Identifying lipid mesophases from a set of Small-Angle X-Ray Scattering (SAXS) peaks

Christopher Brasnett[1] and Annela Seddon[1,2]

[1]H.H. Wills Physics Laboratory, Tyndall Avenue, University of Bristol, Bristol BS8 1FD

[2]Bristol Centre for Functional Nanomaterials, HH Wills Physics Laboratory, Tyndall Avenue, University of Bristol, Bristol BS8 1FD

June 1, 2018

Abstract

A statistical method of identifying lipid mesophases from Small-Angle X-ray Scattering (SAXS), is described, having been implemented in python code at https://github.com/csbrasnett/lipidsaxs

CSB note: this document has been used for various iterations of the code, and there may be some internal discrepancies between the peak finding and phase ID sections relating to past versions, which could cause some level of confusion to arise. I have done my best to eliminate as much of this as possible, but no doubt I have not caught everything. If you find something or would like more explanation, please don't hesitate to get in touch!

1 Introduction

High-throughput methods of SAXS to identiying the mesophase behaviour of aggregated lipid systems necessarily produce a lot of data. In general - or at least most often - the SAXS patterns such systems produce belong to one of five mesophases: Lamellar, Inverse Hexagonal, and the Primitive, Diamond, and Gyroid cubic phases. They are distinguished through the characteristic spacings of the Bragg peaks present in the SAXS patterns [1]. A consequence of the combination of subtle differences in the characteristic peak spacings, combined with rigorously identifying peaks in 1D azimuthally-integrated SAXS data is that the initial analysis of lipid mesophase data is both time-consuming, and repetitive. The methods described in this paper have been developed and implemented so that less time has to be spent analysing phase behaviour of SAXS data, and more time can be spent on designing experiments.

The Github repository contains 3 python scripts: *finder.py*, *phase_ID.py*, and *complete.py*. *finder* is responsible for the finding and fitting of Bragg peaks in 1D I vs. q SAXS data, whilst *phase_ID* tries to assign a phase to the collection of Bragg peaks found. *complete* is an example script calling the others in order to demonstrate the type of results that can be expected from the analysis. *complete.py* contains some example parameters, and only requires a folder of text files (NB: check the file extension for the search - do you have csv or txt files?) of I vs q data in order to run. The exact description of the parameters is contained in this document.

2 Manual step-by-step guide

The key to establishing the mesophase(s) of a single measurement is to analyse the relative spacings of Bragg peaks in the data. Covering the fundamentals of X-ray scattering is beyond the scope of this report, and readers requiring an introduction to the topic are referred the book by D. S. Sivia [2]. However, it is useful to note that scattering patterns are reciprocal space measurements, so the units of measurement are in 1/length. The 'raw' detector image will likely be a series of concentric rings, as a result of scattering from billions of domains in the sample, all orientated differently,

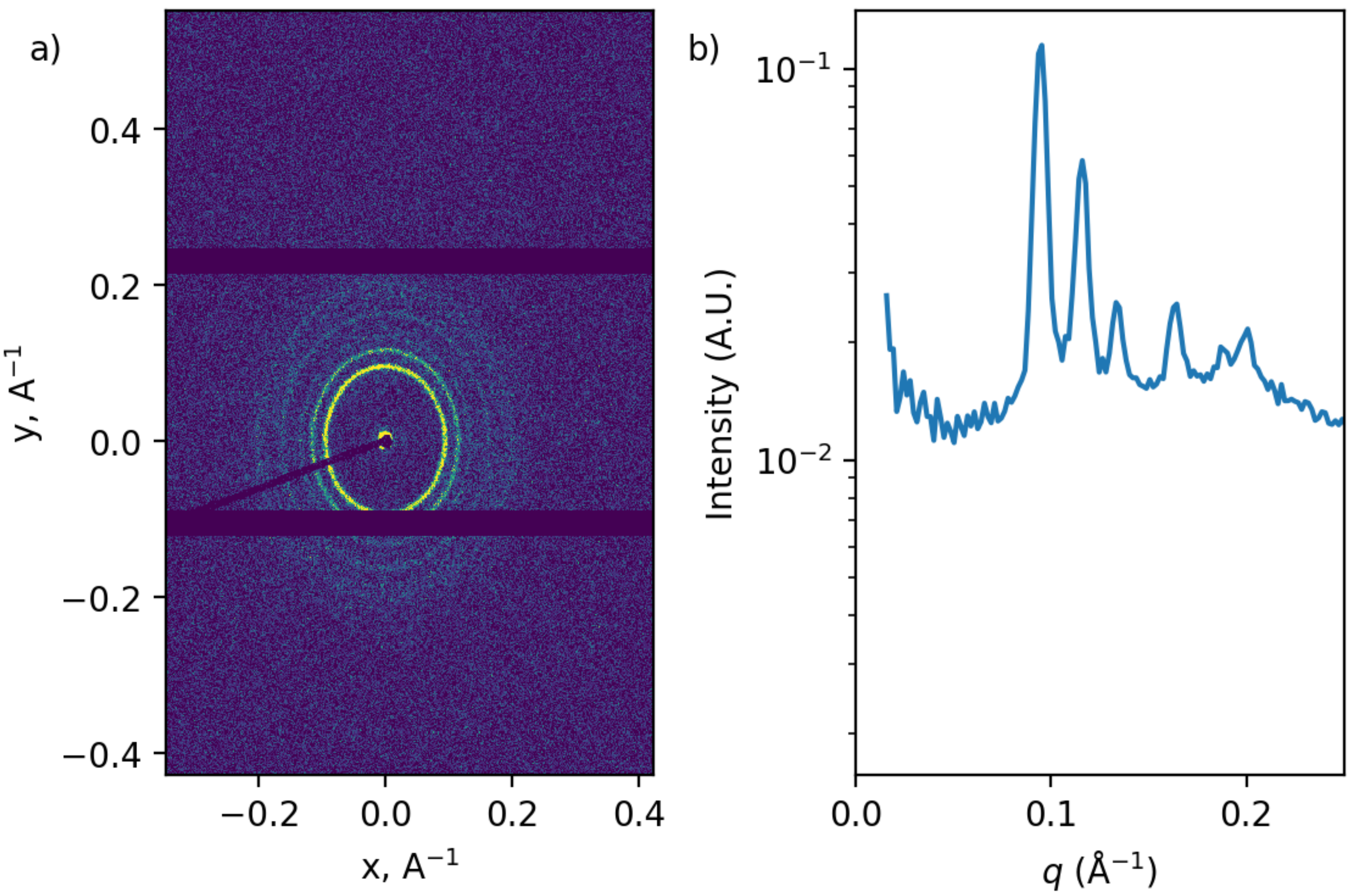


Figure 1: a) The 2D raw detector image and b) The corresponding 1D azimuthally integrated reduced scattering pattterns of an example Q_{II}^{D} phase.

such as that seen in Fig. 1 a). In the case when the sample has been oriented though various methods (see, for example, refs. [3–5]), the rings will become spots, as there is only scattering from a single domain.

The 2d detector image is integrated azimuthally, so that the data is reduced to the so-called 1D form of $I(q)$ vs. q[1]. An example of such an azimuthal reduction is seen in Fig. 1 b).

Once the scattering pattern has been reduced to its 1D form, the next stage is to find the positions of the Bragg peaks. The peak positions can be found using and peak finding and fitting routine[2]. For example, using the inbuilt Pearson VII fitting function in SAXSGUI on the data in Fig. 1, the peak positions are found at 0.094527, 0.11605, 0.134, 0.16345, 0.18819, and 0.2 Å^{-1}.

Having found the position of the peaks in the scattering pattern, the mesophase should be identified. As previously described, the mesophase from which the pattern originated is most probably one of the either the three phases of cubic translational symmetry, a hexagonal phase, or a lamellar phase, which are distinguished by their characteristic peak spacings. The characteristic spacings are as so:

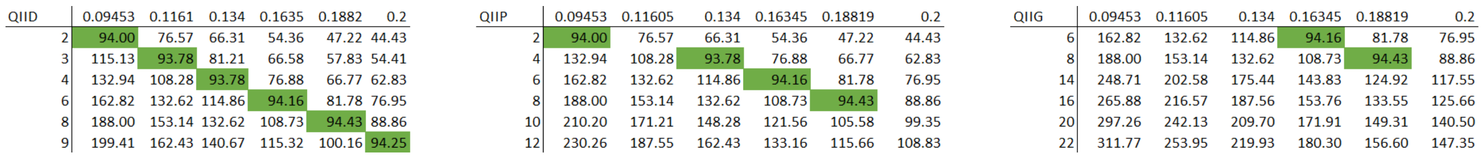
Lamellar (L_α) : 1,2,3

Inverse Hexagonal (H_{II}) : 1, $\sqrt{3}$, $\sqrt{4}$

Primitive Cubic (Q_{II}^{P}) : $\sqrt{2}$, $\sqrt{4}$, $\sqrt{6}$, $\sqrt{8}$, $\sqrt{10}$, $\sqrt{12}$, $\sqrt{14}$

[1]The q (scattering vector) scale will depend on the sample-detector distance, and so is calibrated against calibrants of known lattice spacings. The Bristol SAXSLAB Ganesha system uses Silver Behenate for calibration.

[2]Originlab's in built functions should suffice for these purposes.



QIID	0.09453	0.1161	0.134	0.1635	0.1882	0.2
2	94.00	76.57	66.31	54.36	47.22	44.43
3	115.13	93.78	81.21	66.58	57.83	54.41
4	132.94	108.28	93.78	76.88	66.77	62.83
6	162.82	132.62	114.86	94.16	81.78	76.95
8	188.00	153.14	132.62	108.73	94.43	88.86
9	199.41	162.43	140.67	115.32	100.16	94.25

QIIP	0.09453	0.11605	0.134	0.16345	0.18819	0.2
2	94.00	76.57	66.31	54.36	47.22	44.43
4	132.94	108.28	93.78	76.88	66.77	62.83
6	162.82	132.62	114.86	94.16	81.78	76.95
8	188.00	153.14	132.62	108.73	94.43	88.86
10	210.20	171.21	148.28	121.56	105.58	99.35
12	230.26	187.55	162.43	133.16	115.66	108.83

QIIG	0.09453	0.11605	0.134	0.16345	0.18819	0.2
6	162.82	132.62	114.86	94.16	81.78	76.95
8	188.00	153.14	132.62	108.73	94.43	88.86
14	248.71	202.58	175.44	143.83	124.92	117.55
16	265.88	216.57	187.56	153.76	133.55	125.66
20	297.26	242.13	209.70	171.91	149.31	140.50
22	311.77	253.95	219.93	180.30	156.60	147.35

Figure 2: Exploring the possible cubic indexing of the peaks from Fig. 1 in the Q_{II}^{D}, Q_{II}^{P}, and Q_{II}^{G} phases. The calculation in each grid explores all possibilities of indexing every peak with every factor of each phase, calculated using the equations in the main text. The correct indexing of the peaks have been highlighted in Green. As a result of factors appearing in multiple phases, the 'correct' indexing of each peak appears multiple times across phases.

Diamond Cubic ($\mathbf{Q}_{II}^{D}$) : $\sqrt{2}$, $\sqrt{3}$, $\sqrt{4}$, $\sqrt{6}$, $\sqrt{8}$, $\sqrt{9}$, $\sqrt{10}$, $\sqrt{11}$

Gyroid Cubic ($\mathbf{Q}_{II}^{G}$) : $\sqrt{6}$, $\sqrt{8}$, $\sqrt{14}$, $\sqrt{16}$, $\sqrt{20}$, $\sqrt{22}$, $\sqrt{24}$

If a set of peaks can be said to belong to a particular mesophase, then the lattice parameter (that is, repeat unit cell spacing) of the mesophase can be subsequently calculated:

Lamellar : $\mathrm{a}_{[hkl]}=\frac{2\pi}{q_{[hkl]}}h$

Inverse Hexagonal $\mathrm{a}_{[hk]}=\frac{2}{\sqrt{3}}\frac{\sqrt{h^2+k^2-hk}}{q_{hk}}$

Cubic phases $\mathrm{a}_{[hkl]}=\frac{\sqrt{h^2+k^2+k^2}}{q_{[hkl]}}$

Indeed, most easily, a mesophase can be identified by finding a set of peaks which, when used for the above calculations, and indexed correctly, produce the same value. The same value that the calculations produce is the lattice parameter of the phase observed: a measure of the real space physical periodicity of the phase.

An example of the type of calculation that needs to be done is seen in Fig. 2, where the entire set of possible lattice parameters has been explored by assigning every peak previously found from Fig. 1 b) to every possible index in each of the three cubic phases. In Fig 2, the values in the table equate to calculating:

$$\mathrm{a}_{[hkl]}=\frac{\sqrt{h^2+k^2+k^2}}{q_{[hkl]}}$$

for each of the three cubic phases. We can see that the most matches have been made in the case of the Q_{II}^{D} phase assignment, and the matches found in the cases of the Q_{II}^{P} and Q_{II}^{G} assignments are degenerate with the Q_{II}^{D} one. Therefore, these peaks should be assigned to that phase. With no more peaks outstanding, the work for this file is complete.

At this point, there may still be many more files in which to find the position of the Bragg peaks, and assign them to a mesophase, which can be a daunting and boring task. The rest of this document explains the automation process developed.

3 Peak identification

AS just discussed, the first stage of identifying the mesophase behaviour of lipid systems is finding the scattering vector (position in q) of the Bragg peaks of the scattering pattern. The peak positions are found using the script *finder.py*.The programme has two functions, finder and fitting. The variables for the fitting function are entirely determined by the routine of the finder.*finder.py* has four mandatory variables, and two optional (but really mandatory) ones. The four absolute mandatory ones are:

file_name : The location of a text file of the q and Intensity data, formatted into two columns, the first of which is the q data, and the second the I(q).

lower_limit, upper_limit : Cut offs in values of q for the q range to be searched for peaks, and, later, for confirming permitted mesophase assignments.

min_sep : The width of the histogram bins for the collection of the peak data (see later).

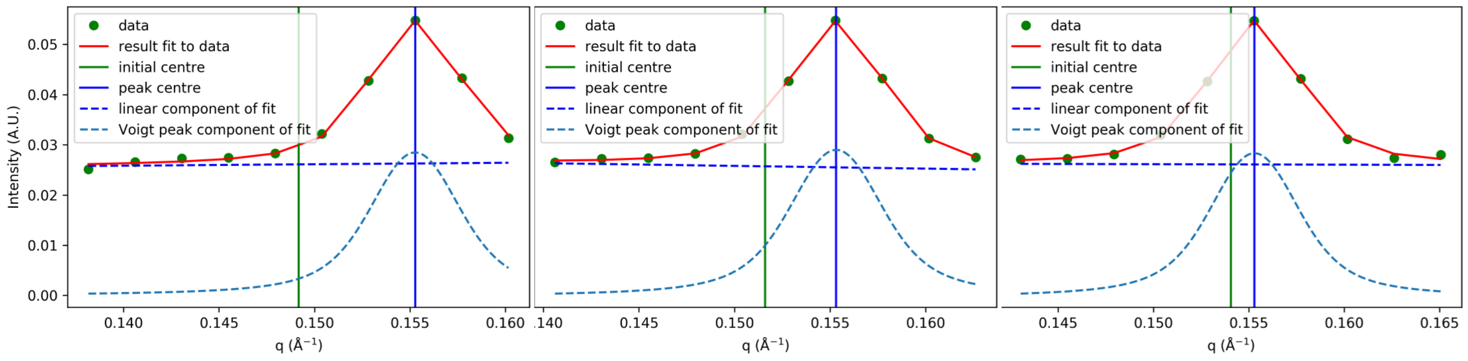


Figure 3: Continuous window fitting to a peak in SAXS data. The same peak is present in all of the frames tested here, with a fit of a convoluted Voigt peak with linear background tried on the data. The window of data points given to fit over has changed between, so the initial guess of the peak position has changed correspondingly at each point. The resultant fit is then also shown.

The two 'optional' parameters are set to *False* by default. Only one needs to be passed, and if neither are, then an error will be raised instead. The optional parameters are:

1. Ganesha
2. DLS

Ie. has the data been collected in-house in Bristol, or at Diamond? Set 'True' as appropriate. Depending on which one is set to 'True', the programme will then adjust variables to find peaks in an appropriate manner. The key parameter that will actually affect the number of peaks found is called *ht_ threshold*, and relates to the height of the peak fitted to the data. Synchrotron sources will naturally be brighter, so the threshold for peaks to overcome is higher. Note that whilst this variable has been fairly extensively tested, it is possible that if peaks appear to be 'missing' by inspection of a I(q) vs. q plot, then it may be worthwhile adjusting these parameters downwards to see if a difference is made.

The peaks will be found be a continuous window search through the q range provided. That is, throughout the q range provided to the programme, an incremental range of the data are fitted with a peak model. The fitting is performed using the lmfit library [6]. lmfit is not installed as a default library on some common distributions of python (eg. Anaconda), so should be installed or checked for installation before the scripts are used.

The peak and background model used is a convolution of a Voigt peak model, and a linear background, as demonstrated in Fig. 3. The peak model has been found not to work well on its own, due to the fact that the form factor of the SAXS pattern means there is a significant background aspect to the data. Additionally to note is the fact that the gamma parameter of the Voigt peak fit is set to equal the sigma parameter by default. These parameters are used to determine aspects of the peak width in the fit of the data. They can, however, be set to not equal each other to improve the fit, but extensive testing of the free variation of the gamma parameter has not proved fruitful in improving the fitting of SAXS data. Most commonly, it has resulted in the centre of the peak being found very far away from where the true centre is. When the free variation of the gamma parameter is turned off, the data are usually found to fit very well. For more information of the fit, the lmfit results table can be returned at each instance of fitting, if so desired, by changing the value of the optional 'plot' variable as True. The fitting function then returns where the centre of the peak has been found to be, along with information about the width and height of the peak, in case this data might be useful.

Figure 3 also demonstrates the continuous window fitting procedure. The data are incrementally selected throughout the permitted q range, and a fit is attempted. The *height_ threshold* parameter is used at this point to determine whether the fit is appropriate to the data: has a 'real' peak been found? All these peaks are returned back to the finder function, and are binned into bins of width determined by the *minimum_ separation* parameter.

4 Phase Identification

The 5 most common mesophases described earlier have characteristic peak spacings like so:

Lamellar (L_α) : 1,2,3

Inverse Hexagonal (H_{II}) : 1, $\sqrt{3}$, $\sqrt{4}$

Primitive Cubic (Q^P_{II}) : $\sqrt{2}$, $\sqrt{4}$, $\sqrt{6}$, $\sqrt{8}$, $\sqrt{10}$, $\sqrt{12}$, $\sqrt{14}$

Diamond Cubic (Q^D_{II}) : $\sqrt{2}$, $\sqrt{3}$, $\sqrt{4}$, $\sqrt{6}$, $\sqrt{8}$, $\sqrt{9}$, $\sqrt{10}$, $\sqrt{11}$

Gyroid Cubic (Q^G_{II}) : $\sqrt{6}$, $\sqrt{8}$, $\sqrt{14}$, $\sqrt{16}$, $\sqrt{20}$, $\sqrt{22}$, $\sqrt{24}$

If a set of peaks can be said to belong to a particular mesophase, then the lattice parameter (that is, repeat unit cell spacing) of the mesophase can be subsequently calculated:

Lamellar : $a_{[hkl]} = \frac{2\pi}{q_{[hkl]}}$

Inverse Hexagonal $a_{[hk]} = \frac{2}{\sqrt{3}} \frac{\sqrt{h^2+k^2-hk}}{q_{hk}}$

Cubic phases $a_{[hkl]} = \frac{\sqrt{h^2+k^2+k^2}}{q_{[hkl]}}$

Indeed, most easily, a mesophase can be identified by finding a set of peaks which, when used for the above calculations, and indexed correctly, produce the same value. However, and especially in the case of coexisting systems, this is a laborious method to tackle the identification problem. It is therefore useful to have a process of automatically indexing peaks to identify the mesophase.

Two facts are useful to note at this point:

- The cubic phases more readily scatter to higher-order Bragg peaks, and thus it is useful to know the characteristic ratios to higher order
- Within the characteristic ratios of the cubic phases, there is a good degree of overlap with regards to the Miller indicies that peaks may be assigned.

With respect to these factors, the first step that is taken is to discriminate the possibilities of the phase to assign to any set of peaks by the number of peaks provided to the phase_ID.py programme. If 3 or fewer peaks have been identified, then it is more likely that the data are from a L_α or H_{II} phase than a cubic one[3]. As noted previously, distinguishing between cubic mesophases is a greater challenge than other possibilities, and so the method used for clarification there will be described first.

The entire *phase_ID.py* script can be called from the main function. main expects two variables:

peaks : An array of peak positions

lo_q : The same lo_q value previously used to define the region in which to search for Bragg peaks.

main firstly discriminates what kind of phase to search for by the number of peaks that have been passed to it. This is run in a *while* loop. The *while* loop accounts for the possibility of cubic/non-cubic phase coexistence. If such a situation has arisen, then a greater number of peaks will have been found, of which a subset may be successfully assigned as cubic, but leaving the non-cubic ones. Once the number of peaks is used to determine which sort of phases to search for, the separate functions are called, whose variables and methods are detailed in the following sections. Subsequently, if the number of peaks left unassigned through this search is still greater than 1, the remaining peaks are re-investigated, until either all peaks have been assigned, or until the *while* loop has been run 10 times. Ultimately, the main programme returns a dictionary of the phases found, along with any unassigned peaks.

[3]Of course from single crystalline mesophase structures, such as those of the Kim and coworkers [7], there is the possibility of systematic Bragg extinctions, resulting in fewer peaks present anyway

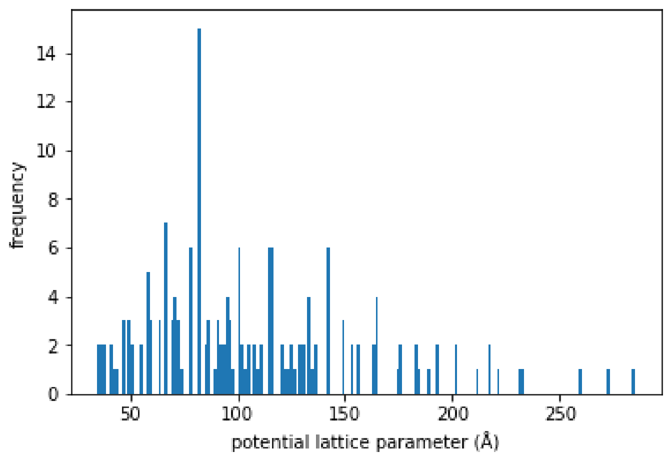


Figure 4: A histogram of all possible lattice constants of a set of input peaks, as a result of indexing every peak with every possible index. There are higher peaks where the same lattice constant has been found more often: one of these will be the correct indexing of the peaks provided.

4.1 The cubic phases

(NB for this section: if a programme variable is described as ‘x_’, x is being used to describe the set of identical variables at the same stage of the method to separate the P, D, and G variables.)

The function Q_main is used to identify cubic mesophases from the SAXS Bragg peaks found. The Q_main function runs two other functions: Q_possible_phases, and Q_projection_testing, both of whose variables are passed from the Q_main. Q_main expects four variables:

peaks : The array of peaks found in the data, from as determined by the

bin_factor : A factor to determine the widths of the bins in a historgram used for finding lattice parameters of cubic phases. Doesn’t have to be big, 2 should suffice.

threshold : A value for which a histogram bin has to be populated more than for confirming that a phase is valid.

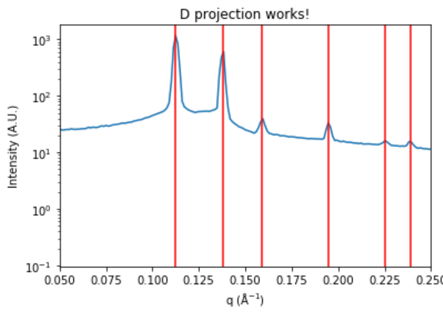
lo_q : should be the same as the lo_q value in the finder.py.

Q_main begins by passing all the **peaks**, **bin_factor**, and **threshold** variables to the Q_possible_phases function. Q_possible_phases begins exploring the entire range of possible lattice parameters that could be found from the data, by creating x_init: a set of multidimensional arrays which simultaneously index all peaks as all x indices. These arrays are then concatenated together. Additionally, n_x are created, along with n. These are arrays of integers running the length of the individual arrays, and the concatenated array. They are later used in order to track back where values in the x_init have originated from: the specific peak and index from which the value arose.

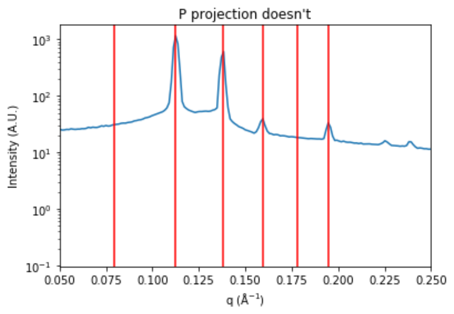
At this point, the frequency of the possible lattice parameters are examined by a histogram: this is where the variable **bin_factor** is used. **bin_factor** is used to determine the widths of the bins, which is defined as the product of the number of possible values and the variable. A value of 2 should suffice for this purpose. As Fig. 4 demonstrates, when a peak has been correctly indexed to calculate the lattice parameter, there are more peaks present in the bin, and therefore the frequency count is higher. It is for this reason a **threshold** variable is passed to Q_possible_phases: if the number of potential lattice parameters in a bin is lower than the threshold, then the bin is discarded, as the indexing has not been correct.

If the number of values in the bin exceeds the **threshold** parameter, then the contents of the bin are examined further. The factors contributing to the height of the bin are traced back to the phase array, peak, and indexing from which they arose. That is to say, across the three cubic phases, it is inevitable that the ‘correct’ index has been used for a peak: as noted previously, the factors $\sqrt{6}$ and $\sqrt{8}$ appear in all three cubic phase indexings, and consequently, regardless of whichever cubic phase the set of peaks has arisen from, they will both make a ‘correct’ contribution.

The individual factors are collected separately. At this point, the number of contributing correct indices is used as the comparison for which phase it is most likely to have arisen from. As a result



(a) The projected peaks from a Q_{II}^D phase proposed by Q_possible_phases match those found in the data. Note here that for this example, the $\sqrt{4}$ peak was not actually found by the finder.py module, and nor has it been found by the projection shown in this figure, which must be taken care of when considering the acceptable threshold of missing peaks.



(b) The projected peaks of a Q_{II}^P phase fall into gaps in the data, and do not index the final two peaks at all.

Figure 5: Projections of peaks from the proposed phases by the Q_possible_phases function across the data being tested.

of the high coincidence between the factors of the Q_{II}^D and Q_{II}^P phases, they are examined slightly separately. Firstly, it is assumed that 4 peaks is the minimum number required to assign a phase by this point, for any phase. Subsequently, if this condition is met by both the Q_{II}^D and Q_{II}^P contributing factors, they are examined. It would be expected that:

1. The correct assignment of the peaks contains more correct peak indices.
2. The incorrect assignment of index and peak pairs is entirely contained in the correct assignment.

These factors are tested for, and the correct assignment between the Q_{II}^D and Q_{II}^P phases is made based on which conditions are met.

The function described above, Q_possible_phases, returns a dictionary of possible phase assignments back to the Q_main function of the *phase_ID* script. At this point, another check is made, cross referencing the set of assigned peaks with the peaks actually present in the data, by the Q_projection_testing function. The function firstly generates a set of peaks based on the lattice parameter and assigned phase, as worked out from the first Q_possible_phases routine. The function then firstly checks that the first peak of this generated set of peaks falls within the user-determined q search range as initially defined. Secondarily, the function cross references the generated set of peaks (all or most of which will be completely coincidental with the 'real' peaks in the data) with the peaks present in the data.

That is, there are two sets of peaks to consider: one from the actual data itself, and a set of peaks that can be generated by knowing both the phase and lattice parameter of an assigned set of peaks. These latter set of peaks are tested against the peaks that have been similarly assigned to the phase. By identifying where the generated peaks match up to the data, the number of peaks that have been correctly assigned to the phase can be identified. This is apparent in Fig 5, where it can be seen the there is a single Q_{II}^D phase in the data, which has been identified as either Q_{II}^D or Q_{II}^P by the Q_possible_peaks function. The $\sqrt{2}$, $\sqrt{3}$, $\sqrt{6}$, $\sqrt{8}$, and $\sqrt{9}$ peaks were found in the data by the finder.py script, and have been correctly indexed in the case of the Q_{II}^D possible assignment. In the Q_{II}^D projection, the $\sqrt{4}$ peak has additionally been projected, so that visually it matches up in Fig 5a, but is not actually identified for comparison purposes. However, as the first peak is within the accepted search range, and all but one of the peaks are present in the data, the phase assignment (complete with peak indexing) has been confirmed as correct. In the case where the Q_{II}^P phase has been tested, as shown in Fig 5b, it can be seen the there are visual matches on four of the peaks, whilst the proposed $\sqrt{10}$ peak of the Q_{II}^P phase falls at a point where the isn't a peak either in the data, let alone by visual inspection. Notably in this case, it is the $\sqrt{4}$, $\sqrt{6}$, and

$\sqrt{16}$ peaks that have been correctly assigned - and matching up with the correctly assigned peaks a factor of $\frac{1}{\sqrt{2}}$ less.[4]

Therefore, the entire set of possible cubic phases have been tested and checked for correct assignment, so if any are present in the dataset of peaks provided to the respective functions, then they should have hopefully been identified by this point.

4.2 Lamellar and Hexagonal phases

As previously highlighted, it might reasonably be expected that for powder scattering, the number of peaks identified in the data will be fewer for non-cubic phase patterns than for cubic-phase structures. If this condition is met in the first instance, then instead of the cubic-phase routines being carried out, a shorter, separate, routine for identifying either L_α or H_{II} mesophases is used. The La_HII_possible_phases function expects only two variables:

peaks : The array of peak positions

bin_factor : A factor to determine bin width of the histogram used for finding lattice parameters of phases.

Overall, the function uses a similar statistical frequency method to the cubic phase routines, but simply distinguishes between which phase has been used by which is the more common value. The incidence between the possible L_α and H_{II} peak assignments coincide such that this statement should always be true if the phase provided to the routine is indeed one of those phases.

5 Acknowledgements

CSB thanks Adam Squires and Tim Snow for their invaluable feedback into the development of the methodology.

References

[1] C. V. Kulkarni, W. Wachter, G. Iglesias-Salto, S. Engelskirchen, and S. Ahualli, “Monoolein: a magic lipid?,” *Phys. Chem. Chem. Phys.*, vol. 13, pp. 3004–3021, 2011.

[2] D. Sivia, *Elementary Scattering Theory.* OUP, 2 ed., 2011.

[3] A. M. Seddon, G. Lotze, T. S. Plivelic, and A. M. Squires, “A highly oriented cubic phase formed by lipids under shear,” *Journal of the American Chemical Society*, vol. 133, no. 35, pp. 13860–13863, 2011. PMID: 21823596.

[4] A. M. Squires, J. E. Hallett, C. M. Beddoes, T. S. Plivelic, and A. M. Seddon, “Preparation of films of a highly aligned lipid cubic phase,” *Langmuir*, vol. 29, no. 6, pp. 1726–1731, 2013. PMID: 23347289.

[5] T. Oka and H. Hojo, “Single crystallization of an inverse bicontinuous cubic phase of a lipid,” *Langmuir*, vol. 30, no. 28, pp. 8253–8257, 2014. PMID: 25007349.

[6] M. Newville, T. Stensitzki, D. B. Allen, and A. Ingargiola, “LMFIT: Non-Linear Least-Square Minimization and Curve-Fitting for Python,” Sept. 2014.

[7] H. Kim, Z. Song, and C. Leal, “Super-swelled lyotropic single crystals,” *Proceedings of the National Academy of Sciences*, vol. 114, no. 41, pp. 10834–10839, 2017.

[4] In fact, in the case of these data, the Q_{II}^{P} test failed at the first hurdle: the first peak was outside of the search range.