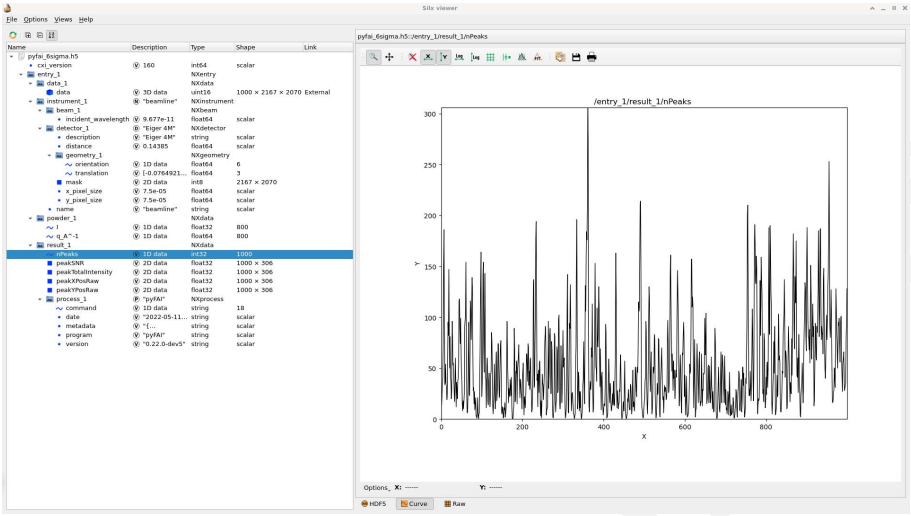

Fig. 4. Peak-picking CXI-file produced by pyFAI and visualized with the *silx viewer* (Vincent *et al.*, 2021). The left-hand side contains the HDF5 tree structure while the right-hand side presents the default plot with the number of peaks found per frame.

The serial crystallography beamline at the ESRF (ID29) uses a LImA2 monitor (Debionne *et al.*, 2022*a*) as visualization tool. It is inspired from NanoPeakCell (Coquelle *et al.*, 2015) for online visualization and feeds back information to check if peaks found correspond actually the crystal lattice expected for the sample. We will first compare the peak-picking algorithm with some reference implementation on a single frame before evaluating the quality of the picked points on a serial-crystallography dataset where the indexation rate will be used as quality indicator.

4.1. Comparison of picked peaks

Figure 5 presents the comparison between the original *peakfinder8* described in Barty *et al.* (2014), interfaced in Python via OnDA (Mariani *et al.*, 2016) and the version implemented into pyFAI on the same Pilatus 6M image, already used in figure 2.

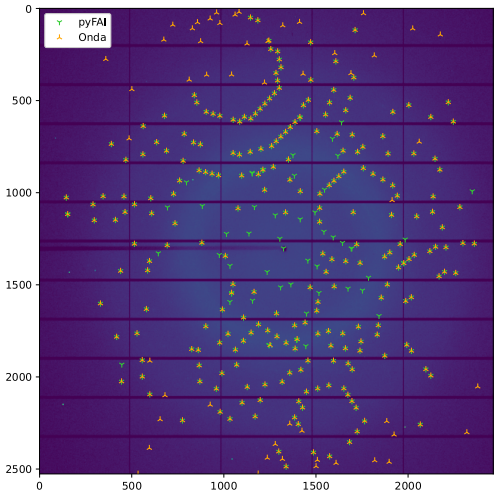

Fig. 5. Comparison of the reference *peakfinder8* interfaced with Onda (in orange, execution time 300ms) and the version from pyFAI (in green, execution time 10ms) on top of a Pilatus 6M diffraction frame of an insulin crystal.

In figure 5, most peaks found by both implementation match and superimpose with Bragg-peaks. There are more green peaks (found by pyFAI) closer to the beam center while more orange peaks (found by Onda) are located in the outer-shell. This plot was made with a minimum SNR of 3 and a noise level of 1 since the Pilatus detector is Poissonian. Peaks were registered if four pixels meet the SNR criterion in a 5x5 pixels patch around the peak. Those parameters have been tuned to obtain a comparable number of peaks with both implementations: 290 with pyFAI and 293 with Onda. The similarity of those figures allows a direct comparison of peaks found per resolution shell, histogram plotted in figure 6.

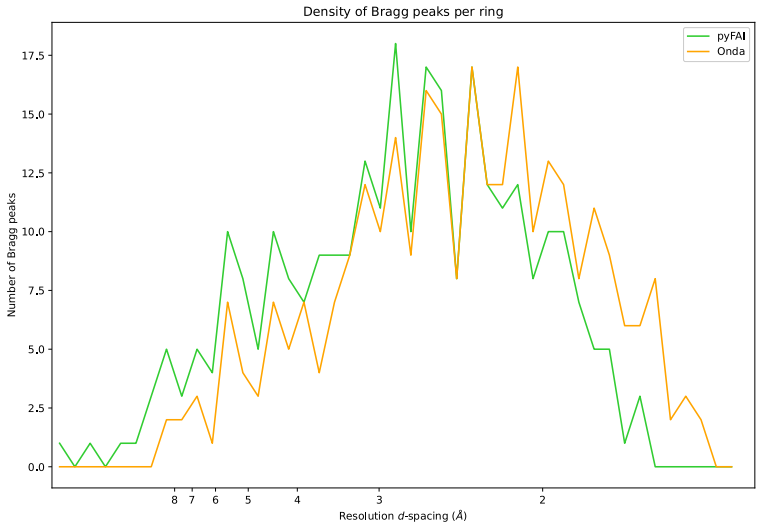

Fig. 6. Number of peaks found in the different resolution shell for the *peakfinder* implemented in pyFAI and in Onda. The width of each radial-shell is one inverse nanometer in $q-$space.

The analysis of the histogram in figure 6 confirms that the implementation in pyFAI is getting more points closer to the beam-center while the reference implementation is picking more point at larger q-values. This is likely due to the curvature of the Debye-Scherrer ring: the original version of peakfinder8 evaluates the variance in a neighborhood defined by some radius around the point of interest and the variance is higher close to the beam-center due to the important curvature of those rings. On the opposite, pyFAI knows about this curvature and measures the variance along the ring. Points picked by Onda at larger q-values do not look like Bragg-peaks but this could be a side-effect of the parameter tuning to get the same number of peaks for both algorithms.

A word on performances: the Python binding in Onda to the peakfinder8 algorithm from Cheetah runs in 180ms on a high-end server CPU (AMD Epyc 7262) and 300ms on a workstation (Intel Xeon E5-1650 v4). The version available in pyFAI was designed

in OpenCL (Khronos, 2008; Stone *et al.*, 2010; Klöckner *et al.*, 2012) and runs best on GPUs: 30ms on a AMD Vega 56 and in 10 ms on a Nvidia RTX A5000. Interestingly, an entry level GPU (Nvidia GT1030), which is seven times cheaper than the Epyc processor, shows the same performances (but this CPU has eight cores).

4.2. Quality of the peakfinder on serial crystallography data

The quality of peaks extracted with this algorithm is finally evaluated on a serial-crystallography dataset: tiny crystals of egg white lysozyme (HEWL) were deposited on a SiN membrane and this membrane was scanned in front of the beam at the ID30A3 beamline at the ESRF with an Eiger 4M detector.A fraction of 1000 frames of the dataset was used as a probe and was indexed with the *indexamajig* tool from CrystFEL. Since all frames show Bragg-peaks (figure 4) the number of indexed frames can be seen as a quality indicator of the peak-picking algorithm used, when all other parameter remain unchanged. The indexation was performed with the XGANDALF algorithm (Gevorkov *et al.*, 2019) with default settings from CrystFEL 0.10.1 and was provided with the cell parameter: tetragonal $a = b = 78.77\text{Å}; c = 39.04\text{Å}; \alpha = \beta = \gamma = 90^o$. The table 2 compares the number of frames properly indexed with the different picking algorithms available in CrystFEL and with the algorithm presented here.

Table 2. *Indexation rate obtained with* XGANDALF *(Gevorkov* et al.*, 2019) from peak positions extracted with different picking algorithms available from CrystFEL (White* et al.*, 2012) on a subset of 1000 frames of microcrystals of Lysozyme (HEWL) collected with an Eiger 4M at ESRF-ID30A3.*

Peak-picking method	XGANDALF (default)		XGANDALF (fast)	
	Index. rate	run-time	Index. rate	run-time
Zaef (Zaefferer, 2000)	10.0%	2178 s	10.0%	430 s
PeakFinder8 (Barty *et al.*, 2014)	49.5%	10397 s	48.5%	1757 s
PeakFinder9 (Gevorkov *et al.*, 2024)	44.2%	8328 s	43.5%	1436 s
RobustPF (Hadian-Jazi *et al.*, 2021)	22.4%	6314 s	21.2%	1628 s
pyFAI (this contribution)	49.7%	9325 s	49.2%	1595 s

Since the Eiger detector is a counting detector, the global threshold for algorithms *zaef* and *peakfinder8* had to be lowered (to 50, which is the maximum of the back-

REVIEW DOCUMENT

Page 25

ground signal on any frame) and the the default SNR value was used for *zaef*, *peakfinder8*. The same SNR value of five was used for *pyFAI*. Default parameters were used for *PeakFinder9* and *RobustPeakFinder* algorithms. The reported run-time correspond to the execution time on a single core of an Intel Xeon Gold 6134.

The indexing rate obtained with the algorithm from pyFAI is in par with the reference implementations like *peakfinder8* or *peakfinder9* available from CrystFEL. Since time is mostly spent in indexing, sets of peaks which are simpler to index present lower run-time as fewer retry were needed. Retries can be deactivated but they increase significantly the success of indexing. Table 2 presents also the indexing performances when using the option *–xgandalf-fast-execution* from *indexamajig*, which is five to six times faster and exhibits a limited degradation of the indexing rate. The peak extraction in pyFAI is about as fast as the sparsification, so it can be used online to perform the pre-analysis and provide peaks to NanoPeakCell. Nevertheless the executable 'peakfinder' available from pyFAI (offline tool) has a total execution time which is much larger: about 30 s for 1000 frames, most of which is spent in reading and writing the different HDF5 files.

4.3. Peak count as veto algorithm

Since the background extraction and peak finding are performed in real-time on the serial-crystallography beamline ID29 at ESRF, the information about the number of Bragg-spots can be used to assess the quality of each individual image and the acquisition system can decide to discard the frame depending on the number of peaks and a live-adjustable threshold.

Since the beginning of operation of ESRF-ID29, in 2022, the beamline operated with the *veto* algorithm deactivated, the number of peaks found was just recorded for future exploration. The figure 7 presents the indexation rate of "hit" (in blue) and "non-

REVIEW DOCUMENT

hit" frames (in orange) when changing the threshold for the minimum peaks-count per frame. The expected compression rate is displayed in green. The dataset consists of 80000 lysozyme micro-crystals between two mylar films, raster scanned with a x-ray beam of 11.56 keV at ESRF-ID29. The offline analysis has been performed with CrystFEL (v0.11.0) using ‘peakfinder8‘ and ‘xgandalf‘ as indexer.

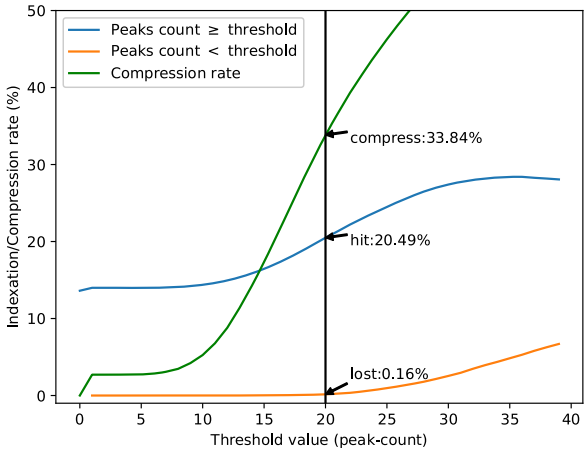

Fig. 7. Indexation rate of frames which would be considered as "hit" or "non-hit" as function of the peak-count threshold. The green curve presents the disk space which could have been saved. The sample dataset consists of 80000 frames of lysozyme micro-crystals indexed off-line with Xgandalf in CrystFEL.

Since the JUNGFRAU is an integrating detector, the detector has relatively more background noise than a photon counting detector: here, data were saved with 8 ADU/photon, making the compression rate of hit and non-hit frames similar. In this example, discarding frames with less than 20 peaks of would have allowed to save one third of the network bandwidth and disk space, loosing only 0.16% of frames indexable. The indexation rate of actually recorded frames also increased to 20%. The *veto* algorithm having proved its robustness, it is now activated for (most) experiments on combination with a graphical helper tools to asses the optimal threshold during the experiment.

4.4. Limitations

There is a strong sensitivity of the indexation rate with data from the Jungfrau detector, related to the description of the detector in CrystFEL. The Jungfrau 4M detector is built from 8 modules manually assembled and exhibits some residual misalignment, on the order of a couple of pixels. This misalignment is much larger than what has been measured on Eiger detectors from Dectris, where misalignment is less than one pixel in size (75 µm). On Eiger detector data, where the detector is defined as a single rigid module, the indexation rate is fairly independent of the peak-picking algorithm as described in 4.2. Work is ongoing to convert the geometry description from pyFAI, where every pixel is independent, to the geometry used in CrystFEL, and vice-versa. Until then, frames are indexed on peak position freshly re-extracted by CrystFEL using its own peak-finding algorithms.

5. Conclusion

Background analysis of single crystal diffraction images can be implemented efficiently using iterative azimuthal integration and allows the separation of the signal originating from Bragg-peaks from amorphous background.

A first application presented is lossy compression of diffraction frames where the average background is saved on the one side and the position and the intensity of pixels belonging to peaks on the the other. The quality of the compression has been demonstrated on rotational macro-molecular crystallography where the degradation of the signal has been monitored after a cycle of compression-decompression and the analysis of those dataset with standard reduction tools (XDS).

The second application is a peak-finder for synchrotron serial crystallography which locates peak positions in real time and can be used to discard frames. The quality of the picking has been evaluated by comparing the indexation rate with reference peak-

REVIEW DOCUMENT

finder algorithms. The number of peaks found is used as veto algorithm to discard frames without diffraction signal and save storage space.

Acknowledgements The authors would like to thank Andy Götz for the management of this project and Vincent Favre-Nicolin for his support. Thanks also to Gavin Vaughan, scientist at Materials beamline at the ESRF, for the constructive discussion about sigma-clipping versus median filtering. We would like to thank also Jonathan P. Wright and Carlotta Giacobbe for offering us the ability to test those algorithms on small molecule data and validate the concept of sparsification on the ID11 beamline at the ESRF. Pierre Paleo and Jerome Lesaint are also acknowledged for the fruitful discussions on numerical methods developed in this document.

References

Barty, A., Kirian, R. A., Maia, F. R. N. C., Hantke, M., Yoon, C. H., White, T. A. & Chapman, H. (2014). *Journal of Applied Crystallography*, **47**(3), 1118–1131. https://doi.org/10.1107/S1600576714007626

Blelloch, G. E. (1996). *Commun. ACM*, **39**(3), 85–97. https://doi.org/10.1145/227234.227246

Bordet, P., Kergourlay, F., Pinto, A., Blanc, N. & Martinetto, P. (2021). *J. Anal. At. Spectrom.* **36**, 1724–1734. http://dx.doi.org/10.1039/D1JA00143D

Boutet, S., Lomb, L., Williams, G. J., Barends, T. R. M., Aquila, A., Doak, R. B., Weierstall, U., DePonte, D. P., Steinbrener, J., Shoeman, R. L., Messerschmidt, M., Barty, A., White, T. A., Kassemeyer, S., Kirian, R. A., Seibert, M. M., Montanez, P. A., Kenney, C., Herbst, R., Hart, P., Pines, J., Haller, G., Gruner, S. M., Philipp, H. T., Tate, M. W., Hromalik, M., Koerner, L. J., van Bakel, N., Morse, J., Ghonsalves, W., Arnlund, D., Bogan, M. J., Caleman, C., Fromme, R., Hampton, C. Y., Hunter, M. S., Johansson, L. C., Katona, G., Kupitz, C., Liang, M., Martin, A. V., Nass, K., Redecke, L., Stellato, F., Timneanu, N., Wang, D., Zatsepin, N. A., Schafer, D., Defever, J., Neutze, R., Fromme, P., Spence, J. C. H., Chapman, H. N. & Schlichting, I. (2012). *Science*, **337**(6092), 362–364. https://www.science.org/doi/abs/10.1126/science.1217737

Broennimann, C., Eikenberry, E. F., Henrich, B., Horisberger, R., Huelsen, G., Pohl, E., Schmitt, B., Schulze-Briese, C., Suzuki, M., Tomizaki, T., Toyokawa, H. & Wagner, A. (2006). *Journal of Synchrotron Radiation*, **13**(2), 120–130. https://doi.org/10.1107/S0909049505038665

Casanas, A., Warshamanage, R., Finke, A. D., Panepucci, E., Olieric, V., Nöll, A., Tampé, R., Brandstetter, S., Förster, A., Mueller, M., Schulze-Briese, C., Bunk, O. & Wang, M. (2016). *Acta Crystallographica Section D*, **72**(9), 1036–1048. https://doi.org/10.1107/S2059798316012304

Chaize, J. *et al.* (2018). In *Proc. of International Conference on Accelerator and Large Experimental Control Systems (ICALEPCS'17), Barcelona, Spain, 8-13 October 2017*, no. 16 in International Conference on Accelerator and Large Experimental Control Systems, pp. 2010–2015. Geneva, Switzerland: JACoW. Https://doi.org/10.18429/JACoW-ICALEPCS2017-FRAPL07. http://jacow.org/icalepcs2017/papers/frapl07.pdf

PREPRINT DOCUMENT

Chapman, H. N., Fromme, P., Barty, A., White, T. A., Kirian, R. A., Aquila, A., Hunter, M. S., Schulz, J., DePonte, D. P., Weierstall, U., Doak, R. B., Maia, F. R. N. C., Martin, A. V., Schlichting, I., Lomb, L., Coppola, N., Shoeman, R. L., Epp, S. W., Hartmann, R., Rolles, D., Rudenko, A., Foucar, L., Kimmel, N., Weidenspointner, G., Holl, P., Liang, M., Barthelmess, M., Caleman, C., Boutet, S., Bogan, M. J., Krzywinski, J., Bostedt, C., Bajt, S., Gumprecht, L., Rudek, B., Erk, B., Schmidt, C., Homke, A., Reich, C., Pietschner, D., Struder, L., Hauser, G., Gorke, H., Ullrich, J., Herrmann, S., Schaller, G., Schopper, F., Soltau, H., Kühnel, K.-U., Messerschmidt, M., Bozek, J. D., Hau-Riege, S. P., Frank, M., Hampton, C. Y., Sierra, R. G., Starodub, D., Williams, G. J., Hajdu, J., Timneanu, N., Seibert, M. M., Andreasson, J., Rocker, A., Jonsson, O., Svenda, M., Stern, S., Nass, K., Andritschke, R., Schröter, C.-D., Krasniqi, F., Bott, M., Schmidt, K. E., Wang, X., Grotjohann, I., Holton, J. M., Barends, T. R. M., Neutze, R., Marchesini, S., Fromme, R., Schorb, S., Rupp, D., Adolph, M., Gorkhover, T., Andersson, I., Hirsemann, H., Potdevin, G., Graafsma, H., Nilsson, B., Spence, J. C. H. & DESY (2011). *Nature*, **470**, 73–77.
https://bib-pubdb1.desy.de/record/87161

Coquelle, N., Brewster, A. S., Kapp, U., Shilova, A., Weinhausen, B., Burghammer, M. & Colletier, J.-P. (2015). *Acta Crystallographica Section D*, **71**(5), 1184–1196.
https://doi.org/10.1107/S1399004715004514

Debionne, S., Homs, A. & Claustre, L. (2022*a*). Library for IMage Acquisition version 2.
https://gitlab.esrf.fr/limagroup/lima2

Debionne, S., Homs, A., Claustre, L., Kieffer, J., De Sanctis, D., Santoni, G., Goetz, A. & Meyer, J. (2022*b*). In *Proceedings of the* 14^{th} *international conference on Synchrotron Radiation Instrumentation (SRI2021)*.
https://indico.desy.de/event/27430/abstracts/

Dectris (2014).
https://www.dectris.com/support/downloads/datasets/

Diederichs, K. & Karplus, P. A. (1997). *Nature Structural Biology*, **4**(4), 269–275.

Diederichs, K. & Wang, M. (2017). *Serial Synchrotron X-Ray Crystallography (SSX)*, pp. 239–272. New York, NY: Springer New York.
https://doi.org/10.1007/978-1-4939-7000-1_10

Fang, F., Wang, T., Wu, S. & Zhang, G. (2020). *Information Sciences*, **514**, 56–70.
https://www.sciencedirect.com/science/article/pii/S0020025519311090

Galchenkova, M., Tolstikova, A., Klopprogge, B., Sprenger, J., Oberthuer, D., Brehm, W., White, T. A., Barty, A., Chapman, H. N. & Yefanov, O. (2024). *IUCrJ*, **11**(2), 190–201.
https://doi.org/10.1107/S205225252400054X

Gasparotto, P., Barba, L., Stadler, H.-C., Assmann, G., Mendonça, H. & Ashton (2023). *chemrxiv*.

Gati, C., Bourenkov, G., Klinge, M., Rehders, D., Stellato, F., Oberthür, D., Yefanov, O., Sommer, B. P., Mogk, S., Duszenko, M., Betzel, C., Schneider, T. R., Chapman, H. N. & Redecke, L. (2014). *IUCrJ*, **1**(2), 87–94.
https://doi.org/10.1107/S2052252513033939

Gevorkov, Y., Barty, A., Brehm, W., White, T. A., Tolstikova, A., Wiedorn, M. O., Meents, A., Grigat, R.-R., Chapman, H. N. & Yefanov, O. (2020). *Acta Crystallographica Section A*, **76**(2), 121–131.
https://doi.org/10.1107/S2053273319015559

Gevorkov, Y., Galchenkova, M., Mariani, V., Barty, A., White, T. A., Chapman, H. N. & Yefanov, O. (2024). *Crystals*, **14**(2).
https://www.mdpi.com/2073-4352/14/2/164

Gevorkov, Y., Yefanov, O., Barty, A., White, T. A., Mariani, V., Brehm, W., Tolstikova, A., Grigat, R.-R. & Chapman, H. N. (2019). *Acta Crystallographica Section A*, **75**(5), 694–704.
https://doi.org/10.1107/S2053273319010593

Ginn, H. M., Roedig, P., Kuo, A., Evans, G., Sauter, N. K., Ernst, O. P., Meents, A., Mueller-Werkmeister, H., Miller, R. J. D. & Stuart, D. I. (2016). *Acta Crystallographica Section D*, **72**(8), 956–965.
https://doi.org/10.1107/S2059798316010706

Hadian-Jazi, M., Sadri, A., Barty, A., Yefanov, O., Galchenkova, M., Oberthuer, D., Komadina, D., Brehm, W., Kirkwood, H., Mills, G., de Wijn, R., Letrun, R., Kloos, M., Vakili, M., Gelisio, L., Darmanin, C., Mancuso, A. P., Chapman, H. N. & Abbey, B. (2021). *Journal of Applied Crystallography*, **54**(5), 1360–1378. https://doi.org/10.1107/S1600576721007317

Hammersley, A. P., Svensson, S. O., Hanfland, M., Fitch, A. N. & Hausermann, D. (1996). *High Press. Res.* **14**(4-6), 235–248.

Joldes, M., Muller, J.-M. & Popescu, V. (2017). *ACM Trans. Math. Softw.* **44**(2). https://doi.org/10.1145/3121432

Kabsch, W. (2010). *Acta Crystallographica Section D*, **66**(2), 133–144. https://doi.org/10.1107/S0907444909047374

Karplus, P. A. & Diederichs, K. (2012). *Science*, **336**(6084), 1030–1033. https://www.science.org/doi/abs/10.1126/science.1218231

Khronos (2008). The open standard for parallel programming of heterogeneous systems. http://www.khronos.org/opencl

Kieffer, J. & Ashiotis, G. (2014). In *PROC. OF THE 7th EUR. CONF. ON PYTHON IN SCIENCE (EUROSCIPY 2014)*. http://arxiv.org/pdf/1412.6367.pdf

Kieffer, J., Valls, V., Blanc, N. & Hennig, C. (2020). *Journal of Synchrotron Radiation*, **27**(2), 558–566. https://doi.org/10.1107/S1600577520000776

Kieffer, J. & Wright, J. (2013). *Powder Diffraction*, **28**(S2), S339–S350.

Klöckner, A., Pinto, N., Lee, Y., Catanzaro, B., Ivanov, P. & Fasih, A. (2012). *Parallel Computing*, **38**(3), 157–174.

Knudsen, E. B., Sørensen, H. O., Wright, J. P., Goret, G. & Kieffer, J. (2013). *J. Appl. Cryst.* **46**, 537–539.

Könnecke, M., Akeroyd, F. A., Bernstein, H. J., Brewster, A. S., Campbell, S. I., Clausen, B., Cottrell, S., Hoffmann, J. U., Jemian, P. R., Männicke, D., Osborn, R., Peterson, P. F., Richter, T., Suzuki, J., Watts, B., Wintersberger, E. & Wuttke, J. (2015). *Journal of Applied Crystallography*, **48**(1), 301–305. https://doi.org/10.1107/S1600576714027575

Leonarski, F., Mozzanica, A., Brückner, M., Lopez-Cuenca, C., Redford, S., Sala, L., Babic, A., Billich, H., Bunk, O., Schmitt, B. & Wang, M. (2020). *Structural Dynamics*, **7**(1), 014305. https://doi.org/10.1063/1.5143480

Maia, F. (2012). *Nature Methods*, **9**, 854–855.

Maples, M. P., Reichart, D. E., Konz, N. C., Berger, T. A., Trotter, A. S., Martin, J. R., Dutton, D. A., Paggen, M. L., Joyner, R. E. & Salemi, C. P. (2018). *The Astrophysical Journal Supplement Series*, **238**(1), 2. https://doi.org/10.3847/1538-4365/aad23d

Mariani, V., Morgan, A., Yoon, C. H., Lane, T. J., White, T. A., O'Grady, C., Kuhn, M., Aplin, S., Koglin, J., Barty, A. & Chapman, H. N. (2016). *Journal of Applied Crystallography*, **49**(3), 1073–1080. https://doi.org/10.1107/S1600576716007469

Masui, K., Amiri, M., Connor, L., Deng, M., Fandino, M., Höfer, C., Halpern, M., Hanna, D., Hincks, A., Hinshaw, G., Parra, J., Newburgh, L., Shaw, J. & Vanderlinde, K. (2015). *Astronomy and Computing*, **12**, 181–190. https://www.sciencedirect.com/science/article/pii/S2213133715000694

Mozzanica, A., Bergamaschi, A., Brueckner, M., Cartier, S., Dinapoli, R., Greiffenberg, D., Jungmann-Smith, J., Maliakal, D., Mezza, D., Ramilli, M., Ruder, C., Schaedler, L., Schmitt, B., Shi, X. & Tinti, G. (2016). *Journal of Instrumentation*, **11**(02), C02047–C02047. https://doi.org/10.1088/1748-0221/11/02/c02047

UNDER DOCUMENT

Nogly, P., James, D., Wang, D., White, T. A., Zatsepin, N., Shilova, A., Nelson, G., Liu, H., Johansson, L., Heymann, M., Jaeger, K., Metz, M., Wickstrand, C., Wu, W., Båth, P., Berntsen, P., Oberthuer, D., Panneels, V., Cherezov, V., Chapman, H., Schertler, G., Neutze, R., Spence, J., Moraes, I., Burghammer, M., Standfuss, J. & Weierstall, U. (2015). *IUCrJ*, **2**(2), 168–176. https://doi.org/10.1107/S2052252514026487

Owen, R. L., Axford, D., Sherrell, D. A., Kuo, A., Ernst, O. P., Schulz, E. C., Miller, R. J. D. & Mueller-Werkmeister, H. M. (2017). *Acta Crystallographica Section D*, **73**(4), 373–378. https://doi.org/10.1107/S2059798317002996

Petitdemange, S. *et al.* (2018). In *Proc. of International Conference on Accelerator and Large Experimental Control Systems (ICALEPCS'17), Barcelona, Spain, 8-13 October 2017*, no. 16 in International Conference on Accelerator and Large Experimental Control Systems, pp. 886–890. Geneva, Switzerland: JACoW. Https://doi.org/10.18429/JACoW-ICALEPCS2017-TUPHA194. http://jacow.org/icalepcs2017/papers/tupha194.pdf

Powell, H. R., Johnson, O. & Leslie, A. G. W. (2013). *Acta Crystallographica Section D*, **69**(7), 1195–1203. https://doi.org/10.1107/S0907444912048524

Rigaku Oxford Diffraction (2015). Crysalispro software system. https://www.rigaku.com/fr/products/smc/crysalis

Schubert, E. & Gertz, M. (2018). In *Proceedings of the 30th International Conference on Scientific and Statistical Database Management*, SSDBM '18. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3221269.3223036

Sivia, D. S. & Skilling, J. (2006). *Data Analysis - A Bayesian Tutorial*. Oxford Science Publications. Oxford University Press, 2nd ed.

Stone, J. E., Gohara, D. & Shi, G. (2010). *Computing in Science and Engineering*, **12**(3), 66–73.

The HDF Group (2000-2021). Hierarchical data format version 5. http://www.hdfgroup.org/HDF5

Toledo, S. (1997). *IBM Journal of Research and Development*, **41**(6), 711–725.

Vincent, T., Valls, V., Payno, H., Kieffer, J., Solé, V. A., Paleo, P., Naudet, D., de Nolf, W., Knobel, P., Garriga, J., Retegan, M., Rovezzi, M., Fangohr, H., Kenter, P., UUSim, Favre-Nicolin, V., Nemoz, C., Picca, F.-E., Caswell, T. A., Bertoldo, J. P. C., Campbell, A., Wright, C. J. C., Communie, G., Kotanski, J., Coutinho, T., N0B0dY, Kim, S.-W., schooft, Farago, T. & linupi (2021). silx-kit/silx: 1.0.0: 2021/12/06. Zenodo. https://doi.org/10.5281/zenodo.5761269

White, T. A., Kirian, R. A., Martin, A. V., Aquila, A., Nass, K., Barty, A. & Chapman, H. N. (2012). *Journal of Applied Crystallography*, **45**(2), 335–341. https://doi.org/10.1107/S0021889812002312

Zaefferer, S. (2000). *Journal of Applied Crystallography*, **33**(1), 10–25. https://doi.org/10.1107/S0021889899010894

Synopsis

Precise background assessment and application to single crystal image compression and serial crystallography data pre-processing.

REVISED DOCUMENT

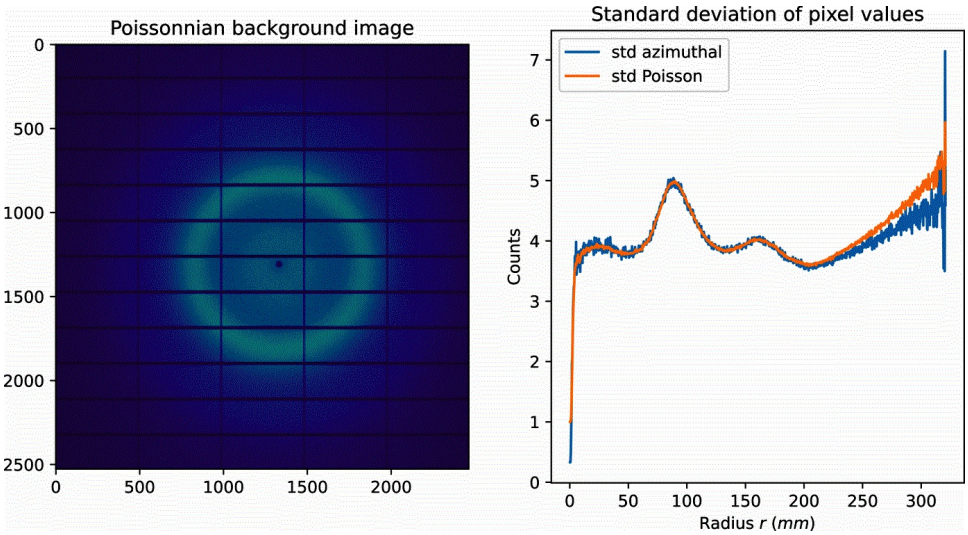
Poissonnian background image
0
500
1000
1500
2000
2500
Standard deviation of pixel values
std azimuthal
std Poisson
Counts
0
1
2
3
4
5
6
7
0
50
100
150
200
250
300
Radius r (mm)

Figure 1

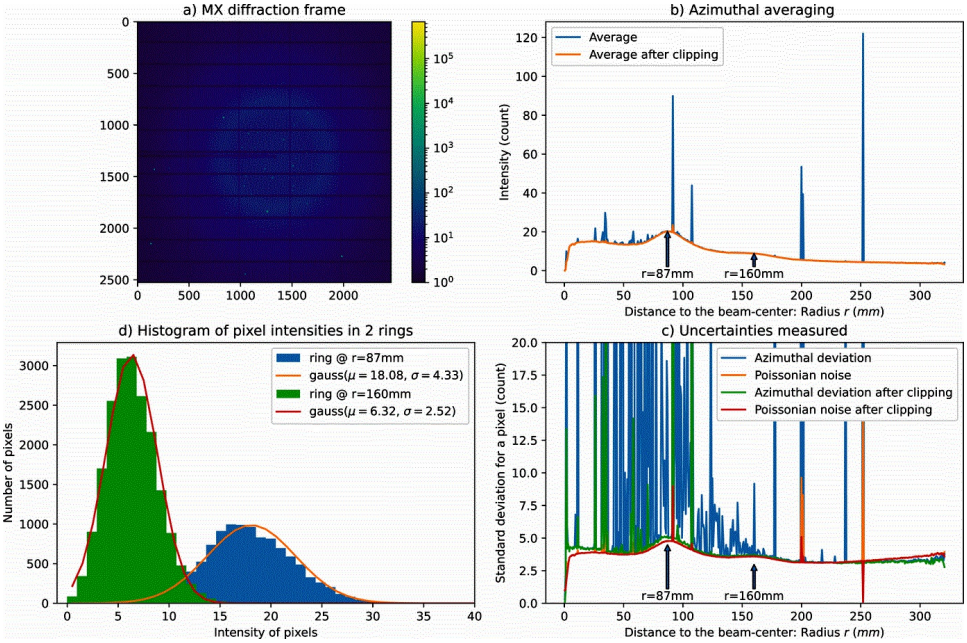
a) MX diffraction frame
b) Azimuthal averaging
Average
Average after clipping
Intensity (count)
r=87mm
r=160mm
Distance to the beam-center: Radius r (mm)
d) Histogram of pixel intensities in 2 rings
ring @ r=87mm
gauss(μ = 18.08, σ = 4.33)
ring @ r=160mm
gauss(μ = 6.32, σ = 2.52)
Number of pixels
Intensity of pixels
c) Uncertainties measured
Azimuthal deviation
Poissonian noise
Azimuthal deviation after clipping
Poissonian noise after clipping
Standard deviation for a pixel (count)
r=87mm
r=160mm
Distance to the beam-center: Radius r (mm)

Figure 2

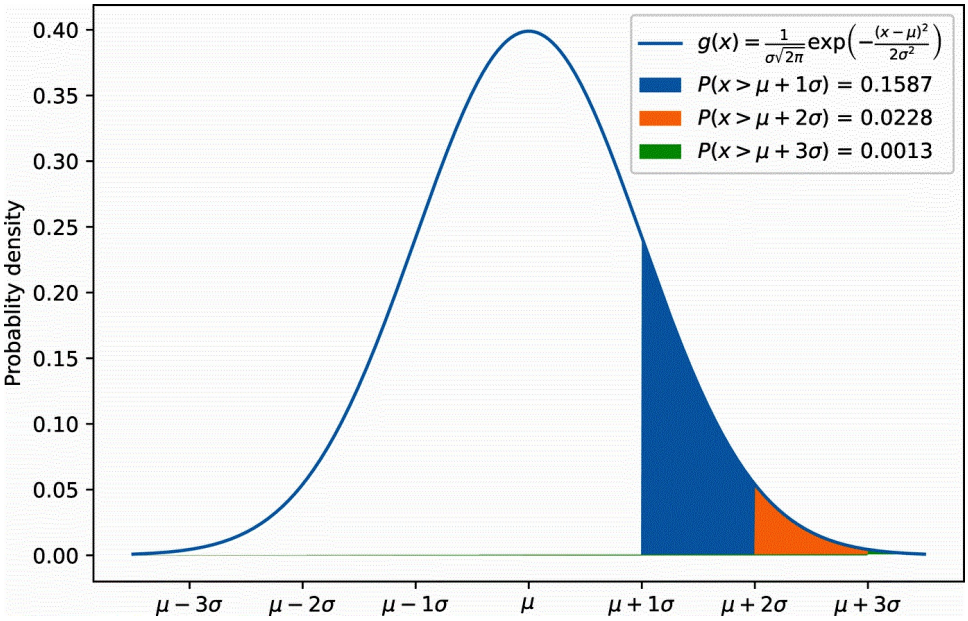
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Probablity density
$g(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
$P(x > \mu + 1\sigma) = 0.1587$
$P(x > \mu + 2\sigma) = 0.0228$
$P(x > \mu + 3\sigma) = 0.0013$
$\mu - 3\sigma$
$\mu - 2\sigma$
$\mu - 1\sigma$
μ
$\mu + 1\sigma$
$\mu + 2\sigma$
$\mu + 3\sigma$

Figure 3

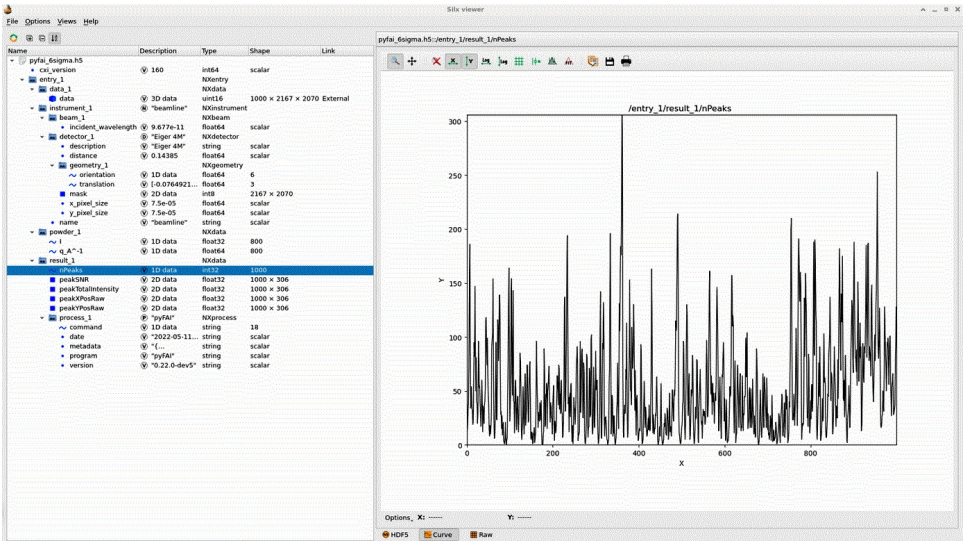
Silx viewer
File Options Views Help
Name Description Type Shape Link
pyfai_6sigma.h5
cxi_version 160 int64 scalar
entry_1 NXentry
data_1 NXdata
data 3D data uint16 1000 × 2167 × 2070 External
instrument_1 "beamline" NXinstrument
beam_1 NXbeam
incident_wavelength 9.677e-11 float64 scalar
detector_1 "Eiger 4M" NXdetector
description "Eiger 4M" string scalar
distance 0.14385 float64 scalar
geometry_1 NXgeometry
orientation 1D data float64 6
translation [-0.0764921... float64 3
mask 2D data int8 2167 × 2070
x_pixel_size 7.5e-05 float64 scalar
y_pixel_size 7.5e-05 float64 scalar
name "beamline" string scalar
powder_1 NXdata
I 1D data float32 800
Q_A^-1 1D data float64 800
result_1 NXdata
nPeaks 1D data int32 1000
peakSNR 2D data float32 1000 × 306
peakTotalIntensity 2D data float32 1000 × 306
peakXPosRaw 2D data float32 1000 × 306
peakYPosRaw 2D data float32 1000 × 306
process_1 "pyFAI" NXprocess
command 1D data string 18
date "2022-05-11... string scalar
metadata "{... string scalar
program "pyFAI" string scalar
version "0.22.0-dev5" string scalar
pyfai_6sigma.h5::/entry_1/result_1/nPeaks
/entry_1/result_1/nPeaks
300
250
200
150
100
50
0
Y
0
200
400
600
800
X
Options X: ----- Y: -----
HDF5 Curve Raw

Figure 4

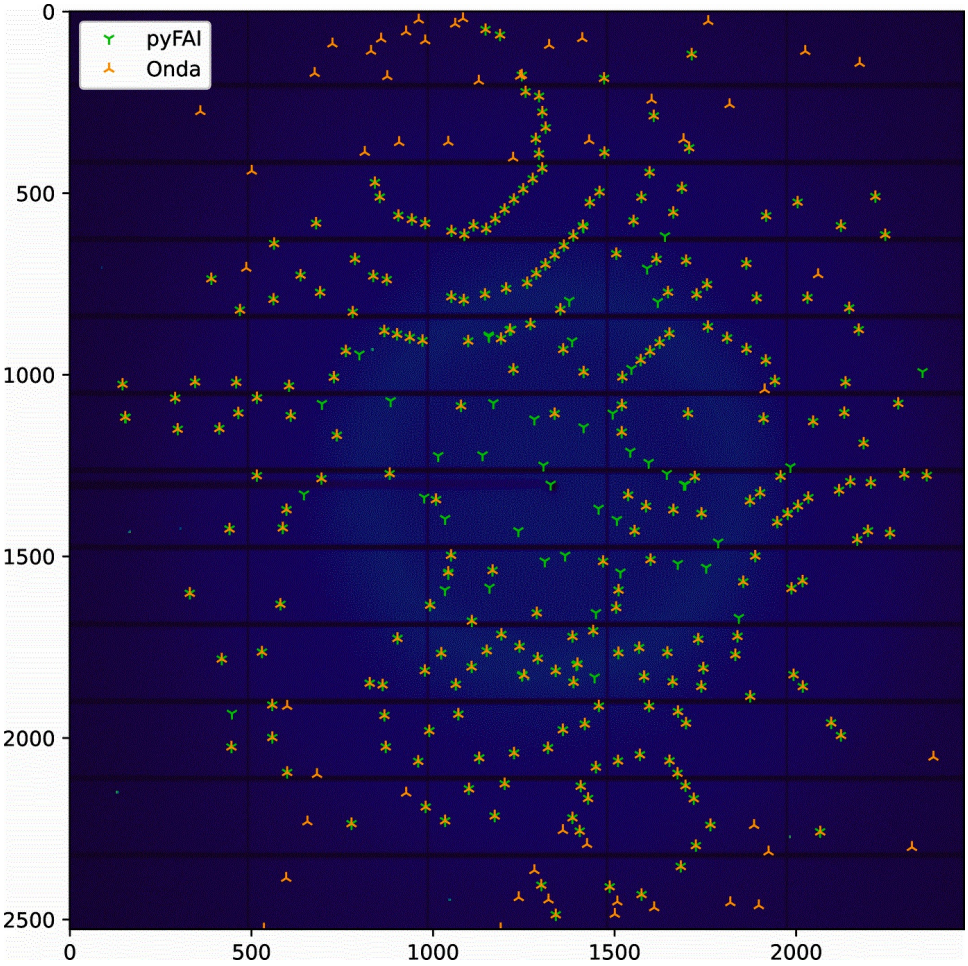

Figure 5

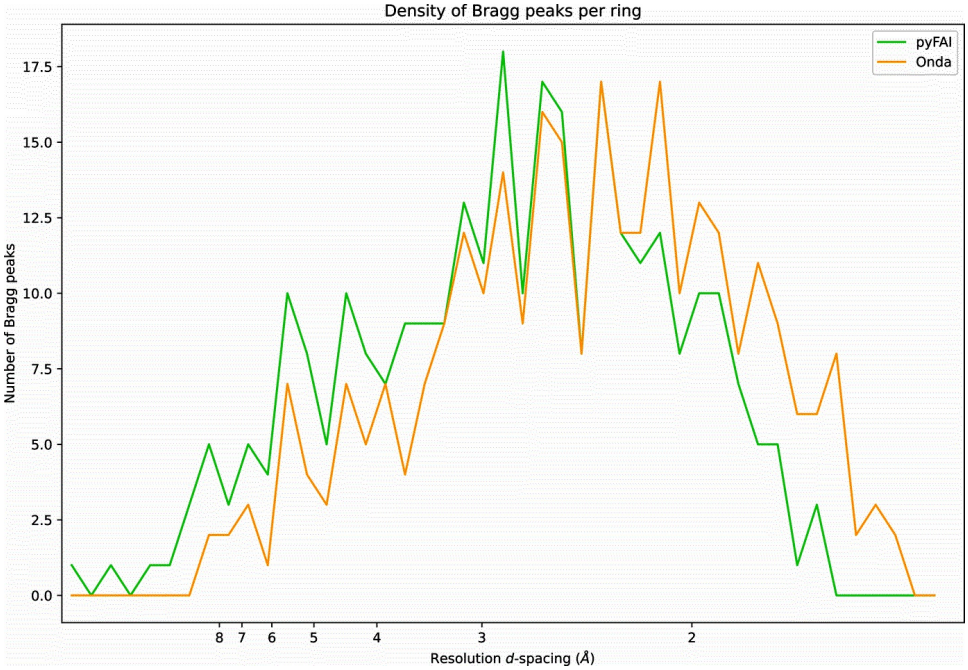
Density of Bragg peaks per ring
pyFAI
Onda
Number of Bragg peaks
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
8
7
6
5
4
3
2
Resolution d-spacing (Å)

Figure 6

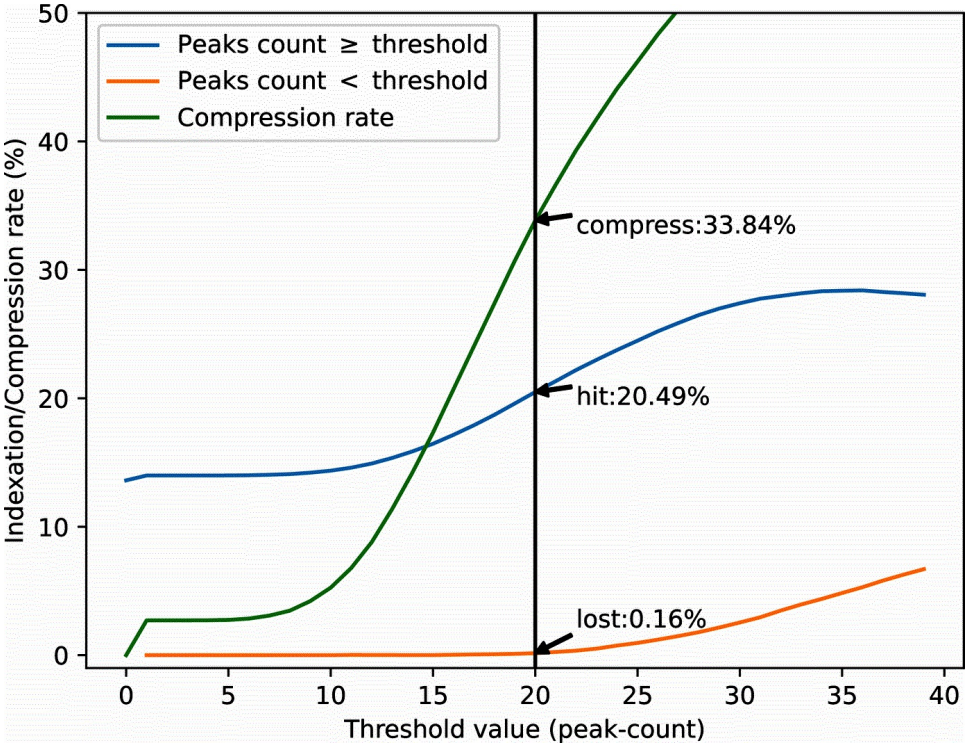
Peaks count ≥ threshold
Peaks count < threshold
Compression rate
compress:33.84%
hit:20.49%
lost:0.16%
Indexation/Compression rate (%)
Threshold value (peak-count)
0
10
20
30
40
50
0
5
10
15
20
25
30
35
40

Figure 7