Improving Data Analysis for Engineering Diffraction

Kwaku Mintah
Loughborough University, Loughborough, Leicestershire, LE11 3TT, United Kingdom

Dr. Joe Kelleher
ISIS Neutron and Muon Source, Science and Technology Facilities Council, Rutherford Appleton Laboratory, Harwell Campus, Didcot, Oxfordshire, OX11 0QX. United Kingdom

Engin-X is a Neutron Diffraction instrument at ISIS Neutron and Muon Source that has been running since 2002. I was brought on from Loughborough University as a Sandwich Student in order to assist with this work, writing and adapting Python Scripts for day-to-day use on the beamline and other more arduous or irregular tasks. The scripts were written using the Python environment Jupyterlab, working on top of or alongside algorithms from Mantid. There are three main areas I have written and improved scripts for, and proved that the data analysis work on ENGIN-X can be greatly improved and automated with the help of Python and the Mantid API.

Keywords: Engineering Diffraction, Data Analysis, Crystallography

I. INTRODUCTION

Engin-X is a Neutron Diffraction instrument at ISIS, primarily focused on engineering materials and components. This instrument has been running since 2002, and so, it has around 5TB of archived data, varying from regular calibration runs to run of the mill experimentation. There are three main projects I have been working on, utilising historic data for analysis and tests.

Improving Calibration Methods

On ENGIN-X, we perform three main calibrations. A Vanadium run at the beginning of each cycle for the Incident Beam Spectrum, a Ceria run every collimator change for the Peak Profiles, and a Pin Scan every few days – often before an experiment – to locate the centre of the gauge volume. There was very little, if anything, to change or analyse about the Vanadium calibration, it was more of a tool for generating data later on, so most of my work was on the other two calibrations.
For the Pin Scan calibration, we take a small steel pin about a millimetre wide and take neutron measurements as we move it incrementally across the X-axis of the motor controls – perpendicular to the neutron beam. After this, we sum the values of intensities for each measurement, creating a function that resembles a normal distribution. From the location of the peak, we can then determine the point at which the most neutrons passed through the pin, and thus, the centre of the gauge volume. We then repeat this process in the Y-axis – parallel to the neutron beam. In the past, this function has been fitted to a gaussian which is useful enough for locating the gauge volume but not accurate enough for any more information we would like to retrieve from the calibration data.

Whenever the slits – jaws in the metadata – or collimator is changed, a measurement of Ceria powder is performed, as we know the lattice parameter of the sample and can adjust the time-of-flight measurements to the d-spacing accordingly. This experiment, or variations thereupon, has been running since 2002, and so I intended to investigate any notable changes.

Analysing Older Datasets

Polycrystalline structures are made up of various grains with a range of sizes and orientations, fitting together like a sort of crystal jigsaw. When analysed using X-Ray Diffraction, the diffraction ring that arises is very clearly impacted by the size and uniformity of these grains, and there are already papers and work being done to analyse this information[1]. This is not the case with Neutron Diffraction on ENGIN-X, however. The neutrons reflected by different grains are all collected by two detector banks and the data is typically summed before the user gets to see it, thus making any grain size or orientation analysis more difficult.

Multi-run Analysis

Many of the experiments on ENGIN-X are performed over many runs. Take, for example, a furnace experiment where the user would be taking a measurement of a steel sample at increments of 10 K from a range of room temperature to about 500 K. We would expect the peaks to decrease in intensity and increase in width, as the bonds between atoms get

[1] Yager, K. G. & Majewski, P. W. (2014)

larger and the neutrons less likely to deflect. The Load function in Mantid Workspace only retrieves one nexus datafile at a time, so this would mean the user would be performing this analysis peak by peak and run by run either by hand or by excel. In the case of this steel example, that would be ten peaks per twenty runs, amounting to the same process about two hundred times, which could get quite tedious. This is where the Mantid API comes in, working with the algorithms of the Mantid Workbench but allowing the versatility of any other python package.

II. METHOD

Improving Calibration Methods

If you were to take an infinitely long pin and perform a Pin Scan, the sum of the intensity values would increase uniformly before a peak then decrease again in the same fashion, providing a triangle function. The pin used on ENGIN-X is not infinitely long, however, so this triangle function is then convoluted with a semicircle function in order to approximate the shape. Three Pin functions are created – one with the pin's width set to $1mm$, one with the width set to $2mm$, and one with a box function rather than a semicircle – and alongside the original gaussian, these are fitted to the data using Lmfit. Lmfit is a Python Package used for Non-Linear Least-Squares Minimization and Curve-Fitting[2] , and after inputting some initial parameters estimated from the data, it retrieves a residual, which is minimises before providing a final fit.

$$e = y - \hat{y}$$

e is the residual, y is the original data, and $\hat{y}$ is the fitted function.

The Pin Scan script then takes the final residuals and compares them, taking the lowest value to be the best fit and returning this to the user.
The Ceria script first searches the Summary Txt file for every mention of Ceria in ENGIN-X experiments, taking these titles and classifying them by gauge volume and settings, e.g. 4x4x20 Steel Settings. Using the LoadCIF function from Mantid, the script reads a Copper CIF file and uses this alongside the known lattice parameter of 5.41 to determine the location of the peaks in a typical Ceria dataset. Then, for each gauge volume and settings combination, it searches the Summary Txt file to find the corresponding runs, then searches for these runs in the internal servers. For each run, it fits to the peaks found earlier and stores information about them in a json file, one for each gauge volume and settings combination, e.g. a file for 4x4x20 Steel and a file for 4x4x20 Aluminium etc. These json files can then be opened for further analysis.

Analysing Older Datasets

ENGIN-X has two detector banks, North and South, at 90°and 270°to the sample environment respectively, with an additional 10°of coverage either way. Each detector bank is made up of 5 rows of 240 detectors, corresponding to 1200 time-of-flight spectra. In a typical ENGIN-X experiment, the 2400 time-of-flight spectra (1200 from each bank) would be summed into one spectrum where the user would read the data from. In order to analyse the specific grains within the sample, however, we divide the detector banks into 5 rows of 24 groups. Then, taking the spectra from these groups rather than summing them all together, we can analyse the neutrons being reflected from more angles within the 20°range on their own, each encompassing an angle of about 0.83°.
The Grain Size script takes a run then detects and fits peak in the summed spectrum. After doing so, it separates the spectrum by group, detecting and fitting the same peaks in the smaller spectra, storing information such as the height, peak location, width, and integrated intensity for each peak within each spectrum.
Grains within a material can often take on random orientations when they solidify, and this can affect the shape of the peak when read as diffraction data. Taking the integrated intensity, a measure of all the space underneath the peak, of each peak and correlating them against each other, you can get an estimate for how similar the two peaks are. From this, we can determine how closely the two grains may be oriented, and possibly, if any twinning has occurred.
Also in the cooling process, when many nearby grains are oriented in the same direction, they group together, in contrast with the differently oriented grains nearby that they lock into shape with. The larger grains would be reflecting more neutrons into a group of detectors, as this would be more closely aligned with its orientation, and significantly less into another group, whereas a smaller grain nearby would be doing the same to a lesser degree, as it would be reflecting less neutrons, thus the range of intensities recorded across the detector groups for a large grain would be much larger than those recorded from a smaller grain.
To represent this, we collect the integrated intensities collected from each peak and then separate the value by group. We then order this list, regardless of detector

[2] Newville, M., Otten, R. and Stensitzki, T. (no date)

group, and approximate this to a function. The gradient of said function, then, should reflect the range of values collected for each grain. A larger gradient means a larger gradient, and a smaller gradient means a smaller grain.

Multi-run Analysis

On the ISIS network, there are folders for every instrument, containing the raw nexus files and anything else created and stored when the instrument is running. These folders can be accessed via ethernet while on site at Rutherford Appleton Laboratory or using the STFC VPN through Forticlient when connected to Eduroam or the internet elsewhere. This allows access to all of the data ENGIN-X has collected since it began operation in 2002, and the fileSearch function initially written for the Multi-run Analysis script retrieves data from said folders using only a run number.

In cycle 2008-1, the number of digits allowed in a run number changed, so all the titles of the runs did too, e.g. going from ENG99406 to ENG00099406 or ENGINX00099406. The algorithm recognises this and accounts for it. Also, over the course of ENGIN-X's lifespan, the names for certain measured values such as the temperature in a Cryogenics experiment or the motor positions have changed, so the algorithms has counters for this too, and if it can't find the information inside the nexus or raw file, it searches for a txt file with the value saved in the same folder[3].

After checking for these, it goes through all the folders in the ISIS servers, searching for the run number the user has provided, loading the spectra depending on which bank the user requested – Bank One, Bank Two, or both. It then sums these spectra and returns it, alongside the directory and the requested measurements for the experiment – temperature for Cryogenic or Furnace experiments, stress rig position for tensile experiments, and motor positions for Residual Stress scans.

This algorithm is a part of two larger scripts, one for saving the neutron data as is and another which fits and stores the peaks. The first script extracts the x and y values from the nexus or raw datafile, collecting them over each run and storing them as a 3D matrix in a mat file. The second script does the same as the first, extracting the data and saving it, before detecting and fitting peaks to it for each run[4]. It then saves the information about these peaks – location, height, width, background, and integrated intensity – and stores them in a Hierarchical Data Format file (HDF), useful for storing and accessing 3D data such as this.

[3] Table in Appendix Four

[4] More on this in Appendix Three

III. RESULTS

Improving Calibration Methods

The Pin Scan with Lmfit seems to provide the same location for the peak, and thus the same location for the centre of the gauge volume, as the Scipy fit within a range of 0.44%. The other parameter measuring the width is changed by a value of 42.21%. The Pin $1mm$ function is returned over and over again as the best fit, its residual significantly smaller than the gaussian by an order of magnitude of four.

Gaussian	Lmfit
71.55355737	8.76×10^{-3}
1.55299	5.33×10^{14}
105.4175545	6.93×10^{-8}

Table I. Table of the residuals for both the original Scipy fit and the chosen lmfit for the three most recent X Scans.

Gaussian	Lmfit
−2.735488174	−2.734535454
−4.355081289	−4.513140214
−3.736510295	−3.720193578

Table II. Table of the peak locations for both the original Scipy fit and the chosen lmfit for the three most recent X Scans.

Analysing the 220 peak of the collected Ceria data, as this is the most reliably detected and fitted, there would seem to be a trend where the location of the peak seems to be gradually shifting to the left overtime. There is no discernible pattern, however, in the width or heights of the peaks.

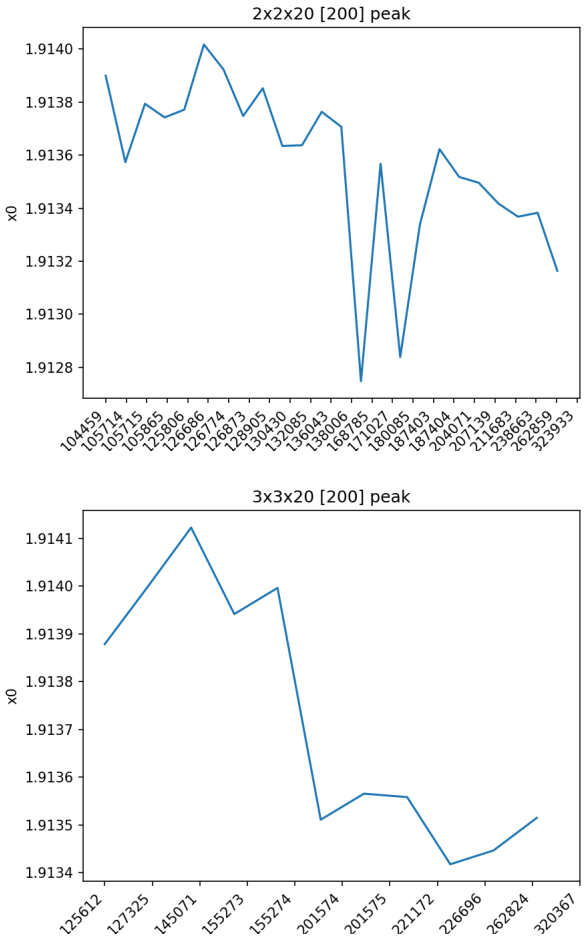
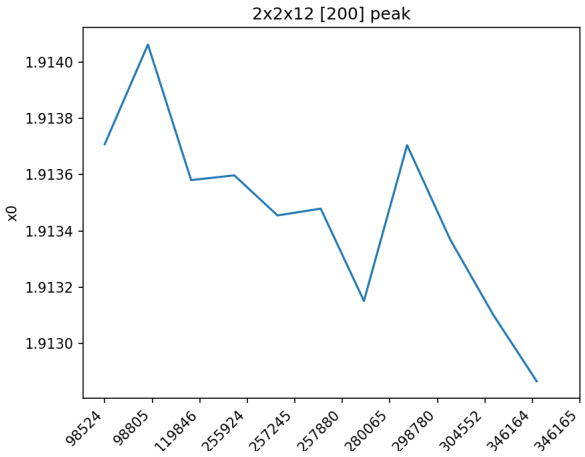


Figure 1. Graphs of the 200 peak location over every Ceria run within a set gauge volume.



Multi-run Analysis

This script is very reliable, allowing the user to extract data from any run performed on ENGIN-X, going all the way back to the first runs in 2002. It can be, however, more unreliable when it comes to reading measurements from the files, especially the further back you go. This could especially be a problem for any historic datasets, where the user may not remember or have on record the measurement being performed. Either way, it works very reliably on datasets within the last decade and a half, providing us with data we can use for plots like these.

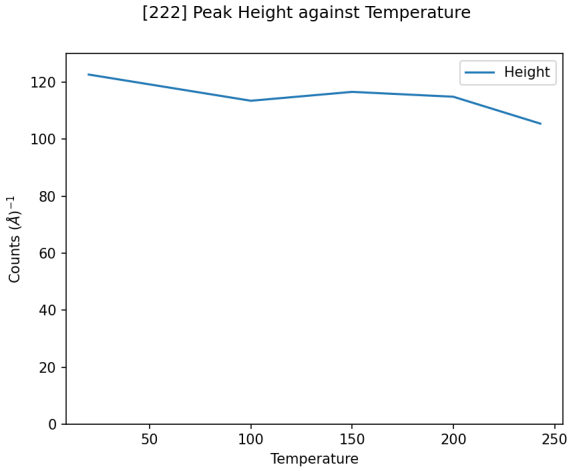
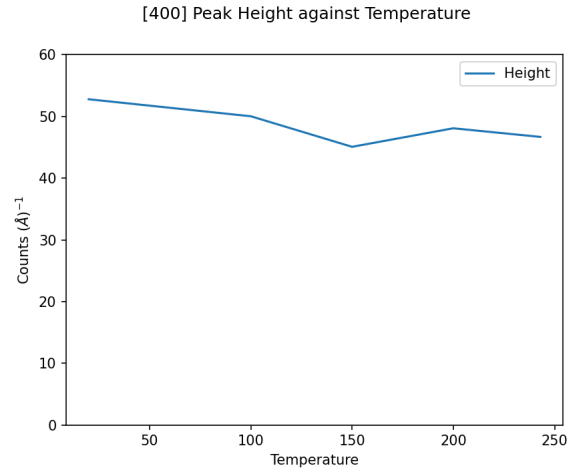


Figure 2. The height of the 222 and 400 peak in an NTTH Cryogenic experiment. As expected, the peak's height is inversely proportional to the temperature.

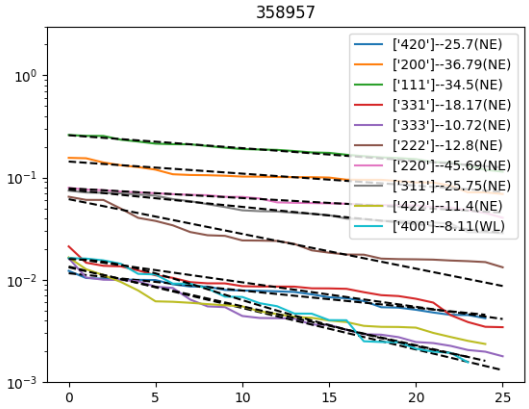


Analysing Older Datasets

As intended, this script allows you to go back and glean information about the grains of samples we may not have considered or noticed at the time, allowing us, too, to compare it with other similar

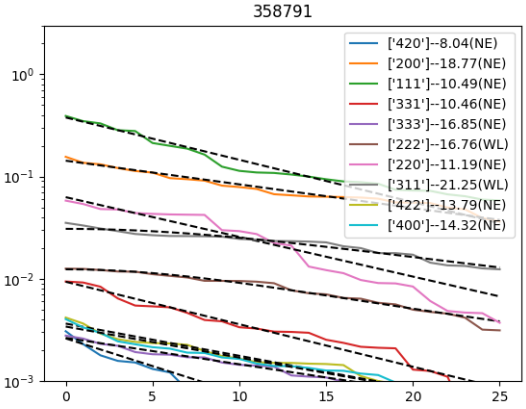
experiments or samples. Take this Copper sample from an Etruscan artefact brought in by Francesco Grazzi.

	['420']	['200']	['111']	['331']	['333']	['222']	['220']	['311']	['422']	['400']
20']	1.0	0.035	0.032	-0.13	-0.09	0.204	-0.06	0.225	-0.034	-0.035
00']	0.035	1.0	0.905	0.442	0.482	-0.147	0.68	0.264	-0.029	0.204
11']	0.032	0.905	1.0	0.284	0.399	-0.175	0.548	0.244	0.026	0.074
31']	-0.13	0.442	0.284	1.0	0.399	0.081	0.204	0.082	-0.046	0.793
33']	-0.09	0.482	0.399	0.399	1.0	0.135	0.282	0.203	0.132	0.178
22']	0.204	-0.147	-0.175	0.081	0.135	1.0	-0.266	-0.007	-0.141	0.306
20']	-0.06	0.68	0.548	0.204	0.282	-0.266	1.0	0.496	-0.037	-0.017
11']	0.225	0.264	0.244	0.082	0.203	-0.007	0.496	1.0	-0.094	-0.04
22']	-0.034	-0.029	0.026	-0.046	0.132	-0.141	-0.037	-0.094	1.0	-0.035
00']	-0.035	0.204	0.074	0.793	0.178	0.306	-0.017	-0.04	-0.035	1.0



From the Orientation Relation table, we can tell there is quite a strong correlation between the 111 and 200 peak, and similar but lesser matches with the 331-400 and the 200-220 pairs. There also seems to be quite a large range in the sizes of the grains. This becomes more interesting when we compare it to a more recently constructed sample brought in by Bristol University.

	['420']	['200']	['111']	['331']	['333']	['222']	['220']	['311']	['422']	['400']
20']	1.0	-0.068	-0.13	0.175	-0.097	0.072	0.283	0.037	-0.212	-0.044
00']	-0.068	1.0	0.82	-0.073	0.056	0.657	-0.355	0.26	0.246	-0.165
11']	-0.13	0.82	1.0	-0.047	-0.031	0.519	-0.451	0.124	0.202	-0.183
31']	0.175	-0.073	-0.047	1.0	-0.057	-0.065	0.512	0.469	0.033	-0.157
33']	-0.097	0.056	-0.031	-0.057	1.0	0.18	0.074	0.475	0.182	0.062
22']	0.072	0.657	0.519	-0.065	0.18	1.0	-0.414	0.413	0.094	-0.086
20']	0.283	-0.355	-0.451	0.512	0.074	-0.414	1.0	0.365	0.164	-0.07
11']	0.037	0.26	0.124	0.469	0.475	0.413	0.365	1.0	0.325	-0.152
22']	-0.212	0.246	0.202	0.033	0.182	0.094	0.164	0.325	1.0	0.202
00']	-0.044	-0.165	-0.183	-0.157	0.062	-0.086	-0.07	-0.152	0.202	1.0



These samples are both Copper-based, however, the correlations found in the Etruscan artefact's peaks tends to be higher than those is the sample from Bristol, implying a higher uniformity of the grains. This may be due to their size, Grazzi's sample seeming to have larger grains and a larger range of sizes, perhaps a result of different manufacturing methods over time.

IV. CONCLUSIONS

It should be a relief that the work on the calibration methods has shown them to already be quite reliable. The margin of error on the Pin Scan is potentially negligible, and the effects on the Ceria overtime is noticeable but not detrimental – though perhaps this could be something to consider for ENGIN-X's use in the long-term.

It has been proven that reasonable and discernible information can be retrieved about the grains of a material from separating the detector banks into smaller angles. This, however, may need to be corroborated with experimentation – perhaps a sample of known texture – before presenting as a valid method going forward.

The multi-run analysis could prove to be a very useful tool, and goes to show, as all these scripts do, that there is great potential in the use of the Mantid API and Jupyter extensions, similar to how one may use any other Python package. Perhaps a greater push or focus on this may be useful, improving the documentation and making the package more accessible to those perhaps outside of ISIS and its immediate vicinity. There are many algorithms found within Mantid that could potentially be useful for many use cases – ones that are simply unavailable or not as useful in other packages.

ACKNOWLEDGEMENTS

The Mantid Team for their work creating the foundation on which these scripts could be written and this analysis performed. The STFC Industrial Placement Scheme for the opportunity to work on this project.

[1] Yager, K. G. & Majewski, P. W. (2014), *Metrics of graininess: robust quantification of grain count from the non-uniformity of scattering rings.*. J. Appl. Cryst. 47, 1855-1865.

[2] Newville, M., Otten, R. and Stensitzki, T. (No Date), *Non-linear least-squares minimization and curve-fitting for python¶, Non-Linear Least-Squares Minimization and Curve-Fitting for Python - Non-Linear Least-Squares Minimization and Curve-Fitting for Python.* Available at: https://lmfit.github.io/lmfit-py/ (Accessed: 24 July 2025).

[3] Hall, S.R., Allen, F.H. and Brown, I.D. (1991), *Commission on Crystallographic Data Commission on Journals Working Party on Crystallographic Information , The Crystallographic Information File (CIF): a New Standard Archive File for Crystallography.* Available at: https://journals.iucr.org/a/issues/1991/06/00/es0164/es0164.pdf (Accessed: 26 July 2025).

[4] Mantid Team (No Date), *LoadCIF v1.* Available at: https://docs.mantidproject.org/v3.7.2/algorithms/LoadCIF-v1.html (Accessed: 27 July 2025).

[5] Luo, W. (2019), *structf.py, GitHub.* Available at: https://github.com/retornado/TAS_Simultor/blob/master/Structf.py(Accessed: 26 July 2025).

[6] Hansen, A.C. (2016), *Numerical Analysis - Part II, Numerical Analysis - Part II: Spectral Methods.* Available at: http://www.damtp.cam.ac.uk/research/afha/lectures/Part_II_NumAn/Lect_15_slides.pdf (Accessed: 26 July 2025).

[7] Mantid Team (No Date), *Gaussian* Available at: https://docs.mantidproject.org/v6.3.0/fitting/ fitfunctions/Gaussian.html (Accessed: 27 July 2025).

[8] Hutchings, M.T., Withers, P.J., Holden, T.M., & Lorentzen, T. (2005), *Introduction to the Characterization of Residual Stress by Neutron Diffraction (1st ed.).* CRC Press. https://doi.org/10.1201/9780203402818

Appendix: Appendix One: CIF Data Generation

For one of the Peak Fitting methods, a known dataset had to be used as a comparison, so it was deemed useful to generate our own. This function uses Crystallographic Information Files or CIFs, a text file format developed by the Internation Union of Crystallography (IUCr) to represent crystallographic information[5]. LoadCIF is a Mantid algorithm which takes a CIF provided by the user and saves it to a workspace[6]. From here, the user can glean information about the sample such as its crystal structure. This is how we obtain the value for the lattice parameter, a, and the hkl reflection of each peak. The LoadCIF function also generates values for the peaks in d-spacing and F squared[7]. The width of each peak is proportional to its location in d-space, and the intensity is calculated as:

$$I = \phi \frac{j F_{hkl}^2 d_{hkl}^4}{a^3} 8$$

Where ϕ is the flux of the incident beam derived from the value of the V-Nb data in the peak's location, F is the structure factor, j is the symmetry equivalents of the peak's reflection, d is the location of the peak in d-spacing, and a is the lattice parameter of the material. This ignores parameters dependent on the experiment being performed, such as gauge volume and flight path. After this, some random noise is created to make the data appear more realistic. This is not a perfect approximation to real-life neutron data, but it's very useful for the comparisons we wanted to perform – all that was necessary was the peak locations and relative heights.

Appendix: Appendix Two: Chebyshev Backgrounds

The Chebyshev Polynomial of degree n is defined as:

$$T_n(x) := cos(n) arcos(x), x \in [-1, 1]^9$$

Being derived from the cosine and sine equations, this polynomial became a very useful tool for approximating the backgrounds of neutron data. Numpy has a Chebyshev Polynomial class which can be fitted to a function and, when the parameters are low enough, it can register the background of a dataset rather than the data itself. We noticed when the sum of the coefficients of the Chebyshev approached zero, the fit became

[5] Hall, S.R., Allen, F.H. and Brown, I.D. (1991)

[6] Mantid Team (No Date)

[7] Luo, W. (2019)

[8] Hutchings, M.T., Withers, P.J., Holden, T.M., Lorentzen, T. (2005)

[9] Hansen, A.C. (2016)

more accurate, so the background fitting function aims to minimize that value, returning the closest within a certain range. This function can then be subtracted from the data itself to reduce background in a plot or added to a peak fit to provide information on the background below the peaks.

Appendix: Appendix Three: Peak Fitting Functions

There are three peak fitting functions used in the 3D data script, only two of which are copied over to the Grain Size Analysis script – Cubic and Manual.
The first function uses Scipy Find Peaks to detect the peaks with the parameters set to threshold at 5.5, prominence at 9.5, distance at 100, and the rest to default. After locating the peaks, the width of each peak is calculated by approximating the Full Width Half Maximum (FWHM) and rearranging an equation found in the Mantid documentation[10]. If this fails for whatever reason, the width is then proportional to the peak's location in the d-space. After this, a background is applied and these initial parameters are put through Lmfit, which minimizes the residual and returns new parameters. This method is often quite unreliable, though, as the Find Peaks function is often very temperamental and inconsistent when it comes to peak detection, especially with noisy data, and Lmfit has a tendency to collect and move peaks whenever there is a significant background to the data.
The second function creates a Copper and Vanadium set for comparison against Face Centred Cubic (FCC) and Body Centred Cubic (BCC) crystals respectively. It then takes these alongside the neutron data and logs the x-values before normalising the y-values. When comparing two elements of the same crystal structure in Neutron Diffraction data, it can seem almost like one has been stretched in order to create the other, so performing a log on the x-values changes this stretch to a shift, making it a lot easier to compare, and the normalising is to make things a lot clearer. Then the first peak from the right is detected – 111 or 110 for FCC and BCC respectively – and used to estimate a potential 200 peak for each crystal. When two datasets of the same crystal are correlated against each other, there should be a prominent peak where the two align the most, whereas if the datasets are of different crystals, this peak should be harder to discern. We use this to estimate the crystal, and if the 200 peak is found where it would be for said crystal, the function determines the locations of the other peaks. After doing so, it estimates the width and background like earlier then puts this through PlotPeakByLogValue, a Mantid algorithm, as this function is much more rigid with the peak locations.
The final function takes an input of peak locations from the user. After making sure the locations or an approximate are found within the data, it estimates the crystal structure based on the ratio between the first and second peak from the right. It then uses this to estimate the hkls of each peak, before passing this information through the width estimation mentioned earlier, and PlotPeakByLogValue.

Appendix: Appendix Five

Figure 3. Table of the measurements stored in nexus/raw files and the names they have been stored as over the years.

Parameter	2024	2023	2022	2021	2020
Cryo	Temp_1	Temp_1	Temp_1	Temp_1	Temp_1
Furnace	Temp_3	Temp_3	Temp_3	Temp_3	Temp_3
Pressure	position	position	position	position	position
XScan	X_position	X_position	X_position	X_position	X_position
YScan	Y_position	Y_position	Y_position	Y_position	Y_position
ZScan	Z_position	Z_position	Z_position	Z_position	Z_position
Filetype	nxs	nxs	nxs	nxs	nxs

Parameter	2019	2018	2017	2016	2015	2014	2013	2012	2011	2010
Cryo	Temp_1	Temp_1	Temp_1	cryo_temp1	cryo_temp1	cryo_temp1	cryo_temp1	cryo_ten	cryo_temp1	cryo_temp1
Furnace	Temp_3	Temp_3	furnace	furnace	furnace	furnace	furnace	furnace	furnace	furnace
Pressure	position	position	No info	No info	No info	No info	No info	No info	No info	No info
XScan	X_position	X_position	x	x	x	x	x	x	x	x
YScan	Y_position	Y_position	y	y	y	y	y	y	y	y
ZScan	Z_position	Z_position	z	z	z	z	z	z	z	z
Filetype	nxs	nxs	nxs	nxs	nxs	nxs	nxs	raw	raw	raw

Parameter	2009	2008	2007	2006	2005	2004	2003	2002
Cryo	cryo_1	No info	No info	No info	No info	No info	No info	No info
Furnace	furnace	No info	No info	No info	No info	No info	No info	No info
Pressure	No info	No info	No info	No info	No info	No info	No info	No info
XScan	No info	No info	No info	No info	No info	No info	No info	No info
YScan	No info	No info	No info	No info	No info	No info	No info	No info
ZScan	No info	No info	No info	No info	No info	No info	No info	No info
Filetype	raw	raw	raw	raw	raw	raw	raw	raw

[10] Mantid Team (No Date)