2dx User Manual

Marcel Arheit, Sebastian Scherer, Cristina Paulino,
Xiangyan Zeng and Henning Stahlberg

May 31, 2015

Contents

1 Introduction

The present user manually describes the entire image processing pipeline for 2D crystals in 2dx. For more detailed background information we refer to [Arheit *et al.* 2012c, Arheit *et al.* 2012a, Arheit *et al.* 2012b].

The latest version of the manual is available on http://www.2dx.unibas.ch/documentation/2dx-software/2dx-user-manual-.pdf.

1.1 What is 2dx?

2dx [Gipson *et al.* 2006, Gipson *et al.* 2007] is an initiative that aims at facilitating computer image processing in electron crystallography. Electron crystallography is our method of choice for the determination of the high-resolution structure of membrane proteins in the membrane-embedded state. This is done by recording cryo-electron microscopy images of two-dimensional membrane crystals of the membrane proteins, and reconstructing the 3D structure of the proteins by computer image processing of a larger number of images.

We have created a software package called 2dx that aims at providing the user with user-guidance and help, and features streamlined processing solutions with optional full automation. The website www.2dx.org is the home of the 2dx software, and hosts the software itself, including the open source code, as well as the manual, that the users can annotate in blog form. 2dx utilizes the excellent MRC software that was developed and is maintained at the MRC by R. Henderson and others[1]. While the latest edition of the original MRC software can be requested from Judith Short[2], we provide here on this 2dx.org server a slightly adapted version of the MRC software code in order to interface it with the 2dx system, and we also have added programs for automation and other functions.

2dx is also a repository for user experience and other software solutions. We provide a manual for the MRC software[3], which was contributed by Vinzenz Unger[4] and Anchi Cheng[5]. We here also collect links and information about other software systems.

[1] http://www2.mrc-lmb.cam.ac.uk
[2] JMS@mrc-lmb.cam.ac.uk
[3] http://www.2dx.unibas.ch/documentation/mrc-software
[4] http://pantheon.yale.edu/~mfr28/members.htm
[5] http://www.scripps.edu/~acheng/

2 Installation

In this chapter we explain how `2dx` is installed on different operating systems. In order to ensure our easy-to-use policy we provide precompiled packages for Mac and some Linux distributions. After first listing all external dependencies we explain in details how to install the precompiled package on a Mac in Section 2.3. Subsequently we discuss the binary packages for Debian-like distributions (Section 2.4) and for RedHeat-like Linux distribution (Section 2.5). Finally we provide a step-by-step manual on how to compile `2dx` on your own machine.

Important Note: All the precompiled packages only run on 64 bit machines. If you are still using a 32 bit operating system you have to compile `2dx` yourself as described in Section 2.6.

Login required! Downloading `2dx` requires that you are logged in on our homepage. Creating an account can be done at anytime and for free. We use the login mechanism for tracking all users and to inform you about new `2dx` versions and upcoming events. We will not spam your mail account and will treat your informations with all the required privacy.

2.1 External Dependencies

`2dx` depends on some third-party programs which have to be installed on your system.

- **TCSH:** Some of our scripts require an alternative C-shell called tc-shell. We advise to install this shell by means of your native package manager, e.g. `port`, `apt-get` or `yum`. For further information we refer to `http://www.tcsh.org/Welcome`.
- **CCP4:** `2dx` uses `CPP4` in order to generate merged maps of the protein. The software can be downloaded from `http://www.ccp4.ac.uk/download.php`. Please follow their installation guide. Please install CPP4 into `/usr/local/ccp4`.
- **Chimera:** 3D density maps can be visualized in a powerful way by means if Chimera. You find all the required information and packages on `http://www.cgl.ucsf.edu/chimera/`.
- **Spider:** If you would like to have density maps larger than 2×2 unit cells `2dx` generates them for you with Spider. We refer to `http://www.wadsworth.org/spider_doc/spider/docs/spider.html`. To ensure that `2dx` finds Spider you should install the software into `/usr/local/spider`. In fact `2dx` searches for an executable file called `spider.exe` in the folder mentioned before.

2.1.1 CCP4 on Linux

As we had serious issues in context of installing and using `ccp4` on Linux, we provide a walkthrough which worked on our systems.

1. Download the latest version (tested with `ccp4-6.3.0`)
2. Extract the downloaded archive
3. Move the generated folder to `/usr/local/ccp4-6.3.0`
4. Go to `/usr/local/ccp4-6.3.0` and run the *BINARY.setup* script as root
5. Generate a symbolic link to `/usr/local/ccp4` by typing:
   ```
   sudo ln -s /usr/local/ccp4-6.3.0 /usr/local/ccp4
   ```

2.2 Downloading `2dx`

`2dx` is freely available under the General GNU public license (GPL). Please use the following page for downloading `2dx` `http://www.2dx.unibas.ch/download/2dx-software`. In general we offer two different versions to download. On one hand you can download the latest version of `2dx` which is a snapshot of the source code repository at midnight (i.e. *nightly build*). Although we give our best we can not guaranty that this version, which contains all the latest features, does alway works perfectly. On the other hand we offer previous versions of `2dx`, which should be more stable but do not contain the latest fixes and features.

Which version should you download? In general we advise to use the nightly build version from `http://www.2dx.unibas.ch/download/2dx-software/2dx-nightly-build`. This folder contains for different archives:

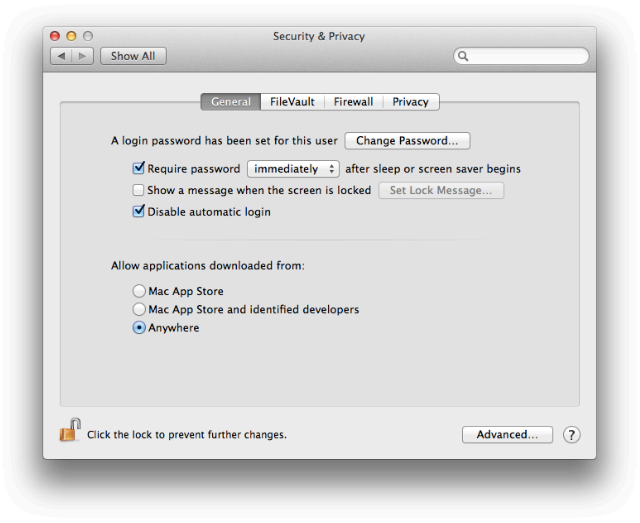
- **2dx_nightly_build_OSX.pkg** is an easy to install package for Mac. We tested this installer on OSX10.6 (Leopard), OSX10.7 (Lion) and OSX10.8 (Mountain Lion). Using this package is the easiest way to install `2dx` on a Mac in general. You find a detailed manual in Section 2.3.
- **2dx_nightly_build_Linux.deb** is a precompiled binary package for debian-like distributions like Ubuntu, Mint or Debian. Further information can be found in Section 2.4.
- **2dx_nightly_build_Linux.rmp** is a binary archive for Red-Heat, Fedora or Scientific Linux. A walkthrough the installation is given in Section 2.5.
- **2dx_nightly_source.tar.gz** should be used in case you want or have to compile `2dx` on your machine yourself. Note that this compilation, which is described in Section 2.6, does not depend on the used operating system.

You also find all the older versions of `2dx` in our download domain. If you feel more comfortable with installing a older but potentially more stable version you also can download the outdated archives. The installation itself is the same as for the nightly build.

2.3 Installation on Mac

The installation on a Mac is straightforward as you just have to run the installer. In order to install you need administrator privileges on your machine. If you don't have them please contact your system administrator. The following steps are required to install the package on your Mac.

1. (Only required on OSX10.8 Mountain Lion) Apple's new security policy per default allows you to install software just from the App Store. To be able to install `2dx` you have to allow the installation from other sources as just the App Store. This can be done under "System Preferences > Security & Privacy > General > Adcanced..." as shown in Figure 1.



A login password has been set for this user

[✓] Require password immediately after sleep or screen saver begins

[ ] Show a message when the screen is locked

[✓] Disable automatic login

Allow applications downloaded from:

( ) Mac App Store

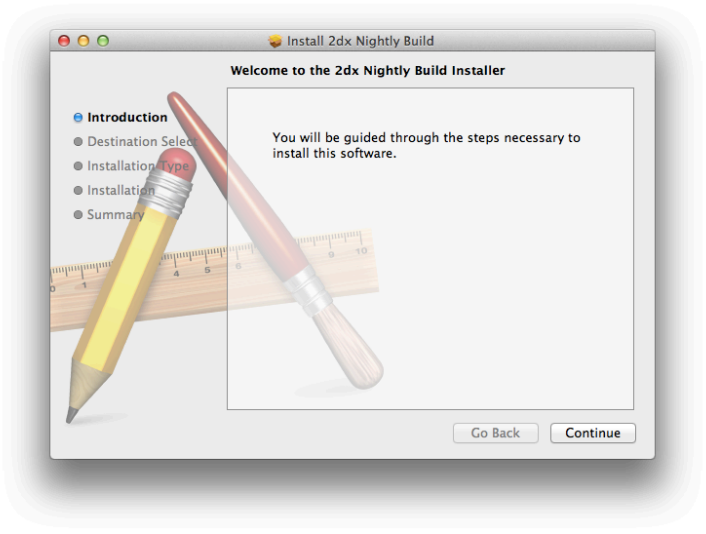
( ) Mac App Store and identified developers

[✓ Anywhere]

Click the lock to prevent further changes.

Figure 1: Mountain Lion Security Settings

2. After downloading double-click on the downloaded package. This will launch the installation dialog shown in Figure 2.



Welcome to the 2dx Nightly Build Installer

You will be guided through the steps necessary to install this software.

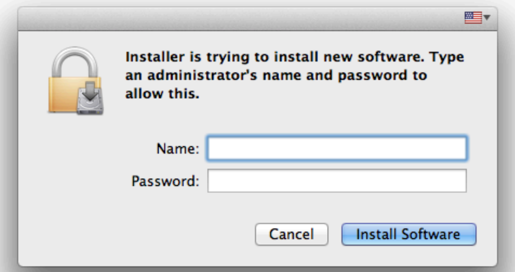
Figure 2: `2dx` installer on Mac

3. The first window tells you that this installer will guide you trough the installation process. You reach the next page by clicking on "continue"..

4. Now you are ask where you want to install the software. The simplest way is to click "continue" and the software will be installed in the default location (`/opt/2dx`) and can be run by all the users of the computer.

5. The following screen shows you a summary of the planed installation. A click on the "install" button will launch the installation

process. Usually you are ask to enter the administrator password (Figure 3) before the software is installed.

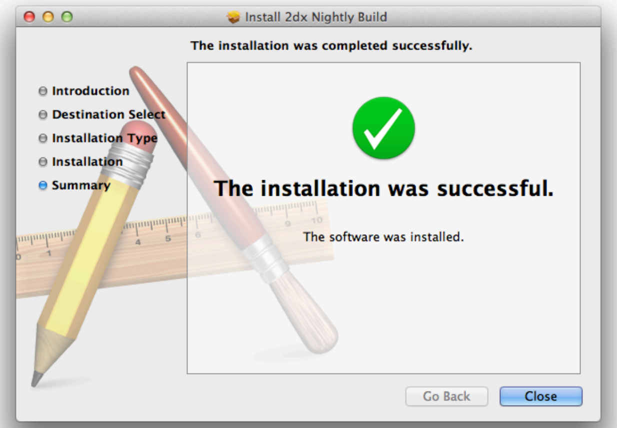

Installer is trying to install new software. Type an administrator's name and password to allow this.

Name:

Password:

Figure 3: Installer asking for administrator privileges on a Mac

6. Once you entered the correct administrator password the installation is performed and you see a growing status bar.
7. If everything worked well the installer tells you that the installation was successful (Figure 4). In the mean time the Application folder is opened and you should find a `2dx` icon.


Install 2dx Nightly Build

The installation was completed successfully.

The installation was successful.

The software was installed.

Figure 4: Successful installation on Mac

8. You can launch `2dx` by double-clicking on this icon.

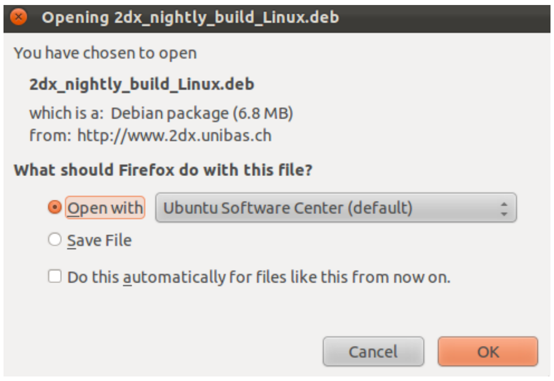
2.4 Installation on Linux (Ubuntu)

Due to some technical problems[6] we can just deliver a binary package which works on Ubuntu-12.04. In case you can not upgrade your distribution you can either compile `2dx` yourself (Section 2.6) or upgrade libc6 to version 2.14.xx on your older Ubuntu. Note that the described update is not officially supported and we don't give any warranty.

Note that `tcsh` is not resolved automatically and that you are responsible for installing this additional shell yourself (Section 2.1)

Below you find a step-by-step walkthrough the installation on a 64 bit Ubuntu-12.04.

1. If you download the package **2dx_nighlty_build_Linux.deb** from our homepage Firefox asks you what you what you want to do with the downloaded file (Figure 5). The easiest way is to open the package directly with the Ubuntu Software Center by means of clicking "ok" in the shown dialog. Alternatively you can also download the package and then double-click on it.



You have chosen to open

2dx_nightly_build_Linux.deb

which is a: Debian package (6.8 MB)
from: http://www.2dx.unibas.ch

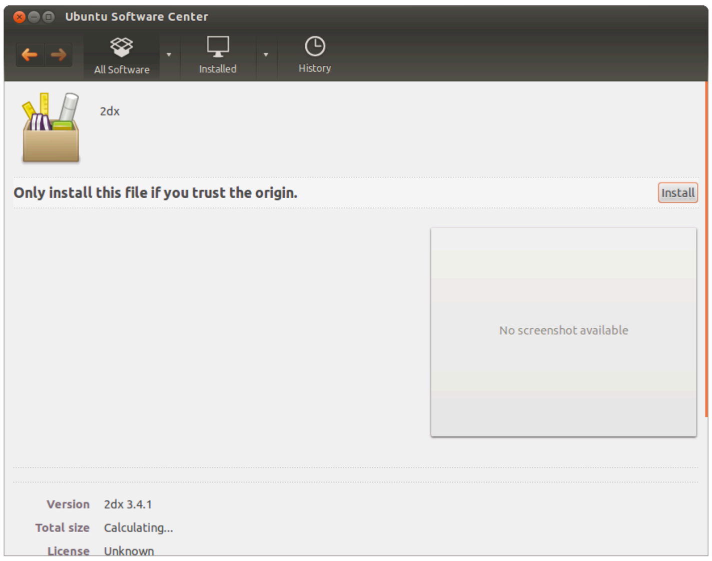
What should Firefox do with this file?

● Open with: Ubuntu Software Center (default)
○ Save File
☐ Do this automatically for files like this from now on.

Figure 5: Download dialog on Ubuntu

2. After opening the package with the Ubuntu Software Center you get the window shown in Figure 6 where you can see a summary of our package.

[6] upgrade of libc6 to version 2.14 which is not compatible with all earlier versions

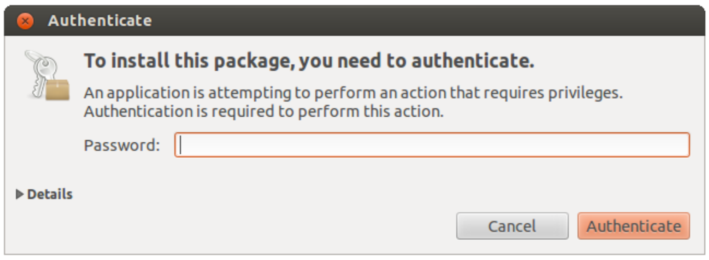


Only install this file if you trust the origin.

Version 2dx 3.4.1
Total size Calculating...
License Unknown

Figure 6: `2dx` installation with Ubuntu's Software Center

3. Clicking on "install" starts the installation process. For security reasons you are asked for the `sodu` password as normal users are not allowed to install software. Please enter the required password in the window shown in Figure 7.



To install this package, you need to authenticate.

An application is attempting to perform an action that requires privileges. Authentication is required to perform this action.

Password:

Figure 7: Ubuntu's Software Center asking for administrator privileges

4. After successfully installing `2dx` you will find a `2dx`-icon under the "Education" menu entry (Figure 8). You now can start `2dx` by simply clicking on this icon.

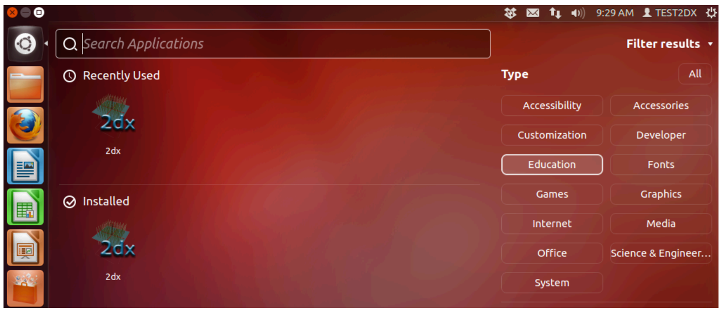


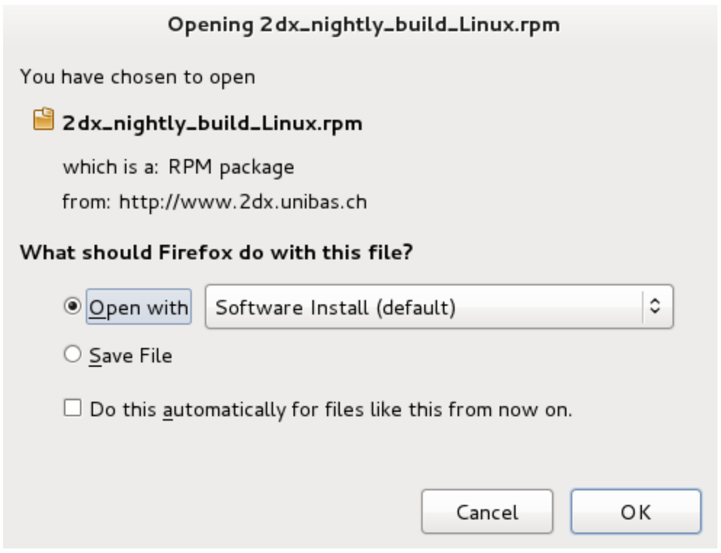
Figure 8: Launching `2dx` on Ubuntu

Additionally we also install `2dx_image` which is used for single image processing. In general you should not open `2dx_image` directly yourself as `2dx_merge` will open it for you with the correct parameters.

2.5 Installation on Linux (Fedora)

The following manual guides you through the installation process on RedHeat-like distributions. We tested the installation on Fedora 16 and Fedora 17 but we don't see any reason why it should not work on other similar platforms.

1. If you download the package **2dx__nighlty__build__Linux.rpm** from our homepage Firefox asks you what you want do do with the downloaded file (Figure 9). The easiest way to launch the installation process of the package directly is clicking on the "ok" button. Alternatively you can also download the package and then double-click on it.



Opening 2dx_nightly_build_Linux.rpm

You have chosen to open

2dx_nightly_build_Linux.rpm

which is a: RPM package

from: http://www.2dx.unibas.ch

What should Firefox do with this file?

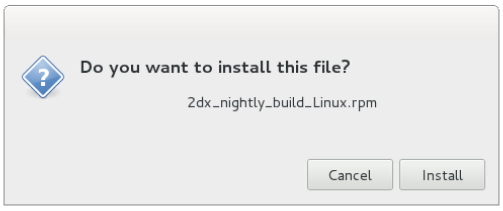
(●) Open with: Software Install (default)

( ) Save File

☐ Do this automatically for files like this from now on.

Figure 9: Download dialog on Fedora

2. Next you are asked whether you really want to install the packages (Figure 10).



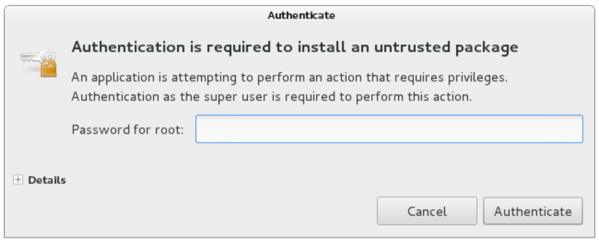
Do you want to install this file?

2dx_nightly_build_Linux.rpm

Figure 10: Installation dialog on Fedora

Clicking on "install" brings you to the next step.

3. Before the installation process is launched you have to enter the `sudo` password as shown in Figure 11. Please note that if you use the native package manager all dependencies will be installed automatically. If you use another package installer you probably have to install some dependencies manually before the installation of `2dx` will be launched.



Authenticate

Authentication is required to install an untrusted package

An application is attempting to perform an action that requires privileges.
Authentication as the super user is required to perform this action.

Password for root:

Figure 11: Fedora installer asking for the root password

4. After the successful installation you will find a `2dx`-icon under the "education" menubar as shown in Figure 12.

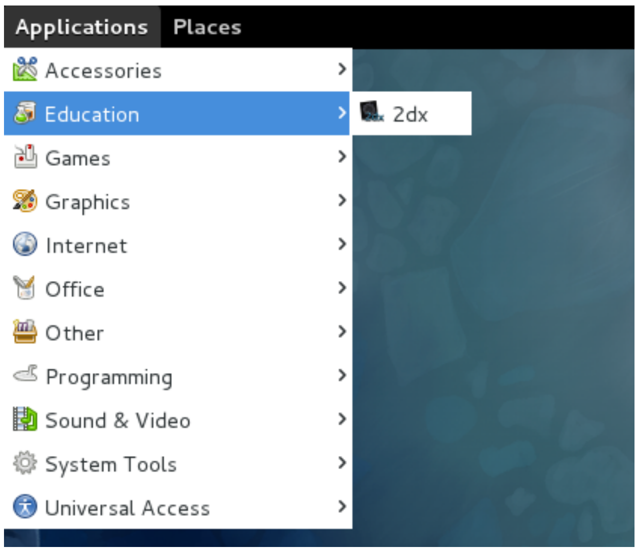


Figure 12: Launching `2dx` on Fedora

Additionally we also install `2dx_image` which is used for single image processing. In general you should not open `2dx_image` directly yourself as `2dx_merge` will open it for you with the correct parameters.

2.6 Compiling from source

In this section we explain how to compile `2dx` on your own machine, which always slightly depends on your specific operating system and the version of the installed software. Thus we only insure that our manual works for Ubuntu 12.04, Fedora 17 and OSX.10.7. Probably some dependencies slightly change for other platforms but the basic concept stays the same. An additional challenge is to know the correct linux specific package name. We state all dependencies on Linux in term of the Ubuntu/Debian naming convention and we give some additional hints for RedHeat-based systems.

For OSX we suggest that you install the dependencies via macports (`http://www.macports.org`).

The only supported compilers are `gcc` and `icc`. On mac please install `gcc4X` via macports and activate the installed compiler with `sudo port select -set gcc mp-gcc4X`. Note that `X` stands for the particular version you installed. The compilation process was successfully tested with `gcc48` and `gcc49`.

2.6.1 Dependencies

The compilation of `2dx` has some additional dependencies compared with the precompiled binary packages. Below you find the list of all dependencies:

- **cmake** As we use `cmake` as build system you should have installed it on your machine. We advise to install this and most of the other dependencies by means of your native package manager. Alternatively you can get `cmake` under the following link: `http://www.cmake.org`.
- **gfortran (incl. bindings)** Some programs of the `2dx`-kernel are written in fortran. Thus compiling `2dx` requires a fortran compiler. Note that we only support the GNU fortran compiler `gfortran`. Dependent on your package source you might have to install some platform dependent binding libraries between fortran and c++. But on Fedora you have to install these additional packages: `gcc-gfortran` and `libgfortran`.
- **g++ (incl. bindings)** The `2dx`-GUI is written in `c++` and thus requires a corresponding compiler. Again on some RedHeat-like Linux distributions you need to install some bindings. On Mac and Ubuntu we never had these issues, but on Fedora `2dx` relies on the`gcc-c++` package.
- **libfftw3** `2dx` requires fftw version 3.0.0 or later. You can install this famous library via the package manager or directly from `http://www.fftw.org`. The package is called `libfftw3-dev` on Ubuntu, but on Fedora you have to install `fftw-devel` and `fftw-libs`.
- **Qt4** The installation of Qt4 on Mac and Linux are fundamentally different:
  - **Qt4 on Mac** Please get the latest Qt4 version from `http://qt-project.org/downloads/` and install the entire QtSDK on your machine. After the installation you have to ensure that

  `qmake` is in your $PATH.

  - **Qt4 on Linux** On Ubuntu you can just install `libqt4-dev` with all the automatically resolved dependencies by means of the native package manager. On Fedora the required package is called `qt-devel` with `qtwebkit-devel` as additional required dependency.

- **libboost** Ubuntu: `libboost-all-dev`, Fedora: `boost-devel`.

Note that the name of the required libraries depends on the used operating system. In case you have a missing library the compilation error should be somewhat. Thus resolving the missing dependency should be easy. If you run into troubles don't hesitate to contact us for support.

2.6.2 Compilation Process

The compilation itself is straightforward as we provide a simple script which coordinates all the required steps:

1. Download the latest version from githug with. Requires that you have git on your system
   `git clone https://github.com/C-CINA/2dx.git`

2. Change into the generated folder:
   `cd 2dx`

3. `2dx` is compiled by running the `build_all` script, which take between zero and two arguments. The list below shows you all the possibilities:

   - **without arguments:** `2dx` will be compiled and installed into `/2dx`.

   - **with one argument:** You can specify your install and build location by passing an absolute path to the build script.

   - **with two arguments:** The build script with two arguments performs an out-of-place installation of `2dx`. The first passed absolute path corresponds to the build directory while the second defines the install directory.

   `2dx` is compiled in two stages. First we check the presence of all required dependencies and in the second stage we compile the entire source tree.

4. Once you successfully compiled `2dx` you can launch `2dx_merge` by changing into the `bin/` directory of the install directory and typing `./2dx_merge`.

5. In order to be able to start `2dx` in the terminal by typing `2dx` you should generate a symbolic link to the executable of `2dx_merge` by the following command:
   `ln -s 'YOUR_INSTALL_PATH'/bin/2dx_merge /usr/bin/2dx`.

   Alternatively we suggest to "source" the binary folder by adding a line similar to `export PATH=$PATH:/'YOUR_INSTALL_PATH'/bin` to your .profile or .bash_rc file.

3 Project Initialization and Folder Structure

In this chapter we will guide you through your first steps with 2dx. You will create your first project and you will learn how to import micrographs.

3.1 Launching 2dx and Creating a New Project

Once you installed 2dx as described in Chapter 2 you have a 2dx-icon in your Applications folder on a Mac or under the Education menu entry under Linux. In case you compiled 2dx yourself please launch 2dx_merge in the /bin directory of your installation location. Clicking on this icon will launch 2dx_merge which is the entry point for all 2dx-projects. 2dx immediately asks you to select your project directory as shown in Figure 13.

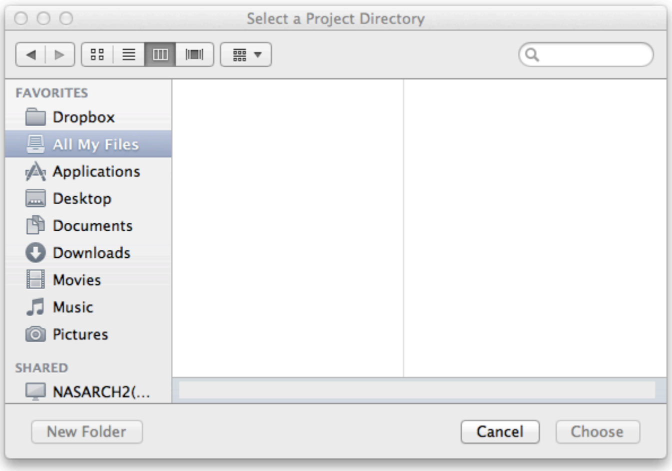
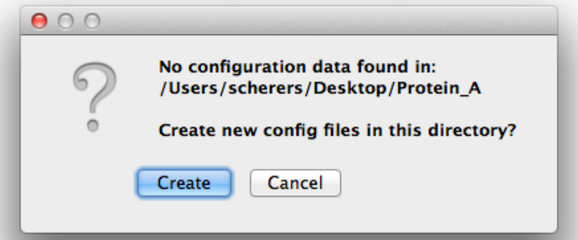


Figure 13: Launching 2dx

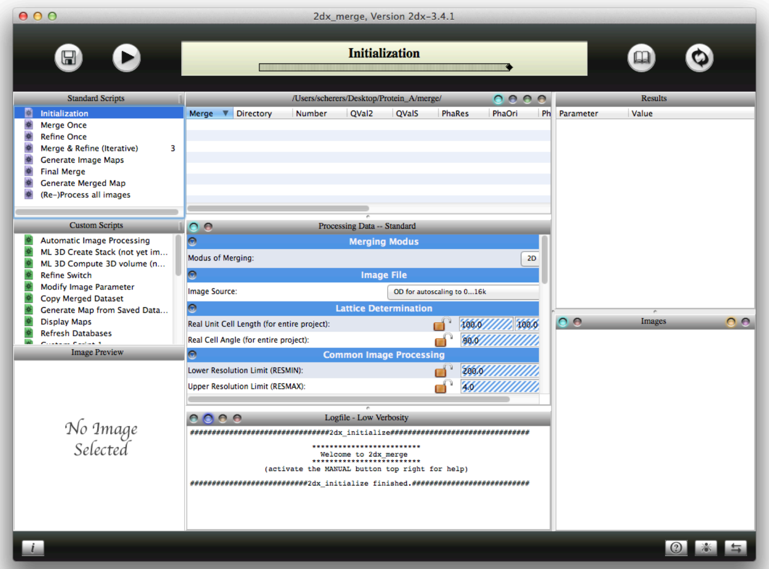
As you don't have a project right now, you should create a new folder called "Protein_A" under "Desktop". Select the newly generated folder and click on "Choose". 2dx now will look for configuration data in the selected folder. If the required data is not found the application will ask you whether a new configuration file should be created in this location as shown in Figure 14. For detailed information on the project structure we refer to Section 4.2.3.


No configuration data found in:
/Users/scherers/Desktop/Protein_A

Create new config files in this directory?

Figure 14: Creating project directory

After the project initialization `2dx_merge` will open and you will see the following window shown in Figure 15. We will discuss all parts of the `2dx` graphical user interface in Section 4.1 and Section 4.3.



Modus of Merging: 2D

Image Source: OD for autoscaling to 0...16k

Real Unit Cell Length (for entire project): 100.0 100.0

Real Cell Angle (for entire project): 90.0

Lower Resolution Limit (RESMIN): 200.0

Upper Resolution Limit (RESMAX): 4.0

```
################################2dx_initialize####################################

                         *************************
                           Welcome to 2dx_merge
                         *************************
                 (activate the MANUAL button top right for help)

#############################2dx_initialize finished.############################
```

Figure 15: `2dx` merge GUI

So far you have initialized a new `2dx`-project on your computer and now we are ready to import the first micrographs.

3.2 Importing Images to `2dx`

The whole tutorial you are holding in your hands is based on one exemplary dataset, which you can download from our homepage (`http://www.2dx.unibas.ch/download/2dx-software/test-data`). Thus you are encouraged to download the archive to your hard disk. After the download you have to uncompress the archive in order to be able to import the micrographs.

To import the images into `2dx` you have to click on "File > Import Images..." in the toolbar at the top of your desktop (Figure 16).

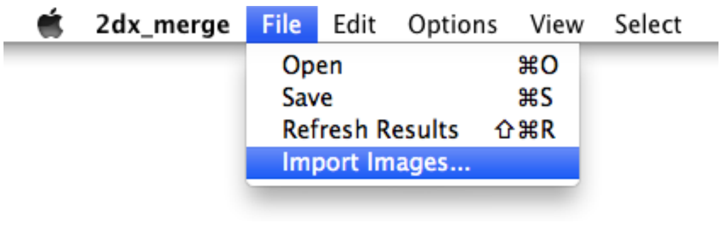
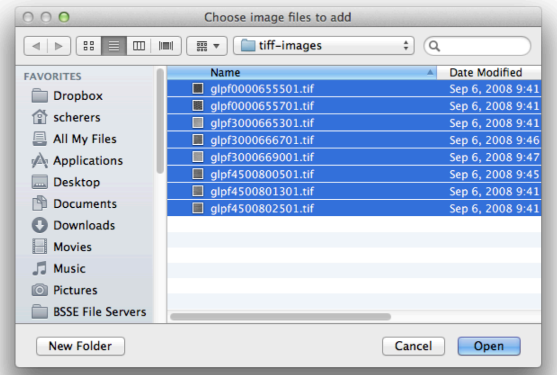


Figure 16: Importing files

"Import Images..." will open a file browser which is used to navigate to the folder containing the micrographs to import. Once you selected all the micrographs as shown in Figure 17 click on "Open". `2dx` allows you to import *.tiff* and *.mrc* images. If your data is stored in another format you have to convert your images by means of an external program.

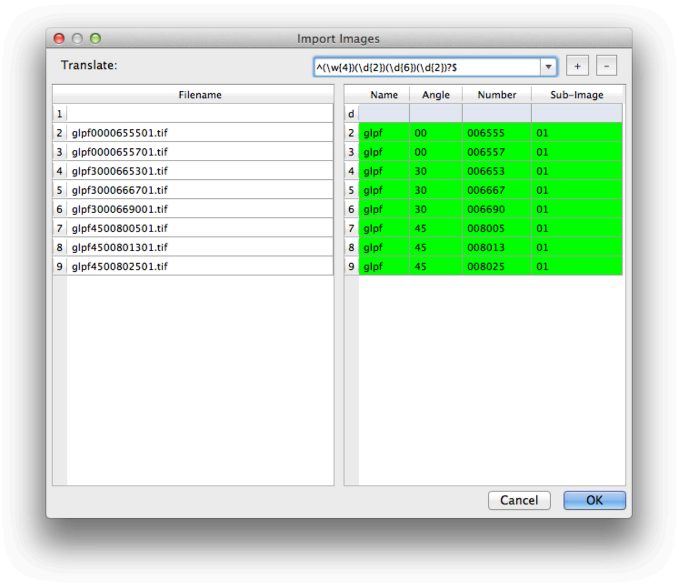


Name	Date Modified
glpf0000655501.tif	Sep 6, 2008 9:41
glpf0000655701.tif	Sep 6, 2008 9:41
glpf3000665301.tif	Sep 6, 2008 9:41
glpf3000666701.tif	Sep 6, 2008 9:46
glpf3000669001.tif	Sep 6, 2008 9:47
glpf4500800501.tif	Sep 6, 2008 9:45
glpf4500801301.tif	Sep 6, 2008 9:41
glpf4500802501.tif	Sep 6, 2008 9:41

Figure 17: Select files to import

Usually each electron microscopy lab has a naming convention for their micrographs, which contains additional information about each micrograph, e.g. tilt angle. For instance in our lab the first four letters of an image name are used to store the name of the protein. The following two digits are the tilt angle followed by the six digits image number. The two last digits represent the sub-image number, which usually is set to `01`.

The dialog shown in Figure 18 allows you to parse the additional image information from the filename. In the left panel of the new dialog all images which will be imported are listed. When finally importing the selected images `2dx` applies the *regular expression* selected from the pull-down menu in top of the dialog in order to extract the additional information. The right panel of the dialog shown you a preview of the actually parsed information. For the provided test dataset the default regular expression works fine and clicking on "OK" imports the images. You can change the applied expression to your needs and click on "+" in order to store the expression for a later reuse. Details on regular expressions are discussed in Section 4.2.4.



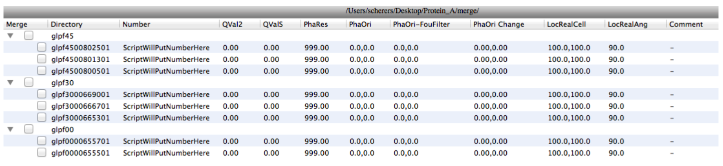
Translate: ^(\w{4})(\d{2})(\d{6})(\d{2})?$

	Filename
1	
2	glpf0000655501.tif
3	glpf0000655701.tif
4	glpf3000665301.tif
5	glpf3000666701.tif
6	glpf3000669001.tif
7	glpf4500800501.tif
8	glpf4500801301.tif
9	glpf4500802501.tif

	Name	Angle	Number	Sub-Image
d				
2	glpf	00	006555	01
3	glpf	00	006557	01
4	glpf	30	006653	01
5	glpf	30	006667	01
6	glpf	30	006690	01
7	glpf	45	008005	01
8	glpf	45	008013	01
9	glpf	45	008025	01

Figure 18: File import dialog

Once you imported the images the central panel of the 2dx user interface will look like shown in Figure 19.



/Users/scherers/Desktop/Protein_A/merge/

Merge	Directory	Number	QVal2	QValS	PhaRes	PhaOri	PhaOri-FouFilter	PhaOri Change	LocRealCell	LocRealAng	Comment
▼ ☐	glpf45										
☐	glpf4500802501	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
☐	glpf4500801301	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
☐	glpf4500800501	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
▼ ☐	glpf30										
☐	glpf3000669001	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
☐	glpf3000666701	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
☐	glpf3000665301	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
▼ ☐	glpf00										
☐	glpf0000655701	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-
☐	glpf0000655501	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0	0.0,0.0	0.00,0.00	100.0,100.0	90.0	-

Figure 19: Project panel with imported micrographs

Note that the images are grouped by their name and tilt angle into different tilt series. Before we start processing the individual image we discuss the graphical user interfaces of 2dx and some useful technical details in the subsequent Chapter 4.

4 Graphical User Interfaces of 2dx

In this chapter we give an overview of the programs in the `2dx`-package. We distinguish between `2dx_merge` (Section 4.1) for merging multiple density maps generated from individual images by `2dx_image` (Section 4.3). Additionally we give some details about further technical subtleties in Section 4.2.

4.1 `2dx_merge` Overview of the Graphical User Interface

Figure 20 shows the `2dx_merge` graphical user interface, which will be discussed in the following section. The present interface is the starting point for each `2dx`-project as all the micrographs are managed and finally merged here.

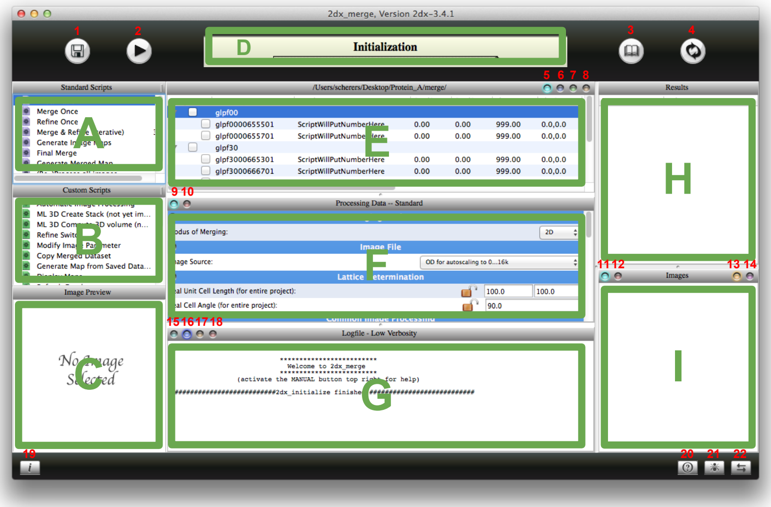


Figure 20: `2dx_merge` graphical user interface

In the following we distinguish between panel (green boxes) and buttons (red numbers). `2dx_merge` consists of the following panel:

A - Standard Scripts :	All standard processing scripts are grouped in this panel. You can select one or multiple scripts by clicking on them. A double-click will open the script in a text editor and allows you to change the script to your needs. Note that it may be that you have to configure which text editor is opened before you open the first script. The corresponding manual can be found in Section 4.2.2.
B - Custom Scripts :	Similar to the *Standard Scripts*, but these scripts usually are not needed for simple processing but may be useful for non standard tasks or managing your project.
C - Image Preview :	Once you selected an image in panel *I - Image Preview Panel* you can either see the header of the selected image or a preview.
D - Status Bar :	The status bar shows the progress of the running script. Note that only approximate value are shown.
E - Project Panel :	All micrographs of your current project are collected in this panel.

The micrographs are grouped by their name and tilt angle into folders. You can select individual micrographs by clicking the checkbox in front of the image name. You can also select an entire tilt series by selecting the corresponding top-level directory. In Section 4.2.1 we explain how you can customize this panel.

F - Parameter Panel : Once you selected a script from panel A or B all the corresponding parameters show up in this panel. On the left side the parameter name is shown and its value can be set on the right side. Clicking on a parameter name with the right mouse button opens a help window where you get a short summary of the parameter. Some of the parameters can be locked (by clicking on the lock) in order to prevent them from being overwritten accidentally. Note that you can change the verbosity level of the parameter panel with the buttons 9 to 10.

G - Log Panel : All our scripts generate some terminal output which is shown in this panel. Note that you can change the verbosity level of the log panel with the buttons 15 to 19.

H - Results Panel : During the calculations some scripts generate resulting values, which might be important to know for experienced users. All these values are listed in the present panel.

I - Image Panel : Some of the scripts generate some resulting images or files which are listed in this panel. For example you will find the finally merged map in this panel.

Beside the panels mentioned above `2dx_merge` has a bunch of buttons which allow the user to launch calculations or change the look if the graphical user interface:

1 - Save Configuration : Saves the current parameter set to the configuration file

2 - Launch Script : Launches the script or scripts selected either in *Standard Scripts* or *Custom Scripts*.

3 - Show/Hide Manual : This button shows you the quick reference of the selected script. The manual can be hidden again by clicking on the button once more.

4 - Refresh Results : Force `2dx` to reload all generated results.

5 - Default View : Change to view back to the default setting.

6 - Maximize Project Panel : Maximizes the *Project Panel* while removing all the panels E-H from the current view.

7 - Maximize Parameter Panel : Similar to *Maximize Project Panel* but this button maximizes the parameter panel at the other's panel presence.

8 - Maximize Log Panel : Maximizes the *Log Panel*.

9 - Show Simple Parameters : `2dx` support two parameter setting modes. With this button you can change to the simple parameter mode.

10 - Show Advanced Parameters : The expert parameter mode can be activated by means of this button.

11 - Show All Images : `2dx` distinguishes between all and important images generated by

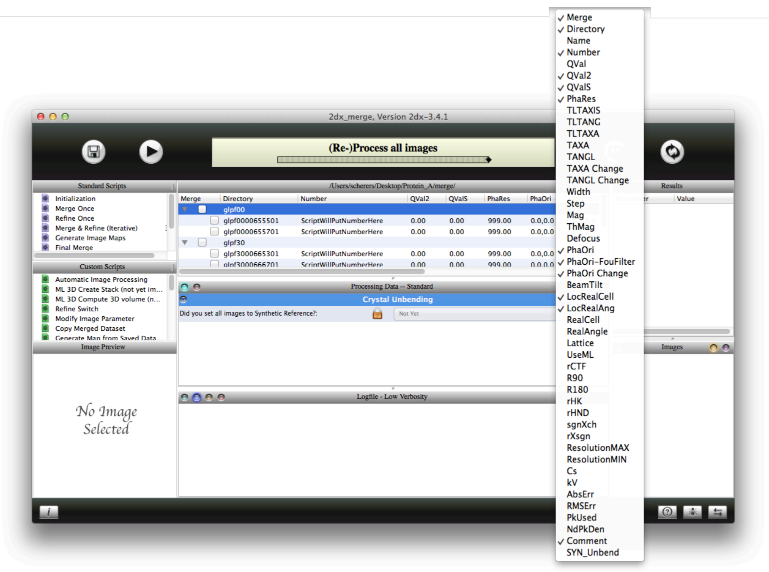
	the launched scripts. The mode activated by this button shows you all generated images.
12 - Show Important Images :	The present button only shows you the important generated images
13 - Show Nicknames :	The names of the generated images can be shown in two different modes. Either you can use nicknames which describe in a compact fashion what kind of information the image contains or you can use the original filename. The nickname mode is activated by pressing this button
14 - Show Full Filenames :	In order to see the original filename of the generated images click on this button.
15 - Silent Log Mode :	`2dx`'s log output can be generated in different verbosity levels. You can select the level of your needs with the button 15 to 18. In this *Log Panel* mode only the script terminations are reported.
16 - Low Log Mode :	Shows all the important information generated by the scripts.
17 - Moderate Log Mode :	Gives you some additional output compared to the low verbosity.
18 - Highest Log Mode :	Shows you all output generated by the scripts. Note this mode may slow down the execution of the script.
19 - Show/Hide Preview :	*Image Preview* can either show the header of the selected file or a preview of the context. This button allows you to change between the two modes.
20 - View Online Help :	Redirects you to our online help
21 - Launch Bug Tracker :	Redirect you to the `2dx` bug tracker. You are kindly requested to report all observed issues.
22 - Stop Sync with Config :	Change to the "Dry Run" mode to test the execution of a script without potentially messing up your configuration file. Don't forget to change the mode back in case you want to overwrite the current configuration.

4.2 Additional Technical Aspects for `2dx_merge`

In this section we discuss additional technical aspects such as customizing the project panel, the structure of a `2dx`-project and regular expressions. The deep understanding of these additional tricks in general is not absolutely required for successfully processing your images.

4.2.1 Customizing the Project Panel

During processing the individual images with `2dx_image` a lot of different parameters are calculated for each images. These parameters can be displayed in `2dx_merge`'s *Project Panel*. The default parameter selection can be adapted to your needs by simply right-clicking on the parameter header. You can choice your parameter selection in the appearing selection list as shown in Figure 21.



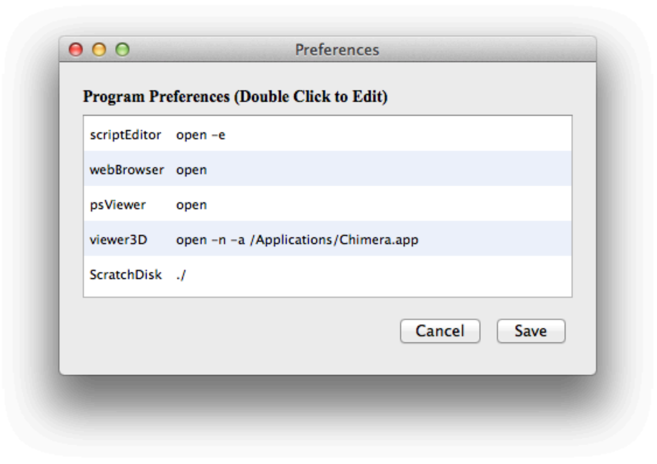
Merge	Directory	Number	QVal2	QValS	PhaRes	PhaOri
▼ ☐	glpf00					
☐	glpf0000655501	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0
☐	glpf0000655701	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0
▼ ☐	glpf30					
☐	glpf3000665301	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0
☐	glpf3000666701	ScriptWillPutNumberHere	0.00	0.00	999.00	0.0,0.0

Did you set all images to Synthetic Reference?: Not Yet

Figure 21: Change the shown parameters of the project panel

4.2.2 Customizing Third-Party Program Calls

Some of the file generated by `2dx` can not be open natively in `2dx`, e.g. *.pdf* or 3D volumes. Double-clicking on these files the *Images Panel* opens a third-party program installed on your machine. The corresponding setting can be adapted under "2dx_merge > Preferences" as shown in Figure 22. In order to change the settings please enter the corresponding command which launches the desired third-party application in the terminal. Dependent on your operating system we use different default values, but dependent on the installed software you probably have to configure the setting.



Program Preferences (Double Click to Edit)

scriptEditor	open -e
webBrowser	open
psViewer	open
viewer3D	open -n -a /Applications/Chimera.app
ScratchDisk	./

Figure 22: Program Preferences

4.2.3 `2dx` Project Structure

A `2dx`-project directory contains the following folder and files:

- **2dx__master.cfg** is the top-level configuration file which serves as default configuration files for all calculations performed on the entire project. Note that you should not edit these files manually as they are usually set trough the graphical user interface.
- **merge-folder** contains all the files generated while merge multiple images by means of `2dx_merge`.
- **Folders per tilt angle:** `2dx` generates a folder for each tilt angle. Each image and all the corresponding files generated during the single image processing are stored in an individual folder in the corresponding tilt angle folder.

4.2.4 Regular Expressions

One of the imported micrographs was called *glpf0000655301.tif*. The additional information was parsed by the following regular expression (Section 3.2): `^(\w{4})(\d{2})(\d{6})(\d{2})$`. The 4 after the "w" defines the number of letters used for the protein name. The next block tells that two digits are used to store the tilt angle. Analogously the image number consists of 6 digits respectively the sub-image number of two digits. If you want to use another convention you have to change the number of digits of the additional information fields. Note that the total amount of used letters should equal the number of letters used for the entire filename. Once you defined the regular expression of your needs you can store the expression by clicking on the "+"-button.

`2dx` comes with a set of different predefined regular expressions from which you can choice or which you also can edit. For instance the following expression allows underscores between the different information fields: `^(\w{4}).*_(\d{2}).*_(\d{6}).*_(\d{2})$`.

4.2.5 The `~/.2dx` folder

`2dx` generates a hidden folder `~/.2dx` in your home directory. In this folder we store some global configurations for `2dx`, such as your personal regular expressions or third party application configurations. Please to not change the content of the folder manually. In case you want to reset all your settings you can simple remove this folder.

4.3 `2dx_image` Overview of the Graphical User Interface

Single micrographs are processed by means of `2dx_image`. Double-clicking a filename in the `2dx_merge`'s *Project Panel* launches `2dx_image` for the corresponding micrograph. Note that before you can merge the images you have to process them in `2dx_image` through the graphical user interface shown in Figure 23 and explained in the following. A bunch of panels and buttons have the same functionality as in `2dx_merge` therefore these points are not discussed in details here and you are asked to lookup the description in Section 4.1.

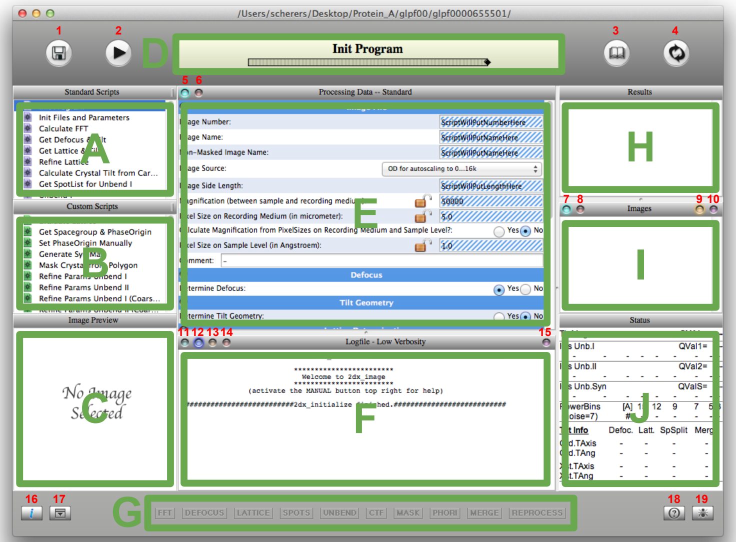


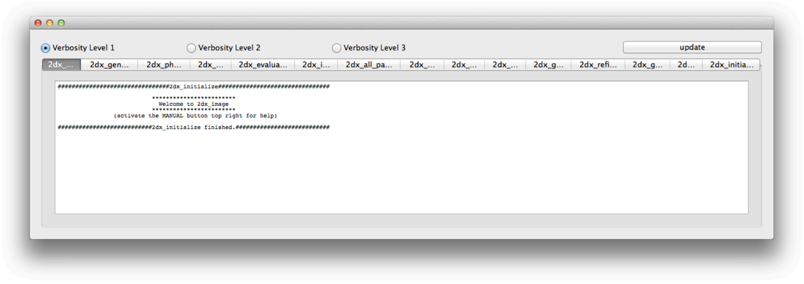
Figure 23: `2dx_image` graphical user interface

A - Standard Scripts :	Scripts for standard single image processing. If you select and launch all the scripts you get the first approximate two-dimensional map from your crystal.
B - Custom Scripts :	More advanced single image processing scripts
C - Image Preview :	Preview of the resulting images
D - Process Bar :	Tells you which script is running and show the actual progress.
E - Parameter Panel :	Panel meant for changing the parameters of the selected script
F - Log Panel :	Log output is shown here.
G - Stamps Panel :	Once a processing step is done the corresponding box gets darker. After successfully processing an image all boxes should be dark. This panel helps you to not forget one of the processing steps.
H - Results Panel :	Results generated by the scripts are listed in this panel.
I - Images Panel :	Intermediate images are collected here.
J - Status Panel :	`2dx` calculates automatically a bunch of values which indicate the quality of the actual processing result. In order to have these values always at hand they are collected in this panel.

1 - Save Config :	Save the config file to disk.
2 - Launch Script :	Launch the selected script(s).
3 - Show/Hide Manual :	Open or closes the manual view of the selected script
4 - Refresh Results :	Reload the *Result Panel*.
5 - Show Simple Parameters :	Shows the simple parameter interface of the selected script.
6 - Show Advanced Parameters :	Shows the advanced parameter interface of the selected script.
7 - Show All Images :	Shows all generated images in the *Images Panel*.
8 - Show Important Images :	Shows only the important generated images in the *Images Panel*.
9 - Show Nicknames :	Shows the nicknames of the generated images in the *Images Panel*.
10 - Show Full Filenames :	Shows the full filename of the generated images in the *Images Panel*.
11 - Silent Log Mode :	Switches the *Log Panel* to silent mode.
12 - Low Log Mode :	Switches the *Log Panel* to low verbosity mode.
13 - Moderate Log Mode :	Switches the *Log Panel* to moderate verbosity mode.
14 - Highest Log Mode :	Switches the *Log Panel* to high verbosity mode.
15 - Show Processing History :	In `2dx_image` you can switch between reading the log and the history of you calculations. The log just shows you the current output of the scripts, but the history browser shows you some parameters and results from previous runs. This feature can be used to optimize certain processing parameters in order to lookup previous quality measures gained by different parameter settings.
16 - Switch Header-Preview :	Changes between header and preview of the generated image selected from *Images Panel*.
17 - Show/Hide Preview :	Shows or Hide the *Image Preview*.
18 - View Online Help :	Opens online help.
19 - Launch Bug Tracker :	Opens the online bug tracker.

4.4 `2dx_logbrowser`

`2dx` generates a bunch of log output during the calculations. Additionally the output is generated in different verbosity levels as explained in Section 4.1. The `2dx_logbrowser` allows you to have a look at the log output in a more human readable way as shown in Figure 24.



```
##################################2dx_initialize##################################

                             ************************
                              Welcome to 2dx_image
                             ************************
                 (activate the MANUAL button top right for help)

#############################2dx_initialize finished.############################
```

Figure 24: `2dx_logbrowser` graphical user interface

You can open the `2dx_logbrowser` by double-clicking on the header of the Log Panel in `2dx`_merge or `2dx_image`.

The graphical user interface of the `2dx_logbrowser` should be self-explanatory in general. You can change the verbosity level of the shown log output be clicking on the corresponding verbosity check box. Above the panel in which the log is finally presented you can select from which program you want to see the log. The "update" button reloads the shown log file in order to display also the latest changes.

5 Image Processing of 2D Crystal Images

Electron crystallography of membrane proteins uses cryo-transmission electron microscopy to image frozen-hydrated/sugar embedded 2D crystals. The processing of recorded images exploits the periodic arrangement of the structures in the images to extract the amplitudes and phases of diffraction spots in Fourier space. However, image imperfections require a crystal unbending procedure to be applied to the image before evaluation in Fourier space. Here we describe the image processing of 2D crystal the `2dx` software system.

5.1 Theoretical Background

The processing of 2D crystal images involves several steps. These steps are the extraction of the lattice, the identification and correction of the crystalline deformation as well as image distortions introduced by the imaging system. This workflow is listed below and simplified in the block diagram in Figure 25.

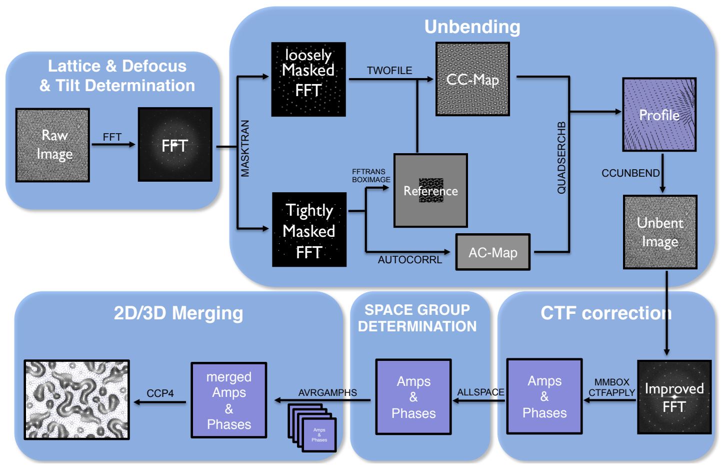


Figure 25: Single Image Processing Overview

1. Defining basic processing parameters
2. Calculating the Fourier transform of the image
3. Measuring the defocus in different locations in the image
4. Calculating potential specimen tilt from the defocus gradient
5. Determining the 2D crystal lattice
6. Calculating potential specimen tilt from distortions of the 2D crystal lattice
7. Determining a spotlist of significant Fourier reflections
8. Determine lattice distortion vectors and perform a first image unbending (Unbend I)
9. Iteratively refine the lattice distortion vectors, do refined unbending (Unbend II)
10. Extract amplitude and phase values for each Fourier reflection from

the unbent image

11. Correct the list of amplitudes and phases for the microscopes CTF

12. Determine space group

13. Calculate a final projection map from this image

These steps will now be discussed in detail, using the 2dx software package as an example.

5.2 Getting started

Starting in `2dx_merge` double-click the image that you want to process listed in the Project panel of the graphical user interface. This will open `2dx_image` for the selected image. The graphical user interface of `2dx_image` consists of different panels displaying data, processing parameters and image-processing routines, as shown in Figure 26.

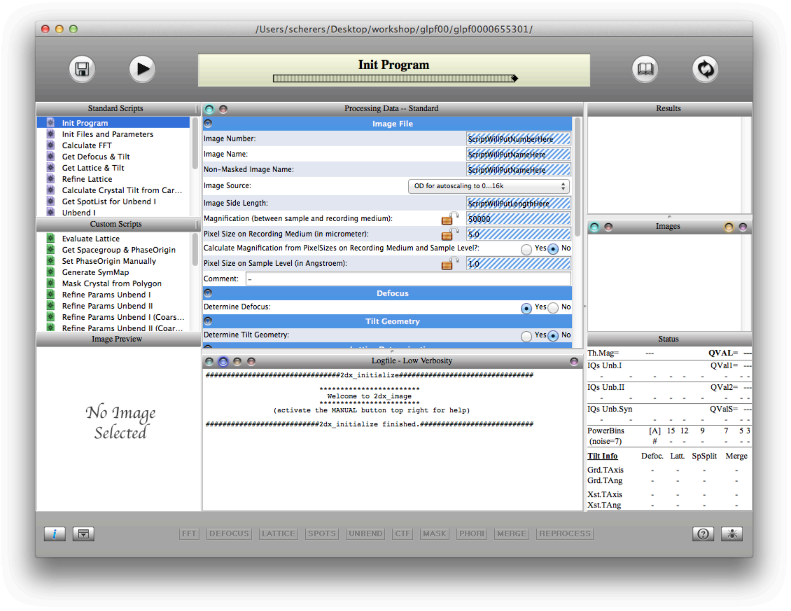


Image File

Image Number:	ScriptWillPutNumberHere
Image Name:	ScriptWillPutNameHere
Non-Masked Image Name:	ScriptWillPutNameHere
Image Source:	OD for autoscaling to 0...16k
Image Side Length:	ScriptWillPutLengthHere
Magnification (between sample and recording medium):	50000
Pixel Size on Recording Medium (in micrometer):	5.0
Calculate Magnification from PixelSizes on Recording Medium and Sample Level?:	Yes [✓ No]
Pixel Size on Sample Level (in Angstroem):	1.0
Comment:	-

Defocus

Determine Defocus:	[✓ Yes] No

Tilt Geometry

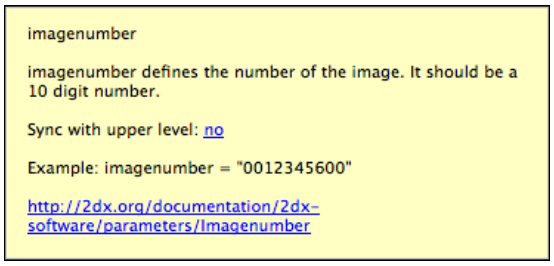
Determine Tilt Geometry:	Yes [✓ No]

```
################################2dx_initialize################################

                        ************************
                         Welcome to 2dx_image
                        ************************
               (activate the MANUAL button top right for help)

###########################2dx_initialize finished.###########################
```

```
Th.Mag=      ---                 QVAL=  ---
IQs Unb.I                        QVal1= ---
   -      -     -    -    -    -     -   - -
IQs Unb.II                       QVal2= ---
   -      -     -    -    -    -     -   - -
IQs Unb.Syn                      QValS= ---
   -      -     -    -    -    -     -   - -
PowerBins        [A]  15   12    9     7   5 3
 (noise=7)        #    -    -    -     -   - -
Tilt Info      Defoc.  Latt.  SpSplit  Merge
Grd.TAxis        -       -       -       -
Grd.TAng         -       -       -       -
Xst.TAxis        -       -       -       -
Xst.TAng         -       -       -       -
```

Figure 26: Individual image processing

When `2dx_image` is launched it automatically runs the *Init Program* script as displayed by the progress bar in the control panel. The script checks the presence of all required external dependencies and the consistency of the image directory. Next you have to launch the *Init Files and Parameters* script, which prepares the input files for the later processing steps. The image parameters that certainly have to be altered are the *Magnification*, the *Pixel Size on Recording Medium*, *CS* and *kV* value of the used microscope. To save this parameters as default for new imported images click in the menu bar –> File –> *Save as Project Default*. If you are unsure what a parameter in the Parameter panel means, you can right click on the according parameter and a description will be displayed, where you also find the link to the online documentation as shown in Figure 27.



imagenumber

imagenumber defines the number of the image. It should be a 10 digit number.

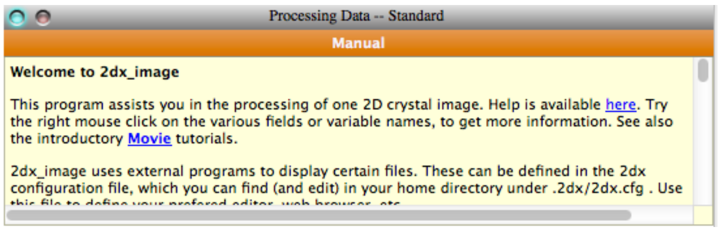
Sync with upper level: no

Example: imagenumber = "0012345600"

http://2dx.org/documentation/2dx-software/parameters/Imagenumber

Figure 27: Parameter help window

On the left of the graphical user interface is the Standard Scripts panel. For every script there is a manual entry that can be displayed when selecting the manual button in the control panel (Figure 28)



Welcome to 2dx_image

This program assists you in the processing of one 2D crystal image. Help is available here. Try the right mouse click on the various fields or variable names, to get more information. See also the introductory Movie tutorials.

2dx_image uses external programs to display certain files. These can be defined in the 2dx configuration file, which you can find (and edit) in your home directory under .2dx/2dx.cfg . Use

Figure 28: Manual View in `2dx`

5.3 Automated processing

The image processing workflow of a 2D crystal is in the order of the scripts in the Standard Scripts panel (Figure 29). To evaluate the periodicity of the structure the image is transformed into Fourier space, where the lattice is then determined determined in reciprocal space. This lattice is then transformed into real space, and is used to identify the aberrations from the ideal lattice by cross correlating with an idealized reference map. The Contrast Transfer Function (CTF) is then used to correct distortions originated from the imaging.

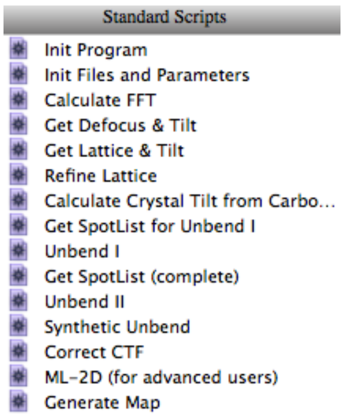


Figure 29: Standard scripts in `2dx_image`

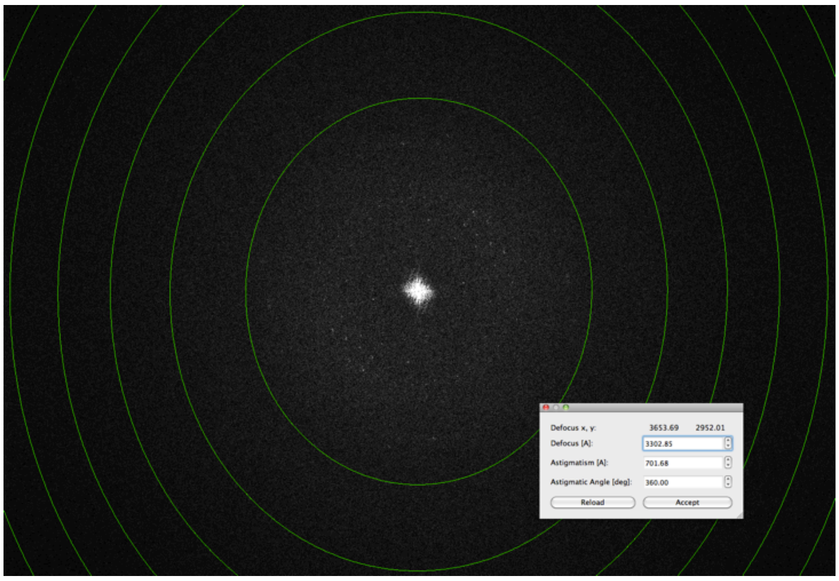
Since `2dx_image` is able to process the image in automated manner, select all the scripts in the Standard Scripts panel and press the *Play* button in the control panel. The scripts are executed consecutively. The progress bar depicts, which script is running momentarily and the Log panel displays the text output of the running script. You can change the verbosity level of the output in the Log panel with the buttons in the left corner of the panel. When `2dx_image` has ran all scripts click on the last script of the workflow *Generate Map* to see its outcome. The result of the automatic unbending and CTF correction can be examined by double-clicking the *Non-symmetrized Map* in the Images panel. This will display 2×2 unit cells.

5.4 Manual processing

Since the automatic processing is not perfect you may want to improve your result manually. You can do this in `2dx_image` at any time and will still profit from the prior executed steps. In this section we will walk you through each step.

5.4.1 Determine defocus

The first step is to determine the defocus and astigmatism for the later CTF correction. To do so run the *Calculate FFT* and the *Get Defocus* script and double click on *FFT of Downsampled Image* in the Images panel. This will open up a down sampled Fourier transformation of the original image in the full screen viewer. You can zoom in with the period key and out with coma key. Press "C" in the full screen browser or select *View CTF* in the *Navigator* menu to see the Thon rings. You can adjust the defocus and astigmatism in the dialog window as shown below in Figure 30.



```
Defocus x, y:                 3653.69    2952.01
Defocus [A]:                  3302.85
Astigmatism [A]:              701.68
Astigmatic Angle [deg]:       360.00
```

Figure 30: Determine the best defocus of an image

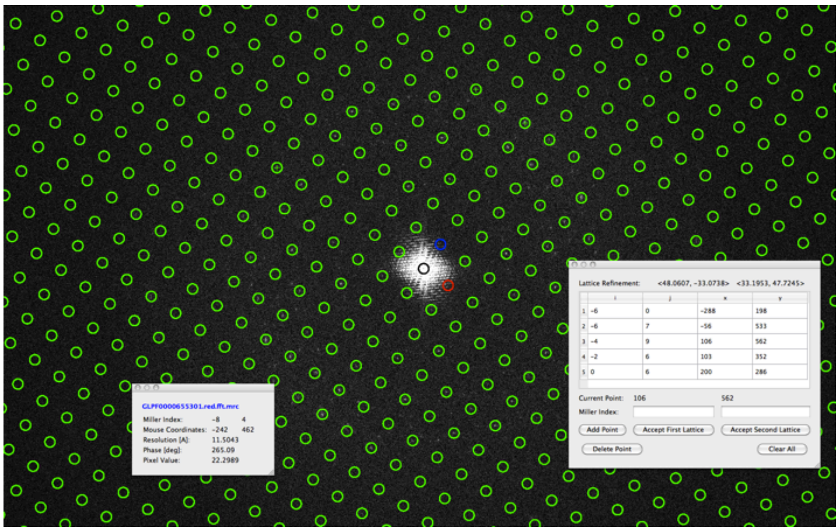
If you have trouble seeing the Thon rings you may want to change the Brightness and the contrast in the full screen viewer. You can do so by selecting *Adjust Contrast/Brightness* dialog from the *Navigator* menu. Press *Accept* if you are happy with the parameters, or use the shortcuts "B" and "N".

If you still have problems seeing the Thon rings you can create a periodogram. To do enter the *Advanced* view in the Parameter panel of the *Calculate FFT* script by clicking the right button in the upper left corner of the panel. This will give you the full parameters list. Select *Yes* for *Generate periodogram?* and run the script again. The periodogram will show up in the Images panel and in this image you might see the Thon rings more clearly and can select them more easily. The selection is made in the same matter as before.

The defocus and astigmatism values will be used later in the *Correct CTF* script.

5.4.2 Determine lattice

The next step is to determine the crystal lattice. You can run the *Get Lattice & Tilt* script in order to find automatically your lattice. Open *FFT of Downsampled Image* in the full screen browser. You can display the lattice by pressing the "L" key. By pressing the D key or selecting *Display Coordinate Info* from the *Navigator* menu, you will get a dialog window that displays the amplitude, phase, the Miller index etc. of the pixel that the mouse pointer hovers over. To define the lattice de novo or to refine the lattice you have to enter the *Lattice Refinement Mode* through the *Lattice Refinement* in the *Navigator* menu as shown in Figure 31.



GLPF0000655301.red.fft.mrc

Miller Index:	-8	4
Mouse Coordinates:	-242	462
Resolution [A]:	11.5043	
Phase [deg]:	265.09	
Pixel Value:	22.2989	

Lattice Refinement: <48.0607, -33.0738> <33.1953, 47.7245>

	i	j	x	y
1	-6	0	-288	198
2	-6	7	-56	533
3	-4	9	106	562
4	-2	6	103	352
5	0	6	200	286

Current Point: 106 562

Miller Index:

Figure 31: Lattice of an image

Now you are able to add spots to your lattice. These would ideally be spots with a high resolution. With a double click the spot with the maximum intensity value in the surrounding area of your mouse pointer is selected. By pressing the Enter key or the *Add Point* in the *Lattice Refinement* dialog window the spot is added to the table. The Miller index is chosen according to the existing lattice. In case this is wrong you can alter it in the dialog window. You can activate the *Display Parameters* dialog window in the Navigator window. Here you can change the *Maximum Value Search Range* that is used when double clicking. Or you can change the method to use a *Gauss Fit*. To get a closer look at the area you can make a right click.

After having added spots press the *Accept First Lattice* button and you will see how the lattice is shifted to match the picked spots.

Ideally you will have only one lattice. But usually there is a second weaker lattice showing in Fourier space. This lattice can be defined and refined in the same manner as with the first lattice. You can view this lattice by pressing the S key. The second lattice could later be used to filter out spots that overlap in the process of unbending.

You can calculate the unit cell dimensions which is defined by this lattice by running the *Evalute Lattice* script from the Custom Scripts panel.

5.4.3 Determine tilt geometry

If you are processing a non-tilted image you should set *Determine Tilt Geometry* to *No* in the *Tilt Geometry Section* of the Processing Data panel. In case you are processing a tilted image there are two ways to determine the tilt geometry of your image. Firstly, you can determine the tilt geometry from the defocus gradient of your Fourier transform (see Figure 32). This will be executed together with the *Get Defocus & Tilt* script. This way you determine the tilt geometry of the carbon film support, which is described by the variables TLTAXIS (tilt axis of the microscope) and TLTANG (angle between tilt axis and the grid). These two variables are valid for the sample support, and have nothing to do with the crystal. In fact, they could also be determined in absence of any crystal. Please note, that for smaller tilt angles, the defocus gradient is the safer method.

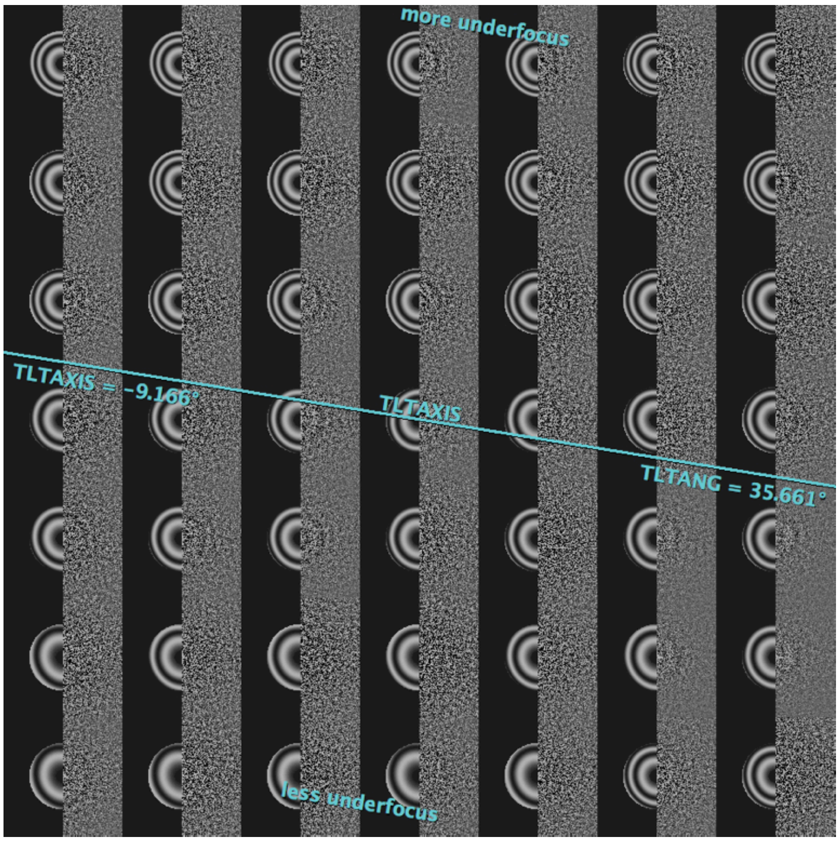


Figure 32: Determining tilt geometry from the defocus gradient.

Secondly, you can determine tilt geometry based on the lattice distortion. For this purpose you have first to define the lattice, for example after running the *Get Lattice & Tilt* script. In addition to the TLTAXIS and the TLTANG it will also calculate TLTAXA, which describes the orientation of the crystal with respect to the tilt axis (see Figure 33). This lattice-based tilt geometry is only reliable for larger tilt angles, which should be greater than 25 degrees. In that case, the *Get Lattice & Tilt* script will use the now determined lattice orientation to translate the defocus-based tilt geometry into the sample tilt geometry.

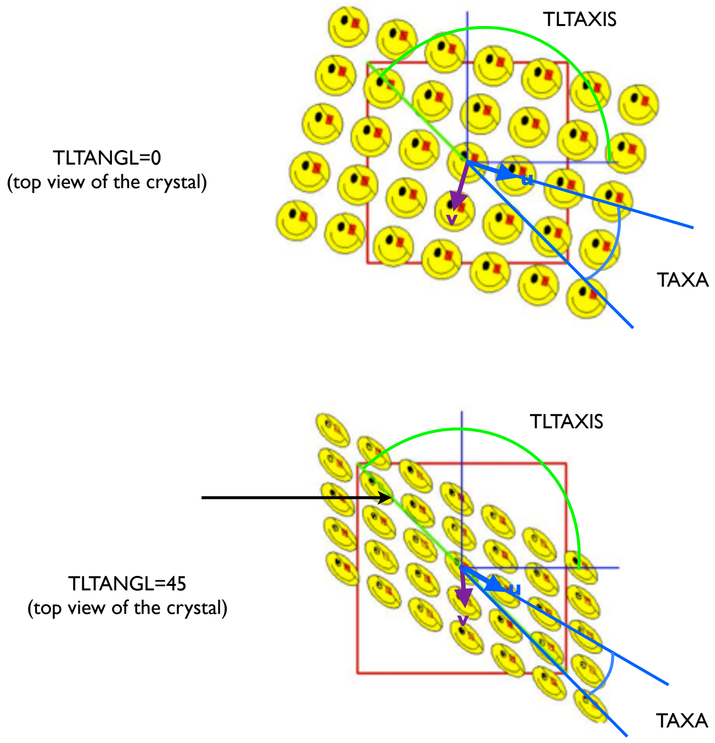


Figure 33: Determining tilt geometry from the lattice distortion.

5.4.4 Unbend

This chapter describes how you can correct for translational crystal distortion in `2dx_image`. Therefor you compare (= cross-correlating) a tightly masked and thus "perfect" transform (= reference) with a loosely masked transform that still contains all the information about irregularities in the lattice. However, to get the best results one first needs to determine the xy-coordinates that will serve to centre the reference area (see **??**). The cross-correlation is done most conveniently in reciprocal space (`twofile`) because in this case it amounts to a multiplication of the loosely masked transform of the image with the complex conjugate of the transform of the reference area. Backtransformation of the cross-correlation map is the first step to translate its information into real space disorder parameters. Consecutively, all unit cells can be searched for the exact position (= offset) of the object and the degree of similarity with the reference area (`quadserchb`). In practical terms this is achieved by first correlating the object (or part of it) with itself. This so called autocorrelation map is calculated from a small part of the reference area and its centre appears as a more or less symmetrical peak whose shape depends on the actual distribution of Fourier terms in the image (`autocorrl`). Finding the best fit of this central peak and the cross-correlation peaks for the individual unit cells provides the length and orientation of the distortion vectors and goodness (=height) of correlation. It should be mentioned that `quadserchb` only determines the translational offset for the cross-correlation peak but does not take into account any rotational disorder. Once the distortion vectors are known, the original image can be corrected by re-interpolating its optical densities to bring the unit cells onto a "perfect" lattice (`ccunbend`). The resulting unbent image is Fourier transformed to give an FFT with sharper diffraction peaks. The unbent FFT is evaluated (`mmbox`), yielding a list of amplitudes and phases for each lattice spot, together with an IQ value for each spot, which express the signal-to-noise ratio. The weighted sum of all IQ spots is depicted in the QVal, which can be used to judge the quality of the results of image processing.

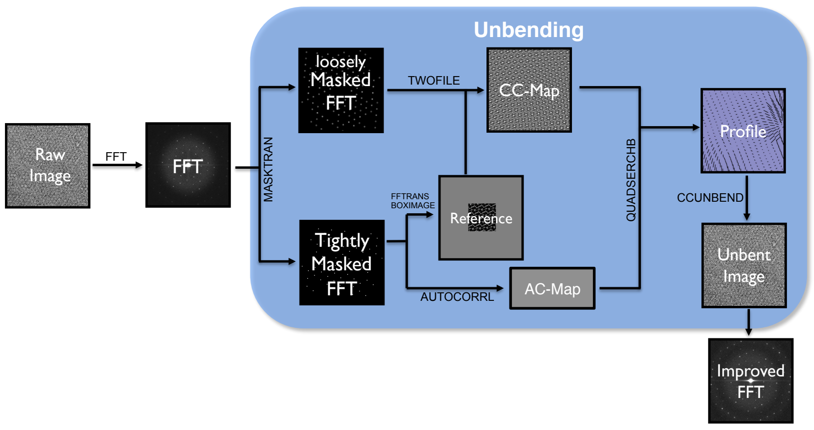


Figure 34: Overview of the unbend process.

The unbending is split into two rounds as depicted in the Standard Scripts panel by the *Unbend I* and *Unbend II* scripts. The needed reference for each unbending step is created from a spot list, which holds spots in Fourier space that lie on the lattice (scripts *Get Spotlist for Unbend I* and *Get Spotlist complete*. You can manually refine this spot list in the full screen view of the Fourier transformation. Enter *Spot Selection Mode* via *Spot Selection* in the Navigator Menu. With the P key you can identify the spots automatically selected. You may want to display the lattices when refining the spots (Figure 35). By clicking on a lattice spot you select it and by clicking it again you undo the selection. In this manner select now the spots that have a high signal to noise ratio. Save the spot list under *Spot List* in the *Navigator* menu.

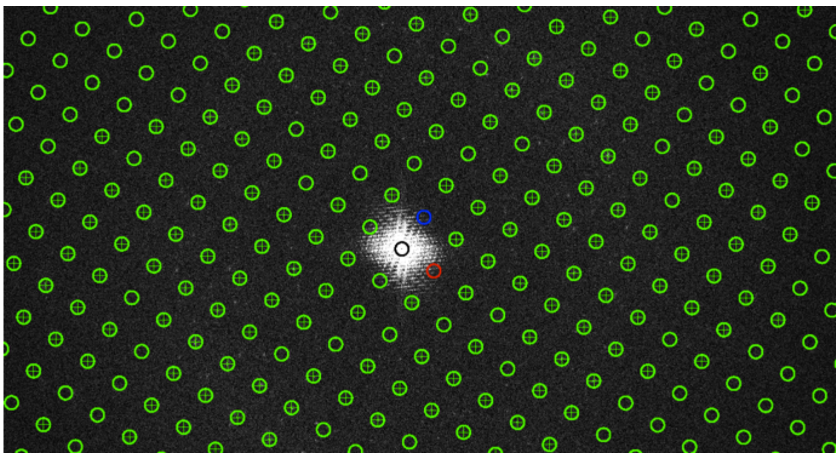


Figure 35: Spot refinement mode

From this outline it becomes clear that the quality of the reference area is the main factor to the success of the "unbending" procedure because any disorder present in the reference will remain uncorrected. Accordingly, improvements in the quality of the reference area by successive passes of processing make the determination of the lattice distortions and consequently, the correction of the image more accurate. In most cases the result will not get any better after 2-3 passes of image filtering and lattice unbending. However, a further improvement can be obtained once a 3D model is available, because in this case a "perfect" reference area can be created de novo by calculating a back projection of the model for the "precise" imaging conditions (`maketran`, see Section 6.2). Further parameters you can play around with are the radius of the loosely masked Fourier (*mask*) and the diameter of the reference (*box*).

5.4.5 Crystal masking

In real world applications the crystal almost never fills the entire image as shown in Figure 36.

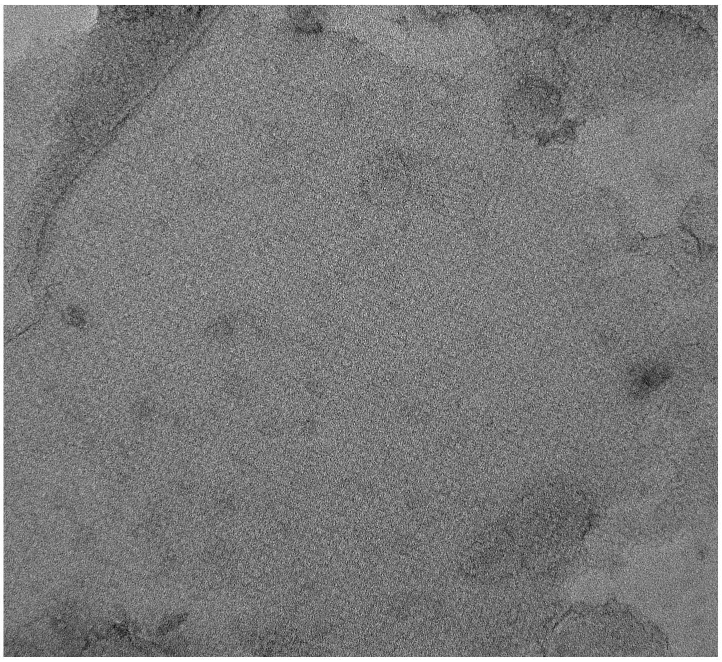


Figure 36: Non perfect crystal

Therefore it does not make sense to use regions where the crystal is not regular or where we don't have any crystalline structure. `2dx_image` offers a tool with which you can manually mask an image. The following walkthrough explains how you can mask the crystal.

1. There are multiple ways how you can mask your image. You can mask the crystal automatically by selecting the option *Do automatic masking of 2D crystal* in the Parameter panel of *Unbend II*. Alternatively, you can mask your crystal manually. Open the original image from the result panel of the initialization script and mask the image there. For many applications this does not work, as the crystal is not visible by eye. Thus we advise to mask in the file *XCF Map for Manual Merging* which can be found in the result panel of *Unbend II*. This file shows you the cross-correlation profile generated during unbending the crystal. The height of the peak corresponds directly to the local quality of the crystal. In regions with no crystalline structures we will not find any peaks. As usual the file is opened in the fullscreen mode by a simple double click.

2. In order to be able to mask the image you have to activate the *Polygon Selection Mode* by right-clicking in the image and selecting "Polygon Selection > Polygon Selection Masking" as shown in Figure 37.

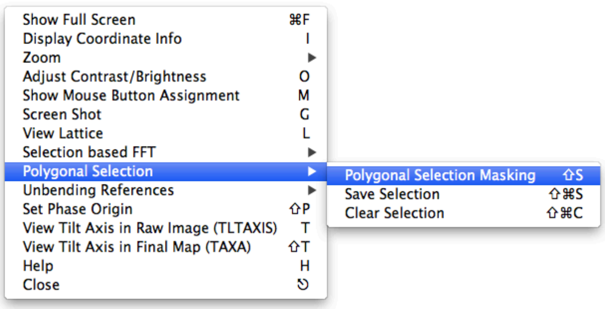

Show Full Screen ⌘F
Display Coordinate Info I
Zoom ▶
Adjust Contrast/Brightness O
Show Mouse Button Assignment M
Screen Shot G
View Lattice L
Selection based FFT ▶
Polygonal Selection ▶
Unbending References ▶
Set Phase Origin ⇧P
View Tilt Axis in Raw Image (TLTAXIS) T
View Tilt Axis in Final Map (TAXA) ⇧T
Help H
Close ⎋
Polygonal Selection Masking ⇧S
Save Selection ⇧⌘S
Clear Selection ⇧⌘C

Figure 37: Activate the Polygonal Mode

3. Now you can mask your crystal by defining the borders with a bunch of left mouse clicks. A double-click will automatically close the selected polygon. Note that you can delete the current selection by selecting "Polygon Selection > Clear Selection" from the menu appearing when right-clicking on the image. Your masked image should be somehow similar to Figure 38.

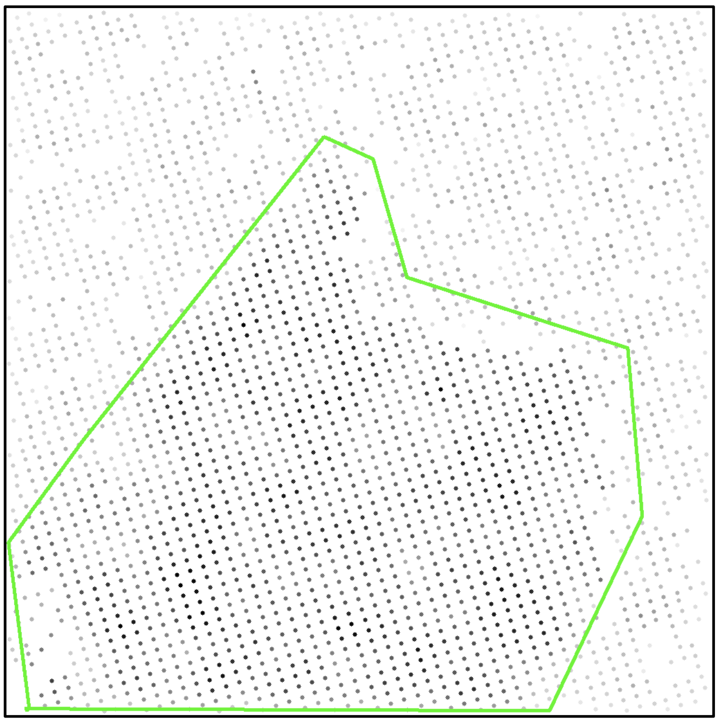


Figure 38: Masked XCF Map

4. Once you are happy with your selection you can save the current selection by selecting "Polygon Selection > Save Selection" from the menu appearing after another right-click.

5. In order to apply your polygon selection you have to run the custom

script *Mask Crystal from Polygon*. This script will replace all the pixels outside the polygon with the average grey value inside the polygon. In the future calculation this new image will be used as input image.

6. After masking the crystal from the saved polygon you have to save the current configuration by means of clicking on the save button in the header of the `2dx_image` graphical user interface. Afterwards you have to rerun all the standard scripts, except *Get Defocus & Tilt* and *Get Lattice & Tilt* as neither the defocus nor the lattice are effected by the masking. This will generate a density map uniquely from the data points inside the selected polygon.

5.4.6 CTF Correction

The next step consists in an initial correction of the phase data for the effect of the contrast transfer function (`ctfapply`). The CTF is a modulation of the scattered waves by the objective lens, which results in periodical contrast reversals across the image. In Fourier space this corresponds to a phase shift by 180° where the CTF is negative. Because this modulation will be different for each image (due to their different amounts of defocus, see Section 5.4.1) the phases must be corrected to allow the combination of data from several images. The visual manifestation of the CTF is a characteristic pattern of alternating light and dark bands (= Thon rings) in the diffuse diffraction patterns of amorphous materials (see Figure 30. The dark areas correspond to frequencies that are not or only poorly transferred and hence do not contribute to the formation of the image. Vice versa, good transfer is achieved in the bright areas. In order to correct for CTF in your images run *Correct CFT* script in the Standard Scripts panel. In addition, this scripts gives you an IQ-Plot.

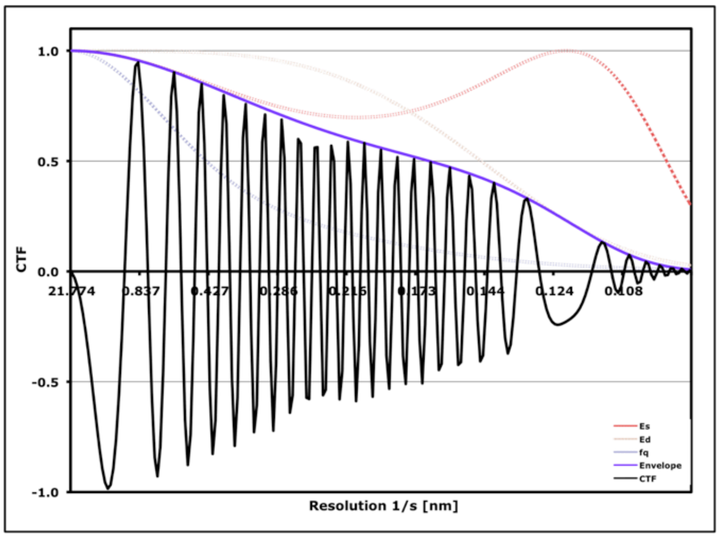


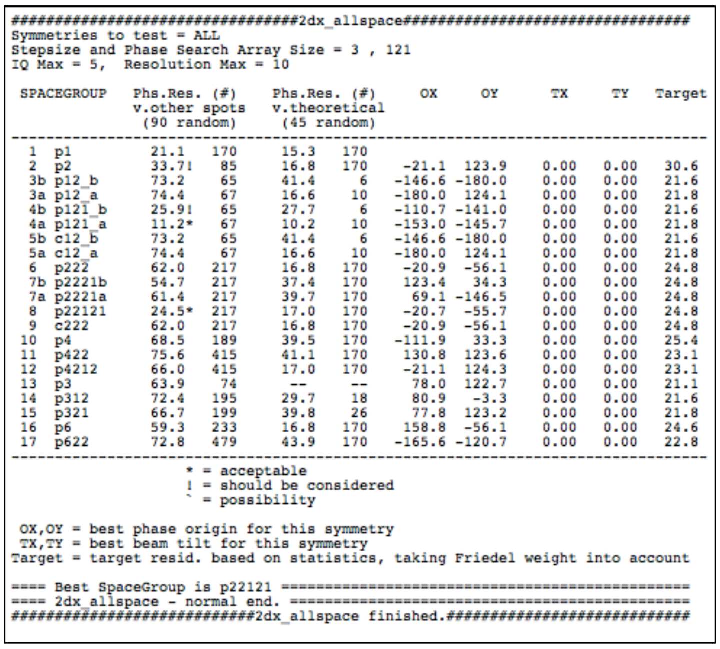
Figure 39: Example of a Contrast Transfer Function

5.4.7 Space Group Determination

CTF-corrected data can always be combined in "plane group p1", i.e. assuming no symmetry at all. However, if the specimen does exhibit a certain symmetry, data handling and refinement becomes a lot easier because symmetry imposes constraints on the molecular transform. For instance, a simple twofold axis of symmetry means that the densities within the unit cell are related by a 180°rotation about this axis. Consequently, the density distribution will appear symmetrical in non-tilted image data (= projection density map). To encode this in reciprocal space requires all the projection terms to be symmetric about the origin, i.e. they are cosine waves that can only be shifted by 0 or 180°with respect to the phase origin. The introduction of redundancy is a further advantage of symmetry. For instance the terms (1 0 0), (0 -1 0) and (-1 1 0) are all different if no symmetry is present. However, if the specimen displays three- or six-fold symmetry these terms have identical phases and amplitudes, i.e. they are symmetry related, describing the same structural feature. In other words while the data of a single image only contribute a single measurement for each reflection in the first case, three independent measurements of the "unique" reflection (= 1 0 0) are obtained for three- or six-fold symmetry. In addition, the actual phase values allow discriminating further between three- (p3) and six-fold (p6) symmetry. Since the latter has intrinsic twofold symmetry all phases would be expected to adopt 0 or 180°values. However, this is not true in p3 where like in p1 no phase constraints are applicable. The agreement between symmetry related reflections is independent of the CTF as long as the image has no significant astigmatism because they have the same distance from the transform origin. In summary, the relationships between the phases obtained from images of non-tilted crystals allow an unambiguous determination of the specimen's symmetry because each of the 17 plane groups has a unique set of constraints.

To determine the space group of your crystal run the *Get Spacegroup & Phase Origin* script in the *Custome Scripts* panel. It will calculate the internal phase residuals for all space groups and thereby choose the optimal space group and the corresponding phase origin. The space group and phase origin are then in `2dx_image` as entries for *Symmetry and Phase Origin* (see Figure 40). To find the right space group it is better to set the *Resolution limitation for phase origin search* to a lower resolution than with what you process your image. In case you know the space group of your crystal you can also use this script to just find the according phase origin for the image. Note that only projection images of non-tilted crystals show symmetry.

Optionally you can set your phase origin manually. This way you could process all your single images to a similar phase origin, making it later easier to find a common phase origin in 2D merging Chapter 6. For this purpose run the *Set PhaseOrigin Manually* script in the *Custom Script* panel. Open the *Old p<X>-symmetrized Map* in the *Image* panel. Position your mouse cursor on the wished phase origin and press shift "P". The needed phase origin changes will be transferred to the *Parameter* panel. To apply the changes you have to rerun the script once again.



```
#################################2dx_allspace#################################
Symmetries to test = ALL
Stepsize and Phase Search Array Size = 3 , 121
IQ Max = 5,  Resolution Max = 10

 SPACEGROUP    Phs.Res. (#)    Phs.Res. (#)      OX      OY       TX      TY   Target
               v.other spots   v.theoretical
                (90 random)     (45 random)
--------------------------------------------------------------------------------------
  1  p1         21.1   170      15.3   170
  2  p2         33.7!   85      16.8   170    -21.1  123.9     0.00    0.00    30.6
  3b p12_b      73.2    65      41.4     6   -146.6 -180.0     0.00    0.00    21.6
  3a p12_a      74.4    67      16.6    10   -180.0  124.1     0.00    0.00    21.8
  4b p121_b     25.9!   65      27.7     6   -110.7 -141.0     0.00    0.00    21.6
  4a p121_a     11.2*   67      10.2    10   -153.0 -145.7     0.00    0.00    21.8
  5b c12_b      73.2    65      41.4     6   -146.6 -180.0     0.00    0.00    21.6
  5a c12_a      74.4    67      16.6    10   -180.0  124.1     0.00    0.00    21.8
  6  p222       62.0   217      16.8   170    -20.9  -56.1     0.00    0.00    24.8
  7b p2221b     54.7   217      37.4   170    123.4   34.3     0.00    0.00    24.8
  7a p2221a     61.4   217      39.7   170     69.1 -146.5     0.00    0.00    24.8
  8  p22121     24.5*  217      17.0   170    -20.7  -55.7     0.00    0.00    24.8
  9  c222       62.0   217      16.8   170    -20.9  -56.1     0.00    0.00    24.8
 10  p4         68.5   189      39.5   170   -111.9   33.3     0.00    0.00    25.4
 11  p422       75.6   415      41.1   170    130.8  123.6     0.00    0.00    23.1
 12  p4212      66.0   415      17.0   170    -21.1  124.3     0.00    0.00    23.1
 13  p3         63.9    74       --     --     78.0  122.7     0.00    0.00    21.1
 14  p312       72.4   195      29.7    18     80.9   -3.3     0.00    0.00    21.6
 15  p321       66.7   199      39.8    26     77.8  123.2     0.00    0.00    21.8
 16  p6         59.3   233      16.8   170    158.8  -56.1     0.00    0.00    24.6
 17  p622       72.8   479      43.9   170   -165.6 -120.7     0.00    0.00    22.8
--------------------------------------------------------------------------------------
                       * = acceptable
                       ! = should be considered
                       ` = possibility

 OX,OY = best phase origin for this symmetry
 TX,TY = best beam tilt for this symmetry
Target = target resid. based on statistics, taking Friedel weight into account

==== Best SpaceGroup is p22121 ===============================================
==== 2dx_allspace - normal end. ==============================================
############################2dx_allspace finished.############################
```

Figure 40: Example of space group determination.

5.4.8 Map Generation

Final step of 2D image processing is the generation of the projection map. You can create an unsymmetrized map by running the *Generate Map* script in the *Standard Scripts* panel. Alternatively you can generate a symmetrized map by applying the selected space group. For this run run *Generate Map* script in the *Custom Scripts* panel. The latter will calculate a table of phase residuals during the symmetrization, from which you can judge the overall quality and resolution of your image.

Please note that you have to set the parameter *Invert contrast to the final map again* to *No* in case you are processing a negative stained image, and to *Yes* in case of a cryo image.

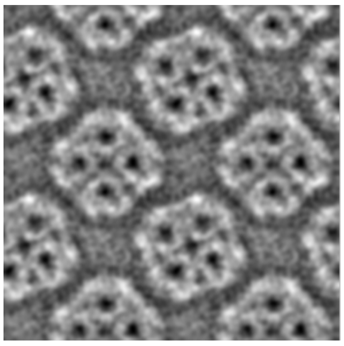


Figure 41: Example of a projection map

6 2D Merging

In the previous sections we have seen how to process individual images of 2D crystals. From a single micrograph we generate a list of spots containing their amplitudes and phases in Fourier space. This section describes how to merge the projection maps from each micrograph of an non-tilted specimen in order to produce one common projection map. This is similar to creating class averages in single particle electron microscopy, where projections of particles with the same orientation are averaged together.

In contrast to single particle classification in the context of 2D crystals we already know the projection direction from the electron microscope, all under the assumption that the crystals are perfectly flat. The central section theorem gives us the mathematical backbone to consider each image created by a transmission electron microscope as a projection of the crystal. Thus all images of non-tilted 2D crystals represent a projection of the crystal in z direction. In the previous section we have seen how filtering the Fourier transform of 2D crystal results equals averaging the proteins in real space. The task of 2D merging is to average the Fourier components of the images of untitled crystals together.

To be able to average the densities gained from each image one has to align the images first. They have to be in register with each other which is best done by aligning them to a common reference. As we see later in the practical part the reference is just one of the images chosen by the user. The alignment can also be understood as setting a common origin.

We again make use of the Fourier theory for this task. For single image processing we only considered at the amplitudes of the Fourier transform and the lattice it spans. This lattice always had a common origin and approximately the same dimensions, only the orientation varies across the images. This is due to that the magnitude of the Fourier Transform does not change even when the image is shifted. Therefore averaging the amplitudes is just averaging the diffraction spots with the same miller indices. The phases of the Fourier transform though determines the origin of the image in real space. Thus aligning all the images results in shifting the phases of each image to match with the reference.

The last step of 2D merging is averaging. The simplest way would be to take the mean of the values for each unique reflection depicted by a Miller index. But the program `AVRGAMPHS` written by Richard Henderson weights each reflection based on its IQ value. Hence each single diffraction spot of every image contributes to the average according to its signal-to-noise ratio. For the user this has the advantage that even images of bad quality can be merged without making the resulting 2D map worse, additionally one can still benefit from the few spots that have a good IQ value. The final merged 2D map is then the inverse Fourier transform of the list of averaged amplitude and phases.

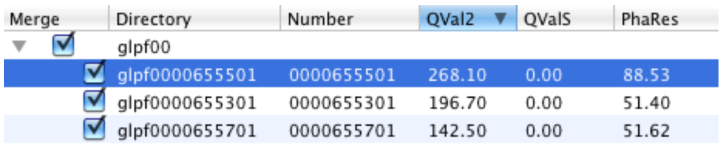
6.1 2D Merging in `2dx`

Having processed all the images of non-tilted specimen with `2dx_image` we are ready to merge them to an average projection map. As the name states we need `2dx_merge` for this task.

1. If not already open, start up `2dx_merge` and select the project you want to work on.

2. The parameter *Modus of Merging* has to be set to *2D* in the "Processing Data" panel.

3. We want to align the images to a common phase origin. Therefore we have to select one image as reference. Usually the image with the highest QVal and/or lowest phase residual error is used. The QVal can be read from the "QVal2" column in the project panel. To make it easier one can also sort the images by the this attribute by clicking on the title of the column as shown in Figure 42.



Merge	Directory	Number	QVal2 ▼	QValS	PhaRes
▼ ✓	glpf00				
✓	glpf0000655501	0000655501	268.10	0.00	88.53
✓	glpf0000655301	0000655301	196.70	0.00	51.40
✓	glpf0000655701	0000655701	142.50	0.00	51.62

Figure 42: Select the image with highest QVal as a reference.

4. When choosing the reference make sure that the phase origin of the image was chosen accordingly the present space group (Section 5.4.7). If the image suffers from a wrong phase origin you should set the phase origin manually for the image as described in the phase origin section Section 5.4.7.

5. The appropriate reference should be selected with the tick mark in the file panel. Make sure no other images are selected.

6. Before creating the reference one should to chose the resolution the data on which the alignment process can rely on. This resolution is specified by the parameter *Resolution of the merged dataset for the reference*. For the GLPF test data set we set this value initially to 15 Å.

7. Make sure the *Symmetry* is set correctly in the processing data panel because the selected symmetry will be applied to the reference, i.e. correction of symmetry-restricted phase values and removal of symmetry-forbidden reflections.

8. Now run the Standard Script *Merge Once*. This script merges the data of the selected image into a reference file. Several output files are created as listed in the Images panel. The essential ones are the merge.aph, which is the list of merged amplitude and phases, and the MTZ file, which is then used as reference named *MTZ: Merged full reciproc. space 2D data for reference*. Double clicking the MTZ file will show the content in your web browser. The file mainly consist of a reflection list as already seen in the APH file. Every reflection is depicted by there index (H,K,L), where the L is set to 0 in the 2D merging. Followed by amplitude and phase, which have been CTF corrected and symmetrized. The MTZ file also contains information about the crystal unit cell dimensions, the symmetry, the resolution range, etc.

Now that we have created a reference we can start aligning the non-tilted image data.

1. Uncheck the image you have selected to be the reference and select all other images recorded from non-tilted specimen that you want

to align.

2. With the script *Refine Once* you can now align the selected images to the reference. All you need to do is to specify the range over which the phase origin search should be performed. Unless you have already manually selected the phase origin in all of the projection maps (Section 5.4.7) we advise you to initially cover the entire crystal unit cell i.e. a search range of 360° phase angle. The search range is defined by the parameters *Stepsize of the phase origin search* and the *Number of steps in the phase origin search* in the *Merging* section. In a first phase origin search you should use a coarse step size of 6° which with a number of 60 steps will go from -180° to 174°. In later stages you should decrease the step size to a smaller value in order to perform a local refinement of the phase origin.

3. Run the *Refine Once* script. This will refine the phase origin for all selected images with the help of the MRC program `origtilt`. The last phase origin change for every image will be listed as *phaori_last_change* in the Results panel as well as in the *PhaOri Change* column of the Project panel.

4. Now we visually inspect the phase origin change for each projection map by running the *Generate Image Maps* script. Therewith the projection map are updated with the refined phase origin and is reflected in the Image Preview panel. You can scroll through the images in the Images panel or the Project panel. If you only want to see the selected images in the Project panel you can select the option *Show Only Selected Directories* from *View* menus illustrated in Figure 43. The same menu option can be used to switch back to view all images.

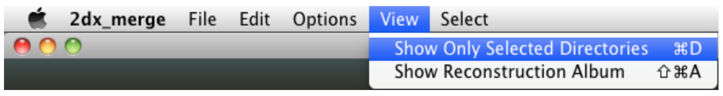

2dx_merge
File
Edit
Options
View
Select
Show Only Selected Directories ⌘D
Show Reconstruction Album ⇧⌘A

Figure 43: Shows only the selected images in the Project panel.

5. Run *Refine Once* again with the current step size until the phase origin of all refined images does not improve anymore. Depending on the step size it might happen that the optimal phase origin is in-between the two considered phase origins and will jump back and forth. This is also a sign that you have reached the optimum for this step size and should decrease it for the next merge and refinement runs.

6. Besides the updated parameters in the results panel, the script also produces output files in the images panel. If you are curious how `origtilt` is called to refine every single image you should have a look at *CSH: refinement script*. The output of `origtilt` might be more interesting and is in the text file listed as *LOG: origtilt B output*. This file should be considered when evaluating the refinement process. It contains the list of reflections for all refined images. The most important value in this file though is the refined phase origin. It is followed by a table of resolution ranges with the associated phase residual for the reflections in that range. This allows the ver-

ification up to what resolution the refinement is trustworthy. One thing to keep in mind is that for the coarse refinement we limited the resolution of the reference. After the resolution table in *LOG: origtilt B output* a cross-correlation map is plotted through a matrix where each entry represents the normalized cross-correltation value between the refined image and the reference. It should contain a clear peak which should end up in the center after the last rounds of refinement.

7. After the first cycle of refinement we should update our reference with the refined data. Therefore include the image you initially used as a reference by adding it to the selection. Now run *Merge Once* again to create a new reference.

8. If you set the *Symmetry* in the Processing Data panel and the symmetry is above space group P2 you will get a table named "Phase Residual In Resolution Ranges". You should consider the phase residuals as measure to what resolution you can trust your data and adapt the *Resolution of the merged dataset for the reference* accordingly. The resolution ranges of the table list the phase residual up to the *Upper Resolution limit* plus 1Å further.

9. With the new reference at hand we can preform a finer refinement of the phase origin. Hence the *Stepsize of the phase origin search* should be decreased to e.g. 0.5Å. In any case make sure that the step size together with the number of steps at least span the range of the previous step size (In our case 6Å).

10. Now run *Refine Once* again for the refinement until the phase origin for each image does not change anymore i.e. *PhaOri Change* will be zero. The cross-correlation maps in *LOG: origtilt B output* should now show sharp and centered peaks.

11. Until now we have always executed the merge and refine process separately. But when you are at the point where the images merged to a reference are the same set as the images being refined i.e. the selection does not change, you can also use the *Merge & Refine (Iterative)* script. As the name states this runs several iterations of the merge and refine cycle by default 3 rounds. The iteration number can be changed by clicking as illustrated in Figure 44.

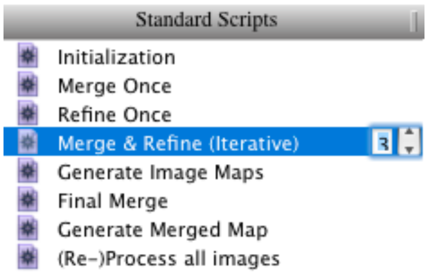
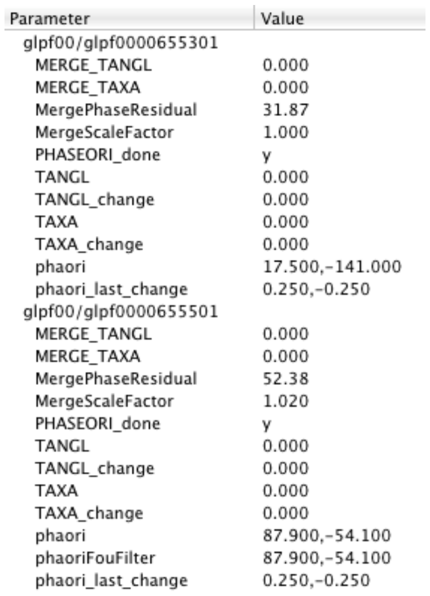


Figure 44: Changing the number of times the *Merge & Refine* script is run by double clicking the counter.

12. When using *Merge & Refine (Iterative)* script one should have a look at the parameters *phaori_last_change* and *MergePhaseResidual* in the Results panel like in Figure 45. As we have mentioned before

the phase origin change should decrease with each iteration and ideally end up being zero. The measure of how good the refined phase origin is quantified *MergePhaseResidual*. This value will also decrease in the process of refinement. If the value remains around 90° then the alignment is not working.



Parameter	Value
glpf00/glpf0000655301	
MERGE_TANGL	0.000
MERGE_TAXA	0.000
MergePhaseResidual	31.87
MergeScaleFactor	1.000
PHASEORI_done	y
TANGL	0.000
TANGL_change	0.000
TAXA	0.000
TAXA_change	0.000
phaori	17.500,-141.000
phaori_last_change	0.250,-0.250
glpf00/glpf0000655501	
MERGE_TANGL	0.000
MERGE_TAXA	0.000
MergePhaseResidual	52.38
MergeScaleFactor	1.020
PHASEORI_done	y
TANGL	0.000
TANGL_change	0.000
TAXA	0.000
TAXA_change	0.000
phaori	87.900,-54.100
phaoriFouFilter	87.900,-54.100
phaori_last_change	0.250,-0.250

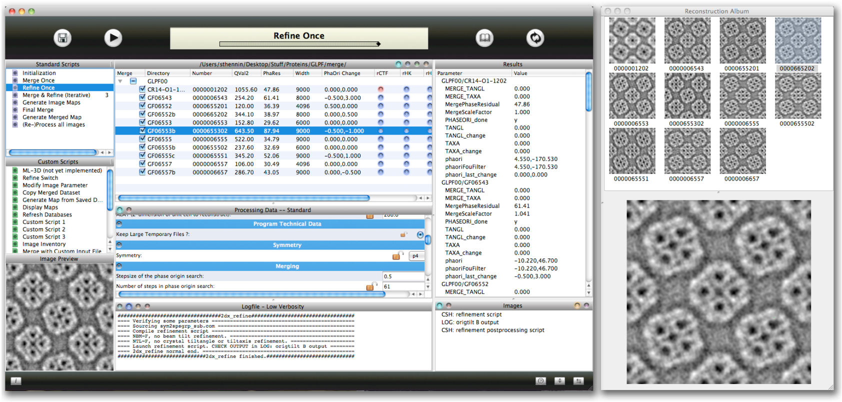
Figure 45: The results of each refinement iteration is displayed as list of parameters for each image in the Results panel. The most important parameters to monitor are *phaori_last_change* and *MergePhaseResidual*.

13. For more in-depth knowledge about your merged data set there is the option *List Reflections into logfile (ILIST)*. If this is enabled a separate log-file is created called *LOG: reflections after origtiltk*. This file is the complete list of reflections from every merged image. This means that for every H,K index the available values from the contributing images is listed. The values associated with each reflection besides the H,K index are the amplitude, phase, image number, IQ, film weight, background amplitude and CTF. Appending these values are dashed lines followed by the IQ value again. The number of dashed lines is inverse proportional to the IQ value i.e. a reflection with an IQ of 1 has the most dashed lines appended. This helps to identify the good reflections even in a large list. The reflections with IQ values of 1 to 4, even have the amplitude and phases repeated at the end. This should make it easier to detect reflections with data differing from the amplitude and phases of the same reflections stemming from other images. This deviation can be a sign for a wrong CTF correction in that image, therefore you should have a look if the defocus was detected correctly in this image.

14. If you did not add all the non-tilted images yet or have gained new images from the microscope you should use the existing reference to align the novel images in the same fashion by jumping back to step 1.

Once all the data of the non-tilted images are aligned the last step is to merge them all together to one.

1. As described above the final alignment should also be visually inspected by running the *Generate Image Maps* script again and then scrolling through the images. An alternative to looking at the individual projection maps one by one in the Image Preview panel is given by the *Reconstruction Album*. The *Reconstruction Album* shows for every image the projection map as thumbnail and when selected the image is shown large in the lower panel as illustrated in Figure 46. The album can be displayed in the View menu via *Show Reconstruction Album*. Deselect the miss-aligned images.



/Users/sthennin/Desktop/Stuff/Proteins/GLPF/merge/

Merge	Directory	Number	QVal2	PhaRes	Width	PhaOri Change
GLPF00						
✓	CR14-O1-1...	0000001202	1055.60	47.86	9000	0.000,0.000
✓	GF06543	0000006543	254.20	61.41	8000	-0.500,3.000
✓	GF06552	0000655201	120.00	36.39	4096	0.500,0.000
✓	GF06552b	0000665202	344.10	38.97	8000	0.000,0.500
✓	GF06553	0000006553	152.80	29.62	6000	0.000,0.000
✓	GF06553b	0000655302	643.50	87.94	9000	-0.500,-1.000
✓	GF06555	0000006555	522.00	34.79	9000	0.000,0.000
✓	GF06555b	0000655502	237.60	32.69	6000	0.000,0.000
✓	GF06555c	0000065551	345.20	52.06	9000	-0.500,1.000
✓	GF06557	0000006557	106.00	30.49	4096	0.000,0.000
✓	GF06557b	0000006657	286.70	43.05	9000	0.000,-0.500

Processing Data -- Standard

Program Technical Data
Keep Large Temporary Files ?:
Symmetry
Symmetry: p4
Merging
Stepsize of the phase origin search: 0.5
Number of steps in phase origin search: 61

Results

Parameter	Value
GLPF00/CR14-O1-1202	
MERGE_TANGL	0.000
MERGE_TAXA	0.000
MergePhaseResidual	47.86
MergeScaleFactor	1.000
PHASEORI_done	y
TANGL	0.000
TANGL_change	0.000
TAXA	0.000
TAXA_change	0.000
phaori	4.550,-170.530
phaoriFouFilter	4.550,-170.530
phaori_last_change	0.000,0.000
GLPF00/GF06543	
MERGE_TANGL	0.000
MERGE_TAXA	0.000
MergePhaseResidual	61.41
MergeScaleFactor	1.041
PHASEORI_done	y
TANGL	0.000
TANGL_change	0.000
TAXA	0.000
TAXA_change	0.000
phaori	-10.220,46.700
phaoriFouFilter	-10.220,46.700
phaori_last_change	-0.500,3.000
GLPF00/GF06552	
MERGE_TANGL	0.000

Logfile - Low Verbosity

```
################################2dx_refine################################
==== Verifying some parameters ==========================================
==== Sourcing sym2spsgrp_sub.com ========================================
==== Compile refinement script ==========================================
==== NBM=F, no beam tilt refinement. ====================================
==== NTL=F, no crystal tiltangle or tiltaxis refinement. ================
==== Launch refinement script. CHECK OUTPUT in LOG: origtilt B output ===
==== 2dx_refine normal end. =============================================
############################2dx_refine finished.##########################
```

Images

CSH: refinement script
LOG: origtilt B output
CSH: refinement postprocessing script

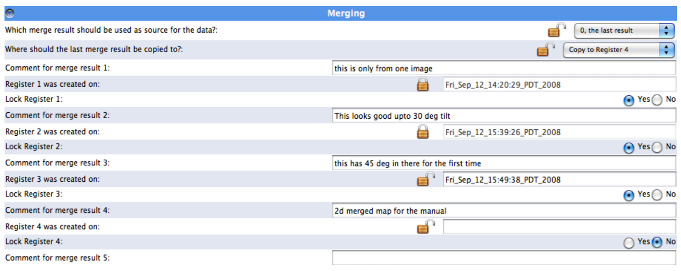
Figure 46: The Reconstruction Album gives an overview of the already processed images shows in form of projection map thumbnails and is a good alternative to the `2dx_merge` interface on the left.

2. Running the *Final Merge* script is similar to the previous merge steps, except that we will not use the result as a reference for further alignment, rather as resulting average 2D map. Hence it is of great importance to chose the *Upper Resolution Limit (RESMAX)* carefully. One should only trust the data up to a resolution where the phase residual are low. If you are looking at the table of phase residuals a good estimate if phase residual reflects meaningful data can be given by the following relation

$$phase\,residual \leq 90^\circ - \frac{90^\circ}{\sqrt{n}} \tag{1}$$

where n is the number of spots in a given resolution range.

3. The last step of 2D merging is to create a merged projection map through the *Generate Merged Map* script. This creates two kind of 2D maps: one in form of an MRC image and the other in form of a contour plot. For our example data set where we set the symmetry to P4 and the files the images are named *p4-symmetrized final 2D map* and *PS: p4-symmetrized final 2D map plot*. The *p4-symmetrized final 2D map* of the GLPF example data set should look similar to Figure 47.

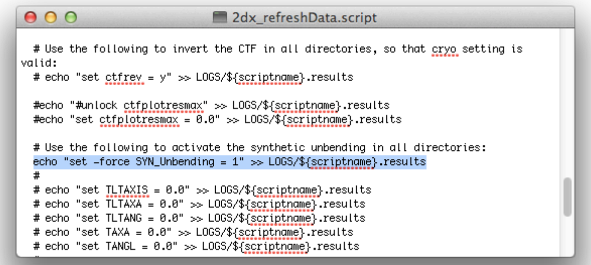


Which merge result should be used as source for the data?:	0, the last result
Where should the last merge result be copied to?:	Copy to Register 4
Comment for merge result 1:	this is only from one image
Register 1 was created on:	Fri_Sep_12_14:20:29_PDT_2008
Lock Register 1:	[✓ Yes] No
Comment for merge result 2:	This looks good upto 30 deg tilt
Register 2 was created on:	Fri_Sep_12_15:39:26_PDT_2008
Lock Register 2:	[✓ Yes] No
Comment for merge result 3:	this has 45 deg in there for the first time
Register 3 was created on:	Fri_Sep_12_15:49:38_PDT_2008
Lock Register 3:	[✓ Yes] No
Comment for merge result 4:	2d merged map for the manual
Register 4 was created on:	
Lock Register 4:	Yes [✓ No]
Comment for merge result 5:	

Figure 47: The 2D merged map of the GLPF example data set.

6.2 Synthetic Unbending in 2dx

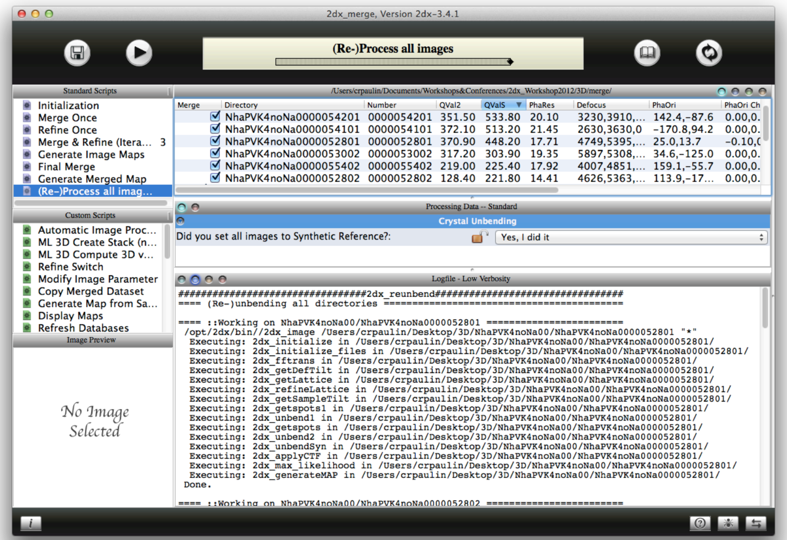
Once you obtained a first merged dataset you can go back and re-process all your single images. Your merged map represents so far the best model and can be used to create a "perfect" reference for unbending (see Section 4.3). This so called "synthetic reference" will be used in the *Synthetic Unbend* script in `2dx_image`.
In order to re-process all images you have first to switch all selected images to *Use Synthetic Reference*, by running the custom script *Custom Script* or *Refresh Databases* script in `2dx_merge`. Here you have to unlock the variable *SYN_Unbending to 1* as illustrated in Figure 48.



```
  # Use the following to invert the CTF in all directories, so that cryo setting is
valid:
  # echo "set ctfrev = y" >> LOGS/${scriptname}.results

  #echo "#unlock ctfplotresmax" >> LOGS/${scriptname}.results
  #echo "set ctfplotresmax = 0.0" >> LOGS/${scriptname}.results

  # Use the following to activate the synthetic unbending in all directories:
  echo "set -force SYN_Unbending = 1" >> LOGS/${scriptname}.results
  #
  # echo "set TLTAXIS = 0.0" >> LOGS/${scriptname}.results
  # echo "set TLTAXA = 0.0" >> LOGS/${scriptname}.results
  # echo "set TLTANG = 0.0" >> LOGS/${scriptname}.results
  # echo "set TAXA = 0.0" >> LOGS/${scriptname}.results
  # echo "set TANGL = 0.0" >> LOGS/${scriptname}.results
```

Figure 48: How to select synthetic reference for all images.

Now you can run the script *Re-Process all Images* in `2dx_merge`, which will automatically rerun all required scripts including *Synthetic Unbend* as illustrated in the Logfile panel in Figure 49. Similar to Unbend I & II, described in Section 5.4.4, you can adjust the *spot radius of the synthetical reference*.



Merge	Directory	Number	QVal2	QValS	PhaRes	Defocus	PhaOri	PhaOri Ch
✓	NhaPVK4noNa0000054201	0000054201	351.50	533.80	20.10	3230,3910,...	142.4,-87.6	0.00,0.
✓	NhaPVK4noNa0000054101	0000054101	372.10	513.20	21.45	2630,3630,0	-170.8,94.2	0.00,0.
✓	NhaPVK4noNa0000052801	0000052801	370.90	448.20	17.71	4749,5395,...	25.0,13.7	-0.10,(
✓	NhaPVK4noNa0000053002	0000053002	317.20	303.90	19.35	5897,5308,...	34.6,-125.0	0.00,0.
✓	NhaPVK4noNa0000055402	0000055402	219.00	225.40	17.92	4007,4851,...	159.1,-55.7	0.00,0.
✓	NhaPVK4noNa0000052802	0000052802	128.40	221.80	14.41	4626,5363,...	113.9,-17...	0.00,0.

Did you set all images to Synthetic Reference?: Yes, I did it

```
################################2dx_reunbend################################
==== (Re-)unbending all directories ========================================

==== ::Working on NhaPVK4noNa00/NhaPVK4noNa0000052801 ======================
 /opt/2dx/bin//2dx_image /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801 "*"
  Executing: 2dx_initialize in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_initialize_files in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_fftrans in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_getDefTilt in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_getLattice in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_refineLattice in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_getSampleTilt in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_getspots1 in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_unbend1 in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_getspots in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_unbend2 in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_unbendSyn in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_applyCTF in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_max_likelihood in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
  Executing: 2dx_generateMAP in /Users/crpaulin/Desktop/3D/NhaPVK4noNa00/NhaPVK4noNa0000052801/
 Done.

==== ::Working on NhaPVK4noNa00/NhaPVK4noNa0000052802 ======================
```

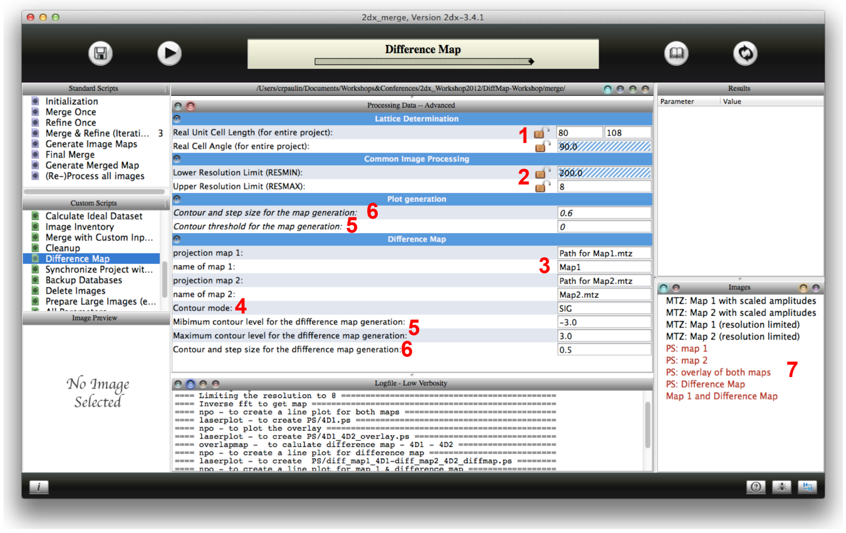
Figure 49: Re-processing images with Synthetic Unbending in `2dx_merge`
.

Reprocessing all images using a synthetical reference for unbending an image should lead to a significant improvement of the QVal values (here QValS) and of the phase residuals (see Figure 49). In some cases it also improves the resolution of your final merge map. Hence, *Synthetic Unbend* can be seen as a first refinement step for a merged dataset.

7 Difference Maps

The fact that 2D crystals show a higher degree of flexibility can be seen as a disadvantage or as an advantage. On the one hand it limits the resolution on the other hand it makes it easier/possible to investigate conformational changes and thereby provide insight into dynamics of the target protein. For this purpose, one can grow 2D crystals at different conditions, or induce conformational changes *in situ* by incubating the crystals at different conditions during sample preparation. Two different dataset are collected and merged separately. Please make sure that both maps show the same phase origin. Finally, by subtracting the phases and amplitudes of the final maps one can check for significant differences.

The difference map is calculated in `2dx_merge`. You find the script *Difference Map* in the Custom Script panel. You can open `2dx_merge` in an existing directory or create a new one. In the Parameter panel you have to specify the path for the two merged *2D.mtz* files (3), enter the right unit cell dimensions (1) and the resolution cutoff (2) you want to apply for the difference map (see Figure 50). Furthermore you can change the contour mode (4), contour range (5) and contour level (6) of you created maps. The script generates a map for both datasets and calculates the difference map with or without superimposed to the first map (7).



/Users/crpaulin/Documents/Workshops&Conferences/2dx_Workshop2012/DiffMap-Workshop/merge/

Lattice Determination
Real Unit Cell Length (for entire project): 1 — 80 | 108
Real Cell Angle (for entire project): 90.0

Common Image Processing
Lower Resolution Limit (RESMIN): 2 — 200.0
Upper Resolution Limit (RESMAX): 8

Plot generation
Contour and step size for the map generation: 6 — *0.6*
Contour threshold for the map generation: 5 — *0*

Difference Map
projection map 1: Path for Map1.mtz
name of map 1: 3 — Map1
projection map 2: Path for Map2.mtz
name of map 2: Map2.mtz
Contour mode: 4 — SIG
Mibimum contour level for the dfifference map generation: 5 — -3.0
Maximum contour level for the dfifference map generation: 3.0
Contour and step size for the dfifference map generation: 6 — 0.5

```
==== Limiting the resolution to 8 ==========================================
==== Inverse fft to get map ================================================
==== npo - to create a line plot for both maps =============================
==== laserplot - to create PS/4D1.ps ========================================
==== npo - to plot the overlay =============================================
==== laserplot - to create PS/4D1_4D2_overlay.ps ===========================
==== overlapmap -  to calulate difference map - 4D1 - 4D2 ==================
==== npo - to create a line plot for difference map ========================
==== laserplot - to create  PS/diff_map1_4D1-diff_map2_4D2_diffmap.ps ========
==== npo - to create a line plot for map 1 & difference map ================
```

MTZ: Map 1 with scaled amplitudes
MTZ: Map 2 with scaled amplitudes
MTZ: Map 1 (resolution limited)
MTZ: Map 2 (resolution limited)
PS: map 1
PS: map 2
PS: overlay of both maps 7
PS: Difference Map
Map 1 and Difference Map

Figure 50: GUI for the calculation of a difference map.

.

A way to define the contour level for a "significant difference" is to divide the images of one dataset into two and process them separately. A difference map between these two datasets should give you a range for the noise level. Hence, you could change the contour level to two levels by changing the *step size for map generation* and define this as your noise level. If you now use this step size for the difference map of interest you can assume that everything above two contour levels is a significant difference.

8 3D Merging

The general goal of electron transmission imaging techniques such as Electron Crystallography, Tomography or Single Particle reconstruction is to gain a 3D computer model by integrating sets of 2D data recorded by an electron microscope. The common mathematical tool that all these techniques share is the Central Section Theorem.

The theorem states that the projection along the z direction of a three dimensional volume contains the same information as the central slice in the reciprocal space ($z^* = 0$) of the 3D-Fourier transform of the volume. This means that we can gain a full sampling of the Fourier Transform of the object by the projections along all spatial directions. Accordingly, the 3D reconstruction become just a Inverse Fourier transform. The theory of Electron Optics allows to some approximation the use of images recorded by a Transfer Electron Microscope (TEM) as a projection of the sample. This means that with sufficient images of samples imaged in different orientation to the electron beam, one can simulate the sample's three dimensional density distribution.

We have seen in section Chapter 5 that the Fourier transform can be used to harvest the repeating signal of the 2D crystal. But since we are dealing with 2D crystals the periodicity is limited to the x,y-plane. The crystal is ideally only formed by one layer in the vertical direction. This results in continuously distributed Fourier component in vertical direction as illustrated in Figure 51. The diffraction spots we have seen in 2D with the index (h, k) now become so called *lattice lines* (h, k, z^*) where h and k still correspond to the same Miller index, but $z*$ defines the position along the lattice line.

Projections of tilted 2D crystals now produce diffraction spots, where each reflection corresponds to the lattice line h, k and height z^*. The height corresponds to the intersection of the tilted plane with the lattice line h, k and thanks to the central section theorem this can be calculated from the tilt geometry of the crystal. So each recorded image produces a set of amplitude and phases with the coordinates (h, k, z^*). The goal is now to densely sample the lattice lines by integrating as much data as possible with varying orientation and tilt angle. This allows us to sample the Fourier transform of our crystal to a certain degree, but due to instrumental restrictions of the TEM, where can not record an image of a any specimen tilted higher than $\pm 70°$ there will always be a missing cone.

8.1 3D Merigng in 2dx

3D merging in 2dx in follows the same workflow as for 2D merging. The problem is still merging data from different images into one common data set. Therefore finding the common phase origin is the task at hand. But in contrast to 2D data where each reflection h, k had a height $z^* = 0$, the amplitude and phases of a tilted sample are usually associated with a reflection at a height that is not zero. These measurements belong to a tilted plane in the 3D Fourier space, where the tilt geometry is defined by the tilt angle ($TANGL$) and the orientation of the lattice on that plane ($TAXA$). If we would now try to refine the phase origin of the tilted image data with non-tilted dataset, then the data points available for comparison would only be the one that lie on the tilt axis. Those measurements are to few to give a sufficient alignment of the tilted data. To deal with the alignment problem of the first tilted data we need to increase the range in $z*$ direction, so that a wider region for allowed

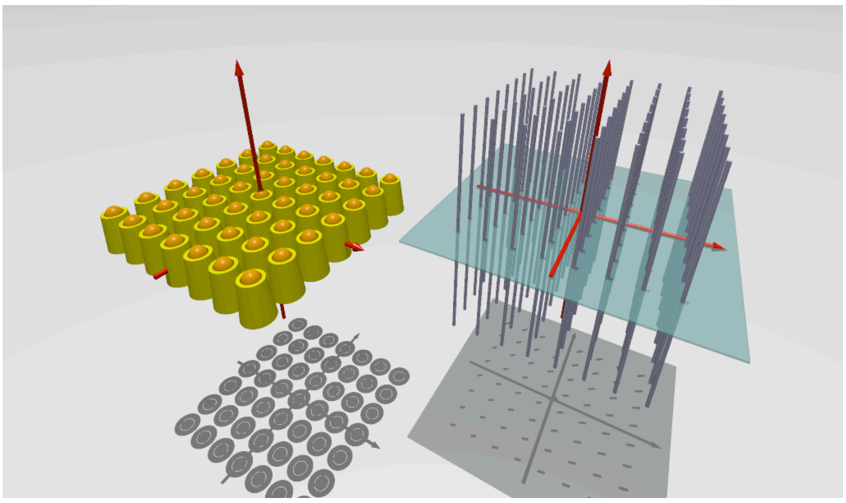


Figure 51: The 3D Fourier transform of a 2D crystal forms so called lattice lines perpendicular to the x,y-plane.

comparisons between the non-tilted and the reference data exists as illustrated in Figure 52.

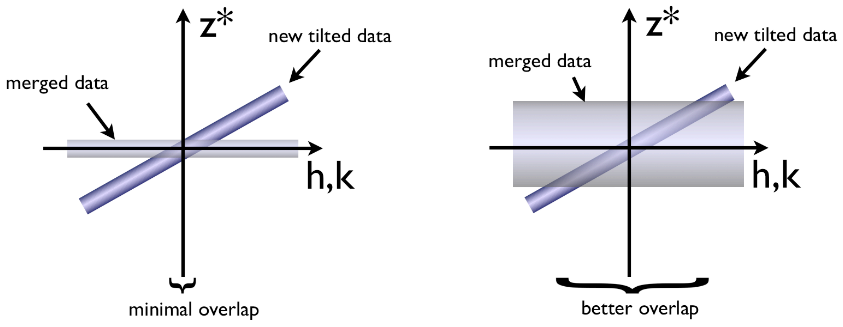


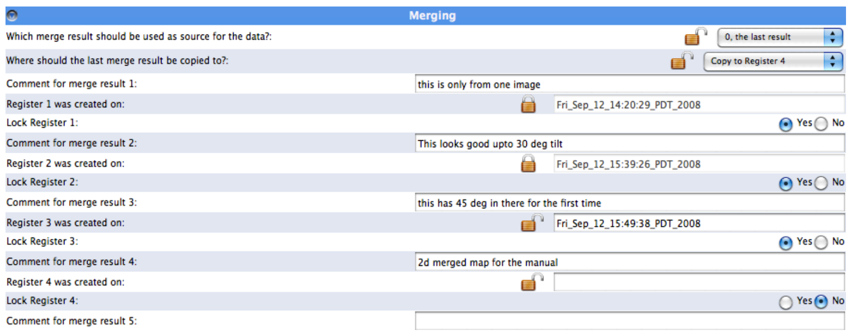
Figure 52: The alignment of data from tilted sample when the reference only consists of data from non-tiled sample is almost impossible, since the only data for comparison lie in the tilt axis (left). In this case the vertical tolerance range for comparisons has to be enlarged, which allows more comparisons (right).

In the following steps we will show how `2dx_merge` can merge data from 2D crystals with different tilt geometry to a 3D reconstruction of your protein structure.

1. We advise you to always backup your project before you start with 3D Merging, using the custom script *Synchronize with Backup*. This script allows you to submit your project data to a different directory. This is described in more detail in **??**.

2. Also save the merged map from 2D merging with help of the *Copy Merged Dataset* custom script. This script allows you to copy the last merging result to any of the given 10 registers as illustrated in

Figure 53. Add a comment to the register so you will recognize the merged result in the future.



Merging

Which merge result should be used as source for the data?:	0, the last result
Where should the last merge result be copied to?:	Copy to Register 4
Comment for merge result 1:	this is only from one image
Register 1 was created on:	Fri_Sep_12_14:20:29_PDT_2008
Lock Register 1:	[✓ Yes] No
Comment for merge result 2:	This looks good upto 30 deg tilt
Register 2 was created on:	Fri_Sep_12_15:39:26_PDT_2008
Lock Register 2:	[✓ Yes] No
Comment for merge result 3:	this has 45 deg in there for the first time
Register 3 was created on:	Fri_Sep_12_15:49:38_PDT_2008
Lock Register 3:	[✓ Yes] No
Comment for merge result 4:	2d merged map for the manual
Register 4 was created on:	
Lock Register 4:	Yes [✓ No]
Comment for merge result 5:	

Figure 53: Saves the merging result to the selected register. This register can then also be selected as a reference at a later point.

3. To let `2dx_merge` know that we are about to merge in three dimensions switch the *Modus of Merging* to *3D* in the Parameter panel.

4. Select all processed images that belong to the lowest tilt group ($TANGL \leq 30$), except the non-tilt data. Preferably, the selection should include ten or more projections with reliable measurements. As we have learned in Section 5.4.3, the projections of sample with low tilt is determined by defocus gradient which can be unreliable to some extent, therefore it is essential that the handedness is correct.

5. We have mentioned the critical step of aligning the first tilted dataset to a 2D merged map. So before running *Refine Once* limit the selected tilted data set to 15 Å with the parameter *RESMAX*. In this first 3D refinement step it is indispensable to have sufficient reference points (> 20) for the phase origin determination. This can be reached by increasing vertical z^* tolerance along a lattice line, which is defined by the processing parameter *3D: zstarwin* in the 3D Merging section. In the beginning of 3D merging set this parameter to 0.2 or something higher. This measure depends on the vertical thickness of your crystal defined by the parameter *ALAT*. When refining the highest tilt data *zstarwin* should be decreased to $\frac{1}{2*ALAT}$. So for a 200 Å sample we eventually set *zstarwin* to 0.0025, just to give you a feeling for the value range.

6. As in 2D merging we start out with a step size of 6.0 (*Stepsize of the phase origin search*)and 60 steps (*Number of steps in the phase origin search*).

7. Run *Refine Once* until the phase origins do not essentially change anymore.

8. Examine the refinement of the tilted data to the non-tilted by looking at the `origtilt` output *LOG: origtilt B output*. You should be familiar with this file from 2D merging, but in contrast to 2D you find that the refined reflection now have non-zero z^* values. Check the cross-correlation maps and if there is a distinct peak in the center. In that file you will also find a line like this "ORIGIN REFINEMENT DONE BETWEEN 48 OF THE NEW REFLEC-

TIONS", make sure the number of listed comparisons are above 20. Also have a look at the phase residual.

9. To visually inspect the refinement result run the *Generate Image Maps* script and scroll through the phase shifted projection maps.

10. Select the images for which the phase origin could be determined and the images which were used for the 2D reference. The selection can be saved via *Save Selection As...* in the *Select* menu. In this menu you can then later also load the stored selections as you can see in Figure 54.

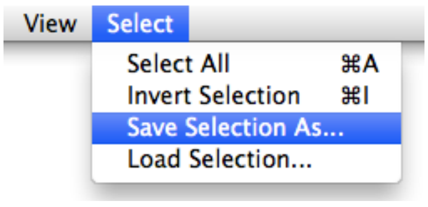


Figure 54: In the Select menu selection can be stored and loaded. This can be helpful during the *Merge Once* and *Refine Once* runs.

11. Merge the selected images all to one common 3D reference through *Merge Once*, but with a *Resolution of the merged dataset for the reference* set to 15 Å.

12. Now refine the alignment of the tilted images through the *Refine Once* script, but with a smaller *Stepsize of the phase origin search* while reducing *zstarwin*.

13. After the refinement have a look at the phase residual (*MergePhaseRes*) for every refined image. If the phase residual is high it could have its origin in the CTF correction with a false defocus. This can be detected by verifying the merged data set. Because the data from most images will be complete up to 15 Å one can profit from the redundancy in the number of reflections to identify errors in the CTF correction. The defocus in the images, where the errors were detected has to be adjusted before proceeding with the refinement.

14. If the cause of the bad phase residual was not the CTF correction, check if refining the tilt geometry can improve the alignment. This would be done in another *Merge&Refine (Iterative)* run while having the parameter *Refine Tilt Geometry (Only in 3D mode)* set to Yes. This should only be done if a thorough reference already exist. You will discover that if the reference is not sufficient most commonly the tilt refinement will cause the tilt angle of all the refined images to increase.

15. While running *Merge&Refine (Iterative)* with a reduced *zstarwin* e.g. 0.1 and a smaller step size (0.5) one should have a look in the Images panel, where several output files appear. Glancing through the list you will discover that we have now created lattice lines. The first lattice line file in the processing workflow is named *Latline after prescal*. It consists of the already known reflections with the index (h, k, z^*), but in contrast to the *merge.aph* the amplitudes are CTF corrected and the intensity values coming from the same micrograph

are scaled by the program `latlineprescal`. Of more interest is the file *Lattice Line fit data.* The difference of the reflections in this file compared to the previous one is apparent when looking at the z^* values. In contrast to the randomly spaced z^* values they are now equidistant. Because the program `latline` has fitted sinc functions to the measured data. This can be seen as an interpolation of the data along the lattice lines. The lattice line fitting can be tuned by the bin size defined by the processing parameter *3D: Bin-size for gathering data for the first lattice line guess (BINSIZ)*. You can verify the fitting outcome in the file *PS: Lattice lines* in the images panel. For both amplitude and phases you should have a smooth curve fitted through the data points given for every lattice line as in Figure 55.

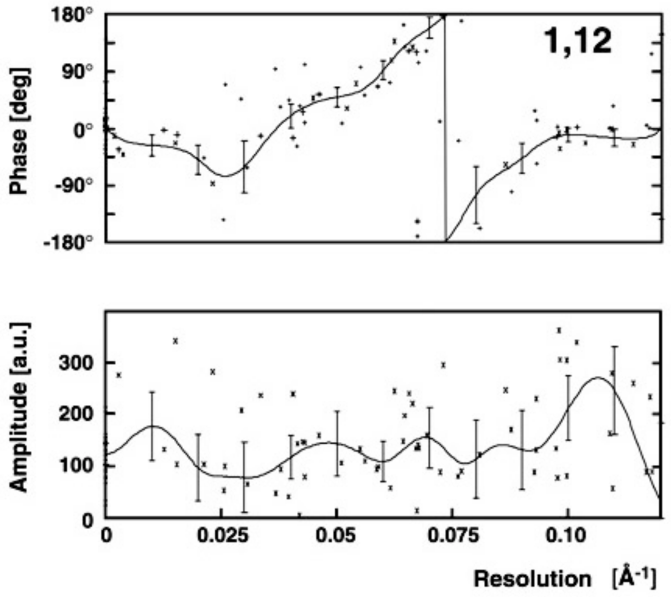


Figure 55: The lattice lines are created by fitting sinc functions to the measured data points. The fit can be inspected *PS: Lattice lines.* Here the lattice line (1,12) is shown.

16. After the first lattice line fit you should merge the refined data again but to a higher resolution with the *Merge Once* script. You could therefore increase the parameter *Merigning resolution limit* or you could also use the individual image resolution limits by changing the switch for *Merging Resolution limit.* Asses the phases statistics of the high resolution reflections.

17. If you find that your image data has a resolution of 5 Å or higher you should refine the beam tilt at this stage. This can be achieved by setting enabling the option *Refine Beam Tilt* in the *Merging Refinement* section for the refinement.

18. Now you can include images of higher tilt and align them to your 3D reference with the *Refine Once* script. The procedure of the refinement is done as before, so you should redo all the steps from item 7 to here. Remember that you can now decrease the parameter *zstarwin.*

19. The benefit of data stemming from sample tilted more than 25°is that the tilt geometry is more trustworthy, since it was determined by lattice distortions through `emtilt`. This means when you have merged a fair amount of higher tilt data you can in turn refine the tilt geometry of the lower tilt data.

20. Once all image data has been included and all refinements have been done you can decide on the final resolution cut-off for the individual images *(*Upper Resolution Limit) and run *Final Merge*.

21. An important file that appears in the Images panel during merging is *PS: TLTPLOT file*. This shows you the completeness of your 3D data set and allows you to determine at what tilt angle more data is needed for an even sampling of the 3D Fourier space.

22. The final step in the 3D reconstruction of your structure is the calculation of the 3D map. This can be achieved with the *Generate Merged Map* script. This creates a volume in CCP4 format named *MAP: Final 3D Volume*. You can examine the result in Chimera by double-clicking it. With the example data set of GlpF the 3D map reconstruction should look similar to Figure 56

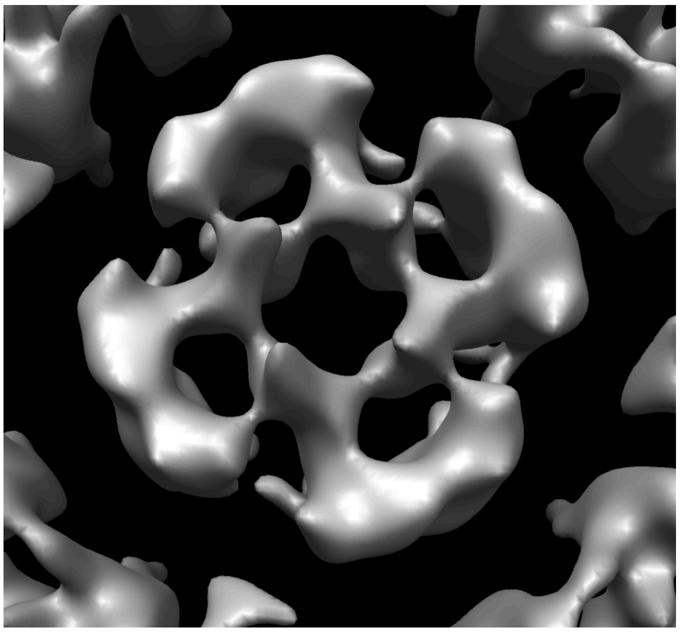


Figure 56: The 3D volume of the GlpF example data set.

23. Once a reliable 3D-model has been generated on can re-process the already used images with the *(Re-)Process all images* script. Thereby one can improve the data from the individual image data by unbending with a reference created through back projection from the model. We call this procedure synthetic unbending and it is described in Section 6.2.

9 Processing an image by 2D Maximum Likelihood

Maximum likelihood (ML) processing is available in `2dx_image` for two-dimensional alignment. This is a refinement process which should be performed after the unbending process. The corresponding script is called *ML-2D (for advanced users)* and can be found in the Standard Script panel. The density map shown in Figure 57 shows the result obtained without maximum likelihood refinement.

For detailed information we refer to [Zeng *et al.* 2007]. Note that using this module is not trivial. Here we are just focusing on the GlpF test data set. In order to conduct the ML processing, you first need to set up related parameters. You can right click on a parameter to view the description. Some tips are provided to help you decide the parameter values.

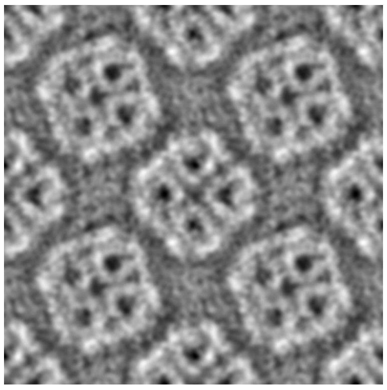


Figure 57: Map before ML

1. Set *Process Single Particle Reconstruction* to *Yes* in the *Maximum Likelihood Algorithm* section in the Parameter panel in order to activate the maximum likelihood refinement in `2dx`. If you run the script without this option nothing will happen.
2. Select *Maximum Likelihood Calculation* for the *Weighting profile from* in the same section of the Parameter panel.
3. Set the *Total Diameter of the windows* to a size that is larger than the unit cell. You find this option in the *Maximum Likelihood Parameters* section of the Parameter panel. For the test dataset we recommend around 1.2 times of the unit cell size, i.e. $120, 120$.
4. Set the *Diameter of the circular mask* to a size that is slightly larger than the unit cell. In our case we used 110.
5. In most cases, select *relative percentage given* for *Threshold determination method for particle selection*, and set the relative percentage to your needs. A value of 50 means the top 50% particles will be used in the ML processing. The particles are extracted from the cross-correlation profile generated during the *Unbend I* script.
6. Select *Gaussian* as *Type of low-pass filter*.
7. Set *Low-pass filter radius* to a value between 0.25 and 0.5. In our test case we use 0.25.

8. Set *Apply noise whitening* to *Yes* in order to enable noise whitening which improves the convergence of the maximum likelihood method.

9. Set *Apply CTF correction* to *Yes* to correct for the contrast transfer function for each particle.

10. Set up *Angular range* and step for angle search. For well ordered crystals, you can use a smaller range. If you don't want to do angle search, use 0 for range. The step size must always be positive (nonzero). For the test dataset we used −10 to 10 degrees with a step size of 1.0.

11. Set *Terminate after maximum number of iterations* to a value usually between 20 and 40.

12. Set *Termination if change in dev_sigma or dev_sigma_theta below* to a small positive number less than 0.1.

13. **Before running the script click on save in the header panel, otherwise your parameter will not be applied**. After setting up and saving the ML related parameters, you click on *ML-2D* in the Standard Scripts panel to launch the process. You can view the results by clicking on reference maps listed in the Images panel. Figure 58 shows our 2D maximum likelihood results.

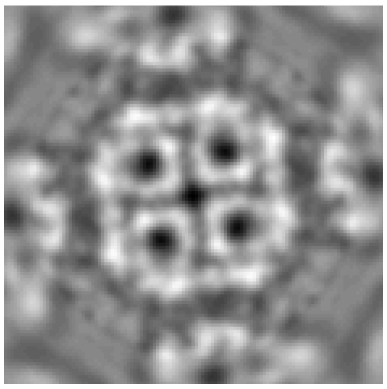


Figure 58: Map after ML

10 Automatic Image Processing

In this section we provide detailed hands-on instructions on automatic image processing within 2dx [Scherer *et al.* 2014], including on-the-fly image drift-correction. The here described pipeline is optimised for data recorded on a direct electron detector, such as the Gatan K2 summit.

10.1 Real-time motion-correction

In case you record dose fractioned movies, we propose to drift-correct them on-the-fly as described in:

https://github.com/C-CINA/2dx/wiki/Automatic-Drift-Correction-C-CINA-setup

Additionally instruction about setting up our automatic drift-correction pipeline is available under:

https://github.com/C-CINA/2dx/wiki/Automated-Drift-Correction-GUI

2dx_automator will later be used to automatically process the drift-corrected average of all frames.

10.2 Required software

2dx_automator is only supported under Linux right now. Please make sure that the list of dependencies is installed on the used system. The exact package names vary between different linux flavours.

- `TkInter` a python GUI package
- `py-thread` to be able to dispatch processing into different tasks
- `py-imaging (aka PIL)` the python image processing libraries
- `eman2/sparx` has to be installed and sourced on the system before launching 2dx_automator.
- `py-matplotlib` to generate plots
- `py-shutil` for low-level operating system interactions
- `ImageMagic` especially the command-line tool `convert` has to be available

Additionally 2dx_automator requires that `2dx_image, 2dx_merge` are available in the command line. If you installed a pre-compiled package please add 2dx's binary directory to your path:

```
export PATH = /opt/2dx/bin:$PATH
```

If you compiled the software from source directly on your computer, you should source the installation-binary folder as well:

```
export PATH = <your_install_dir>/bin:$PATH
```

where `<your_install_dir>` equals the second argument passed to the `build_all`-script. If you did not pass any argument to the build-script then `<your_install_dir>` equals `~/2dx/`.

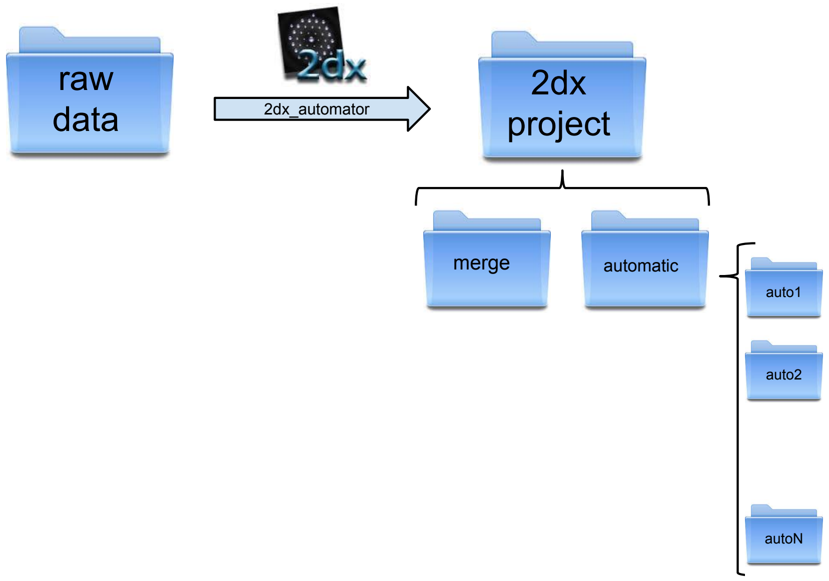


Figure 59: 2dx_automator: Folder structure

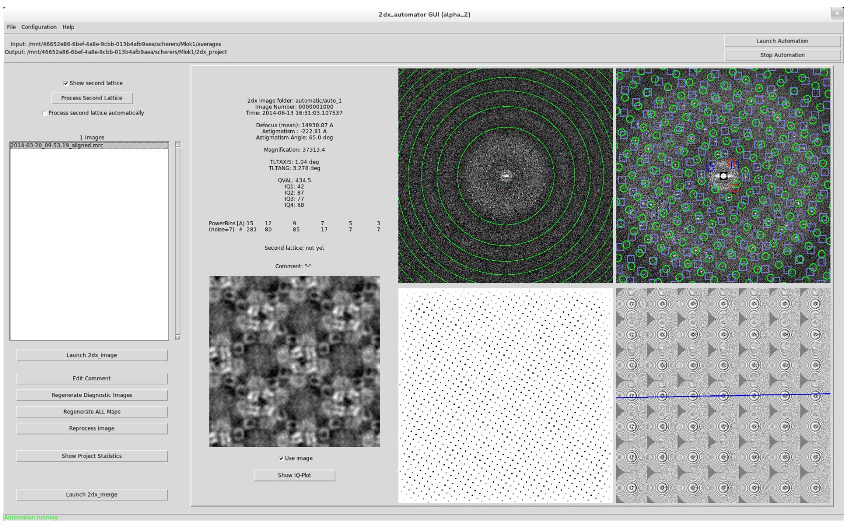
10.3 Automation setup

The basic folder organisation is shown in Figure 59: The raw images are stored in the *raw data folder*. Note that we here only process averages gained from drift-corrected movie-frames. 2dx_automator will check for newly added images in this folder periodically (every second). In case a new image is added, this image is processed automatically with 2dx_image: Defocus determination, lattice estimation, unbending, automatic masking, CTF-correction and map generation. Thereby the image is imported into the currently selected 2dx-project. This project contains a merge directory and a folder storing image-directories for each automatically processed image.

Before launching the automation for the first time, create an empty folder, which will later contain your automatically populated 2dx-project. Then launch 2dx_merge and select this newly generated folder as your project directory. 2dx_merge asks whether the project structure should be generated, which should be done of course. Note that this step is only required when generating a new project. It is very important that the project structure is generated by 2dx_merge before you launch the automation.

Starting the automation software is done by executing the command `2dx_automator` in the shell. Given you installed all dependencies properly, 2dx_automator will ask for two different folders. The first one you have to select is the raw images folder. Please note that you have to navigate into the folder before pressing *"Ok"*. Just selecting the desired folder does not work. Secondly you are asked to select the 2dx-project. Please select the root of the project, i.e. the *2dx project* in Figure 59.

2dx_automator requires a config-file that stores the processing parameters used for image processing of all image. The automation software uses the file *2dx_merge.cfg* in the merge directory of the project folder (more precisely: the file linked in *2dx_master.cfg* from the root level is



```
 Input: /mnt/46652e86-6bef-4a8e-9cbb-013b4afb9aea/scherers/Mlok1/averages
Output: /mnt/46652e86-6bef-4a8e-9cbb-013b4afb9aea/scherers/Mlok1/2dx_project

[✓] Show second lattice
[ ] Process second lattice automatically

1 Images
2014-03-20_09.53.19_aligned.mrc

2dx image folder: automatic/auto_1
Image Number: 0000001000
Time: 2014-06-13 16:31:03.107537

Defocus (mean): 14930.87 A
Astigmatism : -222.81 A
Astigmatism Angle: 65.0 deg

Magnification: 37313.4

TLTAXIS: 1.04 deg
TLTANG: 3.278 deg

QVAL: 434.5
IQ1: 42
IQ2: 87
IQ3: 77
IQ4: 68

PowerBins [A] 15     12      9       7       5       3
(noise=7)  #  281    80      85      17      7       7

Second lattice: not yet

Comment: "-"

[✓] Use image

Automation running
```

Figure 60: 2dx_automator graphical user interface

used). We suggest that you determine the optimal processing parameters in 2dx_merge by manually importing and processing one or a couple of images before launching the automatic processing. Once you found the optimal processing parameters by manually processing an image in 2dx_image you can set the tuned set of parameters as default via *"File > Save as Project Default"*. Alternatively - in case you already have a valid file from a previous similar project - you can load a config file in 2dx_automator via *"Configuration > Load config from file"* directly in 2dx_automator.

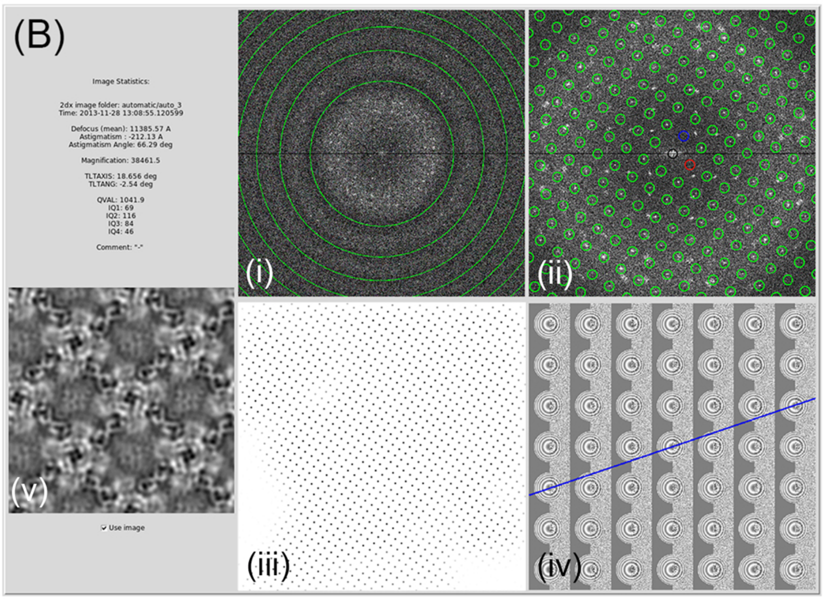
Below a couple of points to consider while tuning the image processing parameters:

- Always determine the tilt-geometry (even for nominally untilted images). As 2dx_automator does not *a priori* know whether a new image is tilted or not, the processing pipeline requires these computing results.
- Don't use a too optimistic *RESMAX* as lattice determination does get unreliable.

Once you loaded or generated an optimal config-file, the automatic processing is finally launched by clicking on the *Launch automation* button (top-right). Note that in case there are so far unprocessed images in the input folder, these images will be processed first.

10.4 2dx_automator

When opening the 2dx_automator for the first time, all fields are empty. These white squares will be populated once the images are processed, see Figure 60. The top row of the GUI states in which folders the automation is running and there you also can launch and stop the automatic processing, i.e. periodically checking for new images. The left panel features some processing options (detailed in Section 10.5) and a list providing an overview of all images in the project. The right panel (detailed in Figure 61) provides all required intermediate processing results, which allow



```
Image Statistics:

2dx image folder: automatic/auto_3
Time: 2013-11-28 13:08:55.120599

Defocus (mean): 11385.57 A
Astigmatism : -212.13 A
Astigmatism Angle: 66.29 deg

Magnification: 38461.5

TLTAXIS: 18.656 deg
TLTANG: -2.54 deg

QVAL: 1041.9
IQ1: 69
IQ2: 116
IQ3: 84
IQ4: 46

Comment: "-"
```

Figure 61: Diagnostic view of the 2dx_automator GUI. (i) Fourier transform with Thon rings and (ii) fitted lattice, (iii) peak profile from the unbending step used to mask the image, (iv) locally estimated defocus values, (v) final 2D projection map.

to judge the reliability of the image processing of the selected image.

In case you observe that image processing failed for one particular image, you can open 2dx_image for the corresponding image by clicking on *"Launch 2dx_image"*. Provided you want to apply the tuned processing parameter for all future image, just save the new config parameter set as project-wide default. Once you close 2dx_image (after fixing the processing) you have to regenerate the diagnostic images for this particular image by clicking on *"Regenerate Diagnostic Images"*. Note that this can take up to 30 sec.

10.5 Advanced tricks

Here we discuss some advanced automation tricks, accessible via the available buttons.

Second lattice processing : 2dx_image by default tries to find two crystal lattices, given there are to (usually originating from tubular crystals). The top-left region of the central widget contains the related functions. There you can select whether the second lattice should be displayed or not. In case the second lattice was found correctly, it might be worth to processes this lattice automatically as well. Clicking on *"Process Second Lattice"* clones the image directory, swaps the lattices and processes the cloned image automatically. Note that the result will not be shown in the GUI, but will be directly in the 2dx project. For instance processing the second lattice of image *"auto_12"* will generate a new image called *"auto_12_c1"*. Enabling automatic processing of all found second lattices removes the manual selection step mentioned above. This option should be used carefully

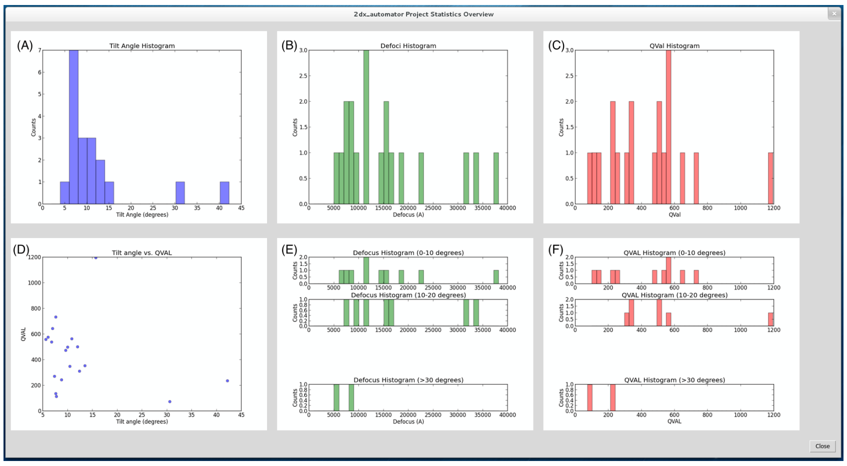


Figure 62: Project stat

and only for high-quality datasets with clear second lattices.

Comment editing : 2dx_automator provides a instant comment editing tool. You can add or edit the comment field via *"Edit Comment"*. The updated comment is synchronised with the config file of the selected image.

Updating image maps : For instance after 3D merging it might be a good idea to regenerate the projection maps given their updated phase origins. You can regenerate all projection maps of the project by means of *"Regenerate ALL Maps"*.

Reprocess an image : *"Reprocess Image"* deletes all processing results for the selected image, copies the new config file and reprocesses the image. This function is particular useful to apply an updated config file to an image.

Project statistics : The judge the completeness and diversity of the recorded data, clicking on *"Show Project Staticstics"* produces a project statics overview (example show in Figure 62). Missing tilt angles are defocus values can be found efficiently by analysing the produces charts.

"Use images" : Only selected images are used to generate the projection statistics overview plots.

IQ-Stat analysis : To judge the quality and distribution of the diffraction spots *"Show IQ-Plot"* opens the *PLOTRES*-file of the selected image.

Configuration file handling : Under *Configuration > Show config file* and *Configuration > Show processing parameters* you can view the full underlying config file, respectively only the extracted processing parameters.

"Launch 2dx_merge" : To merge the entire project in order to get a reconstruction you have to open 2dx_merge. Clicking on the corresponding button opens 2dx_merge, but still asks for the desired project directory.

10.6 Credit

Please don't forget to cite [Scherer *et al.* 2014] if you obtained your results by means of 2dx_automator.

References

[Arheit *et al.* 2012a] M Arheit, D Castano-Diez, R Thierry, P Abeyrathne, B R Gipson and H Stahlberg. *Merging of Image Data in Electron Crystallography.* Electron Crystallography of Soluble and Membrane Proteins, Methods in Molecular Biology, Vol. 955, Chapter 10, 2012. (Cited on page 5.)

[Arheit *et al.* 2012b] M Arheit, D Castano-Diez, R Thierry, B R Gipson, X Zeng and H Stahlberg. *Automation of Image Processing in Electron Crystallography.* Electron Crystallography of Soluble and Membrane Proteins, Methods in Molecular Biology, Vol. 955, Chapter 10, 2012. (Cited on page 5.)

[Arheit *et al.* 2012c] M Arheit, D Castano-Diez, R Thierry, B R Gipson, X Zeng and H Stahlberg. *Image Processing for 2D Crystal Images.* Electron Crystallography of Soluble and Membrane Proteins, Methods in Molecular Biology, Vol. 955, Chapter 10, 2012. (Cited on page 5.)

[Gipson *et al.* 2006] B Gipson, X Zeng, Z Y Zhang and H Stahlberg. *2dx - User-friendly image processing for 2D crystals.* Journal of Structural Biology 157 64-72, 2006. (Cited on page 5.)

[Gipson *et al.* 2007] B Gipson, X Zeng and H Stahlberg. *2dx_merge: Data management and merging for 2D crystal images.* Journal of Structural Biology 160 375-384, 2007. (Cited on page 5.)

[Scherer *et al.* 2014] Sebastian Scherer, Julia Kowal, Mohamed Chami, Venkata Dandey, Marcel Arheit, Philippe Ringler and Henning Stahlberg. *2dx_automator: Implementation of a semiautomatic high-throughput high-resolution cryo-electron crystallography pipeline.* Journal of structural biology, vol. 186, no. 2, pages 302–307, 2014. (Cited on pages 63 and 67.)

[Zeng *et al.* 2007] X Zeng, H Stahlberg and N Grigorieff. *A maximum likelihood approach to two-dimensional crystals.* Journal of Structural Biology 160 362-374, 2007. (Cited on page 61.)

List of Figures

Index