METHODS
published: 19 May 2021
doi: 10.3389/fmolb.2021.624264

Serial Electron Diffraction Data Processing With diffractem and CrystFEL

Robert Bücker[1,2*†], *Pascal Hogan-Lamarre*[1,3,4†] and *R. J. Dwayne Miller*[3,4]

[1]*Max Planck Institute for the Structure and Dynamics of Matter, Center for Free-Electron Laser Science, Hamburg, Germany,* [2]*Centre for Structural Systems Biology, Department of Chemistry, University of Hamburg, Hamburg, Germany,* [3]*Department of Physics, University of Toronto, Toronto, ON, Canada,* [4]*Department of Chemistry, University of Toronto, Toronto, ON, Canada*

Serial electron diffraction (SerialED) is an emerging technique, which applies the snapshot data-collection mode of serial X-ray crystallography to three-dimensional electron diffraction (3D Electron Diffraction), forgoing the conventional rotation method. Similarly to serial X-ray crystallography, this approach leads to almost complete absence of radiation damage effects even for the most sensitive samples, and allows for a high level of automation. However, SerialED also necessitates new techniques of data processing, which combine existing pipelines for rotation electron diffraction and serial X-ray crystallography with some more particular solutions for challenges arising in SerialED specifically. Here, we introduce our analysis pipeline for SerialED data, and its implementation using the *CrystFEL* and *diffractem* program packages. Detailed examples are provided in extensive supplementary code.

Keywords: MicroED, python package, serial crystallography, data processing, crystallography, CryoEM, electron diffraction

OPEN ACCESS

Edited by:
Dan Shi,
Center for Cancer Research,
United States

Reviewed by:
Esko Oksanen,
European Spallation Source, Sweden
Christian Cambillau,
Aix-Marseille Université, France

Correspondence:
Robert Bücker
robert.buecker@mpsd.mpg.de

†These authors have contributed equally to this work

Specialty section:
This article was submitted to Structural Biology, a section of the journal Frontiers in Molecular Biosciences

Received: *31 October 2020*
Accepted: *28 April 2021*
Published: *19 May 2021*

Citation:
Bücker R, Hogan-Lamarre P and Miller RJ D (2021) Serial Electron Diffraction Data Processing With diffractem and CrystFEL. Front. Mol. Biosci. 8:624264. doi: 10.3389/fmolb.2021.624264

1 INTRODUCTION

Transmission electron microscopy as a tool for both material and life science has recently seen revolutionary developments, driven by new types of electron detectors, computational data analysis, automation, and sample preparation. Concomitantly, statistics from the Protein Data Bank (PDB) and the Electron Microscopy Data Bank (EMDB) show a clear increase in the number of protein structures that are recovered through electron-based techniques. Indeed, cryo-electron microscopy (CryoEM) produces the majority of the protein structures in the 3.5–5 Å resolution range that are being released nowadays. The predominant CryoEM techniques comprise single-particle analysis and tomography, the former being especially suitable for elucidating the structure of proteins and larger complexes at near-atomic resolution, whereas the latter allows to image larger, inhomogeneous structures, up to entire cells. However, single-particle analysis is limited in its scope to molecules of weight above ≈ 40 kDa as the signal-to-noize ratio of such small particles in electron micrographs is not sufficient for computational alignment (Henderson, 1995; Glaeser, 2019), and despite recent progress in CryoEM (Nakane et al., 2020; Yip et al., 2020), X-ray crystallography is still clearly predominant for routine structure determination at the atomic resolution scale. Diffractive electron techniques such as crystallography of monolayers of proteins (2D crystallography) led to seminal results (Henderson and Unwin, 1975; Henderson et al., 1990; Gonen et al., 2005), but ultimately remained limited in scope as preparation of suitable two-dimensional crystals is often prohibitively difficult.

On the other hand, there have been successful implementations of three-dimensional electron diffraction (3D ED/MicroED) techniques, where three-dimensional, sub-micron-sized crystals are used, in analogy to X-ray crystallography (Gemmi et al., 2019; Nannenga and Gonen, 2019). As the interaction of electrons with matter is increased by up to six orders of magnitude with respect to X-ray photons, sizable signals can be obtained from even tiny crystals. This, combined with the high dose efficiency of electrons, that is, a favorable ratio of elastic to inelastic events and small energy release during inelastic events, and the signal amplification afforded by diffraction-mode acquisition (Clabbers and Abrahams, 2018) makes 3D ED especially appealing for materials which form only small and radiation-sensitive crystals. The potential of 3D ED techniques have first been realized in material science (Kolb et al., 2007; Zhang et al., 2010). Excellent results could be obtained for radiation-sensitive nanocrystalline materials such as zeolites (Su et al., 2014), or covalent- and metal-organic frameworks (Zhang et al., 2013), which often evade X-ray structure determination. Soon after, 3D ED has been introduced into life science (there mostly known as MicroED) (Nederlof et al., 2013; Shi et al., 2013; Nannenga et al., 2014), where high-resolution structures of small proteins, peptides and pharmaceuticals can now routinely be solved (Nannenga and Gonen, 2019).

Most of the 3D ED/MicroED work has so far been performed by rotating the crystal in the electron beam in various ways (Gemmi and Lanza, 2019), in analogy to goniometer-based X-ray single-crystal diffraction. More recently, *serial* electron diffraction (SerialED) has been introduced (Smeets et al., 2018; Bücker et al., 2020), where, in analogy to synchrotron- and free-electron laser-based techniques (Gati et al., 2014; Stellato et al., 2014; Chapman, 2019), a large ensemble of nanocrystals is employed, each of which only a single diffraction pattern is taken from. While this data collection scheme has important advantages over rotation methods, it requires a different approach to data processing, specifically in the data-reduction steps of a crystallographic pipeline, from raw data to estimated Bragg reflection intensities.

In this paper, we discuss our pipeline for SerialED data processing. The paper is structured as follows: In *Serial Electron Diffraction: Concept and Data Collection*, we briefly recapitulate the concept of SerialED and its implementation in our laboratory, as described in (Bücker et al., 2020). Next, in *Processing Method for Serial Electron Diffraction*, we discuss the general data processing pipeline, illustrated by examples from a typical data set. *Implementation of SerialEM Processing* introduces our program package *diffractem* and outlines it usage for the pipeline described in *Processing Method for Serial Electron Diffraction*. Finally, *Discussion and Outlook* reviews various specific aspects and potential issues of our approach, and future directions of further development.

2 SERIAL ELECTRON DIFFRACTION: CONCEPT AND DATA COLLECTION

While rotation crystallography, whether using electrons or X-rays, can yield high-quality crystallographic data from

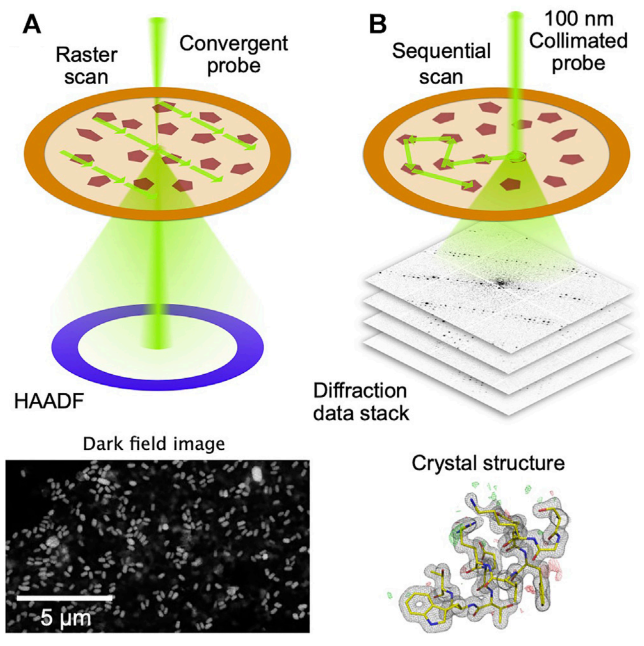


FIGURE 1 | Principle of STEM-based SerialED. **(A)** A low-resolution, low-dose STEM image is taken over a large region on a TEM grid. Signal is generated using the high-angle dark field (HAADF) detector **(B)** After the crystals have been identified in the STEM image, the beam is sequentially steered to each autmatically found crystal. A fast detector records the diffraction patterns in a synchronized way. From the diffraction data, the crystal structure is solved.

nanometric crystals, an inherent limitation is the accumulation of radiation damage during rotation data collection (Hattne et al., 2018), prohibiting acquisition of damage-minimized data. On the other hand, damage accumulation is evaded in serial crystallography, where each crystal is exposed once, using femtosecond X-ray pulses at extreme intensities that record diffraction data before Coulomb explosion (Chapman et al., 2011), or X-ray/electron pulses at lower intensity below a critical dose threshold, which can yield equivalent results (Bücker et al., 2020; Mehrabi et al., 2020).

To automate the process of collection of diffraction data from thousands of crystals randomly dispersed on an electron microscope grid, serial electron diffraction (SerialED) leverages the ability of electron microscopes to map out their locations, using conventional (Smeets et al., 2018) or scanning (Bücker et al., 2020) TEM imaging (**Figure 1A**). Crystals are automatically identified in the map image, and the electron beam is steered sequentially to the found crystals, where diffraction patterns are taken (**Figure 1B**). The process can then be repeated in many regions of a sample grid, each typically tens of microns across. This approach adds a high level of automation to the advantages of SerialED, requiring little specific skill on the user's part for operation. In (Bücker et al., 2020), SerialED was furthermore combined with a dose fractionation scheme as known from single-particle electron microscopy, which allows to obtain damage-minimized data as described above, without the need for prior information about the sample or exact calibrations.

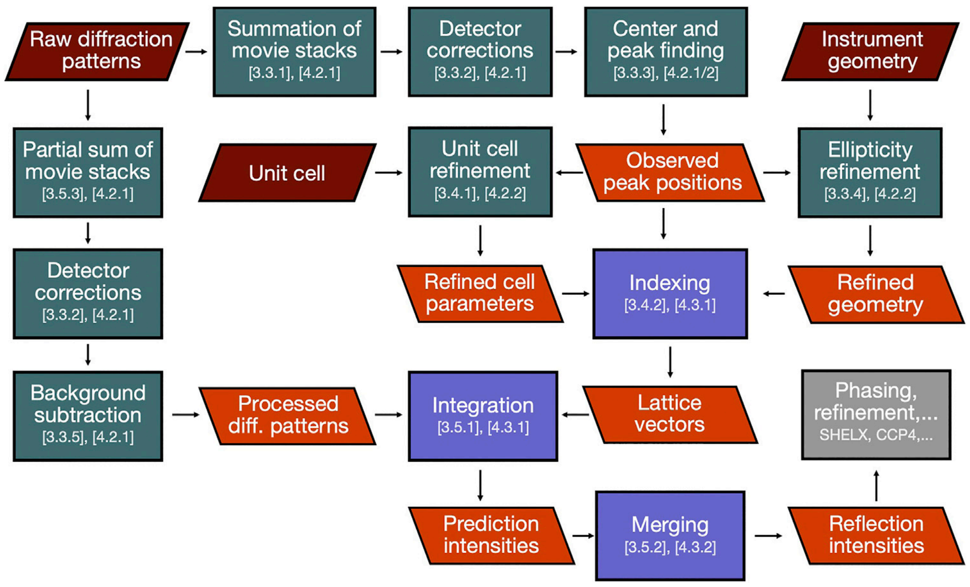

Raw diffraction patterns
Summation of movie stacks [3.3.1], [4.2.1]
Detector corrections [3.3.2], [4.2.1]
Center and peak finding [3.3.3], [4.2.1/2]
Instrument geometry
Partial sum of movie stacks [3.5.3], [4.2.1]
Unit cell
Unit cell refinement [3.4.1], [4.2.2]
Observed peak positions
Ellipticity refinement [3.3.4], [4.2.2]
Detector corrections [3.3.2], [4.2.1]
Refined cell parameters
Indexing [3.4.2], [4.3.1]
Refined geometry
Background subtraction [3.3.5], [4.2.1]
Processed diff. patterns
Integration [3.5.1], [4.3.1]
Lattice vectors
Phasing, refinement,... SHELX, CCP4,...
Prediction intensities
Merging [3.5.2], [4.3.2]
Reflection intensities

FIGURE 2 | Journey of SerialED data through our data reduction pipeline. Green and blue boxes represent processing steps conducted in *diffractem* and *CrystFEL*, respectively, section numbers in this paper relating to each step are indicated. Red and orange parallelograms represent input data and important intermediate results, respectively. The final result (reflection intensities) is then handed over to structure solution software (gray box).

Despite these advantages, with respect to rotation techniques, SerialED poses new challenges with regards to data analysis, specifically pertaining to the steps of data reduction from raw diffraction patterns to merged Bragg spot intensities. In this article, we discuss the processing of SerialED data sets using *CrystFEL* (White et al., 2012) and *diffractem*, a new library specifically developed for SerialED.

3 PROCESSING METHOD FOR SERIAL ELECTRON DIFFRACTION

In this section, we will describe the essential steps of a SerialED data processing pipeline, starting from a set of recorded diffraction patterns to merged reflection intensities, which can then be exported to standard software for phasing and refinement, such as *PHENIX* (Adams et al., 2010), *CCP4* (Winn et al., 2011), or *SHELX* (Sheldrick, 2010). While a large portion of steps to process serial crystallography data have been addressed in established packages such as *CrystFEL* (White et al., 2012), *cctbx.xfel* (Hattne et al., 2014), and *nXDS* (Kabsch, 2014), SerialED processing requires some more specific steps, which we will discuss in more detail. As example data set from which the figures and results shown in this paper are derived, we use that taken from tetragonal hen egg-white lysozyme crystals, as has been published in (Bücker et al., 2020) (PDB-ID: 6S2N). A flow-chart of the process is shown in **Figure 2**; processing steps are further illustrated for a representative diffraction pattern in **Figure 3**. For the more technical details of the processing pipeline, we refer to *Implementation of SerialEM Processing*, where the practical use of our processing program package *diffractem* in conjunction with the serial crystallography package *CrystFEL* (White et al., 2012; White, 2019) is discussed, and the Jupyter notebooks supplied as **Supplementary Material**.

3.1 Pre-Processing

We start by applying several pre-processing steps to the diffraction patterns, that is, aggregation of dose-fractionation stacks, correction of artifacts introduced by the detector, accurate determination of the pattern center (zero-order peak) and position of Bragg peaks, and general handling of metadata.

3.1.1 Sorting and Aggregation

The first processing step is to reject superfluous shots (i.e., single exposures on the camera), which might be present in the dataset due to auxiliary scan points inserted during data collection (Bücker et al., 2020) to mitigate hysteresis effects during beam scanning.

Next, if dose-fractionation movies have been collected where several images correspond to the same diffraction pattern from a single, still crystal (**Figure 3A**), the successive frames are summed over an arbitrary number of frames adding up to an equivalent exposure time, as to provide a reasonable balance between signal-to-noize ratio of low-resolution peaks and pattern resolution (which fades at later times) for each crystal (**Figure 3B**). As most of the processing steps, such as peak finding and indexing, are independent of the exact peak intensities affected by damage effects, this choice of equivalent exposure is not critical at this point, as long as the diffraction peaks are well visible. The optimal exposure time can be exactly determined during the later steps of peak integration and merging (*Processing of Dose-Fractionation Movies*).

3.1.2 Detector Artifact Correction

Any real electron detector shows a range of imperfections, three of which we account for during processing:

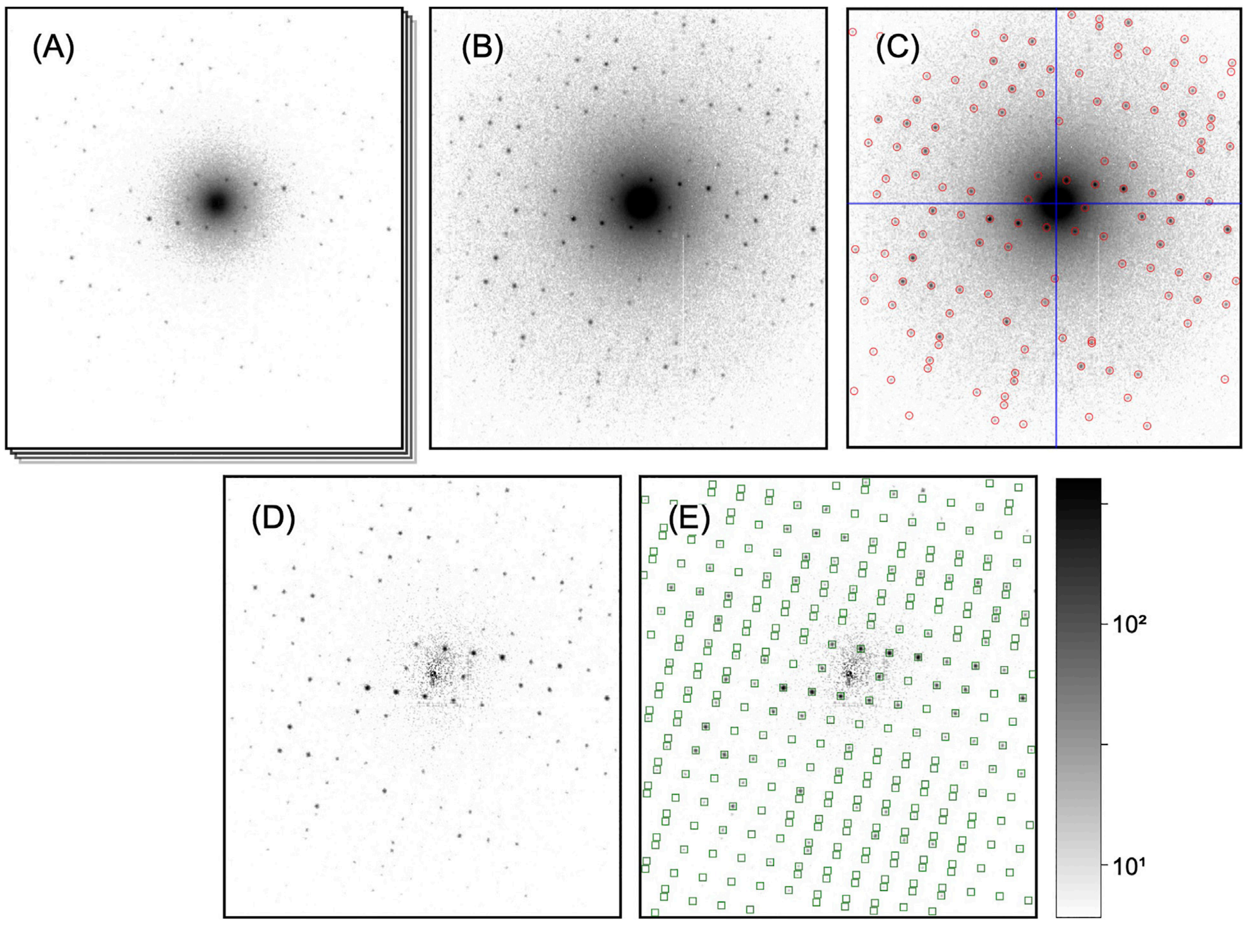


FIGURE 3 | Processing steps of a single diffraction pattern. All patterns are shown on the same, logarithmic scale. **(A)** Initial dose-fractionation stack (first fraction is shown). **(B)** Aggregated pattern over a range of four dose-fractionation frames **(C)** Pattern center (blue cross-hair) and Bragg peaks (red circles) have been determined. **(D)** Aggregated pattern after background subtraction **(E)** Predicted Bragg reflections (green squares) have been computed after successful indexing. In the integration step, those will be included as single observations.

- Faulty pixels, which yield zero, extremely high, or excessively fluctuating values, are a primary source of errors during peak finding, indexing, and integration. In our processing pipeline, we assume the existence of an accurate dead-pixel map, i.e., an image file with defined pixel values at faulty or intact pixels, respectively, which can be obtained by recording images with even illumination. During processing of diffraction patterns, the values of these pixels are overwritten by interpolation from adjacent pixels, or (at the user's choice) flagged for exclusion from further processing steps.
- The response of a detector to a homogeneous illumination (flat-field) is typically non-uniform. If the raw data are not corrected for this effect already, this can be accounted for by a simple normalization during processing.
- For high pixel values, a detector can saturate, in ways which may differ between various models. Integrating detectors such as CCD, indirect CMOS, or linear-mode direct detectors saturate in the total counts per pixel with a sharp cut-off, which can be treated by exclusion from further analysis steps, in a similar way to dead pixels. On the other hand, counting detectors e.g., of counting-mode direct or hybrid-pixel type, suffer from continuously increasing coincidence-loss saturation as a function of count *rate.* For the latter, if previously characterized, a saturation model can be applied.

Our example data set has been recorded using a hybrid-pixel detector with a large number of dead pixels, which have to be taken into account, but a fairly even flat-field. The used count rates range into the saturation range near the center of the diffraction pattern (i.e. close to the transmitted beam), which is accounted for by using a paralyzable dead-time model (Feller, 2015), parametrized from independent measurements. All of those corrections are applied before any further image analysis.

3.1.3 Pattern Centering and Peak Finding

For successful indexing of the diffraction patterns, it is of crucial importance to accurately know the center (zero-order beam) position of each diffraction pattern. For serially collected data,

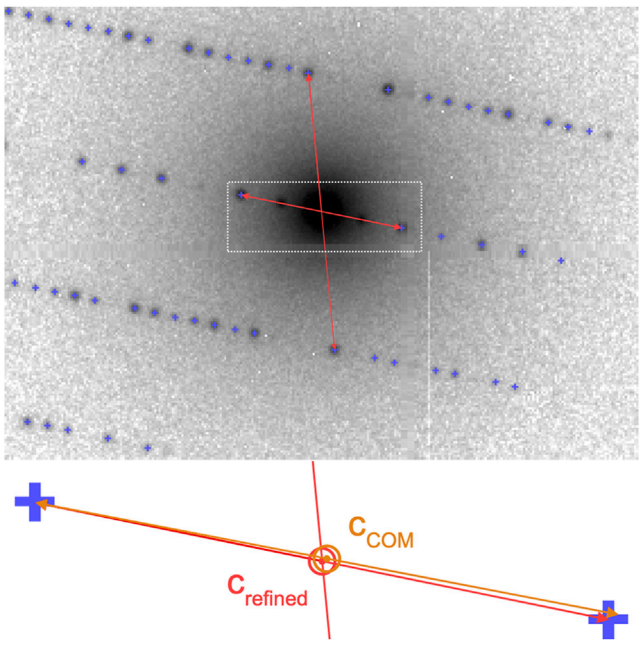


FIGURE 4 | Pattern center refinement from Friedel mates. In electron diffraction patterns, even away from zone axes, a large number of Friedel mates is simultaneously visible, such as those marked by arrows. As they are symmetric about the zero-order beam, the positions of each pair can be used to refine the initial estimate $\vec{c}_{COM}$ of the pattern center to the more accurate $\vec{c}_{refined}$

where the beam is moving between crystals, the pattern center tends to fluctuate between beam positions due to residual alignment issues of the microscope (beam-shift pivot). Hence, the beam center has to be found for each pattern separately, which in our pipeline is tightly coupled to the detection of Bragg peaks.

To find both the pattern center and peak positions, first we determine the center of mass of the diffraction pattern, excluding all pixels whose values fall below a threshold chosen such that only a region around the center is taken into account. Next, we apply a two-dimensional least-squares fit of a Lorentzian function to a region of tens of pixels of diameter around the found center of mass position, to obtain a more accurate estimate for the pattern center. Peaks are now detected using the *peakfinder8* algorithm (Barty et al., 2014), which inherently takes into account a radially symmetric background as typically present in electron diffraction patterns due to multiple inelastic/elastic scattering (Latychevskaia and Abrahams, 2019).

To further refine the center position of each electron diffraction pattern, we make use of the fact that, due to the flat Ewald sphere of electrons, even for patterns away from a zone axis, many Friedel-mate pairs (with Miller indices h,k,l and −h,−k,−l, respectively) can be found in a single image, as shown in **Figure 4**. Each pair is necessarily symmetric with respect to the pattern center, which can be used to further refine the estimate of the pattern center. The refinement is performed by defining a score function:

$$F(x_0, y_0) = \frac{1}{2N_{pk}} \sum_{i,j}^{N_{pk}} \exp\left[-\frac{1}{2\sigma^2} \left[\left(x_i + x_j - 2x_0\right)^2 + \left(y_i + y_j - 2y_0\right)^2 \right] \right],$$

with all found peaks at pixel position (x_i, x_j) of characteristic width $\sigma \approx 2$ pixels, and performing a least-squares minimization on F^{-1} in order to obtain the refined pattern center at (x_0, y_0) with sub-pixel accuracy. We find that further refinement as performed by pattern indexing codes does not lead to any significant improvement. In **Figure 3C**, a typical result of pattern centering and peak finding is shown.

3.1.4 Ellipticity Finding

A common artifact introduced by the electron optics in an electron microscope column is a slight elliptical distortion of the diffraction pattern which, even in the range of only few percent, can severely hamper the efficiency of crystallographic algorithms. Hence, care has to be taken to account for the distorted geometry, especially during the indexing and integration steps.

The ellipticity can be derived from the data itself, by computing a two-dimensional histogram of *all* measured diffraction peak positions (relative to the pattern center) in radial coordinates, as shown in **Figure 5**. In an ideal geometry, there is no dependence of any features (virtual powder rings) on the azimuth angle. The elliptical distortion as seen in **Figure 5A** can hence be computed by iteratively modifying the peak positions according to their azimuth angle, and recomputing, until no dependence is found anymore as seen in **Figure 5B**.

3.1.5 Background Rejection

In contrast to X-rays, inelastically scattered electrons are not removed from the beam, but continue their trajectory toward the detector, thus appearing in the recorded data unless an energy filter is used. While the differential cross section for inelastic scattering drops off quickly at angles small compared to typical Bragg reflection angles, *combined* elastic and inelastic scattering leads to a pronounced, radially symmetric background (Latychevskaia and Abrahams, 2019) in unfiltered electron diffraction patterns. As long as the peak integration algorithm, which serves to extract the summed intensity of each peak from the images, can handle this background appropriately, it in principle does not impact the obtained values, apart from a decreased signal-to-noize ratio at low resolutions. However, we find that subtraction of the radially symmetric background not only aids to visually represent and assess the diffraction patterns as seen in **Figure 3D**, but also simplifies the peak integration process (due to the absence of a background gradient) and leads to more consistent results after merging. The tools provided by *diffractem* as described in *Stack Processing* allow to reject any radially symmetric signal following a prescription as follows:

1. Computation of the radial profile of the inelastic background by azimuthal averaging around the previously found

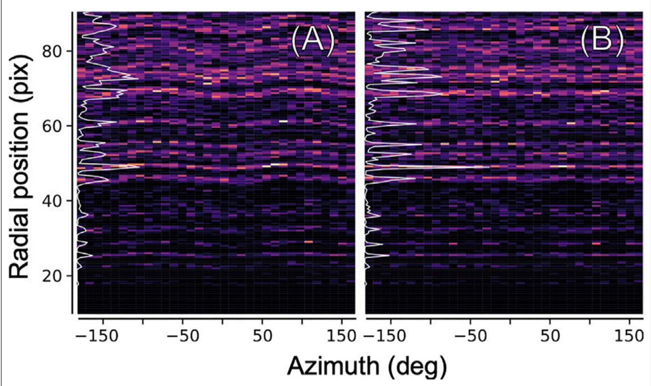


FIGURE 5 | Ellipticity refinement. In order to correct for the elliptical distortion of diffraction patterns introduced by electron optics, we histogramize all found diffraction peaks (from all images) in two-dimensional polar coordinates. **(A)** Elliptical distortion manifests itself in a modulation of the position of major features near the inverse layer spacings of the crystal. Azimuthal integration into a radial profile (white line) yields a blurred, low-contrast pattern **(B)** Same, after correcting the positions of the peaks according to an elliptical model before histogramization.

pattern center, excluding a generous area around each of the found Bragg spots to avoid over-correction.

2. Median filtering of the profile to reduce noise and reject residual ripple caused by weak, unidentified Bragg peaks.
3. Computation of expected background image from profile by assigning pixel values based on the radius with respect to the pattern center.
4. Subtraction of the computed background from the actual diffraction pattern.

We find the outcome of this procedure to be satisfactory even for the dense diffraction patterns of proteins.

3.2 Indexing

After corrected diffraction patterns with annotated center and peak positions have been computed (**Figure 3C**), the next step is indexing the patterns, that is, deriving the unit cell parameters, which are assumed to be narrowly distributed over all crystals, and the orientation of each individual crystal. Common processing pipelines for serial X-ray diffraction data solve the indexing problem by estimating a unit cell for each pattern separately, and if necessary, iteratively refining the obtained solutions (White, 2019). However, owing to the short de-Broglie wavelength of high-energy electrons, diffraction patterns are almost entirely devoid of three-dimensional information, which precludes the required determination of the crystal unit cell from single diffraction patterns. While approaches exist to bootstrap the cell information from all patterns taken as a whole (Jiang et al., 2009), the cell can also be experimentally derived from ancillary rotation-based data or multi-tilt serial data (publication in preparation). We defer the discussion of cell-finding to future publications, and instead focus on two remaining steps of indexing, namely, accurate refinement of the unit cell parameters, and determination of the orientation of each individual crystal.

3.2.1 Unit-Cell Refinement

If the Bravais lattice and reasonable estimates of the cell parameters are known, the latter can be refined against radial distribution functions derived from the found peaks in the *entire* data set, as shown in **Figure 6**. To this end, we consider two types of peak information, both of which are histogramized with respect to their radial coordinate. Firstly, we simply consider the radial position of peaks with respect to the pattern center, which can be related to the Bragg angle 2θ and hence the crystal's inverse layer spacings. The according histogram is known as a *virtual powder pattern*, as it effectively corresponds to a super-resolution measurement of a powder diffraction pattern. Secondly, we compute the distribution of all pair-wize distance vectors between peaks present in each pattern. Due to the small Bragg angles of electrons (paraxial regime), the lengths of those similarly match inverse layer spacings, which can then be averaged over the entire dataset. The advantage of the second method is that the result displays pronounced peaks near the primitive-cell basis vectors, which are hardly or not at all (due to systematic absences) present in the virtual powder pattern. Once the distribution functions are computed, the cell parameters are refined against them by matching the predicted layer spacings to their respective peaks.

3.2.2 Indexing Using pinkIndexer

Now that the unit cell of the crystals is known, the orientation of each crystal with respect to the experiment geometry can be determined by an exhaustive search over all possible rotations, for which several implementations are available (Ginn et al., 2016; Beyerlein et al., 2017; Smeets and Wan, 2017; Li et al., 2019; Gevorkov et al., 2020). We use *pinkIndexer* (Gevorkov et al., 2020), which has been tested extensively on electron data, and is directly integrated into the *indexamajig* program of CrystFEL. Before running the indexing process, it may be required to screen the parameters of the indexing on a small sub-set of diffraction patterns, which should be selected by the number of found peaks and visual appearance. Factors impacting the successful indexing rate are the accuracy of unit cell parameters, proper centering of the patterns, sampling density of rotational space, and the assumed radius of Bragg spots in reciprocal space. While the first two can be refined using the methods described above, the others have to be found heuristically for the sample under study. While the sampling density depends critically on the unit cell size, the optimal setting for the Bragg spot radius is defined by the interaction region between the electron beam and continuous crystalline blocks, which can be limited by crystal size, beam size, or crystal mosaicity and bending (Gallagher-Jones et al., 2019), as well as the beam convergence angle. Given the typical parameters of three-dimensional electron diffraction, realistic values are below $0.005\ \text{Å}^{-1}$. While a too large value tends to assign patterns to a near-zone-axis geometry with densely packed peaks, a too small value can preclude any successful indexing.

Depending on the sampling density used for the orientation search, up to 1 min of computation time is required for each

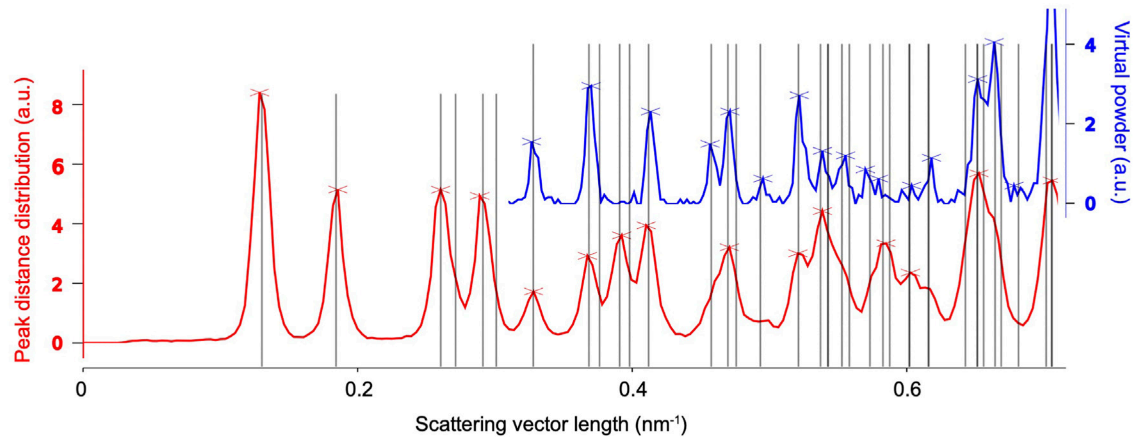


FIGURE 6 | Unit-cell refinement of the example system's tetragonal cell with *a* = 78.9Å, c = 37.9Å. After peak finding, the scattering vector length, and the pair-wize distances between the peaks (within each pattern) are histogramized over all patterns (blue and red curves, respectively), yielding peaks at the inverse layer spacings of the crystal. The unit cell can accurately be refined by fitting the computed layer spacings (gray lines) to the observed peaks in both distributions.

crystal; however it is straightforward to distribute the calculation over arbitrarily many processor cores on a cluster system, which is automated in our processing software 4.3.1.

3.3 Peak Integration, Merging, and Validation

Having determined the orientation of each crystal, we can proceed to integration and merging, that is, from the manifold of indexing solutions deriving a complete set of estimates for the Bragg spot intensities for firstly individual patterns (integration), and secondly the entire dataset (merging).

3.3.1 Integration of Intensities From Indexing Results

The unit cell vectors of each diffraction pattern as found by the indexing are used to extract the intensities of observations of Bragg reflections, which may be partial (White, 2014). To accomplish this *peak integration* step, we use functionality built into CrystFEL, as outlined in more detail in *Indexing and Integration*. Briefly, the positions of all Bragg reflections that could reasonably be present in each diffraction pattern are computed (spot *predictions*) from the crystal orientation and a refined reciprocal spot radius, as shown in **Figure 3E**. Then, the pixel intensities around each prediction position are integrated, using one out of several available methods such as profile fitting (Rossmann, 1979) and simple summation within an appropriately chosen radius (White et al., 2013). We usually find the simplest method, that is, summation without any additional refinement steps, to be most effective; background-gradient correction as also offered is only required, if the diffraction patterns are not background-subtracted.

3.3.2 Merging and Validation of Integrated Intensities

After the measured Bragg spot intensities from all shots are extracted and stored, they have to be merged into a full crystallographic data set. Firstly, depending on wether the crystal's space group shows an indexing ambiguity, it needs to be resolved using, which can be performed using a clustering algorithm (Brehm and Diederichs, 2014; Kabsch, 2014; White et al., 2016), which is provided as a part of CrystFEL (*ambigator*). Then, the many observations of each Bragg reflection are combined, in the simplest case by averaging without further weighting ("Monte-Carlo method"). Additionally, iterative global and resolution-dependent scaling can be introduced, which leads to significant improvements (White et al., 2016). Finally more elaborate models for merging such as the *xsphere* algorithm, which explicitly model the amount partiality of each observation, are available, which in our experience leads to varying, sample-dependent results (**Figure 7**).

In order to assess the overall statistics and quality of the merging result (and hence, the effectively the entire data reduction pipeline), it is of crucial importance to evaluate some important validation metrics. While traditional merging quality indicators such as R_{merge} are inadequate for serial dataset due to their strong partiality (White et al., 2012), the half-set Pearson correlation coefficient between Bragg intensities merged form a half-sets of the crystals $CC_{1/2}$ (Karplus and Diederichs, 2012) provides a robust figure of merit for consistency of the dataset. Furthermore, the completeness of the dataset, as well as the mean number of observations of each reflection (redundancy) are of highest concern. In **Figure 7**, these quantities are shown for our example data set (PDB-ID 6N2S (Bücker et al., 2020)), as a function of resolution shell, and of the number of merged crystals (by picking a sub-set from the data). We can observe that the correlation coefficient, which is near-unity at low resolution, reamains above threshold of 0.143 (corresponding to CC^{*} = 0.5 (Karplus and Diederichs, 2012)) at about 1/(1.8 Å), which is hence a reasonable resolution cut-off for phasing and refinement steps. Another important observation is that the completeness of the dataset appears to converge to a value significantly less than 100% when we increase the crystal number; this clearly indicates the presence of preferred crystal

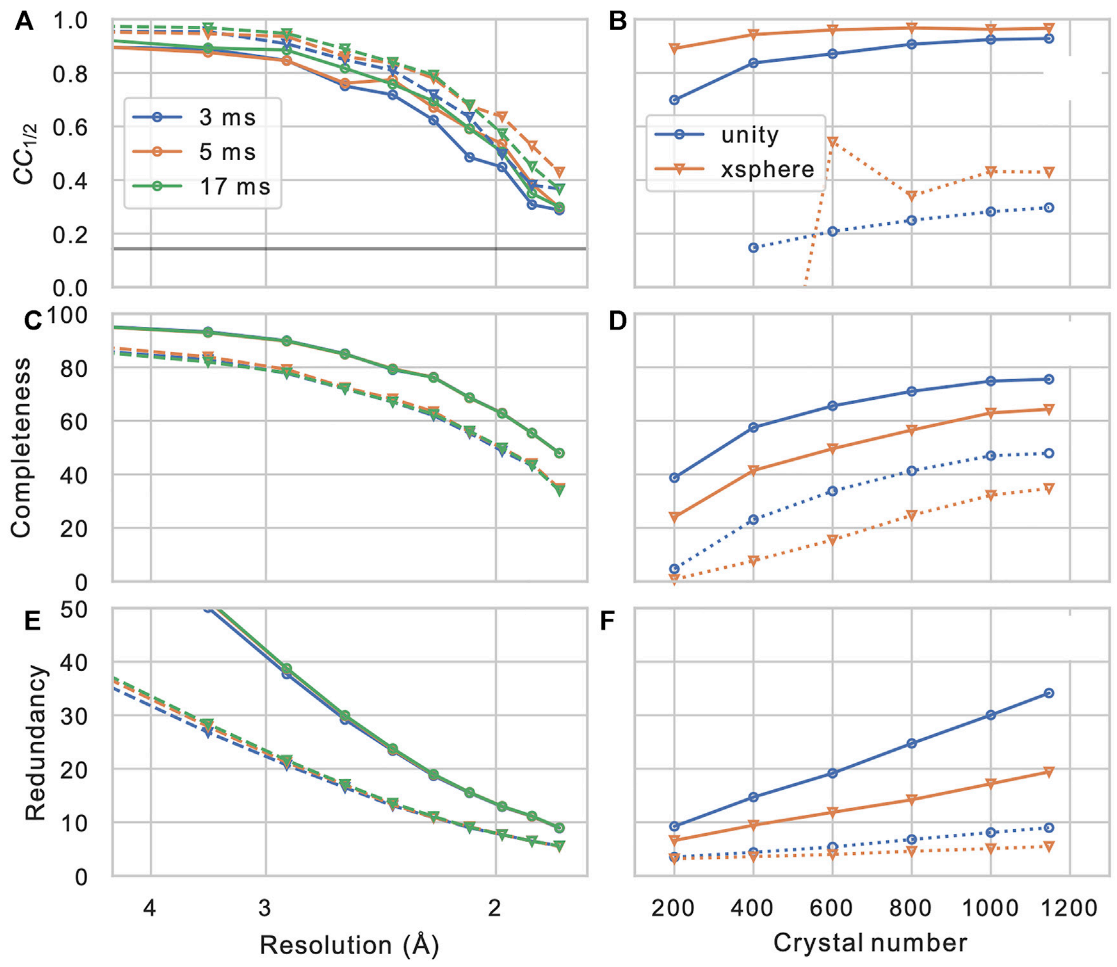


FIGURE 7 | Merging statistics as a function of resolution **(A–C)** and crystal number **(D–F)**: Half-set Pearson correlation $CC_{1/2}$, dataset completeness, and observation redundancy. In **(A–C)**, results are shown for three different integration times in different colors, and for merging without (solid lines, circles) or with (dashed lines, triangles) the *xsphere* partiality model (White, 2014). In **(D–F)**, solid and dashed lines represent values from the entire resolution range and from the highest resolution shell (centered at 1.83 Å), respectively. Blue circles and orange triangles represent results from merging without and with partiality modeling, respectively, iterative scaling was enabled in both cases.

orientation, which cannot be significantly mitigated by increasing the number of crystals. Such preferred orientation issues can however be mitigated using a tilt of the sample stage or specifically prepared sample grids (Wennmacher et al., 2019). In **Table 1** full merging and refinement statistics are shown for four representative merging configurations, where the structure as published in (Bücker et al., 2020) (PDB-ID: 6S2N) was re-refined against each of the merged set of reflection intensities using *phenix.refine* (Liebschner et al., 2019) and *Coot* (Emsley et al., 2010). It can be observed that for a given number of crystals, besides a higher value of $CC_{1/2}$ at lower completeness, partiality modeling leads to an increase in the refinement quality metrics R_{work} and R_{free}. This can be understood by considering the propensity of simple Monte-Carlo merging to include reflections in patterns that are only weakly or not at all excited; such peaks are entirely deleted from merging in the more sophisticated partiality modeling, which is also reflected in the significantly lower number of total reflections. A more detailed discussion of partiality modeling for SerialED will be the subject of future work. For a full discussion of the lysozyme data set, we refer the reader to (Bücker et al., 2020).

3.3.3 Processing of Dose-Fractionation Movies

If a sufficiently fast diffraction detector is available, it is advisable to collect SerialED in dose-fractionation mode, that is, taking a series of frames (movie) for each crystal in rapid succession, as shown in **Figure 3**. This technique is commonly applied in single-particle microscopy, and while motion blur is not of concern for diffraction data, dose fractionation allows to select an optimal exposure time, and hence radiation dose per crystal *a posteriori* (Bücker et al., 2020). Assuming that the orientation of crystals does not significantly change between the movie frames, and hence the indexing solution is valid for all frames equivalently, the exact choice of considered integration time is mostly irrelevant up to the point of integration, as long as the visible Bragg peaks at low to intermediate resolution can be reliably found. It is only in the

TABLE 1 | Merging and refinement statistics for four representative processing runs, all derived from the same data set as discussed in the text. For each of the two merging strategies *xsphere* (partiality modeling) and *unity* (Monte-Carlo averaging) data are shown for the full dataset (1147 crystals) and a sub-set of approximately half that size (600 crystals). Quantities discussed in the main text are printed in bold face.

	Hen egg-white lysozyme			
Number of crystals	1147	600	1147	600
Merging method	Xsphere		Unity	
Wavelength (Å)	0.0251			
Resolution range (Å)	39.5–1.8 (1.865–1.8)			
Space group	$P4_32_12$ (No. 96)			
Unit cell	79, 79, 38.1, 90, 90, 90			
Total reflections	**265,587 (4083)**	**125,160 (1341)**	**548,217 (9100)**	**267,281 (3849)**
Unique reflections	8305 (485)	6746 (274)	9410 (614)	8483 (485)
Multiplicity	19.4 (5.5)	11.9 (4.0)	34.2 (9.0)	19.2 (5.4)
Completeness (%)	**71.49 (42.38)**	**58.09 (23.87)**	**80.97 (53.76)**	**72.88 (42.07)**
$CC_{1/2}$	**0.96 (0.43)**	**0.96 (0.54)**	**0.93 (0.30)**	**0.87 (0.21)**
CC*	0.99 (0.78)	0.99 (0.84)	0.98 (0.68)	0.96 (0.59)
Mean I/σ_I	4.6 (1.2)	4.3 (1.1)	3.0 (0.85)	2.4 (0.74)
R-split	16.9 (102.0)	17.8 (97.8)	26.7 (121.5)	32.9 (155.8)
Wilson B-factor (Å^2)	19.59	19.63	17.27	14.34
Reflections used in refinement	8289 (478)	6734 (269)	9391 (607)	8453 (475)
Reflections used for R-free	830 (48)	674 (27)	938 (59)	846 (47)
R-work	**0.24 (0.34)**	**0.26 (0.41)**	**0.26 (0.33)**	**0.31 (0.39)**
R-free	**0.27(0.37)**	**0.29 (0.42)**	**0.29 (0.37)**	**0.32 (0.38)**
Number of non-hydrogen atoms	1072	1046	1069	1060
Macromolecules	1009	1009	1009	1009
Ligands	0	0	0	0
Solvent	63	37	60	51
Protein residues	129	129	129	129
RMS(bonds)	0.008	0.003	0.009	0.002
RMS(angles)	0.8	0.58	0.98	0.49
Ramachandran favored (%)	96.85	96.85	96.85	97.64
Ramachandran allowed (%)	3.15	3.15	3.15	2.36
Ramachandran outliers (%)	0	0	0	0
Rotamer outliers (%)	3.77	0.94	2.83	5.66
Clashscore	3.03	3.54	6.57	6.07
Average B-factor	20.51	19.11	18.71	13.98
Macromolecules	20.49	19.07	18.7	13.91
Solvent	20.86	20.11	18.8	15.37

final steps that results should be derived for different integration time separately. This can be accomplished by "broadcasting" the position of stop predictions to a dataset that comprises diffraction patterns with varying aggregation length as described in *Sorting and Aggregation*, and re-running integration and merging on those sets. Our analysis programs provide convenient functions to automate this process and guide the user to an optimal choice of exposure time.

4 IMPLEMENTATION OF SERIALEM PROCESSING

The various steps of data processing explained in the previous sections can be performed in our *Python* software package *diffractem*, which provides the necessary functionality directly, or *via* tight integration with *CrystFEL* through wrapper functions. Besides a few command-line tools, diffractem is intended for comfortable use within Jupyter notebooks, a common platform for scientific data analysis and data science in general. This section will introduce some key concepts of diffractem. For more in-depth examples and explanations, we refer the reader to the annotated Jupyter notebooks provided as supplementary information to this paper.

4.1 Data Structures and File Format

Diffraction images and meta data are accessed and managed *via* instances of diffractem's `Dataset` class. A single `Dataset` object represents arbitrarily many data files that each correspond to a SerialED acquisition run from one grid region.

4.1.1 Data Hanlded by Diffractem

In diffractem's terminology, a *shot* corresponds to a diffraction pattern recorded by the detector (equivalent to an *event* in CrystFEL), whether it constitutes a hit on a crystal or not. If dose-fractionation is used, the many shots obtained from the same crystal are referred to as the *frames* of said crystal. In SerialED, thousands of raw diffraction patterns can be acquired per hour. Thus, the initial raw data comprises a large number of 2D diffraction patterns (shots), forming together a 3D data cube referred to as *image stack* in the following, with associated meta-data. Such meta-data can be defined per diffraction pattern (*shot* *table*), per crystal (*feature table*), or per grid region (*global meta-data*). The number of peaks in a diffraction pattern, the position of a crystal on the sample grid, and the camera length setting are examples of per-shot, per-feature, and global data, respectively. The metadata can extensively be changed and extended along the data processing pipeline, where the `Dataset` object ensures consistency of image and meta data. Destructive processing steps that either change actual image data (such as background correction) or remove shots are handled by generating a new, modified `Dataset` object.

Within the `Dataset` object, the shot and feature tables are accessible as *pandas* DataFrame objects (Pandas Development Team, 2020) *via* the attributes `Dataset.shots` and `Dataset.features`. Their number of rows always correspond to the number of shots and crystals stored in the `Dataset` object, respectively. On the other hand, the number of columns is arbitrary, and commonly increases once new per-pattern analysis results become available. In any case, key columns such as the file name and location of diffraction data in the image stack, as well as the sample name, and identification numbers of each crystal and grid regions have to be present. Global meta-data (typically comprising instrument parameters such as camera length or exposure time) can be accessed or directly merged into the shot table using the `Dataset.merge_meta` method.

4.1.2 Stacks and Memory Management

The image stack comprising the actual diffraction data is often too large to fit into the main memory of a typical mid-range workstation computer. Hence, to manage this amount of data and the ensuing parallel computations, we employ the *dask* package (Dask Development Team, 2016), which allows to transparently access larger-than-memory data arrays from disk, and to build lazy computation pipelines, that can be executed efficiently in parallel (see supplementary Jupyter Notebooks for details). A `Dataset` object can contain an arbitrary number of N-dimensional dask arrays (which behave analogously to *NumPy* arrays), referred to as *stacks*, the first dimension (dimension 0) of which must always equal the number of shots contained in the `Dataset` (and hence the number of rows in the shots table). Besides the actual diffraction data stack (constituting a three-dimensional stack), typical stacks in a `Dataset` object are the data of found diffraction peaks in each image, in *CXI* format (Maia, 2012). Generally, data stacks can be added or overwritten using the `Dataset.add_stack` method and accessed *via* attributes of the form `Dataset.<stackname>`.

4.1.3 Slicing, Selecting, and Aggregating Data

A common task during the preprocessing of a diffraction data set is to reject shots based on criteria such as a minimum number of Bragg peaks or a maximum level of background signal. Such selections can easily be performed using the `Dataset.get_selection` method, which allows for selections of sub-sets *via* query strings acting on columns of the shot list. As an example, the code line `ds_sel = ds.get_selection ('num_peaks≥15')` generates a new `Dataset` object `ds_sel`, containing only shots from ds where more than 15 Bragg peaks have been detected. In this step (as in all other methods of `Dataset`), it is ensured that all stacks and tables are kept consistent. The related method `Dataset.aggregate`, which accepts a similar query string, will, on top of slicing, apply different group-wize aggregation functions to the data stacks, or a subset thereof; its typical application of the summation of dose fractions of the same diffraction pattern as described in *Processing of Dose-Fractionation Movies*. Please see the supplementary Jupyter notebooks for more detailed explanations and examples.

4.1.4 Diffractem Data Files

Diffractem stores its data in *HDF5* files largely follwing the NeXus convention (Könnecke et al., 2015), which is becoming a common standard in X-Ray crystallography, and can by now be processed by most crystallography libraries. The data within the files can be accessed from all common programming languages through bindings of the HDF5 library, such as `h5py` for *Python*, and can directly be mapped into larger-than-memory arrays using the dask package, as described above. Each file holds data from a continuous acquisition run on a single region on the sample, corresponding to a single map image as shown in **Figure 1** A on which crystals have been identified prior to diffraction data collection. A multitude of acquisition runs from the same sample which shall be analyzed as a whole can be defined using simple text files with corresponding HDF5 file names on each line, and a `.lst` extension by convention. Using the `Dataset.from_files` method, data can be loaded from a single file, a list file, or a range of files implicitly defined using wildcard characters. Both the HDF5 and list file specifications are consistent with CrystFEL.

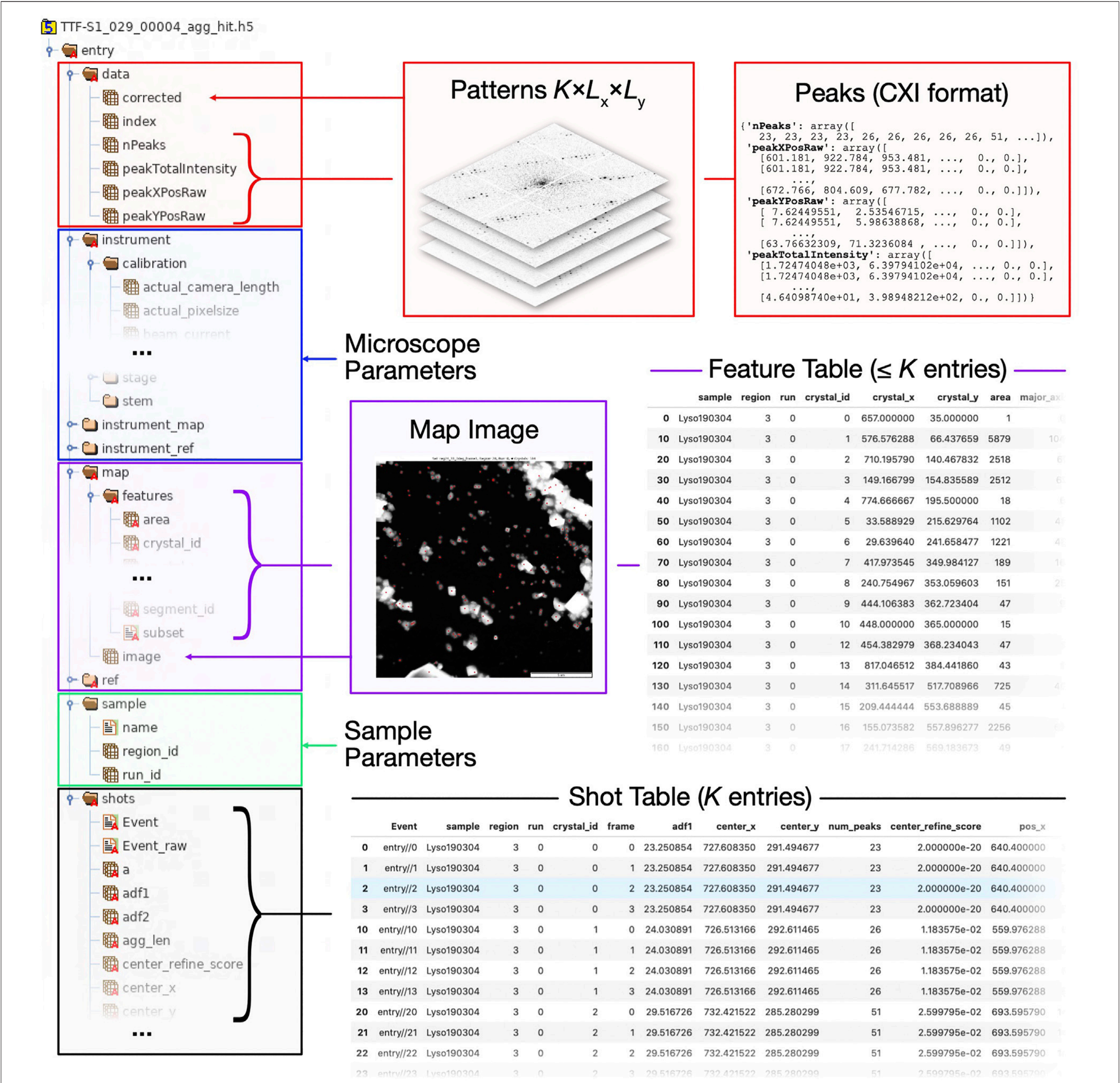
HDF5 files are internally organized into *groups* and *datasets*, roughly corresponding to folders and files in a file system. Datasets can be arrays of arbitrary dimension, and have a uniquely assigned data type. Mirroring the structure of a `Dataset` object, a diffractem data file contains primarily three types of entities:

- Tabular data such as the shot list and the feature list, are stored as groups comprising one-dimensional HDF5 datasets, each corresponding to a single table column (**Figures 8C,E**, respectively). Those tables are loaded into memory as pandas DataFrames on loading the dataset, as described in *Data Structures and File Format*.
- Data stacks, that is, arrays with an arbitrary number of dimension, where the first (dimension 0 in *Python* convention) dimension corresponds to a given shot (*Data Structures and File Format*), are stored as HDF5 datasets within the group `data` (**Figure 8A**). All stacks are mapped into dask arrays when loading the dataset.
- Ancillary per-file instrument metadata, which can be accessed using `Dataset.merge_meta` is stored in a hierarchical structures (**Figures 8B,D**).

In **Figure 8**, a typical HDF5 file structure, and how it maps to the attributes of a `Dataset` object, is illustrated.

4.2 Processing Functions

In this section we describe functions that act on data stored within `Dataset` objects, specifically image stacks and Bragg peak data.



```
{'nPeaks': array([
   23, 23, 23, 23, 26, 26, 26, 26, 26, 51, ...]),
 'peakXPosRaw': array([
   [601.181, 922.784, 953.481, ...,  0., 0.],
   [601.181, 922.784, 953.481, ...,  0., 0.],
       ...,
   [672.766, 804.609, 677.782, ...,  0., 0.]]),
 'peakYPosRaw': array([
   [ 7.62449551,  2.53546715, ...,  0., 0.],
   [ 7.62449551,  5.98638868, ...,  0., 0.],
       ...,
   [63.76632309, 71.3236084 , ...,  0., 0.]]),
 'peakTotalIntensity': array([
   [1.72474048e+03, 6.39794102e+04, ..., 0., 0.],
   [1.72474048e+03, 6.39794102e+04, ..., 0., 0.],
       ...,
   [4.64098740e+01, 3.98948212e+02, 0., 0.]])}
```

	sample	region	run	crystal_id	crystal_x	crystal_y	area	major_ax
0	Lyso190304	3	0	0	657.000000	35.000000	1	
10	Lyso190304	3	0	1	576.576288	66.437659	5879	
20	Lyso190304	3	0	2	710.195790	140.467832	2518	
30	Lyso190304	3	0	3	149.166799	154.835589	2512	
40	Lyso190304	3	0	4	774.666667	195.500000	18	
50	Lyso190304	3	0	5	33.588929	215.629764	1102	
60	Lyso190304	3	0	6	29.639640	241.658477	1221	
70	Lyso190304	3	0	7	417.973545	349.984127	189	
80	Lyso190304	3	0	8	240.754967	353.059603	151	
90	Lyso190304	3	0	9	444.106383	362.723404	47	
100	Lyso190304	3	0	10	448.000000	365.000000	15	
110	Lyso190304	3	0	12	454.382979	368.234043	47	
120	Lyso190304	3	0	13	817.046512	384.441860	43	
130	Lyso190304	3	0	14	311.645517	517.708966	725	
140	Lyso190304	3	0	15	209.444444	553.688889	45	
150	Lyso190304	3	0	16	155.073582	557.896277	2256	
160	Lyso190304	3	0	17	241.714286	569.183673	49	

	Event	sample	region	run	crystal_id	frame	adf1	center_x	center_y	num_peaks	center_refine_score	pos_x
0	entry//0	Lyso190304	3	0	0	0	23.250854	727.608350	291.494677	23	2.000000e-20	640.400000
1	entry//1	Lyso190304	3	0	0	1	23.250854	727.608350	291.494677	23	2.000000e-20	640.400000
2	entry//2	Lyso190304	3	0	0	2	23.250854	727.608350	291.494677	23	2.000000e-20	640.400000
3	entry//3	Lyso190304	3	0	0	3	23.250854	727.608350	291.494677	23	2.000000e-20	640.400000
10	entry//10	Lyso190304	3	0	1	0	24.030891	726.513166	292.611465	26	1.183575e-02	559.976288
11	entry//11	Lyso190304	3	0	1	1	24.030891	726.513166	292.611465	26	1.183575e-02	559.976288
12	entry//12	Lyso190304	3	0	1	2	24.030891	726.513166	292.611465	26	1.183575e-02	559.976288
13	entry//13	Lyso190304	3	0	1	3	24.030891	726.513166	292.611465	26	1.183575e-02	559.976288
20	entry//20	Lyso190304	3	0	2	0	29.516726	732.421522	285.280299	51	2.599795e-02	693.595790
21	entry//21	Lyso190304	3	0	2	1	29.516726	732.421522	285.280299	51	2.599795e-02	693.595790
22	entry//22	Lyso190304	3	0	2	2	29.516726	732.421522	285.280299	51	2.599795e-02	693.595790
23	entry//23	Lyso190304	3	0	2	3	29.516726	732.421522	285.280299	51	2.599795e-02	693.595790

FIGURE 8 | Structure of HDF5 file as used by *diffractem*. Typically, each HDF5 holds data from a single SerialED run on one grid region; a `Dataset` object typically manages data from many such files, automatically concatenating all information. On the left side, a tree view of the internal folder/dataset hierarchy of a HDF5 file is shown. On the right side, various types of information and (*via* arrows and braces) their location within the HDF5 file are shown. In compliance with the *NeXus* convention, all data is stored under a global/`entry` group. Explanations of the top-level groups **(A)** `data`, **(B)** `instrument` **(C)** `map`, **(D)** `sample`, and **(E)** `shots` are given in the main text.

A commonly used ancillary tool for the functionality described in this and the next section is the `PreProcOpts` class contained in the `diffractem.pre_proc_opts` module. The attributes of this class hold values of a large number of options pertaining to the entire data processing workflow, such as which steps of the pipeline should be applied by default, but also experiment parameters such as the accurate camera length and distortion. The attribute values of a `PreProcOpts` object are stored to and read from in a human-readable `.yaml` file, which can be continuously adjusted while working interactively on processing a dataset, and will in its final state document the exact parameters used, ensuring full reproducibility.

4.2.1 Stack Processing

Diffractem's functions for processing image stacks as required for pre-processing (see *Pre-Processing*) are contained in the `diffractem.proc2d` module. Examples for such functions are `correct_dead_pixels`, `lorentz_fit`, or `get_peaks`. All those take an image stack as described above (as NumPy arrays) as their first argument (with more arguments for individual options). They return either a processed version of the input stack (e.g., dead-pixel correction, background subtraction), per-shot data which can directly be merged into a `Dataset` shot list (e.g., pattern center finding, virtual detector signals), or more complex per-shot data which can be stored into stacks of a `Dataset` object (e.g., peak finding, azimuthal averaging).

Two special, particularly relevant functions contained in `proc2d` are `get_pattern_info` and `correct_image`, both of which represent multi-step pipelines for getting information (such as pattern center and Bragg peaks) from each shot, and for computing processed images (having undergone e.g. dead-pixel correction and background subtraction), respectively. In contrast to the other functions, these two act on larger-than-memory image stacks stored as dask arrays (like in a `Dataset` object, see *Stacks and Memory Management*), and have their parameters defined *via* `PreProcOpts` objects. These two functions encapsulate computationally heavy, but independent (per-shot) steps of pre-processing, and hence are preferably using parallel execution. This is implemented using the `dask.distributed` scheduler, which besides its ease of use provides convenient real-time progress reporting *via* a web interface. Please consult the supplementary Jupyter notebook `preprocessing.ipynb` for an example pre-processing workflow.

4.2.2 Peak Processing

Another set of processing functions, acting on Bragg peak positions, is contained in the `diffractem.proc_peaks` module. This comprises functions for refinement of the zero-order peak positions (pattern center) *via* matching of Friedel mates (see *Pattern Centering and Peak Finding*), getting pair-wize distances from all observed peaks (pattern autocorrelation function), and the `Cell` class, which provides functionality for unit-cell refinement as described in *Unit-Cell Refinement*. An example for the peak refinement workflow using a `Cell` object and pattern autocorrelation functions is provided in the supplementary Jupyter notebook `peak_processing.ipynb`.

4.3 Integration With CrystFEL

For all tasks that are less specific to SerialED, but pertain to (serial) crystallography in general, diffractem provides interfaces to the CrystFEL package, in particular its central command-line tools `indexamajig` and `partialator`, as well as the validation programs for merged diffraction intensities `compare_hkl` and `check_hkl`. Also, functionality to parse and manipulate `.stream-files`, CrystFEL's output format for pattern indexing and integration results is included. Depend on the task at hand, diffractem either calls the executables directly, or generates the required input files and a shell script containing the corresponding function calls. The functionality for integration with CrystFEL are mostly contained in the `diffractem.tools` module. While in the supplementary Jupyter notebooks, the usage of the pertinent tools is explained in detail, here we only give a brief overview of the most important functionality, especially where deviating from the standard CrystFEL workflow.

4.3.1 Indexing and Integration

Indexing and integration (*Indexing Using pinkIndexer* and *Integration of Intensities From Indexing Results*, respecitvely) in CrystFEL are performed using the `indexamajig` program As input, it requires a list of HDF5 data files (`.lst`) containing diffraction patterns and (optionally) peak positions, a geometry file (`.geom`), and a unit cell specification (`.cell` or `.pdb`). Using the `tools.make_geometry` function, the geometry file can be automatically generated from a `PreProcOpts` object (or, respectively, the corresponding `.yaml` file), which automatically handles elliptical distortion found as described in *Ellipticity Finding*. Similarly, the specification of a unit cell after refinement as described in *Unit-Cell Refinement* can automatically be generated using the `export` method of a `proc_peaks.Cell` object. The `indexamajig` executable can be called including all pertinent options (as defined in a diffractem `PreProcOpts` object) using the `tools.call_indexamajig` and `tools.call_indexamajig_slurm` functions, where the latter sets up intermediate files and a shell script for execution through a *SLURM* queue submission system. Optionally, those, along with the geometry, cell, and virtual data files (see below), can be packed into a `.tar.gz` archive for convenient uploading to a computing cluster.

Diffraction pattern indexing as described in *Indexing Using pinkIndexer* requires the positions of found peaks in each diffraction pattern as its primary raw-data input. CrystFEL's `indexamajig` tightly couples indexing and integration of peak intensities from image data into a single, inseparable step, as described in (White, 2019). While the file format described in *Diffractem Data Files* is compatible with CrystFEL and could directly used for indexing and integration in a single run, for SerialED this approach is hampered by two prohibitive shortcomings. First, the residual movement of the zero-order beam inherent to SerialED, even if known, cannot be natively accounted for by CrystFEL, precluding proper indexing of SerialED patterns from the Bragg reflections either found in the patterns or already stored in the files during preprocessing. Second, SerialED requires a computationally intensive grid search approach to indexing. Coupling indexing and peak integration into a single step hence makes it impractical to optimize the (relatively fast) integration, and would require transfer of the full dataset (as needed for integration) if indexing is offloaded to off-site computing clusters.

As shown in **Figure 2**, diffractem circumvents these issues by not running indexing on the actual data files, but on a (single) *virtual* file, which is generated using the `Dataset.write_virtual_file` method and does not carry actual diffraction data. The virtual file, while being a fully valid diffractem and CrystFEL HDF5 file, only contains the shot list and found Bragg peaks in CXI format, which are shifted for each pattern such that position of the zero-order beam remains at the center of the detector. `indexamajig` can now be run on the virtual file, yielding the indexing results (that is, the reciprocal-space lattice vectors in the laboratory frame, for each crystal found in the diffraction patterns) in `.stream` format. All book-keeping to associate patterns in the virtual and actual files is transparently performed using items in the shot tables, and the `--copy-hdf5-field` option of `indexamajig`.

For peak integration, we modified CrystFEL by introducing a new option to, instead of finding indexing solutions from Bragg reflections, read reciprocal-space lattice vectors and beam shift coordinates from a plain-text *solution* file (extension `.sol`), and proceed with the standard prediction and integration pipeline from there. To generate the solution file from the computed indexing parameters (in `.stream` format), the method `Dataset.get_indexing_solution` can be used, which transparently handles the case of integrating patterns that have been computed from a different range of movie frames (see *Sorting and Aggregation*) than that initially used for indexing. For the more simple case where the data that shall be integrated is identical to those that were used to generate the indexing solution, the command-line tool `stream2sol` which is included in diffractem, can be used alternatively.

Please see the supplementary notebook `indexing.ipynb` for a detailed step-by-step guide of indexing and integration.

4.3.2 Merging and Validation

The merging of single Bragg peak observations from all recorded diffraction patterns as described in *Indexing Using pinkIndexer* is performed using the `partialator` command-line program contained in CrystFEL. Diffractem includes a corresponding wrapper function `tools.call_partialator`. It provides a convenient way to generate `partialator` calls from within Jupyter notebooks, also providing options to run different merging settings (e.g., with and without post-refinement or resolution cut-offs) in parallel or sequentially, optionally generating a script for submission to a *SLURM* cluster queue submission system.

Finally, the merged intensities contained in `.hkl` files can be analyzed from Jupyter notebooks by wrapping CrystFEL's `check_hkl` and `compare_hkl` command-line tools into the `tools.analyze_hkl` function, which provides means to automatically validate the results of many different integration and merging parameters in parallel, and wraps the results in *pandas* DataFrames. Please see the supplementary notebook `merging.ipynb` for an example of the merging and validation steps.

4.4 Displaying Data

In order to visualize datasets being processed by diffractem, two tools with markedly different scope are provided, as shown in **Figure 9**. Firstly, the view method of a `Dataset` (*Data Structures and File Format*) allows for quick interactive inspection of diffraction data within a Jupyter notebook, which is especially helpful for tuning of processing parameters. Secondly, the stand-alone program `edview` provides a simple graphical interface to browse through SerialED data, including correlative display of mapping and diffraction data. In **Figure 9**, screen-shots of both tools are shown.

4.4.1 Dataset.view

An interactive viewer for diffraction data can be directly used within Jupyter notebooks in the web browser, where data are being processed. The viewer is called by invoking `ds.view (<...>)`, where `ds` is a `Dataset` object and `<...>` represents additional calling arguments. The viewer shows the data stack accessible *via* the `Dataset. diff_data` attribute (which points to the data stack containing diffraction data), and, if present as CXI-formatted data stacks, detected Bragg peaks. Finally, if columns `center_x` and `center_y` are present in the shot table, the position of the pattern center (zero-order beam) is shown as a cross-hair.

Importantly, `Dataset.view` acts on diffraction data stored as *dask* array (Dask Development Team, 2016), which are typically not in memory, but either on disk, or not even computed yet (lazy evaluation), if the `Dataset` object has not been written to disk. They are then loaded and/or computed on-the-fly for each displayed image. This makes `Dataset.view` especially suitable for interactive tuning of pre-processing parameters (such as peak-finding sensitivity thresholds) on a few selected shots, before the full computation is performed.

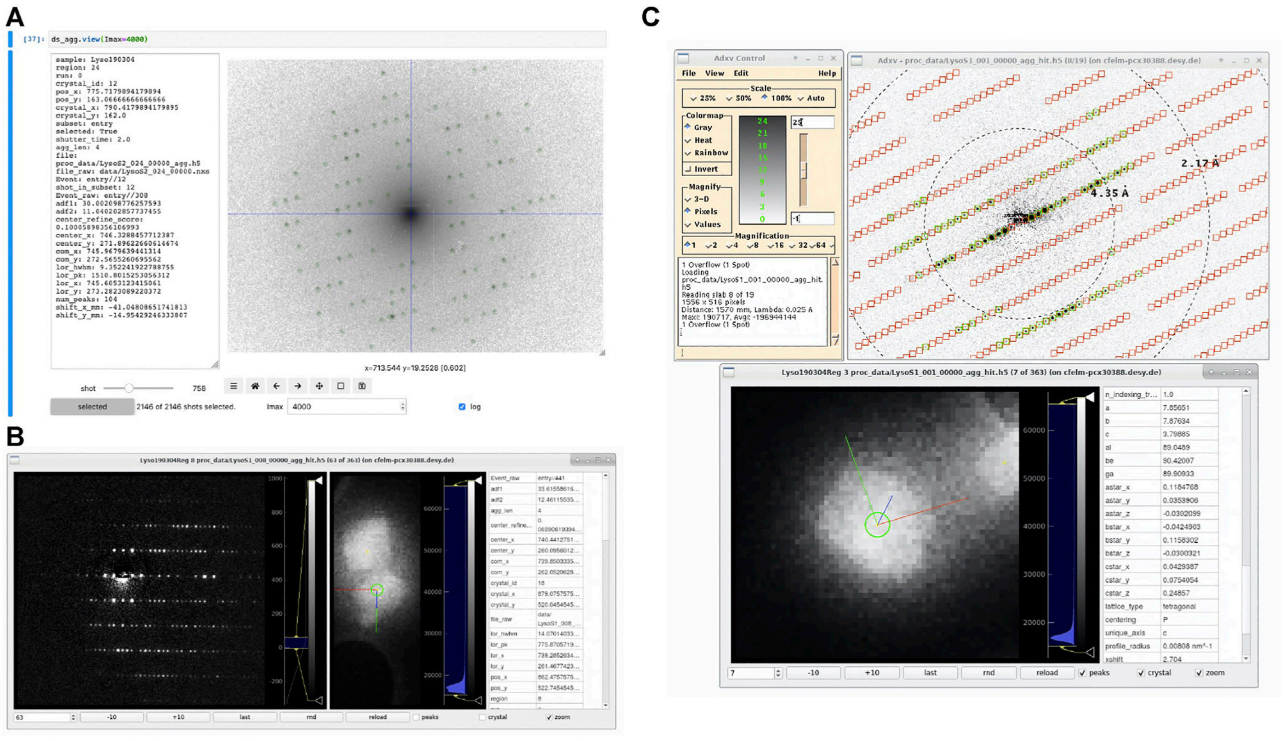
In the supplementary Jupyter notebook `preprocessing.ipynb`, the use of `Dataset.view` is illustrated at various points.

4.4.2 edview

The second option for displaying diffraction data is the stand-alone viewer `edview`, which is available from the command line after installation of diffractem. As input to `edview`, single HDF5 data files, list files, multiple data files (*via* file wildcards), or a `.stream` file can be provided. In the latter case, indexing solutions (Bragg spot predictions and real-space lattice vectors) can be displayed. `edview` shows both diffraction data and, if present, the overview maps taken in the course of a SerialED data acquisition from a grid region, including an indicator to show which crystal on the map an individual pattern belongs to. For displaying the diffraction data, either a built-in display window (*via* the command-line option `--internal`) or *adxv* (Arvai, 2020), which is controlled by `edview` *via* a local communication socket, can be used. If indexing information is present for a given shot, the projected directions of the real-space lattice vectors $\mathrm{a}, \mathrm{b}, \mathrm{c}$ (with fixed length) are overlaid on the currently displayed crystal (if "zoom" is checked).

4.5 Simple On-Line Pre-processing Using quick_proc

While diffractem has been designed with the usage from Jupyter notebooks in mind, there may be situations where it is preferable to run the pre-processing pipeline, up to the point of aggregated, corrected, and background-subtracted images, from the command-line. Hence, the command-line tool `quick_proc` is provided by diffractem, which executes those steps according to settings defined in a `.yaml` file, just as for the standard processing in notebooks (*Processing Functions*). Furthermore, `quick_proc` can run in an on-line analysis mode (using the flag `--wait-for-files`), where it waits for new data files from the experiment to arrive, then executes the processing, and adds the newly processed files to a `.lst` file for use with CrystFEL or viewing using `edview`. Running `quick_proc -h` provides a full reference of options.


A
B
C
x=713.544 y=19.2528 [0.602]
shot
758
selected
Imax
4000
log
Lyso190304Reg 8 proc_data/LysoS1_008_00000_agg_hit.h5 (61 of 363) (on cfelm-pcx30388.desy.de)
63
-10
+10
last
rnd
reload
peaks
crystal
zoom
Adxv Control
File
View
Edit
Help
Scale
25%
50%
100%
Auto
Colormap
Gray
Heat
Rainbow
Invert
Magnify
3-D
Pixels
Values
Magnification
1
2
4
8
16
32
64
Adxv - proc_data/LysoS1_001_00000_agg_hit.h5 (8/19) (on cfelm-pcx30388.desy.de)
2.17 Å
4.35 Å
Lyso190304Reg 3 proc_data/LysoS1_001_00000_agg_hit.h5 (7 of 363) (on cfelm-pcx30388.desy.de)
7
-10
+10
last
rnd
reload
peaks
crystal
zoom

```
[37]: ds_agg.view(Imax=4000)
```

```
sample: Lyso190304
region: 24
run: 0
crystal_id: 12
pos_x: 775.7179894179894
pos_y: 163.06666666666666
crystal_x: 790.4179894179895
crystal_y: 162.0
subset: entry
selected: True
shutter_time: 2.0
agg_len: 4
file:
proc_data/LysoS2_024_00000_agg.h5
file_raw: data/LysoS2_024_00000.nxs
Event: entry//12
shot_in_subset: 12
Event_raw: entry//308
adf1: 30.002098776257593
adf2: 11.040202857737455
center_refine_score:
0.10005898356106993
center_x: 746.3288457712387
center_y: 271.89622660614674
com_x: 745.9679639441314
com_y: 272.5655260695562
lor_hwhm: 9.352241922788755
lor_pk: 1510.8015253056312
lor_x: 745.6053123415061
lor_y: 273.2823089220372
num_peaks: 104
shift_x_mm: -41.04808651741813
shift_y_mm: -14.95429246333807
```

2146 of 2146 shots selected.

Event_raw	entry//441
adf1	33.61558616...
adf2	12.46115535...
agg_len	4
center_refine...	0. 06590619394...
center_x	740.4412751...
center_y	260.0956012...
com_x	739.8503335...
com_y	262.0520628...
crystal_id	18
crystal_x	879.0757575...
crystal_y	520.0454545...
file_raw	data/ LysoS1_008_...
lor_hwhm	14.07014033...
lor_pk	775.8705719...
lor_x	739.2852634...
lor_y	261.4677423...
pos_x	862.4757575...
pos_y	522.7454545...
region	8

```
1 Overflow (1 Spot)
Loading
proc_data/LysoS1_001_00000_agg_hit.
h5
Reading slab 8 of 19
1556 x 516 pixels
Distance: 1570 mm, Lambda: 0.025 A
MaxI: 190717, AvgI: -196944144
1 Overflow (1 Spot)
```

n_indexing_tr...	1.0
a	7.85651
b	7.87634
c	3.79885
al	89.0469
be	90.42007
ga	89.90933
astar_x	0.1184768
astar_y	0.0353906
astar_z	-0.0302099
bstar_x	-0.0424903
bstar_y	0.1158302
bstar_z	-0.0300321
cstar_x	0.0429387
cstar_y	0.0754054
cstar_z	0.24857
lattice_type	tetragonal
centering	P
unique_axis	c
profile_radius	0.00808 nm^-1
xshift	2.704

FIGURE 9 | Screenshots of diffraction viewing tools of *diffractem*. **(A)** Dataset. view running as an interactive widget inside a Jupyter notebook in a web browser. The diffraction pattern is shown on logarithmic scale, which is particularly useful to assess the quality of pattern center and peak finding at low resolutions; the pattern center and Bragg peaks are shown as blue cross-hair and green circles, respectively. On the left, data from the shot table for the shown pattern are displayed. The controls at the bottom allow to move between shots and set display parameters **(B)** edview running in internal-viewer mode. In three columns, the diffraction pattern, the map image (optionally zoomed into the shown crystal) and meta-data from the shot table *and* the indexing result from a. stream file for the shown diffraction pattern are shown, respectively. Image controls are at the bottom. **(C)** edview in external-viewer mode, in which the diffraction pattern is displayed through *adxv* (Arvai, 2020). In the pattern, found peaks (green circles) and predicted Bragg spot positions from the indexing solution (red squares) are shown. In the bottom edview window, the corresponding crystal is shown. The directions of the real-space lattice vectors $\vec{a}, \vec{b}, \vec{c}$ are shown in red, green, and blue, respectively.

5 DISCUSSION AND OUTLOOK

Using the pipeline comprising *CrystFEL* and *diffractem* as described in this article, processing SerialED datasets of high quality becomes a straightforward exercise, and tackling more challenging cases becomes viable. Still, there is plenty of room for future work. Besides usability improvements for non-expert users, such as a graphical program interface for basic operations or functions for reasonable automatic adjustment of parameters for a given sample, there are more fundamental aspects which can profit from further development. A rather obvious starting point for future work could be inclusion of a cell-finding algorithm similar to that presented in (Jiang et al., 2009), or even an entirely new method for indexing that would be based on considering peak data from the entire dataset instead of acting on individual patterns, similarly to single-particle analysis (Scheres, 2012) or expand-maximize-compress algorithms in diffractive imaging (Loh and Elser, 2009). Similarly, a more systematic study of partiality modeling for electrons is required, where partiality is especially prevalent due to the small crystal sizes (and concomitantly wide rocking curves) combined with a very monochromatic beam. Another field of study are the effects of dynamical diffraction arising from multiple scattering, which depend on subtle details that are often challenging to grasp, in particular for biological samples made from light elements (Subramanian et al., 2015; Gallagher-Jones et al., 2019; Latychevskaia and Abrahams, 2019; Nannenga and Gonen, 2019). While often considered deleterious for structure solution, careful inclusion of dynamical diffraction can lead to unique insight into molecular configurations (Palatinus et al., 2017; Brázda et al., 2019) and even might be able to solve the phasing problem for electron crystallography (Donatelli and Spence, 2020). Especially regarding the latter point, SerialED can provide the unique advantage of being able to selectively solve structures from sub-sets of data containing crystals from a given size bracket only.

While there is a large scope for future developments, already in its current state of development SerialED can provide high-resolution structures of even the most demanding nano-crystalline samples (Bücker et al., 2020). Data analysis, while not yet as established as for rotation techniques, is becoming a more and more routine task, helped by packages such as those described in this work. Meanwhile, the *diffractem* package (as well as *CrystFEL*, which provides much of the fundamental functionality) is under constant development, as to keep making SerialED data processing more efficient, powerful, and user friendly; we hence suggest to regularly check the webpage at https://github.com/robertbuecker/diffractem for updates.

DATA AVAILABILITY STATEMENT

The raw diffraction data and indexed/integrated `stream` files which the examples in this paper have been derived from is available from the Max Planck Digital Library EDMOND data repository at https://dx.doi.org/10.17617/3.53. The diffractem software, along with installation instructions, is available under the terms of the GNU Lesser General Public License 2.1 or higher at https://github.com/robertbuecker/diffractem. CrystFEL is available at https://www.desy.de/~twhite/crystfel/ under the terms of the GNU General Public License 3.0. A set of example Jupyter notebooks explaining the processing pipeline in detail is included as **Supplementary Material**. The notebooks themselves, including all ancillary files required to reproduce our workflow can be downloaded at https://github.com/robertbuecker/serialed-examples.

AUTHOR CONTRIBUTIONS

RB and RM. conceived the serial electron diffraction concept. RB and PH developed the SerialED processing pipeline. RB wrote the *diffractem* software. PH wrote the extensions to *CrystFEL* to adapt to our analysis pipeline. RB and PH wrote the manuscript.

FUNDING

This work was funded by the Max Planck Society, the Natural Sciences and Engineering Research Council of Canada (PH, RM.), the Fonds de recherche du Québec (PH), and the BWFGB Israel-Hamburg project LOM 2018 (RB).

ACKNOWLEDGMENTS

We acknowledge many helpful discussions with and support for modifying *CrystFEL* by Thomas White, and assistance with crystallographic refinement by Pedram Mehrabi. We thank Anton Barty, Valerio Mariani, and Oleksandr Yefanov for making *peakfinder8* available under the terms of the GNU Lesser General Public License.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fmolb.2021.624264/full#supplementary-material

REFERENCES

Adams, P. D., Afonine, P. V., Bunkóczi, G., Chen, V. B., Davis, I. W., Echols, N., et al. (2010). PHENIX: A Comprehensive Python-Based System for Macromolecular Structure Solution. *Acta Crystallogr. D Biol. Cryst.* 66, 213–221. doi:10.1107/S0907444909052925

Arvai, A. (2020). *Adxv–A Program to Display X-Ray Diffraction Images*. doi:10.1109/iccc49264.2020.9257230

Barty, A., Kirian, R. A., Maia, F. R. N. C., Hantke, M., Yoon, C. H., White, T. A., et al. (2014). Cheetah: Software for High-Throughput Reduction and Analysis of Serial Femtosecond X-Ray Diffraction Data. *J. Appl. Cryst.* 47, 1118–1131. doi:10.1107/S1600576714007626

Beyerlein, K. R., White, T. A., Yefanov, O., Gati, C., Kazantsev, I. G., Nielsen, N. F.-G., et al. (2017). FELIX: An Algorithm for Indexing Multiple Crystallites in X-Ray Free-Electron Laser Snapshot Diffraction Images. *J. Appl. Cryst.* 50, 1075–1083. doi:10.1107/S1600576717007506

Brázda, P., Palatinus, L., and Babor, M. (2019). Electron Diffraction Determines Molecular Absolute Configuration in a Pharmaceutical Nanocrystal. *Science*. 364, 667–669. doi:10.1126/science.aaw2560

Brehm, W., and Diederichs, K. (2014). Breaking the Indexing Ambiguity in Serial Crystallography. *Acta Cryst. D Biol. Crystallogr.* 70, 101–109. doi:10.1107/S1399004713025431

Bücker, R., Hogan-Lamarre, P., Mehrabi, P., Schulz, E. C., Bultema, L. A., Gevorkov, Y., et al. (2020). Serial Protein Crystallography in an Electron Microscope. *Nat. Commun.* 11, 996. doi:10.1038/s41467-020-14793-0

Chapman, H. N., Fromme, P., Barty, A., White, T. A., Kirian, R. A., Aquila, A., et al. (2011). Femtosecond X-Ray Protein Nanocrystallography. *Nature*. 470, 73–77. doi:10.1038/nature09750

Chapman, H. N. (2019). X-Ray Free-Electron Lasers for the Structure and Dynamics of Macromolecules. *Annu. Rev. Biochem.* 88, 35–58. doi:10.1146/annurev-biochem-013118-110744

Clabbers, M. T. B., and Abrahams, J. P. (2018). Electron Diffraction and Three-Dimensional Crystallography for Structural Biology. *Crystallogr. Rev.* 24, 176–204. doi:10.1080/0889311X.2018.1446427

Dask Development Team (2016). *Dask: Library for Dynamic Task Scheduling.*

Donatelli, J. J., and Spence, J. C. H. (2020). Inversion of Many-Beam Bragg Intensities for Phasing by Iterated Projections: Removal of Multiple Scattering Artifacts from Diffraction Data. *Phys. Rev. Lett.* 125, 065502. doi:10.1103/PhysRevLett.125.065502

Emsley, P., Lohkamp, B., Scott, W. G., and Cowtan, K. (2010). Features and Development ofCoot. *Acta Crystallogr. D Biol. Cryst.* 66, 486–501. doi:10.1107/s0907444910007493

Feller, W. (2015). "On Probability Problems in the Theory of Counters," in *Selected Papers I*. Editors W. Feller, R. L. Schilling, Z. Vondraček, and W. A. Woyczyński (Cham: Springer International Publishing), 751–759. doi:10.1007/978-3-319-16859-3_39

Gallagher-Jones, M., Ophus, C., Bustillo, K. C., Boyer, D. R., Panova, O., Glynn, C., et al. (2019). Nanoscale Mosaicity Revealed in Peptide Microcrystals by Scanning Electron Nanodiffraction. *Commun. Biol.* 2, 26. doi:10.1038/s42003-018-0263-8

Gati, C., Bourenkov, G., Klinge, M., Rehders, D., Stellato, F., Oberthür, D., et al. (2014). Serial Crystallography Onin Vivogrown Microcrystals Using Synchrotron Radiation. *Int. Union Crystallogr. J.* 1, 87–94. doi:10.1107/S2052252513033939

Gemmi, M., and Lanza, A. E. (2019). 3D Electron Diffraction Techniques. *Acta Crystallogr. Sect B.* 75, 495–504. doi:10.1107/s2052520619007510

Gemmi, M., Mugnaioli, E., Gorelik, T. E., Kolb, U., Palatinus, L., Boullay, P., et al. (2019). 3D Electron Diffraction: The Nanocrystallography Revolution. *ACS Cent. Sci.* 5, 1315–1329. doi:10.1021/acscentsci.9b00394

Gevorkov, Y., Barty, A., Brehm, W., White, T. A., Tolstikova, A., Wiedorn, M. O., et al. (2020). pinkIndexer - a Universal Indexer for Pink-Beam X-Ray and Electron Diffraction Snapshots. *Acta Cryst. Sect A.* 76, 121–131. doi:10.1107/S2053273319015559

Ginn, H. M., Roedig, P., Kuo, A., Evans, G., Sauter, N. K., Ernst, O. P., et al. (2016). TakeTwo: An Indexing Algorithm Suited to Still Images with Known Crystal Parameters. *Acta Cryst. Sect D Struct. Biol.* 72, 956–965. doi:10.1107/S2059798316010706

Glaeser, R. M. (2019). How Good Can Single-Particle Cryo-EM Become? what Remains before it Approaches its Physical Limits? *Annu. Rev. Biophys.* 48, 45–61. doi:10.1146/annurev-biophys-070317-032828

Gonen, T., Cheng, Y., Sliz, P., Hiroaki, Y., Fujiyoshi, Y., Harrison, S. C., et al. (2005). Lipid-protein Interactions in Double-Layered Two-Dimensional AQP0 Crystals. *Nature*. 438, 633–638. doi:10.1038/nature04321

Hattne, J., Echols, N., Tran, R., Kern, J., Gildea, R. J., Brewster, A. S., et al. (2014). Accurate Macromolecular Structures Using Minimal Measurements from X-Ray Free-Electron Lasers. *Nat. Methods*. 11, 545–548. doi:10.1038/nmeth.2887

Hattne, J., Shi, D., Glynn, C., Zee, C.-T., Gallagher-Jones, M., Martynowycz, M. W., et al. (2018). Analysis of Global and Site-specific Radiation Damage in Cryo-EM. *Structure*. 26, 759–766.e4. doi:10.1016/j.str.2018.03.021

Henderson, R., and Unwin, P. N. T. (1975). Three-dimensional Model of Purple Membrane Obtained by Electron Microscopy. *Nature*. 257, 28–32. doi:10.1038/257028a0

Henderson, R., Baldwin, J. M., Ceska, T. A., Zemlin, F., Beckmann, E., and Downing, K. H. (1990). Model for the Structure of Bacteriorhodopsin Based on High-Resolution Electron Cryo-Microscopy. *J. Mol. Biol.* 213, 899–929. doi:10.1016/S0022-2836(05)80271-2

Henderson, R. (1995). The Potential and Limitations of Neutrons, Electrons and X-Rays for Atomic Resolution Microscopy of Unstained Biological Molecules. *Quart. Rev. Biophys.* 28, 171–193. doi:10.1017/S003358350000305X

Jiang, L., Georgieva, D., Zandbergen, H. W., and Abrahams, J. P. (2009). Unit-cell Determination from Randomly Oriented Electron-Diffraction Patterns. *Acta Crystallogr. D Biol. Cryst.* 65, 625–632. doi:10.1107/S0907444909003163

Kabsch, W. (2014). Processing of X-Ray Snapshots from Crystals in Random Orientations. *Acta Cryst. D Biol. Crystallogr.* 70, 2204–2216. doi:10.1107/S1399004714013534

Karplus, P. A., and Diederichs, K. (2012). Linking Crystallographic Model and Data Quality. *Science*. 336, 1030–1033. doi:10.1126/science.1218231

Kolb, U., Gorelik, T., Kübel, C., Otten, M. T., and Hubert, D. (2007). Towards Automated Diffraction Tomography: Part I-Data Acquisition. *Ultramicroscopy*. 107, 507–513. doi:10.1016/j.ultramic.2006.10.007

Könnecke, M., Akeroyd, F. A., Bernstein, H. J., Brewster, A. S., Campbell, S. I., Clausen, B., et al. (2015). The NeXus Data Format. *J. Appl. Cryst.* 48, 301–305. doi:10.1107/s1600576714027575

Latychevskaia, T., and Abrahams, J. P. (2019). Inelastic Scattering and Solvent Scattering Reduce Dynamical Diffraction in Biological Crystals. *Acta Crystallogr. Sect B*. 75, 523–531. doi:10.1107/s2052520619009661

Li, C., Li, X., Kirian, R., Spence, J. C. H., Liu, H., and Zatsepin, N. A. (2019). SPIND: A Reference-Based Auto-Indexing Algorithm for Sparse Serial Crystallography Data. *Int. Union Crystallogr. J.* 6, 72–84. doi:10.1107/S2052252518014951

Liebschner, D., Afonine, P. V., Baker, M. L., Bunkóczi, G., Chen, V. B., Croll, T. I., et al. (2019). Macromolecular Structure Determination Using X-Rays, Neutrons and Electrons: Recent Developments in Phenix. *Acta Cryst. Sect D Struct. Biol.* 75, 861–877. doi:10.1107/S2059798319011471

Loh, N-T. D., and Elser, V. (2009). Reconstruction Algorithm for Single-Particle Diffraction Imaging Experiments. *Phys. Rev. E* 80, 026705. doi:10.1103/PhysRevE.80.026705

Maia, F. R. N. C. (2012). The Coherent X-Ray Imaging Data Bank. *Nat. Methods*. 9, 854–855. doi:10.1038/nmeth.2110

Mehrabi, P., Bücker, R., Bourenkov, G., Ginn, H., von Stetten, D., Mueller-Werkmeister, H. M., et al. (2020). Serial Femtosecond and Serial Synchrotron Crystallography Yield Data of Equivalent Quality: A Systematic Comparison. *bioRxiv* 2020, 08.21.257170. doi:10.1101/2020.08.21.257170

Nakane, T., Kotecha, A., Sente, A., McMullan, G., Masiulis, S., Brown, P. M. G. E., et al. (2020). Single-particle Cryo-EM at Atomic Resolution. *Nature*. 587, 152–156. doi:10.1038/s41586-020-2829-0

Nannenga, B. L., and Gonen, T. (2019). The Cryo-EM Method Microcrystal Electron Diffraction (MicroED). *Nat. Methods*. 16, 369–379. doi:10.1038/s41592-019-0395-x

Nannenga, B. L., Shi, D., Leslie, A. G. W., and Gonen, T. (2014). High-resolution Structure Determination by Continuous-Rotation Data Collection in MicroED. *Nat. Methods*. 11, 927–930. doi:10.1038/nmeth.3043

Nederlof, I., van Genderen, E., Li, Y.-W., and Abrahams, J. P. (2013). A Medipix Quantum Area Detector Allows Rotation Electron Diffraction Data Collection from Submicrometre Three-Dimensional Protein Crystals. *Acta Crystallogr. D Biol. Cryst.* 69, 1223–1230. doi:10.1107/S0907444913009700

Palatinus, L., Brázda, P., Boullay, P., Perez, O., Klementová, M., Petit, S., et al. (2017). Hydrogen Positions in Single Nanocrystals Revealed by Electron Diffraction. *Science* 355, 166–169. doi:10.1126/science.aak9652

Pandas Development Team (2020). *Pandas*

Rossmann, M. G. (1979). Processing Oscillation Diffraction Data for Very Large Unit Cells with an Automatic Convolution Technique and Profile Fitting. *J. Appl. Cryst.* 12, 225–238. doi:10.1107/S0021889879012218

Scheres, S. H. W. (2012). A Bayesian View on Cryo-EM Structure Determination. *J. Mol. Biol.* 415, 406–418. doi:10.1016/j.jmb.2011.11.010

Sheldrick, G. M. (2010). Experimental Phasing withSHELXC/D/E: Combining Chain Tracing with Density Modification. *Acta Crystallogr. D Biol. Cryst.* 66, 479–485. doi:10.1107/S0907444909038360

Shi, D., Nannenga, B. L., Iadanza, M. G., and Gonen, T. (2013). Three-dimensional Electron Crystallography of Protein Microcrystals. *eLife*. 2, e01345. doi:10.1088/1367-2630/16/1/01304110.7554/elife.01345

Smeets, S., and Wan, W. (2017). Serial Electron Crystallography: Merging Diffraction Data through Rank Aggregation. *J. Appl. Cryst.* 50, 885–892. doi:10.1107/S1600576717005854

Smeets, S., Zou, X., and Wan, W. (2018). Serial Electron Crystallography for Structure Determination and Phase Analysis of Nanocrystalline Materials. *J. Appl. Cryst.* 51, 1262–1273. doi:10.1107/S1600576718009500

Stellato, F., Oberthür, D., Liang, M., Bean, R., Gati, C., Yefanov, O., et al. (2014). Room-temperature Macromolecular Serial Crystallography Using Synchrotron Radiation. *Int. Union Crystallogr. J.* 1, 204–212. doi:10.1107/S2052252514010070

Su, J., Kapaca, E., Liu, L., Georgieva, V., Wan, W., Sun, J., et al. (2014). Structure Analysis of Zeolites by Rotation Electron Diffraction (RED). *Microporous Mesoporous Mater.* 189, 115–125. doi:10.1016/j.micromeso.2013.10.014

Subramanian, G., Basu, S., Liu, H., Zuo, J.-M., and Spence, J. C. H. (2015). Solving Protein Nanocrystals by Cryo-EM Diffraction: Multiple Scattering Artifacts. *Ultramicroscopy*. 148, 87–93. doi:10.1016/j.ultramic.2014.08.013

Wennmacher, J. T. C., Zaubitzer, C., Li, T., Bahk, Y. K., Wang, J., van Bokhoven, J. A., et al. (2019). 3D-structured Supports Create Complete Data Sets for Electron Crystallography. *Nat. Commun.* 10, 1–6. doi:10.1038/s41467-019-11326-2

White, T. A., Kirian, R. A., Martin, A. V., Aquila, A., Nass, K., Barty, A., et al. (2012). CrystFEL: a Software Suite for Snapshot Serial Crystallography. *J. Appl. Cryst.* 45, 335–341. doi:10.1107/S0021889812002312

White, T. A., Barty, A., Stellato, F., Holton, J. M., Kirian, R. A., Zatsepin, N. A., et al. (2013). Crystallographic Data Processing for Free-Electron Laser Sources. *Acta Crystallogr. D Biol. Cryst.* 69, 1231–1240. doi:10.1107/S0907444913013620

White, T. A., Mariani, V., Brehm, W., Yefanov, O., Barty, A., Beyerlein, K. R., et al. (2016). Recent Developments in CrystFEL. *J. Appl. Cryst.* 49, 680–689. doi:10.1107/S1600576716004751

White, T. A. (2014). Post-refinement Method for Snapshot Serial Crystallography. *Phil. Trans. R. Soc. B*. 369, 20130330–20130335. doi:10.1098/rstb.2013.0330

White, T. A. (2019). Processing Serial Crystallography Data with CrystFEL: a Step-by-step Guide. *Acta Cryst. Sect D Struct. Biol.* 75, 219–233. doi:10.1107/s205979831801238x

Winn, M. D., Ballard, C. C., Cowtan, K. D., Dodson, E. J., Emsley, P., Evans, P. R., et al. (2011). Overview of theCCP4 Suite and Current Developments. *Acta Crystallogr. D Biol. Cryst.* 67, 235–242. doi:10.1107/S0907444910045749

Yip, K. M., Fischer, N., Paknia, E., Chari, A., and Stark, H. (2020). Atomic-resolution Protein Structure Determination by Cryo-EM. *Nature*. 587, 157–161. doi:10.1038/s41586-020-2833-4

Zhang, D., Oleynikov, P., Hovmöller, S., and Zou, X. (2010). Collecting 3D Electron Diffraction Data by the Rotation Method. *Z. für Kristallographie* 225, 94–102. doi:10.1524/zkri.2010.1202

Zhang, Y.-b., Su, J., Furukawa, H., Yun, Y., Gándara, F., Duong, A., et al. (2013). Single-Crystal Structure of a Covalent Organic Framework. *J. Am. Chem. Soc.* 135, 16336–16339. doi:10.1021/ja409033p

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.