

Web-Ice: Integrated Data Collection and Analysis for Macromolecular Crystallography

A. González^{a,*}, P. Moorhead^a, S. E. McPhillips^a, J. Song^a,
K. Sharp^a, J. R. Taylor^{b,c}, P. D. Adams^{b,c}, N. K. Sauter^c and S. M. Soltis^a

^aStanford Synchrotron Radiation Laboratory, 2575 Sand Hill Road, MS99, Menlo Park, CA 94025 U.S.A., ^bBerkeley Center for Structural Biology and ^cPhysical Biosciences Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720 U.S.A.

* e-mail: ana@slac.stanford.edu

Synopsis Web-Ice is a scalable, extendable and portable software application for rapid on-line diffraction image analysis, autoindexing and strategy calculation. The Web-Ice architecture, software components and functionality both as a stand-alone application and as part of a beamline control system are described.

Abstract

New software tools are introduced to facilitate diffraction experiments involving large numbers of crystals. While existing programs have long provided a framework for lattice indexing, Bragg spot integration, and symmetry determination, these initial data processing steps often require significant manual effort. This limits the timely availability of

data analysis needed for high-throughput procedures, including the selection of the best crystals from a large sample pool, and the calculation of optimal data collection parameters to assure complete spot coverage with minimal radiation damage. To make these protocols more efficient, we developed a network of software applications and application servers, collectively known as *Web-Ice*. When the package is installed at a crystallography beamline, a programming interface allows the beamline control software (*e.g.*, *Blu-Ice* / *DCSS*) to trigger data analysis automatically. Results are organized based on a list of samples that the user provides, and are examined within a Web page, accessible both locally at the beamline or remotely. Optional programming interfaces permit the user to control data acquisition through the Web browser. The system as a whole is implemented to support multiple users and multiple processors, and can be expanded to provide additional scientific functionality. *Web-Ice* has a distributed architecture consisting of several stand-alone software components working together via a well defined interface. Other synchrotrons or institutions may integrate selected components or the whole of *Web-Ice* with their own data acquisition software. Updated information about current developments may be obtained at <http://smb.slac.stanford.edu/research/developments/webice>.

1 Introduction

Recent crystallography research at synchrotron sources has taken an increasingly high-throughput approach. Large institutional projects and many individual labs are determining structures for multiple protein targets or molecular complexes, examining large numbers of crystals, and deriving the final results within an expedited time frame. For example, NIH-funded structural genomics initiatives in the U.S. determined 1100 new macromolecular structures during the first five years of funding (Stevens *et al.*, 2001) and crystallographic screening of compound libraries (Nienaber *et al.*, 2000) is now a widely used technique for drug lead discovery in the pharma-

ceutical industry. Recent large-scale structural studies report the examination of an average of 40 crystals per protein target in order to identify growth and cryocooling conditions that give the best diffraction (Page *et al.*, 2005). Individual studies of large molecular complexes also face similar challenges, with recent virus and polymerase projects each requiring the evaluation of 100 cryocooled crystals to find a single suitable sample (Kaufmann *et al.*, 2004; Westover *et al.*, 2004). Indeed, due to radiation damage effects, many high profile structures including those of ribosomal subunits (Wimberly *et al.*, 2000) and viruses (Verdaguer *et al.*, 2000; Grimes *et al.*, 1998) could only be determined by merging partial datasets from between 10 and 1000 crystals. Since future experiments will target even more challenging structures, it is important to make the data collection and processing steps at synchrotron beamline facilities as efficient as possible.

Many aspects of experimental efficiency have been addressed by advances in beamline hardware. Studies on very small crystalline samples have been enabled by a combining low-divergence microfocus sources with fast read-out detectors. Developments in automation have made it possible to consecutively study over 100 samples using robotic sample transfer (Cohen *et al.*, 2002; Snell *et al.*, 2004), with all mechanical aspects of the system being controlled through a central software interface (*e.g.*, McPhillips *et al.*, 2002). Developments such as these focus largely on the improvement of data acquisition. But while it is clearly beneficial to increase the data acquisition rate, overall productivity also relies on the ability to concurrently analyze the data, making results immediately available in order to influence subsequent experimental decisions (Fig. 1). To choose the best sample from among a group of crystals, for example, an automated system to mutually rank the diffraction patterns would remove the manual burden of comparing potentially hundreds of images (Criswell *et al.*, 2004). Likewise, an optimized data collection strategy (Dauter, 1999) for measuring a complete set of structure factors could be derived automatically from the analysis of a few initial images at the beginning of data collection.

Providing a computational framework for such calculations has been the subject of discussion for several years, leading to the development of programs like *DNA* (Leslie *et al.*, 2002) and *XIA-DPA* (Winter, 2005) that seek to improve the link between data collection and analysis. Particular software tools useful for automating individual data processing steps have been developed both by us (*e.g.* Zhang *et al.*, 2006; Sauter *et al.*, 2004) and others (Leslie, 1999; Popov & Bourenkov, 2003; Murray *et al.*, 2004; Murray *et al.*, 2005). The work presented here utilizes these established programs, but does so within a new computational environment designed to address problems of scale. Specifically, high-throughput automatic data analysis can only succeed if the system can process data at a rapid turnover rate (matched to the data acquisition rate), and accommodate data sets collected from multiple crystals, grouped together into multiple projects. Within the synchrotron facility, it is important to handle the condition that data belong to different users, and are acquired at multiple beamline endstations. Another aim is to develop a system that will be extendable in the future, as additional experimental protocols become amenable to automation. Both of these design considerations (scalability and extendability) are embodied within our new software system, *Web-Ice*.

The work described here also deals with the issue of portability. *Web-Ice* was initially prototyped at the Stanford Synchrotron Radiation Laboratory (SSRL), and subsequently adapted for the Berkeley Center for Structural Biology (BCSB), one of several macromolecular crystallography beamline groups at the Advanced Light Source. Although the two facilities use separate software systems to control beamline motors and govern data acquisition, we realized that both groups share common software needs for downstream data processing. This led us to develop programming interfaces so that both beamline control systems can communicate with *Web-Ice*. A single software development effort therefore allows us to provide data processing and remote access for users at both facilities. The package enables users to view, analyze, and store results before and after their primary beamtime has ended, thus simplifying the planning of future experiments. In certain situations, data acquisition can be controlled through

Web-Ice. However, in contrast to the underlying beamline control systems — *Blu-Ice* / *DCSS* at SSRL (McPhillips *et al.*, 2002) and *BOS* at BCSB — the graphical interface for *Web-Ice* provides a higher-level view of the experiment. The program presents fewer mechanical details related to data acquisition, but more details related to the crystallographic results. It is therefore an important resource for the larger user community that is concerned with solving new biological problems, rather than focusing on the development of crystallographic techniques.

2 Software design

Web-Ice is an application server, built upon and providing access to a set of existing crystallographic software packages that perform core scientific functions (Table 1). In conventional use, these programs are executed by running shell scripts at the Unix command line, producing either screen or file output. In the case of *Web-Ice*, the necessary shell scripts are automatically generated and executed in the background by a child Unix shell, and results are parsed to provide graphical or tabular displays.

The *Web-Ice* architecture has been designed to loosely couple the beamline control software (*e.g.*, *Blu-Ice/DCS* at SSRL) to the data processing suite. Even if there is an unanticipated fatal software bug related to data analysis, the data collection and other beamline operations are able to continue uninterrupted; this is achieved by the modular design shown in Figs. 1 and 2, where data acquisition and processing are controlled by separate entities. Simple signaling protocols and a shared file system allow the two systems to work together. An added benefit of this modular design is that *Web-Ice* can be adapted to work with beamline control systems in use at other facilities.

Another consideration is that data processing-related activities often commence with the

uploading of a sample list before the synchrotron trip begins (Fig. 1), and can last until well after the trip ends. It is therefore important to make the interface remotely accessible, without specific time limits like in the case of *Blu-Ice*, which is typically accessible only by the current beamline user. Our solution is based on the World Wide Web; the front-end *Web-Ice* interface is written as a Java servlet hosted by an Apache Tomcat Web server (The Apache Software Foundation, 2006). Other components of the system (Fig. 2) are implemented using a variety of approaches, which are described below.

2.1 *Web-Ice* as an interface to the normal Unix file system

With one exception (§2.6), a central database is not used to track calculations within *Web-Ice*. Instead, information is stored in the form of regular files owned by the end user. By default, a separate directory is created for this purpose, `$WEBICE-HOME/webice`, where `$WEBICE-HOME` is a configurable directory in the user's disk area. All command scripts, log files, and intermediate results are stored in this area. As a consequence, expert users retain the ability to directly inspect calculations at the Unix command line, and files can be edited and re-run if desired. Users can also manage disk space and transfer key results back to their own labs using secure copy protocol (`scp`).

2.2 Authentication for multiple users across multiple applications: the Authentication Server

Fig. 2 shows that the overall computational environment involves several components, which are typically distributed among several computer hosts to balance the computational load. It is important for the various components to be able to work together without requiring end

users to type in their user names and passwords repeatedly. The process of authentication is therefore centralized within a single component, the Authentication Server. This Web application performs three duties: authentication of users with their Unix passwords, issuance of active tickets that are valid for a period of time, and verification that a particular ticket has not expired (Fig. 3). Within this scheme, when the user logs in to a first application, it obtains an active ticket on the user's behalf that can be passed to other applications in lieu of a second login. For example, a user may ask for the acquisition of new diffraction data through the *Web-Ice* front end interface. In this case, the *Web-Ice* front end will pass the request, along with the authenticated ticket, to the beamline control system. The beamline control system then validates the ticket with the Authentication Server before granting access to data collection.

The implementation of a separate Authentication Server allows access permissions to be defined in a more flexible way than is achievable by simple Unix authentication. Various beam line functions can be restricted to users who have active beam time. Functions that require active beam status at SSRL are the beamline video viewer, the determination of optimal data collection parameters for the assigned beamline and the ability to initiate data collection or export a strategy to the beamline control software. Users who are not currently collecting data can perform a different set of "off-line" functions (such as autoindexing and analysis of pre-collected images) at any time. At SSRL, customized beamline permissions are checked against a *MySQL* database, while passwords are checked against the Unix yellow pages password file. A new module was implemented to check passwords against an LDAP (Lightweight Directory Access Protocol) database, used for Unix login at BCSB. The Authentication Server can be configured to use either the *MySQL* or LDAP modules. As depicted in Fig. 2, each facility deploys a single instance of the Authentication Server, so as to maintain a globally consistent list of active access tickets for all applications participating in the *Web-Ice* system.

2.3 Running software on behalf of the user: the Impersonation Daemon

The assignment of a separate Unix account to each synchrotron user group is a convenient mechanism for organizing beamline operations and securing users' data. Each user group can access its own data at any time via the Unix login shell, and utilize beamline computers and disk storage for data processing jobs, yet at the same time restrict access to the data by setting Unix read/write flags. *Web-Ice* is intended to be an extension of the pre-existing login interface, whereby simple shell scripts are automatically generated and executed to run third-party software programs. As such, it is important to guarantee that *Web-Ice* respects the same file permissions that have been set under the Unix login shell. Since the Apache Tomcat server is run under a system-wide account, implementing this access control poses a special burden, which is solved by using the specialized component labeled "Impersonation Daemon" in Figs. 2 and 4.

Web-Ice relies on the following framework to guarantee fidelity of the Unix file permission flags while executing programs on behalf of multiple simultaneous users. Running under the Unix root account, the Impersonation Daemon listens for resource requests on a predefined TCP port. Any application with a valid ticket from the Authentication Server can send a command (along with the Unix account name and the ticket itself) to the Impersonation Daemon using an HTTP protocol. The daemon validates the account name and ticket against the Authentication Server and then spawns a child process. Ownership of the child process is switched to user's account, and the requested command is executed. The daemon supports a set of predefined commands similar to Unix commands for performing various tasks on the file system, for example, reading and writing files, listing directories, and checking file permissions. Fig. 4 illustrates the role of the Impersonation Daemon during automated sample screening.

The Impersonation Daemon is implemented as a separate C++ program running within the Unix inetd facility. This architecture accomplishes two additional aims. First, it provides an entry point for load-balanced jobs. *Web-Ice* servlets running within the Apache Tomcat server (Fig. 2) can delegate processing jobs to any computer host that is running an Impersonation Daemon. Secondly, it permits the entire Apache Tomcat package, including the *Web-Ice* front end and Authentication servlets, to run under an unprivileged user account, thus closing a potential security hole.

2.4 Graphical display of diffraction images: the Diffraction Image Server

A previously described Web-based viewer for the display of raw diffraction images (Chiu *et al.*, 2002) has been ported to the *Web-Ice* front end. A key design constraint for this feature is the large file size of the raw image (up to 75 MB), which prohibits the rapid transfer of complete images to the remote Web browser. The Diffraction Image Server run locally at the synchrotron facility (Fig. 2) addresses this problem by creating a compressed snapshot of the raw image, or a portion thereof, which is sent back to the viewer in response to a defined HTTP request. The snapshot is used only for visual inspection, not for data processing. The user can pan through the image, change the zoom level, or adjust the grayscale with rapid turnaround times of less than 0.5 s. At SSRL, the Diffraction Image Server also provides image snapshots to the data collection GUI, *Blu-Ice*.

The Server is implemented as a C++ program running inside the root account. Prior to responding to an HTTP request, a call is made to the Impersonation Daemon to verify the user's read permission for the file in question (see Fig. 4). Several images (typically 10) are cached in dynamic memory so that multiple clients (started by the same or different users) can

simultaneously receive data from individual Server threads. The Server can be configured to return image snapshots using either JPEG or PNG compression. The former makes the most efficient use of the available bandwidth, while the latter produces slightly clearer snapshots since the compression algorithm is non-lossy.

To facilitate visual interpretation, data processing results are shown in the snapshot view. The first time an image is viewed, it is automatically processed with *DISTL*. A color overlay is produced to highlight candidate Bragg spots, ice ring artifacts, and estimated resolution limits (Fig. 5a). A corresponding overlay is created to show the calculated *vs.* observed lattice pattern from *LABELIT* autoindexing (Fig. 5b).

The color displays are implemented in a way that minimizes the impact on processing time and file system storage. For each raw image, the overlays from *DISTL* and *LABELIT* are calculated only once. A geometric description of the color features is encoded in a scalable vector graphics format, and cached in a file in the user's `webice` directory. The Diffraction Image Server produces a viewable snapshot by combining the vector graphics description with the separately-stored raw image. This allows the picture to be rendered as many times as desired at different zoom levels and pan positions (Fig. 5, insets). The vector graphics file requires only 10 to 100 KB for file storage, a negligible amount compared to the file size of the raw image.

2.5 Modular software architecture for data analysis: the Crystal Analysis Server

As illustrated in Fig. 2, the *Web-Ice* front end acts as a Web-based gateway for automatic crystal analysis. For users sitting at the end station controls, it is equally important for the data analysis software to be accessible through the beamline control GUI. To satisfy both types

of access needs, the main data analysis functions are collected into a separate application, the Crystal Analysis Server. This module is structured as a Web application running within Apache Tomcat. It can be accessed by any client through an HTTP protocol, thus providing autoindexing and data integration to either the *Web-Ice* front end or to the beamline control application (as shown in Fig. 4). This modular implementation assures that any unexpected crystal analysis failure will have minimal impact on data collection and beamline operations. Jobs submitted to the Crystal Analysis Server are placed in queues and executed consecutively.

2.6 Information about screened samples: the Sample Information Server

Web-Ice provides the ability to group similar crystal samples together, allowing a direct comparison of their diffraction properties by inspecting either a results table or the original images. Beamline users can thus harness *Web-Ice* to help select the most promising samples for study. To afford these sample comparisons *Web-Ice* implements a flat-file data structure known as the “Sample Information List” (SIL), which is managed by a separate Tomcat Web application, the Sample Information Server (Fig. 2). We have implemented two different approaches to construct the SIL. At SSRL, users can prepare a *Microsoft Excel* spreadsheet prior to their scheduled beam time, listing all of their samples, along with sample locations within the cryogenic storage puck (Cohen *et al.*, 2002). This spreadsheet is uploaded to the Sample Information Server, where it is converted to an Extensible Markup Language (XML) format directly used by other *Web-Ice* applications. For use at BCSB, we are extending the Sample Information Server’s application programming interface to allow the beamline control system to directly populate the SIL list. In either case, once the SIL is constructed and images are acquired, the data are analyzed with the Crystal Analysis server (§2.5). The processing results are then sent to the Sample Information Server using an HTTP protocol, updating the SIL data in the

XML format. Finally, the *Web-Ice* front end utilizes the SIL data to construct Web pages with tabular views. Extensible Stylesheet Language (XSL) stylesheets permit the view to be customized; SSRL and BCSB presently use similar but separate style definitions to translate the XML format into the final Web page. The function of the Sample Information Server is illustrated in Fig. 4.

3 Interaction with the beamline control software

Web-Ice can function in three different modes depending on the level of interaction established with the beamline controls. We discuss them here in order of increasing function. As will be clear, varying degrees of automation are appropriate for addressing specific problems.

3.1 Passive (standalone) analysis

In the most basic implementation, *Web-Ice* can be set up to be completely independent of the beamline controls, sharing just the disk storage space so as to have read only access to the raw diffraction data. This type of operation is extremely useful for beamline staff during the development of *Web-Ice*, as well as for end users who wish to reprocess data remotely. In this mode, the user logs in through a Web page generated by the *Web-Ice* front end. In response to user commands, the front end obtains any necessary processing parameters (*e.g.*, wavelength, sample-to-detector distance, oscillation angle, *etc.*) from the image header and initiates the relevant data analysis scripts. Status of the processing jobs is monitored by the *Web-Ice* front end, by periodically reading the contents of a file that is written out by the processing script to the user's `webice` directory. Once the job has been started, the user can log out of the *Web-Ice* front end (or close the browser window) without interrupting the script execution. Another

Web-Ice session can be initiated at any time to monitor the job and inspect the results.

3.2 Active analysis

By making *Web-Ice* aware of the current state of a beamline control system, it is possible to automatically trigger data analysis as the images are being collected. If this is done in the context of high throughput sample screening, the beamline user can inspect the results after processing is complete, without the overhead of issuing manual commands to run processing programs. Furthermore, communication between the beamline control system and *Web-Ice* enables the accommodation of varying experimental parameters (such as beamline intensity; motor range and speed) when calculating the optimal data collection strategy. For example, if the optimal oscillation speed is determined to be faster than what the phi motor can achieve, the software calculates the beamline attenuation factor required to collect the images at a slower speed.

Active analysis has been implemented at SSRL via an interface between *Web-Ice* and the control system server *DCSS* using the same protocol employed by the beamline control GUI, *Blu-Ice*. At BCSB, a *Web-Ice* interface for *BOS* is under development.

3.3 Full beamline control mode

At SSRL, *Web-Ice* can initiate data collection by sending signals to the beamline control software. This is useful for acquiring data in accordance with the optimal strategy that *Web-Ice* has calculated. The *Web-Ice* front end establishes a connection to *DCSS*, and issues the appropriate commands. Thereafter, the connection reverts to the “active analysis” modality described above.

The *Web-Ice* front end thus provides a simplified alternative to *Blu-Ice* for conventional experiments amenable to a high degree of automation. In the future, we anticipate that such two-way communication between the beamline control system and *Web-Ice* will make complex experiments more tractable. For example, the study of large unit-cell crystals, acquisition of ultra-high resolution diffraction, and multi-crystal experiments for radiation-sensitive samples can all potentially benefit by responding in real-time to results derived from automated data analysis.

4 Automated high throughput crystal screening at SSRL

A diffraction-quality comparison can be automatically generated from a collection of samples, once a software connection has been established between the beamline control system and the data processing environment. At the SSRL beamlines, there is room to store up to 288 samples in the cryogenic Dewar inside the beamline hutch. To study these samples sequentially, the user initializes a new sample information list (§2.6), identifying each crystal as well as its position within the cryo-storage cassette. Once the samples of interest are highlighted and data acquisition parameters are defined, the entire process unfolds automatically. Under robotic control, each crystal is placed on the goniometer, the nylon loop containing the sample is optically centered on the X-ray beam, two X-ray oscillation photographs are acquired at ϕ settings about 90° apart, two video snapshots are stored from these same ϕ settings, and the sample is returned to the Dewar. Concurrently, the Crystal Analysis Server component of *Web-Ice* is signaled to begin characterizing and autoindexing the diffraction patterns using the programs *DISTL* and *LABELIT*. Manual input is not required at this point, as all of the required information is extracted from the image header by the Diffraction Image Server and from *DCSS*.

Although the above steps can be carried out in a totally automated fashion, the users are offered the possibility to pause the procedure at any time, *e.g.*, to adjust the position of the sample after automated loop centering. This is quickly done using the click-to-center feature of *Blu-Ice*, and it results in a more accurate assessment of the crystal quality in cases where the crystal is much smaller than the loop. Although a very reliable diffraction-based crystal centering method has been developed by Song *et al.* (2007), it is more time-efficient at the screening stage to center the loop, and resort to manual intervention to center the crystal if required.

As the screening experiment progresses (and also when it is finished) the user can view the table of comparative statistics written by the Sample Information Server within the *Web-Ice* Web page, or within the *Blu-Ice* GUI. Useful criteria for selecting the best samples include high spot signal-to-noise, round spot shape, minimal number of ice-ring artifacts, correct Bravais symmetry of the indexed lattice, low crystal mosaicity and high resolution limit. The tabular view within *Web-Ice* can be sorted by any of these criteria (see Fig. 6). Results can also be downloaded in *Excel* spreadsheet format. Importantly, diffraction images can be visually inspected in detail within the *Web-Ice* image viewer, with samples sorted in quality-order as defined by the user. This detailed presentation is very useful for identifying the highest-priority samples for a complete data set acquisition.

5 Strategy calculation

Once all the crystals have been screened and the user has identified the ones to use for data collection based on the initial autoindexing results, optimal data acquisition parameters must be derived for collecting a complete dataset with either minimal goniometer rotation (Dauter, 1999), or with minimal elapsed time or absorbed X-ray dose (Popov & Bourenkov, 2003).

For this derivation, it is advantageous to acquire two test images separated in ϕ angle, giving a more accurate crystal orientation and a more complete picture of the sample's diffraction quality. When the user is enabled at a beamline, *Web-Ice* provides an interface to acquire these test images. Alternatively, existing test images may be used, including those previously acquired using *Blu-Ice*. Optimized acquisition parameters are exported to the beamline control system in the same format as a manually generated data collection run definition.

Crystal characterization and autoindexing are carried out in the same way as for crystal screening. After autoindexing, *MOSFLM* is used to calculate the oscillation angle required to maximize data completeness based on unique reflections or Bijvoet pairs. Because the diffraction symmetry cannot be determined with certainty by autoindexing, angular rotation strategies are calculated for all Patterson groups that are consistent with the data (the additional seconds required for this calculation could be saved in the future by performing parallel jobs for each test group). If the symmetry is known ahead of time, the Patterson group and unit cell can be declared; this is very useful if a pseudo-symmetry exists that autoindexing programs cannot easily detect (*e.g.* in the relatively common case of a monoclinic crystal with a β angle very close to 90°). The per-image oscillation angle is determined by *MOSFLM* based on the results of the spot overlap analysis for the calculated angular range, given the mosaicity value estimated by *LABELIT*.

The user can select the desired experiment type (*e.g.*, monochromatic or anomalous dispersion), which determines several data collection strategy parameters, (*e.g.* the use of inverse beam setting and ϕ angle wedges). For anomalous dispersion experiments a fluorescence scan can be performed to determine the absorption-edge energies with *AUTOCHOOCH* (Evans & Pettifer, 2001). Alternatively, the software can read the energies from a previously generated *AUTOCHOOCH* output file, or use values entered directly by the user.

The optimal sample-to-detector distance is calculated based on the estimated resolution

limit of the images. In the case where the diffraction limit extends beyond the edge of the detector, the software writes out a warning message but does not change the current value for the sample-to-detector distance, as new diffraction images would be necessary to estimate the correct exposure time to collect data to the maximum resolution. The exposure time, based on the estimated value of $I/\sigma(I)$ at the diffraction limit or detector-edge resolution (whichever is more conservative), is calculated with *BEST*.

The beamline intensity and beam size (defined by the collimating slits) are input to *RADDOSE* to provide an estimate of the dose absorbed by the crystal. Typical assumptions are made about the solvent content of the crystal. Currently, heavy atom absorption edges are not taken into account. A more refined dose estimate will be carried out in the future, as features are introduced to support experimental phasing experiments and structure solution.

Fig. 7 shows the data collection summary displayed in *Web-Ice*. Calculation of optimal data collection parameters depends to a more or less large extent on accurate sample centering and the choice of an appropriate beam size; currently this step must be performed manually. Further automation is needed both to aid the user in this task (e.g., in cases where the crystal is not clearly visible) and to realize the final aim of a totally automated experiment. As a first step in this direction, diffraction based centering and beam size calculation (Song *et al.*, 2007) will be implemented in the near future.

6 Current status and future plans

Although developments in beamline hardware have been capable of supporting high-throughput structural biology projects for several years, it has been difficult to obtain software that allows the user to keep up with the rapid rate of data acquisition. Fully automated experiments will

clearly require real-time data analysis (where “real-time” is defined to mean an elapsed time on the order of 10-60 seconds after image readout), so that it can be judged whether the acquired data actually supports the experimental goals, and if not, so that experimental adjustments can be made. Progress has been hindered by the perception that requirements for real-time software must be addressed separately by developers at each synchrotron facility. While acknowledging that facilities differ in their computing environments, we favor the approach of designing software to be as portable as possible, thereby tendering the greatest use from a common software development effort.

It is for this reason that we divide the infrastructure for providing real-time data processing into discrete components (Fig. 2), which can be adopted either separately or together by different facilities. The two facilities described here (SSRL and BCSB), differ with respect to several configuration details. Notably, the two facilities employ separate beamline control systems. In order to harness the crystal analysis and sample information list functions, separate application programming interfaces were developed. Different authentication schemes were accommodated, and varying numbers of application servers were set up to handle the computational load. Finally, the table of beamline properties, needed for the calculation of optimal data collection strategy, is configured separately. However, the basic role of *Web-Ice* is the same at both institutions: it allows existing data processing applications to run in an automated manner that supports an increased experimental load.

Source code for the various *Web-Ice* modules is freely available under a non-restrictive MIT style license. Detailed installation instructions are given in a “Wiki” document posted at the Web site listed in the Abstract. The source code can be compiled on most popular platforms including Linux, Windows, MacOSX, Irix, Tru64, and Solaris. Individual packages for data processing (Table 1) are available from their respective authors.

The work described here offers initial steps toward automatic data processing. At SSRL,

where the framework for integration between beamline control and data analysis software is well developed, about 85% of users benefit from the tools for automated crystal screening. Ultimately, data acquisition will need to be validated by the usefulness of the resulting data set, which can only be determined by propagating the calculations to resultant experimental phases or refined structural models. Therefore not only data integration, but also scaling and phasing need to be part of the fully automated experimental environment. A number of robust software packages permit a high degree of automation with minimal input from the user (*e.g.*, Adams *et al.*, 2002). A selection of these tools will be incorporated into *Web-Ice*, but clearly it will be a challenging task to accelerate this degree of data analysis so that it keeps pace with data collection.

Acknowledgements

Work conducted within the SSRL Structural Molecular Biology Program was supported by the U.S. Department of Energy/Office of Biological and Environmental Research, by the National Institutes of Health/National Institute of General Medical Sciences and by the NIH/National Center for Research Resources, Biomedical Technology Program. Portions of the research were conducted at the Advanced Light Source, a national user facility operated by Lawrence Berkeley National Laboratory on behalf of the DOE/Office of Basic Energy Sciences. The Berkeley Center for Structural Biology was supported in part by the NIH/NIGMS. This work was supported in part by DOE Contract No. DE-AC02-05CH11231, and by NIH/NIGMS funding under grant numbers 1P50GM62412 and 1R01GM77071.

References

Adams, P. D., Grosse-Kunstleve, R. W., Hung, L.-W., Ioerger, T. R., McCoy, A. J., Moriarty, N. W., Read, R. J., Sacchettini, J. C., Sauter, N. K. & Terwilliger, T. C. (2002). *Acta Cryst.* **D58**, 1948-1954.

Bushnell, D. A., Westover, K. D., Davis, R. E. & Kornberg, R. D. (2004) *Science* **303**, 983-988.

Chiu, H. J., McPhillips, T., McPhillips, S., Sharp, K., Eriksson, T., Sauter, N., Soltis, M. & Kuhn, P. (2002) In *Proceedings of the Networked Learning in a Global Environment Challenges and Solutions for Virtual Education World Congress* May 1 - 4, 2002 (Berlin, Germany)

Cohen, A. E., Ellis, P. J., Miller, M. D., Deacon, A. M. & Phizackerley, R. P. (2002). *J. Appl. Cryst.* **35**, 720-726.

Criswell, A. R., Bolotovskiy, R., Niemeyer, T., Athay, R. & Pflugrath, J. W. (2004). *Acta Cryst.* **A60**, s112.

Dauter, Z. (1999) *Acta Cryst.* **D55**, 1703-1717.

Evans G., Pettifer R. F. (2001). *J. Appl. Cryst.* **34**, 82-86.

Ferrer, J.-L. (2001). *Acta Cryst.* **D57**, 1752-1753.

Grimes, J. M., Burroughs, J. N., Gouet, P., Diprose, J. M., Malby, R., Ziéntara, S., Peter P. C. Mertens, P. P. C. & Stuart, D.I. (1998). *Nature* **395**, 470-478.

Holton, J. & Alber, T. (2004). *Proc. Natl. Acad. Sci. USA*, **101**, 1537-1542.

Kaufmann, B., Simpson, A. A. & Rossmann, M. G. (2004). *Proc. Natl. Acad. Sci. USA* **101**, 11628-11633.

Leslie, A. G. W. (1999) *Acta Cryst.* **D55**, 1696-1702.

Leslie, A. G. W., Powell, H. R., Winter, G., Svensson, O., Spruce, D., McSweeney, S., Love, D., Kinder, S., Duke, E. & Nave, C. (2002). *Acta Cryst.* **D58**, 1924-1928.

McPhillips, T. M., McPhillips, S. E., Chiu, H.-J., Cohen, A. E., Deacon, A. M., Ellis, P. J., Garman, E., Gonzalez, A., Sauter, N. K., Phizackerley, R. P., Soltis, S. M. & Kuhn, P. (2002). *J. Synchrotron Rad.* **9**, 401-406.

Murray, J. W., Garman, E. F. & Ravelli, R. B. G. (2004) *J. Appl. Cryst.* **37**, 513-522.

Murray, J. W., Rudiño-Piñera, E., Owen, R. L., Grininger, M., Ravelli, R. B. G. & Garman, E. F. (2005) *J. Synchrotron Rad.* **12**, 268-275.

Nienaber, V. L., Richardson, P. L., Klighofer, V., Bouska, J. J., Giranda, V. L & Greer, J. (2000) *Nature Biotechnology* **18**, 1105-1108.

Page, R., Deacon, A. M., Lesley, S. A. & Stevens, R. C. (2005) *J. Structural and Functional Genomics* **6**, 209-217.

Popov, A. & Bourenkov, G. (2003) *Acta Cryst.* **D59**, 1145-1153.

Roth, M., Carpentier, P., Kaikati, O., Joly, J., Charraut, P., Pirocchi, M., Kahn, R., Fanchon, E., Jacquamet, L., Borel, F., Bertoni, A., Israel-Gouy, P. & Ferrer, J.-L. (2002). *Acta Cryst.* **D58**, 805-814.

Sauter, N. K., Grosse-Kunstleve, R. W. & Adams, P. D. (2004) *J. Appl. Cryst.* **37**, 399-409.

Sauter, N. K., Grosse-Kunstleve, R. W. & Adams, P. D. (2006) *J. Appl. Cryst.* **39**, 158-168.

Snell, G., Cork, C., Nordmeyer, R., Cornell, E., Meigs, G., Yegian, D., Jaklevic, J., Jin, J., Stevens, R. C. & Earnest, T. (2004) *Structure*, **12**, 537-545.

Skarzynski, T. & Thorpe, J. (2006) *Acta Cryst.* **D62**, 102-107.

Skinner, J.M. & Sweet, R. M. (1998). *Acta Cryst.* **D54**, 718-725.

Song, J., Mathew, D., Jacob, S. A., Corbett, L., Moorhead, P. and Soltis, S. M. (2007) *J. Synchrotron Rad.*, **14**, 191-195.

Stevens, R. C., Yokoyama, S. & Wilson, I. A. (2001). *Science*, **294**, 89-92.

Verdaguer, N., Blaas, D. & Fita, I. (2000) *J. Mol. Biol.* **300**, 1179-1194.

Westover, K. D., Bushnell, D. A. & Kornberg, R. D. (2004) *Science* **303**, 1014-1016.

Wimberly, B. T., Brodersen, D. E., Clemons, W. M., Jr., Morgan-Warren, R. J., Carter, A. P., Vornheim, C., Hartschk, T. & Ramakrishnan, V. (2000) *Nature* **407**, 327-339.

Winter, G. (2005) *CCP4 Newsletter on Protein Crystallography* No. 43, 18-25.

Zhang, Z., Sauter, N. K., van den Bedem, H., Snell, G. & Deacon, A. M. (2006) *J. Appl. Cryst.* **39**, 112-119.

Tables

Table 1: Programs used by *Web-Ice* for data analysis.

Program	Function	Reference
<i>DISTL</i>	Preliminary image characterization	Zhang <i>et al.</i> , 2006
<i>LABELIT</i>	Autoindexing	Sauter <i>et al.</i> , 2004
<i>MOSFLM</i>	Bragg spot integration	Leslie, 1999
<i>BEST</i>	Optimization of acquisition parameters	Popov & Bourenkov, 2003
<i>RADDOSE</i>	Analysis of radiation damage	Murray <i>et al.</i> , 2004; 2005
<i>AUTOCHOOCH</i>	Determination of absorption edge	Evans & Pettifer, 2001

Figures

Figure 1: Concurrent data collection and analysis for high-throughput experiments. It is the eventual goal to require user action only at the planning stage. As currently implemented at SSRL, the user's overview is also required for reviewing the sample selection and data collection strategy. However, these steps could potentially be automated in a large number of cases.

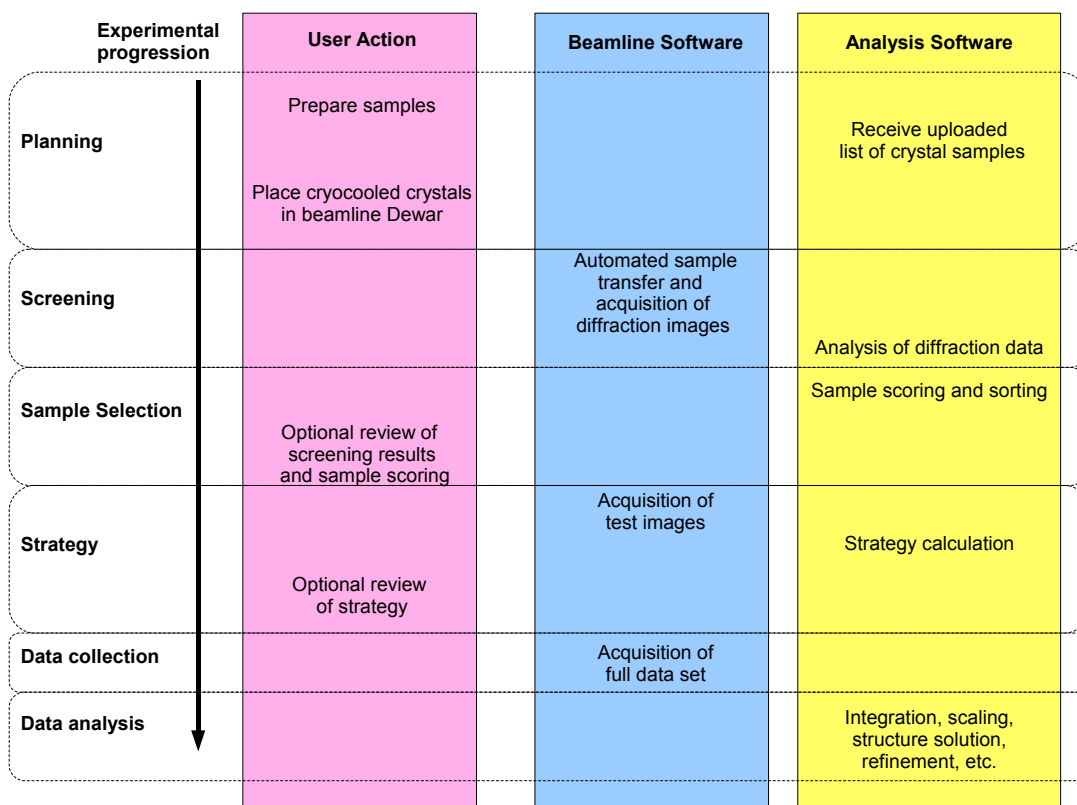


Figure 2: Modular organization of data acquisition and processing software at SSRL. Applications dedicated to beamline control and data acquisition (*Blu-Ice* and *DCS*) are represented by ellipses. Diamonds denote applications primarily used for data analysis: The Crystal Analysis Server receives commands from the *Web-Ice* front end or *DCS* and generates data analysis commands that are executed by crystallographic software (see 1) via the Impersonation Daemon. The rectangles indicate services and applications such as user authentication (Authentication Server), sample information and results storage (Sample Information Server, Database and file system), and data retrieval (Diffraction Image Server) that are shared by both data collection and analysis software.

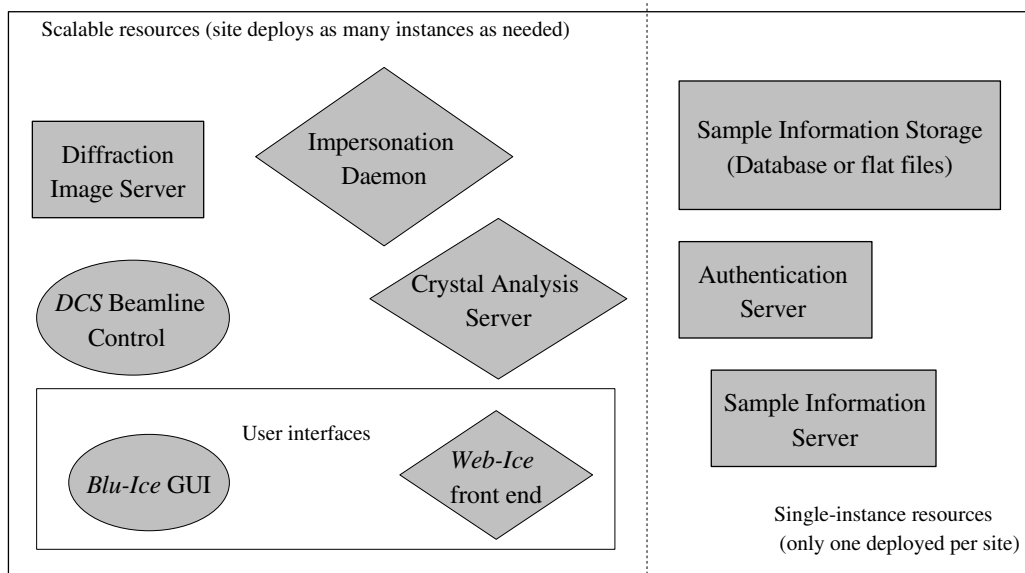


Figure 3: User authentication. The user logs in to an application (Application 1) with a user ID and password. The application encrypts and sends this information to the Authentication server, which compares it with the Unix password file. The Authentication Server also verifies whether the user is authorized to use any beamlines; at SSRL this information is stored in a MySQL database. The Authentication Server issues a ticket identified by a unique session ID (a string of 32 hexadecimal characters) and returns it to the application together with relevant user information, such as beamline access permissions. The ticket can also be passed along to another application (Application 2), which allows the user to run the new application without having to re-login. Application 2 validates the ticket with the Authentication Server before allowing the user to proceed.

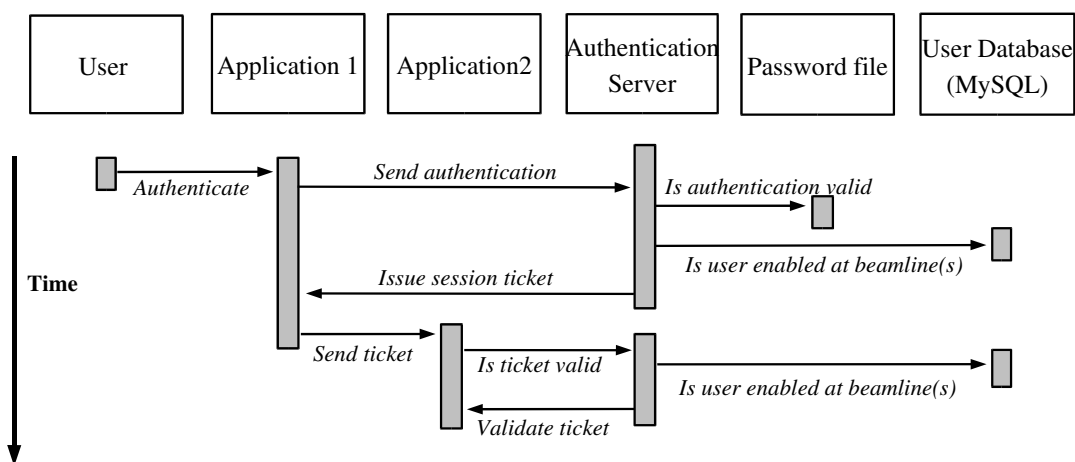


Figure 4: Integration of data acquisition and data analysis with *Web-Ice* components: a time flow view. All the interactions between software applications are in the form of HTTP requests; each request establishes a socket connection which remains open until the application responds or the connection times out. 1) The Crystal Analysis Server receives a command to analyze image from *DCSS* and obtains the information to generate the analysis scripts from the image header via the Image Server. 2) The command to run the scripts and session ID (issued to the data acquisition software upon user login as described in Fig. 3) is sent to the Impersonation Daemon. 3) The Crystal Analysis Server continuously monitors the progress of the task by reading the contents of a control file written by the scripts, and reads the image analysis output files as they are generated through this application. 4) A summary of the results is sent to the Sample Information Server, which appends the results to the Sample Information List and 5) also makes the list available to the User Interface (either *Blu-Ice* or the *Web-Ice* front end).

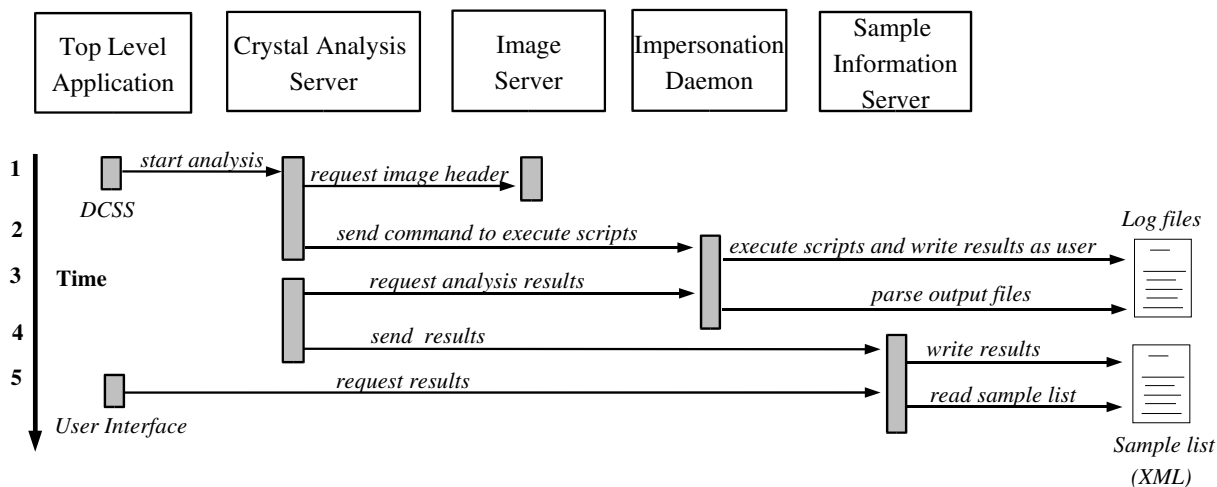


Figure 5: Diffraction pattern detail produced by the Image Server, with color highlighting rendered by the software package ImageMagick++. *a)* Best-fit model ellipses for candidate Bragg spots, with pixel maxima indicated by white dots. Yellow ellipses are from an initial analysis of all potential maxima, while green ellipses survive outlier rejection based on resolution cutoff, spot intensity, spot eccentricity, spot skewness, and possible overlap. Also shown are ice ring artifacts (orange) and the initial estimate of limiting resolution (green). *b)* Following autoindexing with *LABELIT*, the calculated lattice pattern is shown (full reflections: blue; partial reflections: yellow), superimposed on top of dots showing the centers-of-mass of observed spots (yellow and green).

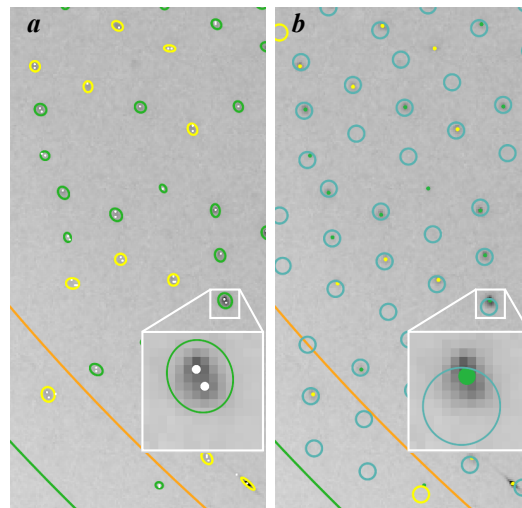


Figure 6: *Web-Ice* interface at SSRL displaying autoindexing results. So that samples suitable for data collection can be identified quickly, the results can be sorted by any of the criteria whose column headers are underlined. Links are provided to the diffraction patterns recorded for each sample, as well as to video snapshots of the mounted crystals, and the detailed analysis results. A summary list is also available in *Blu-Ice*.









Spreadsheet ID: 6320												
		Update	Edit Crystal	Analyze Crystal	E2	View Strategy						
<u>Port</u>	<u>CrystalID</u>	<u>Protein</u>	<u>Images</u>	<u>IceRings</u>	<u>Comment</u>	<u>Score</u>	<u>UnitCell</u>	<u>Mosaicity</u>	<u>Rmsd</u>	<u>BravaisLattice</u>	<u>Resolution</u>	<u>SystemWarning</u> 
 E2	rnaseE2	rnase	rnaseE2_001.img rnaseE2_002.img	0 0	oil	0.783	78.73 135.98 104.07 90.00 90.00 90.00	0.30°	0.043 mm	C222	1.98 Å	
 E1	rnaseE1	rnase	rnaseE1_001.img rnaseE1_002.img	2 3	no cryo	0.783	77.46 77.46 103.40 90.00 90.00 120.00	0.40°	0.036 mm	P3,P312,P321,P6,P622	1.88 Å	
 D8	rnaseD8	rnase	rnaseD8_001.img rnaseD8_002.img	0 0	solution 4e	0.144	78.95 78.95 103.60 90.00 90.00 120.00	1.50°	0.071 mm	P3,P312,P321,P6,P622	2.59 Å	
 D7	rnaseD7	rnase	rnaseD7_001.img rnaseD7_002.img	0 0	solution 4e	0.630	77.44 77.44 104.09 90.00 90.00 120.00	0.70°	0.076 mm	P3,P312,P321,P6,P622	2.22 Å	
 D6	rnaseD6	rnase	rnaseD6_001.img rnaseD6_002.img	0 0	solution 4e	0.656	78.03 78.03 103.95 90.00 90.00 120.00	0.70°	0.083 mm	P3,P312,P321,P6,P622	1.84 Å	
 D5	rnaseD5	rnase	rnaseD5_001.img rnaseD5_002.img	6 0	solution 4e	0.637	78.20 78.20 104.08 90.00 90.00 120.00	0.80°	0.090 mm	P3,P312,P321,P6,P622	1.72 Å	
 D4	rnaseD4	rnase	rnaseD4_001.img rnaseD4_002.img	2 2	solution 4e	N/A				N/A		No_Indexing_Solution: Too few candidate Bragg spots (34) in image 1

Figure 7: SSRL *Web-Ice* interface displaying the summarized results of data collection strategy calculation. After inspecting the results, the users have the choice to proceed to data collection, edit the data collection parameters or recollect the initial test images and calculate the data collection strategy again for different initial conditions.

Solution12

Space Groups	Phi Range		Completeness		Max Delta Phi
	Unique	Anomalous	Unique	Anomalous	
P3	-36.0 to 54.0	-36.0 to 54.0	100.0%	93.7%	4.60
P312	-42.0 to 48.0	-42.0 to 48.0	100.0%	100.0%	4.60
→ P321	-47.0 to 43.0	-32.0 to 58.0	100.0%	100.0%	4.60
P6	-47.0 to 43.0	-47.0 to 43.0	100.0%	99.4%	4.60
P622	-52.0 to 38.0	-57.0 to 33.0	100.0%	100.0%	4.60

Space Group P321:

Strategy	Overlap Analysis	Completeness Analysis: Unique	Anomalous
Data Collection Strategy <input type="button" value="i"/>			
Experiment Type	Monochromatic		
Mosaicity and Score	80% mosaicity=0.20 deg., score = 0.72		
Oscillation Start	-47.0°		
Oscillation End	43.0°		
Oscillation Angle	1.0°		
Oscillation Wedge	180.0°		
Resolution	1.55 Å		
Exposure Time	2.31 sec		
Attenuation	0.0 %		
Optimal Detector Distance	210.5 mm		
Beamstop Distance	32.7 mm		
Energy	12670.1 eV		
Detector Type	ADSC_QUANTUM315		
Detector Mode	binned		
Inverse Beam	No		
Number of images	90		
Estimated Absorbed Dose <input type="button" value="i"/>	1.89e+06 Gy (21000 Gy per image) total. <i>Limit is 3.0e+07 Gy.</i>		