Journal of
Applied
Crystallography

ISSN 0021-8898

Received 12 March 2012
Accepted 13 June 2012

Bonsu: the interactive phase retrieval suite

Marcus C. Newton,[a]* Yoshinori Nishino[a] and Ian K. Robinson[b]

[a]Laboratory of Coherent X-ray Optics, Hokkaido University, Japan, and [b]London Centre for Nanotechnology, University College London, England. Correspondence e-mail: m.newton@es.hokudai.ac.jp

Coherent X-ray diffraction imaging has received considerable attention as a nondestructive method for probing material structure at the nanoscale. However, tools for reconstructing and analysing data in both two and three dimensions have lagged somewhat behind. *Bonsu*, the interactive phase retrieval suite, is the first software package that allows real-time visualization of the reconstruction of phase information in both two and three dimensions. It comes complete with an inventory of algorithms and routines for data manipulation and reconstruction. *Bonsu* is open source, is designed around the Python language (with C++ bindings) and is largely platform independent. *Bonsu* is made available under version three of the GNU General Public License and can be found at http://www.code.google.com/p/bonsu.

1. Introduction

The development and use of high-energy high-brilliance X-ray synchrotron radiation facilities have proven indispensable for studying the structural properties of materials and various fundamental physical phenomena with atomic scale sensitivity (Chapman, 2009; Robinson & Harder, 2009; Newton *et al.*, 2010; Nishino *et al.*, 2009). The advent of fourth-generation X-ray free electron laser (XFEL) synchrotron radiation facilities such as LCLS in the USA and SACLA in Japan, as a tool for materials analysis, creates the potential for unprecedented insight into atomic scale structure (Schmüser *et al.*, 2008; Chapman *et al.*, 2011).

Coherent X-ray Diffraction Imaging (CXDI) methods have emerged from the realization that oversampled diffraction patterns can be inverted to obtain real-space images (Sayre, 1952). Provided the diffraction intensity is recorded with a finer spacing than twice the Shannon–Nyquist spatial frequency, the inversion of the diffraction pattern to an image is an overdetermined problem. Except for elementary symmetries of the solution and certain pathological cases, notably factorization in the one-dimensional case, a single unique solution can be expected (Bates, 1982). Nevertheless, CXDI remains a powerful tool for probing materials with sub-nanometre resolution (Robinson & Miao, 2004).

There are currently several important variations on the conventional CXDI method of illuminating a sample with a spatially coherent X-ray beam and recording the diffraction pattern on an area detector with pixels smaller than half the fringe spacing, to ensure oversampling. It was found that focusing the coherent beam to measure smaller samples did not worsen CDI's ability to obtain images (Robinson *et al.*, 2003). A significant improvement was found in the convergence of the phasing algorithm when a deliberately curved wavefront was used for illumination, resulting in a method called Fresnel CDI (Nugent, 2010). Further gains were achieved by scanning a coherent probe beam across a sample with enough overlap; this ptychography method (Rodenburg & Faulkner, 2004; Rodenburg *et al.*, 2007) uses the added 'diversity' of diffraction data to further constrain their inversion to an image. This technique is notable for allowing reliable quantitative phase determination. Ptychography also allows access to an arbitrarily large field of view, since the probe beam is smaller than the sample.

This paper focuses on the classical CDI situation where the coherent beam size, and hence its coherence length, exceeds the dimensions of the crystal. Once the sample is aligned to excite a Bragg reflection, the scattered waves from the entire sample volume of the crystal interfere in the far field, producing a two-dimensional k-space diffraction pattern on an area detector (Robinson & Miao, 2004; Miao *et al.*, 2000; Warren, 1990). Rocking the sample through a small angular range causes the full three-dimensional distribution of the Bragg peak to pass through the Ewald sphere, resulting in a three-dimensional stack of two-dimensional sections. This three-dimensional diffraction pattern is complete in the sense that the intensity drops off to the noise level in all three reciprocal-space directions. In an appropriate coordinate system, the diffraction pattern represents the square magnitude of the Fourier transform of the three-dimensional electron density distribution inside the sample (Williams *et al.*, 2003). A powerful generalization of the problem is that crystal distortions due to strains can be mapped, through projection onto the fixed momentum transfer vector of the Bragg reflection, onto a complex density function in real space, still connected by Fourier transformation to the diffraction pattern measured in reciprocal space (Robinson & Harder, 2009).

By using small enough detector pixels, a large enough detector distance and fine enough angle steps in the rocking scan, this pattern can be oversampled relative to the Shannon–Nyquist sampling frequency in all three directions. Technically, this operation of measuring a rocking curve with a two-dimensional detector results in 'binned' (angularly averaged) data in two directions and 'sampled' data in the third direction, but this does not make a big difference to the performance of computational algorithms (Song *et al.*, 2007).

The Ewald sphere curvature effects are usually small enough that the pattern is also sampled on a regular grid in all three directions, so can be transformed in the inversion algorithm using a discrete Fourier transform, such as the well known fast Fourier transform. This three-dimensional diffraction pattern, sampled as a three-dimensional array of numbers, is then inverted with the aid of computational iterative phase retrieval techniques, developed over a number of years. The algorithm package described in this paper is the realization of these techniques.

When coherent diffraction measurements are performed phase information (φ) is lost and in general we obtain the reciprocal-space intensity distribution $I(\mathbf{k}) = |\hat{\rho}_0(\mathbf{k})|^2 = |\hat{\rho}_0(\mathbf{k})\exp[i\varphi(\mathbf{k})]|^2$, where $\hat{\rho}(\mathbf{k})$ is the complex density in Fourier space and the subscript 0 indicates measured values of the density. We wish to recover the complex real-space density $\rho(\mathbf{r})$. This is achieved using iterative reconstruction algorithms that traverse back and forth between direct and Fourier space while applying a constraint at each turn. A successful approach is to define a support region in real space for which the amplitude of the object density is unrestricted. Outside this region the amplitude of the density $\rho(\mathbf{r})$ is minimized in some way. The Fourier-space constraint ($\mathbf{P}_{\rm FS}$) requires the object's amplitude to be proportional in some way to the original measurement on some set $\mathcal{M}$, such that $\mathbf{P}_{\rm FS}\,|\hat{\rho}(\mathbf{k})|\exp[i\varphi(\mathbf{k})] = |\hat{\rho}_0(\mathbf{k})|\exp[i\varphi(\mathbf{k})]$ for all $\mathbf{k}$ in $\mathcal{M}$.

The development of iterative phase retrieval algorithms has allowed the field of CXDI to grow rapidly in recent years as a robust tool for lensless imaging in both two and three dimensions. Noteworthy examples include the Gerchberg & Saxton (1972) algorithm and Fienup's hybrid input–output algorithm (Fienup, 1982).

To date, however, there are few tools if any that integrate data analysis with the phase reconstruction process. Such a scenario is, for example, desirable when reconstructing phase information using a shrink-wrapped support in three dimensions (Newton *et al.*, 2010). *Bonsu*, the interactive phase retrieval suite, aims to be an encompassing tool for the reconstruction of phase information from continuous diffraction measurement data.

Emphasis is placed on structure determination in three dimensions while two dimensional support is also provided. Key advancements include real-time structure visualization of static data and data during reconstruction in both three and two dimensions and an inventory of tools for data manipulation.

Bonsu is open-source copyrighted software: it aims to be useful to all within the community and beyond where possible. This enables scientific users within differing circles to modify and share the source code as and when the need arises. Such a paradigm provides the flexibility often required in the dynamic field of coherent X-ray imaging. In the following, we outline the specifications of *Bonsu* and basic user operation.

2. Program outline

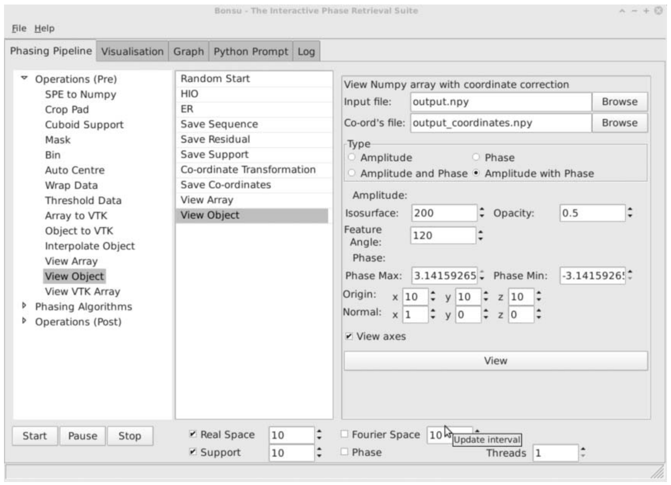
2.1. Architecture

Bonsu is open-source software written primarily in the Python (http://python.org) object-oriented programming language (Lutz, 2011). Python is written in the C language with primitives accessible to the developer. As Python is open source and largely platform independent, *Bonsu* also inherits these properties. The interface is built around wxWidgets (http://wxwidgets.org) with the standard Python bindings as provided. All arrays for data manipulation use the NumPy array format (http://numpy.scipy.org). NumPy is part of the 'Scientific Computing Tools For Python' or SciPy package (http://scipy.org) available in Python. Data manipulation routines utilize tools largely from NumPy and SciPy. Some tools are written directly in C++. Fourier transforms are performed using the FFTW package (http://fftw.org) through the C++ language (Frigo & Johnson, 2005). This is seamlessly compiled and integrated into *Bonsu* by using opaque developer tools for Python. For this reason, *Bonsu* makes passes to a compiler at runtime and therefore requires a C++ compiler such as the GNU C++ compiler (http://gcc.gnu.org) to be present. The dynamic graphical rendering of data objects is achieved using the Visualization Toolkit (VTK) library with standard Python bindings (http://vtk.org). *Bonsu* allows for ease of translation from data to article figures with the aid of its visual interface and additional Python tools such as matplotlib (http://matplotlib.sourceforge.net).

Perhaps the simplest way to install *Bonsu* on Microsoft Windows is to install all the necessary Python tools and FFTW first. This is made simple with the Enthought Python Distribution (EPD) for Windows (http://enthought.com). A similar approach will suffice for Apple Macintosh users. For Unix and Linux users, the vast majority of packages are available from the package distributors while the remainder should be compiled.

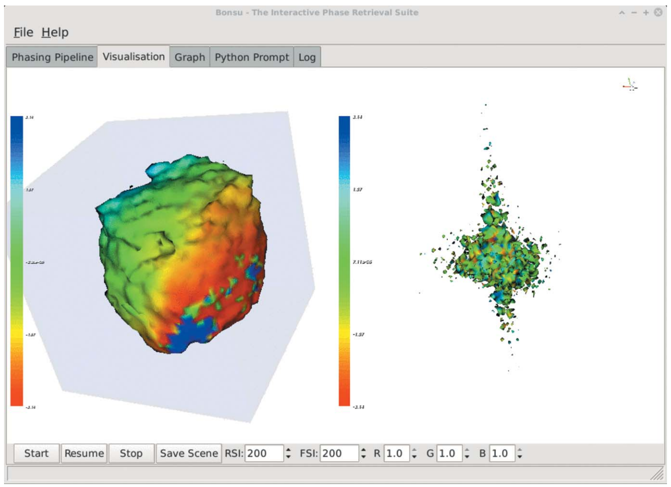
2.2. Interface

The graphical user interface is divided into a number of tabs as shown in Fig. 1. These are 'Phasing Pipeline', 'Visualisation', 'Graph', 'Python Prompt' and 'Log'. Most user interaction occurs in the 'Phasing Pipeline' tab. This tab is divided into three columns (Fig. 1). From left to right, the first shows a drop-down menu divided into three sections where various array operations, phasing algorithms and posterior phasing operations can be found. Once selected from



View Numpy array with coordinate correction
Input file: output.npy
Co-ord's file: output_coordinates.npy
Type: Amplitude ○ Phase ○ Amplitude and Phase ○ [✓ Amplitude with Phase]
Amplitude:
Isosurface: 200 Opacity: 0.5
Feature Angle: 120
Phase:
Phase Max: 3.14159265 Phase Min: -3.1415926
Origin: x 10 y 10 z 10
Normal: x 1 y 0 z 0
✓ View axes
✓ Real Space 10
✓ Support 10
☐ Fourier Space 10
☐ Phase
Threads 1

Figure 1
Graphical user interface of *Bonsu*, showing the 'Phasing Pipeline' tab.



RSI: 200 FSI: 200 R 1.0 G 1.0 B 1.0

Figure 2
Graphical user interface of *Bonsu*, showing the 'Visualisation' tab during the phase reconstruction process. Both the amplitude of the real-space object and its Fourier transform are shown. The phase is mapped onto the iso-surface of the object.

the drop-down menu, an instance of the item appears in the centre column (or pipeline), where it can be selected and properties modified for that instance. The properties appear in the right-hand column. Once items have been added to the pipeline as shown in Fig. 1, the user can save the contents of the pipeline by simply choosing 'Save' from the menu. The user will be prompted for a file name. *Bonsu* saves pipeline data in `.fin` file format. Similarly, to recover a pipeline, simply open a `.fin` file. *Bonsu* also allows pipelines to be appended. For example, opening a `.fin` file twice will append the list of tasks twice into the pipeline. To clear the pipeline, simply select 'New' from the menu.

Below the three columns are the control buttons 'Start', 'Pause' and 'Stop'. These buttons will, respectively, execute all items in the pipeline in the order that they appear, pause the execution of an algorithm or stop the execution of an algorithm. The remaining checkboxes control the type of visualization that appears on the 'Visualisation' tab and the maximum number of threads used by the Fourier transform algorithm. The options include no visualization, real-space amplitude, real-space amplitude with phase, Fourier-space amplitude and Fourier-space amplitude with phase.

The 'Visualisation' tab is where the reconstruction will appear. This is illustrated in Fig. 2 for a three-dimensional object. From here the user can interact fully with and manipulate the object as it is being reconstructed. Images presented here can also be saved for further use. The 'Graph' tab displays the reconstruction error plotted on a graph with dynamic axes. From here the graph can be paused (independently of the reconstruction process) and the data saved to a text file. The 'Python Prompt' tab enables the user to perform additional tasks through the Python interpreter. This might be the execution of additional Python scripts or additional array operations with standard SciPy tools. The 'Log' tab allows the user to view additional information on the program's operation and any graph data values in text form. From here the log data can be saved to a text file.

3. Algorithms

Phase retrieval algorithms are implemented in the C++ language for fast execution and seamlessly integrated into the application. *Bonsu* comes complete with a number of algorithms for phase retrieval. They include Fienup's hybrid input–output (HIO) (Fienup, 1982), HIO with positivity constraint, phase-constrained HIO (Harder *et al.*, 2010), hybrid projection reflection (Bauschke *et al.*, 2003), relaxed average alternating reflection (Luke, 2005) and the error reduction algorithm (Gerchberg & Saxton, 1972; Fienup, 1982). In addition, a shrink-wrap algorithm that is able to make use of either of the above algorithms is also provided (Marchesini *et al.*, 2003; Newton *et al.*, 2010). Here the support constraint can be updated at defined iteration intervals. The support update is defined from the object amplitude by first removing data below a threshold in amplitude and then convolving with a Gaussian with standard deviation σ.

When multiple algorithms are used in a pipeline, they can share the same support data and will share the same reconstruction data. This is particularly useful when using the shrink-wrapped support. This allows the application of multiple algorithms with varying parameters to a single reconstruction attempt. Session data and arrays can be cleared by simply selecting 'New' from the menu. Fig. 3 illustrates a typical use for phase reconstruction.

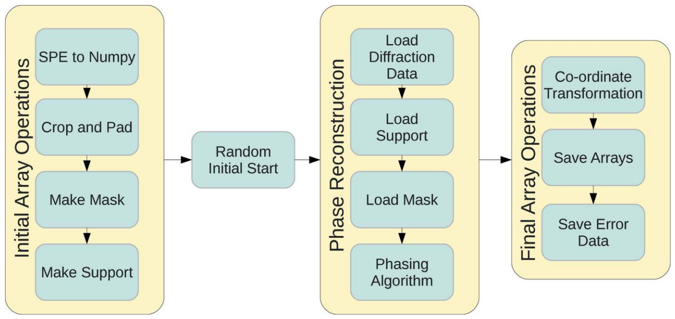


Figure 3
Flow chart showing a typical use of the phasing pipeline. Initial array preparation is followed by phase reconstruction. Finally a coordinate transformation is performed and the data are saved.

4. Array operations

There are a number of standard tools made available for array conversion, manipulation and viewing. *Bonsu* provides import filters for converting SPE file format (Princeton Instruments) into NumPy format. The Python language, *via* NumPy, natively provides a number of tools for importing images in formats such as TIFF and converting them to NumPy arrays. It is also possible to import HDF5 format files into Python. Tools for cropping, padding, binning, support array creation and mask array creation are provided. Arrays can be exported in both NumPy array and VTK format.

Posterior phasing operations are generally applied to the object after the phase reconstruction process. This includes saving error residual information for reconstruction data. Coordinate transformations are also possible on three-dimensional rocking curve data (Robinson & Miao, 2004). The implementation used is based on geometric transformations from the diffraction geometry to a rectilinear coordinate space with orthonormal basis (Pfeifer, 2005). The coordinate transformation operation yields a coordinate array that can be used for visualization. Visualization of NumPy arrays such as diffraction and reconstructed data is possible with both uncorrected and coordinate-corrected data in both two and three dimensions.

5. Summary

Bonsu is a versatile tool for phase retrieval and data visualization in both two and three dimensions. It is developed exclusively using free and open-source software tools, making it widely accessible. It is hoped that this software will be of use to the coherent diffraction imaging community. The developers aim to provide further enhancements in future releases with additional algorithms being incorporated. As always we are open to comments and suggestions.

References

Bates, R. H. T. (1982). *Optik*, **61**, 247–262.
Bauschke, H. H., Combettes, P. L. & Luke, D. R. (2003). *J. Opt. Soc. Am. A*, **20**, 1025–1034.
Chapman, H. N. (2009). *Nat. Mater.* **8**, 299–301.
Chapman, H. N., Fromme, P. *et al.* (2011). *Nature* (*London*), **470**, 73–77.
Fienup, J. R. (1982). *Appl. Opt.* **21**, 2758–2769.
Frigo, M. & Johnson, S. G. (2005). *Proc. IEEE*, **93**, 216–231.
Gerchberg, R. W. & Saxton, W. O. (1972). *Optik*, **35**, 237–246.
Harder, R., Liang, M., Sun, Y., Xia, Y. & Robinson, I. K. (2010). *New J. Phys.* **12**, 035019.
Luke, D. R. (2005). *Inverse Probl.* **21**, 37–50.
Lutz, M. (2011). *Programming Python.* Sebastopol: O'Reilly.
Marchesini, S., He, H., Chapman, H. N., Hau-Riege, S. P., Noy, A., Howells, M. R., Weierstall, U. & Spence, J. C. H. (2003). *Phys. Rev. B*, **68**, 140101.

842 Marcus C. Newton *et al.* • *Bonsu* *J. Appl. Cryst.* (2012). **45**, 840–843

Miao, J., Kirz, J. & Sayre, D. (2000). *Acta Cryst.* D**56**, 1312–1315.
Newton, M. C., Leake, S. J., Harder, R. & Robinson, I. K. (2010). *Nat. Mater.* **9**, 120–124.
Nishino, Y., Takahashi, Y., Imamoto, N., Ishikawa, T. & Maeshima, K. (2009). *Phys. Rev. Lett.* **102**, 018101.
Nugent, K. A. (2010). *Adv. Phys.* **59**, 1–99.
Pfeifer, M. A. (2005). PhD thesis, University of Illinois, Urbana-Champaign, IL, USA.
Robinson, I. & Harder, R. (2009). *Nat. Mater.* **8**, 291–298.
Robinson, I. & Miao, J. (2004). *MRS Bull.* **29**, 177–181.
Robinson, I., Pfeiffer, F., Vartanyants, I., Sun, Y. & Xia, Y. (2003). *Opt. Express*, **11**, 2329–2334.
Rodenburg, J. M. & Faulkner, H. M. L. (2004). *Appl. Phys. Lett.* **85**, 4795–4797.
Rodenburg, J. M., Hurst, A. C., Cullis, A. G., Dobson, B. R., Pfeiffer, F., Bunk, O., David, C., Jefimovs, K. & Johnson, I. (2007). *Phys. Rev. Lett.* **98**, 034801.
Sayre, D. (1952). *Acta Cryst.* **5**, 843.
Schmüser, P., Dohlus, M. & Rossbach, J. (2008). *Ultraviolet and Soft X-ray Free-Electron Lasers*, Springer Tracts In Modern Physics, Vol. 229. Berlin: Springer-Verlag.
Song, C., Ramunno-Johnson, D., Nishino, Y., Kohmura, Y., Ishikawa, T., Chen, C.-C., Lee, T.-K. & Miao, J. (2007). *Phys. Rev. B*, **75**, 012102.
Warren, B. E. (1990). *X-ray Diffraction.* Mineola: Dover Publications.
Williams, G. J., Pfeifer, M. A., Vartanyants, I. A. & Robinson, I. K. (2003). *Phys. Rev. Lett.* **90**, 175501.