XChemExplorer

Manual

Current version: v1.2.1

Date

20/06/2018

Author

Tobias Krojer

tobias.krojer@sgc.ox.ac.uk

Table of Contents

Introduction 3
Reference 3
Getting started 4
Prerequisites 4
Installation 4
Usage 5
Compatibility 5
Notes 5
Acknowledgements 5
Settings 6
Preferences 8
Database 9
Database update 9
Dataset reprocessing 14
Running xia2 or dials 14
Getting the results into the database 16
Initial Map Calculation 18
PanDDA 22
Suggested Workflow 22
Reference model building 22
pandda.analyse 22
pandda.inspect 22
PanDDA model export to the project directory 22
Refinement 23
Refinement stages 23
Overview 24
Deposition 28
Adding compulsory information 28
Preparation of files for PDB upload 30
Using XChemExplorer outside the labxchem environment 32
Project Directory structure 32
Updating the database 33
Running PanDDA 33
Refinement 33
Deposition 34
Frequently asked questions 35
What happens when you step through the models? 35
What happens when I make changes to the model? 35
What happens when I refine the model? 35
Known Problems 36

Introduction

XChemExplorer (XCE) is a data management and workflow tool, which supports large-scale analysis of protein-ligand structures by X-ray crystallography. It is not an actual algorithm, but serves as a launch pad for batch submission and analysis of the essential steps in the structure determination of protein-ligand structures.

Reference

Krojer, T., Talon, R., Pearce, N., Collins, P., Douangamath, A., Brandao-Neto, J., Dias, A., Marsden, B., and von Delft, F. (2017). The XChemExplorer graphical workflow tool for routine or large-scale protein–ligand structure determination. *Acta Cryst D 73*, 267–278.

XChemExplorer makes extensive use of other people's software, therefore please cite their work accordingly:

- XIA2
- DIMPLE
- ACEDRG
- PanDDA
- COOT
- REFMAC
- PHENIX
- MolProbity
- GRADE

Getting started

Prerequisites

XCE works on any Mac OSX or Linux system, but it is essential that CCP4 (www.ccp4.ac.uk) version 7.0 or higher is installed and correctly configured. XCE uses the python version that comes with it and will therefore not work if it does not exist. Additionally, it may be useful to install PHENIX, since XCE uses several of its tools for validation purposes.

Installation

Download XChemExplorer from http://tkrojer.github.io/XChemExplorer

Put the gzipped tar archive to wherever you want XCE to be installed. In case you have no root privileges, put it somewhere into your home directory, e.g.:

/home/tkrojer/software

Then change to the respective directory and unpack the archive, e.g.:

cd /home/tkrojer/software

gunzip XChemExplorer-1.2.tar.gz

tar -xvf XChemExplorer-1.2.tar

This will create a new directory, i.e. the XChemExplorer directory. Change into this directory, e.g.:

cd XChemExplorer-1.2

The contents of the directory should look something like this when you type '**ls -l**':

```
-rwxr-xr-x 1 tkrojer users 238K Jun 18 13:58 XChemExplorer.py
-rwxr-xr-x 1 tkrojer users  269 Jun 18 13:58 XChemExplorer_local.sh
-rwxr-xr-x 1 tkrojer users  316 Jun 18 13:58 XChemExplorer_dmd.sh
drwxr-xr-x 3 tkrojer users 4.0K Jun 18 13:58 web
-rwxr-xr-x 1 tkrojer users  553 Jun 18 13:58 setupssh.sh
drwxr-xr-x 2 tkrojer users 4.0K Jun 18 13:58 setup-scripts
-rwxr-xr-x 1 tkrojer users 2.8K Jun 18 13:58 README.md
drwxr-xr-x 2 tkrojer users 4.0K Jun 18 13:58 lib
drwxr-xr-x 2 tkrojer users 4.0K Jun 18 13:58 image
drwxr-xr-x 2 tkrojer users 4.0K Jun 18 13:58 icons
drwxr-xr-x 2 tkrojer users 4.0K Jun 18 13:58 helpers
drwxr-xr-x 2 tkrojer users 4.0K Jun 18 13:58 gui_scripts
-rwxr-xr-x 1 tkrojer users  182 Jun 18 13:58 Dockerfile
-rwxr-xr-x 1 tkrojer users  465 Jun 18 13:58 compile_test.py
lrwxrwxrwx 1 tkrojer users   20 Jun 18 14:33 XChemExplorer -> XChemExplorer_dmd.sh
```

The only thing left to do is to edit the XChemExplorer_dmd.sh file with a text editor. Change the line

```
export XChemExplorer_DIR='/usr/local/scripts/tobias/XChemExplorer'
```

to where you XChemExplorer is installed. In our example this would be

```
export XChemExplorer_DIR='/home/tkrojer/software/XChemExplorer-1.2'
```

That's it!

Limitations: XCE does currently not have a dedicated update mechanism. It is possible to clone the current master branch of the XCE github repository (https://github.com/xchem/XChemExplorer) and then frequently pull from it!

Usage

You can now run XCE by typing

```
/home/tkrojer/software/XChemExplorer-1.2/XChemExplorer_dmd.sh
```

It may however be easier if you insert an alias into your .bashrc or .cshrc file:

```
alias XChemExplorer='/home/tkrojer/software/XChemExplorer-1.2/XChemExplorer.sh
```

Compatibility

XCE does not have a problem with compatibility between different versions. This is because the program is essentially file system centred. Please see the original publications for more information.

Notes

The terms *datasource* and *database* are used somewhat interchangeably in the XCE GUI and in this manual.

Acknowledgements

XCE was originally written by me, but the program is currently being co-developed and maintained together with Rachael Skyner. Elliot Nelson contributed significantly to the design and development of the PanDDA refinement module. Frequent feedback from our amazing colleagues at the SGC and Diamond Light Source and input from the XChem user community was of utmost importance for software development. Special thanks goes to Romain Talon, Patrick Collins, Alice Douangamath, Jose Brandao-Neto and Alexandre Dias. The Research Informatics team at the SGC was instrumental for implementing the generation of HTML summary pages. The team at RCSB Rutgers have massively helped to support the group deposition mechanism. Finally, this work would not have been possible without the generous support from various SGC funders: the SGC is a registered charity (No. 1097737) that receives funds from AbbVie, Bayer, Boehringer Ingelheim, the Canada Foundation for Innovation, the Canadian Institutes for Health Research, Genome Canada, GlaxoSmithKline, Janssen, Lilly Canada, the Novartis Research Foundation, the Ontario Ministry of Economic Development and Innovation, Pfizer, Takeda and the Wellcome Trust (092809/Z/10/Z).

Settings

The settings tab (Figure 1) contains information where XChemExplorer (XCE) will write files to and where it can find certain files. It also tells XCE the name and location of the SQLite database file.

If you use XCE at DLS as part of an XChem project and open it somewhere in /dls/labxchem/… then you will usually not have to change anything. XCE will populate the directories and information about the SQLite database file with defaults.

The situation is different when you leave the labxchem environment; now you need to specify the information. However, there are only 3 pieces of information required to get started:

Project directory

This is where all files that XCE creates will end up. The project directory contains a sub-directory for every crystal and the name of every sub-directory corresponds to the *CrystalName* field in the database. Please note that the names can be completely arbitrary, although it is highly recommended to choose meaningful names. Please, check the XCE publication for more information about the structure and filename conventions of the project directory.

Reference Structure Directory

You can provide reference PDB, MTZ and CIF files in this directory. These files will be used for map calculation with DIMPLE or for selection of the best auto-processing result. The filenames can be arbitrary, however, if the files belong together they need to have the same root.

Data Source

Here you need to specify the database file that you want to use. If you are starting from scratch, please check the next section, which explains how to create a new database file

CCP4_SCR director

This is the same as your CCP4_SCR directory. It is essentially a scratch directory, which XCE uses to save input scripts for processing, restraints generation and refinement to. Usually one can ignore this directory, however, it is a good place to start trouble-shooting in case a job did not behave as expected.

Group Deposition Directory

This is the directory where XCE will write the gzipped tar archive file for upload into the Protein Data Bank.

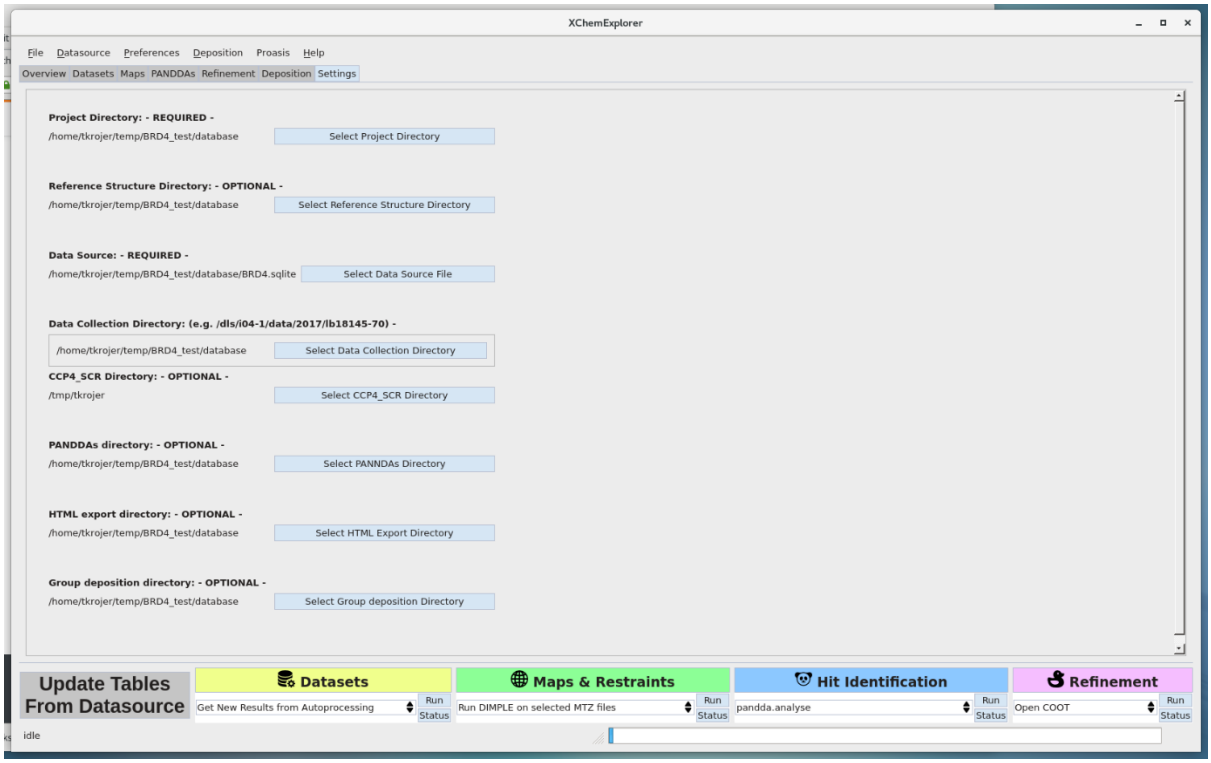


Figure 1. The settings tab in XChemExplorer.

Preferences

--- coming soon ---

Database

XCE uses a simple SQlite database to capture information, results and outcomes that are generated during the project. If your samples were collected at the XChem facility at the Diamond Light Source, then you will usually just take the soakDB file that was created during crystal preparation. Otherwise, select '*Create New Datasource (SQLite)*' from the *Datasource* menu and create a new database (Figure 2).

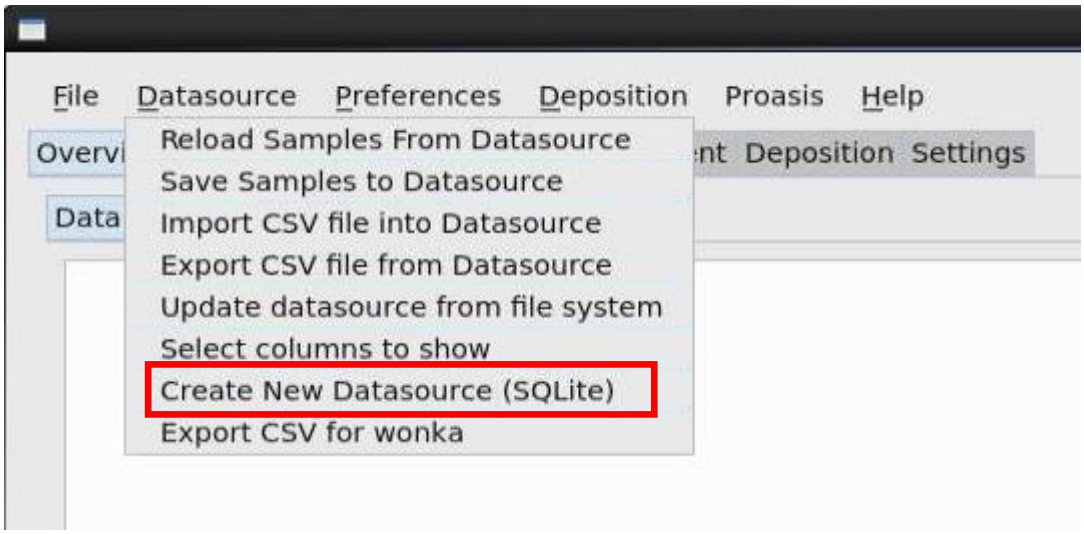


Figure 2. Datasource menu item 'Create New Datasource (SQLite)'.

Database update

There should be no need to make changes to the database when working on an XChem project. One notable exception is the entry of information about soaked/ co-crystallised compounds in case these compound are not part of the XChem fragment libraries. The initial ambition was to have a database update functionality available as part of XCE, but this approach will not be pursued for the time being. The *Datasource* menu contains some rudimentary functionalities for database changes, but it is recommended to follow the instructions outlined here in case the need for database manipulation arises. Also, keep in mind that the tables in XCE only display the contents of the database. Changing the value of the fields, although it is possible, has not effect on the database! A single click on '*Update Table From Datasource*' will bring back the actual content of the database.

Software

There are several programs for manipulation of SQLite files, the one we typically use is SQLite Browser (https://sqlitebrowser.org) (Figure 3). It is available for Windows, Mac and Linux.

Instructions

It is easily possible to make all the required changes directly within SQLite Browser, but this really only make sense if you need to change a few fields. For large-scale data entry, you should export the

respective table (usually only the *mainTable*) into CSV format and then make the required changes in Microsoft Excel. Other programs like OpenOffice should work as well, however, there have been anecdotal reports of import problems.

Before you start, make a copy of the current database in case something goes wrong!!!

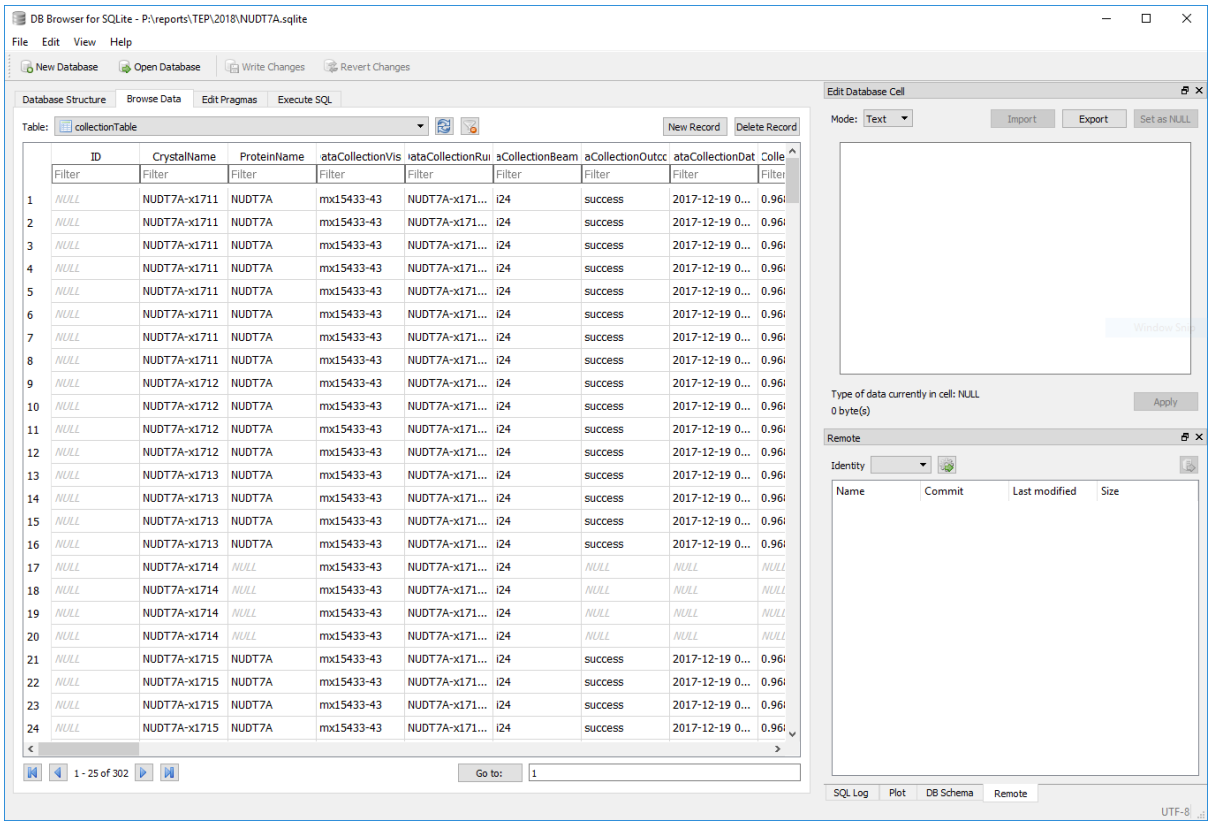


Figure 3. SQLite browser main window.

Next, open the sqlite file and export the export *mainTable* to a CSV file:

File -> Export -> Table(s) as CSV file…

Save as *mainTable.csv* if asked. Make sure that the box '*Column names in first line*' is checked (Figure 4).

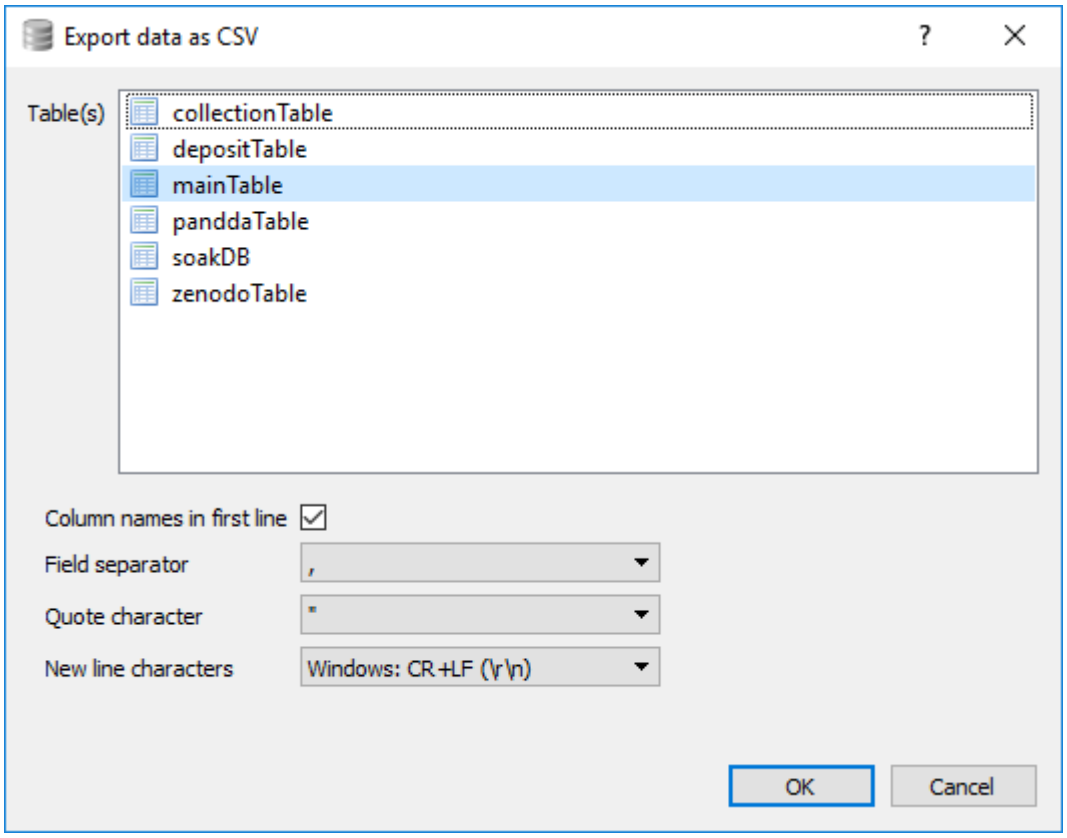


Figure 4. SQLite browser - export table to CSV file.

Next, delete the *mainTable* from database. In the '*Database Structure*' tab, highlight the table you want to delete, then, go to Edit -> Delete Table (Figure 5). Press 'Write Changes' and close the file.

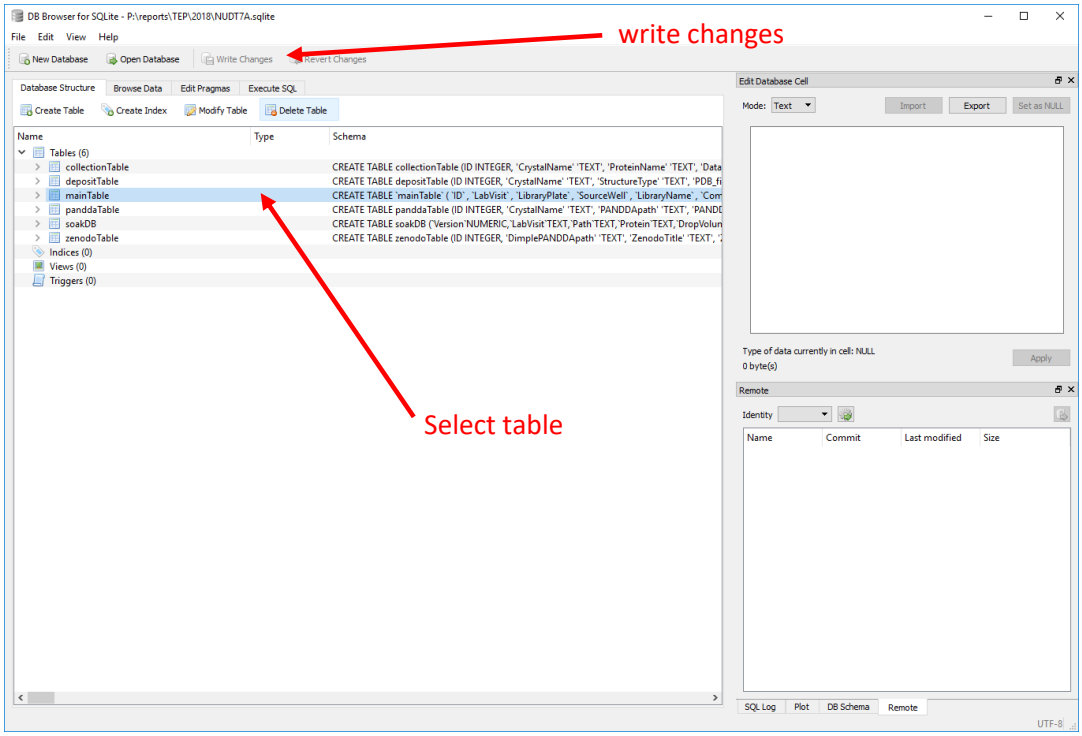


Figure 5. SQLite browser - select table for deletion.

Now open the CSV file in Microsoft EXCEL and make changes as necessary. The fields that most often will need changing are *CompoundSMILES* and *CompoundCODE*.

Open the SQLite file again and select File -> Import -> Table from CSV file…

Make sure the table name is *mainTable* and that '*Column names in first line*' is checked (Figure 6).

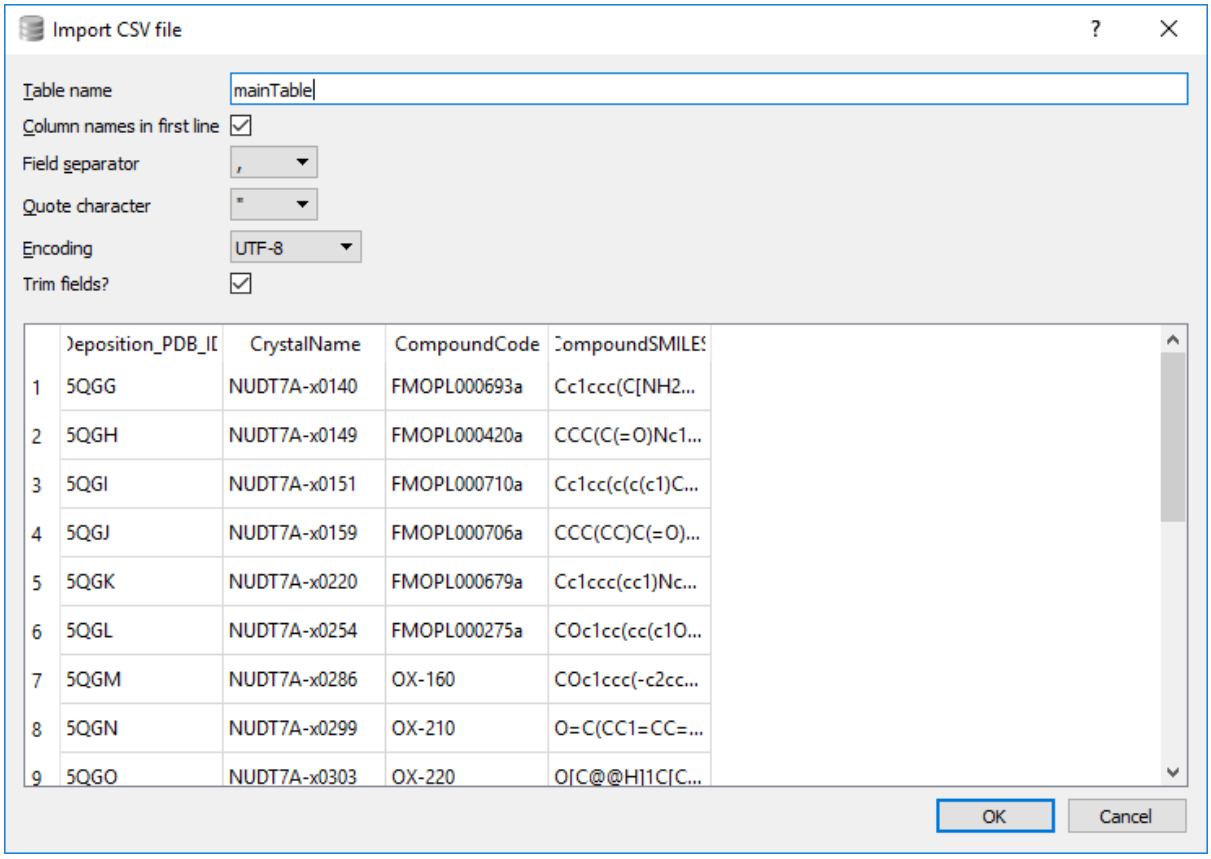


Figure 6. SQLite browser - import CSV file as table into SQLite database.

Dataset reprocessing

Running xia2 or dials

XCE can be used to reprocess datasets with either XIA2 or DIALS. This option is available in the Datasets/ Reprocess tab (Figure 7).

First, select the datasets directory, then, press '*Search Datasets*'. Note that if you want to process multiple-datasets then select the respective top-level directory under which the individual datasets are stored.

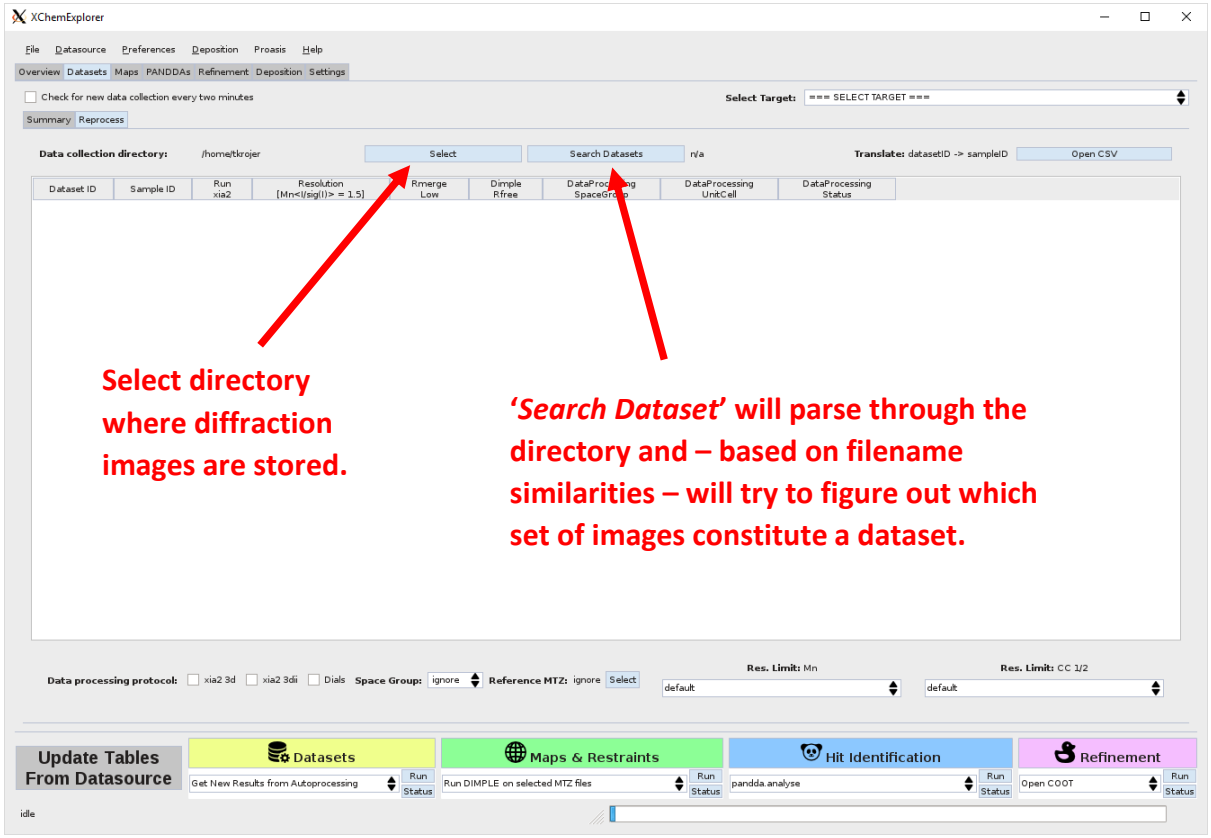


Figure 7. (Re-) processing of diffraction data in XCE.

Depending on the number of datasets, parsing of the respective directory may take a while. However, the progress bar will indicate how far along the program is and finally the table should get populated as in the exemplary screenshot below. There are a few things one should keep in mind:

- the *datasetID* is the name of each folder in the '*Data collection directory*'; e.g. all subfolders in /xdata/BRD4A

- only folders with more than 20 diffraction images in them will be listed

- XCE assumes that *datasetID=sampleID*; and if there is already an entry in the datasource for a particular *sampleID* then it will display things like Rmerge etc.

- you can change the sampleID simply by entering another value in the cell (Figure 8)

- if you want to use different sample IDs for many datasets, then you can use the '*Translate dataset ID -> sample ID*' option.

- you can either select all samples you want to run by clicking on the checkbox or you select a range of rows with the mouse, right-click, click on 'mark selected for reprocessing'

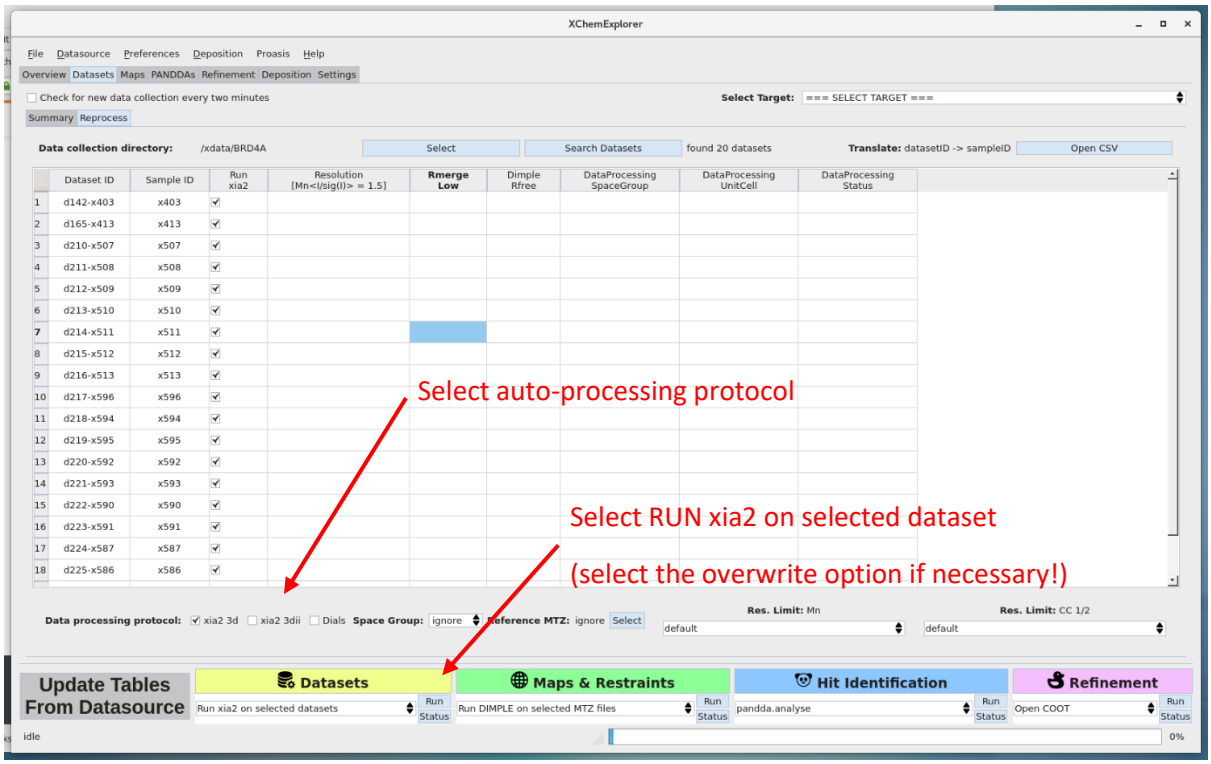


Figure 8. Dataset reprocessing tab after XCE found and assigned the diffraction images. Note that the values in the Sample ID column were manually entered. It would have been the same as in the Dataset ID column when XCE populated the table the first time.

Next, select the data processing protocol(s) you want to use (red arrow above), then select '*Run xia2 on selected datasets*' from the yellow *Datasets* actionbox and press '*RUN*'. Note that all data processing results will end up in the project directory.

If you want to provided reference files then put the respective files into the reference directory.

Please note that if your local machine is not automatically able to submit jobs to a computer cluster via qsub, it will process the datasets sequentially on the local machine. Needless to say that this may keep the machine busy for a while in case you want to process tens or even hundreds of datasets. In this case, it is advisable to use only one data processing protocol.

Please also note that XCE does currently not indicate when the jobs are finished (v1.2). You need to check the workload on your machine to find out.

Getting the results into the database

Now you need to get the data processing results/ outcomes back into the database. Stay in the main *Datasets* tab, but switch to the *Summary* sub-tab (Figure 9). Go to the select targets dropdown and select '*=== project directory ===*'. Select '*Get New results from Autoprocessing*' from the yellow action box and press *RUN*.

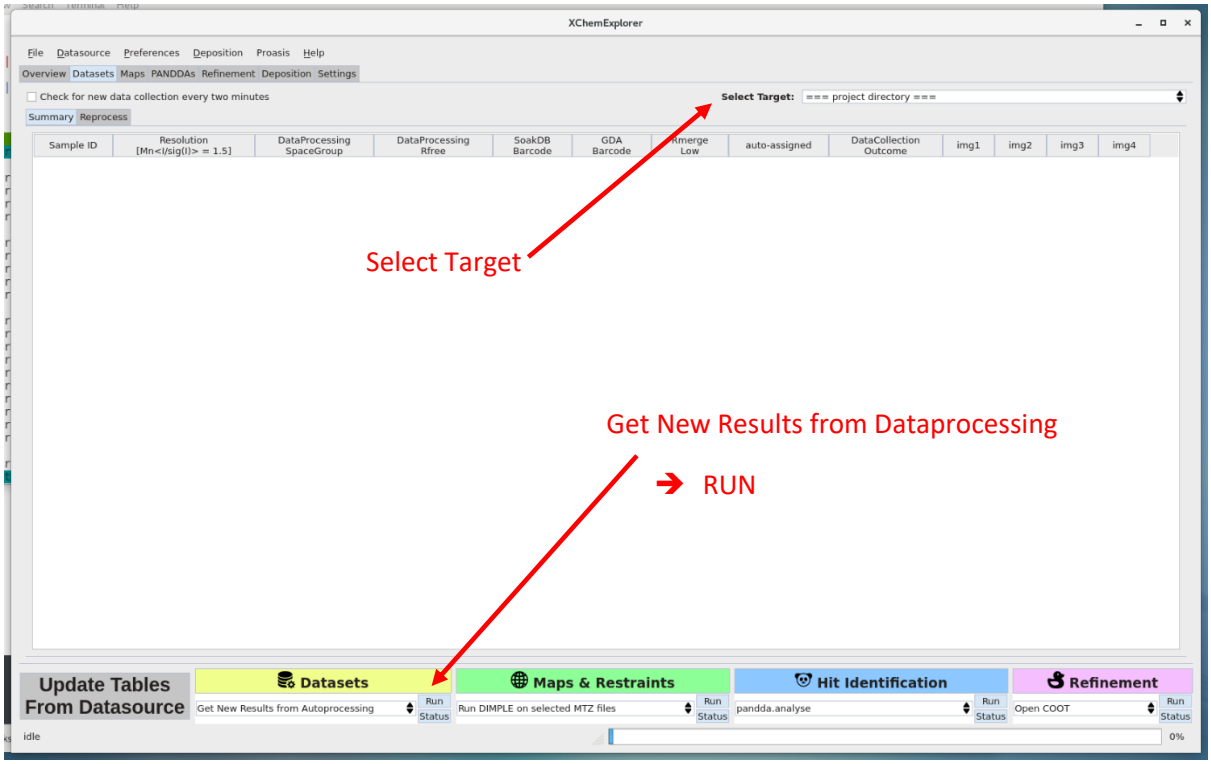


Figure 9. (Re-) processing of diffraction data in XCE. Reading the results.

If everything worked well, each sample directory with a successful data processing outcome should look something like this:

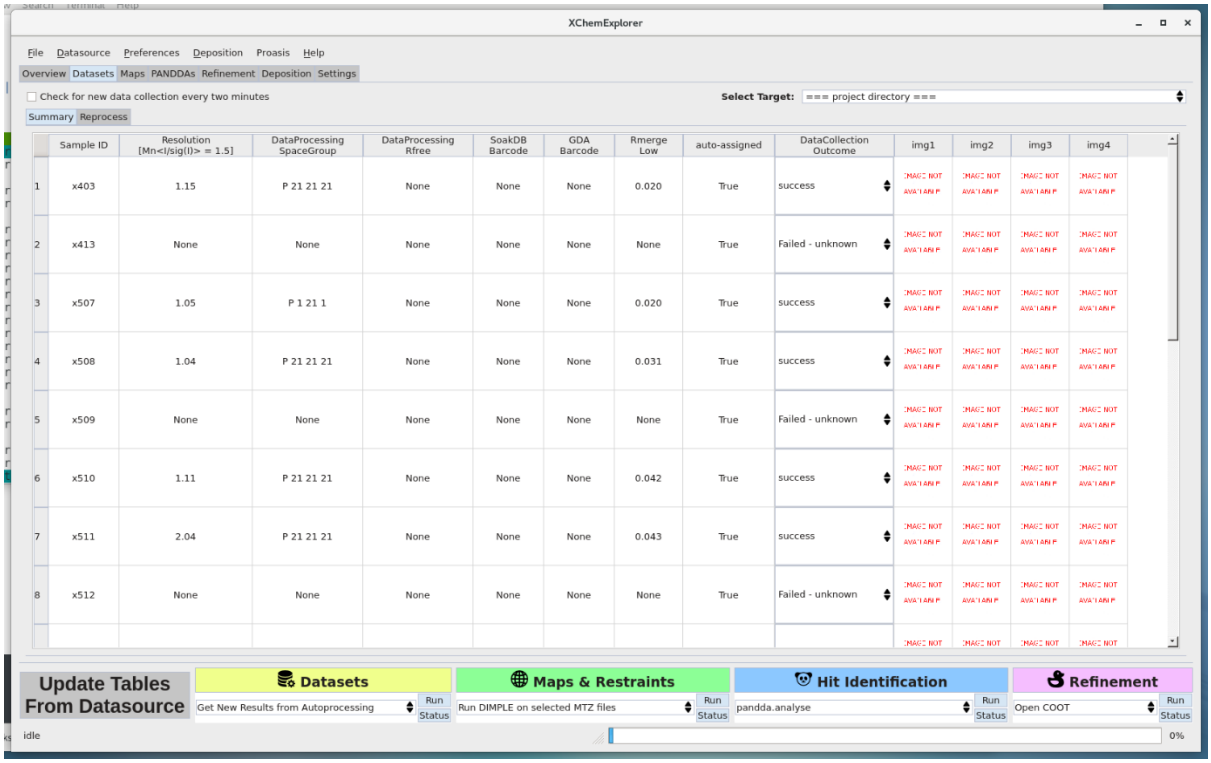


Figure 10. Results from reproccsing with xia2.

You can check the respective sample directories; as you can see below, there should be symbolic links to the *MTZ* and *AIMLESS* logfile from xia2; and the files have the same root as the respective sample directory.

```
drwxr-xr-x 3 tkrojer users 4.0K Jun 18 13:37 diffraction_images

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 14:42 processed

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 15:50 jpg

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 15:50 autoprocessing

lrwxrwxrwx 1 tkrojer users   58 Jun 18 15:50 x596.mtz -> autoprocessing/database-run_13d/AUTOMATIC_DEFAULT_free.mtz

lrwxrwxrwx 1 tkrojer users   61 Jun 18 15:50 x596.log -> autoprocessing/database-run_13d/AUTOMATIC_DEFAULT_aimless.log
```

Initial Map Calculation

XCE uses DIMPLE to perform an initial round of refinement and to calculate the resulting 2fofc and fofc maps.

Please note that the tables in XCE do no update automatically! It does so during startup, but they will not refresh automatically. Also, there is currently a bug which means that the status indicators in the MAPS table are not always up to date (v1.2).

If you have successfully processed your datasets or read in the results from DLS auto-processing, press on the grey '*update tables from datasource*' button on the lower left hand corner (Figure 11). You can press this button as often as you want and it will update all tables with the exception being the Datasets Summary table with the latest information from the database.

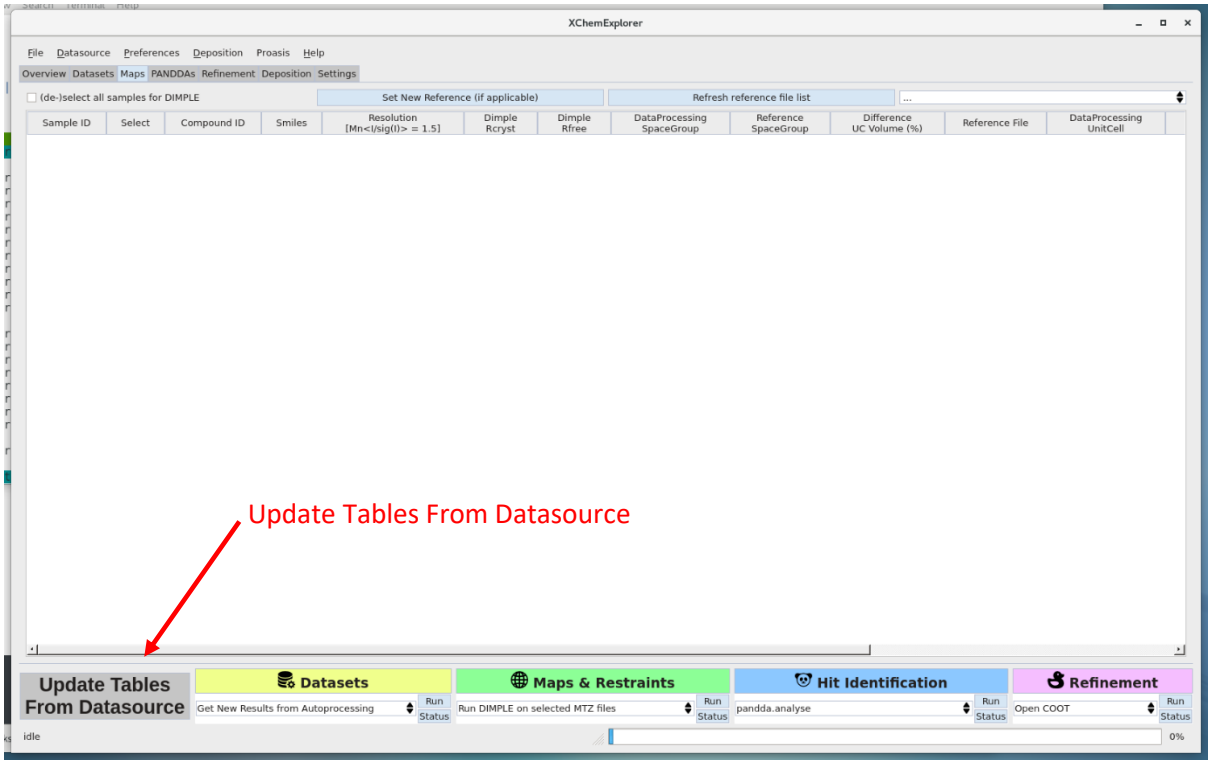


Figure 11. Update tables from datasource. All tables in XCE are currently not updated automatically and need active pulling from the database.

In our current example, the table will look like this:

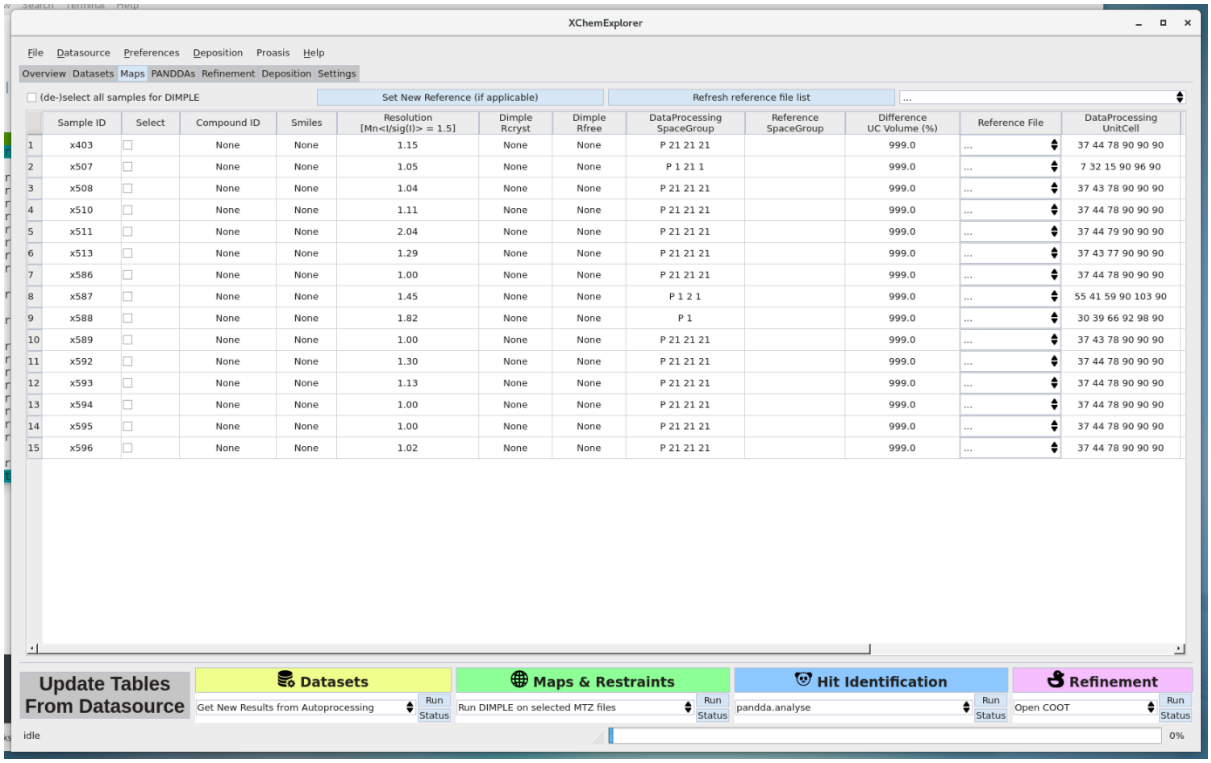


Figure 12. Populated Maps table.

We can now see information about the high resolution limit and space group for each crystal in the table, but the information about which reference PDB file to use is still empty. This is because in the example I have not yet provided a suitable reference PDB file in the reference directory. After this is done, one needs to press '*Refresh reference file list*', then select the reference file that you want to use, and press '*Set New Reference (if applicable)*'.

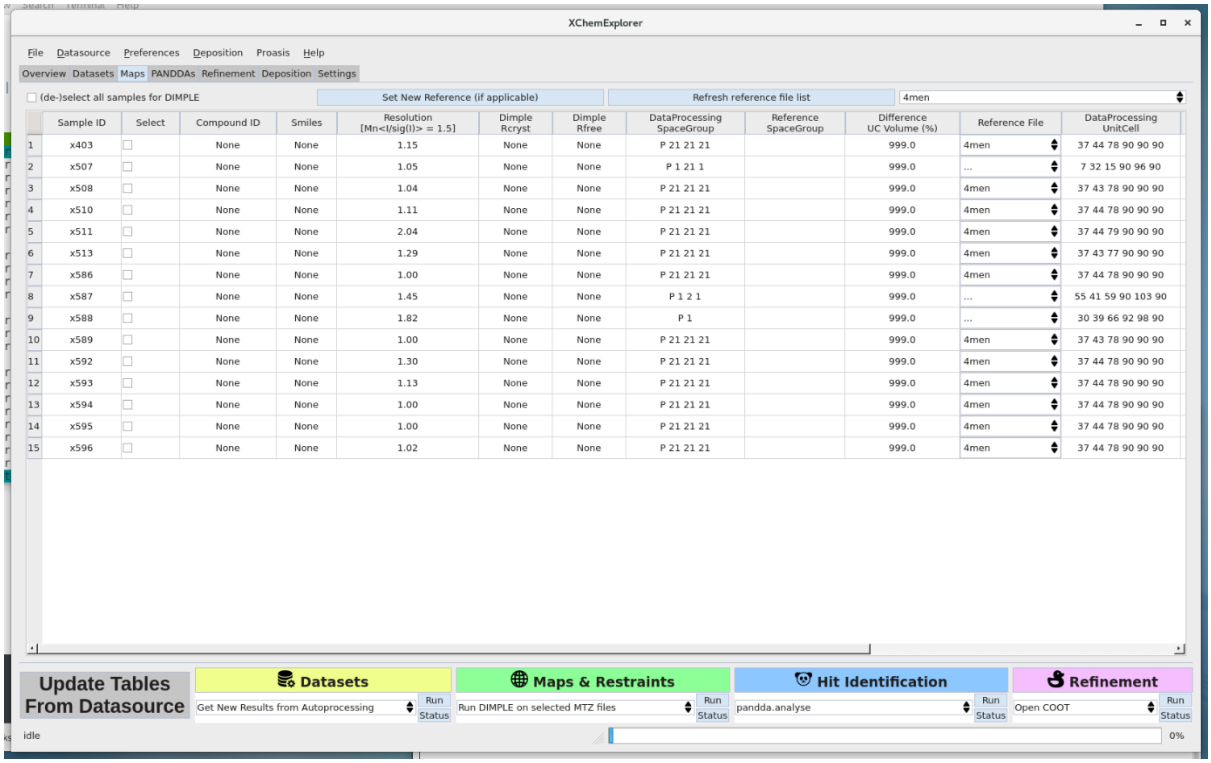
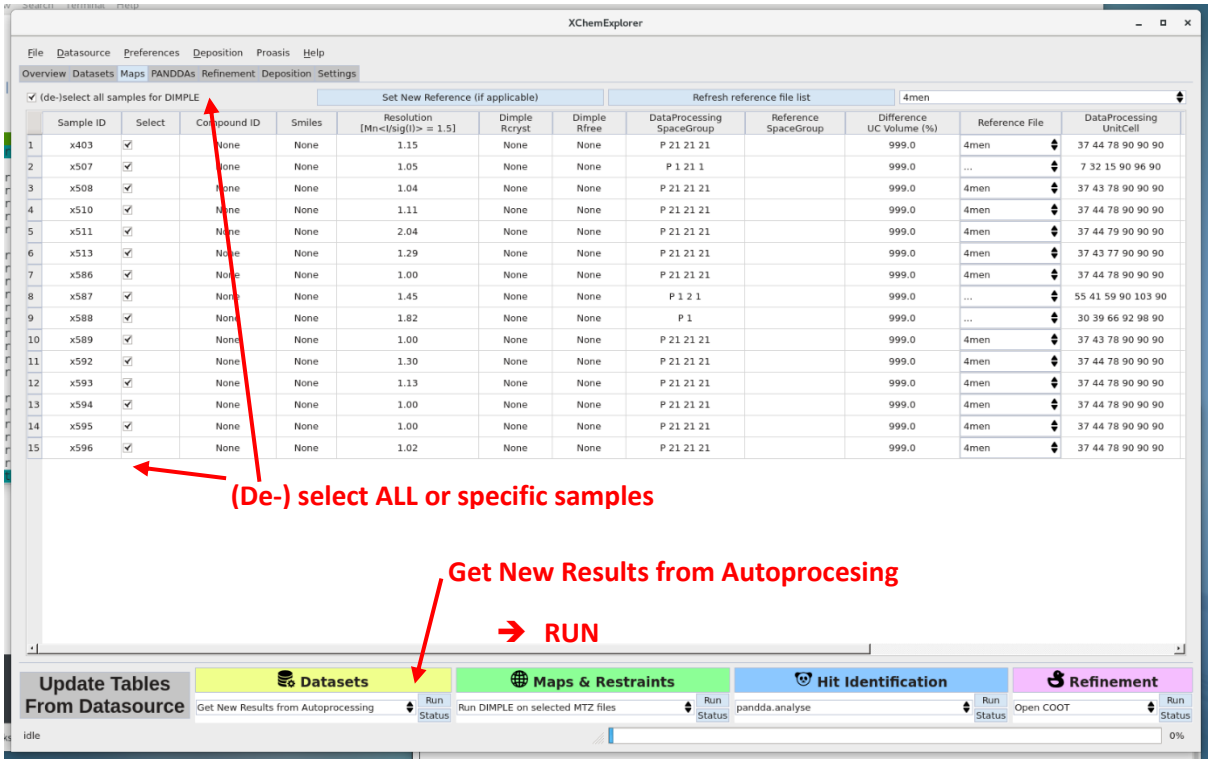


Figure 13. Maps table after reference files for DIMPLE have been assigned.

You can check the XCE publication for more information about how XCE selects which reference file to use. Briefly, XCE will go through all PDB files that you have provided in the reference directory and read the CRYST card in their header. From the CRYST card, it will calculate the unit cell volume and determine the point group. Then it will compare this to the unit cell volume and point group of your crystal. If the unit cell volume differs by less than 12% (this can be adjusted in the *Preferences* menu) and the two have the same point group, then XCE will consider this to be a suitable input file for Dimple. This selection mechanism works well as long as you have either different point groups or the unit cell volume differs significantly between different space groups. It will most likely struggle in cases where you have different crystal forms that are not detectable by the rather coarse selection mechanism! However, I have often found that if XCE does not find a suitable reference file it is recommended to manually check what is going on. It may be that the presence of a certain ligand triggered some change that is best investigated outside the XCE workflow. If it really is a different crystal form, solve the structure as you usually would, then add the resulting structure to your reference file directory.

Now select which crystals you want to process (or simply choose *select all samples for DIMPLE*) and then choose '*Run DIMPLE on selected MTZ files*' from the green action box and press '*RUN*'.



If everything worked as expected, then then respective sample directory should now look like this:

```
drwxr-xr-x 3 tkrojer users 4.0K Jun 18 13:37 diffraction_images

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 15:15 processed

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 15:50 jpg

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 15:50 autoprocessing

lrwxrwxrwx 1 tkrojer users   58 Jun 18 15:50 x403.mtz -> autoprocessing/database-run_13d/AUTOMATIC_DEFAULT_free.mtz

lrwxrwxrwx 1 tkrojer users   61 Jun 18 15:50 x403.log -> autoprocessing/database-run_13d/AUTOMATIC_DEFAULT_aimless.log

drwxr-xr-x 3 tkrojer users 4.0K Jun 18 16:18 dimple

lrwxrwxrwx 1 tkrojer users   53 Jun 18 16:21 dimple.pdb -> dimple/dimple_rerun_on_selected_file/dimple/final.pdb

lrwxrwxrwx 1 tkrojer users   53 Jun 18 16:21 dimple.mtz -> dimple/dimple_rerun_on_selected_file/dimple/final.mtz

lrwxrwxrwx 1 tkrojer users   53 Jun 18 16:21 2fofc.map -> dimple/dimple_rerun_on_selected_file/dimple/2fofc.map

lrwxrwxrwx 1 tkrojer users   52 Jun 18 16:21 fofc.map -> dimple/dimple_rerun_on_selected_file/dimple/fofc.map

-rw-r--r-- 1 tkrojer users 1.3M Jun 18 16:21 x403.free.mtz
```

You can now either run *PanDDA* or move straight to the Refinement tab and look at your electron density maps after initial refinement.

PanDDA

XCE can be used to conveniently launch the PanDDA software. However, this is not necessary! One can also run *pandda.analyse* and *pandda.inspect* from the command line and only later, during the *pandda.export* stage use XCE to bring the results back into the XCE folder structure for further refinement. Also, keep in mind that PanDDA does not know about the XCE database! PanDDa only looks at the files in the specified data directory, which is the XCE project directory. The actual PanDDA analysis is performed in the PanDDA directory.

Suggested Workflow

--- coming soon ---

Reference model building

--- coming soon ---

pandda.analyse

--- coming soon ---

pandda.inspect

--- coming soon ---

PanDDA model export to the project directory

--- coming soon ---

Refinement

Refinement stages

Figure 14 Refinement progress model of XCE

The figure summarises the refinement stage model that XCE uses to track the progress of each sample and which is also used to triage samples. You need at least do some initial refinement to be able to look at samples in the refinement interface.

Overview

A list of all models that are currently ‘in refinement’ can be viewed in the ‘Refinement’ tab.

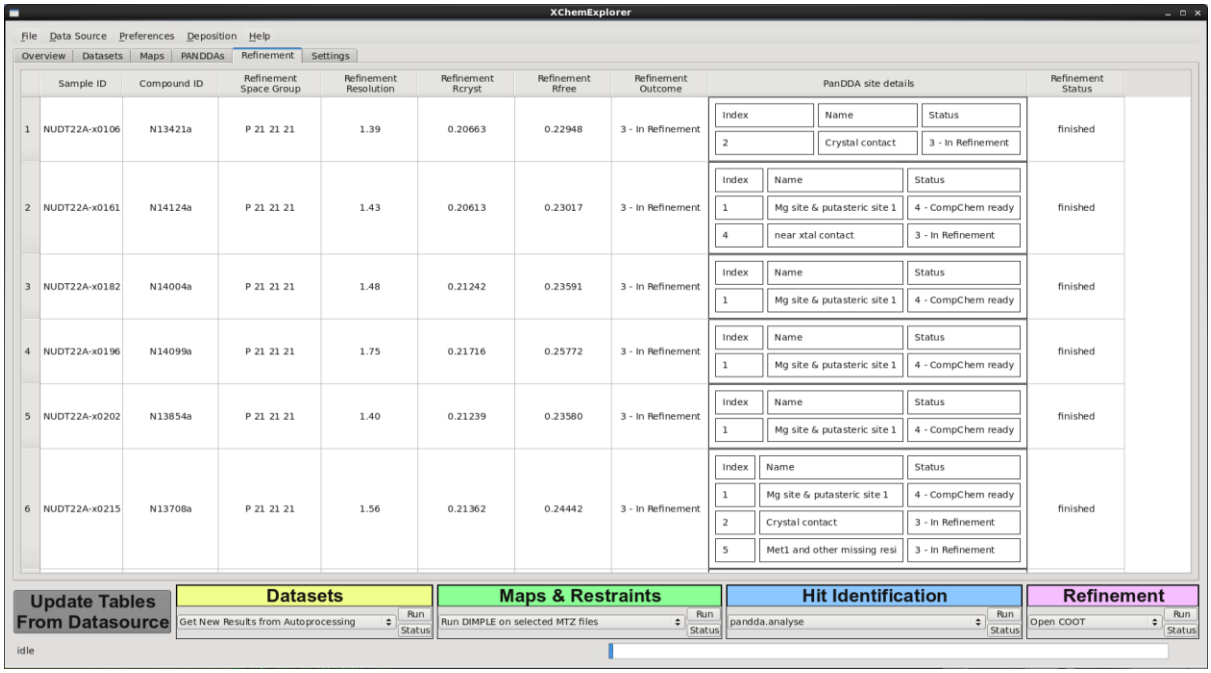


Figure 15 Refinement tab in XCE

If you want to inspect them in COOT and if necessary want to refine them further, choose ‘*Open COOT*’ from the magenta action box and press RUN. F

Figure 16 Refinement task box

This will launch COOT and the respective XCE interface (Figure 17).

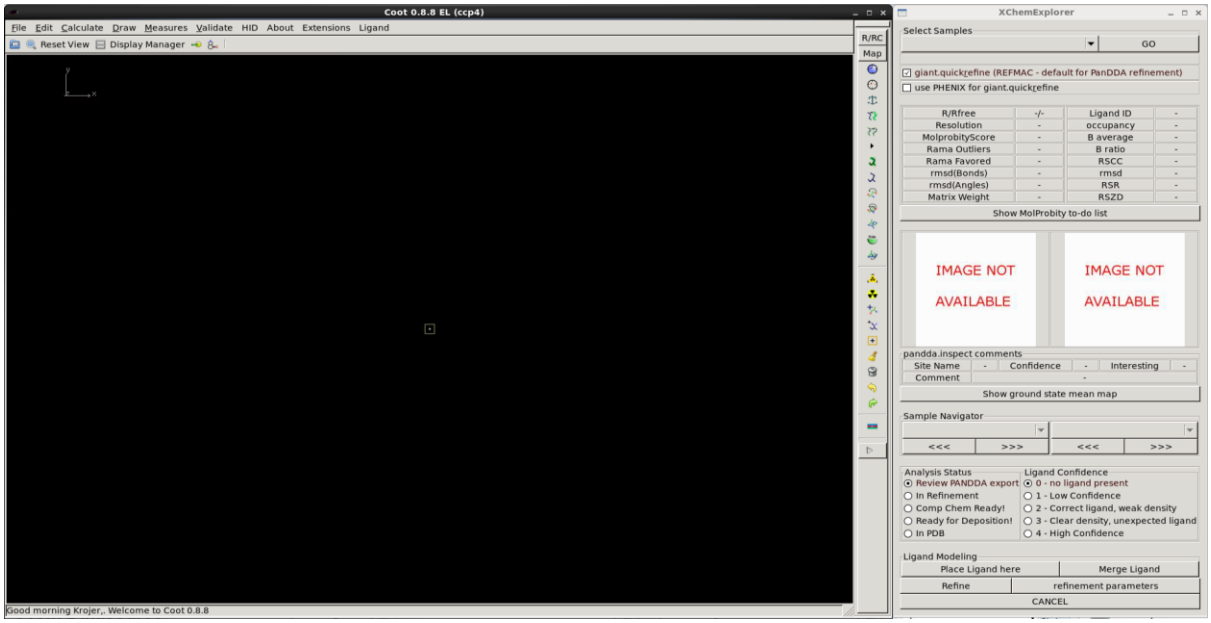


Figure 17 XCE Refinement interface for COOT

First you need to select the subset of samples you want to review/ refine from the drop-down menu at the top of the XCE interface (Figure 17). The drop-down lets you choose the Refinement Stage and once you press GO will load all samples that are in the respective Refinement stage. The image below shows you all the available categories

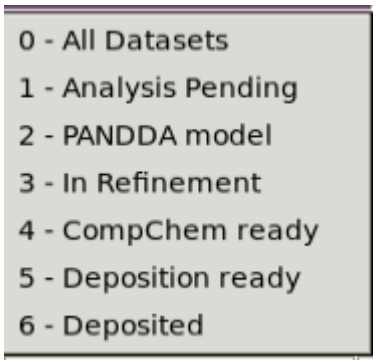


Figure 18 Sample selection criteria.

For example: if you want to load all models which are already in refinement, then select category 3 and press GO. In the example below, 79 structures are currently being refined (Figure 19).

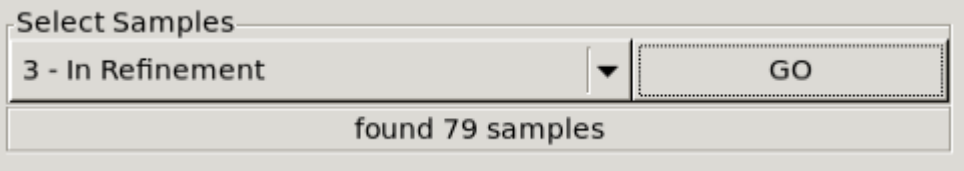
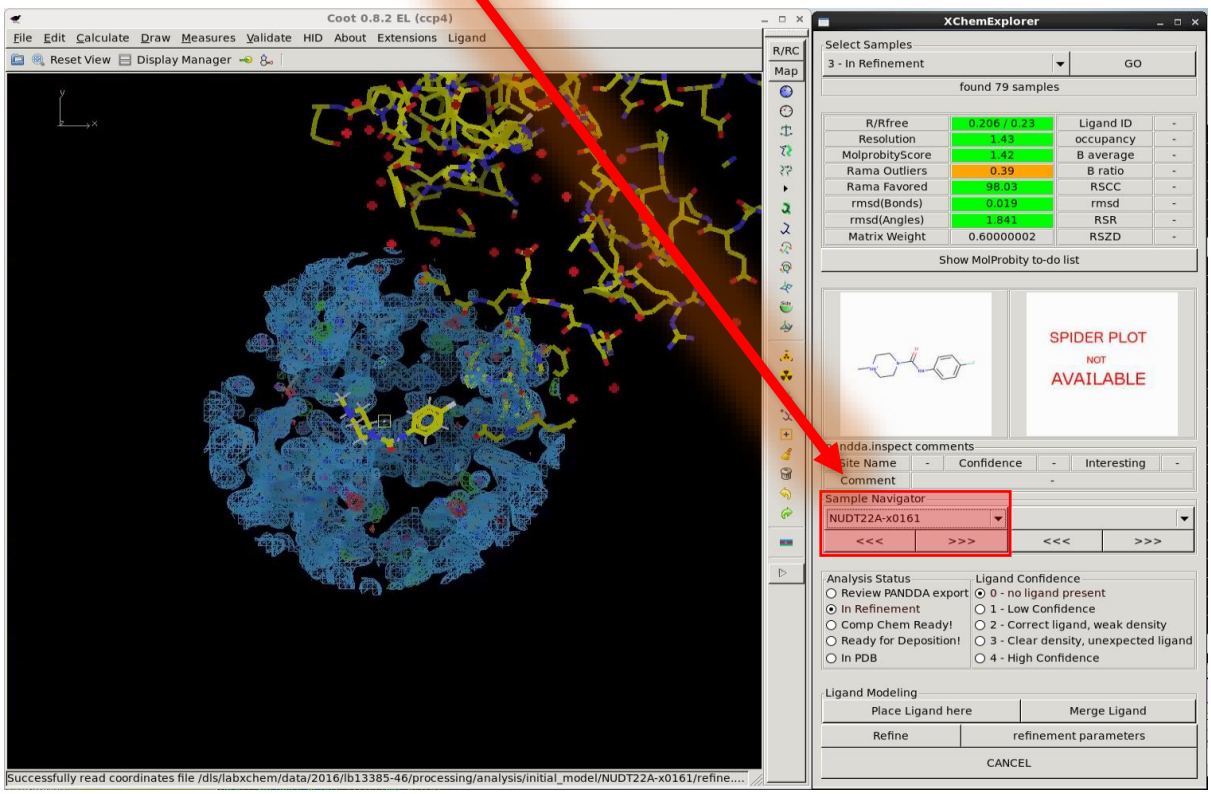


Figure 19 Exemplary reponse after sample selection.

In the left column of the 'Sample Navigator' Section, use the arrow buttons or the drop-down menu to select the structure of interest:



XCE will load the structure[1], 2fofc map (blue), fofc (green/red) map and a pdb file (+ dictionary) of the ligand if the latter was created and specified in the database. This ligand molecule may be slightly confusing because it may seem to just float in space. However, the molecule is completely

[1] Structure refers to the file called refine.pdb in the respective sample directory, or if no refinement has been carried out so far and category 0 or 1 are selected, then it will try to load dimple.pdb. Note: if you use XCE throughout the process, then refine.pdb as well as dimple.pdb are not actual files but symbolic links that point to the most recently refined file.

ignored as long as it is not merged into the main structure! It is only loaded to enable quick modelling if the ligand is not already part of the structure.

Deposition

XCE can be used to generate files for deposition via the new group deposition tool at the PDB (https://deposit-group-1.rcsb.rutgers.edu/groupdeposit) (Figure 20).

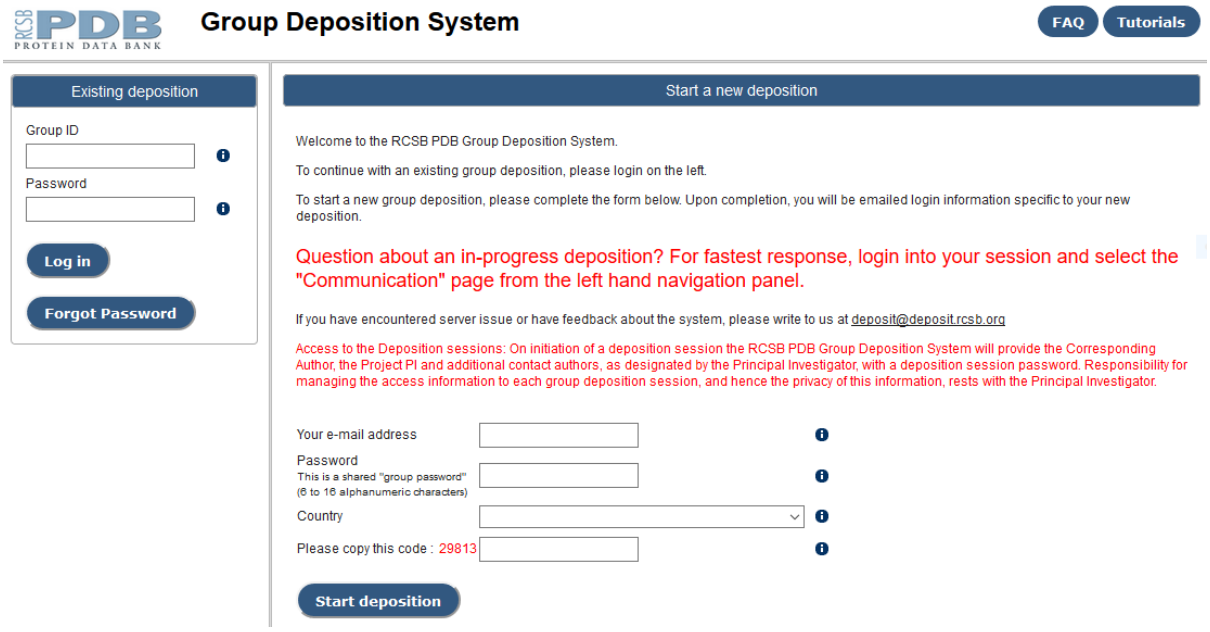


Figure 20. PDB Group Deposition website.

In order for a dataset to be considered for deposition, the *RefinementOutcome* flag has to be set to '*5 - Deposition ready*'. Once this is done, the respective sample will appear as a new line in the *depositTable* of the database.

Adding compulsory information

Every deposition needs information about title, authors, crystallization condition etc. This section describes how to enter this section. From the *Deposition* menu select '*Edit information*' (Figure 21).

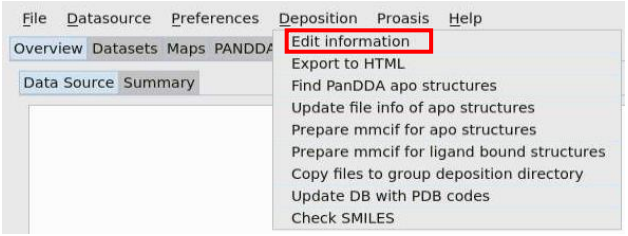


Figure 21. Deposition menu.

This will open a new window which provides a mask for input of all the essential pieces of information (Figure 22).

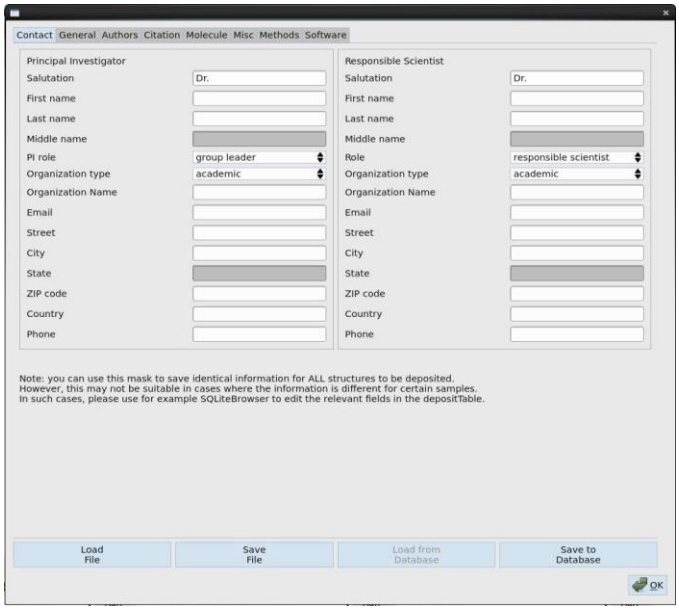


Figure 22. XCE entry widget for essential deposition information. Note that the compulsory fields have a white background whereas the fields where the user can provide additional information have a grey background.

It is possible to fill out all or part of the information and save the entered values into a file. Alternatively, it is possible to download an exemplary .deposit file from the XCE website (see Resources section in http://tkrojer.github.io/XChemExplorer). Download the file and select '*Load File*' and the fields of the widget should change:

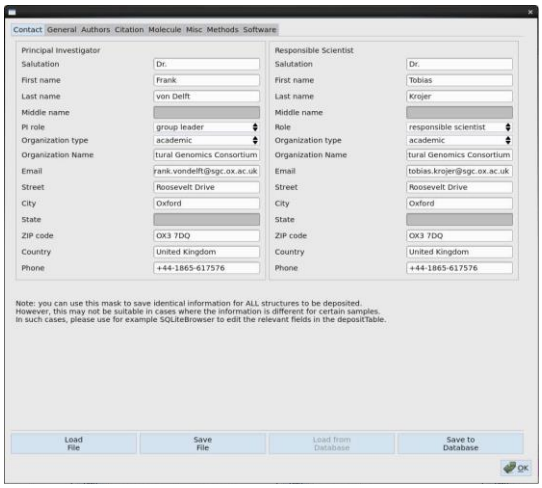


Figure 23. XCE entry widget for essential deposition information with data.

Most of the fields are self-explanatory, however, it is worth drawing the attention to the *General* tab where one can provide information about the title The given example contains two wildcards, *$ProteinName* and *$CompoundName*. The *$* sign indicates that XCE will fill in whatever is given as ProteinName and CompoundName in the database. While the former will most likely be the same for all the entries so that one can provide the specific protein name instead of a wildcard, it is useful to provide the compound name one wants to see in the title of the PDB deposition in the database. Once all the information is entered, use '*Save file*' to save a blueprint in case something needs changing, then press '*Save to Database*' in order to update all the entries that are currently in the deposition table. Note that if you add more samples later, you will again have to press again '*Save to Database*'.

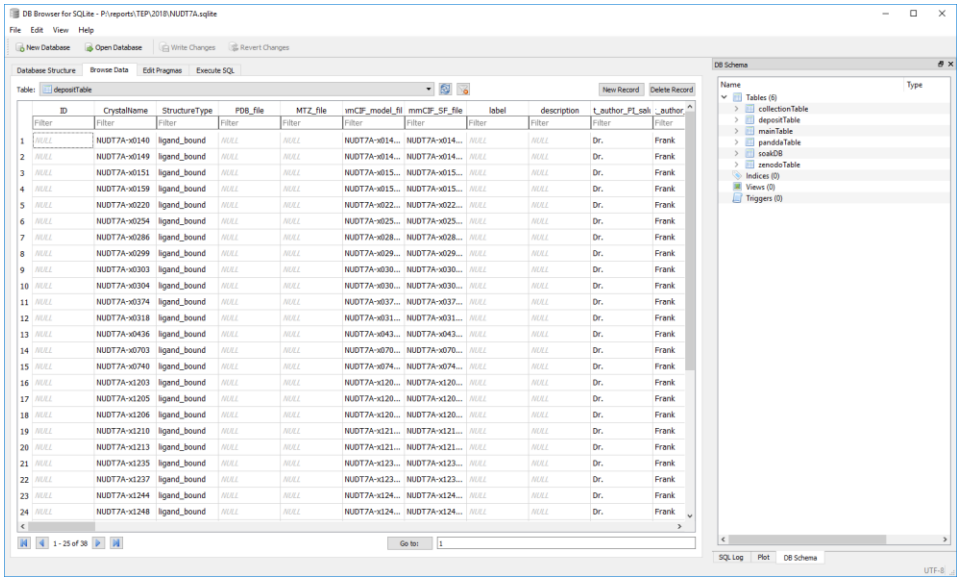


Figure 24. depositTable of the XCE database.

The interface obviously assumes that the information will be the same for every entry. But depending on the project, this may not be the case. Therefore, if you want to edit information for a specific entry, this is best done through the SQLite browser tool (Figure 1). Navigate to the *collectionTable* and then change the information as you see fit. Do not forget to press '*write changes*' once you have finished! But remember, if you add more samples later and you press '*Save to Database*', all the changed information will be overwritten. This is not ideal and will be addressed in one of the next software iterations.

Preparation of files for PDB upload

XCE will create a single gzipped tar file for direct upload into the Protein Data Bank. The *Group Deposition Directory* specifies the location where the file will be written to (see Settings, page 6). The next step is rather simple, from the *Deposition* menu choose '*Prepare mmcif for ligand bound structures*' and XCE will start the prepare separate mmcif files for the PDB file and MTZ files for each entry in the deposition table. Note that if this deposition is part of a PanDDA analysis, then the program will also add Fourier coefficients for the event maps to the structure factor mmcif file. This may take a while depending of the number of files that are earmarked for deposition.

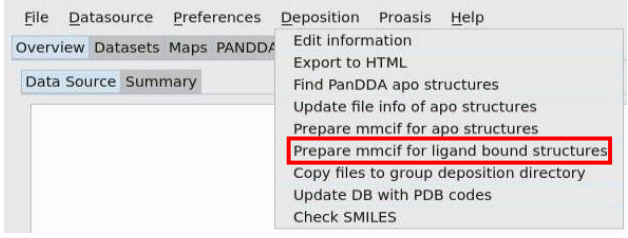


Figure 25. Prepare mmcif for ligand bound structures.

At the end, all mmcif files will be combined into a gzipped tar archive file, which can be directly uploaded into the PDB. At the moment, the only way to follow the progress of the process is by monitoring the output in the terminal window.

Troubeshooting

```
grep ERROR xce.log
```

Note: the XCE GUI does currently contain several obsolete items, which will be removed during the next software iteration.

Using XChemExplorer outside the labxchem environment

XCE was over the years optimised to support crystallographic fragment screening at the DLS XChem facility. It is however a generic tool and it will accept any data as long as they are organised in a certain hierarchical manner.

If you collect data at the Diamond Light Source, just make sure that you have set *${proteinname}/${samplename}* as your data collection directory and you are good to go. XCE is currently only able to parse the auto-processing results at DLS, but it should not be too difficult to adapt it to other synchrotron sites. Just get in touch in case you think this could be useful for your research!

The following section will describe what you need to do if your data are already processed and you want to import them into XCE.

Project Directory structure

XCE is essentially a file system based tool. It uses a SQLite database to record results and outcome and to track progress, but it can restore most of the information from the file system alone. Not everything of course, but enough to continue refining and depositing crystal structures. Therefore, as long as the data adhere to a few conventions and are present in the expected XCE directory structure, they can be imported. Please also check the **Database** section for how to generate and edit a new database.

All files are stored in the same project directory and have to be organised in a hierarchical manner, i.e. files belonging to one crystal are stored in a subfolder and the name of this folder is equal to the sample identifier. The filenames have to adhere to a naming convention, but the requirements are minimal and it is easy to add data manually by adjusting folder and file names to the expected nomenclature. All that needs to be provided are an MTZ and an AIMLESS logfile in the respective sample directory. The files must have the same filename root as the folder name, e.g. if the folder name is TEST-x001, then the MTZ and logfile must be called TEST-x001.mtz and TEST-x001.log, respectively.

The following example will illustrate the data structure further:

The project directory structure of XChemExplorer is as follows:

<project_directory>/<sample_id>

e.g.

/Users/tobiaskrojer/SGC/PHIPA/fragment_screen/PHIPA-x001

/Users/tobiaskrojer/SGC/PHIPA/fragment_screen/PHIPA-x002

/Users/tobiaskrojer/SGC/PHIPA/fragment_screen/PHIPA-x003

etc.

Each sample folder needs to contain a MTZ file and the corresponding AIMLESS logfile, e.g.:

PHIPA-x001.mtz

PHIPA-x001.log

XCE uses a file called <sample_id>.free.mtz as input for refinement. This file is either automatically generated by the DIMPLE difference map pipeline or users can choose to append an existing Rfree set from a reference file by providing it in the reference folder. If you have already done some refinement and you have already an Rfree set in your MTZ file, then just make a symbolic link, e.g.

```
ln –s PHIPA-x001.mtz PHIPA-x001.free.mtz
```

MTZ column labels must have CCP4 default names, otherwise XCE may show unexpected behaviour, i.e. IMEAN, SIGIMEAN, F, SIGF, FreeR_flag. Additionally, the program can only parse AIMLESS logfiles at the moment.

After DIMPLE was run successfully, the resulting PDB and MTZ files will be linked as dimple.pdb and dimple.mtz into the respective sample directory.

Once the refinement stage is reached a subfolder for each refinement cycle will be created: Refine_<cycle number>. This subfolder contains the modified PDB file, executable shell script and output. The script contains the complete refinement and validation schedule. After successful refinement, the resulting PBD and MTZ files will be linked as refine.pdb and refine.mtz into the sample directory. Again, you can provide the files yourself!

It is possible to create this folder structure manually and then choose Data Source -> Update Data Source from filesystem from the XCE menu to import all the information into the database.

This is all you need to do. It does not matter if these are real files or symbolic links. Exemplary sample folders could look like this once you are finished:

XXX

Updating the database

The database will already contain most of the information after you ran ‘Update Datasource from filesystem’. But if you want to edit or add more information like compound IDs or smiles strings (or anything else) then refer to Database update (page 9).

Running PanDDA

--- coming soon ---

Refinement

--- coming soon ---

Deposition

This chapter explains how to prepare data for deposition in case this is the only task you want to use XCE for. You need to prepare your data as described before, most importantly, the final PDB and MTZ files need to be present as *refine.pdb* and *refine.mtz* in the respective sample directories.

The only things left to do before you can continue with depositing your data as described in the Deposition (page 28), is to set the *RefinementOutcome* field in the *mainTable* of the database to '*5 - Deposition ready*'. This can be achieved by either manipulating the database with SQLite browser or by looking at your data in the XCE COOT interface and setting the *RefinementOutcome* field there.

Frequently asked questions

What happens when you step through the models?

XCE will load a file called refine.pdb (or dimple.pdb in case no refinement has been carried out so far) from the sample directory and if available a pdb file of the ligand and the respective restraints. Additionally, 2fofc as well as fofc maps are loaded (or they are calculated on the fly from refine.mtz/ dimple.mtz if the map files were for whatever reason no pre-calculated). Note that refine.pdb/mtz and dimple.pdb/mtz are therefore reserved file names. Hence, if you wanted to manipulate your model outside the XCE environment you can easily do so and XCE will read the manipulated model in as long as it's called refine.pdb and present in the respective sample directory. One thing to keep in mind though: XCE carries out the actual refinement in a subfolder called Refine_<cycle number> and only links the resulting pdb/mtz files as refine.pdb/mtz into the project directory. Every time a new refinement is launched, it will first delete the respective symbolic links. But it will also delete it if it is an actual file. So if you want to keep the original, better create a symbolic link called refine.mtz. When you go through the models, it will remove all currently loaded models and load the aforementioned files from the next sample directory.

What happens when I make changes to the model?

All changes that you make to the model called refine.pdb will be preserved if you launch refinement (see next point). They will however be lost if you go to the next dataset. XCE will currently not ask if you want to keep the changes, the pdb file will be deleted from the list of molecules in COOT and it will be lost forever! Also, be careful if you read in additional molecules with the same name, for example if you want to analyse something. COOT does not mind if molecules have the same name since every molecule that is read in get a unique internal identifier. XCE however recognises molecules by filename it may get confused in case of duplicates.

What happens when I refine the model?

When you press refine, XCE will take the model called refine.pdb and save it to a subfolder called Refine_<cycle number + 1> as in.pdb together with the shell script that will be used for refinement. If something goes wrong and the reason for failure is not clear it is usually a good starting point to look at the logfiles in the respective folder. Keep in mind that if you added for example water or other solvent molecules and did not merge them into refine.pdb, then they will not be included. XCE does not do any automatic merging!

Known Problems

Bash vs C-shell

XCE was mostly tested in the bash shell. It should work in any of the C-shells, but there could be problems when XCE executes scripts in a subshell.