Developments towards Bragg edge imaging on the IMAT beamline at the ISIS pulsed neutron and muon source: BEAn software

Alexander Liptak\textsuperscript{1,5}, Genoveva Burca\textsuperscript{2}, Joe Kelleher\textsuperscript{2}, Evgueni Ovtchinnikov\textsuperscript{3}, Jacob Maresca\textsuperscript{4} and Aled Horner\textsuperscript{1}

\textsuperscript{1} Royal Holloway, University of London, Egham, TW20 0EX, United Kingdom
\textsuperscript{2} ISIS Pulsed Neutron and Muon Source, Rutherford Appleton Laboratory, Harwell Campus, Didcot, OX11 0QX, United Kingdom
\textsuperscript{3} STFC, Rutherford Appleton Laboratory, Harwell Campus, Didcot, OX11 0QX, United Kingdom
\textsuperscript{4} School of Physics and Astronomy, University of Nottingham, Nottingham, NG7 2RD, United Kingdom
\textsuperscript{5} Author to whom any correspondence should be addressed.

E-mail: Alexander.Liptak.2015@live.rhul.ac.uk, genoveva.burca@stfc.ac.uk, joe.kelleher@stfc.ac.uk, evgueni.ovtchinnikov@stfc.ac.uk, jacob.maresca@nottingham.ac.uk and aled.horner@gmail.com

Keywords: neutron imaging, Bragg edge imaging, principal component analysis, time-of-flight neutron source

Abstract
The BEAn (Bragg Edge Analysis) software has been developed as a toolkit for analysis of Bragg edges and strain maps from data obtained at the time-of-flight imaging instrument IMAT or other compatible instruments. The code is built primarily using Python 3 and the Qt framework, and includes tools useful for neutron imaging such as principal component analysis. This paper introduces BEAn and its features, briefly discusses the scientific concepts behind them, and concludes with planned future work on the code.

1. Introduction

The ISIS Neutron and Muon Source is a world-leading pulsed neutron and muon source owned and operated by the Science and Technology Facilities Council (STFC). Situated at the Rutherford Appleton Laboratory in Oxfordshire, United Kingdom, it is one of 15 major neutron facilities in the world, and one of few to have a significant research group working on new data collection techniques. The site houses over 30 neutron instruments specialised for various diffraction or spectroscopy techniques, including 4 newly built Phase Two instruments on Target Station 2. One of these instruments is IMAT (Imaging and Materials Science & Engineering)\textsuperscript{1, 2}, a neutron imaging and diffraction instrument for a broad range of materials and engineering sciences.

Energy-resolved neutron imaging is a technique that was recently implemented on IMAT, enabling the instrument to carry out spatially-resolved measurements of physical quantities and elemental distributions within a sample. Because this method non-destructively investigates the crystallographic and metallographic structure of materials, it has attracted attention in the research field of material characterisation calling for further developments to achieve better results\textsuperscript{3}. By utilising a pulsed neutron source such as the one at ISIS, the time-of-flight method can be used to carry out these energy-resolved neutron imaging experiments. A transmission spectrum of neutrons through a sample can then be obtained from this data, which shows sharp increases in neutron transmission at certain wavelengths (Bragg edges).

A Bragg-edge transmission spectrum analysis code would be very useful for users of the IMAT instrument, providing an easier way to extract information from their experimental data. A number of implementations such as the RITS code (a whole-spectrum fitting analysis code developed in C++ at the J-PARC facility, Japan)\textsuperscript{3}, iBeatles (individual edge fitting code with a GUI developed in Python at the SNS facility, U.S.)\textsuperscript{4} and TPXEdgeFit (individual edge fitting MS Excel spreadsheet with an associated C++ code)\textsuperscript{5} have been created at other facilities for their users. The BEAn (Bragg Edge Analysis) software has been developed to respond to the IMAT user demand, designed as a semi-automated data normalisation, processing and analysis toolkit written in the Python framework.

© 2019 The Author(s). Published by IOP Publishing Ltd
2. Bragg edge imaging

At the heart of ISIS, neutron pulses are created by accelerating bunches of protons inside a proton synchrotron to high energies, then subsequently colliding them with a tungsten target. When impacted by the proton beam, neutrons spill off the tungsten atoms and are distributed via beamlines to all ISIS instruments. As neutrons are neutral and usually have low absorption by solids, they can penetrate into matter much deeper than electrons or x-rays can and are therefore much more useful for non-destructive studies of bulk materials. In addition, because neutrons have both mass and spin, their magnetic moment can be used to probe magnetism on an atomic scale.

Neutrons that impact on a crystalline solid sample are scattered by a number of well-defined angles defined according to Bragg’s law, the same law that also governs the diffraction of x-rays. Bragg’s law states that when a neutron with wavelength \( \lambda \) scatters from a crystal lattice with a lattice separation \( d \), the \( n \)th order of scattering occurs at angle \( \theta \), according to equation (1).

\[
n\lambda = 2d \sin \theta
\]  

Despite the fact that both neutron and x-ray diffraction is described by the same law, the fundamental interaction processes involved are very different. Unlike x-ray photons, which scatter due to the electromagnetic interaction with an atom’s electron cloud, neutrons with a particular wavelength form highly penetrating radiation that interacts directly with atomic nuclei, mainly via the residual strong force. The neutron scattering power of an atom is not dependent on the atom’s atomic number and varies from isotope to isotope without a simple continuous function. This behaviour is useful for distinguishing between isotopes of the same element or atoms with similar atomic numbers that would otherwise be indistinguishable when imaged with x-rays [6].

Standard neutron diffraction experiments measure scattered neutrons as a wavelength-dependent neutron intensity spectrum at some fixed angle from the neutron beam. The peaks in the measured intensity spectrum are termed Bragg peaks and are a result of interference between neutrons scattered by the sample’s crystal lattice. As neutrons scatter in all directions, it is clear that the detector measures only a small fraction of neutrons incident on the sample. Alternatively, by measuring the number of transmitted neutrons through the sample instead of neutrons scattered by the sample, intensity spectra with a much higher total neutron count can be obtained. With this setup, the resulting intensity spectrum is often referred to as a transmission spectrum, and the observed Bragg edges represent the crystal lattice parameter averaged over the path of the neutron beam through the sample. A detector with a good enough spatial resolution would also allow for the reconstruction of a two-dimensional map of averaged lattice parameter variations from the transmission spectrum [7], with the advantage that the number of transmitted neutrons is often much higher and thus the data collection can be performed much faster [6].

In crystallography, a family of lattice planes is uniquely labelled using a notation system involving the integers \( h, k \) and \( l \), referred to as Miller indices, representing the coefficients of the three basis vectors of a crystal’s reciprocal lattice. ISIS provides a wide range of neutron energies with respective neutron wavelengths in a single pulse, so that a large number of Bragg edges corresponding to different \( hkl \) planes can be observed in a single measurement. Using the time-of-flight (TOF) method to sort the energies of neutrons arriving at the detector, the wavelength \( \lambda \) of any neutron can then be calculated from a measurement of its total travel time \( t \) (from the start of the pulse until detection) and the prior knowledge of the distance \( L \) it travelled in that time [1]. The de Broglie equation relates neutron wavelength to its momentum \( p \), which is equivalent to the product of its mass \( m \) and velocity \( v \). By substituting \( v = L/t \), the relation between the neutron wavelength and its time-of-flight is given by equation (2).

\[
\lambda = \frac{h}{p} = \frac{h}{mv} = \frac{ht}{mL}
\]  

The resolution of TOF measurements is largely determined by the width of the neutron pulse, the length of the flight path, and the time resolution of the neutron detector [7].

3. BEAn package

The Bragg Edge Analysis code (BEAn) has been developed as a software package for easy and fast analysis of Bragg edges from transmission data obtained at ISIS IMAT or any other compatible instrument. Due to the standardised way in which data from Timepix detectors such as those used at IMAT is stored [8], the code may already be compatible (or can be easily made compatible) with other neutron instruments at ISIS that specialise in Bragg edge imaging measurements (e.g. Larmor or Engin-X). The BEAn code could also be used for processing data from other detectors designed for Bragg edge measurements such as GP2 [9].
BEAn is written in Python 3.7 and uses PyQt5 bindings to C++ Qt libraries for its user interface and threading. NumPy is used for all computationally expensive calculations and the majority of numerical data is stored using fixed data-type NumPy arrays in order to minimise BEAn’s memory footprint and computation time. A custom Matplotlib widget for PyQt5 is used for all data visualisation as it provides a familiar and responsive interface for interactive plotting. BEAn is also packaged using PyInstaller to allow users to launch the software without having to install a compatible version of Python and the required libraries. Finally, both the BEAn package and the source code are tested to fully support running under Windows, Linux and macOS.

3.1. Motivation
In order to fully utilise the scientific potential of the IMAT instrument, which was formally opened in October 2016, its users need to be able to perform quick data analysis on-site. As energy-resolved neutron imaging is a new technique, no open-source software packages are available on the market, and therefore the development of an open-source Bragg Edge Analysis code that will be available on the IMAT instrument for all users is a goal that needs to be achieved. A plugin for ImageJ, a Java-based image processing program, has been developed at IMAT to aid with the routine processing of transmission data [10], however, it was found to perform slower than expected and does not support many import and export formats.

The BEAn software has been proposed to fill the gap by providing a portable and easy-to-use toolkit for Bragg edge analysis, strain mapping, principal component analysis and other auxiliary tasks that are commonly performed by users at IMAT or other ISIS beamlines that support time-of-flight imaging measurements.

Figure 1 shows the user interface of the BEAn main window with a loaded sample data set.

3.2. Summary of significant features
The following is a list of BEAn’s most significant features, intended as a summary of reasons why the software is suitable for implementation at IMAT and why it may be suitable for use at other Bragg edge imaging beamlines as well:

- **Availability of features**: The calculation of transmission spectra from energy-resolved neutron data, automatic Bragg edge detection and fitting, comparison of transmission data against a Bragg edge database, plotting of summed neutron images, calculation of the z-axis profile, residual strain mapping and principal component analysis are some of the features currently available in BEAn.

- **Versatility of input**: Data can be loaded into BEAn from a number of different file formats, including the native FITS format used by the Timepix chip of the microchannel plate (MCP) detector [8], ZIP archives of FITS files (without decompressing to disk), as well as the standard NPZ format used to persist NumPy arrays on disk.

- **Simplicity of use**: BEAn is portable and works out-of-the-box; it does not require Python to run, works straight after extraction from the archive by which it is distributed, and can be launched from an external drive. The software has an intuitive GUI and uses the familiar Matplotlib library for all interactive plotting.

- **Performance**: Despite being built in Python, all computationally expensive calculations are carried out using NumPy (written in C), and threading is utilised throughout the code. The combination of these solutions allows BEAn to already achieve C-like speeds in most aspects, with further performance upgrades planned.

- **Scalability**: BEAn’s modular design and the standardised way in which most Bragg edge imaging instruments output their data allows for BEAn to be made compatible with those instrument with little required modification of the code.

3.3. Data manipulation
The FITS data format, an acronym for Flexible Image Transport System, is a NASA-endorsed standard format for data used in astronomy [11], that is also utilised by the Timepix chip of the MCP detector [8] used at IMAT. A data acquisition run produces a folder of a few thousand FITS files and other auxiliary files that often exceed 1 GB in size, and is therefore usually compressed for the sake of portability. BEAn fully supports both the loading of data directly from folders, or from zipped archives without having to first extract the contents of the archive to disk. During the loading process, the data is also formatted and corrected depending on the settings chosen prior to starting the process. The following is a summary of the procedure involved in loading data into BEAn when all recommended settings are used:

- **Integrity test**: If loading data from a .zip archive, an integrity check is carried out by default to test whether the archive isn’t corrupted prior to loading data.
• **Data loading:** Image data is loaded from all available files into memory as a 3D NumPy array with the smallest fixed data-type that can accommodate the data. If loading data from an archive, the files are expanded to memory before being used.

• **Faulty pixel correction:** The data is scanned for any dead pixels (those that recorded no events for all neutron wavelengths) and ‘misbehaving’ pixels (those that recorded an abnormally large number of events). The identified pixels are corrected by assuming their value to be the average of all directly neighbouring pixels. This stage often reduces the smallest required data type of the array, significantly lowering its memory requirements.

• **Overlap and scaling correction:** All data is optionally overlap-corrected, and open beam data is also scaling-corrected. The following section describes these correction processes in more detail.

Depending on the size of loaded data and the number of optional tasks to be performed, loading a single data set may take up to a few minutes. For this reason, it is possible to save the loaded (and corrected) data to disk as an NPZ or compressed NPZ file—the standard format for persisting multiple NumPy arrays on disk [12]. The loading of NPZ files usually takes only a few seconds, and NPZ data often has significantly lower disk space requirements compared to the original FITS data set, especially when compressed.

### 3.4. Overlap and scaling correction

The Timepix sensor has the capability to detect the position of an incident neutron with an accuracy of 55 μm and measure its time of arrival with a resolution of up to 10 ns [13]. With a 1 kHz parallel readout capability, the
majority of neutron events will be detected correctly by the sensor, but on occasions where the incoming neutron flux is too large, many neutrons arriving late in the shutter will not be registered as the relevant pixels are already occupied by previous events. This decrease in the efficiency of the sensor towards the end of the shutter is termed overlap, and the corresponding overlap correction is carried out as follows; considering the nth shutter, the probability \( P(t_i) \) that a pixel is occupied and is unable to detect anymore neutrons at time \( t_i \) relative to the start of the shutter is given by equation (3)

\[
P(t_i) = \sum_{j=0}^{j=i-1} \frac{N(t_j)}{S_n}; \quad i > 1; \quad P(t_0) = 0
\]

Here, \( N(t_j) \) is the number of neutron events detected at time \( t_j \) and \( S_n \) represents the number of triggers that occurred during the nth shutter. From this result, it is possible to calculate the number of events \( N'(t_i) \) that should have been detected at time \( t_i \) using Poisson statistics, as shown in equation (4).

\[
N'(t_i) = \frac{N(t_i)}{1 - P(t_i)}; \quad i > 1; \quad N'(t_0) = N(t_0)
\]

Equations (3) and (4) can be applied to all frames on a pixel-by-pixel basis [13] and to all shutters in the data set. As overlap-corrected data is a much better representation of the true neutron transmission, it is carried out by default during the data loading process.

In order for BEAn to be able to calculate transmission spectra, both sample and open beam data sets are required. The process involved is described further in the following section, however, it should be taken into account that the two data sets will not necessarily have been obtained with the same number of triggers (neutron pulses). Although this does not affect the wavelengths of the resulting Bragg edges, it does shift the neutron transmission away from its true value. Scaling correction is used to adjust the number of neutron events recorded in each shutter \( s_i \) of the open beam data \( N_{ob}(s_i) \) by a scale factor, defined as the ratio of the number of triggers in the current shutter of the sample data \( T(s_i) \) to the number of triggers in the open beam data set for the same shutter \( T_{ob}(s_i) \), as shown in equation (5).

\[
N'_{ob}(s_i) = N_{ob}(s_i) \frac{T(s_i)}{T_{ob}(s_i)}
\]

### 3.5. Bragg edge detection and fitting

BEAn obtains a transmission spectrum by calculating the neutron transmission percentage \( T_{trans}(\lambda) \) as a ratio of the transmitted neutron counts (sample data) \( I(\lambda) \) and the unobstructed neutron counts (open beam data) \( I_0(\lambda) \) for all recorded neutron wavelengths \( \lambda \), as shown in equation (6). If a 2D region-of-interest (ROI) is selected, the transmission and open beam data sets are clipped to that region before any calculations are carried out. An example of a transmission spectrum plotted by BEAn can be seen in figure 2.

\[
T_{trans}(\lambda) = \frac{I(\lambda)}{I_0(\lambda)} \times 100
\]

The process of manually fitting a single Bragg edge involves the selection of a user-defined region of the transmission spectrum in which the Bragg edge occurs, then using a function that adequately represents the
neutron transmission near a Bragg edge for non-linear least-squares fitting, which returns the fitted function parameters that correspond to, for example, the wavelength at which the Bragg edge occurs. The model refinement function \([15]\), a variation of a function derived by J. R. Santisteban in 2000 \([16]\), was the first function implemented in BEAn for this purpose. This function is given by equation (7).

\[
Tr(\lambda) = A_1 + A_2(\lambda - A_6) + \frac{A_5 - A_2}{2} \times \left( \lambda - A_6 \right) \times \left\{ \text{erf} \left( \frac{\lambda - \lambda_0}{\sqrt{2}\sigma} \right) - \exp \left( \frac{\lambda - \lambda_0 - \sigma^2}{2\alpha^2} \right) \right\} \text{erf} \left( \frac{\lambda - \lambda_0 - \sigma}{\sqrt{2}\sigma} \right) \right\}
\]

(7)

Here, \(\lambda_0\) represents the position of the Bragg edge, \(\sigma\) gives the Gaussian broadening caused by the sample and instrument setup, \(\alpha\) is the exponential decay constant defined by the geometry and temperature of the moderator, and \(A_1, A_2, A_5\) and \(A_6\) are simply linear function constants. Due to the relatively large number of parameters, the model refinement function often yielded poor fits of Bragg edges, and therefore a modified function \([17]\) with a more geometric approach that is shown in equation (8) was tested.

\[
Tr(\lambda) = C_1 + C_2 \left[ \text{erfc} \left( \frac{\lambda_0 - \lambda}{\sqrt{2}\sigma} \right) - \exp \left( \frac{\lambda_0 - \lambda - \sigma^2}{2\pi^2} \right) \text{erfc} \left( \frac{\lambda_0 - \lambda - \sigma}{\sqrt{2}\sigma} \right) \right]
\]

(8)

In this fitting function, \(\lambda_0\) and \(\sigma\) have the same meaning as when used in the model refinement function, but \(\tau\) is defined as a measure of asymmetry of the Bragg edge, \(C_1\) represents its ‘pedestal’, and \(C_2\) represents its height. Both functions have now been successfully used to obtain good fits of transmission data near a Bragg edge with the help of an auxiliary function that attempts to quickly guess the approximate parameter values prior to fitting, as can be seen in figure 3. BEAn provides additional tools to evaluate the quality of fit, such as plotting of the root-mean-square deviation (RMSD) and mean-absolute-error (MAE) bands.

In order to rapidly detect and fit the position of multiple Bragg edges automatically, BEAn uses a Savitzky-Golay (savgol) filter to smooth the data, which is then differentiated and smoothed again, then finally passed to a peak-finding function that returns a list of approximate Bragg edge positions up to a specified prominence threshold. This process is visualised in figure 4. Differentiation is carried out on the assumption that the
wavelength of a Bragg edge is approximately equivalent to the wavelength at which the highest gradient of the edge occurs. With that logic, differentiating the transmission data near a Bragg edge results in a peak, where the peak’s width, height and maximum correspond to the width, prominence and wavelength of the edge, respectively, and for which a number of well-performing peak-finding libraries exist. The savgol filter was chosen as it has been found to emulate the behaviour of both the Bragg edges and peaks in the differentiated data well. The initial filtering stage is necessary to minimise the effects of statistical noise on the differentiated data; without it, the differentiated data would contain a higher density of sharp peaks that might be erroneously treated as peaks corresponding to actual Bragg edges. The effect of the second filtering stage is two-fold: the sharpest and most prominent peaks are softened to allow the peak-finding function to make better estimates of the width and prominence of a peak, while the less prominent peaks are softened further to ensure that they are ignored by the peak-finding function. A dynamic fitting window boundary, the size of which partly depends on the estimated width and prominence of the Bragg edge, is then used for fitting. All fitting window sizes within the boundary are then trialled, with the chosen fit corresponding to the one with the lowest average error. The fitting window boundary defines both the minimum window that covers the smallest acceptable part of an edge, as well as the maximum window in which the entire edge is contained alongside with a minimal amount of transmission data that is not considered to be a part of that edge. As the available fitting functions adequately describe only a limited region around a Bragg edge, the strategy of selecting a window size with minimal error is useful for finding a fit that maximises the coverage of edge-related data points (that have a negligible or negative contribution to the error) while minimising the coverage of non-edge points (which contribute significantly to the error). The fitting process itself involves using the Levenberg-Marquardt algorithm \[18\] to solve a nonlinear least-squares problem for points that lie inside the fitting window. It is important to note that filtered data is only used to obtain the estimates of wavelengths, prominences and widths of Bragg edges, and is not used during the fitting procedure.

For further analysis, BEAn is able to plot Bragg edge positions corresponding to theoretical spacing between adjacent crystallographic \{hkl\} planes for a number of elements. The database for this has been generated using data from Mirko Boin’s nxs code for neutron cross section calculations \[14\].

### 3.6. Principal component analysis

For a large set of data, it is often useful to reduce its large set of variables into a smaller set that still preserves most of the statistical information from the larger set, thereby reducing the dimensionality of the problem in order to make the data more interpretable. One of the oldest and most widely used techniques for this purpose is principle component analysis (PCA), a statistical procedure that transforms a number of potentially correlated variables into a smaller number of uncorrelated variables called principal components (PCs). In terms of linear algebra, PCs are the eigenvectors of the data covariance matrix, or alternatively, the left-singular vectors of the data matrix, the columns of which are zero-meaned data samples.

BEAn uses the pre-release version of the Python package RALEIGH, a general-purpose eigenvalue problem solver that implements the block conjugate gradient algorithm \[19\], to compute eigenvectors of the data covariance matrix. Unlike most other eigensolvers, RALEIGH does not require the number of desired eigenvalues to be specified by the user. Instead, RALEIGH uses user-defined computation-stopping criteria that are evaluated every time a number of eigenvectors have converged. During the evaluation of the stopping criteria,
it is also possible to obtain runtime information regarding the progress of the solver, such as extracting recently converged eigenvectors or visualising the spectrum computed so far.

Based on RALEIGH’s incremental computation capabilities, BEAn is able to plot left-singular vectors or the corresponding PCA images as they are computed. As this process is computationally expensive and usually only the first few PCs are useful, user-defined stopping criteria such as the target number of eigenimages, maximum number of iterations and minimum threshold variance are used to terminate the PCA process as early as possible. If a number of meaningful PCs are identified, up to three can be associated with a red, green or blue channel and reconstructed into a false-colour image that highlights certain feature of the data set that were previously indistinguishable. This is demonstrated in figures 5 and 6.

In neutron transmission analysis, PCA can be used as an aid in systematic pattern recognition of data by highlighting patterns and significant features in neutron images. As all PCs are linearly independent, PC images may also be used to highlight subtle changes in the transmission data (such changes may represent the beam profile, as an example) [20]. It is important to note that, as the goal of PCA is dimension reduction, there is no guarantee that the dimensions obtained using PCA are indeed interpretable. As a result, the main uses of PCA are descriptive rather than inferential [21].

4. Future development

Currently, BEAn distinguishes itself in being portable and running out-of-the-box: the Python app is packaged into an executable with all required dependencies and does not require any installation. Although the code is still in its first stage of development, the loading of FITS files and calculation of transmission spectra has performed faster in BEAn than any other code that has been tested. Furthermore, BEAn is designed to be an all-in-one Bragg edge analysis toolkit with a number of features discussed in this paper that are already implemented, and many features either in active development or planned for the future.

The main features currently in active development include intelligent strain-mapping, a more versatile ROI selection tool, and a debugging and logging window. When completed, strain mapping will enable users to plot a
2-dimensional map of the residual strain experienced by the sample, either in reference to a user-selected Bragg edge, or the most prominent Bragg edge identified by BEAn that is visible throughout the entire sample. Further strain mapping plans include using a signal-to-noise map to intelligently identify areas that do not correspond to the sample and exclude them from the strain map.

The current ROI selection tool only allows for the free-hand selection of a rectangular region. A more advanced tool that is being developed will support circular, polygon and free-form selections, keep a history of previously selected ROIs that can be restored, and allow for selection of exact regions defined by a list of coordinates.

The new debugging and logging window that is now partially implemented in the upcoming version of the code is a tool that mainly focuses on speeding up the development of BEAn and providing a way for BEAn users to test and feedback on the code. Besides logging of events, the tool will support parsing of simple commands that can be sent to BEAn in order to get immediate information about its internal variables or dump them to disk.

Future plans for BEAn include additional Bragg edge fitting functions, batch mode for processing and analysing a number of data sets with a single command, auxiliary features like calculating the bounded average transmission, as well as some performance improvements. The majority of these plans have been submitted as feature requests from current users of BEAn, and as a result, the future of the BEAn code is largely dependent on its users and the features they would like to see implemented first.

Acknowledgments

The authors would like to thank Dr J Koestel (SLU, Sweden) for the initial testing of BEAn and his valuable feedback and suggestions. We would also like to extend our gratitude to the SEPnet association as well as the RHUL Department of Physics; a number of summer placements at the ISIS facility in collaboration with SEPnet provided a solid foundation on top of which BEAn was built, while the staff at RHUL offered support and guidance throughout a Master’s degree during which BEAn was initially developed. Finally, we are also thankful to all those who provided us with data for the purposes of testing the functionality of BEAn, without whom the development of BEAn would not go as smoothly as it did.

ORCID iDs

Alexander Liptak @ https://orcid.org/0000-0003-1597-4193
Genoveva Burca @ https://orcid.org/0000-0001-6867-9628
Joe Kelleher @ https://orcid.org/0000-0001-7402-939X
Evgueni Ovtchinnikov @ https://orcid.org/0000-0002-9359-6514
Jacob Maresca @ https://orcid.org/0000-0002-7922-5274
Aled Horner @ https://orcid.org/0000-0001-8219-3053
References

[1] Burca G, Kockelmann W, James J and Fitzpatrick M 2013 Modelling of an imaging beamline at the ISIS pulsed neutron source J. Instrum. 8 P10001

[2] Kockelmann W et al 2015 Status of the neutron imaging and diffraction instrument IMAT Phys. Proc. 69 71–8

[3] Sato H, Kamiyama T and Kiyana Y 2011 A rietveld-type analysis code for pulsed neutron Bragg-edge transmission imaging and quantitative evaluation of texture and microstructure of a welded alpha-iron plate Materials Transaction 52 1294–302

[4] Bilheux J 2017 Imaging Bragg edge analysis tool for engineering structure (iBeatles) NeuWave’17

[5] Tremsin A 2014 Personal Communication

[6] Vogel S 2000 A Rietveld-approach for the analysis of neutron time-of-flight transmission data Ph.D. Thesis (Christian-Albrechts-Universita (https://nbn-resolving.org/urn:nbn:de:gbv:8-diss-3306)

[7] Tremsin A et al 2010 Transmission Bragg edge spectroscopy measurements at ORNL spallation neutron source J. Phys. Conf. Ser. 251 012069

[8] Vallerga J, Tremsin A, DeFazio J, Michel T, Alozy J, Tick T and Campbell M 2014 Optical MCP image tube with a quad timepix readout: initial performance characterization J. Instrum. 9 C05035

[9] Pooley D, Lee J, Brouard M, John J, Kockelmann W, Rhodes N, Schooneveld E, Sedgwick I, Turchetta R and Vallance C 2017 Development of the ‘GP2’ detector: modification of the PIMMS CMOS sensor for energy-resolved neutron radiography IEEE Trans. Nucl. Sci. 64 2970–81

[10] Burca G, Kelleher J, Peacock A, Hayes M and Deliry P 2016 An integrated tool for Bragg edge analysis at ISIS NeuWave’16

[11] McGlynn T 2017 FITS Support Office (fits.gsfc.nasa.gov)

[12] Scipy Community 2019 NumPy v1.16 Manual (docs.scipy.org)

[13] Tremsin A, Vallerga J, McPhate J and Siegmund O 2014 Optimization of timepix count rate capabilities for the applications with a periodic input signal J. Instrum. 9 C05026

[14] Boin M 2012 nxs: a program library for neutron cross section calculations J. Appl. Crystallogr. 45 603–7

[15] Steuwer A, Withers P, Santisteban J, Edwards L, Bruno G, Fitzpatrick M, Daymond M, Johnson M and Wang D 2001 Bragg edge determination for accurate lattice parameter and elastic strain measurement Physica Status Solidi (a) 185 221–30

[16] Santisteban J, Edwards L, Steuwer A and Withers P 2001 Time-of-flight neutron transmission diffraction J. Appl. Crystallogr. 34 289–97

[17] Tremsin A, McPhate J, Kockelmann W, Vallerga J, Siegmund O and Bruce Feller W 2008 Energy-resolving neutron transmission radiography at the ISIS pulsed spallation source with a high-resolution neutron counting detector IEEE Nuclear Science Symposium Conference Record (https://doi.org/10.1109/NSSMIC.2008.4774973)

[18] More J 1978 The Levenberg-Marquardt algorithm: implementation and theory Lect. Notes Math. 116 105–16

[19] Ovtchinnikov E 2008 Computing several eigenpairs of Hermitian problems by conjugate gradient iterations J. Comput. Phys. 227 9477–97

[20] Kelleher J and Burca G 2016 Feature extraction from full-spectrum neutron images using principal component analysis NeuWave’16

[21] Jolliffe I and Cadima J 2016 Principal component analysis: a review and recent developments Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 374 20150202