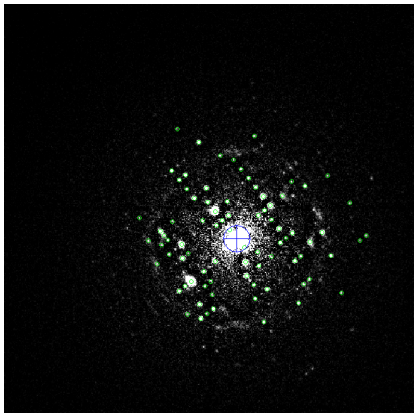
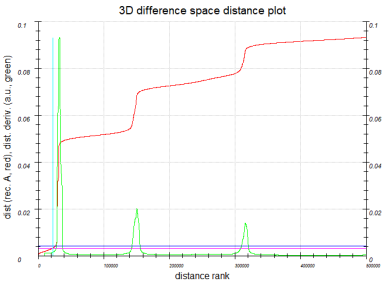
NTNU

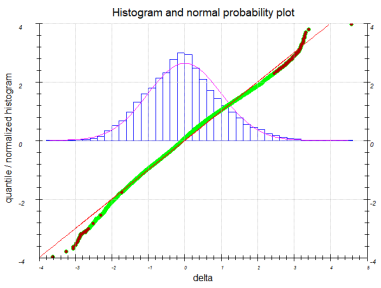
Kunnskap for en bedre verden

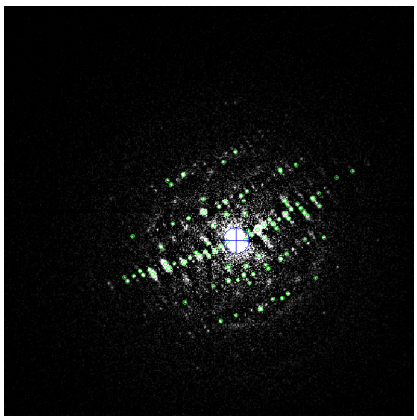
Manual for processing of 3D electron diffraction data using PETS

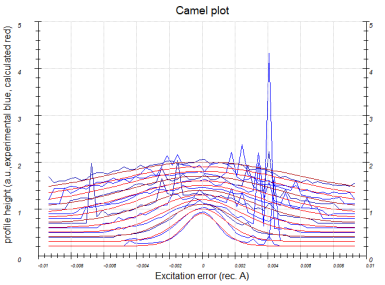
Iver Karlsbakk Småge

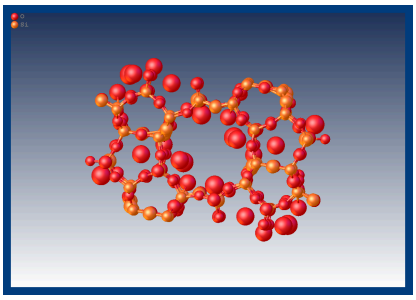












Last updated 15.06.2025

Contents

1 Introduction

This manual covers data reduction using Process Electron Tilt Series (software) (PETS) for NTNU-specific equipment, and lightly touches on structure solution using SHELX. For structure refinement we refer to the previous manual whose original version is available openly in [1]. An updated version is available at https://github.com/TEM-Gemini-Centre/developments/blob/main/cRED/3D%20ED%20data%20processing/User%20manual%203D%20ED%20data%20processing%20NTNU.pdf. For access to this private repository contact Emil F. Christiansen, NTNU Department of Physics. For data acquisition using Instamatic at the TGS/NorTEM JEOL at NTNU, see the manual on acquisition made by Oskar Ryggetangen. This is also available at https://github.com/TEM-Gemini-Centre/developments/blob/main/cRED/cRED_data_collection_manual.pdf.

2 Tool overview and Installation

The data processing procedure consists of two main steps, data reduction and structure solution and refinement. The tools used in the procedure are listed in Table 2.1. An overview of key files or folders and their purposes are listed in Table 2.2.

Table 2.1: Tools used in the data processing procedure.

Tool	Purpose	Input	Output
crysm	Utility scripts, mainly preprocessing		
PETS	Data reduction, intensity determination	.pts, .pts2	.hkl, .ins, .cif_pets
edtools	Creating input file for SHELXT		.ins
SHELXT	Structure solution	.ins, .hkl	.res, .hkl
SHELXL	Structure refinement	.res, .hkl	
Olex2	GUI for SHELXT and SHELXL	-	-

Table 2.2: File formats/folders and their purpose.

File suffix	Output from	Purpose
.pts	Instamatic	PETS input config
.pts2	PETS (ctrl + s)	PETS stored state
<project>_petsdata	PETS	PETS stored state
_shelx.hkl	PETS integration	Input for SHELX (reflexes)
_shelx.ins	PETS integration	Input for SHELX (instructions/parameters)

2.1 PETS

PETS is a windows only program used for data reduction of 3D Electron Diffraction (3D ED) data from Continuous Rotation Electron Diffraction (cRED) or Precession Electron Diffraction (PED), which includes peak searching, refinement of various parameters and integration of intensities. PETS can be downloaded from http://pets.fzu.cz/, and requires a registration which is free for academic users.

2.2 SHELX and Olex2

SHELX [2,3] is a collection of command line executables that can be downloaded from https://shelx.uni-goettingen.de/download.php. Like PETS and Jana it requires a registration that is free for academic users. Once installed, their location must be added to the PATH. Olex2 is a windows only Graphical User Interface (GUI) for the SHELX programmes, and can be downloaded from https://www.olexsys.org/olex2/.

2.3 Python scripts: Crysm and edtools

Crysm

Crysm is an in-house developed collection of python scripts to aid the entire data analysis process. It requires python 3.10 or greater. For users, we recommend the quick installation from pip using `pip install crysm`.

Editable installation For developers — Note 2.1

Crysm must be cloned from https://github.com/iverks/crysm. To do so, navigate to a fitting directory and run `git clone git@github.com:iverks/crysm.git`. It uses `uv` as a project manager, which can be installed from https://docs.astral.sh/uv/getting-started/installation/ To install crysm, run `uv tool install -e .` in the newly cloned folder. Any changes made to the code will automatically be reflected in the program. If the dependency list in `pyproject.toml` is updated, `crysm` must be uninstalled and reinstalled using `uv tool uninstall crysm; uv tool install -e .`.

Make sure that it is available from any location by running `crysm --help`. You should get a message showing how to use `crysm` and listing all available commands, as shown in Figure 2.1.

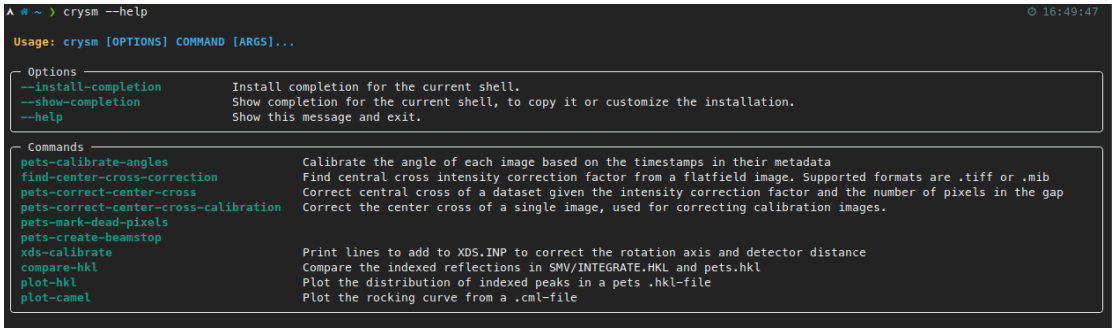


Figure 2.1: Running `crysm --help` gives us the instructions for how to use crysm.

Similarly it is possible to get instructions for how to use subcommands by running `crysm COMMAND --help`, shown in Figure 2.2.

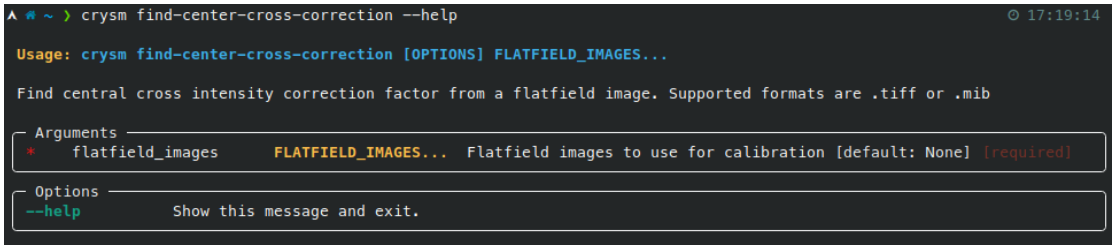


Figure 2.2: Running `crysm COMMAND --help` gives us the instructions for how to the specific subcommand from crysm.

If anything fails and the versions dependencies have to be changed, the package needs to be reinstalled.

edtools

Edtools can be installed from pip using the command `pip install edtools`. The command we use from edtools, `edtools.make_shelx` depends on one of `cctbx.python` and `sginfo`. We thus need to install either `cctbx` or `sginfo`

For Windows, installing `cctbx` is the simplest. An installation of conda is required, then `cctbx` is installed using `conda install -c conda-forge cctbx-base`. Make sure that `cctbx.python` is available for `edtools` by running `edtools.make_shelx --help`. If the text "Either sginfo or cctbx.python must be in the path" is returned, then the location of `cctbx.python.bat` must be added to path. The program should be in `C:\Users\USERNAME\ANACONDA_DISTRIBUTION\Library\bin\`.

For linux users, installing `sginfo` is simpler. The project can be found at https://github.com/rwgk/sginfo, cloned with `git clone git@github.com:rwgk/sginfo.git sginfo` and built with any C compiler using `clang -o sginfo sgclib.c sgfind.c sghkl.c sgio.c sgsi.c sginfo.c -lm`. The executable should either be moved to a folder on `PATH` or the folder added to `PATH`.

3 Preprocessing

3.1 Correcting for the central cross

The QuantumDetectors Merlin 4R electron detector of our JEOL ARM200F Transmission Electron Microscope (TEM) consists of four subdetectors as shown in Figure 3.1 a). These subdetectors are separated from each other by a small gap. This gap is on the scale of a few pixels. We need to correct for this both in our calibration images and in the data used for reconstruction.

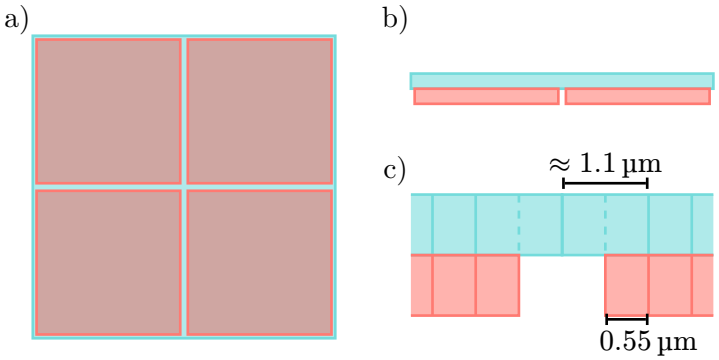


Figure 3.1: a) The detector consists of 4 smaller detectors that have a small gap between them. b) Side view of the detector geometry. The two layers are a silicon chip (teal) and the four electron detectors (red). c) The detector gap creates pixels at either edge of the border that collect electrons from a larger area than the rest.

In addition, electrons that hit this gap are detected by the nearby pixels with a certain probability, effectively increasing their intensity by a fixed factor. This is what gives rise to the characteristic bright cross in the raw images. We call this fixed factor the “correction factor”. The central four pixels collect electrons from an even larger area, and thus have their intensity increased by another factor. This factor we name the “central four factor”. To correct for these effects we first need to characterize our detector on a flatfield image such as the one shown in Figure 3.2. A flatfield image is captured by illuminating the entire detector evenly and capturing an image of vacuum, meaning there is nothing inside the field of view.

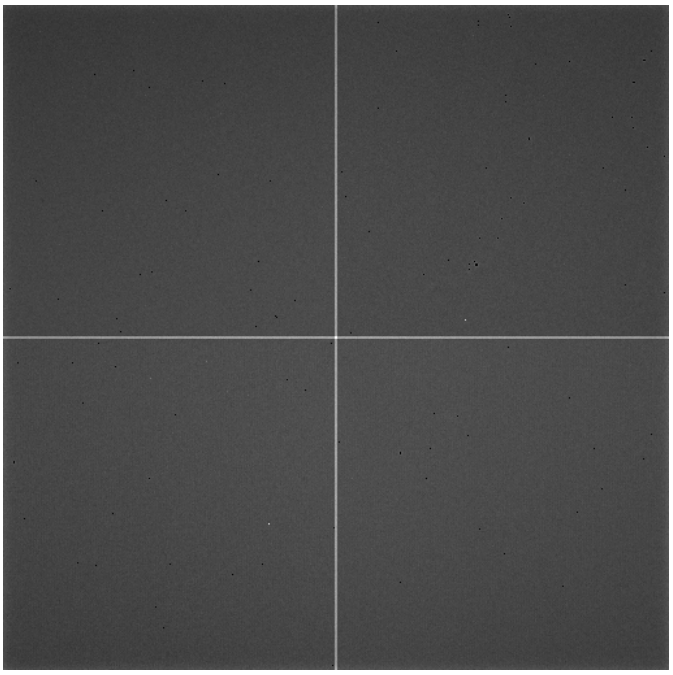


Figure 3.2: Example flatfield image. The central cross is brighter than the average pixel. The central four pixels are even brighter.

The crysm package contains a script to calculate the correction factors from a flatfield image. To run it, we use the command `crysm find-center-cross-correction PATH/TO/FLATFIELD_IMAGE.mib`. The flatfield image can be a .mib or a .tiff image. This calculates the correction factor and the central four factor, and depending on them suggests either two or four additional pixels be added in the center, but ultimately this should be chosen by the analyst. The currently used correction factors are show in Table 3.1.

Table 3.1: Currently used central cross correction values as of 15.05.2025.

Variable	Value
Additional pixels	2
Correction factor	2.196
Central four factor	4.051

After finding the correction factors, the images must be converted. To do this, navigate to the experiment folder and run the command `crysm pets-correct-central-cross --additional-pixels 2 --correction-factor 2.196 --central-four-factor 4.051` using the previously determined values. This generates a new folder `tiff_corr` containing the corrected images that will be used in Section 3.5.

3.2 Calibration of pixel size

Finding the correct pixel size, in PETS referred to as “Aperpixel”, is essential. The parameter tells us how many inverse Ångström in reciprocal space each pixel in the image represents. It requires some knowledge of crystallography and the dedicated sample, so it is recommended to get help with this the first time, and its details are thus not covered in this manual.

The calibration of pixel size depends on the chosen number of additional pixels when correcting for the central cross. In order to create a widened calibration image we can use the command `crysm pets-correct-central-cross-calibration --additional-pixels 2 PATH/TO/INPUT_IMAGE.tiff PATH/TO/OUTPUT_IMAGE.tiff`. The input image can be either .tiff or .mib, but the output must be .tiff.

Correction tool | Advanced use — Note 3.1

Optionally it is also possible to set `--correction-factor 2.196 --central-four-factor 4.051`, but it is not necessary since we do not care about the intensity correction for the pixel size calibration image.

By using `crysm pets-correct-central-cross-calibration --additional-pixels 0 --correction-factor 1 --central-four-factor 1 input.mib output.tiff`, this script can be used as a .mib to .tiff conversion tool.

The used calibration values as of writing this are presented in Table 3.2.

Table 3.2: Pixel size calibration values for the acceleration voltage of 200 kV used for three different camera lengths as of 01.05.2025.

Gap correction	120 cm	150 cm	200 cm
Not corrected	0.010207	0.007929	0.006167
Corrected 2px	-	0.007859	0.006062
Corrected 4px	-	0.007682	-

3.3 Setting Aperpixel and other constants in the .pts file

It is recommended to set as many parameters in the initial config file as possible even though they often are possible to override through the graphical user interface. This is to save time whenever the analysis has to be restarted. The frame scale should be set to the value calculated above. Remember that the calibration constant is different when correcting for the central cross widening than when not doing it.

lambda

The lambda value is the wavelength of the electrons. Instamatic sets this to a configurable constant. For our setup this is 0.025 $Å^{-1}$, which is correct for 200 kV. Be aware that this value is **not** automatically updated when the acceleration voltage is changed. The value should be calculated based on the used acceleration voltage. The corresponding wavelengths for our selection of acceleration voltages are listed in Table 3.3.

Table 3.3: Wavelengths for our selection of acceleration voltages.

Acceleration voltage [kV]	Wavelength [$Å^{-1}$]
80	0.041757
200	0.025079

Aperpixel

The Aperpixel value is the size of the pixel in reciprocal space in $Å^{-1}$. It should be set according to the calibration as explained in Section 3.2 above.

phi

Phi is the semiangle of a tilt step, meaning it should be half of the tilt step between two images. Starting from Instamatic v2.1.1 this value will be set automatically. When using `crysm` to calibrate angles (next step) it will be overridden. A reasonable value is 0.035°.

omega

Omega is the rotation axis in the image plane, defined likewise to the angles on the unit circle where 0° is directly to the right. Even if it ends up being the same axis, the direction seems to matter somehow, e.g. $90° \neq 270°$. It is **not** automatically set by Instamatic. To find it we need to inspect the images and observe which axis the Ewald sphere moves across. Only an approximate value is needed as PETS does a good job of refining it. The value should be the same across experiments. If it is not known yet, it can be found at a later step in the process. A reasonable value is 232°.

bin

Bin is how many pixels should be binned together. According to the official PETS documentation, this makes the analysis faster and more robust for images that are larger than 1000x1000 pixels. Since our images are not, we leave it at 1.

reflectionsize

The reflection size is the diameter used of the reflections. It should be large enough to encompass strong spots, but not so large that it overlaps with neighboring points. An example of correctly sized reflections can be seen in Figure 4.3, where the sizes of the small green circles are set by the reflectionsize. It is fine to leave the reflectionsize as 20 and correct it interactively before running the peak search.

noiseparameters

Noiseparameters are two numbers. In the GUI they are referred to as $G\gamma$ and ψ.

The first number, $G\gamma$, is the expectation value for how many pixels light up per electron hitting the detector. This value should usually be between $1.3 - 1.5$. Finding the value is not simple, but can be done by inspecting flatfield images with very low dose such that single electron impacts are discernible. It should be further discussed with the producer of the detector. The value used for our detector is 1.5, according to analysis done by Lukas Palatinus, May 2025.

The second number, ψ, is the constant noise when no electrons are hitting the detector. For our detector this is simply 0.0.

After setting the parameters, the file should look something like this:

```
lambda 0.025079
Aperpixel 0.00285546
phi 0.035
omega 231.0
bin 1
reflectionsize 5
noiseparameters 1.5 0

imagelist
tiff/00001.tiff   -61.4524 0.00
tiff/00002.tiff   -61.3748 0.00
tiff/00003.tiff   -61.2973 0.00
...
tiff/01610.tiff    63.3724 0.00
tiff/01611.tiff    63.4500 0.00
endimagelist
```

It is advised to create a backup of this `pets.pts` file in case anything happens to it by copying it to `pets.bak.pts`.

3.4 Calibration of rotation angles

The PETS input files (.pts) contain a list of all frames, their image and their corresponding angle in the `imagelist` section.

Instamatic assumes the angles of each image are equally spaced when generating the PETS input file. This is not exactly true. The images have timestamps in their header, which we can use to recalibrate the rotation angles. In the dataset folder run `crysm pets-calibrate-angles`. This generates a new input file `pets-fromtime.pts` where the orientation angle of each frame is calibrated, and the `phi` parameter is set to the median tilt step of the new dataset.

3.5 Use the central cross corrected images

The PETS config file must be modified to use the central cross corrected images generated in Section 3.1. Using the multiline editing capabilities of your text editor, add change the image path of all images from `tiff/*.tiff` to `tiff_corr/*tiff`. In VSCode or Notepad++ this can be done by placing the cursor on the first line of the image list, scrolling to the bottom of the file and clicking the last line of the image list while holding **SHIFT** and **ALT**. The file should then look something like:

```
lambda 0.025079
Aperpixel 0.00285546
phi 0.035
omega 231.0
bin 1
reflectionsize 5
noiseparameters 1.5 0

imagelist
tiff_corr/00001.tiff   -60.4956 0.00
tiff_corr/00002.tiff   -59.8215 0.00
tiff_corr/00003.tiff   -59.7463 0.00
...
tiff_corr/01610.tiff   63.3753 0.00
tiff_corr/01611.tiff   63.4500 0.00
endimagelist
```

Hint Using the corrected images in XDS — Note 3.2

The central cross corrected images can also be used in XDS. This is done by setting the `NAME_TEMPLATE_OF_DATA_FRAMES` to the correct pattern, and setting the image format to `TIFF`.

```
1 NAME_TEMPLATE_OF_DATA_FRAMES= ../tiff_corr/0????.tiff TIFF
```

3.6 Generate a beamstop-file for the central cross

When using central cross correction Section 3.1, a beam stop for the central cross should not be necessary. If a beam stop mask is to be used anyways for the central cross, it can be created with `crysm` using the command `crysm pets-create-beamstop`. The command takes two parameters, image width and beamstop width. The image width is the width of the image after being corrected for the central cross. If using no additional pixels this is usually 512, and with an additional pixels value of 2 it becomes 514. The beamstop width is up to the user. The command is then `crysm pets-create-beamstop --image-width 514 --beamstop-width 6 beamstop.xyz`. If it is only desirable to mask the central four pixels, a beamstop can be created with the command `crysm pets-create-beamstop --image-width 514 --beamstop-width 4 --beamstop-kind square beamstop.xyz`. The beam stop files can be reused in other projects with the same image and beamstop widths.

Summary Crysm preprocessing commands — Note 3.3

Find correction constants
`crysm find-center-cross-correction PATH/TO/flatfield_200kV_24bit.mib`

Central cross correct all images in dataset
`crysm pets-correct-central-cross-calibration --additional-pixels 0 --correction-factor 1 --central-four-factor 1 input.mib output.tiff`

Central cross correct pixel calibration image
`crysm pets-correct-central-cross-calibration --additional-pixels 2 PATH/TO/INPUT_IMAGE.tiff PATH/TO/OUTPUT_IMAGE.tiff`

Correct angles in config file
`crysm pets-calibrate-angles`

Generate beamstop file
`crysm pets-create-beamstop --image-width 516 --beamstop-width 6 --beamstop-kind cross beamstop.xyz`

4 Data reduction in PETS

Details of the steps in PETS are presented in the PETS manual, which is recommended supplementary reading [4]. This manual is meant as a more instructive manual for specifically our workflow. An overview of the workflow is presented in Figure 4.1.

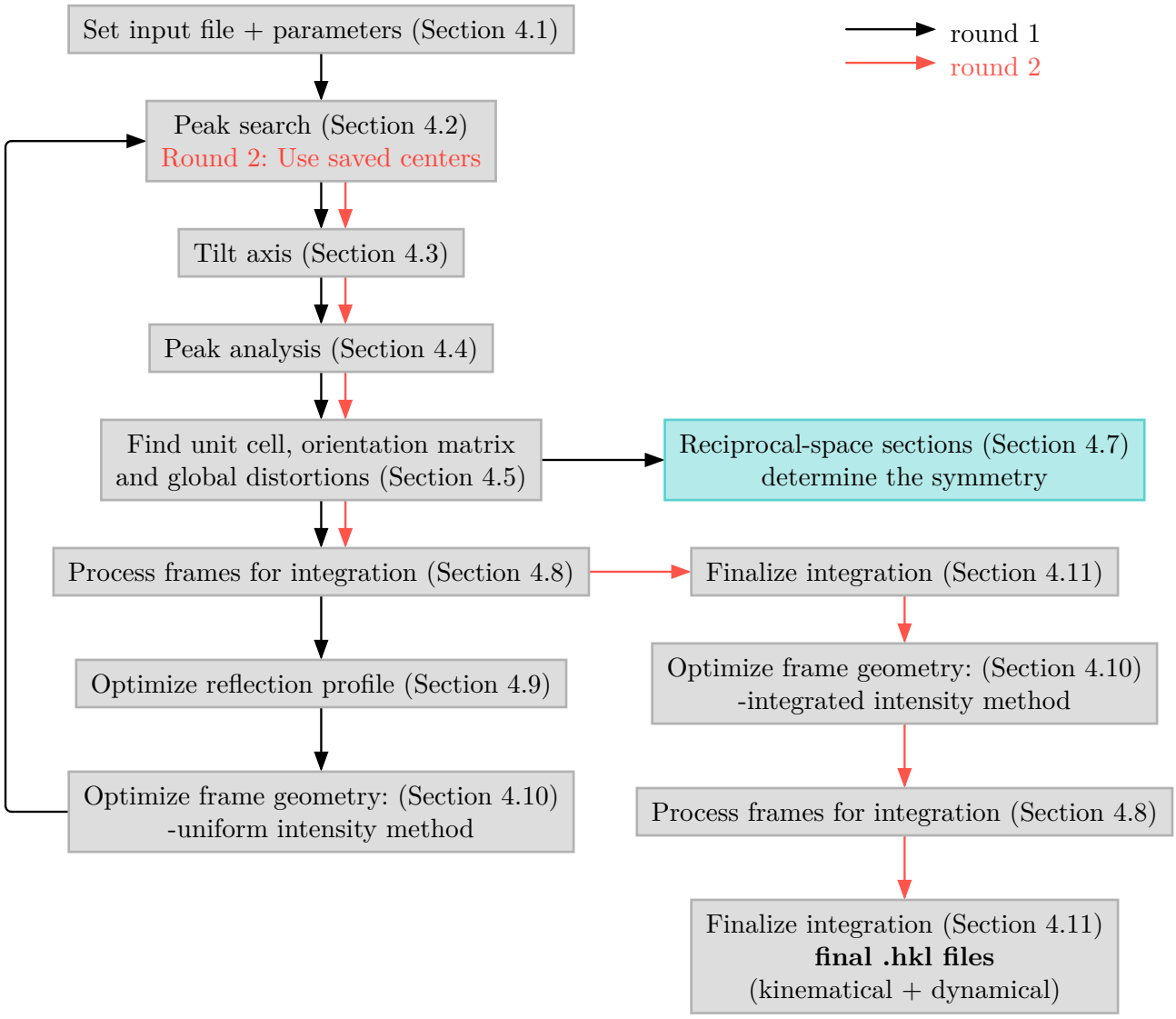


Figure 4.1: Overview schematic of the PETS workflow. Adapted from section C of the PETS manual [4].

4.1 Parameters

Set geometry to continuous rotation. Estimate and set reflection diameter, should be a bit smaller than a medium sized peak. Verify calibration constant is set to the calibrated value. The defaults for the parameters: max d* for integration, max d* for peak search and min d* of 1.4, 1.4 and 0.05 are fine. The detector noise parameters should be set from Section 3.3. For 12 bit datasets the detector saturation limit should be set to 4 095. For 24 bit the saturation limit should be set as high as possible, which is 2 000 000 000.

If a beamstop was generated in Section 3.6, it should be loaded at this point by clicking “Yes” and “Load” and selecting the given file.

Click on “Detect” to detect bad pixels. It usually detects 82. Usually, some pixels go undetected, and should be added manually using `crysm`. First export the pixels from the PETS GUI to a file `dead_pixels.txt`. Then run the command `crysm mark-dead-pixels --dead-pixels dead_pixels.txt tiff_corr/00001.tiff`. This loads the dead pixels from the file and the image `tiff_corr/00001.tiff`. To toggle a dead pixel, double click the pixel in the image. To verify that the pixel really is dead, you can scroll through the images using the buttons “d” (aDvance) and “a” (bAck). Closing the window saves the modifications to the opened file automatically. Then load the pixels into PETS by clicking the “Import” button. After manual intervention, the dead pixel count should be around 110. The saved dead pixel files can be reused in other projects with the same detector and gap correction width.

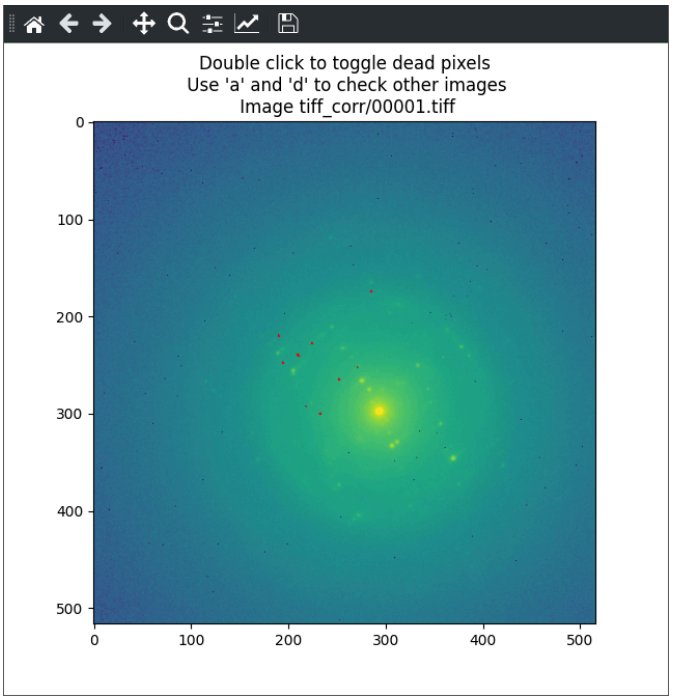


Figure 4.2: GUI for marking dead pixels in `crysm`.

4.2 Peak search

When doing the peak search, we have generally had better success using the direct beam for center determination than with the friedel pairs. According to the creator of PETS, using the direct bean introduces a bias, while using friedel pairs introduces noise. Sometimes increasing the I/sigma ratio to 15 or 20 is a good way to filter out noise if there are too many peaks detected. If too few peaks are detected it can be reduced to 5 or 3. Usually, leaving it at the default value of 10 is fine for cRED data. On the second run (red arrows in Figure 4.1), the center determination should be set to “Use saved centers” to use the centers found in “Optimize frame geometry”.

Possible issue | The direct beam is in the central cross | Note 4.1

If the direct beam is in the central cross, proceed by using the friedel pairs method. Then after finding the unit cell, run "Optimize frame geometry" with only "Frame orientation angles" and "Center of the diffraction patterns" enabled. Finally, rerun the peak search step using "Use saved centers".

4.3 Tilt axis

The initial ω angle should already be set to an initial estimate from Section 3.3. Run the step without enabling "Global search for tilt axis position ω". Usually enabling the optimization of the δ angle leads to worse results. Note that the δ angle is refined on a frame by frame basis in Section 4.10.

Help | Finding an initial guess for the tilt axis | Note 4.2

In Figure 4.3 we can see three equally spaced frames from a reference dataset. We can see the Laue circle, the intersection between a layer of the reciprocal crystal and the Ewald sphere, moving across the image. This should be present in most data sets, but not all, and the distance of the path from the origin might vary depending on the crystal orientation. The center of the Laue circle can be traced across the dataset. In the figure, an estimation of the circle has been superimposed in red along with its center. In Figure 4.3c), the centers from the previous frames are added as well in transparent red, and the trace has been sketched by the dashed red line. Then we can see that the rotation axis (green double lines) is perpendicular to the trace of the Laue circle center. It is sufficient to approximate this direction, since PETS refines the rotation axis.

The green lines for the tilt axis can be enabled by checking "Resolution rings, tilt axis and ice rings" in the "Image options" tab of the right window. The found peaks are marked with green rings when "Peak search" right below is checked and a peak search has been run.

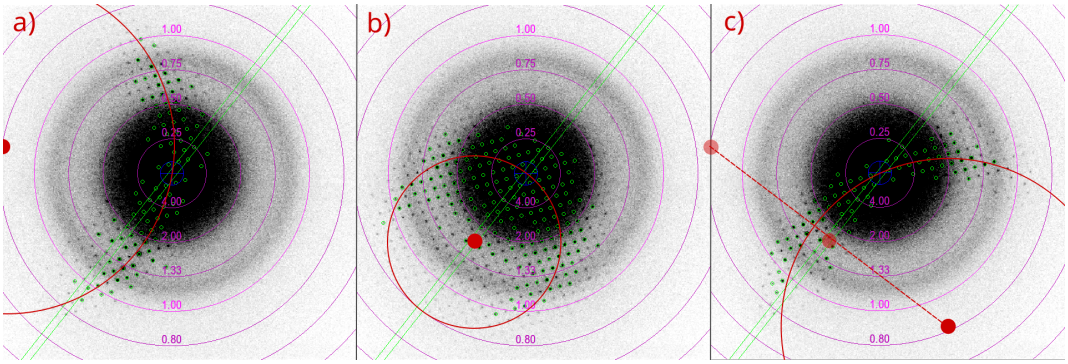


Figure 4.3: Three equidistant frames from the Mordenite 1 dataset. The found peaks are marked by small green rings and the rotation axis by a double line. Edited onto the image is an estimation of the Laue circle (red ring) and its center (red dot). In the last frame (c) the Laue circle centers of the other frames are marked by transparent red dots. The path of the center of the Laue circle is marked with a red dashed line.

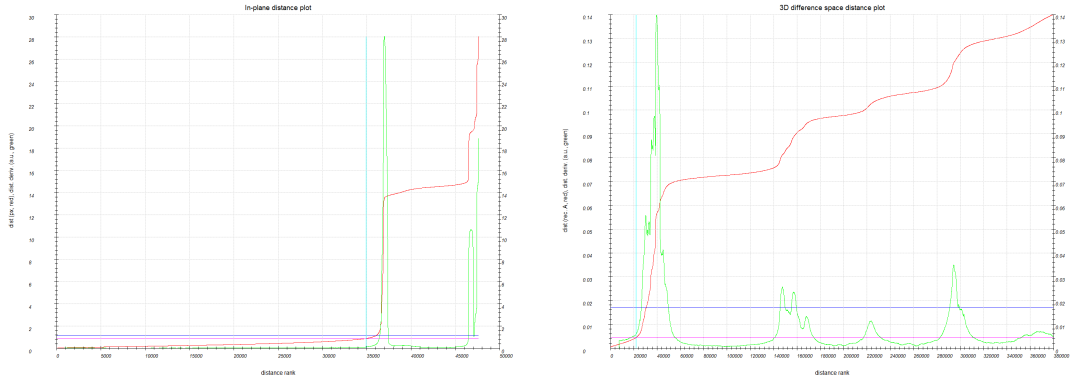
Possible issue | The tilt axis is unknown | Note 4.3

If the tilt axis is unknown and can't be guessed from observing the path of the Ewald sphere over the images, "Global search for tilt axis position ω" can be enabled. Sometimes this finds a local minimum instead. If that is the case, it might be better to do many local searches from several initial guesses, for example every 10° or 20°.

4.4 Peak analysis

Clicking "Peak analysis" a plot of in-plane distances is shown. The plot has two curves, a red and a green curve. The red curve is the distances between each pair of peaks sorted in ascending order. The green curve is the derivative of the red curve.

We expect the red curve to have several distinct steps, and thus the green to have defined peaks.



a) b)

Figure 4.4: a) In-plane distances. b) 3D distances.

Next, after clicking "Peak analysis (continue)" the same plot is presented, but for distances in the constructed 3D reciprocal space. We again expect the red curve to have several distinct steps, and thus the green to have defined peaks.

Finally, click "Peak analysis (continue)" to finish the step. This step creates the reciprocal space constructions "xyz", "clust" and "dist" used in indexing.

4.5 Find unit cell and orientation matrix

Open the reconstruction by clicking "Find unit cell and orientation matrix". Click find possible cells automatically. If no cell is found, try to change the data used for indexing to "diff", the difference space. If the issue persists, change back to "xyz" and select "from triplets" in the "Find possible cells automatically" section. Finally, if the issue still persists use "diff" and "from triplets". If the found cell is incorrect we can try to modify the cell. Do however note that this is a sign that the data quality is low.

Possible issue The unit cell vectors are too long — Note 4.4

If the detected cell has correct orientation, but one of the vectors is twice as long as it should be, open “Modify cell” and click “Go to supercell”. The unit cell should then be corrected to the smallest one as shown in Figure 4.5.

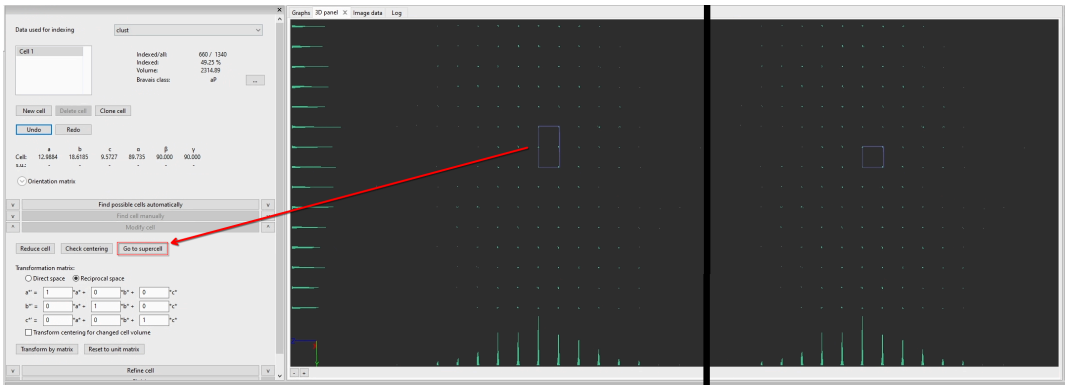


Figure 4.5: The unit vector might be too long.

Possible issue | Cannot find unit cell | Note 4.5

If the unit cell cannot be found by automatic means it is possible to define it manually. It is advised to use the “diff” space or “clust” space for this task, due to their higher data density near the origin.

Navigate to a high-symmetry direction, and rotate so that the grid is aligned to the x and y-axes. The histograms on the bottom and left should be as sharp as possible. Click the “Define directions” button to start. First, place the cursor on a point on the same row as the origin point. Make sure that the “order” is properly detected as the number of intervals your line spans, as shown in Figure 4.6, otherwise set it or retry. Then check the “b” radio button, and place the cursor on a point on the same column as the origin point. Finally, click the “a” button under “View along direction”, and draw the last line such that all angles are 90°. The cell does not need to be perfect as it will be refined.

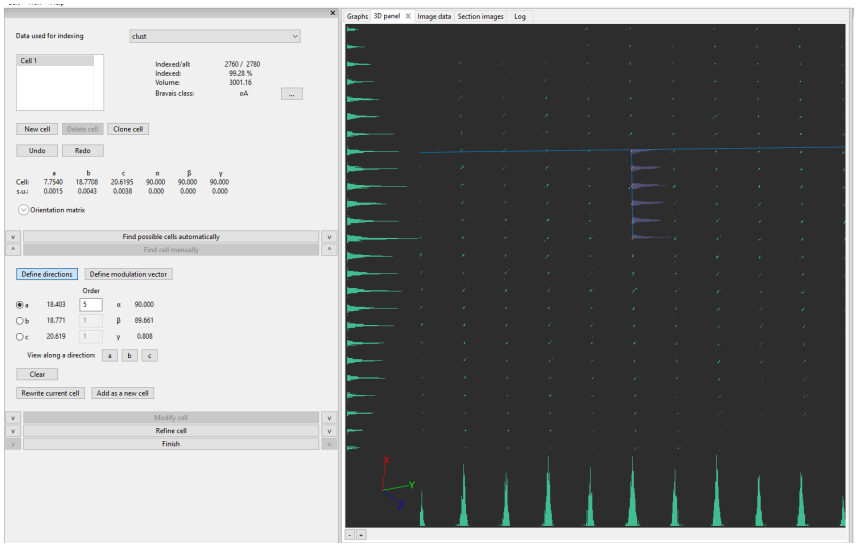


Figure 4.6: Screenshot of finding the cell manually. Here the number of intervals spanned is 5, seen by the 6 red dots representing the different planes. The order is properly set to 5 for the a direction.

Finally, open the “Modify cell” tab and click “Reduce cell”. This finds the Bravais class and orders the lattice parameters in a consistent order.

Refining cell

In this step there is no determined best way to do it. Instead two options are suggested.

Option 1 | Refinement with distortions | Note 4.6

Refine the cell and distortions once per distortion in the order they are listed. Then refine the cell using “Refine cell from d”. finally “Refine UB + cell” with the crystal system set either to “Triclinic” or the one suggested by PETS.

Option 2 Simple refinement Note 4.7

In the first run use “Refine cell from d”. In the second run use “Refine UB + cell” with the crystal system set either to “Triclinic” or the one suggested by PETS.

4.6 Removing bad frames

If at this point some frames or ranges of frames have been deemed to be bad, for example because the particle moved out of the selected area aperture at the end of the rotation, the offending frames should be excluded from computation. This can be done with the “Frame dialog” window. In order to remove a range of frames from calculation, hold **SHIFT** while clicking the top and bottom images in the range. Then uncheck the box labeled “Use for calculation” as shown in Figure 4.7.

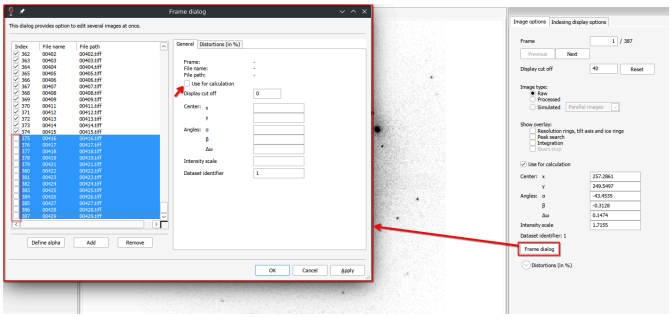


Figure 4.7: Remove a range of multiple images

4.7 Reciprocal space sections

In the reciprocal space sections, we can specify the space group by comparing the reflection conditions/extinction rules to the tables. This section might not be necessary as SHELX finds the space group automatically, but can be done to increase our certainty or confirm a suspicion without solving the structure.

Clicking the “Reciprocal-space sections” button creates a set of images of the reciprocal space planes. Applying symmetry in the reconstruction helps completeness of the images, but might be a false safety net. In the reciprocal space sections, the planes of where one of h, k, l is set to one of $0, 1, 2$. Comparing with the tables in IUCr vol A tables [5]. The tables are available digitally at https://onlinelibrary.wiley.com/iucr/itc/Ac/ch1o6v0001/sec1o6o5.pdf.

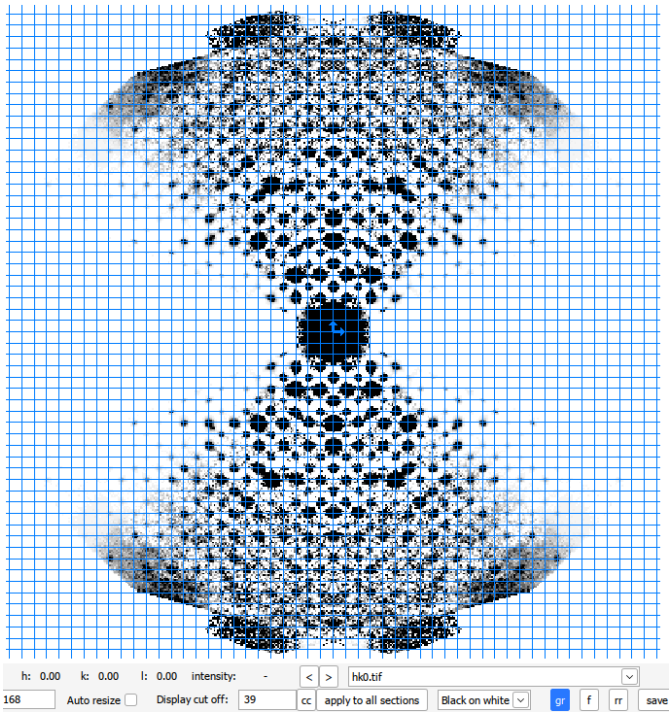


Figure 4.8: The $hk0$ plane for Mordenite (Cmcm) has the reflection condition $h + k$, which can be seen by the missing reflections at $h = 3, k = 0$, $h = 4, k = 1$, $h = 5, k = 2$ etc. The grid is enabled with the "gr" button.

Possible issue | The reciprocal images are hard to read | Note 4.8

Sometimes, the peaks in the generated reciprocal space images are very big, like the $h = 8, k = 0$ peak or the direct beam in Figure 4.8. If too many of the peaks are like this, the image is impossible to interpret. To avoid this issue, this step can be done manually in the 3D reconstruction instead.

First, open the reconstruction by clicking "Find unit cell and orientation matrix". Use the buttons on the right labeled "a*", "b*" and "c*" to align the reconstruction with the lattice. Use the "rectangle area selection" tool to select a plane, and the mentioned buttons to orient the view directly above the plane. Enable the lattice by checking "Show lattice", and grow it in the in-plane directions using the number inputs to the right of "Fold to cell". The lattice might obscure the points, but this can be avoided by toggling it off again.

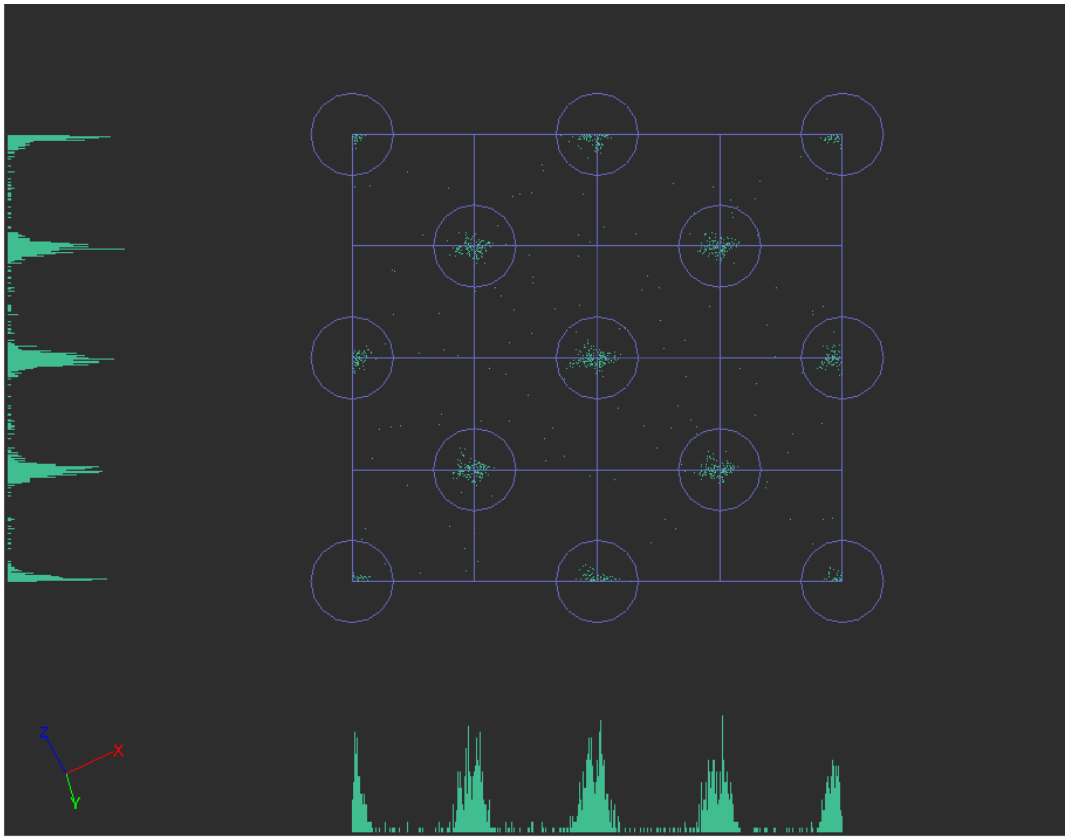


Figure 4.9: The $hk0$ plane for Mordenite (Cmcm) has the reflection condition $h + k$. Here, the reflections have been highlighted using "Show reflections", and the space is folded to 4 by 4 unit cells to get a higher density of points for illustrative purposes.

4.8 Process frames for integration

When processing frames, always use the "Fit profile" intensity determination method. The values "Rocking curve width" and "Apparent mosaicity" are shared between "Process frames", "Optimize reflection profile" and "Optimize frame geometry". For

the first run they should already be set to reasonable values. The rocking curve width default of 0.001 is good, and the apparent mosaicity default of 0.1 is good.

4.9 Optimize reflection profile

The "Precession angle/tilt semiangle" should be set from Section 3.3. The "Minimum $I/\sigma(I)$" is fine set to its default 10, but can be increased to 15 or 20 as in Section 4.2 to filter out noise. It is only used for the display of the rocking curve, not for any calculations. In this step the "Rocking curve width" and "Apparent mosaicity" values are refined.

4.10 Optimize frame geometry

Optimize frame geometry optimizes for common frame-by-frame errors in 3D ED data.

The first time running this, "Uniform intensities" should be chosen as simulation method. If a successful integration has been performed, e.g. it's the second round of optimization, the "Integrated intensities" method is generally more accurate.

When selecting the geometrical parameters to optimize for, the default is to optimize for "Frame orientation angles", "Center of the diffraction pattern" and "Apparent mosaicity", but not "RC width" or any of the "Distortions". Beware to *not* enable both "RC width" and "Apparent mosaicity", as they are correlated.

For the rotation angles, the assumption is that they are related to the previous and next frames, thus by default there is a smoothing applied. By default this is set to a polynomial of 4th order. For very misaligned data, a moving average might be better. For continuous rotation data with a small tilt step ($< 0.5°$) the order of the moving average smoothing can be increased to 10-20.

4.11 Finalize integration

To speed up the process slightly you can disable dynamic refinement as it is not used in the rest of the process as of writing. After running, make sure the detected Laue class for scaling is correct.

4.12 Automatic workflow

Once the process is mastered, it can be sped up using the automatic workflow. I prefer finding and refining the unit cell manually and then running the rest of the steps automatically.

5 Structure solution in Olex2 + SHELX

5.1 With edtools

Copy the file `<jobname>_shelx.hkl` into a new folder "shelx". Run the terminal command:

```
# Format
python3 -m edtools.make_shelx -s <SPACE_GROUP> -m <COMPOSITION> -c <A> <B> <C> <ALPHA> <BETA> <GAMMA>
# Example
python3 -m edtools.make_shelx -s Pnma -m Si1 O2 -c 13.029 19.994 20.102 90 90 90
```

Note that you must explicitly specify the 1 that is normally omitted from SiO_2.

Rename the generated `shelx.ins` such that it has the same name as your `.hkl` file. It might be nice to save a backup of the `.ins` file if you want to try multiple `.hkl` files with the same unit cell, because Olex2 modifies the `.ins` file when run.

Start the Olex2 program and open the `.ins` file. Open the "Work" tab and click the arrow to the right of the "solve" button. This opens the options for structure solution. Since we expect slightly higher errors than in X-ray data, we need to increase the α threshold for selecting possible space groups. This is done in the "COMMAND LINE" section under the "Solution settings extra" dropdown as shown in Figure 5.1. There we can set the `-a` flag to a higher value, for example `-a"0.6"`. If your structure is centrosymmetric, make sure the output from SHELXT does **not** say "0 Centrosymmetric and N non-centrosymmetric space groups evaluated". If this is the case, increase the value passed to the `-a` flag.

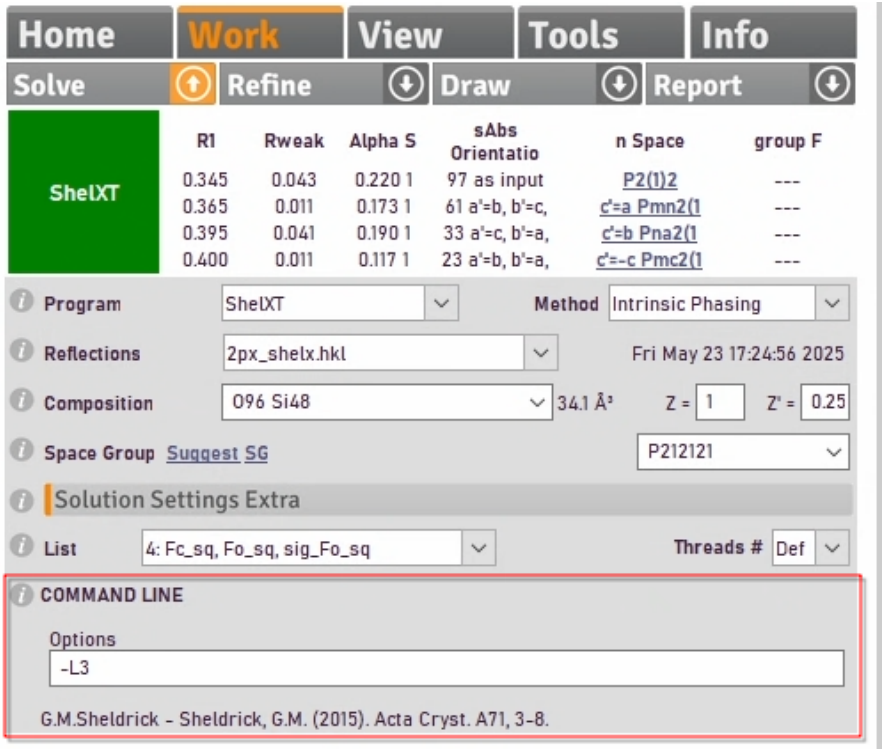


Figure 5.1: Flags can be set for SHELXT in Olex2 using the "Solution settings extra" dropdown.

Then, solution can be started by clicking the “solve” button. Use the command `pack cell` in the command line in the bottom left corner to grow the structure to a full cell for easier inspection. Use the command `fuse` to revert back to the anisotropic unit before refinement.

For refinement we refer to the previous manual, whose first version is openly available at [1]. An updated version can be found in the private repository https://github.com/TEM-Gemini-Centre/developments.

Possible issue The solution looks correct, but stretched Note 5.1

If the solved structure has the components of the expected structure, for example the rings we expect from the ZSM-5 structure in this example, this may be caused by the order of the unit cell parameters being wrong. This can be solved by reordering the lattice parameters in the .ins-file and running the structure solution again.

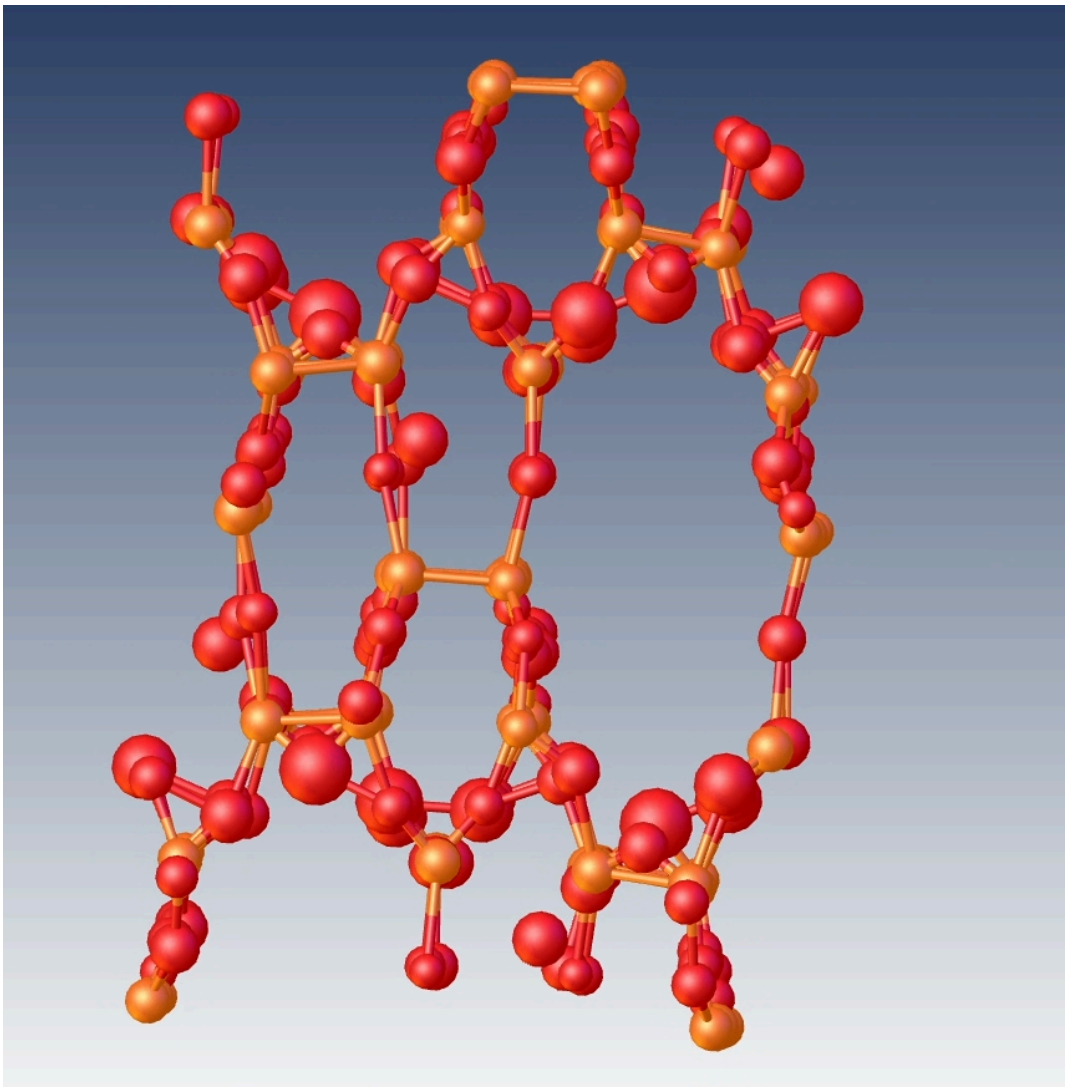


Figure 5.2: Sometimes the structure looks stretched compared to the compared result, here in the y-direction.

Potential issue SHELXT finds the wrong space group — Note 5.2

If SHELXT is unable to find the correct space group, the space group can be manually enforced using the command line flag `-S[SPACE GROUP]`, e.g. `-SPnma`. Using Olex2, this is done in the "COMMAND LINE" section under the "Solution settings extra" dropdown as shown in Figure 5.1.

5.2 Without edtools

Edtools is not hard to install, but it's dependencies `cctbx.python` or `sginfo` are. When making .ins files, edtools does two things differently from PETS. It sets the symmetries from the space group supplied, and the composition with structure factors. To compensate for the first, SHELXT can find the symmetries from the Laue group that can be supplied using the command line parameter `-l[LAUE GROUP NUMBER]`. When running in Olex2, the `-l` flag can be set in the GUI as shown in Figure 5.1. This has the added bonus that we don't need to determine the space group, only the Laue group, which is automatically determined by PETS during integration.

To compensate for the second is not necessary, we only need to add the composition, for example adding SiO_2:

```
1 SFAC O Si
2 UNIT 2 1
```

SHELX knows the scattering factors for the given atoms, so the result is equal to when inserting the composition along with the scattering factors.

The laue group numbers are between 1 and 17, and the human-readable Laue group is printed by SHELXT when running. The echoed Laue groups for each number are tabulated in Table 5.1.

Table 5.1: The laue groups corresponding to the laue group numbers in SHELXT.

Number	Laue group
1	$\overline{1}$
2	2/m
3	mmm
4	4/m
5	4/mmm
6	$\overline{3}$ (rhombohedral primitive)
7	$\overline{3}$ (hexagonal axes)
8	$\overline{3}$m (rhombohedral primitive)
9	$\overline{3}$1m (hexagonal axes)

Number	Laue group
10	$\overline{3}$m1 (hexagonal axes)
11	6/m
12	6/mmm
13	m$\overline{3}$
14	m$\overline{3}$m
15	all hexagonal and trigonal
16	monoclinic with *a* unique
17	monoclinic with *c* unique

The process described above is automated into the command `crysm pets-solve -m <COMPOSITION> PROJECT.pts2`. This command copies the `PROJECT_shelx.ins` and `PROJECT_shelx.hkl` files into the `shelx` folder, overwriting any colliding files, writes the composition into the `.ins` file, and runs `shelxt` using `-a0.6` and `-l[LAUE GROUP]`. It reads the Laue group from the settings that are stored in your `.pts2`-file, so make sure that it is

saved, and that the “Laue group for scaling” value in the “Finalize integration” step is correct. This should be verified by looking at the output from SHELXT.

Bibliography

[1] A. Teien, Optimizing the Data Collection and Data Processing Procedure for 3D Electron Diffraction Crystal Structure Analysis, 2024. https://hdl.handle.net/11250/3151292 (accessed December 11, 2024).

[2] G.M. Sheldrick, A Short History of SHELX, Acta Crystallographica Section A: Foundations of Crystallography 64 (2008) 112–122. https://doi.org/10.1107/S0108767307043930.

[3] G.M. Sheldrick, SHELXT – Integrated Space-Group and Crystal-Structure Determination, Acta Crystallographica Section A: Foundations and Advances 71 (2015) 3–8. https://doi.org/10.1107/S2053273314026370.

[4] G. Steciuk, L. Palatinus, J. Hrdá, PETS2 Process Electron Tilt Series User Manual, (n.d.). http://pets.fzu.cz/download/pets2_manual.pdf.

[5] T. Hahn, H. Fuess, T. Hahn, H. Wondratschek, U. Müller, U. Shmueli, E. Prince, A. Authier, V. Kopský, D.B. Litvin, M.G. Rossmann, E. Arnold, S. Hall, B. McMahon, eds., International Tables for Crystallography: Space-group Symmetry, 1st ed., International Union of Crystallography, Chester, England, 2006. https://doi.org/10.1107/97809553602060000100.