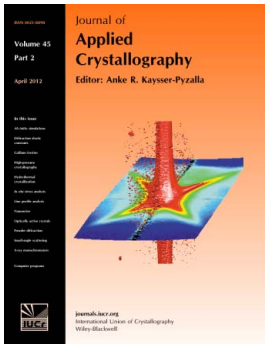
Journal of
Applied Crystallography
ISSN 0021-8898
Editor: **Anke R. Kaysser-Pyzalla**

CrystFEL: a software suite for snapshot serial crystallography

Thomas A. White, Richard A. Kirian, Andrew V. Martin, Andrew Aquila, Karol Nass, Anton Barty and Henry N. Chapman

J. Appl. Cryst. (2012). **45**, 335–341



Journal of
Applied Crystallography
ISSN 0021-8898

Received 20 July 2011
Accepted 18 January 2012

CrystFEL: a software suite for snapshot serial crystallography

Thomas A. White,[a]* Richard A. Kirian,[b] Andrew V. Martin,[a] Andrew Aquila,[a] Karol Nass,[a,c] Anton Barty[a] and Henry N. Chapman[a,c]

[a]Center for Free-Electron Laser Science, DESY, Notkestrasse 85, 22607 Hamburg, Germany, [b]Department of Physics, Arizona State University, Tempe, Arizona 85287, USA, and [c]University of Hamburg, Luruper Chaussee 149, 22761 Hamburg, Germany. Correspondence e-mail: taw@physics.org

In order to address the specific needs of the emerging technique of 'serial femtosecond crystallography', in which structural information is obtained from small crystals illuminated by an X-ray free-electron laser, a new software suite has been created. The constituent programs deal with viewing, indexing, integrating, merging and evaluating the quality of the data, and also simulating patterns. The specific challenges addressed chiefly concern the indexing and integration of large numbers of diffraction patterns in an automated manner, and so the software is designed to be fast and to make use of multi-core hardware. Other constituent programs deal with the merging and scaling of large numbers of intensities from randomly oriented snapshot diffraction patterns. The suite uses a generalized representation of a detector to ease the use of more complicated geometries than those familiar in conventional crystallography. The suite is written in C with supporting Perl and shell scripts, and is available as source code under version 3 or later of the GNU General Public License.

1. Introduction

The new technique of serial femtosecond crystallography (Chapman *et al.*, 2011) involves the illumination of many small crystals of proteins sequentially using an intense fourth-generation light source such as the Linac Coherent Light Source (LCLS; Emma *et al.*, 2010). Each pulse of the X-ray laser lasts for only a few tens or hundreds of femtoseconds and, since the radiation dose is very large, the crystal will be destroyed. Each crystal is used for only one exposure, and there is not sufficient time for any oscillation or rotation of the sample. As a result, the final integrated Bragg intensities must be constructed from 'snapshot' diffraction patterns containing partially recorded intensities, each pattern corresponding to a different crystal, and with no orientational relationships between the crystals. Furthermore, when the crystals are very small, the Fourier transform of the crystal shape itself (the 'shape transform') may come to dominate the sizes of the peak profiles. Each snapshot provides one slice through this shape transform; however, the required reflection intensities are proportional to its volume integral. In addition, the size and form of the shape transform vary from crystal to crystal. The Monte Carlo method of integration (Kirian *et al.*, 2010) provides a theoretical method to process data in this situation but relies on a very high redundancy in the data set. This fact makes it necessary to index and measure intensities from many thousands of patterns, so unsupervised automatic processing of such data is of the utmost importance. The software developed as a result could be applied to any technique in which 'snapshot' diffraction patterns are acquired in such a 'serial' manner.

A new software suite, called *CrystFEL*, has been created to address these concerns. Three programs are central to the suite in the initial version:

(1) *indexamajig*, for quickly indexing and integrating large numbers of diffraction patterns.

(2) *pattern_sim*, for diffraction pattern simulation.

(3) *process_hkl*, for merging Bragg intensities using the Monte Carlo method.

In addition to these, extra programs are provided to help with the individual stages of the data analysis:

(1) *check_hkl*, for calculating figures of merit for merged data.

(2) *compare_hkl*, for examining the differences between two sets of merged intensities.

(3) *render_hkl*, for plotting intensities, structure factors and redundancies in two and three dimensions.

(4) *sum_stack*, for summing diffraction patterns after peak detection to produce a two-dimensional 'virtual powder pattern', which can be used to quickly evaluate the amount of data collected.

(5) *powder_plot*, for summing data from a wider range of formats (image, reflection list or peak-list form) into one-dimensional 'powder' traces.

(6) *get_hkl*, which can perform various manipulations on reflection data, such as artificially 'twinning' their intensities, expanding them out to point groups of lower symmetry, adding noise or filtering reflections according to a template file.

(7) *hdfsee*, an image viewer.

CrystFEL accepts image data contained within a hierarchical data format (HDF5) container. The image viewer, *hdfsee*, may be used to examine images in this format, and can use a calibrated detector geometry or can overlay peak locations from the indexing or peak detection steps. Future versions of the software will incorporate an abstraction layer to allow the use of many more formats, enabling full use of the flexible detector geometry specification system described in the next section.

The software is intended to process 'clean' diffraction patterns, meaning that steps to remove electronic artefacts should have already been performed. Blank images in which no crystal intersected the

X-ray beam at the moment of the X-ray pulse should not be included in the input as far as possible, but the inclusion of such images should have no effect on the processing other than the speed. Since crystal hit rates in experiments so far have been around 5%, attempting to index all frames without this selection procedure would increase the indexing time by a factor of around 20. However, since these pre-processing steps are likely to be specific to a particular detector or experimental configuration, they are not implemented by any part of the *CrystFEL* suite. One possible route for performing this pre-processing is to create a piece of software based on the LCLS data analysis tools '*myana*', '*pyana*' or '*psana*'. *CrystFEL*'s indexing component could, in principle, be executed directly by these processing steps to create a streamlined data processing path.

The tightly knit structure of the suite, where file formats and code are shared between programs, has enabled a high degree of code reuse. This not only simplified the structure of the suite but led to a constant re-visitation of many parts of the code. This has resulted in many bugs, and even potential bugs, being removed at an early stage.

2. Detector geometry

Fourth-generation light sources have brought with them new detector technology, required to match the repetition rates called for by the 'serial crystallography' methodology. Many of these detectors, such as the pnCCD detector used in the CAMP instrument (Strüder *et al.*, 2010) or the CSPAD detector used in the CXI instrument at the LCLS, consist of multiple smaller detectors in some fixed or movable arrangement. The small sizes of the panels, combined with separate sets of readout electronics for each panel, help to achieve the required high readout rates. To take this into account, *CrystFEL*'s representation of a detector is broken down into one or more 'regions', each of which has its own camera length, position, resolution and other parameters. Programs forming part of the suite take a description of the detector geometry in a text file, allowing the use of the suite with many and varied detector geometries. To avoid the many problems that can arise from confusion over definitions of geometry – all too familiar to macromolecular crystallographers – a precise specification is used to define the detector geometry, illustrated in Fig. 1 and described below.

The raw data of each panel fit into the array of data taken from the input file, with the relevant range of pixels defined only in terms of minimum and maximum coordinates in the 'fast-scan' (fs) and 'slow-scan' (ss) directions. 'Fast scan' refers to the direction whose coordinate changes most quickly as the bytes in the input file are moved through in order, and 'slow scan' refers to the direction whose coordinate changes most slowly. All pixels in the input data block must be assigned to a panel, but regions of the detector can be marked as 'bad' if required, which means that they will be ignored at all stages of the analysis.

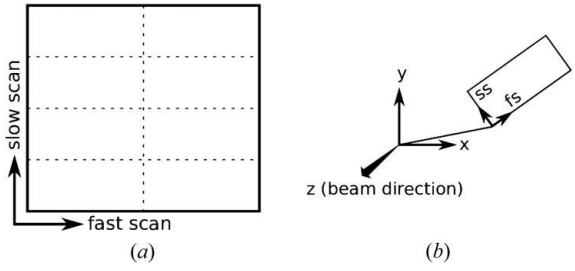


Figure 1
Specification of detector geometry in *CrystFEL*. (*a*) The input data array is broken into rectangular regions by specifying the minimum and maximum coordinates in the 'fast-scan' and 'slow-scan' directions, which are unambiguously defined with respect to the arrangement in memory of the data array itself. (*b*) For each region, the position of the corner (closest to the start of the data array) and the vectors corresponding to the fast-scan (**fs**) and slow-scan (**ss**) directions are specified in terms of a fixed laboratory coordinate system.

The role of the geometry description file is to set up the relationship between pixel locations in the raw image data and in the laboratory coordinate system. The laboratory coordinate system is defined by *CrystFEL* to have $+\mathbf{z}$ being the beam direction, $+\mathbf{y}$ pointing towards the zenith (directly upwards) and $+\mathbf{x}$ completing a right-handed coordinate system. However, this definition does not place any requirements on the representation of the data in the file. For each panel, the geometry description file specifies the coordinates in the laboratory coordinate system of the corner of the panel, meaning the point in the image that would appear first in the raw image array, in units of pixels. The file must then specify the direction, also in the laboratory coordinate system, that corresponds to each of the fast- and slow-scan directions. The direction is defined as a linear combination of the **x** and **y** directions, constituting a transformation matrix, and so arbitrary in-plane rotations of the detector are possible. Since there is no requirement for the direction vectors to have equal moduli, rectangular pixels could be accommodated, if it were to become necessary in the future, by more creative use of the vector combinations such as $\mathbf{fs} = \mathbf{x}$ and $\mathbf{ss} = 2\mathbf{y}$. Hexagonal pixels could also be used within this framework, with some wastage of space in the input data array.

3. Simulation of data: *pattern_sim*

It is important to be able to simulate data in order to test the algorithms. A fast nanocrystal diffraction simulation program, named *pattern_sim*, is included in *CrystFEL*, which simulates patterns in a manner similar to that described in the previous literature (Kirian *et al.*, 2010). The unit-cell dimensions are taken from a Protein Data Bank (PDB; Berman *et al.*, 2000) file; however, the structure factors themselves must be calculated by a separate means, such as using the *sfall* tool in *CCP4* (Winn *et al.*, 2011). The program is, in its initial version, only able to model the crystals as parallelepipeds with a specified number of unit cells along each of the three crystallographic axes, meaning that more complex shapes such as a hexagonal prism or a spherical crystal cannot currently be used. However, this restriction allows the intensity to be calculated as the product of the Laue interference functions and the squared structure factors, which is much more efficient than the equivalent sum over all unit cells:

$$I = \frac{\sin^2(\pi n_a \mathbf{q}\cdot\mathbf{a})}{\sin^2(\pi \mathbf{q}\cdot\mathbf{a})}\frac{\sin^2(\pi n_b \mathbf{q}\cdot\mathbf{b})}{\sin^2(\pi \mathbf{q}\cdot\mathbf{b})}\frac{\sin^2(\pi n_c \mathbf{q}\cdot\mathbf{c})}{\sin^2(\pi \mathbf{q}\cdot\mathbf{c})}\left|\mathbf{F}_\mathbf{q}\right|^2, \qquad (1)$$

where n_a, n_b and n_c are the number of unit cells that the parallelepiped has along the **a**, **b** and **c** directions, respectively. The vector **q** represents the point in reciprocal space at which the diffraction is to be calculated, and **a**, **b** and **c** are the direct-space unit-cell axes (which encode the orientation of the crystal as well as the shape of the unit cell). $\mathbf{F}_\mathbf{q}$ is the complex structure factor at reciprocal space point **q**. Suitable expressions for the scattering vector **q** are given by Kirian *et al.* (2010).

The program calculates the Laue functions in advance for the three required values of n_a, n_b and n_c in terms of $\mathbf{q}\cdot\mathbf{a}$, $\mathbf{q}\cdot\mathbf{b}$ and $\mathbf{q}\cdot\mathbf{c}$, respectively. By storing these values in lookup tables, the values required later can be quickly calculated by interpolation. This method avoids repetitive calculation of trigonometric functions and

has the additional advantage of providing a numerically stable calculation when the scalar product of the reciprocal space vector and a cell axis is zero or very small.

Since the pre-calculated structure factors give only the intensities when Bragg's law is exactly satisfied, further calculation must be performed to find the values between the Bragg peaks. Three methods are available in *pattern_sim*: 'mosaic', where the structure factor at the nearest reciprocal lattice point is taken; 'interpolate', where the intensities at the nearby reciprocal lattice points are interpolated trilinearly; and 'phased', where the relative phases of neighboring structure factors are taken into account. The 'phased' method accounts for the dark region that should appear between two extended Bragg peaks that have structure factors with opposite phases, but requires the most calculation. The 'mosaic' method is expected to be the least computationally demanding. For the highest possible accuracy, a future version of the software could allow a full, oversampled, three-dimensional molecular transform to be input.

The program calculates the absolute scattered intensity by multiplying the result of equation (1) by the incident photon flux density, the square of the Thomson scattering length and the solid angle of the pixel. The calculation is repeated for a number of sub-pixel units and the mean of the resulting values taken, in order to reduce the probability of missing fine structure in the pattern and consequently underestimating the intensity in each Bragg peak. Finite wavelength spread of the incident radiation can also be simulated by a similar method of sampling many different closely spaced wavelengths. In addition, convergence of the incident X-ray beam can be simulated within the limits of a small-angle approximation.

To further accelerate the calculation and enable the simulation of large data sets of tens of thousands of patterns or more, *pattern_sim* can take advantage of a graphics processing unit (GPU) *via* OpenCL, if it is available. The GPU calculation can be enabled by setting a command line switch and gave a speed-up of around 30 times on the test hardware. A separate test program, easily executed amongst other test programs as part of the installation procedure, verifies that no significant differences exist between the two implementations of the calculation. The typical total difference is around 0.3% of the total intensity given by the 'conventional' version, this small difference perhaps being accounted for by the single precision of the GPU's floating-point arithmetic as opposed to the double precision used in the conventional version.

In a final 'post-processing' step, *pattern_sim* can add Poisson noise to the results. It then stores the image in an HDF5 file suitable for input into the indexing stage or other processing pipelines, if required.

4. Pattern indexing and integration: *indexamajig*

The purpose of the indexing component of *CrystFEL*, *indexamajig*, is to facilitate the processing of large numbers of diffraction patterns in an automated and largely unsupervised manner. It does not, in the current version, implement any autoindexing algorithms itself but rather takes advantage of the previous work in this field by executing other indexing programs as sub-processes. In the initial version, *DirAx* (Duisenberg, 1992) and *MOSFLM* (Leslie, 2006) can be used. The user can select more than one indexing method, in which case the program will try the later specified methods should the earlier methods fail to yield a result that passes some basic checks.

Indexamajig takes a list of image filenames as its input. For each image in the list it performs peak detection and sends the peaks to the selected auto-indexing programs in the format required by those programs. Alternatively, peak lists generated by previous processing steps and incorporated in the HDF5 files can be used. If indexing is successful, a unit cell is read back from the auto-indexer and, optionally, compared against a reference cell. Cell comparison can be performed *via* a variety of methods, the default being to check all possible linear combinations of the cell basis vectors for correspondence, within a certain tolerance, to the axes of the reference cell. When indexing using *MOSFLM*, the lattice symmetry (if known) may be used to restrict the number of candidate unit cells returned. *DirAx* has no such option and simply returns a list of possible primitive unit cells.

It is further required by the software that the unit-cell vectors form a right-handed basis after the matching process, meaning that Bijvoet pairs are not confused with one another. This could potentially allow the extraction of an anomalous diffraction signal subject to considerations described in the next section.

If the cell is found to match to the sought unit cell, predicted peak positions are calculated for the image and compared with the initial peak positions sent into the auto-indexing program. A basic check for correctness is performed, where the result is rejected if fewer than 10% of the initial peak positions are close to predicted locations, to within a tolerance defined by the program input. A lower success rate might be expected for the smallest nanocrystals, where the peaks in the diffraction pattern arise from the extended tails of the shape transforms.

The method of unit-cell reduction described above is vulnerable to errors in cases where the length of one unit-cell axis is close to a small multiple of the length of another. This situation is analogous to reticular twinning familiar in conventional crystallography. For such cases, the user may opt instead to compare the lattice vectors without combining them in linear combinations, although this would be expected to result in a lower number of successfully indexed patterns. Alternatively, the cell-matching procedure can be entirely disabled, in which case the result from the auto-indexer is used directly, provided the basic check described above is passed.

Once the pattern has been successfully indexed, intensities are integrated from the predicted peak locations, thereby including peaks that were not found by the initial peak search. In the data evaluated so far, the peak shapes in the image were found to vary widely within a single image (Fig. 2), in a way similar to that seen in simulated patterns for small crystals as a result of the extended shape transforms surrounding each reciprocal lattice point. Therefore, *indexamajig* does not attempt two-dimensional profile fitting in the current version and a method of summing pixel counts within a fixed radius is used instead. The radius of integration can be configured in the detector geometry file. The local background surrounding the peak is estimated and subtracted by measuring the intensity in a thin ring surrounding the integration region. Improved methods for background estimation and subtraction and proper calculation of the errors in the integrated intensities, and for automated rejection of

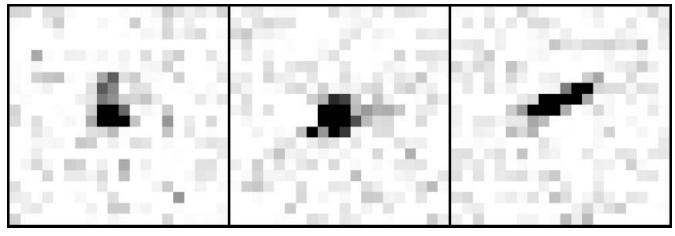


Figure 2
Three peak profiles from a single diffraction pattern from photosystem I, taken from the data described by Chapman *et al.* (2011).

peaks that cannot be satisfactorily integrated, are under development.

The resulting intensities from each image are written to a text file, alongside other information such as the image filename, the reciprocal axis vectors and optionally the peak locations from the initial peak search. The results from all input images are sequentially written to the same text file – which is referred to as a 'stream' – for ease of later handling. The overall flow of diffraction pattern processing by *indexamajig* is shown in Fig. 3.

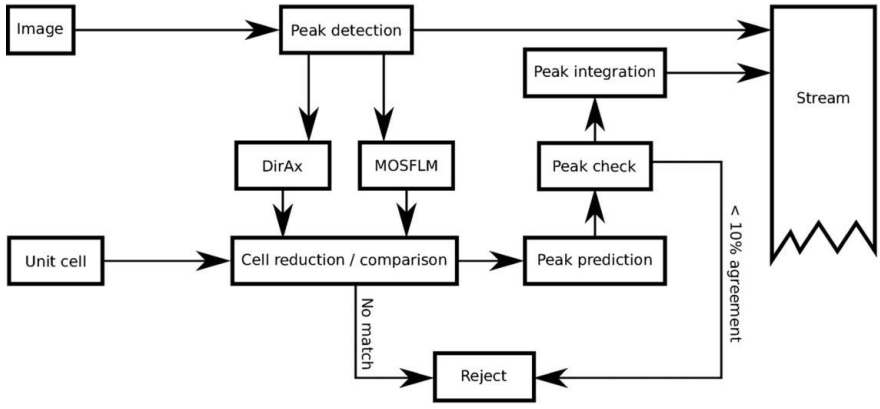


Figure 3
Flow diagram of diffraction pattern processing in *indexamajig*.

The indexing and integration of many thousands of diffraction patterns may take many hours depending on the speed of computer and data retrieval, and *indexamajig* is able to run many indexing jobs in parallel to reduce the time required for the whole data set. *Indexamajig* writes reports of the rate of processing and the 'yield' of the process, defined as the ratio of the number of successfully indexed patterns to the overall number of patterns, to the terminal at intervals of a few seconds.

Once a 'stream' has been compiled for a data set, the results of indexing can be visualized by running a short Perl script, called *check_near_bragg*. This script finds, for each image in sequence, the positions of the peaks predicted by indexing, the coordinates of which are also stored in the stream. The script then runs the image viewer *hdfsee* to show the image and the peak locations (Fig. 4). When the viewer window is closed, the script displays the next image from the stream in the same way. A very similar script called *check_peak_detection* operates in the same way, but displays the results of the initial peak search instead of the predicted peaks.

5. Merging of intensities: *process_hkl*

Merging of individual intensity measurements is performed by the method of Kirian *et al.* (2010), in which the mean of all measurements for each individual reflection is taken. When the number of measurements is large, this procedure constitutes a Monte Carlo integration over the three-dimensional reflection profiles, resulting in values that are proportional to the integrated intensity. A well known feature of Monte Carlo methods is that, in the limit of a sufficiently large quantity of data, all stochastic variables (such as variations in X-ray pulse intensity or crystal size) will be 'integrated out' and become constant factors affecting all intensities equally. For data obtained using a free-electron laser source, further complications arise because of the stochastic nature of the lasing process: the incident beam intensity, wavelength and spectrum are different for each pulse. It is clear that a large number of indexed patterns are necessary for this method to be successful, justifying the highly automated processing of patterns described in the previous section.

The program *process_hkl* performs merging using this method taking into account the symmetry of the structure. It was found to be convenient for the software to neglect information about systematic absences due to glide planes and screw axes. Instead, only the point symmetry of the structure is considered when merging intensities.

The program is also able to perform scaling of the intensities in an attempt to improve the quality of the results. Scaling can be performed by normalizing the intensities according to the mean intensity of the Bragg peaks in each pattern or the overall total intensity in each pattern, or by a two-pass process where the intensities are scaled to most closely fit the values produced by a previous unscaled run of the program. Improved algorithms are under development.

Since each individual diffraction pattern is indexed independently, ambiguities may result if the symmetry of the structure is lower than that of the lattice. These are precisely the same conditions under which the structure may exhibit twinning by merohedry, and the effect of such an ambiguity when combining results from many patterns will be that the data appear to be perfectly twinned. Unfortunately, all attempts so far to resolve such ambiguities by correlating the intensities in the patterns have failed, perhaps owing to the partialities associated with the individual reflection measurements. As a result, the symmetry according to which the intensities must be merged is dictated by the asymmetric unit that the indexing

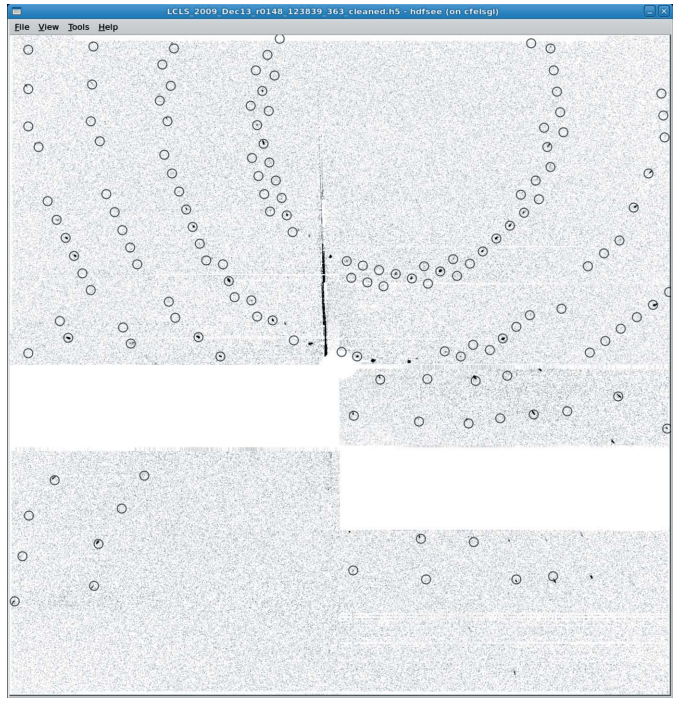


Figure 4
Screenshot of the image viewer *hdfsee* displaying an image from the data described by Chapman *et al.* (2011), with predicted peak locations circled.

procedure can recognize unambiguously. In the merohedral case, this asymmetric unit is reduced in size and so the merging symmetry must be increased. A printable table is provided as part of the documentation for *CrystFEL* listing the 230 space groups according to point group, Laue class and holohedry, with the point groups positioned such that the appropriate merging symmetry can be quickly determined for any given 'true' symmetry.

The above considerations apply also in the case of pseudo-merohedry, for example when an orthorhombic structure has two axis lengths almost equal. In this example, if the difference between the two similar lattice parameters is smaller than the tolerance allowed by the cell-reduction procedure described earlier then the holohedral symmetry for a tetragonal lattice (belonging to Laue class $4/mmm$) must be used instead. It was found useful to introduce the concepts of 'source' and 'target' symmetries when describing the merging process. The source symmetry describes the symmetry that the indexing procedure is able to discern, and the target symmetry describes the symmetry of the true structure. Left coset decomposition (Flack, 1987) of one symmetry group with respect to the other provides the required 'twin laws', which specify the symmetry operations of the ambiguities to be resolved. However, in contrast to the determination of conventional twin laws, mirror and inversion operations are not permitted since the crystallographic axes must form a right-handed basis as described earlier. It should be noted further that no 'twin fraction' is required when describing the ambiguities described here, since the relevant ambiguities can always be identified and merging performed according to the higher symmetry. The contributions from each of the 'sides' of the applicable 'twin laws' are therefore always exactly equal.

In the current implementation, which does not have the means to resolve indexing ambiguities, the target symmetry is always equal to the source symmetry and the resulting intensities must appear twinned. If future versions of the software were to incorporate the required algorithms, the target symmetry could be reduced to the true symmetry of the structure. If the true symmetry were unknown, resolution could be attempted into progressively lower and lower symmetries until no resolution could be successfully performed. If there were multiple ambiguities, a partial resolution into a target group of intermediate symmetry could be performed. This might be useful in specialized cases, perhaps where a full resolution of the ambiguity is difficult and it is considered preferable to have 'twinned' data according to one of the ambiguities than a poor resolution of both. If the intensities are to be merged under the assumption that Friedel's law holds, meaning that Friedel pairs of intensities are to be merged with one another, then the target symmetry can simply be specified as the Laue class corresponding to the appropriate point group.

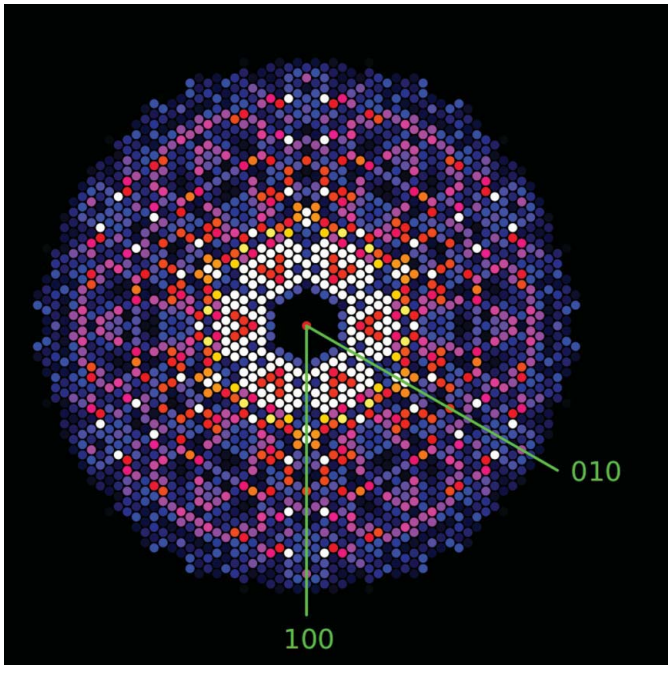


Figure 5
Zone axis structure factors from Chapman *et al.* (2011), plotted using *render_hkl*.

The intensities can be visualized by plotting intensities in flat central sections through reciprocal space (similar to a simulated precession diffraction pattern) using a color scale as shown in Fig. 5. A choice of color scales is available, and the program can also plot the values in three dimensions by creating an input file for the Persistence of Vision ray-tracing program (available from http://www.povray.org/) and invoking it automatically.

A helper script is included which can be invoked once merging has been completed in order to create an MTZ file for import into *CCP4*. This script operates by invoking the *CCP4* program *f2mtz* with a format specification appropriate for *CrystFEL*'s plain text reflection data format.

6. Evaluation of data quality

The evaluation of the data quality is difficult for the type of experiment for which *CrystFEL* has been designed. Since the Monte Carlo merging procedure operates by taking samples from a number of different distributions, it is clear that the individual values to be merged will not share a high degree of similarity. Indeed, it is not desired that they have a high degree of similarity, since the aim is to sample all the underlying distributions as fully as possible, and a full sampling of all the distributions will produce both large and small intensities for any given reflection. As a result, the traditional data quality metrics such as $R_{\rm merge}$ cannot give a meaningful measure of the data quality. A more useful figure can be obtained by splitting the data into two separate interleaved sets, which are merged independently, and then examining the agreement between the two resulting intensity lists. Since the data have been split into two sets, it is expected that the degree of convergence in each subset would be lower and so this method could underestimate the quality of the combined data by a factor of $2^{1/2}$. A suitable figure of merit could therefore be defined as

$$R_{\rm split} = 2^{-1/2}\,\frac{\sum |I_{\rm even} - I_{\rm odd}|}{\frac{1}{2}\sum (I_{\rm even} + I_{\rm odd})}, \qquad (2)$$

where $I_{\rm even}$ represents the intensity of a reflection produced by merging even-numbered patterns, $I_{\rm odd}$ represents the intensity of the equivalent reflection from the odd-numbered patterns and the sum is over all reflections. This method can be performed with *CrystFEL* by using a helper script to split the 'stream' into two, merging the two resulting streams using *process_hkl* and comparing the two merged intensity lists using *compare_hkl*. The program can also calculate the *R* factor as a function of resolution.

A method has been described for estimating the error in the final estimate of the intensity of each reflection in the Monte Carlo method, using

$$\sigma_{hkl} = \left[\sum \left(I_{\rm spot} - \langle I_{hkl}\rangle\right)^2\right]^{1/2} \Big/ N_{hkl}, \qquad (3)$$

where I_{spot} is an individual measurement of the reflection hkl, $\langle I_{hkl} \rangle$ is the mean of all such measurements, N_{hkl} is the number of measurements and the summation is over all measurements of a particular reflection (Chapman *et al.*, 2011, supplementary information). Errors are estimated in this way by *process_hkl* and recorded in the final merged intensity list. By the central limit theorem, the distribution of measured mean intensity values for a given reflection will closely approach a Gaussian provided a sufficiently large number of random samples are taken. However, caution must be used in interpreting the meaning of σ_{hkl} when this condition is not met, in which case the error estimate could over- or underestimate the true error in the intensities.

The program *check_hkl* can calculate statistics, such as the mean $I/\sigma(I)$ or the redundancy and completeness of the data, as a function of resolution.

The overall flow of the indexing, merging and evaluation process is shown in Fig. 6.

7. Creation of 'virtual powder patterns': *powder_plot* and *sum_stack*

Information can be derived from the inspection of serial crystallographic data in 'powder' form, such as when attempting to evaluate the effects of radiation damage on the overall falloff of intensities with resolution. Such analysis can be performed without indexing the individual patterns, for example by summing the patterns themselves (provided the background subtraction is sufficient) or by summing the peaks found by the peak search. *CrystFEL* supports this type of analysis through the programs *sum_stack* and *powder_plot*. The former program, *sum_stack*, performs the usual peak detection on each of its input images and adds pixels within a small circle around each peak to a final image. The latter program, *powder_plot*, creates one-dimensional powder traces from input in many different formats, for example a stream, an individual diffraction pattern or merged intensities. If a stream file is used for input, the user may opt to create the powder plot from the peak locations found by the initial peak search or the integrated intensities after indexing. If the integrated intensities are used, the correct bin to place the intensity in can be calculated by combining the Miller indices either with a provided unit cell or with the unit cell specific to the individual pattern, which can differ from the average unit cell by up to an amount equal to the tolerance of the cell-reduction procedure. A large amount of control can therefore be exercised when performing analysis with virtual powder patterns.

8. Documentation

All programs accept a '`--help`' argument on the command line, which produces a summary of the usage of the program and its various options. Installation instructions detailing the required libraries and other environmental factors are provided, as well as standard 'man' pages describing many of the features in further detail. The symmetry table described earlier is also included, and may also be downloaded separately from the same web site as the software itself. Low-level documentation of the internals of the software, detailing the interfaces available to programs forming part of the suite (such as functions for handling reflection data), is also included.

9. Future work

CrystFEL is a young software project created for use in a very new and rapidly developing field, and so new features and improvements to the analysis pipeline are currently under active development. One continuous development is to improve the yield of the indexing process while filtering out inaccurate indexing results. This will be achieved by implementing new indexing algorithms which operate by searching for *known* reciprocal lattice vector lengths instead of by providing an *ab initio* unit cell for comparison. In addition, interfaces to other auto-indexing programs such as *XDS* (Kabsch, 2010), *DENZO* (Otwinowski & Minor, 1997) and *LABELIT* (Sauter *et al.*, 2004) will be added. The software will also be developed to allow the processing of diffraction patterns corresponding to more than one crystal (Vaughan *et al.*, 2004), which would allow the concentration of crystals in the suspension injected into the path of the X-ray beam to be increased beyond the level at which, on average, one crystal contributes to each pattern.

Improved methods for scaling the intensities will be the subject of much future work. The current method makes no attempt to model the diffraction process, and instead simply averages a large amount of data to obtain an accurate result. By introducing such a model, even a crude one, and fitting parameters such as the incident intensity, the crystal orientation and the wavelength of the radiation, it should be possible to arrive at more accurate estimates of the underlying structure factors, perhaps with a smaller number of patterns. Such a method would be similar to the methods employed in conventional X-ray crystallography, such as post-refinement (Rossmann & van Beek, 1999), but with some complications potentially arising from indexing ambiguities. Initial work in this direction, although not currently usable, is already included in the current version of *CrystFEL* as the program *partialator*.

These improvements will act to reduce the number of diffraction patterns required to obtain an accurate set of intensity data, which is of great importance given the scarcity of experimental time at hard-X-ray free-electron laser sources.

10. Software availability

To ensure the maximum possible use and understanding of the software, *CrystFEL* is available in source code form under version 3 or later of the GNU General Public License (GPL). It can be

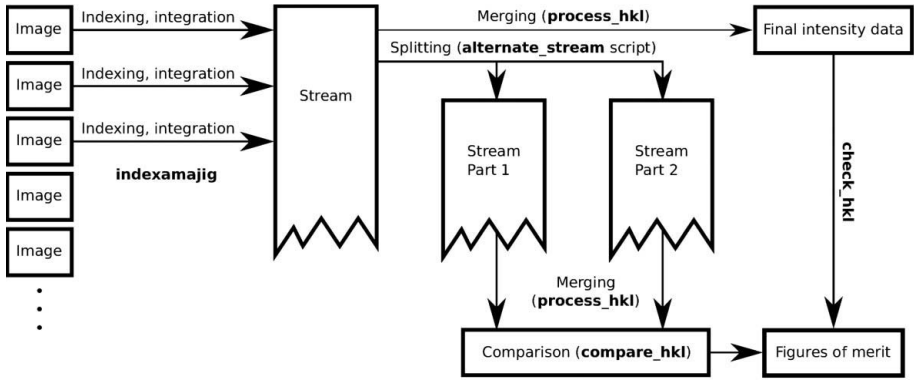


Figure 6
The overall pipeline of the indexing, merging and evaluation workflow. Individual images are indexed and integrated in parallel, and the resulting Bragg intensities written to a single long file known as a 'stream'. The contents of the stream can be merged to produce the final intensity data, or the stream can be split into two smaller streams which can be merged individually. Comparison of the two individually merged results produces figures of merit that can be used to evaluate the data quality. Different figures of merit can be produced from the final intensities themselves.

downloaded from http://www.cfel.de/. Contributions in the form of bug reports, comments and source code patches are actively invited.

Mark Hunter, Francesco Stellato, Linda Johansson, David Arnlund, Nadia Zatsepin and Lorenzo Galli tested the software and provided bug reports as well as feedback on the manuscript. John Spence and Ilme Schlichting also read the manuscript and made corrections and improvements.

References

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N. & Bourne, P. E. (2000). *Nucleic Acids Res.* **28**, 235–242.
Chapman, H. N. *et al.* (2011). *Nature* (*London*), **470**, 73–77.
Duisenberg, A. J. M. (1992). *J. Appl. Cryst.* **25**, 92–96.
Emma, P. *et al.* (2010). *Nat. Photonics*, **4**, 641–647.
Flack, H. D. (1987). *Acta Cryst.* A**43**, 564–568.
Kabsch, W. (2010). *Acta Cryst.* D**66**, 125–132.
Kirian, R. A., Wang, X., Weierstall, U., Schmidt, K. E., Spence, J. C., Hunter, M., Fromme, P., White, T., Chapman, H. N. & Holton, J. (2010). *Opt. Express*, **18**, 5713–5723.
Leslie, A. G. W. (2006). *Acta Cryst.* D**62**, 48–57.
Otwinowski, Z. & Minor, W. (1997). *Methods in Enzymology*, edited by C. W. Carter & R. M. Sweet, Vol. 276, *Macromolecular Crystallography*, part A, pp. 307–326. New York: Academic Press.
Rossmann, M. G. & van Beek, C. G. (1999). *Acta Cryst.* D**55**, 1631–1640.
Sauter, N. K., Grosse-Kunstleve, R. W. & Adams, P. D. (2004). *J. Appl. Cryst.* **37**, 399–409.
Strüder, L. *et al.* (2010). *Nucl. Instrum. Methods Phys. Res. Sect. A*, **614**, 483–496.
Vaughan, G. B. M., Schmidt, S. & Poulsen, H. F. (2004). *Z. Kristallogr.* **219**, 813–825.
Winn, M. D. *et al.* (2011). *Acta Cryst.* D**67**, 235–242.