

SPH



RE

SParx for **H**igh **R**esolution **E**lectron Microscopy

structure determination with SPHIRE

a practical guide

Info?
Turn here!

December 2016

We completely reworked **SPARX** (Hohn et al. 2007), but maintained its appreciated core features. The program was streamlined, made easy to use and is now ideally suited for solving high-resolution **cryo-EM** structures. Importantly, the program did not turn into a black box and experienced users still have the possibility to perform their individual fine-tuning. This should be especially useful for non-standard projects.

**To mark the new era of our program:
SPARX became **SPHIRE** (SParx for HIgh-REsolution electron microscopy)**

This guide contains a description of a general workflow of a **cryo-EM** project. It is based on the **GUI** of **SPHIRE**, and demonstrates how to obtain high-resolution 3D maps of macromolecular complexes from **cryo-EM** images. It will direct you through all steps of **SPA** based on a provided experimental data set. After completing the tutorial you should be able to independently process your own data. In addition, almost every step contains a “**TIP section**” section, with more advanced instructions that might help you to obtain optimal results from your own data. In case an alternative processing workflow is required. Optional steps, performed outside the **GUI**, as well as known issues of the alpha-release are described within info-boxes.

A detailed **SPHIRE** documentation is available at:

<http://sphire.mpg.de>

There is a mailing list for user support, discussions and future announcements.

You can subscribe at:

<https://listserv.gwdg.de/mailman/listinfo/sphire>
and search the archives for your topic of interest.

Once you have subscribed, you can also use the address:

sphire@lists.mpg.de
to send e-mails to the **SPHIRE** team.

Contents

1	Software Installation	1
1.1	Install SPHIRE and MPI support	1
1.2	Install Other EM Software Packages	1
2	Project	2
3	Movie	6
3.1	Micrograph Movie Alignment	6
3.2	Drift Assessment	10
4	CTER	13
4.1	CTF Estimation	13
4.2	CTF Assessment	16
5	Window	22
5.1	Particle Picking	22
5.2	Particle Extraction	25
5.3	Particle Stack	28
6	ISAC	30
6.1	Pre-cleaning	31
6.2	ISAC 2D Clustering	37
7	VIPER	43
7.1	Initial 3D Model - RVIPER	43
7.2	Resample/Clip VIPER Model	48
8	MERIDIEN	50
8.1	Adaptive 3D Mask	51
8.2	3D Refinement	53
8.3	Sharpening	58
9	SORT3D	62
9.1	3D Variability	62
9.2	3D Clustering - RSORT3D	67
9.3	Local Subset Refinement and Final Reconstruction	73
10	LocalRes	78
10.1	Local Resolution	78
10.2	Local Filtering	82

11 Acronyms	86
12 Citing SPHIRE	87
Bibliography	88

Software Installation

Install SPHIRE and MPI support

To follow this guide and run **SPHIRE** you will need to properly install **SPHIRE** on a system with an **MPI** installation (Linux computing cluster or Linux/Mac multi-core desktop). Download **SPHIRE** for free from http://sphire.mpg.de/downloads/sphire_beta_20161216.tar.gz and follow the instructions regarding **SPHIRE** and **MPI** installation which can be found here: <http://sphire.mpg.de/wiki/doku.php?id=howto:tutorials>. It should be noted that **SPHIRE** and **EMAN2** (Tang et al. 2007) are part of the same installation package, i.e. after installation of **SPHIRE**, **EMAN2** is also installed and *vice versa*. Note that **SPHIRE** requires **MPI** installation to use most advanced commands, while **EMAN2** does not. Both **SPHIRE** and **EMAN2** are issued under a joint BSD/GNU license (see the documentation) and are provided free of charge as a service to the scientific community.

To make sure the installation finished successfully, open a terminal, type *sphire* and check if the **SPHIRE GUI** pops up. To make sure the **MPI** installation finished successfully, type *sx.py* at the terminal window and then type *import mpi*. No import error messages should appear. Exit the interactive mode by typing *quit*.

Install Other EM Software Packages

Alignment of direct detector movie frames is currently not included in **SPHIRE**. However, within the framework of the **GUI** we provide a wrapper to **Unblur** and **Summovie** (Grant et al. 2015). Please download the programs **Unblur** and **Summovie** (<http://grigoriefflab.janelia.org/unblur>, Grigorieff lab) and follow the installation instructions.

For interactive visualization of the resulting structures, we will use the molecular graphics program **UCSF Chimera** (Pettersen et al. 2004b) (<https://www.cgl.ucsf.edu/chimera/download.html>). A nice tutorial to get familiar with the features of **UCSF Chimera** we use throughout this guide can be found here: <https://www.cgl.ucsf.edu/chimera/data/tutorials/groel/groel.html>



Project

In this tutorial, we use the **SPHIRE GUI** to obtain a 3D reconstruction of the bacterial toxin component TcdA1 (Gatsogiannis et al. 2013) from a test dataset of 112 micrographs. The images were collected on a **Cs** corrected FEI Titan Krios, operated at 300 kV and equipped with a **XFEG** and a Falcon II direct detector. The digital micrographs have a pixel size of 1.14 Å/pixel and the particle is a pentameric assembly with C5 point group symmetry. The data can be downloaded here: <http://sphire.mpg.de/wiki/doku.php?id=howto:tutorials>. In order to extract the downloaded file, type:

```
tar -zxvf sphire_testdata_tcd1_20161216_movies.tar.gz
```

The **project directory SPHIRE-demo** will be created. As it is necessary to always start the **SPHIRE GUI** from the **project directory** change to this directory.

```
cd SPHIRE-demo
```

To launch **SPHIRE GUI**, type:

```
sphire &
```

When **SPHIRE** is started for the first time, the main window will pop up.

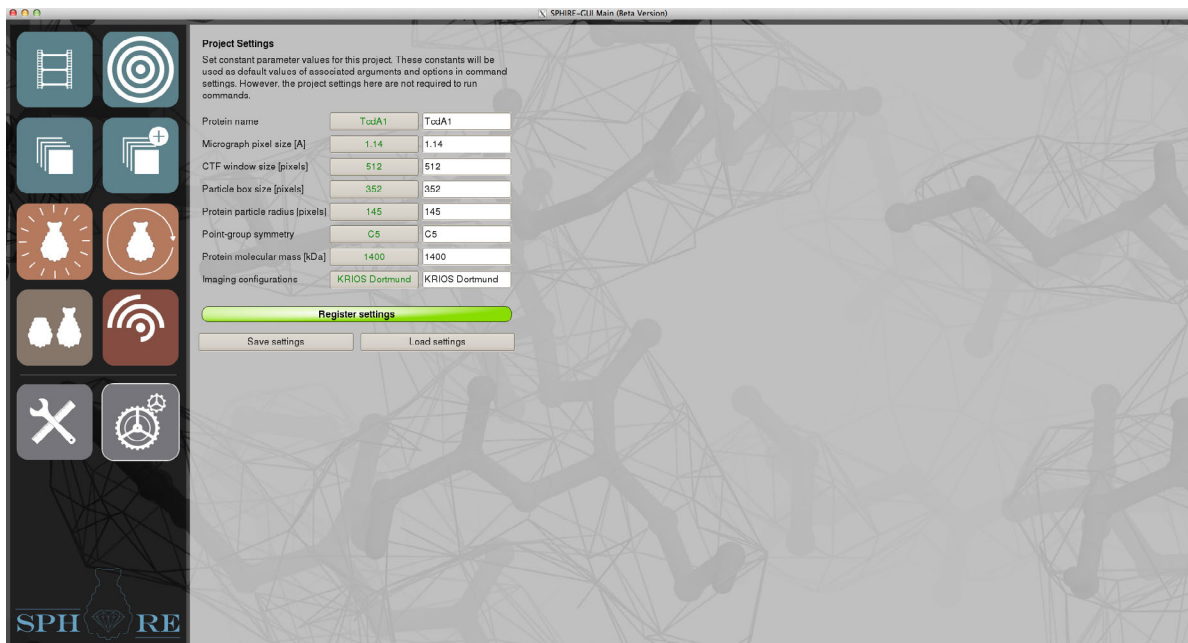


The column on the left contains of several pictograms that allow you to select a certain step in the single particle image analysis workflow. The first step is to provide the project-wide constant-value parameters. These constants will be used as default values in the settings of all subsequent steps of the processing workflow.



TIP: You can also skip registering these settings now, but in this case you might have to specify these standard parameters multiple times during the processing procedure.

Press the **Project** pictogram and provide the following parameters at the respective GUI interface:



NOTE: If you **click** on the gray button, next to the respective input field, a default value can be retrieved. After you registered the settings here, they will become default values in the subsequent steps.

- **Project name:** TcdA1
- **Micrograph pixel size [Å]:** 1.14

TIP: Do not forget to adjust the pixel size, in case you binned your micrographs. For example, micrographs recorded with the K2 at super-resolution mode have a very small pixel size and usually we bin them before we start with the image processing in **SPHIRE**.

- **CTF window size [pixels]:** 512

NOTE: Should be slightly larger than particle size

- **Particle box size [pixels]:** 352

NOTE: The particle box size should be at least $1.5 \times$ the length of the longest axis of the particle. The tutorial particle has a diameter of 25 nm (250 Å) and the pixel size is 1.14 Å/pixel. Thus, the longest axis of the particle in pixels is $250 \text{ Å} / 1.14 \text{ Å/pixel} = 220 \text{ pixel}$. Here we set the box size to 352 pixel, which corresponds to $1.6 \times$ the size of our particle.



NOTE: *The window size has to be an even number and various algorithms depend non-linearly on the window size.*

TIP: *In case you recorded images at rather high defocus values (i.e. $> 3 \mu\text{m}$), you might consider using an even larger box size.*

- **Protein particle radius [pixels]: 145**

NOTE: *Adjust this parameter carefully, so the radius set here will correspond at least to the half of the longest axis of the particle*

TIP: *This value will be used to create a circular mask with the respective radius, to mask the particles in 2D. Therefore, at the beginning of the project and especially if the center of the particle does not correspond to the center of mass, it might be preferable to use a slightly larger particle radius. Only use a rather tight radius if the particles are reasonably centered.*

- **Point-group symmetry: C5**

NOTE: *If the point-group symmetry of the particle is uncertain, do not assume symmetry.*

- **Protein molecular mass [kDa]: 1400**

NOTE: *It can be approximate. The molecular mass of the protein is used for validation purposes in some steps of data processing.*

- **Imaging configurations: KRIOS Dortmund**

NOTE: *Your microscope name/configuration. It is optional and is kept for the record only.*

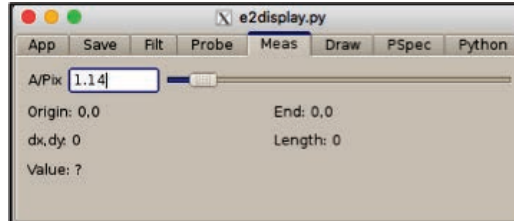
Press the **Register settings** button, to register these values as defaults for all subsequent steps of the workflow, and then press the **Save settings** button, to save these settings in a text file. This text file might be useful for your lab book or can be used as an input to reload these values if necessary. A detailed description regarding the options is provided on our [wiki](#) page. Even if you do not save these settings now, they will be automatically displayed every time you start the **GUI** from the **project directory**.



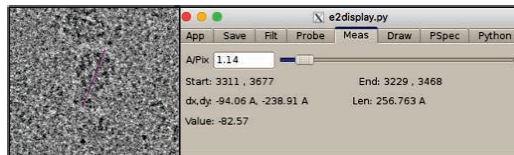
Measuring the size of your particle

Before collecting high-resolution movies, you most probably screened the quality of your sample by negative stain **EM** and also collected some **cryo-EM** overview images. You can use one of these images to measure the size of your particle using **e2display.py**.

1. At a terminal type: **e2display.py &**.
2. Open an overview image with sufficient contrast.
3. Press the **mouse wheel button** somewhere on the image. A pop up window will appear.
4. Activate the **Meas** tab in the pop up window, set the pixel size correctly (A/Pix) and hit the **RETURN** key.



5. Now identify the projection view of a particle with possible largest size. Keep the **left mouse button** pressed and draw a line to measure the longest axis of the particle. The length of the line in Å will be reported e2display pop up window (**Len**).





Movie

Micrograph Movie Alignment

Modern direct detectors record high-resolution images as movie frames. The first step in the workflow is the alignment of these frames for each image to correct the overall motion.

***TIP:** Not all microscopes are equipped with cameras with movie capabilities and generally only high-resolution projects require recording movie frames. **SPHIRE** is very flexible in this context and allow you to skip certain steps of the workflow as long as appropriate settings are entered. This is critical when importing data created with different software packages. Please read the TIP section for each step of the workflow carefully and follow the instructions that best fit your processing scenario.*

SPHIRE provides a wrapper to the program **Unblur**, developed by the [Grigorieff lab](#). During unpacking of the tutorial data, a folder called **Movies** was created within the **project directory**.

To see the content of this folder, type:

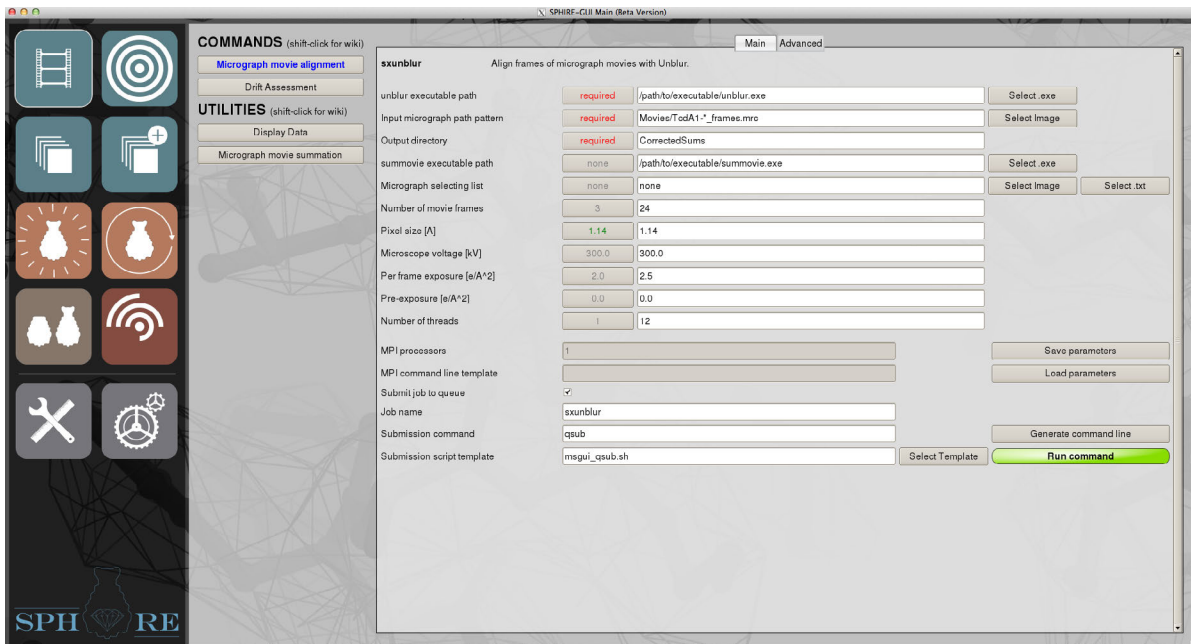
```
ls Movies/*.mrc
```

You will see that the folder contains 112 low-dose movies of TcdA1 in the standard file format *mrc*.



Movie

Go back to the main window of the **SPHIRE GUI** and press the button **Movie** on the left and then the **Micrograph movie alignment** button in the middle. An activated command button contains blue highlighted text.



Set the following parameters in the respective input fields:

- **Unblur executable path:** /path/to/executable/unblur.exe

*NOTE: Type here the path to the **Unblur** executable file or click the **Select .exe** button to use a file browser to select it.*

- **Input micrograph path pattern:** Movies/TcdA1-*_frames.mrc

*NOTE: Press the **Select Image** button and use the file browser to select a movie file. Then replace the variable part of the file name with the wildcard character “*”. In case of this tutorial dataset, the variable part of the file name is only the serial number (**TcdA1-0001_frames.mrc**).*

TIP: It is preferable to use serial numbers in file names that have the same number of digits (e.g. use mic_0001.mrc, mic_0010.mrc instead of mic_1.mrc, mic_10.mrc).

- **Output directory:** CorrectedSums
- **Summovie executable path:** /path/to/executable/summovie.exe

*NOTE: Type here the path to the **Summovie** executable file or click the **Select .exe** button to use a file browser to select it.*

- **Micrograph selecting list:** none



NOTE: Press the **Select .txt** button and use the file browser to load the list of selected micrographs of a later step.

- **Number of movie frames:** 24

NOTE: The number of frames in each movie file. Our tutorial data contain 24-frame movie files.

IMPORTANT: All movie files in a project must contain the same number of frames and the same pixel size!

- **Pixel size [Å]:** 1.14
- **Microscope voltage [kV]:** 300
- **Per frame Exposure [e/Å²]:** 2.5

NOTE: The 24-frame movies of this dataset have an overall dose of 60 e/Å². Thus, the exposure for each frame in this case is

$$\frac{\text{Total dose}}{\text{Number of movie frames}} = \frac{60 \text{ e}/\text{Å}^2}{24} = 2.5 \text{ e}/\text{Å}^2 .$$

- **Pre-exposure [e/Å²]:** 0
- **Number of threads:** 12

TIP: Expert users can also press the **Advanced** button to display the advanced settings. However, the default values rarely need to be changed for standard projects, therefore these parameters will not be described in this tutorial.

Regrettably, it is not possible yet to run the program using multiple **MPI** processes. On our Linux cluster, with a single process, the alignment of the 112 movies finished after about 90 min. We will inform you via the mailing list, as soon as we finish the parallel version of our **Unblur** wrapper.

If you have enough time during this tutorial, submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. A **SGE** template file can be found in your **project directory** (**msgui_qsub.sh**). You might need to configure or prepare a different template file for your own system. The name of the logfile containing the standard output depends on your submission file. If you are not familiar with your cluster and its queuing system, you might need help from your sysadmin in order to prepare the correct file. Details about setting up the template file can be found on the **SPHIRE** [wiki](#).

To keep track of the progress of the job, type at the terminal:

```
tail -f name_of_sxunblur_logfile
```

```
Progress: 4.46%; Time: 0h:4m:4s/1h:31m:12s; Micrograph done:TcdA1-sialic-104_frames
```




Otherwise, if you need to save time, you can copy precalculated results to your **project directory** and continue with those.

In this case, type:

```
cp -r SphireDemoResults/CorrectedSums ./
```

To see the content of the **CorrectedSums** output folder, type:

```
ls CorrectedSums
```

The output folder contains following subfolders:

- **corrsum**: Contains the motion-corrected, dose-weighted averages for all movie files.
- **corrsum_dose_filtered**: Contains the motion-corrected *but* dose-*un*weighted averages
- **logfiles**: Contains the outputs of **Unblur** and **Summovie**
- **shift**: Contains text-files with the calculated x,y-shifts per frame by the **Unblur** frame alignment procedure
- **frc**: Contains text-files with the frc information of **Summovie**
- **unblur_micrographs.txt**: List with the file names of the motion corrected averages
- **unblur_shiftfiles.txt**: List containing the unblur shift file names

The dose-unweighted motion-corrected averages are necessary only for **CTF** estimation, whereas the dose-weighted averages will be used for all other steps of the workflow.

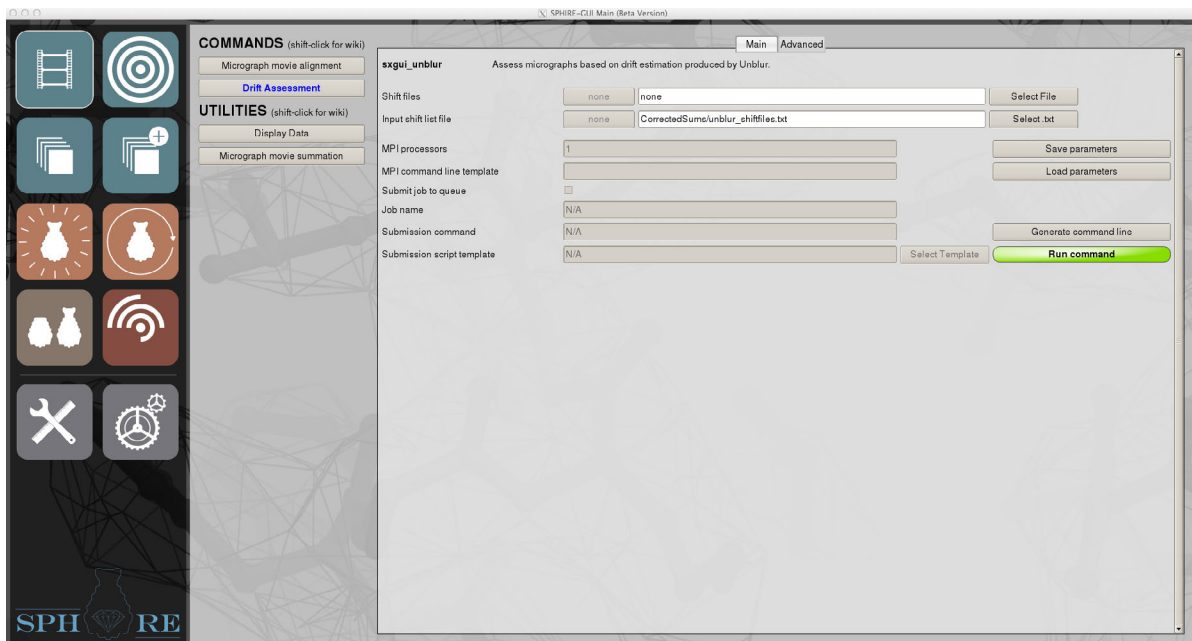
TIP: *If you have your movies already aligned using a different software package, you can directly copy the averages of each movie file to this output folder and skip the next step.*



Drift Assessment

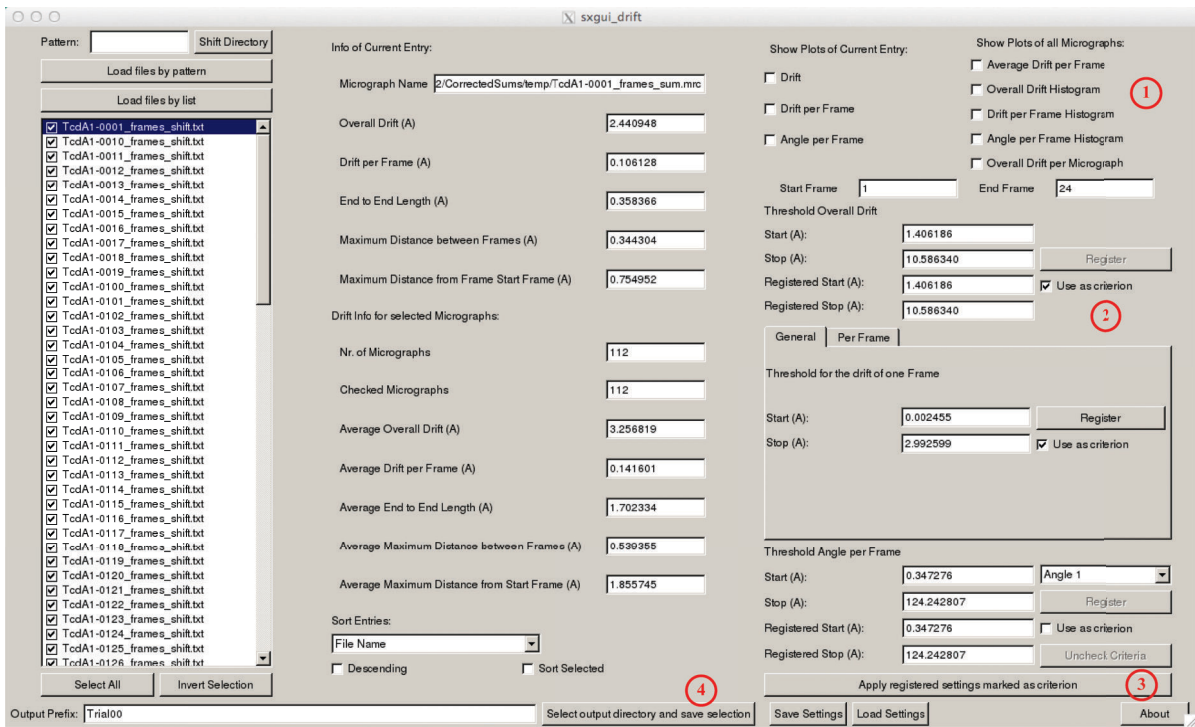
Certain average images still have a significant amount of drift, even after frame-alignment (due to charging, holder instability or local grid damage) and thereby high-resolution information is lost. The next step is to identify and exclude these images from the further steps of the single particle image processing. For this purpose we will analyze the results of Unblur using our **Drift Assessment GUI** tool to select images with the lowest amount of drift.

Go to the main window of the **SPHIRE GUI** and press the button **Movie** on the left and then the button **Drift Assessment** in the middle.



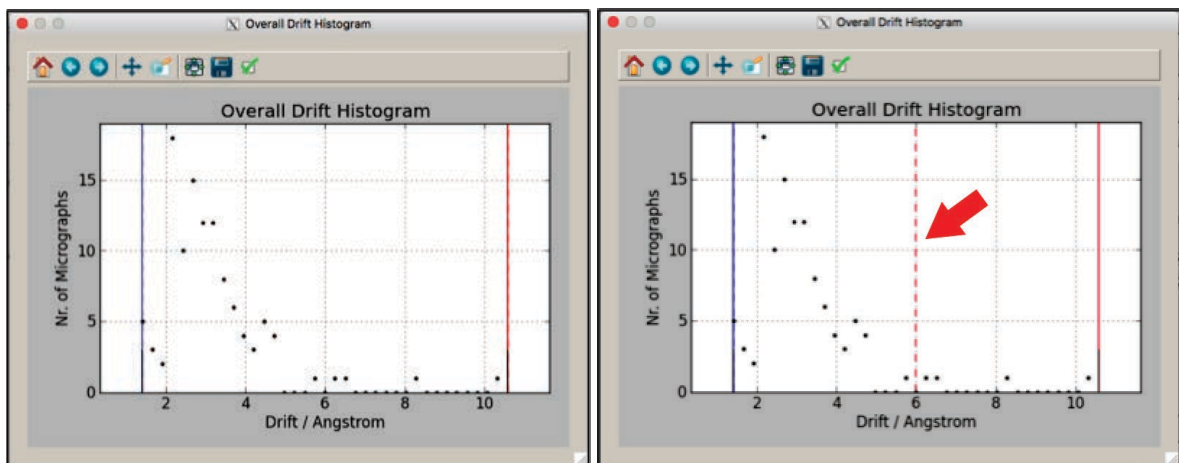


Click on the **Select .txt** button next to **Input shift list file** and use the file browser to select the input shift list file (**CorrectedSums/unblur_shiftfiles.txt**) and press the **Run command** button to open the **GUI**.



Check the **Overall Drift Histogram** (1) checkbox. On the plot for the overall drift (y-axis: number of micrographs, x-axis: overall drift in Å) you can set a threshold to discard micrographs with an overall drift higher than a specific value (red line).

In the tutorial dataset, the average overall drift is 3.2 Å (**Drift info for selected micrographs**). Right click on the plot to set the discard threshold to roughly 6 Å.





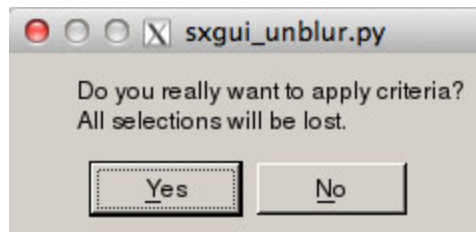
In order to register this threshold, press the **Register** button (2).

Threshold Overall Drift	
Start (A):	1.406186
Stop (A):	6.020822
Registered Start (A):	1.406186
Registered Stop (A):	6.020822

Register

Use as criterion

Click **Apply registered settings marked as criterion** (3) and confirm the pop-up message box:



Now type *Tutorial* as the prefix name for the output list files and click the **select output directory and save selection** button (4) to specify an **output directory** (make a new folder with the name **DriftAssess**)

Output Prefix: Tutorial

Select output directory and save selection

Four text files are then written to the folder **DriftAsses** in your **project directory**.

- **Tutorial_selected.txt**: List of selected micrographs; 108 micrographs
- **Tutorial_discarded.txt**: List of discarded micrographs; 4 micrographs
- **Tutorial_shift_selected.txt**: Shifts of selected micrographs
- **Tutorial_shift_discarded.txt**: Shifts of discarded micrographs

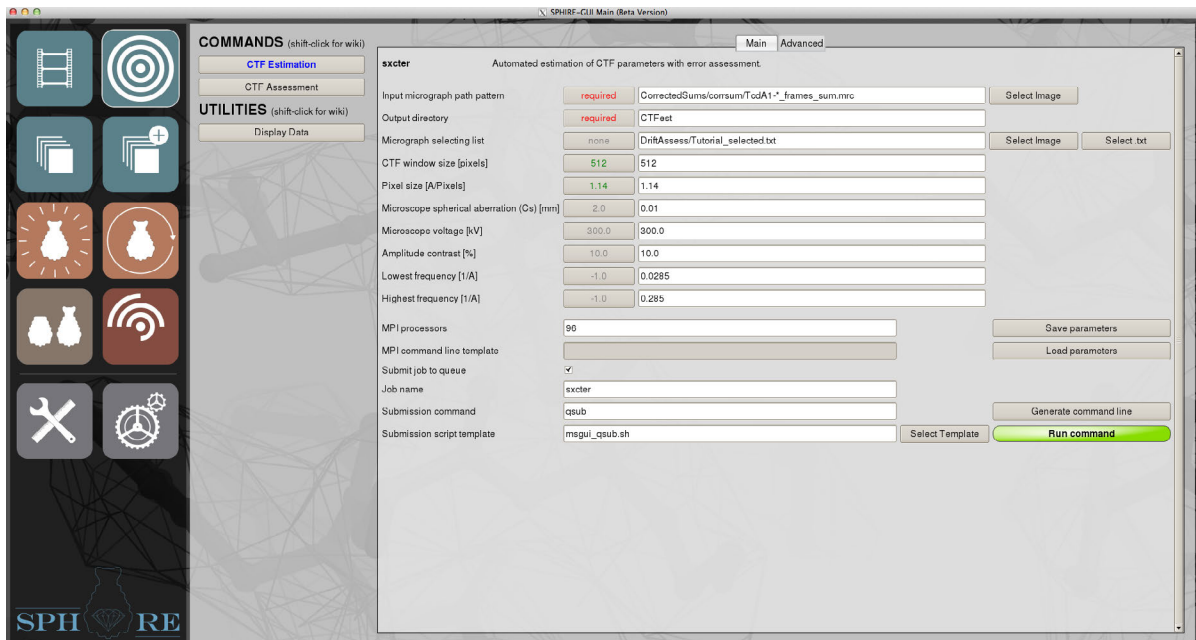
TIP: A more advanced usage of the **Drift Assessment** tool is described on our [wiki](#) page. For example, if your dataset contains a sufficient number of micrographs, selecting the movies with the lowest drift rate during the first frames (drift per frame criterion) might have a positive effect on the final resolution of the reconstruction.



CTER

CTF Estimation

The next step is to estimate the **CTF** of the selected motion-corrected micrographs that are included in the list **Tutorial_selected.txt** using **CTER** (Automated estimation of **CTF** parameters with error assessment) (Penczek et al. 2014). Note that the **CTF** estimation will be performed exclusively on the *not* dose-weighted micrographs. Go to the main window of the **SPHIRE GUI** and press the button **CTER** on the left and then the button **CTF Estimation** in the middle.





Provide following parameters:

- **Input micrograph path pattern:** CorrectedSums/corrsum/TcdA1-*_frames_sum.mrc

NOTE: Press the **Select Image** button and use the file browser to select one of the motion-corrected but dose-unweighted averages in the **corrsum** folder. Then replace the variable part of the file name (the serial number) with the wildcard character “*”.

- **Output directory:** CTFest
- **Micrograph selecting list:** DriftAssess/Tutorial_selected.txt

NOTE: Press the **select .txt** button and use the file browser to load the list of selected micrographs created in the previous step

- **CTF window size [pixels]:** 512

NOTE: Should be slightly larger than particle size

- **Pixel size [Å/Pixels]:** 1.14
- **Microscope spherical aberration (Cs) [mm]:** 0.01

NOTE: The images of the tutorial dataset were collected on a Cs corrected Titan Krios

- **Microscope voltage [kV]:** 300
- **Amplitude contrast [%]:** 10

TIP: Amplitude contrast is typically in the range of 7% to 14%. 10% yields good results for many projects in our lab.

- **Lowest Frequency [1/Å]:** 0.0285

NOTE: low-resolution cut-off

IMPORTANT: Set the search range for the CTF estimation approximately from $1/40$ 1/Å to $1/4$ 1/Å for optimal results. Provide the starting and stop frequency in spatial frequency units. For example, starting frequency in spatial frequency units in this case corresponds to:

$$\text{Resolution} = \frac{1}{\text{spatial frequency}} = \frac{1}{0.0285 \text{ 1/Å}} \approx 35 \text{ Å}$$

- **Highest Frequency [1/Å]:** 0.285

NOTE: high-resolution cut-off

Specify the number of processors (we used 96 cores for this job) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button.



IMPORTANT: Number of processors should be always lower than the total number of micrographs.

Monitor the progress of the job through the standard output. On our cluster, this process finished after about 3 min. However, if you do not have enough time to wait for the results, copy our pre-calculated results to your **project directory**.

```
cp -r SphireDemoResults/CTFest ./
```

Once the job has finished, the CTF results are stored in **CTFest/partres.txt** that contains 18 columns with the following information:

Example row of **CTFest/partres.txt** for the micrograph **TcdA1-0001_frames_sum.mrc**. The results of your **CTER** run can slightly differ.

Column Number	Parameter	Value	Unit
1	Image Defocus	2.253 2	μm
2	Microscope Cs	0.01	mm
3	Microscope Voltage	300	kV
4	Pixel Size	1.14	\AA
5	B-Factor	0	\AA^2
6	Amplitude Contrast	10	%
7	Astigmatism Amplitude	0.036 446	μm
8	Astigmatism angle	24.074	$^\circ$
9	SD of defocus	0.001 228 3	μm
10	SD of astigm. amplitude	0.002 470 1	μm
11	SD of astigm. angle	2.590 3	$^\circ$
12	Coeff. of defocus variation	0.054 513	-
13	Coeff. of astigm. amplitude variation	6.777 4	-
14	Spectrum mean Diff	0.015 021	-
15	Defocus Frequency limit	0.436 88	$1/\text{\AA}$
16	Defocus and astigm. frequency limit	0.270 7	$1/\text{\AA}$
17	Frequency limit of the CTF model	0.438 6	$1/\text{\AA}$
18	Micrograph Name	Directory/TcdA1-100_frames_sum.mrc	

TIP: Open the **CTER** output using a text editor and confirm that the image defocus is within the range you used during data collection and that the standard deviation for the respective micrographs (column 9) is rather low

More information regarding the different parameters of the **CTER** output can be found on our [wiki](#) and the **CTER** paper (Penczek et al. 2014).



CTF Assessment

The **CTF** results can be analyzed using the **CTF Assessment** tool (`sxgui_cter.py`). This tool is mostly useful to assess the quality of the **CTF** estimation, the overall quality of the dataset and to identify and remove outliers that might have a negative impact on the final result. Most importantly, this selection can be performed using multiple criteria simultaneously. For example, if you collected thousands of images in an automated manner, with the **CTF GUI** can you to select images that match certain criteria within seconds. Here are some example criteria, to select images:

⇒ within a specific defocus range

NOTE: *If you aim to reconstruct a rather “large” particle, your dataset contains enough images and contrast is not an issue, why would you like to keep far-from-focus images in your dataset?*

⇒ with low standard deviations regarding the defocus

NOTE: *A precise defocus estimation is critical for a near atomic resolution reconstruction*

⇒ with a certain frequency limit

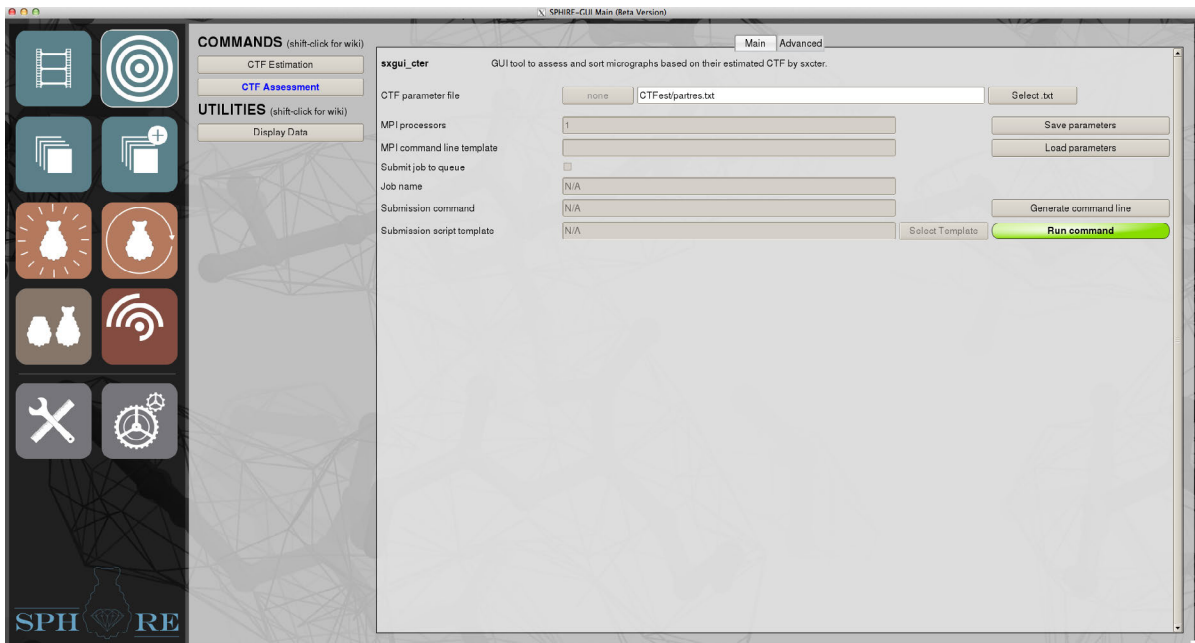
NOTE: *For example, a dataset that includes micrographs with Thon-rings $< 3 \text{ \AA}$ exclusively, will most probably produce a better resolved structure than a dataset which also includes micrographs with Thon-rings limited to 7 \AA .*

This procedure not only reduces the size of the dataset and the processing time, but also might improve the quality of the reconstruction.

For detailed information about this tool, please refer to our [wiki](#).



In this tutorial, we will only perform a rather simplified screening by setting thresholds for the defocus value and the astigmatism frequency limit (1st and 16th column in **partres.txt**). Go to the main window of the **SPHIRE GUI** and press the button **CTER** on the left and then the button **CTF Assessment** in the middle.

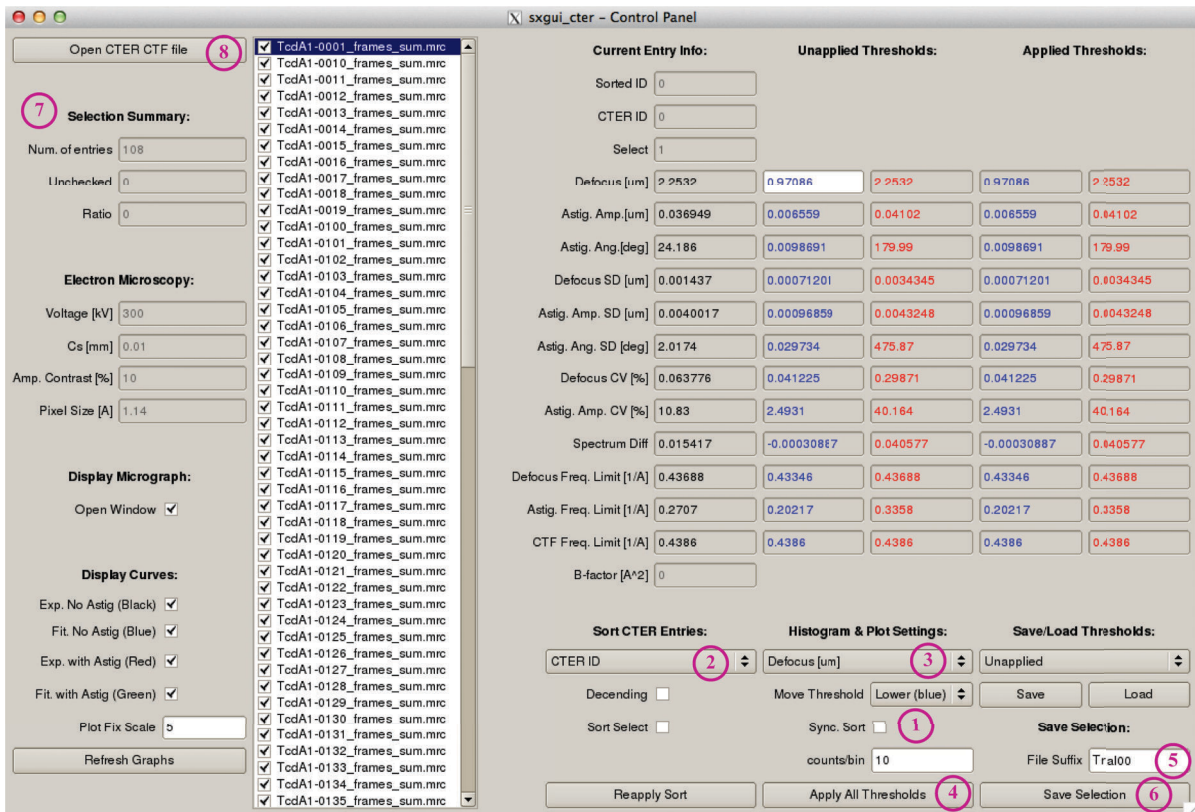


Click the **Select.txt** button and use the file browser to select the **CTF** parameter file **partres.txt** in the **CTFest**-folder and press the **Run command** button to launch the **GUI** tool and six windows will pop up:

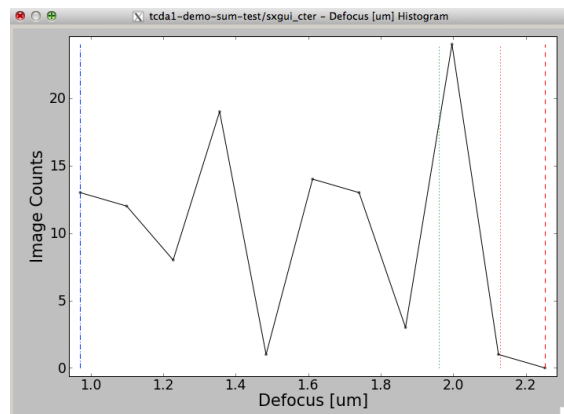
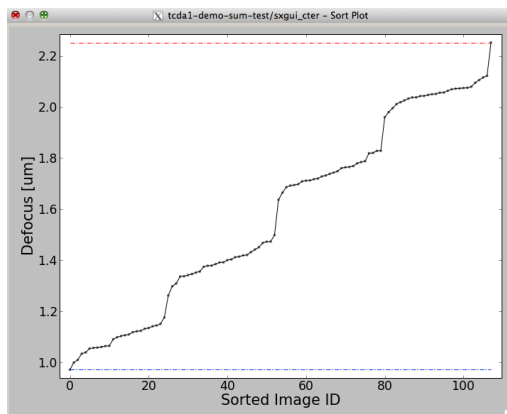
- **Control Panel**
- **Histogram**
- **Sort Plot**
- **Zoom Plot**
- **Plot**
- **Micrograph**



This is the **Control Panel** window:

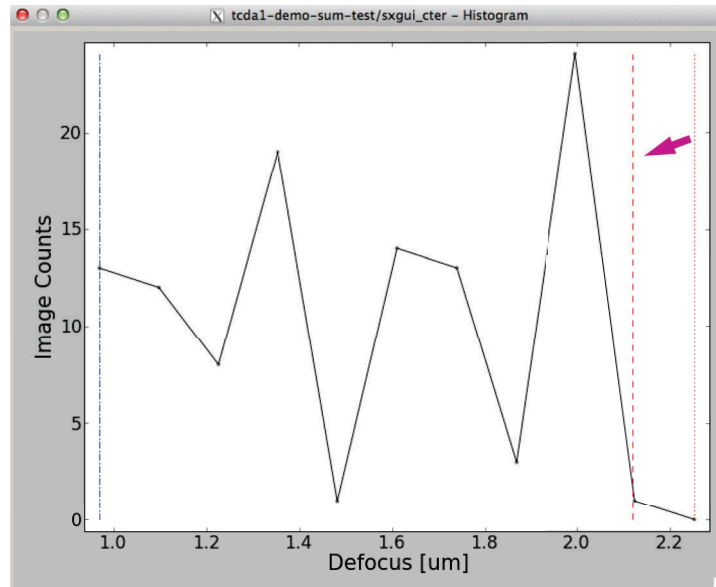


First, check the **Sync. Sort** option (1). This will synchronize your parameters selection in **Sort CTER Entries** (2) and **Histogram & Plot Settings** (3) and their respective windows. Now we will analyze the defocus distribution of this tutorial dataset. Select **Defocus [um]** from the combo box (3) and check the **Sort Plot** and **Histogram** windows.





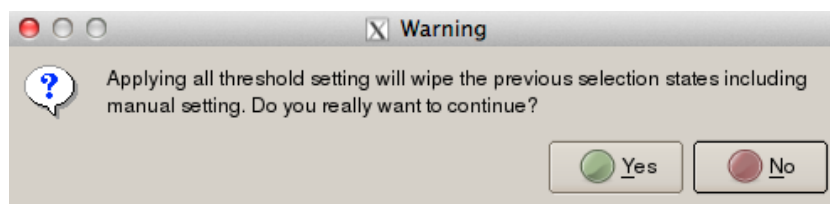
From both plots, we can see that the defocus of this dataset ranges from 0.97 μm to 2.25 μm . To demonstrate the functionality of this tool, we will now discard all micrographs with a defocus $> 2.13 \mu\text{m}$ by setting the respective threshold. **Left click** on the **Histogram** window while pressing the **SHIFT** button to set the threshold at 2.13 μm and a red dashed line will appear at the respective position. Alternatively, you can set this threshold directly at the respective input field.



TIP: You can also set a low cutoff-threshold, to discard micrographs with a defocus lower than a specific value. In this case **left click** on the **Histogram** window at the respective value and a blue dashed line will appear at the respective position.

NOTE: The **Sort Plot** window supports the same mouse control. Usually, we use this approach to identify defocus-“outliers” and/or select micrographs within a specific search range.

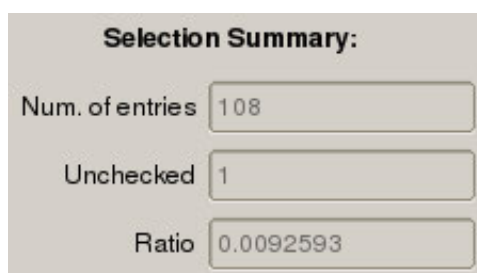
In order to apply the user-defined defocus threshold, press the **Apply all thresholds (4)** button. A message dialog will pop up. Press **Yes** to apply the threshold.



NOTE: The respective outliers have now been unchecked in the micrograph list.

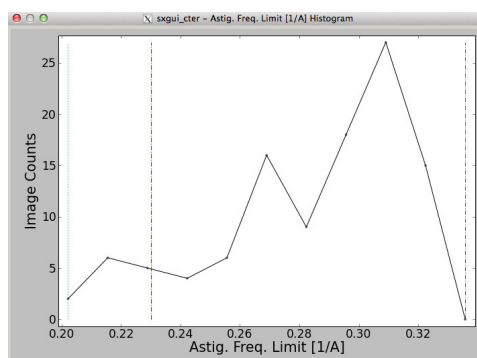


You can see the number and ratio of the unchecked images in the **Selection Summary** (7).



TIP: You can also visually examine unchecked micrographs before discarding them. For this purpose, activate the **Sort Select** checkbox under **Sort CTER Entries**. All unchecked micrographs will now be placed at the top of the micrograph list. **Left click** on an unchecked entry in the list to display the associated micrograph, graphs and parameter values. In the **Histogram** and **Sort Plot** windows, the green line indicates the value of the selected entry. If you want to keep the respective micrograph, **check** the checkbox next to the file name in the micrograph list.

Now, we will use a second selection criterion to discard micrographs: **Astigm. Freq. limit [1/Å]**. This parameter is always a good criterion to identify micrographs with low high-resolution information content. In the **control panel**, instead of **Defocus [um]**, now select the **Astigm. Freq. limit [1/Å]** parameter from the combo box (3) and check again the **Histogram** and **Sort Plot** windows. Using the procedure described above, set a low cut-off threshold at about 0.24 1/Å and press again the **apply all thresholds** button (4) (the higher the limit of astigmatism, the better the quality of the micrograph is; thus we want now to discard micrographs having a limit below this value). After applying this threshold, the low value cut-off (blue dashed line) should be at the following position in the **Histogram** plot:



NOTE: The value of 0.24 1/Å, corresponds to a resolution limit of one per threshold = $1 / 0.24 \text{ 1/Å} = 4.17 \text{ Å}$. Thus, in this case we will discard all micrographs that have an astigmatism resolution limit $> 4.17 \text{ Å}$.



TIP: You can also use other parameters to screen micrographs. However, from our experience **Astigm. Freq. limit** [$1/\text{\AA}$] is the most convenient criterion to identify bad micrographs. Thus, shifting the mean value of the **Astigm. Freq. limit** [$1/\text{\AA}$] of your dataset to a higher value, might improve the final outcome. Moreover, we recommend to discard extreme defocus values and also use **Astig. Amp.** [μm], if in addition you prefer to discard micrographs with high amount of astigmatism. However, as long as Thon rings in images extend sufficiently far and the astigmatism estimations are precise, astigmatism is not per se detrimental to high-resolution work. Regarding the standard deviations (**SD** parameters), always remove micrographs with unusually high errors.

Now the number of unchecked micrographs has been increased from 1 to 10. Enter **Tutorial** in **File Suffix** (5) under **Save Selection** in the **Control Panel**, and press the **Save Selection** button (6). A message dialog will pop up and inform you that following files are saved:

– **Tutorial_micrographs_select.txt:**

NOTE: List of selected Micrographs

– **Tutorial_shift_discard.txt:**

NOTE: List of discarded micrographs

– **Tutorial_partres_select.txt:**

NOTE: **CTF** parameters of the selected micrographs

– **Tutorial_partres_discard.txt:**

NOTE: **CTF** parameters of discarded micrographs

– **Tutorial_thresholds.txt:**

NOTE: Applied thresholds

These files are automatically stored in the location of the input **CTF** parameter file **partres.txt** in the folder **CTFest**.

TIP: After this first selection step with this **GUI** tool you can load the **CTF** parameter file of the selected micrographs (**tutorial_partres_select.txt**) by clicking the **CTER CTF File** button (8) and perform a second round of screening. A more detailed description of the **CTF Assessment** tool is provided on our [wiki](#) page.



Window

Particle Picking

In order to extract particles from the selected micrographs, we need the respective coordinate files. **SPHIRE** does not directly provide a graphical program for selecting particles from micrographs. Nevertheless, this task can be performed with the particle picker **e2boxer.py** from the **EMAN2** software package, which is installed jointly with our package. The first step is to move all our discarded images into a different folder (we want to pick particles only from the selected micrographs). For this purpose, create a new folder named **DISCARDED** in the directory **CorrectedSums/corrsum_dose_filtered**, which contains your dose corrected micrographs.

At the terminal type:

```
path=CorrectedSums/corrsum_dose_filtered;/mkdir ${path}/DISCARDED
```

Now we will use the lists of discarded micrographs, which were created in the previous steps, to move these micrographs to the **DISCARDED** folder.

In your project directory, type:

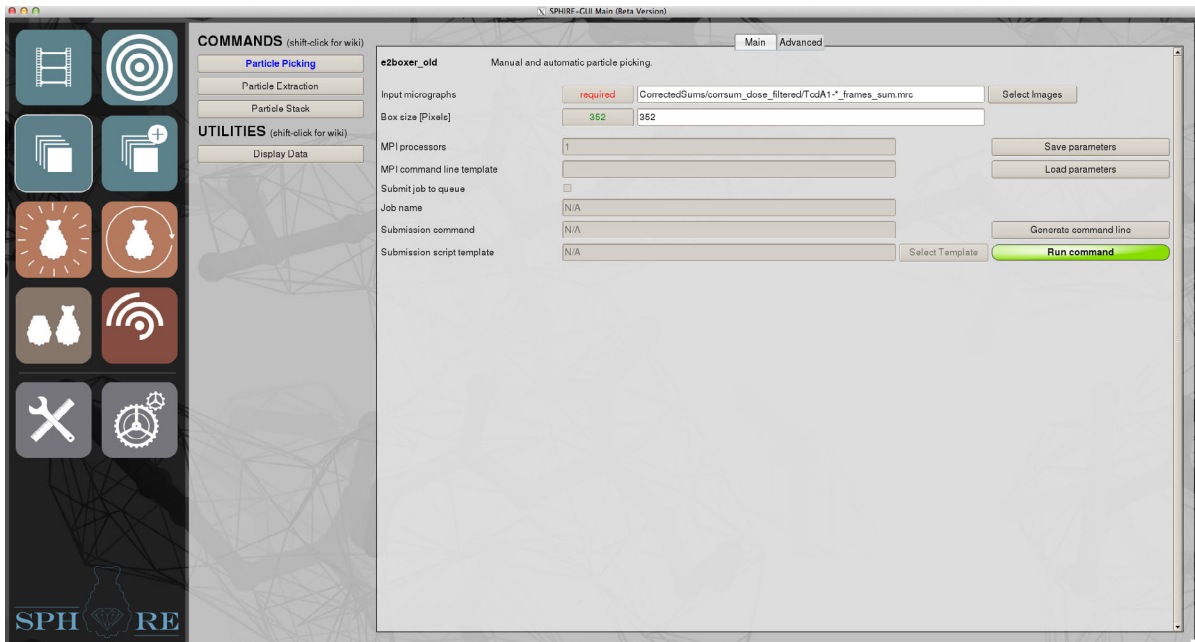
```
while read file; do mv ${path}/${file} ${path}/DISCARDED/;
done < CTFest/Tutorial_micrographs_discard.txt

while read file; do mv ${path}/${file} ${path}/DISCARDED/;
done < DriftAssess/Tutorial_discarded.txt
```



Window

Now, go back to the main window of the **SPHIRE GUI** and press the button **Window** on the left and then the **Particle Picking** button in the middle.



Click the **Select Images** button and use the file browser to select one of the micrographs in **CorrectedSums/corrsum_dose_filtered**

NOTE: *This folder contains only good micrographs.*

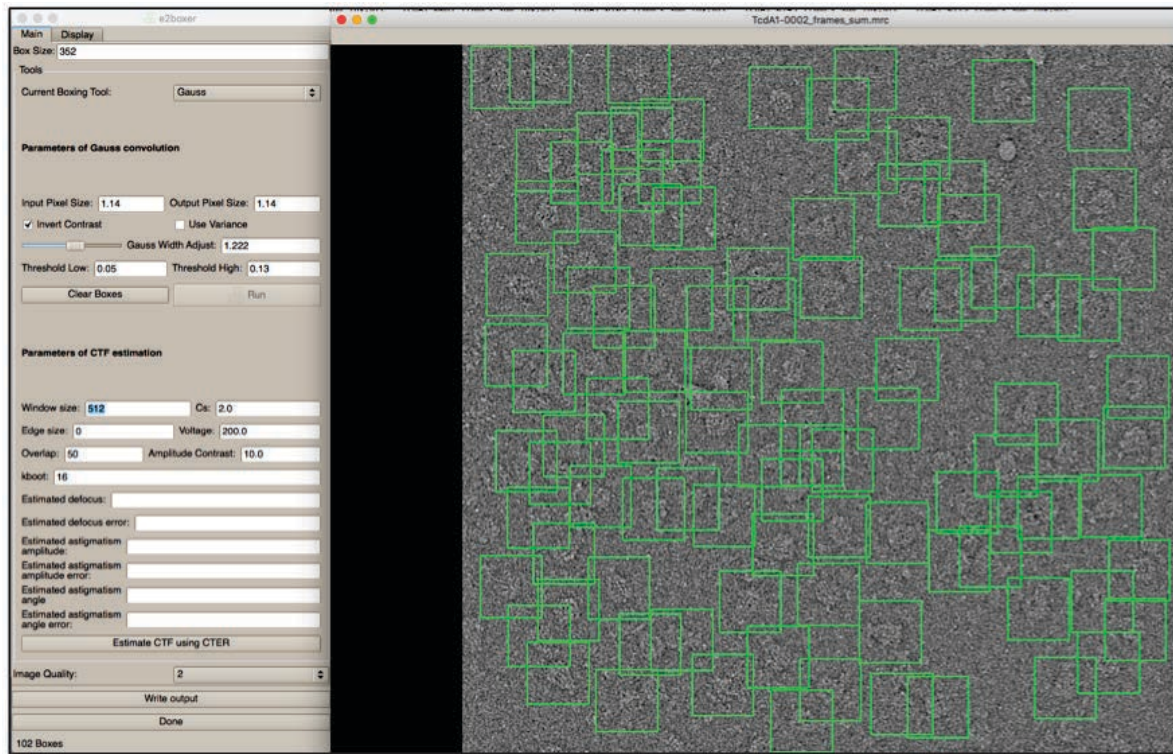
Replace the variable part of the file name (in this case the serial number) with the wildcard character “*” (e.g. from **TcdA1-0010_sum.mrc** to **TcdA1-*_sum.mrc**) and press the **Run command** button to launch **e2boxer GUI**. Pick the particles manually or automatically (using the manual or the gauss tool, respectively) and store their coordinates in the **.box** file format (**EMAN1**) in a folder named **Coordinates**. Typical settings for the gauss boxer are shown in the image below.

TIP: *The e2boxer swarm auto-picker, would also perform well for this dataset, but in case you want to use it, you will have to invert contrast of micrographs first. For this purpose, we usually use EMAN2’s command e2proc2d.py*

```
e2proc2d.py input-micrograph.mrc inverted-micrograph.mrc --mult=-1
```

For additional information about the **e2boxer.py** tool please refer to:
<http://blake.bcm.edu/emanwiki/EMAN2/Programs/e2boxer>

TIP: *Due to vast differences between samples, testing of different methods and external software packages might be necessary to facilitate the best performing automated particle selection method.*



In case you do not have the time to perform autopicking yourself, you can use the coordinate files we obtained when we picked these selected micrographs with gauss autoboxing using **e2boxer.py**.

In this case type:

```
cp -r SphireDemoResults/Coordinates ./
```

The folder **Coordinates** within the **project directory** will include a coordinate file for every micrograph.

When typing

```
ls Coordinates/ | wc -l
```

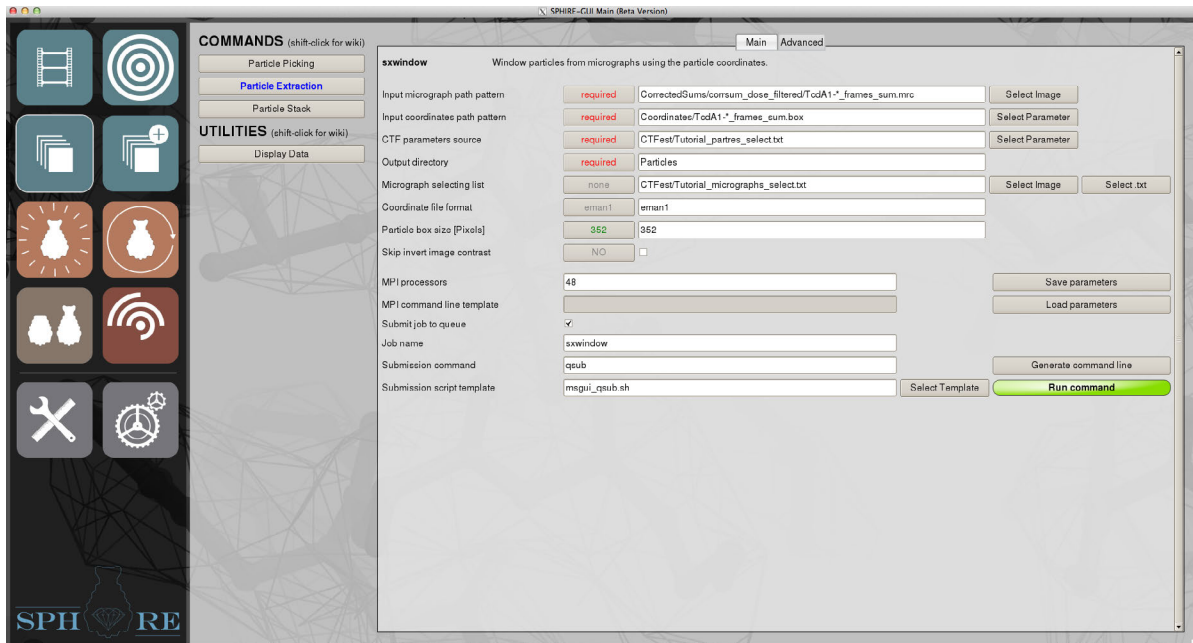
you will see that the folder contains 97 coordinate-files.

TIP: If you picked your micrographs with a different particle picker, copy the coordinate files to the **Coordinates** folder and continue from here. It is not necessary to use the tutorial naming convention. We prefer using **EMAN1**'s coordinate file format (.box), but **SPHIRE** also supports coordinates from **EMAN2** and **SPIDER** with the .json and .spi extension, respectively. In case you picked your micrographs using **Relion** (.star), you can use the application **sxrelion2sparx.py** to convert them to **EMAN1** file format (.box). Detailed instructions are provided in the **Import a star file** page of the [wiki](#).



Particle Extraction

Go back to the main window of the **SPHIRE GUI** and press the button **Window** on the left and then the **Particle Extraction** button in the middle.



Provide the following parameters at the **GUI** interface:

- **Input micrograph path pattern:** CorrectedSums/corrsum_dose_filtered/
TcdA1-*_frames_sum.mrc

NOTE: Click the **Select Image** button and use the file browser to select one of the micrographs in the *CorrectedSums/corrsum_dose_filtered* folder. Replace the variable part of the file name with the wildcard character “*” (e.g. from *TcdA1-0010_frames_sum.mrc* to *TcdA1-*_frames_sum.mrc*).

TIP: If you process negative stain data and the particle coordinates are available, you can start directly with particle extraction and skip all previous steps. In this case, insert here **The/path/to/your/Micrographs/*_mrc** instead

- **Input coordinates path pattern:** Coordinates/TcdA1-*_frames_sum.box

NOTE: Click the **Select Parameter** button and use the file browser to select one of the coordinate files in the *Coordinates* folder. Replace the variable part of the file name with the wildcard character “*” (e.g. from *TcdA1-0010_frames_sum.box* to *TcdA1-*_frames_sum.box*). In this demo the folder *Coordinates* contains 97 .box files)



TIP: *It is not necessary to use the same root names for micrographs and coordinates files. However, the variable part, as defined by the wildcard character, should be identical for the associated pair of input micrograph and coordinate file.*

- **CTF parameters source:** CTFest/Tutorial_partres_select.txt

NOTE: *Click the **Select Parameter** button and use the file browser to select the **tutorial_partres_select.txt** file in the **CTFest** folder. This file contains the **CTF** parameters of the selected micrographs. The **CTF** information will be forwarded to the header of all extracted particles from the respective micrograph.*

TIP: *If you process negative stain data and do not perform CTF correction, insert the pixel size of your micrographs instead.*

- **Output directory:** Particles

- **Micrograph selecting list:** CTFest/Tutorial_micrographs_select.txt

NOTE: *Click the **Select Parameter** button and use the file browser to select the **Tutorial_micrographs_select.txt** file (list of selected micrographs) in the **CTFest** folder.*

TIP: *If you process negative stain data and/or skipped the micrograph selection steps, leave this field empty.*

- **Coordinate file format:** eman1

NOTE: *Supported file formats are **eman1** (.box), **eman2** (.json) and **spider** (.spi)*

- **Particle box size [pixels]:** 352

- **Skip invert image contrast:** No

NOTE: *Leave this flag as it is. The program will invert the contrast of your **cryo-EM** images automatically.*

TIP: *If your particles appear white and the background dark (e.g. negative stain or inverted **cryo-EM** images) activate this flag.*

Specify the number of processors (for this job we used 48 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. Monitor the progress of the job through the standard output.

```
Global summary of coordinates level processing...
Detected : 8553
Processed : 8416
Rejected by out of boundary : 137
```

However, if you do not have enough time to wait for the results, copy our precalculated results to your **project directory**.

```
cp -r SphireDemoResults/Particles ./
```



Once the job has finished, the extracted particles for each micrograph are stored in the folder **Particles**. On our cluster, this step finished after about 1 min.

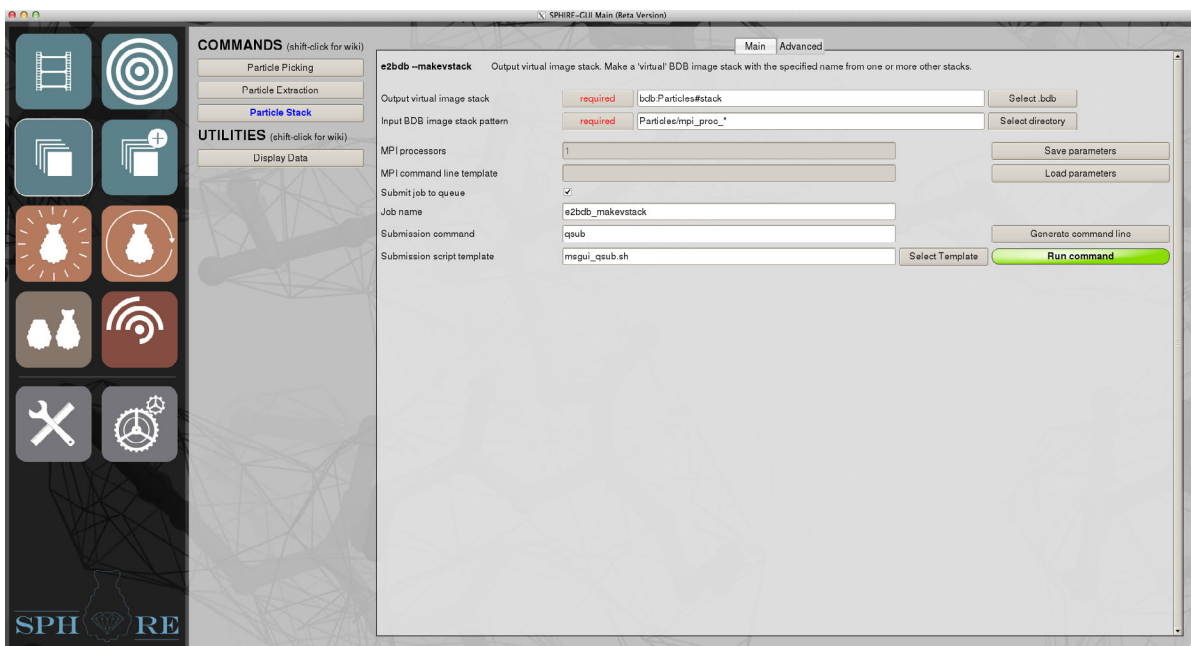


Particle Stack

The next step is to combine the separate particle stacks for each micrograph, into a single virtual particle stack. The virtual stack will be in the **EMAN2's BDB** format.

TIP: A so-called virtual stack contains only header information and links to the original images and therefore requires only little free disk space.

Go back to the main window of the **SPHIRE GUI** and press the button **Window** on the left and then the **Particle Stack** button in the middle.



Provide following information in the respective fields:

- **Output virtual image stack:** bdb:Particles#stack

NOTE: This will define the **Particles** folder as destination for the virtual stack

- **Input BDB image stack pattern:** Particles/mpi_proc_*

NOTE: Click the **Select directory** button and use the file browser to select folder **Particles** and then one of the **mpi_proc** folders, which includes particle stacks for micrograph files extracted from a specific processor. Replace the variable part of the name of the folder by the wildcard character “*” (e.g. **mpi_proc_000** to **mpi_proc_***).



Press the ***Run command*** button to create the stack and monitor the progress of this job through the standard output. This job might take 20 s to 30 s.

After the job has finished, you can check the number of particles in the resulting stack using EMAN2's **e2iminfo.py** command from the **project directory**.

```
e2iminfo.py bdb:Particles#stack
```

The stack should contain ~8 400 single particles.

You can also display the virtual stack using the utility ***Display Data***. For this purpose, go back to the main window of the **SPHIRE GUI** and press the ***Display Data*** button in the middle.



ISAC

The 2D classification is one of the most crucial steps of the workflow. It will give you a good idea about the quality and heterogeneity of your data and can become a great tool to sort out bad particles. **SPHIRE** uses the **ISAC** method (Yang et al. 2012) to perform 2D classification and extract validated and homogeneous subsets of images with minimal human intervention.

The **ISAC** program will:

- ⇒ Phase-flip the 2D images in the stack and prealign them using a reference-free approach.
- ⇒ Afterwards iterate through the so-called “generations” by alternating between calculation of candidate and validated class averages.

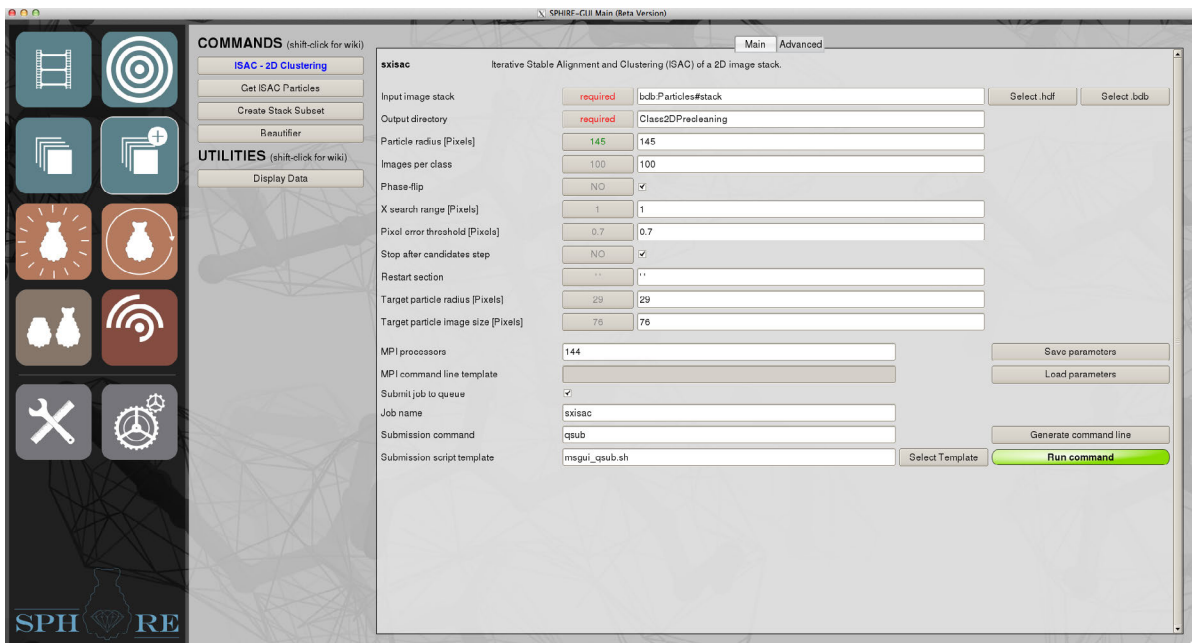
Regrettably, due to the high number of reproducibility tests during the validation procedure, **ISAC** is very time- and memory-consuming. That is why we recommend performing a pre-cleaning of your data first, especially if you auto-picked your micrographs, in order to reduce the size of your dataset before running the full version of the program and calculating reproducible classes.

TIP: *If you manually picked your data and/or you are confident that your dataset is very homogeneous, you might want to skip the next step. We also usually skip this step when we process negative stain data.*



Pre-cleaning

In this step, we will start a 2D clustering with **ISAC**, but stop the program directly after generating the candidate class averages. Then, we will identify *bad* class averages and remove their members from our dataset, in order to prepare a higher-quality and reduced-size dataset for the computationally expensive, but validated 2D Classification with **ISAC**. Go back to the main window of the **SPHIRE GUI** and press button **ISAC** on the left and then the **ISAC-2D Clustering** button in the middle.



Provide now the following parameters at the respective **GUI** interface:

- **Input image stack:** bdb:Particles#stack

NOTE: Click the **Select .bdb** button and use the file browser to select the virtual stack created in the previous step, containing all single particles, from all micrographs.

TIP: In principle, you can also directly load any type of single particle dataset that was created by a different program, as long as it has been converted to **EMAN2's BDB** or **HDF** file format (preferably **BDB**) and the **CTF** information is stored in the header of particles, in the expected format. In case you use **RELION**, after particle extraction, you can easily switch to **SPHIRE**, at any stage of processing, using **sxrelion2sparx.py** utility. Detailed instructions are provided in the **Import a star file** page of the [wiki](#).

- **Output directory:** Class2DPrecleaning



- **Particle radius [pixels]:** 145
- **Images per class:** 100

NOTE: Our dataset contains about 8 400 particles, thus by setting this parameter to 100, the expected number of candidate 2D averages will be about

$$\text{Number of classes} = \frac{\text{Total Number of Particles}}{\text{particles per class}} = \frac{8400}{150} \approx 84$$

TIP: Adjust this number, according to the size of your dataset. 200 particles per class is a good starting point for *cryo-EM* datasets of this size (~10 000 particles), but for larger datasets you might want to consider using a larger number (e.g. 500 particles per class). The computational time will increase significantly with increased number of class averages. Moreover, depending on the appearance of the resulting candidate class averages, you might need to adjust this value accordingly. For example, if the resulting averages after this pre-cleaning step look excessively noisy, you might want to repeat this step using a larger number of particles per class in order to suppress the noise. However, this is always at the expense of the amount of detail in averages: the larger the number of members per averages the poorer the apparent resolution.

TIP: Use 50 to 150 members per class for negative stain datasets of 5 000 to 15 000 particles.

- **Phase-flip:** Yes

TIP: Do not activate this flag if you process negative stain data.

- **X search range [pixels]:** 1
- **Pixel error threshold [pixels]:** 0.7

NOTE: *ISAC* evaluates quality of class averages by performing multiple reference-free *ab initio* alignments of images within a class. Orientation parameters adopted by individual images in these independent runs are compared and their average dispersion is reported as **pixel error** of a given class. Class averages whose **pixel errors** exceed the threshold are discarded and their image members are passed for processing in subsequent generation. **Pixel error** is thus one of the most important *ISAC* parameters. However, for this step, we will stop *ISAC* directly after generation of the candidate class averages, so no validation will be performed and there is no need to set this value at this stage.

- **Stop after candidates step:** Yes

NOTE: This is useful only for the pre-cleaning step

- **Restart section:** ”

NOTE: Leave the default value.



- **Target particle radius [pixels]:** 29

NOTE: Leave the default value.

- **Target particle image size [pixels]:** 76

NOTE: Leave the default value.

NOTE: To speed up the process, the particles of the input dataset will be resized to match the user provided particle radius will be 29 pixel and the box size 76 pixel. Thus, the resulting class averages will have a new (usually larger) pixel size and most probably high-resolution features (like secondary structure elements) will not be visible. We are not concerned about this right now, since our goal at this point is to identify homogeneous subsets of images. The useful conversion formula is:

$$\text{Target particle radius} = \frac{\text{Target particle image size} \cdot \text{Original particle radius}}{\text{Original particle image size}}$$

NOTE: Computational time will increase exponentially with increasing target particle image size.

Specify the number of processors (for this job we used 96 cores; the number of processes has to be divisible by 4) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. Monitor the progress of the job through the standard output. On our cluster, this job with 96 processes finished after about 30 min. However, if you do not have enough time to wait for the results, copy our precalculated results to your **project directory**

```
cp -r SphireDemoResults/Class2DPrecleaning ./
```

Once the job has finished, following folders will be created in the **Class2DPrecleaning** folder:

- **2dalignment:** Contains the results of the reference-free prealignment.

NOTE: Display the total-average of your dataset after the pre-alignment (*Class2DPrecleaning/2dalignment/aqf.hdf*) using the **Display Data Utility**.

TIP: Confirm that the 2D total-average is properly centered and that the circular 2D mask includes the complete particle. If this is not the case, adjust the particle radius to a larger value (project-wide parameter) and rerun the pre-cleaning procedure.

- **generation_001:** Contains the candidate class averages.

NOTE: Display the file (*Class2DPrecleaning/generation_0001/class_averages_candidates_generation_1.hdf*) using the **Display Data** utility. The high quality of this dataset is reflected in the quality of the 2D class averages. This file contains only 1 to 2 low quality noisy class averages.



The next step is to delete all *bad* candidates class averages (e.g. blurry class-averages, ice contamination, carbon-edge etc..) using the **Display Data** Utility. To do so, press the **mouse wheel** button somewhere on the graphics window and activate the **Del** button (1), in the pop up window.

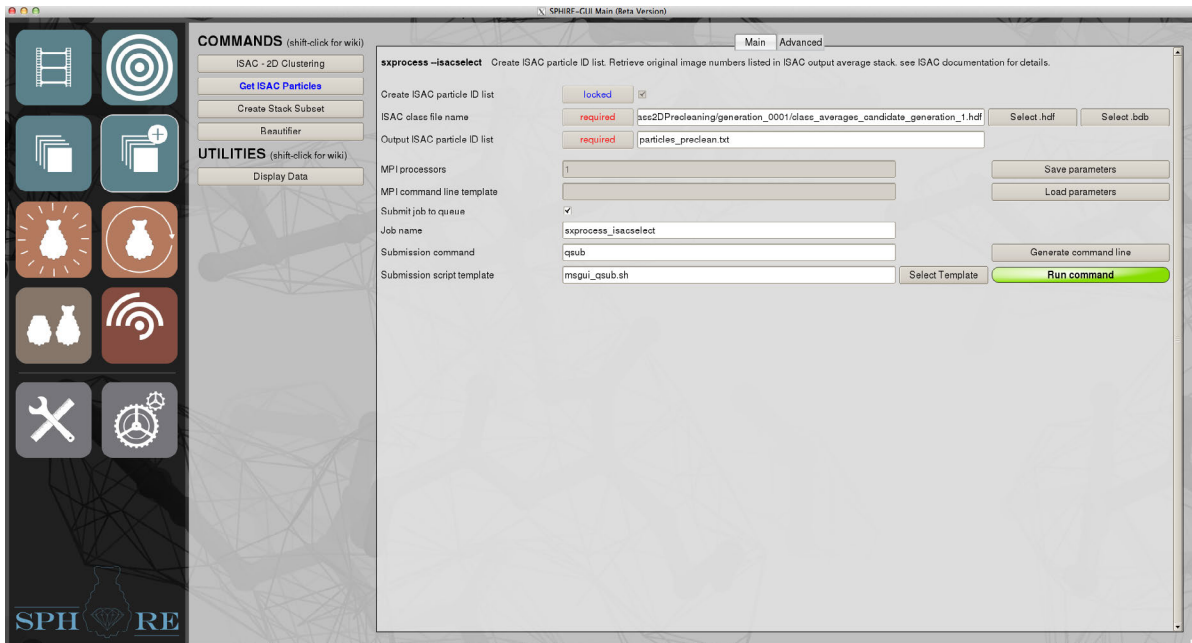


Then select the *bad* class averages by **clicking** on the respective images (2) and press the **Save** button (3), in order to save the remaining images under a new name: **class_averages_candidate_generation_1_best.hdf**.

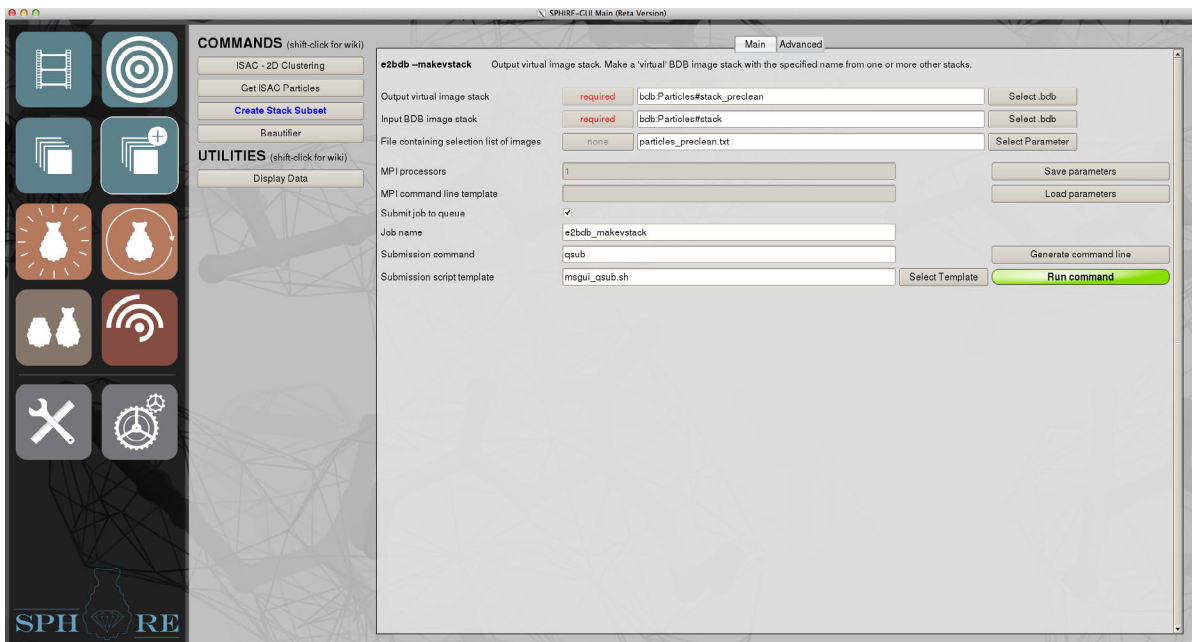
TIP: On the pop up graphics window, press the **Values** button and select the **n_objects** parameter to display the number of images for each class average.



Now we will retrieve the **particle-IDs** of the members of the remaining class averages. Go back to the main window of the **SPHIRE GUI** and press the button **ISAC** on the left and then the **Get ISAC Particles** button in the middle.



Use the file browser to select the file containing the *good* class averages only (**Class2DPrecleaning/generation_0001/class_averages_candidates_generation_1_best.hdf**), define **particles_preclean.txt** as the output file and press the **Run Command** button.





Now we will create a virtual *clean* stack that will contain only the particles listed in this file. Go back to the main window of the **SPHIRE GUI** and press the button **Create Stack Subset** in the middle.

Now provide the following parameters in the **GUI** input fields:

- **Output virtual image stack:** bdb:Particles#stack_preclean
- **Input BDB image stack:** bdb:Particles#stack

*NOTE: Click the **Select .bdb** button and use the file browser to select the virtual stack containing all particles in the **Particles** folder.*

- **File containing selection list of images:** particles_preclean.txt

*NOTE: Click the **Select Parameter** button and use the file browser to select the list with the IDs of good particles created in the previous step.*

and press the **Run command** button.

Use **EMAN2's e2iminfo.py** to check the number of particles of the new *clean* dataset.

```
e2iminfo.py bdb:Particles#stack_preclean
```

The *precleaned* stack contains approximately 7 700 particles.

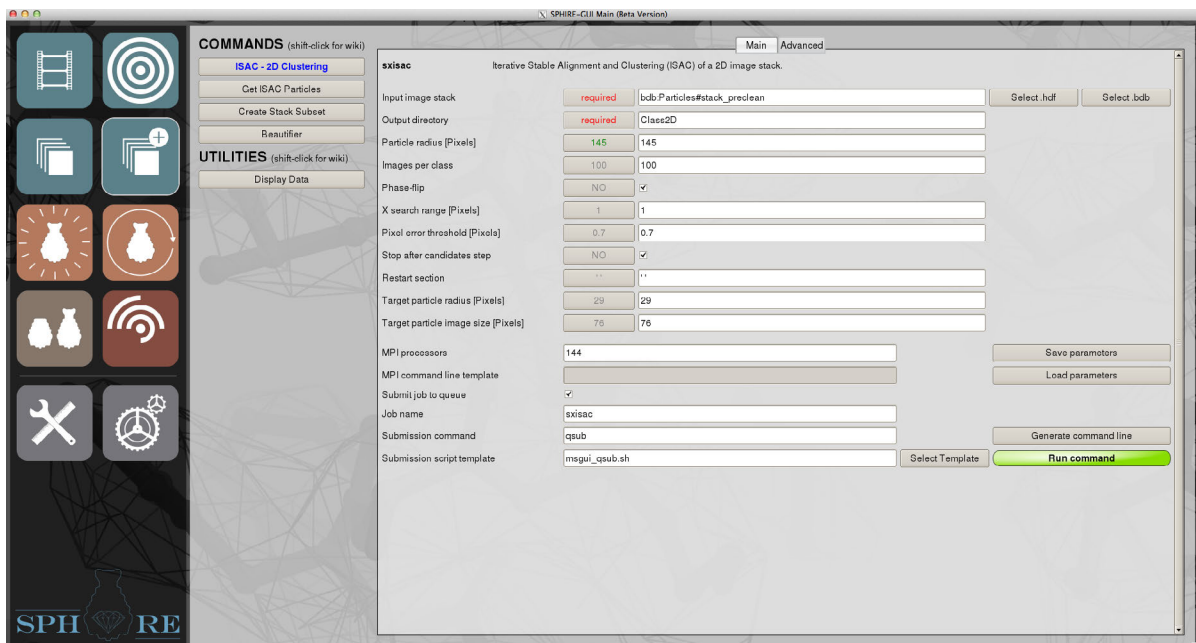
NOTE: The number of particles can slightly differ for your run.



ISAC 2D Clustering

We will use the **ISAC** approach to obtain validated class-averages of the *precleaned* dataset. This step is much more computationally demanding than the previous one and the running time increases significantly with the number of particles and classes.

Go back to the main window of the **SPHIRE GUI** and press the button **ISAC** on the left and then the **ISAC-2D Clustering** button in the middle.



Provide the following parameters at the respective **GUI** interface using the same settings we used before, but this time not stopping after candidate step:

- **Input image stack:** bdb:Particles#stack_prclean
- **Output directory:** Class2D
- **Particle radius [pixels]:** 145
- **Images per class:** 100
- **Phase-flip:** Yes
- **X search range [pixels]:** 1
- **Pixel error threshold [pixels]:** 0.7



TIP: For *ISAC* runs with large datasets of excellent quality and low number of particles per class (e.g., 100), you should use a rather conservative value for this threshold (e.g., 0.7). With increased number of particles per class or in case you did not do pre-cleaning, a higher threshold value should be used, otherwise *ISAC* will most likely discard too many particles.

- **Stop after candidates step:** No
- **Restart section:** ”

NOTE: Leave the default value.

- **Target particle radius [pixels]:** 29
- **Target particle image size [pixels]:** 76

Specify the number of processors (for this job we used 96 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. Monitor the progress of the job through the standard output. On our cluster, this job with 96 processes finished after about 70 min.

However, if you do not have enough time to wait for the results, copy our precalculated results to your **project directory**

```
cp -r SphireDemoResults/Class2D ./
```

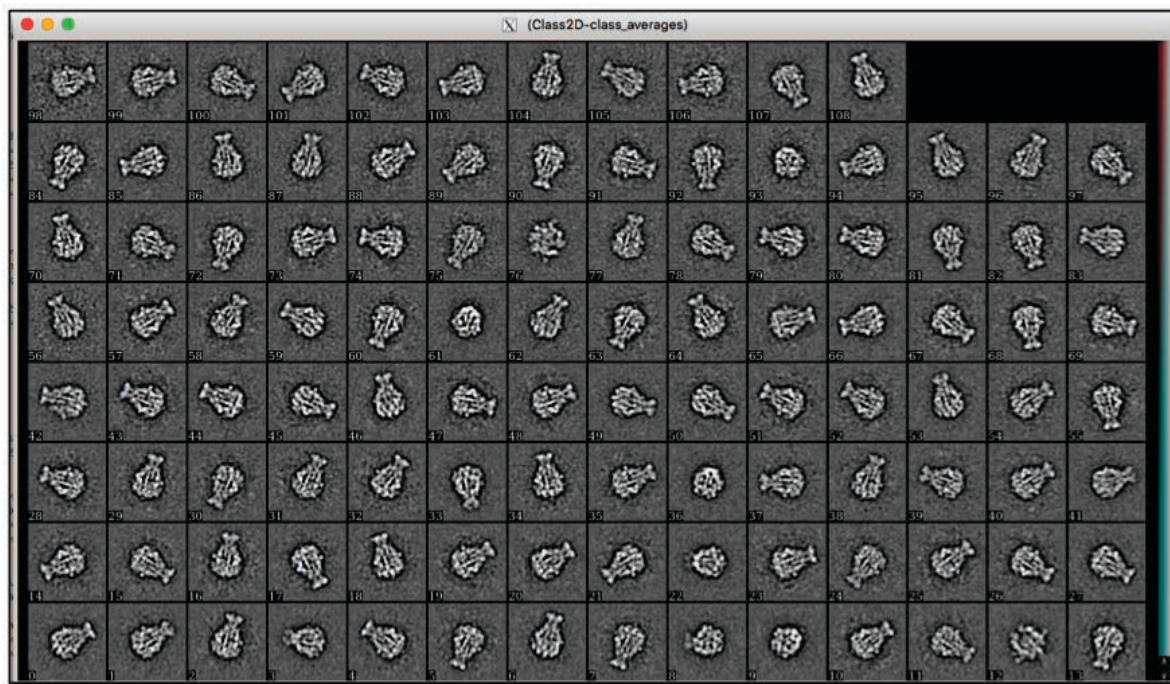
TIP: This is a computationally expensive process, that is why we strongly recommend checking the quality of the candidate class averages already during the first generation of *ISAC* when you process your own data. Display them (**2DClass/generation0001/class_averages_candidates_generation1.hdf**) using the **Display Data** utility. If they contain many ugly class averages, it is advisable to stop the program and rerun an additional pre-cleaning step (see previous section) before repeating *ISAC*. Please also check the numbers of members in each group. While displaying press the **mouse wheel button** somewhere on the **graphics window** and activate the **n_objects** attribute of the **Values** button of the pop up window, as described before. If the number of particles in each class is unusually low, you might want to re-run *ISAC* after setting the parameter **Pixel error threshold [pixels]** to a less conservative value (e.g. 1.7 instead 0.7) and also check if the radius of the particle (**project setting; longest axis of the particle in pixels**) is set correctly. If the files does not contain any class-averages at all (or very few), this means that no reproducible class averages could be found at this **Pixel error threshold [pixels]**. Try again to re-run *ISAC* after setting this parameter to a higher value. In case this does not help either, consider recording a dataset of higher quality or improve the quality of the autopicking. In most cases it is not possible to calculate a reliable reconstruction from a dataset that does not yield good 2D class averages.

Analogous to the pre-cleaning step, *ISAC*, if requested, will first perform a reference-free pre-centering of the particles and output the results in the subdirectory **2dalignment** of the folder **Class2D**.



The aligned total-average of the dataset is stored in the **aqfinal.hdf** file in the **2dalignment** folder. Use the **Display Data** utility to confirm it is correctly centered.

Next the program will continue with the first **ISAC** generation, but this time we will not stop it after producing candidate averages, as we did in the pre-cleaning step. Instead, **ISAC** will perform an addition stability test and subsequently output the validated class averages. Furthermore, after the first generation is finished, it will continue with a second generation, using only particles not accounted for the validated classes of the first generation, etc. This iterative procedure will stop when no more class average candidates can be found. For this tutorial dataset, the program will finish after five generations. The file **class_averages.hdf** in the **Class2D** folder will then contain the validated and reproducible class averages from all generation subdirectories combined in one file.



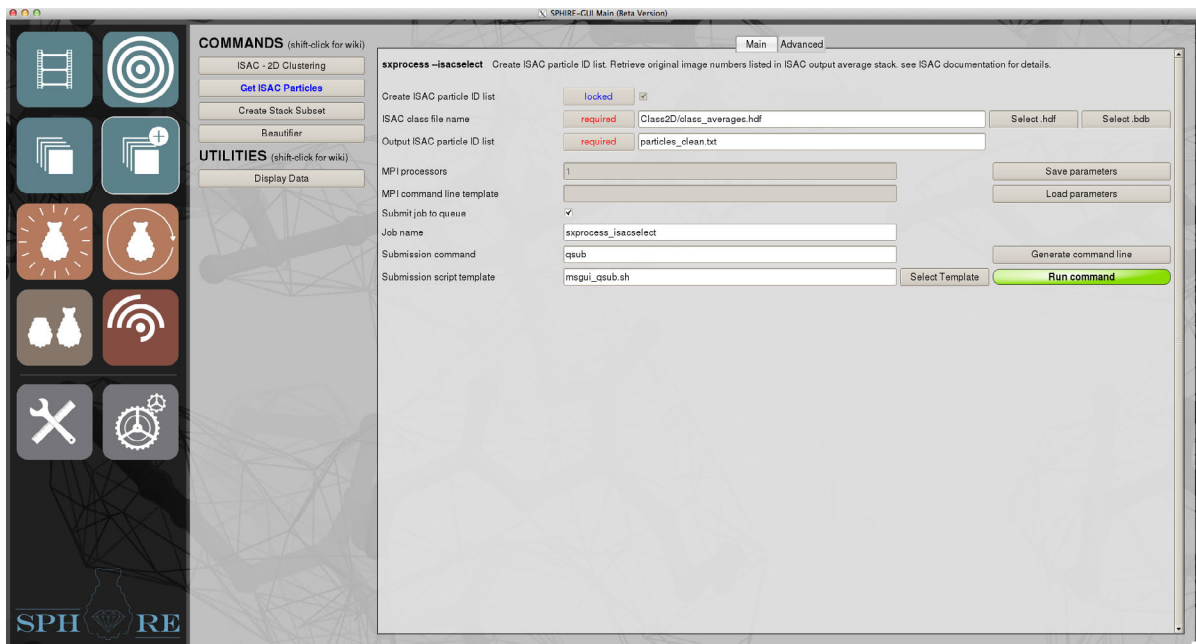
Use the **Display Data** utility to display the validated class averages. To speed up the process, similarly to the pre-cleaning step, the particles of the input dataset are resized, to the user defined parameters (in this case radius 27 pixel and box size 76 pixel). Thus, these final class averages have a new (larger) pixel size and high-resolution features (like secondary structure elements) are not visible. We are not concerned with this right at the moment as our goal at this point is the extraction of a validated, homogeneous subset of images.

TIP: In case you need the pixel size of the **ISAC** output, you need to divide the original pixel size by the shrink ratio, which can be found in a file called **README_shrink_ratio.txt** in the **Class2D** folder.



Although **ISAC** is a rather time consuming procedure, it will eliminate most of the *junk* particles of your dataset. This is a crucial step towards a high-resolution reconstruction. In other packages, this is mostly done during even more computational expensive 3D classifications, which usually have to be performed multiple times. In contrast, we prefer to work with *clean* datasets in 3D and usually we need to perform a heterogeneity analysis in 3D only once.

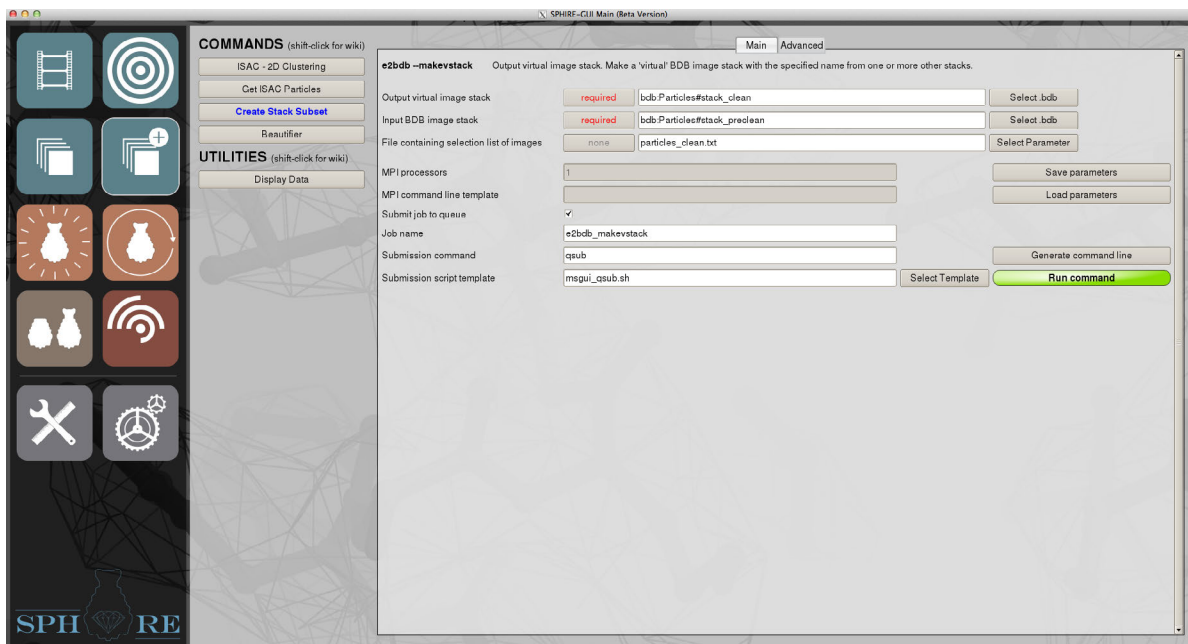
We will create a dataset that will include only the particle members of the final class averages. First, we retrieve the **particle-IDs** of the members of the final class averages and output them in a text file. Go back to the main window of the **SPHIRE GUI** and press the **Get ISAC Particles** button in the middle.



Use the file browser to select the final class averages (**Class2D/class_averages.hdf**), define **particles_clean.txt** as the output file and press the **Run command** button.



Now create a virtual *clean* stack containing only the particles listed in this text file. Go back to the main window of the **SPHIRE GUI** and press the button **Create Stack Subset** button in the middle.



Provide now the following parameters at the respective **GUI** input fields:

- **Output virtual image stack:** bdb:Particles#stack_clean
- **Input BDB image stack:** bdb:Particles#stack_preClean

*NOTE: Click the **Select .bdb** button and use the file browser to select the virtual stack that was used as input to run **ISAC***

- **File containing selection list of images:** particles_clean.txt

*NOTE: Click the **Select Parameter** button and use the file browser to select the file containing the list with the IDs of particles.*

and press the **Run command** button.

Use **EMAN2's e2iminfo** to check the number of particles of the dataset.

```
e2iminfo.py bdb:Particles#stack_clean
```

The *clean* stack should contain about 7 350 particles. In case of this high-quality dataset, **ISAC** eliminated only about 4 % of the particles.

***TIP:** For most of our datasets, the number of eliminated particles is usually much higher (20 % to 30 %). Sometimes the number of accounted particles does not even reach 50 %, but it is nevertheless preferable to work with smaller but cleaner datasets.*



The final class averages will be used to generate an initial 3D model in the next step of the tutorial. This model in turn will be refined against the particles members of these class averages (the clean stack) in a subsequent step.

TIP: *If you need high-resolution class averages (e.g. for publication purposes), use the utility **Beautifier**. This program will rebuild each requested class average output of **ISAC** with the original pixel size, after applying full **CTF**-correction and local re-alignment of the members against their respective average. A detailed description regarding the usage of this program is provided on our [wiki page](#).*

!Known issue in beta-release!

At the moment we still optimize the parallelization of **ISAC**. Regrettably, the current **ISAC** version cannot handle large datasets. On our cluster with 128 GB RAM and 24 cores per node, the limit is approximately 60 000 particles. The suggested workaround is to split the data into subsets, run **ISAC** as described here for each subset separately and combine the results at the end.

For example, to split a dataset of 200 000 particles in 4 subsets type in the terminal:

```
⇒ e2proc2d.py bdb:Particles#stack_preclean bdb:Particles#stack_preclean_1 --first=0
  --last=50000
⇒ e2proc2d.py bdb:Particles#stack_preclean bdb:Particles#stack_preclean_2
  --first=50001 --last=100000
⇒ e2proc2d.py bdb:Particles#stack_preclean bdb:Particles#stack_preclean_3
  --first=100001 --last=150000
⇒ e2proc2d.py bdb:Particles#stack_preclean bdb:Particles#stack_preclean_4
  --first=150001 --last=200000
```

To combine the resulting *clean* stacks at the end into a single virtual stack, type (one line):

```
e2bdb.py bdb:Particles#stack_preclean1 bdb:Particles#stack_preclean2
bdb:Particles#stack_preclean3 bdb:Particles#stack_preclean4
--makevstack=bdb:Particles#stack_preclean_all
```

We are working very hard to improve the performance of **ISAC** and will get back to you soon with an update.

TIP: *In case your dataset contains populations of particles whose shape differ significantly (e.g. different oligomers, proteins etc.) and which are easily recognizable, we recommend storing the respective class averages and their members in different files and perform the subsequent steps for each subset independently.*

In case you do not have enough time to perform these steps, copy our precalculated **ISAC** results and the resulting *clean* dataset to your **project directory**.

```
cp -r SphireDemoResults/Class2D ./
cp -r SphireDemoResults/Particles ./
```



VIPER

Initial 3D Model - RVIPER

The **RVIPER** program is designed to determine a reproducible and validated initial model at intermediate resolution, using a small subset of class averages produced by **ISAC**.

First, check the number of your **ISAC** class averages.

```
e2iminfo.py Class2D/class_averages.hdf
```

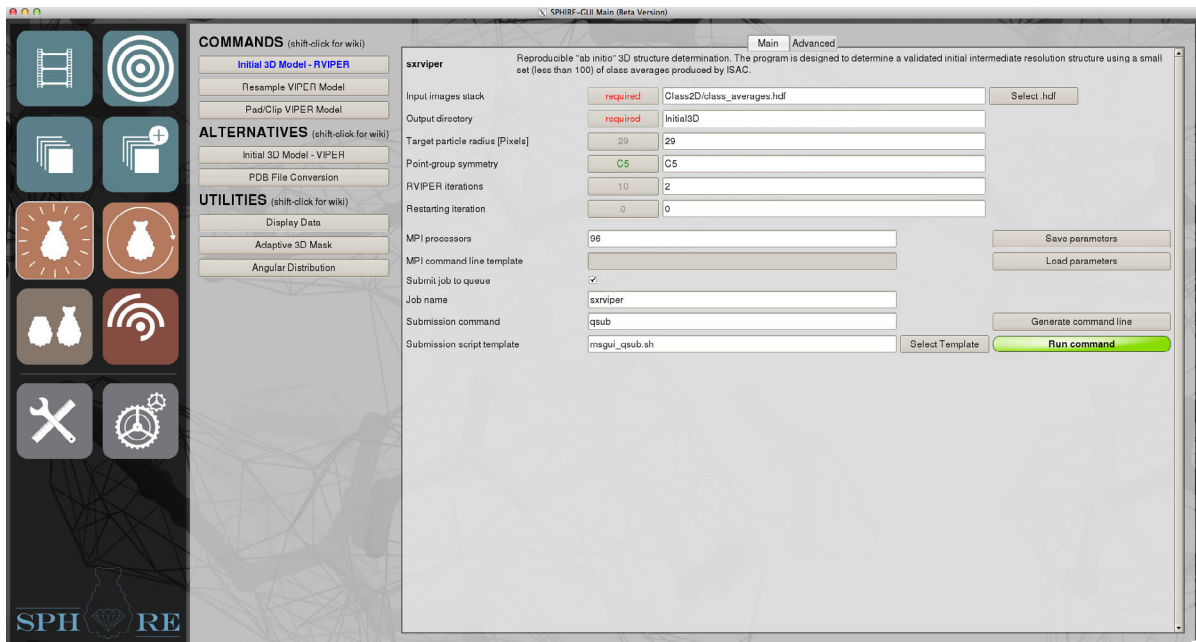
In general about 150 high quality class averages will be sufficient to obtain a validated reconstruction in a reasonable amount of time. Although **RVIPER** would run successfully with more than 150 images, computational time increases exponentially, usually without a significant improvement of the resulting model.

Our **ISAC** output, **class_averages.hdf** contains only about 109 class averages, thus we will continue directly with **RVIPER** without choosing the best classes (the exact number of your class averages can slightly differ due to randomness of **ISAC** clustering process).

TIP: *In order to run **RVIPER** successfully when processing your own data, you will need at least 60 to 80 high quality averages each with about 100 to 200 members for cryo-EM or 100 members for negative stain data. Be sure that the input class averages include as many orientations of the particle, as possible. It will not be possible to calculate a reconstruction from images showing clearly a preferred orientation. If your class averages contain only few members or have a low signal to noise ratio, you might need to use > 150 class averages for **RVIPER**, especially if your structure is asymmetric.*



Go back to the main window of the **SPHIRE GUI** and press the button **VIPER** on the left and then the **Initial 3D Model - RVIPER** button in the middle.



Set the following parameters:

- **Input images stack:** Class2D/class_averages.hdf
- **Output directory:** Initial3D
- **Target particle radius [pixels]:** 29

IMPORTANT: Use the *same* target particle radius as you used for ISAC.

- **Point-group symmetry:** C5

TIP: Use C1 if the symmetry of your particle is not known.

- **RVIPER iterations:** 2

TIP: The optimal number of iterations depends on the starting number of class averages. We usually use these settings: 100 averages: 2 iterations; 400 averages: 5 iterations.

- **Restarting iteration:** 0

Specify the number of processors (for this job we used 96 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. Monitor the progress of the job through the standard output. On our cluster, this **RVIPER** job with 96 processes finished after about 15 min. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

**!Known issue!**

The program will crash if the number of processors exceeds the number of class averages in your input file.

RVIPER (reproducible **VIPER**) will perform multiple **VIPER** runs (ab initio 3D structure determination using a small set of **ISAC** class averages) and thus, calculate multiple initial models. It will perform a stability test and produce a single validated initial model, after the removal of unstable projections.

After the program has finished, two folders can be found in the **Initial3D** output directory: **main001** and **main002**. The number of main folders corresponds to the number of **RVIPER** iterations. The folders named **Run** within each **main** folder contain the results of each independent **VIPER** run. The final reproducible volume can be found in **main002/average_volume.hdf**.

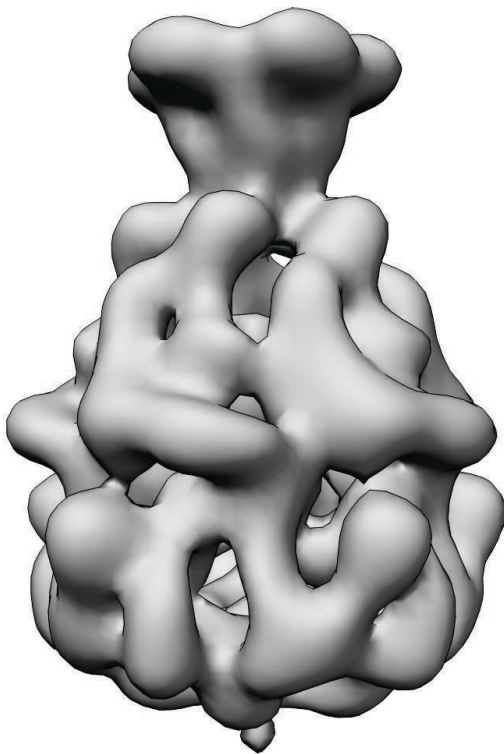
***TIP:** If you process your own data and the internal stability criterion is not met after 10 independent **VIPER** runs, the program will stop (it will not continue to the final iteration) and no average volume will be created. However, we still recommend careful examination of all output reconstructions from the independent runs (e.g. **main001/run000/volf.hdf** and **refvol2.hdf**) as some of them might fit the input data sufficiently well and thus, might be suitable as initial models in the consecutive single particle reconstruction. Nevertheless, it should be emphasized that these models are not validated and in any case, make sure that the chosen model looks reasonable, taking into account how do the class averages look like and whether the model shows enough integrity.*

Please keep in mind that in **cryo-EM** projections, the 3D handedness information is lost. Therefore, **RVIPER** might produce an enantiomer (mirror reconstruction). There is no way to establish correct handedness for low-resolution models short of performing tilt experiments.

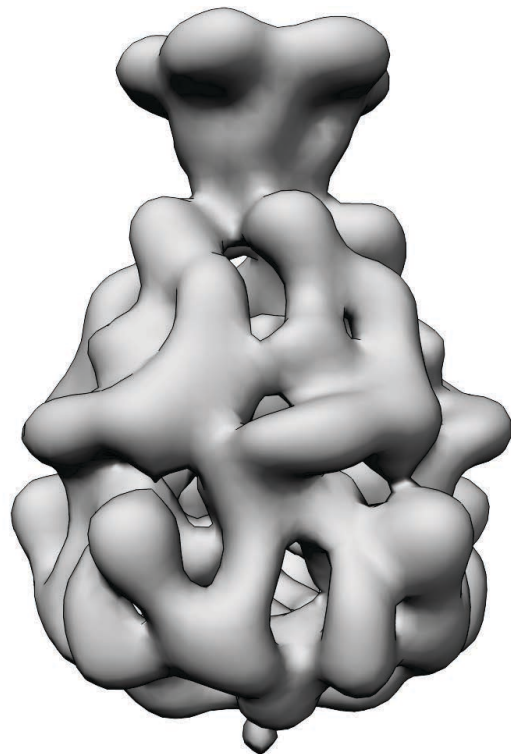
***TIP:** If the handedness of your structure or the structure of a homolog is not known, you should proceed using the just obtained model and once sufficient resolution is accomplished, establish the correct hand based on the visual appearance of secondary structure elements.*



In this case the structure is known and the correct hand can be established already at this point. Display the volume with the molecular graphics program **UCSF Chimera** (Pettersen et al. 2004a)



Correct hand



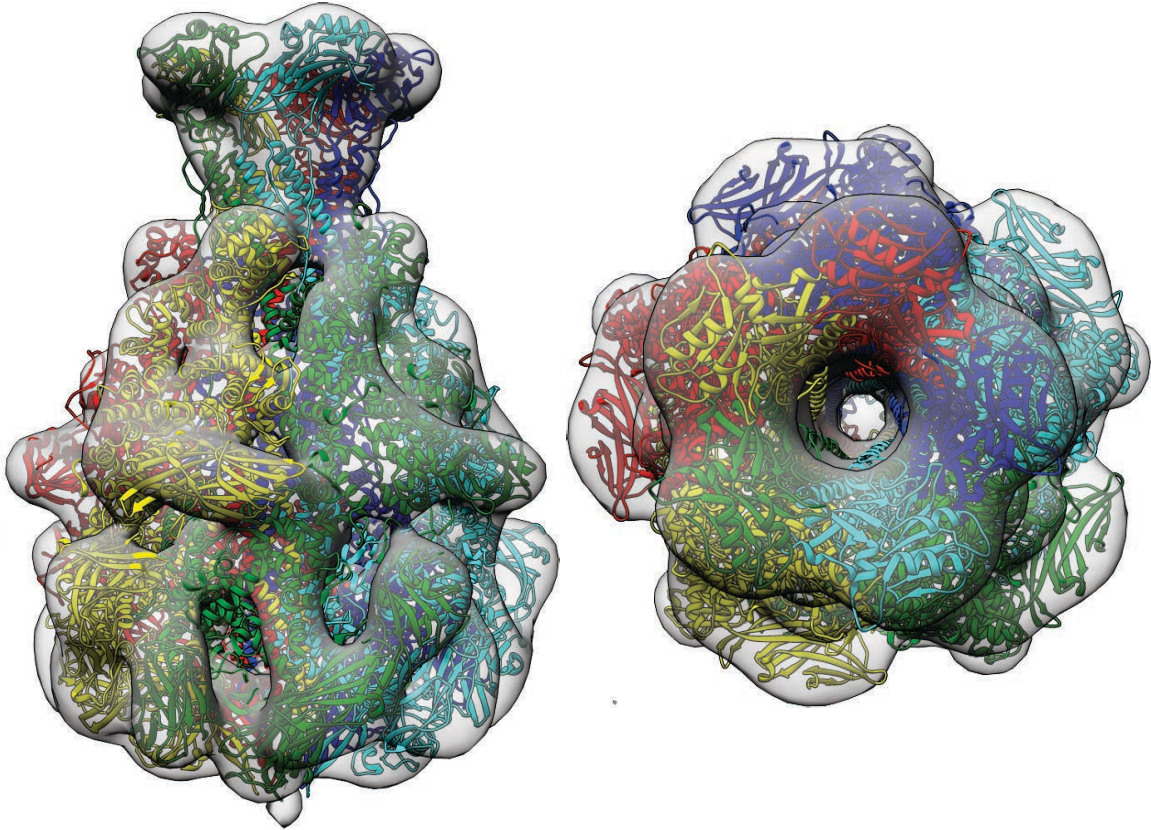
Wrong hand

Compare the output volume with the images above and check if your output volume has the correct handedness. If this is not the case, it will be necessary to mirror the volume along the x-y plane using **EMAN2's e2proc3d.py** command

```
e2proc3d.py Initial3D/main002/average_volume.hdf average_volume_flip.hdf  
--process xform.flip:axis=z
```




To compare the **RVIPER** volume with the available X-ray structure of TcdA1 (3.9 Å, **PDB-ID: 1VW1** (Meusch et al. 2014)) load the volume and the **PDB** file in **UCSF Chimera**, set the voxel size of the volume to 5.7 Å/pixel (see **Resample/Clip VIPER Model** section below for details) and fit the atomic coordinates into the density map using **Fit in Map**. They should show an excellent agreement, even though at the current resolution only the overall envelope of the **EM** model is reliable.



You can also create a *.bild* file with the **Angular Distribution** utility to assess the angular distribution of the class averages used in the **VIPER** reconstruction as described on our **SPHIRE** [wiki](#) page. The file can also be displayed in **UCSF Chimera**.

TIP: *In case **RVIPER** does not produce a final reproducible volume, you can try to activate the **Eliminate disconnected regions** option (advanced parameter) and repeat the procedure. This will remove disconnected pieces from the model, at the threshold automatically estimated from its molecular weight during the iterations. This might be beneficial, in particular, for noisy datasets or elongated particles.*



Resample/Clip VIPER Model

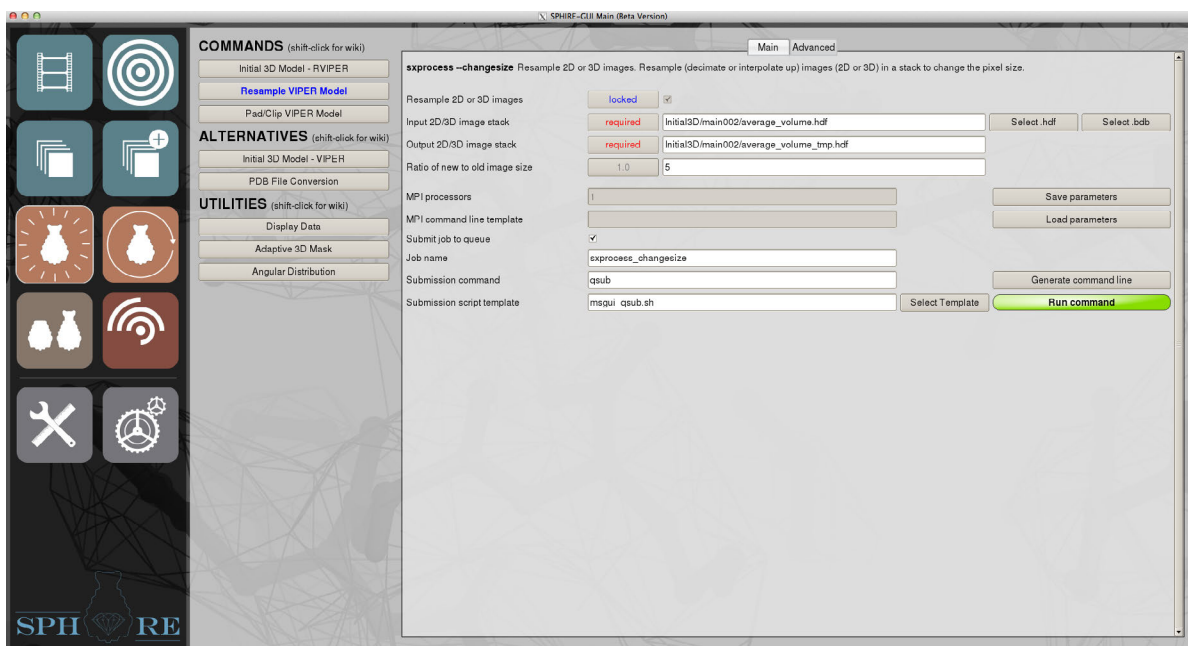
The particles of the dataset have been resized during **ISAC** (in this case to a radius of 29 pixel and box size of 76 pixel). Therefore, the **ISAC** averages and the **VIPER** 3D model have now an increased pixel size of 5.7 Å/pixel. In order to use this model as a reference for the 3D refinement of the dataset with the original pixel size, we have to resize and clip the **VIPER** model in order to match the dimensions of the original particle stack. To check the shrink ratio used to downscale the images before **ISAC** type:

```
cat Class2D/README_shrink_ratio.txt
```

The shrink ratio is 0.2, thus the pixel size of this **VIPER** model is

$$\frac{\text{Original Pixel Size}}{\text{shrink ratio}} = \frac{1.14 \text{ \AA/pixel}}{0.2} = 5.7 \text{ \AA/pixel}.$$

Now press the **Resample VIPER Model** button in the middle of the **SPHIRE GUI**.



Set following parameters:

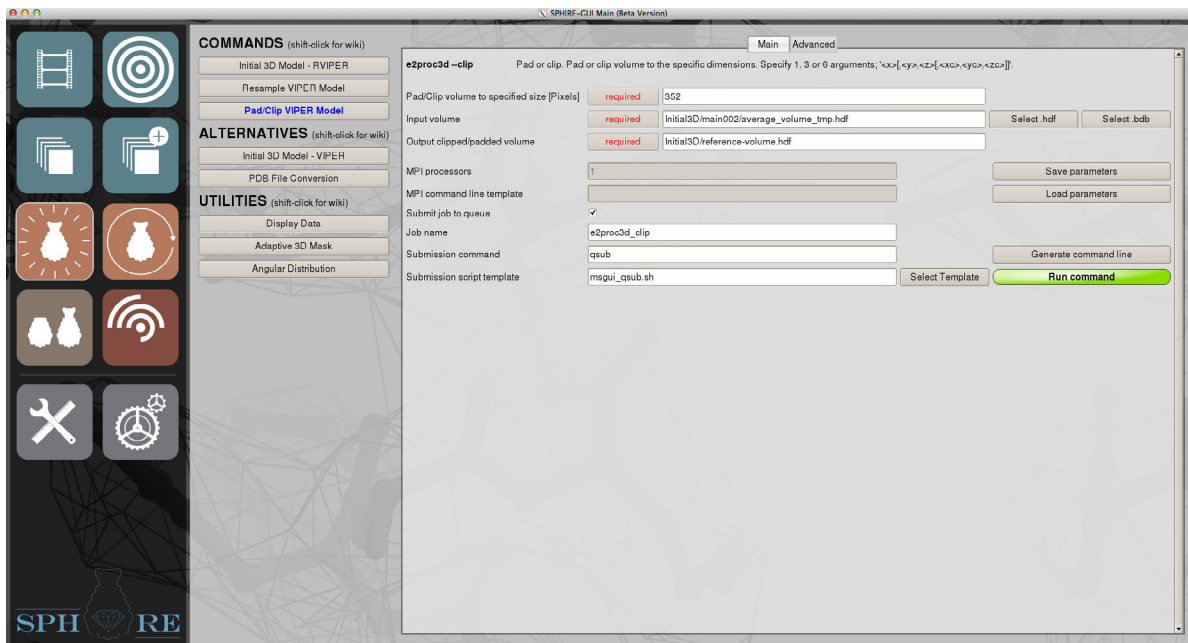
- **Input 2D/3D image stack:** Initial3D/main002/average_volume.hdf
- **Output 2D/3D image stack:** Initial3D/main002/average_volume_tmp.hdf
- **Ratio of new to old image size:** 5



NOTE: $\text{Ratio of new to old image size} = \frac{1}{\text{shrink_ratio}} = \frac{1}{0.2} = 5$

and press the **Run command** button.

Now we will clip this up-scaled volume to the original box size. In the middle of the **SPHIRE GUI** press the **Pad/Clip VIPER Model** button:



Set following parameters:

- **Pad/Clip volume to specified size [pixels]: 352**

NOTE: *Original box size*

- **Input volume:** Initial3D/main002/average_volume_tmp.hdf
- **Output clipped/padded volume:** Initial3D/reference-volume.hdf

TIP: *In case there is a X-ray structure of a homologous protein available and we can expect a high-degree of similarity, one can omit the ab initio structure determination with **VIPER** and use the available atomic coordinates instead. To do so, we use the **PDB File Conversion** utility (Alternatives) to convert the atomic model (typically downloaded from the **PDB** data base) to a simulated density map.*

A detailed description regarding the usage of this program is provided on our [wiki](#) page. If you do not have enough time to perform these steps, you can also copy our precalculated results to your **project directory**.

```
cp -r SphireDemoResults/Initial3D ./
```



MERIDIEN

In the previous steps we removed all *junk* images and produced a *clean* subset of particles that could be aligned and clustered in a stable and reproducible manner by using the powerful **ISAC** approach. Subsequently, from the validated **ISAC** class averages we were able to calculate a reproducible model using **RVIPER**. This gives us confidence that we can now proceed with the structure refinement using the obtained model as a starting reference (**3D Refinement**). During this procedure the half-datasets will be refined quasi-independently in order to minimize noise bias and over-refinement.

Both the structures and resolution (**FSC**) generated at each iteration of the program serve only as internal approximations (e.g. the resolution reported after each iteration is usually underestimated). Usable results are obtained after the refinement has finished, using the *Map Adjustment* utility that takes the so-called unfiltered final half-volumes as input.

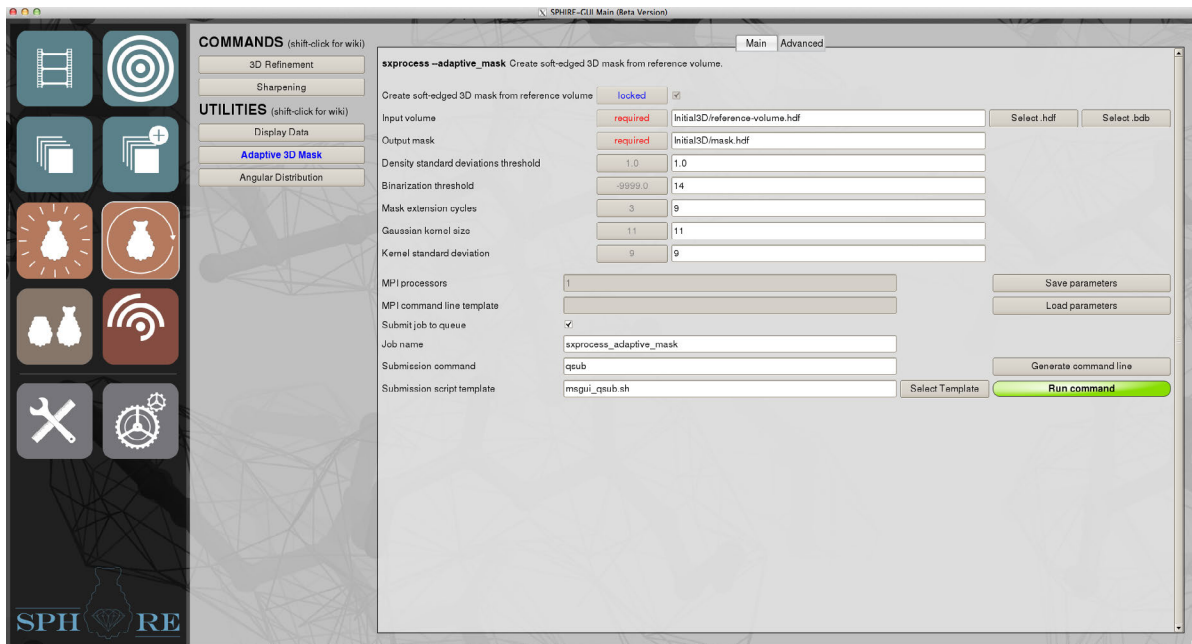
For the high-resolution 3D refinement, we will first have to create a soft 3D mask.

***TIP:** In most cases **RVIPER** will deliver highly reliable and detailed initial model. Therefore, we recommend beginning the refinement immediately with a 3D mask. Structure refinement with a reasonably tight mask proceeds faster and yields better (higher resolution) results. In rare cases when the initial model is very poor or unreliable one should proceed with caution: run first 3D refinement without using 3D mask, then validate the outcome, then create a 3D mask and repeat the refinement.*



Adaptive 3D Mask

Go back to the main window of the **SPHIRE GUI** and press the button **Meridien** on the left and then the **Adaptive 3D Mask** button in the middle (utilities-section):



Adjust following parameters in the respective input fields:

- **Input volume:** Initial3D/reference-volume.hdf

NOTE: *This is the **RVIPER** final result after resampling and clipping*

- **Output mask:** Initial3D/mask.hdf
- **Density standard deviations threshold:** 1.0
- **Binarization threshold:** 14

NOTE: *Threshold for initial binarization of the volume; see the section below for instructions.*

- **Mask extension cycles:** 9

NOTE: *Numbers of cycles to extend this initial binarized volume; see the section below for instructions.*

- **Gaussian kernel size:** 11

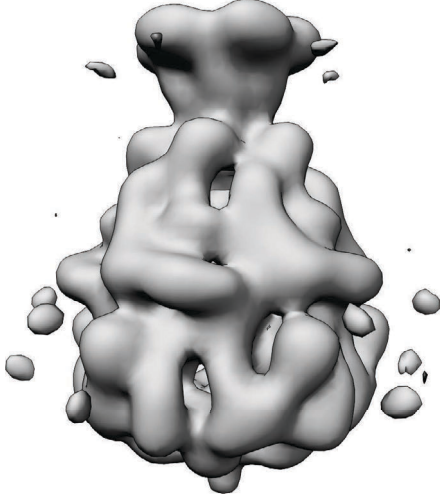
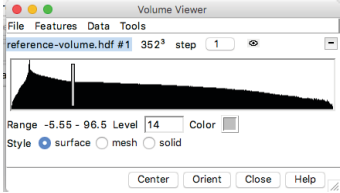


- **Kernel standard deviation: 9**

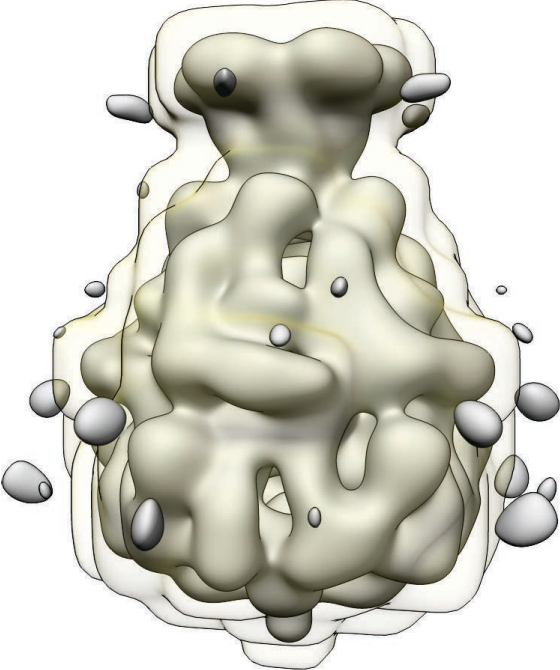
and press the **Run command** button. This process is usually finished after few minutes. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

Density Binarization and Mask Extension

Open the Initial Volume (**Initial3D/reference-volume.hdf**) with **UCSF Chimera**. Use a threshold (**Volume Viewer**) clearly lower than you would normally use when displaying the structure. At this threshold, noise artifacts should be visible but not connected to the molecule density. For this volume, a binarization threshold of 14 is appropriate and this value is required as input for the soft mask creation utility.



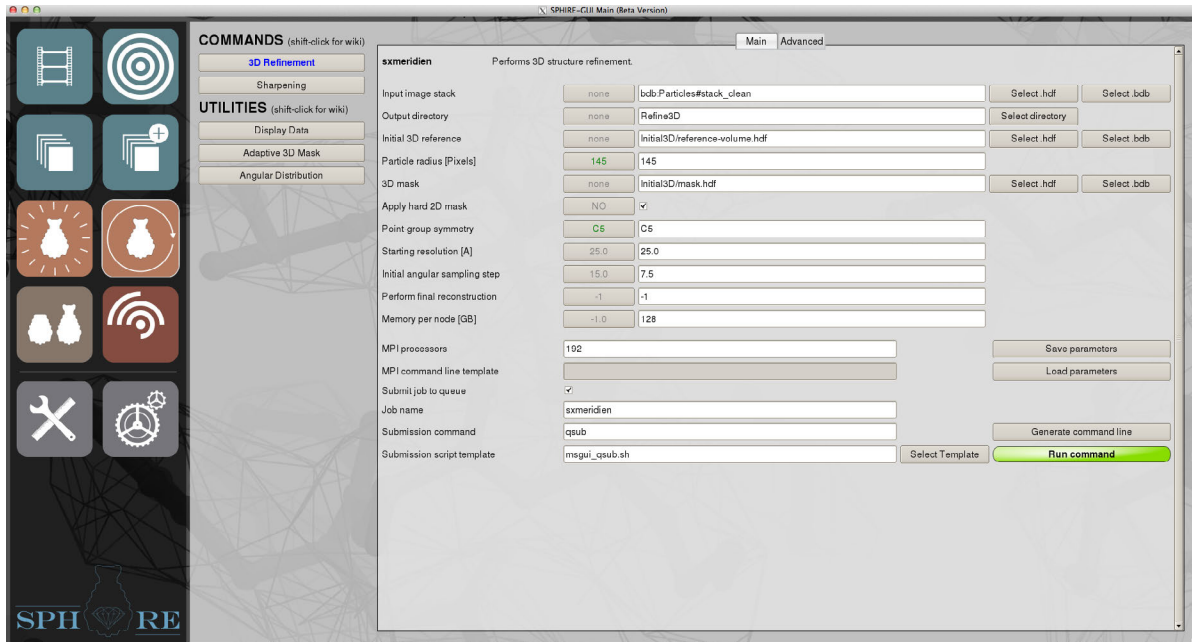
The generated mask should still have the shape of the particle, but it should extend in all directions by a sufficient number of pixels to prevent masking artifacts during 3D refinement. The amount of extension is controlled by the mask extension cycles parameter (set to 9 cycles; you might have to adjust this value). Check the size of the resulting mask relative to the initial volume again using **UCSF Chimera** (we usually display the 3D mask transparent as shown in the figure below). The resulting mask should have sufficient clearance around to the volume and exclude all satellite densities





3D Refinement

3D refinement of the structure is done using original individual particle images. Go back to the main window of the **SPHIRE GUI** and press the button **Meridien** on the left and then the **3D Refinement** button in the middle and enter the following parameters at the respective **GUI** interface:



- **Input image stack:** bdb:Particles#stack_clean

NOTE: Click the **Select .bdb** button and use the file browser to select the unbinned subset of particles, containing the members of the selected **ISAC** class averages

- **Output directory:** Refine3D
- **Initial 3D reference:** Initial3D/reference-volume.hdf

NOTE: Click the **Select .hdf** button and use the file browser to select the reassembled and clipped **RVIPER** volume

TIP: Here you can directly load any type of 3D volume that you want to use as starting reference (e.g. volume produced by another program, structure derived from X-ray crystal model, density from crystal structure, featureless ball in case of high symmetry, etc.; .mrc format is supported). However, keep in mind that the box size and pixel size of this volume have to match the input dataset. If this is not the case, resample and pad/clip the reference volume using the respective **VIPER** utilities as described above.



- **Particle radius [pixels]:** 145

- **3D mask:** Initial3D/mask.hdf

NOTE: *this is the 3D mask produced in the previous step.*

- **Apply hard 2D mask:** Yes
- **Point-group symmetry:** C5
- **Starting resolution [Å]:** 25.0

NOTE: *The initial VIPER model will be low-pass filtered to this limit to avoid model bias.*

TIP: *In case you use a simulated density from a crystal structure as starting reference (see above), you should be more cautious and use a stronger low-pass filter (e.g. 40 Å to 50 Å).*

- **Initial angular sampling step:** 7.5

NOTE: *Usually the default value of 15° is appropriate to create enough projections for the initial global parameter search for almost every asymmetric structure. However, the tutorial structure has C5 symmetry and therefore it is required to adjust this parameter to a lower value (7.5°). Currently, we support only these two starting values. Choosing another value is likely to create unexpected behavior of the program.*

- **Perform final reconstruction:** -1

TIP: *To restart a run that stopped intentionally or unintentionally set this parameter to -1 and specify the same output directory. The program will identify the iteration with the highest resolution and continue from there. If set to 0, the program will directly calculate the final reconstruction from the last iteration round and stop. If you just want to calculate the final reconstruction from a specific iteration you can also specify the iteration number here.*

- **Memory per node [GB]:** 128

NOTE: *Check your cluster specifications. The program has to know how much memory available on each node as it uses “per node” MPI parallelization in many places. Nodes are basic units of a cluster and each node has a number of CPUs (with few exceptions of heterogeneous clusters whose use should be avoided, number of CPUs is the same on each node). While clusters are often characterized by the amount of memory per CPU, here we ask for the total amount of **memory per node** as the program may adjust internally the number of CPUs it is using. For example, a cluster that has 3 GB memory per CPU and 16 CPUs per node has $3 \text{ GB} \cdot 16 = 48 \text{ GB}$ memory per*



node. The default value used by the program is 2 GB per node and the program will determine internally the number of CPUs to arrive at the estimate of total memory. If you enter a wrong value here, the program will most likely crash (if the value is too high) or will be forced to use small memory mode (if the value is too low), which results in performance deterioration.

IMPORTANT: Below are advanced options, please do not change the default values unless you are well informed.

Advanced Parameters:

- **Skip 2D pre-alignment step:** No

***TIP:** By default, the program will do an initial pre-centering of the data. This significantly improves the performance by accelerating the convergence and improving the final resolution. Use this flag if you do not wish to use this step. Note also that setting flag **Read shifts from header** (see below) to yes turns off initial pre-alignment.*

- **Read shifts from header:** No

***TIP:** If the data were already subjected to 3D refinement, 3D orientation parameters are available. In this case one can use the already determined shifts (Euler angles are ignored, as we start from exhaustive search) to significantly accelerate the refinement process and also to improve the final resolution.*

In order to import previously determined parameters into the data file, use command:

```
sxheader.py bdb:yourstack --params=xform.projection  
--import=previous_run_main_directory/main015/params_015.txt
```

- **Initial translation search range [pixels]:** 5
- **Initial translation search step [pixels]:** 2

***NOTE:** Change of the default value of 5 pixel and 2 pixel, respectively, may cause problems. Increasing the initial search range and step may result in unexpectedly long execution time. Decreasing the range and step may cause the program to underperform. Decreasing the range and step is justifiable only in cases when previously determined orientation parameters are available, see option **Read shifts from header** above.*

Specify the number of processors (on our cluster with 24 cores per node, we used 192 processes) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. This step is computationally demanding and the running time increases significantly with the number of particles and the box size. A detailed description regarding the usage of this program is provided on our [wiki](#) page.



The refinement progresses through a sequence of search modes: *INITIAL*, *PRIMARY*, *EXHAUSTIVE*, *RESTRICTED*, and *FINAL*. The main distinction is between *EXHAUSTIVE* searches, in which every data image is matched with all quasi-evenly generated reprojections of the model, and *RESTRICTED*, in which matching is done only within a vicinity of orientation determined in the previous iteration. The program uses elements of *Maximum Likelihood* machinery. **Meridien** is driven by the internal assessment of resolution between two quasi-independent concurrently run refinements. The current resolution is then used to set main parameters of the refinement process (e.g. **angular step**, **shift search range**, **maximum resolution** thus the **window size**) using a set of simple heuristics. In order to prevent premature termination at suboptimal resolution stage, the program also uses heuristical randomization of the refinement process. Thus, repeated runs of the program from the same starting point will yield slightly different results, both orientation parameters and the ultimate resolution. You can monitor the progress of the job through the standard output, which will give you valuable information about the refinement. For example, during every iteration the program will output the time it took to process a chunk of 20 % of data. Based on that one can easily estimate the remaining time for the respective iteration.

```
016-12-14_16:57:03 => ENTERING Xali LOCAL buffered per node only norm Euc Angle:
17772 20.0% 0.1min
```

In order to quickly check the progress of the run and the resolution from iteration to iteration, type:

```
grep ITERATION my_standard_outputfile NOTE: the name of the text file containing the standard output depends on your submission file
```

```
2016-12-14_20:13:08 => ITERATION # 1. Resolution achieved so far: 16 pixels, 25.08A.
Current state: INITIAL, nxinit: 52, delta: 7.5000, xr: 5.0000, ts: 2.0000
2016-12-14_20:14:11 => ITERATION # 2. Resolution achieved so far: 16 pixels, 25.08A.
Current state: PRIMARY, nxinit: 120, delta: 7.5000, xr: 5.0000, ts: 2.0000
2016-12-14_20:18:18 => ITERATION # 3. Resolution achieved so far: 33 pixels, 12.16A.
Current state: PRIMARY, nxinit: 96, delta: 7.5000, xr: 5.0000, ts: 2.0000
2016-12-14_20:21:05 => ITERATION # 4. Resolution achieved so far: 35 pixels, 11.47A.
Current state: PRIMARY, nxinit: 102, delta: 7.5000, xr: 5.0000, ts: 2.0000
```

The printout contains a sequence of line for each iteration, that specifies the time the iteration started, its number, the resolution achieved in the previous iteration, both in Fourier pixels and in Å as read from **FSC@0.5**. Note however that the **FSC** computed during iterations and thus the resolution printed are *not* the ultimate results but merely approximations calculated quickly for expediency to guide the program and used for adjustment of its parameters.

```
2016-12-14_20:33:19 => ITERATION # 8. Resolution achieved so far: 46 pixels, 8.72A.
Current state: RESTRICTED, nxinit: 126, delta: 3.7500, xr: 2.9130, ts: 0.9710
```

For example, this line for **ITERATION 8** also informs us that at this resolution restricted searches will be performed. Finally, the current parameters are: **window size (nxinit)** 126, **angular step (delta)** 3.75°, **translational search range (xr)** 2.913 0 pixel and **step (ts)** 0.971 0 pixel. Once the program converged, it will output a statement and then calculate the final reconstructions:



```
Refinement convergence criteria A are reached
The current state is RESTRICTED
3-D refinement converged
The best solution is in the directory main025
Computing 3-D reconstruction using the best solution3
```

The final half-volumes are stored in **Refine3D: vol_0_unfil.hdf** and **vol_1_unfil.hdf**.

!Known issue in beta-release!

The last iteration is memory intensive and the run might stop if sufficient memory is not available. In case this happens, try to reduce the number of processes per node and restart the program as described above. The program will then automatically continue from the last iteration. In case the program does not finalize even if you use one process per node, the only alternative would be to downscale your particles (and your reference volume) to a lower pixel size and re-run the program in a different output folder. This can be done with the following command:

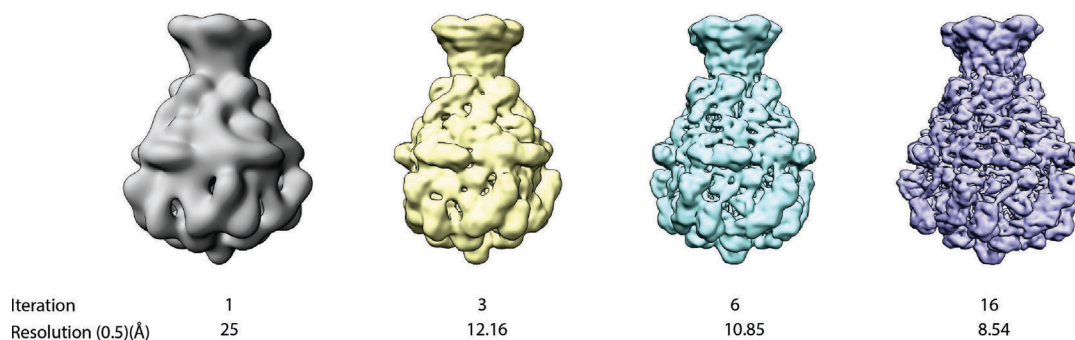
```
e2proc2d.py bdb:mystack bdb:mybinnedstack --scale=(scaling factor) --clip=(new box size)
```

On our cluster with 128 GB per node we had to reduce the number of processes per node from 24 to 6 for reconstructions of datasets with a box-size of 512 pixel, but binning was not necessary.

We are working very hard to reduce the memory requirements during the last iteration and optimized parameter control through the **GUI** and will get back to you soon with an update.

The final orientation parameters are stored in **Refine3D/final_params.txt**. You can load this file to the **Angular Distribution** utility to produce a *.bild* file with the angular distribution of your particles, as described on the [wiki](#) page. You can load one of the volumes together with the *.bild* file in **UCSF Chimera**. The resulting *VRML model* will depict the orientations of your particles on the asymmetric unit of your volume. Note these are only the most probable orientations assigned to data. In agreement with *Maximum Likelihood* methodology even at the convergence each projection may have more than one set of orientation parameters assigned to it with corresponding probabilities. The full information about all orientation parameters is stored in each `main***/oldparamstructure`. However, at this point we do not have any utilities that would allow user to examine this information in a compact form.

TIP: *In order to monitor the progress of the refinement we usually also load the half-volumes of each iteration in UCSF Chimera. The nominal improvement in resolution should agree well with the visual appearance of the maps. Note that these maps are not meant for interpretation, they are merely approximations internally used by the program. Thus, one should not attach much weight to their appearance and particularly to the apparent lack of details. Only the so-called “final” maps can be properly interpreted (see below for instructions how to compute them).*



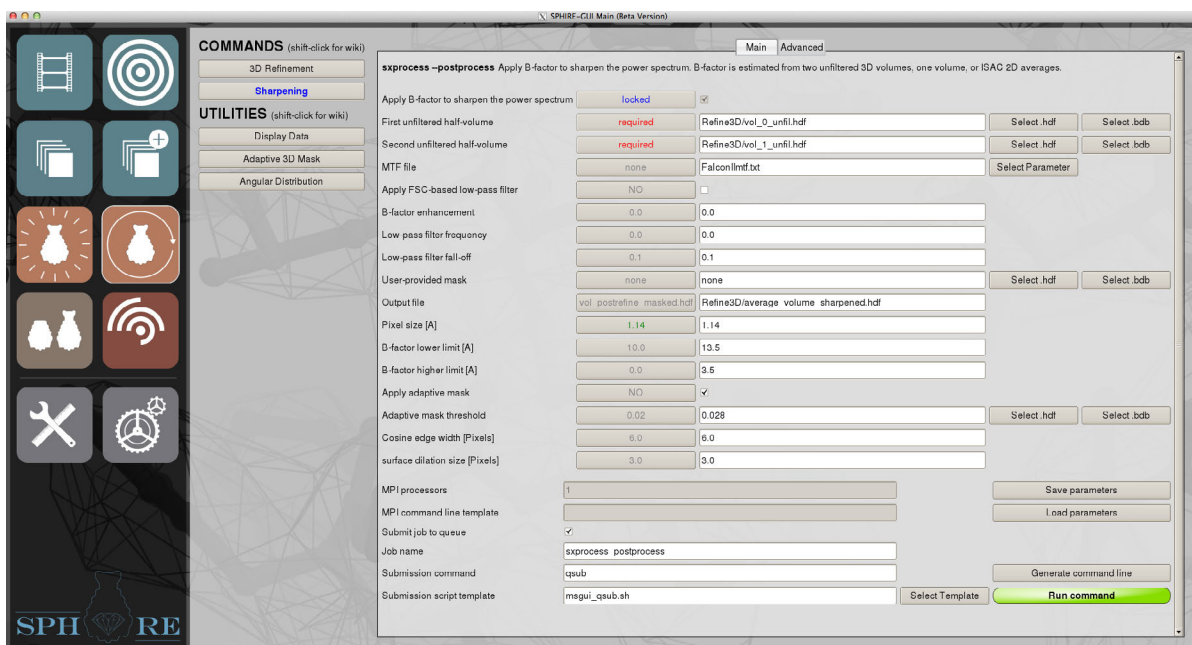


On our cluster this **MERIDIEN** job with 192 processes finished after ~2.5 h. If you do not have enough time to perform the 3D refinement, you can also copy our precalculated results to your **project directory**

```
cp -r SphireDemoResults/Refine3D ./
```

Sharpening

The purpose of this step is to compute final reconstruction and report the final resolution. The command will calculate the **FSC** between two masked maps, merge two half-volumes, apply **MTF** correction, adjust power spectrum by **FSC**, estimate the B-factor from the merged volume, adjust the power spectrum of the merged volume by B-factor (either as defined by the user or estimated automatically by the program), and apply low-pass filter (either at a cut-off chosen by user or at the estimated resolution). It will also create a 3D soft mask. The command reports resolution according to **FSC@0.5** and **FSC@0.143**. In this case, we will apply a low pass filter to the average volume at its final resolution, as calculated from the masked **FSC@0.143**.



Go back to the main window of the **SPHIRE GUI** and press the button **Sharpening** button in the middle.

Adjust following parameters:

- **First unfiltered half-volume:** Refine3D/vol_0_unfil.hdf



NOTE: Click the **Select .hdf** button and use the file browser to select the first final half-volume.

- **Second unfiltered half-volume:** Refine3D/vol_1_unfil.hdf

NOTE: Click the **Select .hdf** button and use the file browser to select the second final half-volume.

- **MTF File:** FalconIImtf.txt

NOTE: Click the **Select Parameter** button and use the file browser to select the **MTF** file that comes along with our test data set for the Falcon II detector at 300 kV, stored in your **project directory**.

- **Apply FSC-based low-pass filter:** No

- **B-Factor enhancement:** 0

NOTE: Set to 0, in order to estimate B-factor automatically and apply B-Factor. Alternative Options:

–1: B-factor is not applied.

e.g. 128: a user specified B-factor of -128 \AA^2 will be applied.

- **Low-pass filter frequency:** 0

NOTE: Frequency used to filter final map. Set to 0 to filter the map according to **FSC@0.143**. Alternative options:

e.g. 5.8: low pass filter to 5.8 \AA .

e.g. 0.2: low pass filter to 0.2 1/pixel

- **Low-pass filter fall-off:** 0.1

- **User provided mask:** none

NOTE: We will calculate a new mask with the help of the adaptive mask during the **Sharpening** procedure.

- **Output file:** Refine3D/average_volume_sharpened.hdf

- **Pixel size [A]:** 1.14

- **B-Factor lower limit [A]:** 13.5

- **B-Factor higher limit [A]:** 3.5

NOTE: Select frequency range for the automated calculation of B-factor.

TIP: As a rule of thumb: B-factor higher limit = expected resolution; B-Factor lower limit = B-factor higher limit + 10

- **Apply adaptive mask:** Yes

NOTE: Activate this option to create a 3D mask for the **FSC** and the merged volume.



- **Adaptive mask threshold:** 0.028

NOTE: Open one of the half volumes with UCSF Chimera and adjust a proper threshold value. The density map should not show any disconnected regions.

TIP: 0.02 to 0.03 are usually good values.

- **Cosine edge width [pixels]:** 6.0
- **surface dilation size [pixels]:** 3.0

and press the **Run command** button. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

Monitor the progress of the **Sharpening** job at the terminal through the logfile **log.txt**:

```
tail -f log.txt
```

The output looks like this:

```
2016-12-15_09:31:36 => ----->>processing<<-----
2016-12-15_09:31:36 => 3-D refinement postprocess
2016-12-15_09:31:36 => The first input volume: Refine3D/vol_0_unfil.hdf
2016-12-15_09:31:36 => The second input volume: Refine3D/vol_1_unfil.hdf
2016-12-15_09:31:37 => starts creating surface mask, and wait...
2016-12-15_09:37:22 => MTF correction is applied
2016-12-15_09:37:22 => MTF file is FalconIImtf.txt
2016-12-15_09:37:47 => Similiarity between the fitted line and 1-D rotationally
average power spectrum within [30, 116] is 0.655850
2016-12-15_09:37:47 => The slope is -5.070000 Angstrom^2
2016-12-15_09:38:01 => ----- >>Summary<<-----
2016-12-15_09:38:01 => Resolution at criteria 0.143 is 3.401000 Angstrom
2016-12-15_09:38:01 => Resolution at criteria 0.5 is 4.053000 Angstrom
2016-12-15_09:38:01 => B-factor is -20.300000 Angstrom^2
2016-12-15_09:38:01 => FSC curve is saved in fsc.txt
2016-12-15_09:38:01 => Final processed volume is average_volume_sharpened.hdf
2016-12-15_09:38:01 => guinierlines in logscale are saved in guinierlines.txt
2016-12-15_09:38:01 => Top hat low-pass filter is applied to cut off high frequencies from
resolution 1.0/3.400000 Angstrom
```

On our Linux cluster, with a single process, this job finished after about 8 min.

However, if you do not have enough time to wait for the results, you can find the precalculated results for this step in the folder **SphireDemoResults/Sharpening-after-meridien**

According to the masked **FSC@0.143** the final volume has a resolution of 3.4 Å. A B-Factor of -20.3 \AA^2 and a low-pass filter at 3.4 Å have been applied to the volume. In the **project directory** you can now find the following files:

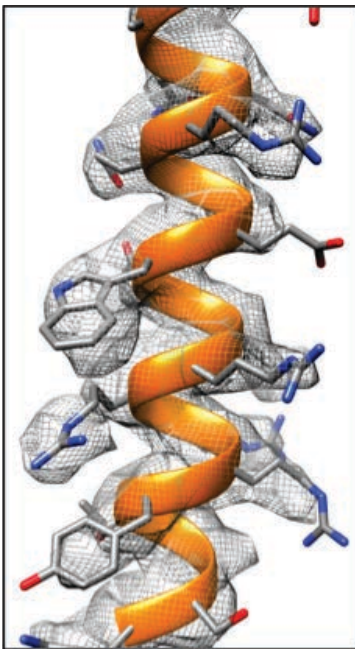
- **vol_postprocess_nomask.hdf:** The sharpened unmasked final volume
- **average_volume_sharpened.hdf:** The sharpened and masked final volume
- **fsc.txt:** The **FSC** of masked volume
- **vol_adaptive_mask.hdf:** The adaptive 3D mask



Display the final volume with **UCSF Chimera** and also plot and check the FSC. Compare again the density with the available X-ray structure.

!Known issue in beta-release!

Unfortunately, in the present version of **Sharpening** phase randomization of the two half-reconstructions is not performed and therefore, in case you use a rather *tight mask*, the resolution might be affected by the convolution effects of masking. Thus, especially beginners in the field should be cautious and avoid over-masking! Check your **FSC** carefully and create a larger mask in case you obtain strange B-factor values and/or observe strange peaks or raises at the high frequencies of your **FSC**. Such issues are nicely described in (Penczek 2010). In case you want to measure the local resolution of a specific area of your volume, instead of using local masks to calculate the respective **FSC**, use our local resolution and filtering approach instead (see below). You should always visually inspect the resulting map and **FSC** carefully and confirm that the features of the density agree with the nominal resolution (e.g. a high resolution map should show clearly discernible side chains). We are working very hard to introduce this feature as soon as possible and will get back to you soon with an update.



Congratulations!
You produced your first near-atomic resolution reconstruction
with
SPHIRE



SORT3D

3D Variability

When the 3D structure refinement is completed, we calculate a 3D variability map using the **EM** data and their orientation parameters. This will allow us to detect poorly resolved regions of the map that might be due to conformational changes, ligand binding and/or compositional heterogeneity. In general, regions with higher variability are less reliable and the data may require further analysis. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

TIP: *Although regions with higher variance are most often less well resolved, this step is not being performed to estimate the local resolution, but is rather used to guide the subsequent step of 3D classification. **SPHIRE** includes a dedicated tool for the local resolution estimation (**LocalRes**), which is usually performed at the end of the workflow, as described below.*

NOTE: *Ideally, we would want to compute a 3D variance map using the projection data. However, it is not immediately apparent that this is mathematically possible as the data is given in form of projections. Moreover, methods suggested in the literature call for massive calculation. Therefore, in **SPHIRE** we settle for a good approximation of the variance map, that we termed variability map (**Loerke j. et al.; manuscript in preparation**).*

First we will import the projection parameters of the final refinement round (with the highest resolution) to the header of our *clean* particle stack.

```
sxheader.py bdb:Particles#stack_clean --params=xform.projection
--import=Refine3D/final_params.txt
```

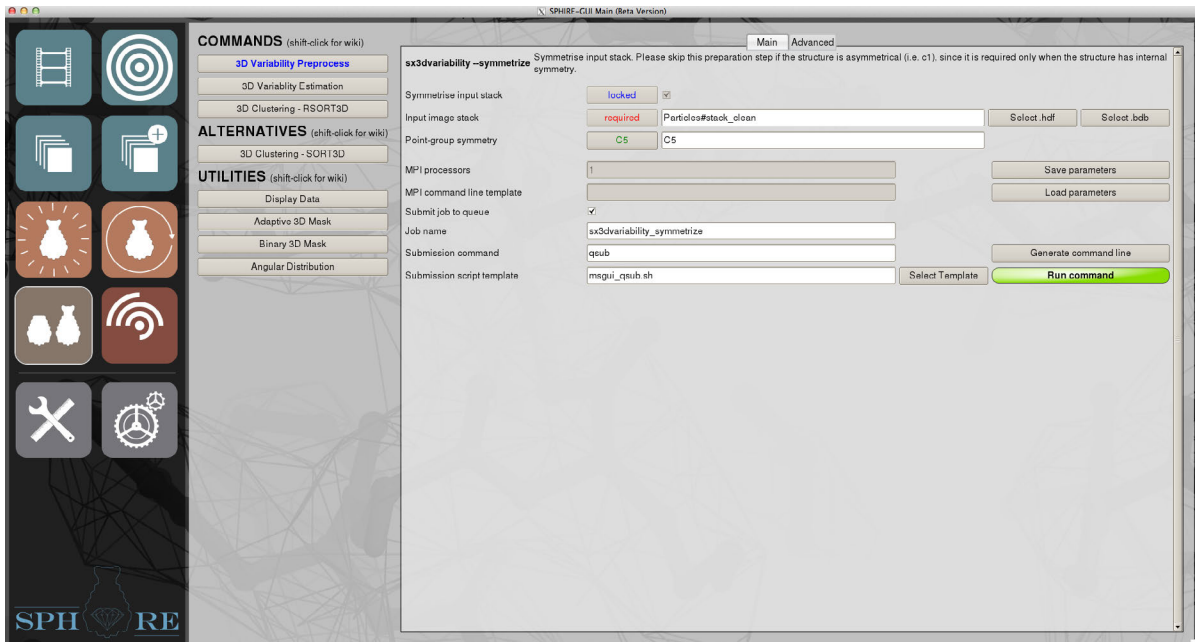
NOTE: *During the 3D refinement, **Meridien** outputs and uses multiple projection directions (probability weighted) for each particle. However, for the purpose of variability analysis, we use only the highest probability assignment for each particle.*

In the next step, we will perform a symmetrization of this dataset in order to cover the entire 3D angular space for the subsequent calculation of the 3D variability field.

TIP: *Skip this step if you process an asymmetric particle.*



Go back to the main window of the **SPHIRE GUI** and press the button **Sort3D** on the left and then the **3D Variability Preprocess** button in the middle:



Adjust following parameters and press the **Run command** button.

- **Input image stack:** Particles#stack_clean

NOTE: Click the **Select .bdb** button and use the file browser to select the stack you used as input for the high-resolution refinement.

- **Point-group symmetry:** C5

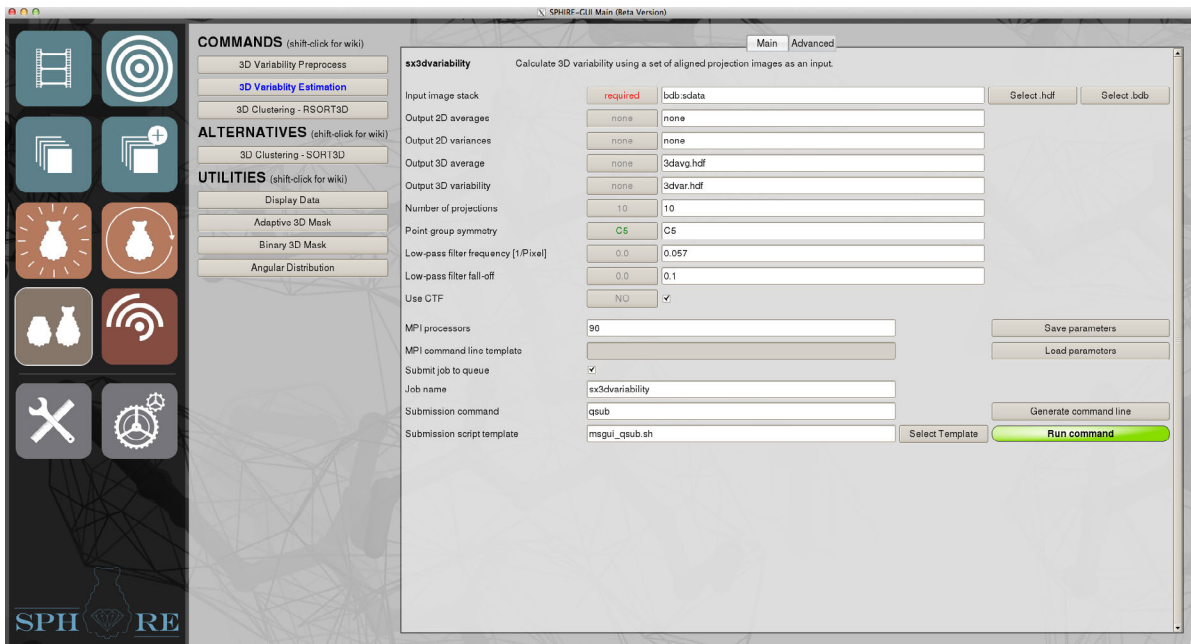
A *symmetrized* dataset named **bdb:sdata** will then be stored directly in your project directory. Use **EMAN2's e2iminfo.py** to check the number of particles in this dataset.

```
e2iminfo.py bdb:sdata
```

After symmetrization, this stack should contain 5 times more projections than your input dataset ($5 \times 7\,350 = 36\,750$ particles). This dataset is a so-called virtual stack, i.e. only header information was duplicated (here 5 times), while the original data remained unmodified, so no excessive disk space was consumed.



Go back to the main window of the **SPHIRE GUI** and press the button **3D Variability Estimation** in the middle:



Adjust following parameters:

- **Input image stack:** bdb:sdata

*NOTE: Click the **Select .bdb** button and use the file browser to select the symmetrized stack located in your project directory.*

- **Output 2d averages:** none
- **Output 2d variances:** none

NOTE: After the successful 3D refinement, the projection parameters for each particle are now available. Thus, for each particle the program will find and borrow its neighbors (the size of the angular neighborhood is user-defined) and will compute average and variance using this grouping. During this tutorial, we will not output these 2D averages and variances, as they consume large amount of disc space and are not particularly interesting per se, but the program will use them to calculate a 3D average and variability volume (see below).

*TIP: It might be worthwhile to check, especially for a high resolution refinement, the high-resolution features of these 2D averages and compare them with the **ISAC** class averages. However, in case you did not use a **VIPER** density map as initial model for the refinement, be aware that these averages are not the result of a validated and reference-free approach!*

- **Output 3d average:** 3davg.hdf



NOTE: 3D reconstruction computed from these 2D averages.

- **Output 3d variability:** 3dvar.hdf

NOTE: 3D variability volume.

- **Number of projections:** 10

TIP: The larger the number, the less noisy the variability map but the lower the resolution. When you process your own data, you have to consider here the size of your dataset and the type of heterogeneity you expect and want to visualize and adjust this value accordingly.

- **Point Group symmetry:** C5
- **Low-pass filter frequency [1/Pixel]:** 0.057
- **Low-pass filter fall-off:** 0.1

NOTE: We will apply a low-pass filter of 20 Å (Pixel size / Resolution = 1.14 Å/pixel / 20 Å = 0.057 1/pixel) to the 2D-averages, before the variability calculation, in order to suppress noise.

- **Use CTF:** Yes

TIP: Deactivate this flag if you process negative-stain data. In this case, such an analysis might be very challenging and should be performed only if you expect compositional heterogeneity of large molecular weight components or very large and well-defined conformational changes. In any case, consider analyzing the variability only from particles extracted from areas of similar stain thickness.

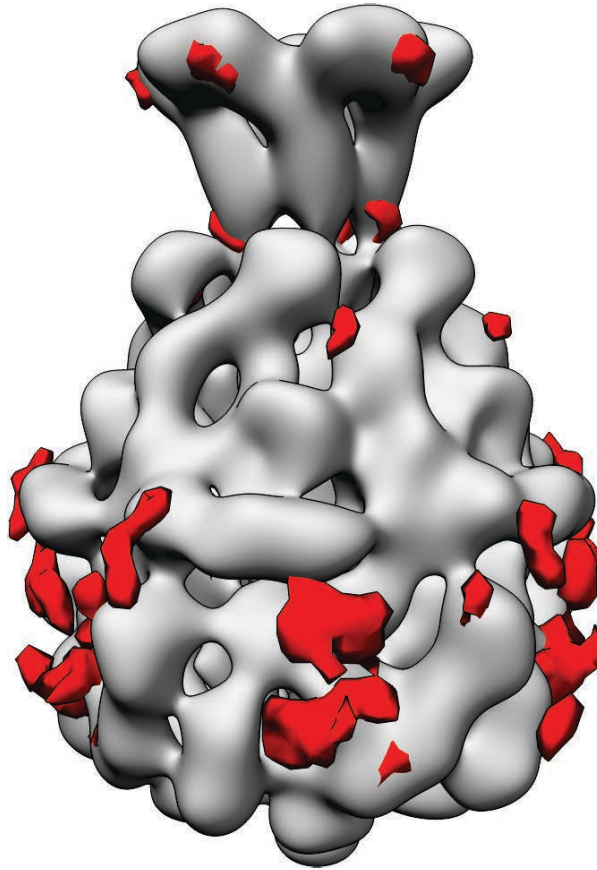
Specify the number of processors (for this job we used 96 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. This process should be finished after few minutes.

You can find now the 3D average and variability volumes (**3davg.hdf**, **3dvar.hdf**) in the **project directory**. Display both volumes with the molecular graphics program **UCSF Chimera**.

Otherwise, if you need to save time, you can copy our precalculated variance and average maps to your **project directory** and display those. In this case, type:

```
cp SphireDemoResults/3d*.hdf ./
```

The first impression is that the variability volume looks is rather noisy. Nevertheless, in order to identify regions of the particle that might have higher variability we will try to simplify the display of the map. For this purpose, set the threshold of the variability map to a rather high value (0.0007) and remove smaller blobs using the **Hide Dust** tool (size 11.6) of the **UCSF Chimera** software package (see image, below). The average volume and the simplified variability map are shown in gray and red, respectively.



Although the simplified variability map is also very noisy, it is apparent that the lower part of the particle is structurally more flexible than the upper part. This agrees well with the conclusion drawn from the analysis of the available X-ray structure of TcdA1. The lower region contains the receptor binding domains, which were slightly less well resolved in the X-ray map and which are expected to be structurally more flexible. Interestingly, the area with the highest variability is observed at the expected location of the His-tagged N'-terminal α -helix, which could not be resolved in the crystal structure. In summary, our tutorial particle can be considered rigid and the 3D variability analysis yields rather limited information. Nevertheless, we will continue in the next step with a 3D heterogeneity analysis, in order to exploit the possibility to identify a more homogeneous subpopulation of particles.



3D Clustering - RSORT3D

We will now perform a 3D classification using the **RSORT3D** approach. The program repetitively applies equal size *K-means* (*EQK-means*) and *K-means* clustering methods to sort the particles. It begins with independent N *EQK-means* runs to accomplish initial division of the dataset into equal size groups, which is done to prevent generation of implausibly large or negligibly small groups and then applies a two ways comparison between the outcomes of the independent runs to identify particles that were reproducibly (stably) assigned to groups. Next *K-means* clustering is applied to the stable particles. These two main steps are applied repeatedly to arrive at a number of particle clusters, which due to reproducibility tests performed can be considered validated. Sufficiently large particle clusters can be used for further 3D refinement with **Meridien**.

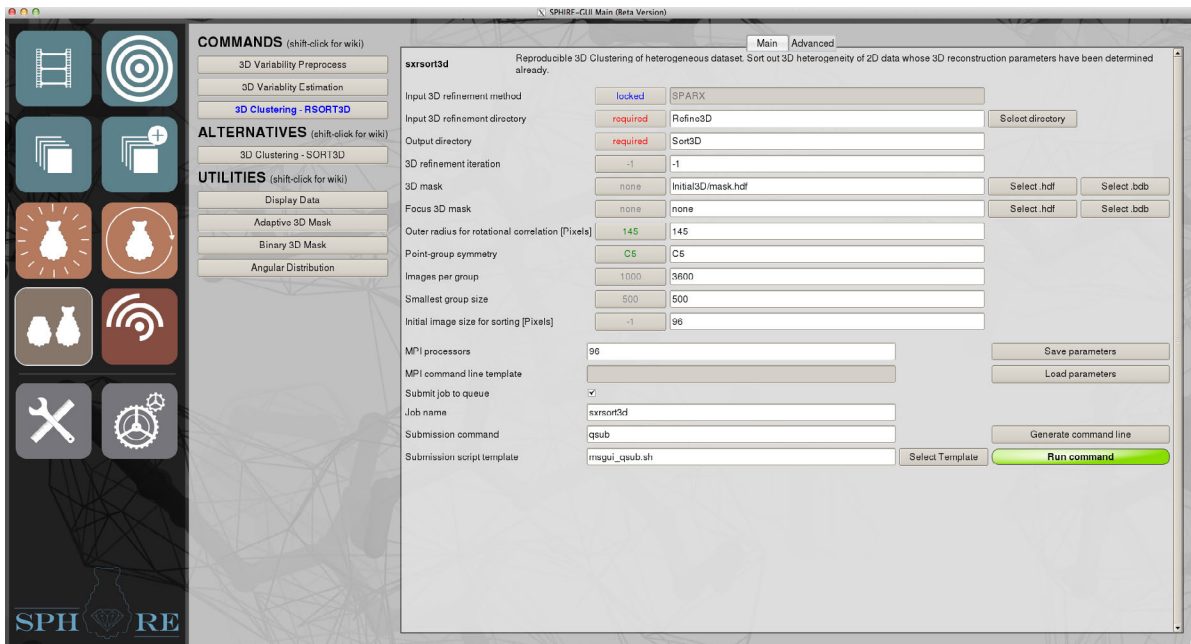
Should the resolution of such a subset improve, as compared to the resolution of the full set, one would obtain a clear indication that the sorting improved homogeneity of the subset and thus the resulting structure has reduced structural variability. This step is computationally expensive and the running time increases significantly with number of particles and classes. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

TIP: *In other software packages, 3D classification is sometimes performed to remove junk particles. In SPHIRE most of these particles have already been eliminated during the ISAC approach in 2D. Thus, we recommend performing the subsequent computationally intensive step of 3D sorting, only if your reconstruction and the 3D variability analysis indicate heterogeneity in your data set.*

NOTE: *All proteins undergo internal motion in solution and all datasets are heterogeneous, albeit to a different extent. The uniformly variability map of this dataset and the high resolution achieved in the 3D refinement from a rather low number of particles, suggest that our tutorial particle is rather rigid. Nevertheless, we will continue with a 3D heterogeneity analysis to demonstrate the usage of the program. If you have time and want to test the performance of the program on a particle showing defined conformational states, we recommend in addition to download a well-characterized dataset from the **EMPIAR** database and follow the procedures described in this tutorial.*



Go back to the main window of the **SPHIRE GUI** and press the button **3D Clustering – RSORT3D** in the middle and adjust following parameters:



- **Input 3D refinement directory:** Refine3D

*NOTE: Click the **Select directory** button and use the file browser to select the directory of your 3D Refinement*

- **Output directory:** Sort3D
- **3D refinement iteration:** –1

*NOTE: The program will automatically search in the **Refinement directory** produced by **Meridien** for the iteration with the highest resolution and start the 3D classification with corresponding 3D projection parameters. Alternatively, you can also specify here an iteration number*

- **3D mask:** Initial3D/mask.hdf

*NOTE: This is the global mask used for the sorting procedure. It has to be soft-edged (Gaussian). Click the **Select .hdf** button and use the file browser to select the soft 3D mask used during the refinement. We prefer to use a rather large mask here, because the 3D variability analysis revealed heterogeneity in regions not resolved in our refined volume. In general, avoid using tight masks, in order to avoid introduction of a bias to the sorting procedure*

*TIP: When you process your own data, display the 3D variability map, your refined volume and the 3D mask using **UCSF Chimera**, and confirm that the soft 3D mask includes the protein completely (even dynamic components that are not resolved in the refined volume, but clearly present in the variability map)*



- **Focus 3D mask:** none

NOTE: This is a 3D mask that one can use to focus the 3D sorting procedure on a particular region of the structure. In contrast to the global mask, this has to be a sharp-edge binary mask. Since our 3D variability map did not reveal a local heterogeneity (e.g. ligand density or a flexible domain), we will not focus our 3D classification on a specific region of our protein by using such a mask.

TIP: In case the variability analysis indicates a clear local variability in your density when you process your own data, binarize the 3D variability map to create a focus mask for the 3D clustering approach, using the Binary 3D mask Utility.

- **Images per group:** 3 600

NOTE: The expected number of 3D groups is the total number of particles divided by this value. Our dataset contains about 7 300 particles, thus by setting this parameter to 3 600, the starting number of groups for the 3D sorting will be:

$$\frac{\text{Total Number of Particles}}{\text{Particles per group}} = \frac{7300}{3600} \approx 2.$$

However, the final number of 3D averages outputted by the program depends on the heterogeneity of the dataset and may differ from the user-defined starting number of groups.

TIP: Adjust this number, according to the size of your dataset. The computational time will increase significantly with increasing number of class averages. While the program attempts to generate groups with the requested numbers of images per group, ultimate group sizes are usually lower. It is difficult to provide a simple “rule of thumb” for the establishment of the proper desired number of particles per group, as it depends on the dataset, the Signal to noise ratio (SNR), the degree of heterogeneity and the total number of available particles. For example, in case you expect a certain number of functional states, a good choice is the total number of particles divided by the expected number of functional states, preferably slightly larger. In addition, groups should be large enough to yield reconstructed maps with a sufficiently high SNR and thus resolution, so one can reliably analyze the outcome. In case of asymmetric complexes, 15 000 particles per group is usually a good number. You might have to increase this number if your data set is rather noisy. For the tutorial dataset we are able to use fewer particles per group (3 600), due to the C5 symmetry of the particle.

- **Smallest group size:** 500

NOTE: The program disregards the groups that have fewer members than this value. 3D sorting is computationally expensive and the running time increases significantly with number of particles and 3D groups. In order to speed up the process, we usually reduce in addition the size of the particles.



TIP: This value depends on the *SNR* of the dataset and point-group symmetry of the complex.

- **Initial image size for sorting (pixels): 96**

NOTE: This parameter is used to downsample (bin) the clipped images to a larger pixel size. Working on binned images can speed up the process significantly. You can calculate the new pixel size as following:

$$\frac{352 \text{ pixel}}{96 \text{ pixel}} \cdot 1.14 \text{ \AA/pixel} = 4.18 \text{ \AA/pixel}$$

The Nyquist in this case is at about 8.36 Å, thus this pixel size is sufficient to allow depiction of conformational changes at secondary structure level of detail.

TIP: The right pixel size depends on the question you aim to answer. For example, in case you want to analyze compositional heterogeneity, depending on the size of the respective component, you might be able to use an even larger pixel size. On the other hand, if you want to analyze for example, the conformation of a ligand using focused classification mode, you might have to use a much smaller pixel size, otherwise the sorted volumes will not contain enough information, to allow this kind of analysis.

Specify the number of processors (for this job we used 96 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button.

!Known issue in beta-release!

Although the program runs stable for our test-datasets, we want to emphasize, that the current MPI-implementation of **RSORT3D** is still under optimization. We expect a significant speedup of the program and improved scalability in the near future and will get back to you soon with an update.

You can monitor the progress of the job, though the logfile **log.txt** stored in the output folder.

```
tail -f Sort3d/log.txt
```

Otherwise, if you need to save time, you can copy precalculated results to your **project directory** and continue with those. In this case, type:

```
cp -r SphireDemoResults/Sort3D ./
```

On our cluster this job finished after about 3 h. The last part of the log file provides an overview about the final number of groups, number of particles in each group and the resolution of the reconstruction for each group at **FSC@0.5** and **FSC@0.143**.

```
2016-12-15_13:11:11 => Final sort3d summary
2016-12-15_13:11:11 => group 0 number of images: 4843 FSC05 0.479167 FSC143 0.489583
2016-12-15_13:11:11 => group 1 number of images: 2522 FSC05 0.395833 FSC143 0.489583
```

NOTE: The resolution is given in absolute frequency units. To calculate the resolution in Å for example of group 0 at **FSC@0.5**:

$$\frac{\text{New pixel size}}{\text{Resolution}} = \frac{3.705 \text{ \AA/pixel}}{0.4791671/\text{pixel}} = 7.73 \text{ \AA}$$



Thus, the program accounted for all images and identified 2 groups (numbered 0 and 1, respectively)(due to intrinsic randomness of the sorting algorithm, number of particles may slightly differ somewhat for your run) each with 4 843 and 2 522 particle members, respectively. Group #0 shows the highest resolution, most probably due to the highest number of member particles.

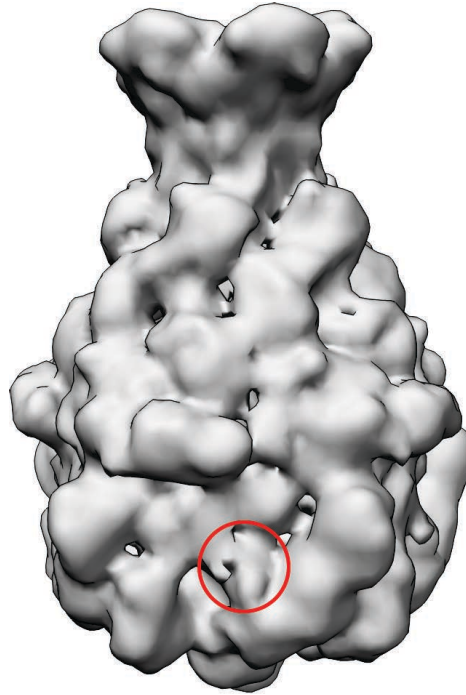
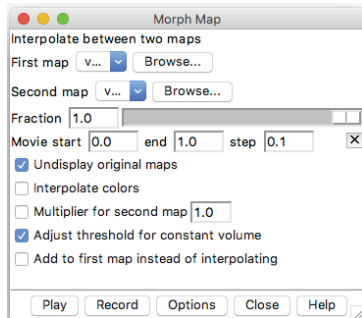
***TIP:** If you process a high-quality dataset of sufficient size, which has been precleaned by ISAC, in most cases, **RSORT3D** will account for 80 % to 100 % of the particles in the final groups. Avoid starting the sorting procedure with too many groups; the running time will increase significantly and so will the number of discarded particles.*

Following files are stored in the output folder **Sort3D** for each group:

- **Cluster*.txt:** Each cluster files contain a list with the IDs of the particle members of the respective group.
- **vol_grp*_iter*.hdf:** Each vol file contains the map calculated from the particles of the respective group (unfiltered)
- **volf_grp*_iter*.hdf:** Each volf file contains the map calculated from the particles of the respective group and in addition filtered according to the **FSC@0.5**.
- **resolution_grp000_000.txt:** **FSC** curve of the respective group.



Display the unfiltered volumes with **UCSF Chimera** and then use **UCSF Chimera's Morph Map Utility** to compare them. An even better way to compare the volumes is to low-pass filter them all using the **FSC@0.5** resolution of the lowest resolution map and then calculate difference maps and/or morph animations. As expected, the two classes look similar to each other (as mentioned before, TcdA1 is a rather rigid complex), but they indeed show differences in the regions predicted by the variability analysis.



The density showing the largest movements (marked by the red circle) corresponds to the N-term of the protein. This flexible domain was not resolved in the available X-ray structure. In this case, we expect a rather continuous flexibility that should further analyzed by clustering focused on this region and also performed at higher resolution. However, such an analysis would require a much larger dataset and would be beyond the scope of this tutorial. Instead, we will perform a final local refinement of group 0 (the group with the most members). A subset with increased homogeneity might produce a reconstruction with improved overall resolution.



Local Subset Refinement and Final Reconstruction

We will now perform a local refinement on the subset selected in the step (group 0). For this purpose, we will select a specific iteration from our previous high resolution 3D refinement. The program will continue from there with inherited refinement settings and projection parameters, but use only the selected particles. To save some time, we will choose the settings of Iteration 25, which was the one with the highest resolution. Because the refinement converged at this iteration, if we continue from there, the local refinement will perform only few additional iterations. To check the settings of Iteration 25, type:

```
grep "ITERATION #25" sxmeridien.o108884
```

NOTE: *The name of your logfile that contains the standard output of the 3D refinement, depends on your submission script.*

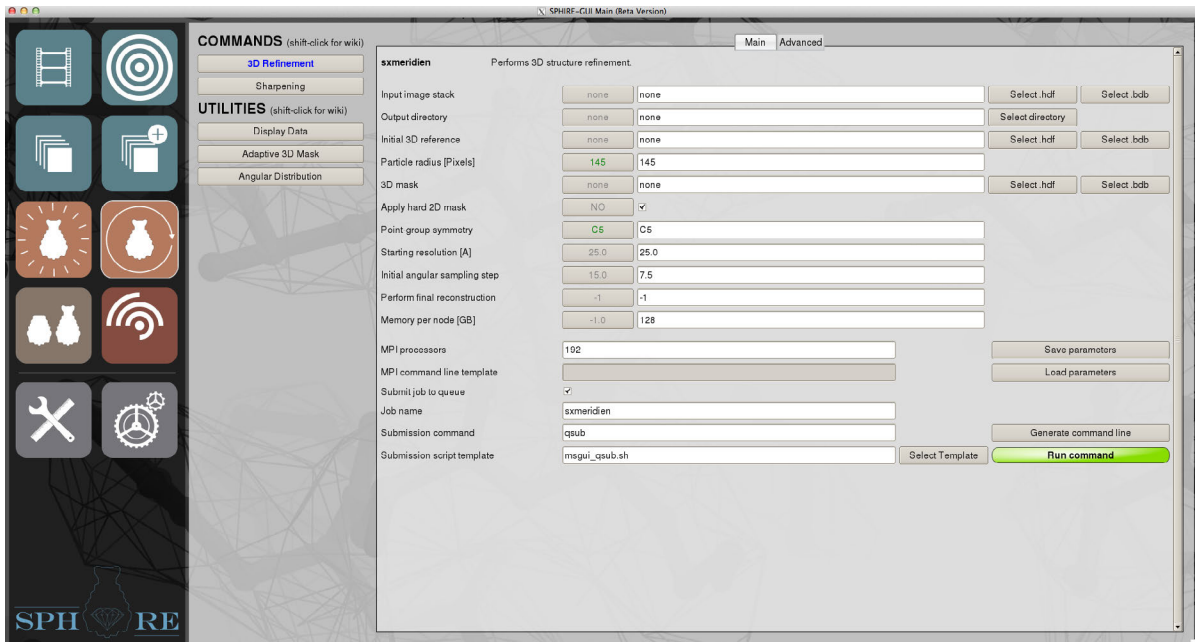
```
2016-12-14_22:26:32 => ITERATION #25. Resolution achieved so far: 47 pixels, 8.54Å.  
Current state: RESTRICTED, nxinit: 204, delta: 0.2344, xr: 0.8440, ts: 0.39492016
```

Thus, for the particles of this subset, we will continue with the projection parameters of iteration 25 and perform a refinement in *Restricted* mode with an angular step for the reference projections of 0.2344° and a range for translational search of 0.3945 pixel. The angular neighborhood used in the *Restricted* mode for the angular searches is 1.4° .

TIP: *In case in which **RSORT3D** revealed rather large conformational changes, it is preferable to continue the refinement of each subset with coarse settings, as the true projection parameters might be far off from the parameters obtained during the refinement of the full set containing the mixed populations. For this purpose, choose the settings of an early Exhaustive mode iteration.*



Go back to the main window of the **SPHIRE GUI** and press the button **Meridien** on the left and then the **3D Refinement** button in the middle.



Adjust the following parameters:

- **Input image stack:** none

NOTE: We do not need to specify an input dataset this time, since we will initiate a refinement in the continuation mode.

- **Output Directory:** none

*NOTE: We do not need to specify output directory this time, since in the continuation mode the program creates automatically an output folder named **ctrefrom-sort3d_(+time stamp)***

- **Initial 3D reference:** none

NOTE: No initial reference is necessary this time. The program will use the particles of this subset and the projection parameters obtained at the specific iteration, to calculate a reference volume for the next iteration.

- **3D mask:** none

NOTE: Do not specify a 3D mask here, unless you want to use a different one than the 3D soft-mask used during the refinement of the full set.

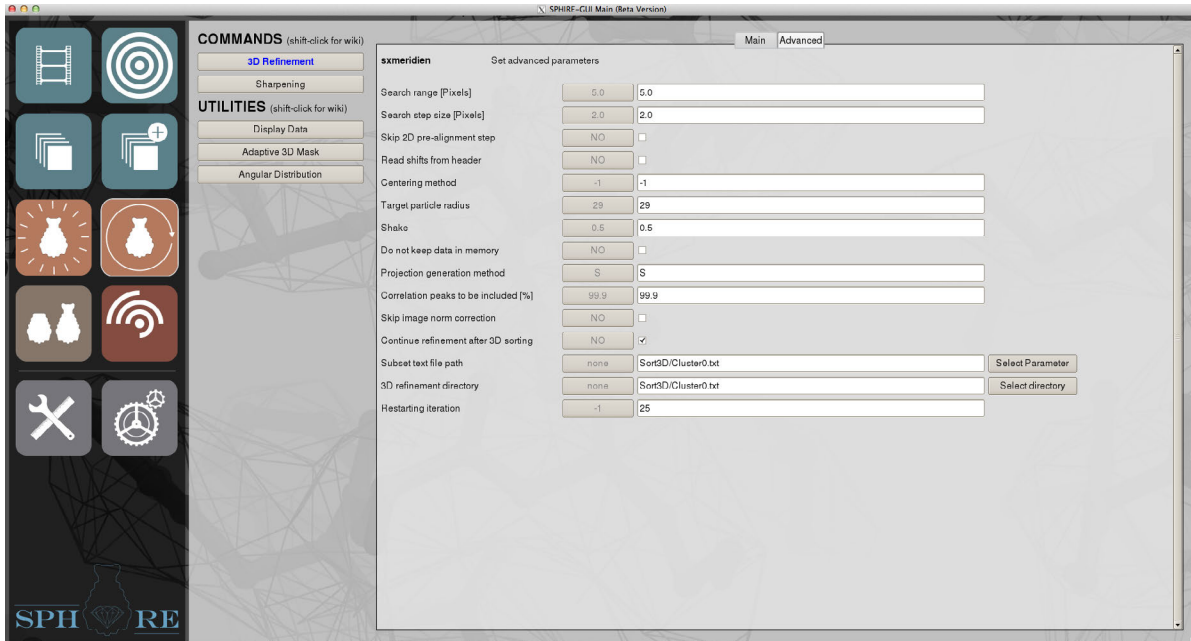
- **Initial angular sampling step:** 15



IMPORTANT: Use the default value (15), otherwise the program will not continue with the angular sampling step used in the respective iteration of the full set refinement, but use this value instead.

- **Memory per node [GB]:** 128.

Now you have to specify the subset of particles and the iteration of the 3D refinement you want to continue with. In the **advanced** parameters, set:



Adjust the following parameters:

- **Continue refinement after 3D sorting:** Sort3D/Cluster0.txt
- **Subset text file path:** Sort3D/Cluster0.txt

NOTE: Click the **Select Parameter** button and use the file browser to select the text file containing the particle IDs of the **RSORT3D** population, with the most particles (group 0).

- **3D refinement directory:** Refine3D

NOTE: Click the **Select Directory** button and use the file browser to select the output folder of your previous high resolution refinement.

- **Restarting Iteration:** 25

NOTE: Choose the Iteration number of the previous 3D refinement, with the highest resolution.



Specify the number of processors (on our cluster we used 192 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

The program will automatically create a folder named `ctrefromsort3d_(+time stamp)`. Once the local refinement converged, it will store in this folder the final half-volumes: **vol_0_unfil.hdf** and **vol_1_unfil.hdf** and the respective final projection parameters.

Otherwise, if you need to save time, you can copy the precalculated results to your **project directory** and continue with those.

On our cluster, this job finished after approximately 30 min.

Now we will use the **Sharpening** tool to compute the final reconstruction from these half-maps (sharpened, filtered and masked volume) and report the final resolution. A detailed description regarding the usage of this program is provided on our [wiki](#) page.

Go back to the main window of the **SPHIRE GUI** and press the **Sharpening** button in the middle. Load the two half maps (`ctrefromsort3d_(+time stamp)/vol_0_unfil.hdf` and `vol_1_unfil.hdf`), define **average_volume_sharpened_group0.hdf** as output volume, set the appropriate binarization threshold and use exactly the same parameters as in the previous run and then press the **Run command** button.

Monitor the progress of the **Sharpening** job at the terminal through the logfile **log.txt**:

```
tail -f log.txt
```

The output will look like this:

```
2016-12-15_15:45:34 => ----->>processing<<-----
2016-12-15_15:45:34 => 3-D refinement postprocess
2016-12-15_15:45:34 => The first input volume: ctrefromsort3d_15_Dec_2016_14_49_09/
vol_0_unfil.hdf
2016-12-15_15:45:34 => The second input volume: ctrefromsort3d_15_Dec_2016_14_49_09/
vol_1_unfil.hdf
2016-12-15_15:45:34 => starts creating surface mask, and wait...
2016-12-15_15:51:21 => MTF correction is applied
2016-12-15_15:51:21 => MTF file is FalconIImtf.txt
2016-12-15_15:51:47 => Similarity between the fitted line and 1-D rotationally
average power spectrum within [30, 116] is 0.270819
2016-12-15_15:51:47 => The slope is -1.410000 Angstrom^2
2016-12-15_15:52:00 => ----->>Summary<<-----
2016-12-15_15:52:00 => Resolution at criteria 0.143 is 3.459000 Angstrom
2016-12-15_15:52:00 => Resolution at criteria 0.5 is 4.137000 Angstrom
2016-12-15_15:52:00 => B-factor is -5.630000 Angstrom^2
2016-12-15_15:52:00 => FSC curve is saved in fsc.txt
2016-12-15_15:52:00 => Final processed volume is average_volume_sharpened_group0.hdf
2016-12-15_15:52:00 => guinierlines in logscale are saved in guinierlines.txt
2016-12-15_15:52:00 => Top hat low-pass filter is applied to cut off high frequencies from
resolution 1.0/3.460000 Angstrom
```




Thus, although we removed about 35 % of the particles and the final reconstruction is now obtained only from about 4 800 particles, we were still able to obtain a reconstruction at comparably high resolution (3.46 Å). Note also that the estimated accuracy for the angle assignment increased from 0.59 to 0.47 degrees and the estimated overall B-factor decreased. This further indicates that the selected subset is indeed more homogeneous and/or of higher quality and the ultimate resolution is most likely limited by the low number of particles.

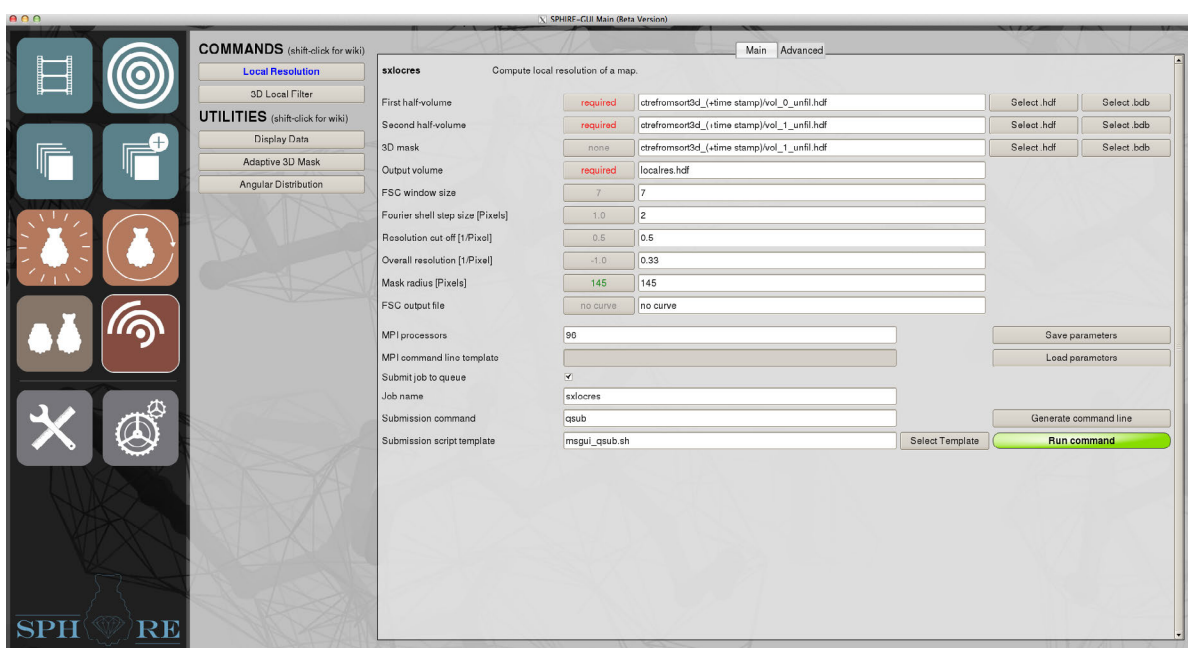
Display the final volume with **UCSF Chimera** and compare again the density with the available X-ray structure. The flexible N-terminal domain is still not resolved, but as mentioned before, successful sorting in case of localized continuous heterogeneity would require a much larger dataset and use of the focused 3D sorting strategy.



LocalRes

Local Resolution

To interpret a **cryo-EM** density map properly, it is necessary to know the resolution to which reliable structural information extends. The reported value of 3.45 Å in the previous step, obtained by the **FSC@0.143**, represents an average resolution for the entire map. However, due to intrinsic flexibility of the complex and image processing artifacts, resolution may vary locally. Using the **LocalRes** tool, we will now compute the local resolution of our final map. A detailed description regarding the usage of this program is provided on our [wiki](#) page. Go back to the main window of the **SPHIRE GUI** and press the button **LocalRes** on the left and then the **Local resolution** button in the middle.



Adjust the following parameters:

- **First half-volume:** ctfreformsort3d_(+time stamp)/vol_0_unfil.hdf
- **Second half-volume:** ctfreformsort3d_(+time stamp)/vol_1_unfil.hdf

NOTE: Click the **Select .hdf** button and use the file browser to select the final half-volumes, obtained from the high-resolution refinement of the selected subset.



- **3D mask:** vol_adaptive_mask.hdf

NOTE: Click the **Select .hdf** button and use the file browser to select the soft-edge 3D mask obtained in the **Sharpening** step.

- **Output volume:** localres.hdf

NOTE: Voxels of the output local resolution volume, are assigned a resolution value (in absolute frequency units). The analysis will be restricted to the voxels within the regions specified by the 3D mask. This will speed up the process.

- **FSC window size [pixels]:** 7

NOTE: Real space **FSC** will be performed within small window areas. This parameter defines the size of these window areas in pixels. The correct size depends on the resolution of the map. Setting a too small window size will result in inaccurate estimations that will vary widely from location to location, whereas a too large window will produce a rather coarse sampled resolution map, which most likely will not reflect local resolution variations sufficiently. We usually use a window size of 7 Å to 10 Å. As an example, if the resolution of a map is 7 Å and pixel size 1 Å, the window size should be at least 7 pixel. For a map with nominal resolution of 12 Å and pixel size 2 Å, the window size should be at least 6 pixel.

- **Fourier shell step size [pixels]:** 2

NOTE: Voxel step size for windowing. Setting a larger step size, will speed up the process, but produce a less precise resolution map.

- **Resolution cut-off [1/pixel]:** 0.5

NOTE: Always use the default value.

- **Overall Resolution [1/pixel]:** 0.33

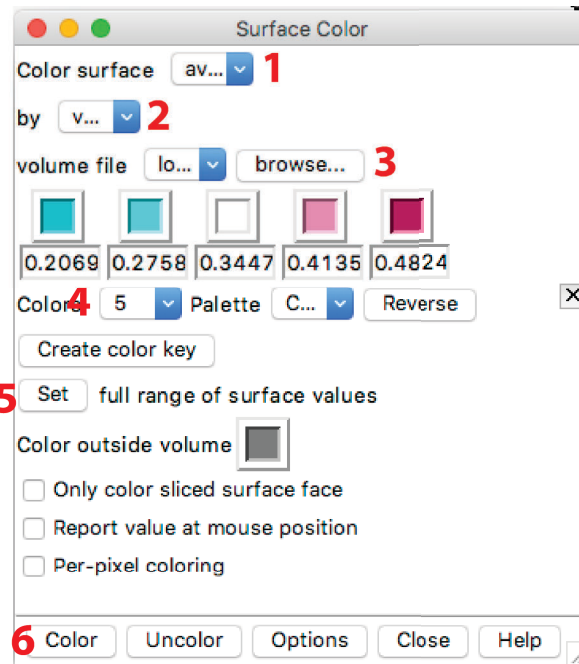
NOTE: Using a small window size might result in inaccurate and usually underestimated **FSC** measurements. This parameter allows calibrating the average value of all measured local resolution to the overall resolution, which was estimated between the two half-volumes in the previous step (3.46 Å). The resolution has to be defined in absolute frequency units.

$$\frac{\text{Pixel size}}{\text{Resolution}} = \frac{1.14 \text{ \AA}/\text{pixel}}{3.45 \text{ \AA}} = 0.329 \text{ 1/pixel}.$$

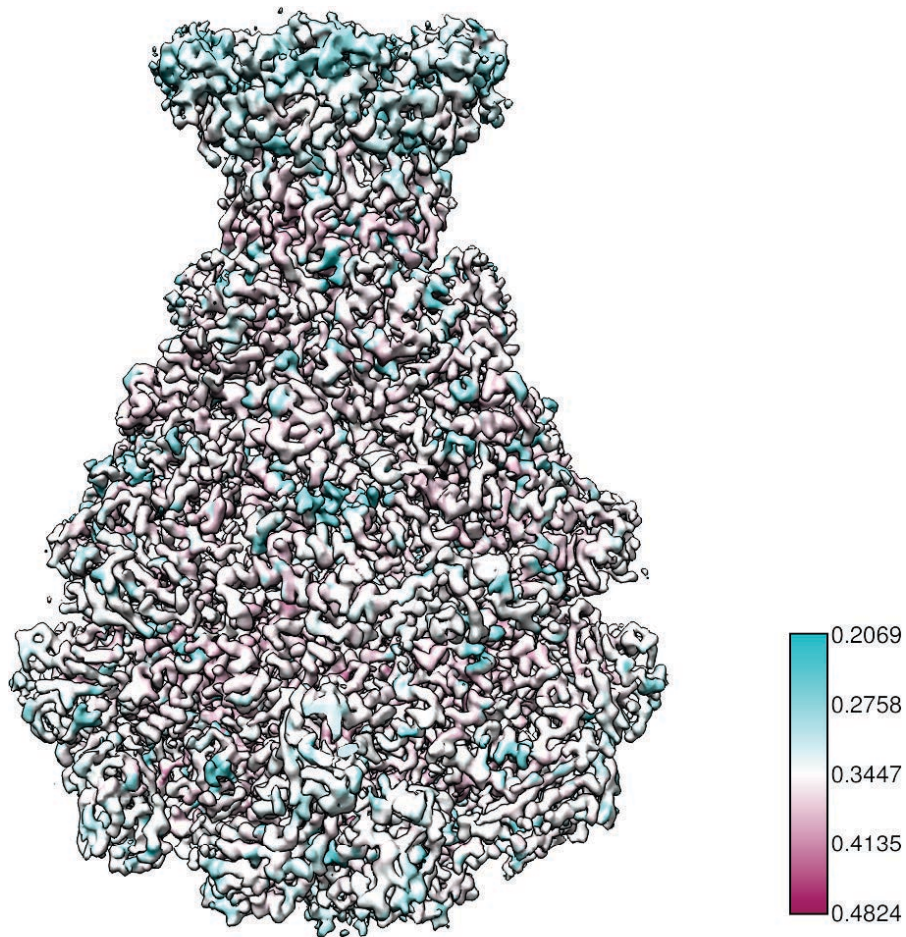
Specify the number of processors and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the Run command button.

On our cluster, this job with 96 processes finished after about 3 min.

Load the resulting resolution volume and the sharpened map in **UCSF Chimera**. Open the **Surface color** tool in **UCSF Chimera**.



- (1) Color surface of volume: average_volume_sharpened_group0.hdf
- (2) by volume data value
- (3) of volume file localres.hdf
- (4) Set number of colors to 5
- (5) press the *Set* button
- (6) and then the *Color* button



Note that the resolution values are given in absolute frequency units (e.g. a value of 0.26 corresponds to Pixel size / Abs. Frequency = 1.14 Å/pixel / 0.26 1/pixel = 4.38 Å).

TIP: *The respective resolution value should match the level of detail of the cryo-EM density map.*

However, if you do not have enough time, you can copy the precalculated local resolution map to your **project directory**:

```
cp SphireDemoResults/localres.hdf ./
```

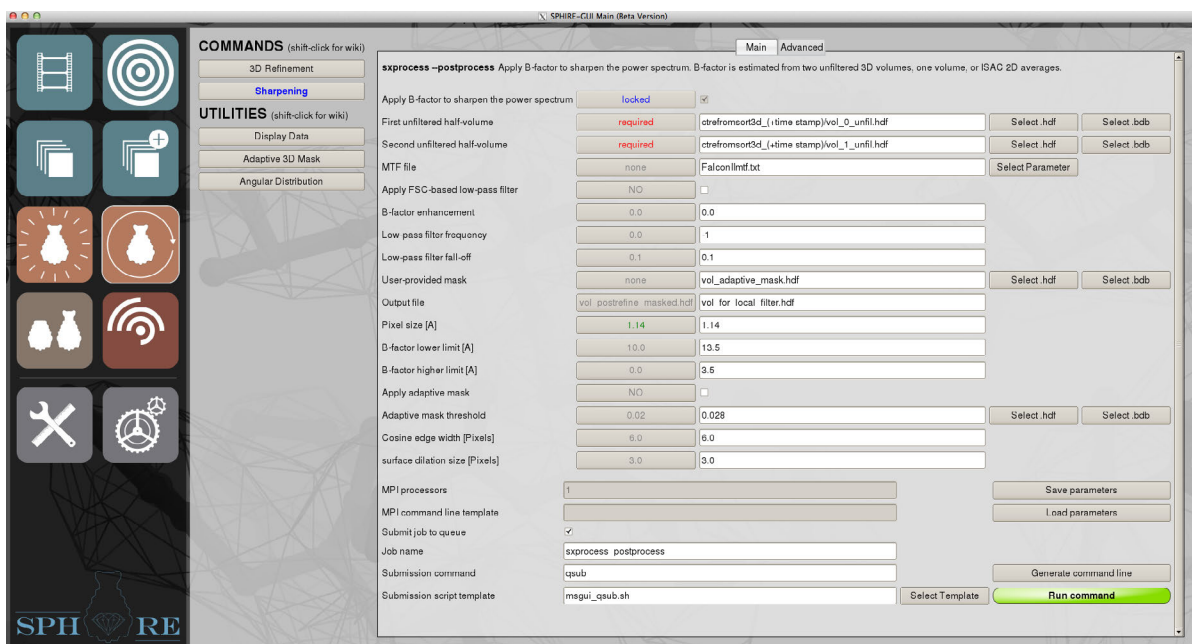


Local Filtering

After calculating the local resolutions of our final reconstruction, we will now filter the volume accordingly. Such a locally filtered volume is more suitable for interpretation and can be used for further analysis (e.g. model building) without the risk of over-interpreting individual features, since less well resolved regions (e.g. flexible domains) will now be filtered to their respective resolution and not to the average resolution. Similarly, regions better resolved than the average will not be suppressed and local filtering might allow improved molecular modeling at the respective region.

However, before applying the local filter, we need to re-run the **Sharpening** step in order to obtain a masked, sharpened, but unfiltered volume (the previous volume was filtered according to the **FSC@0.143**).

Go back to the main window of the **SPHIRE GUI** and press the button **Meridien** and then the **Sharpening** button in the middle.



Adjust following parameters and press the Run command button

- **First half-volume:** ctrefromsort3d_(+time stamp)/vol_0_unfil.hdf
- **Second half-volume:** ctrefromsort3d_(+time stamp)/vol_1_unfil.hdf
- **MTF File:** FalconIImtf.txt



- **Apply FSC-based low-pass filter:** No
- **B-Factor enhancement:** 0
- **Low-pass filter Frequency:** -1
- **User provided mask:** vol_adaptive_mask.hdf

NOTE: -1 means no low-pass filter

NOTE: we will use the adaptive mask produced in the previous sharpening run

- **Output file:** vol_for_local_filter.hdf
- **B-Factor lower limit (Å):** 13.5
- **B-Factor Higher limit (Å):** 3.5
- **Apply adaptive mask:** NO

NOTE: deactivate this option, since we provide a mask to the program

Monitor the progress of the **Sharpening** job at the terminal through the logfile **log.txt**:

```
tail -f log.txt
```

The output will look like this:

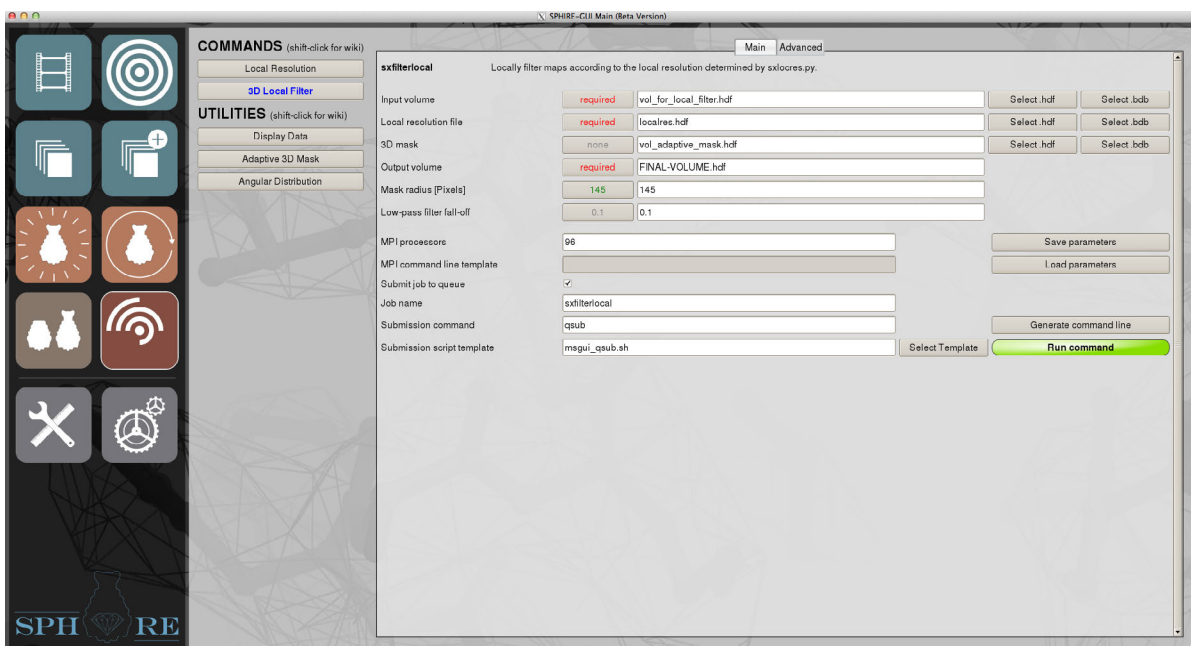
```
2016-12-15_16:20:19 => ----->>processing<<-----
2016-12-15_16:20:19 => 3-D refinement postprocess
2016-12-15_16:20:19 => The first input volume: ctrefromsort3d_15_Dec_2016_14_49_09/
vol_0_unfil.hdf
2016-12-15_16:20:19 => The second input volume: ctrefromsort3d_15_Dec_2016_14_49_09/
vol_1_unfil.hdf
2016-12-15_16:20:19 => User provided mask: vol_adaptive_mask.hdf
2016-12-15_16:20:27 => MTF correction is applied
2016-12-15_16:20:27 => MTF file is FalconIImtf.txt
2016-12-15_16:20:53 => Similiarity between the fitted line and 1-D rotationally average
power spectrum within [30, 116] is 0.270819
2016-12-15_16:20:53 => The slope is -1.410000 Angstrom^2
2016-12-15_16:21:03 => ----->>Summary<<-----
2016-12-15_16:21:03 => Resolution at criteria 0.143 is 3.459000 Angstrom
2016-12-15_16:21:03 => Resolution at criteria 0.5 is 4.137000 Angstrom
2016-12-15_16:21:03 => B-factor is -5.630000 Angstrom^2
2016-12-15_16:21:03 => FSC curve is saved in fsc.txt
2016-12-15_16:21:03 => Final processed volume is vol_for_local_filter.hdf
2016-12-15_16:21:03 => guinierlines in logscale are saved in guinierlines.txt
2016-12-15_16:21:03 => The final volume is not low_pass filtered.
```

This process is much faster (about 1 min), because we do not create a new adaptive mask this time.

Now we will use this output volume to apply the local filter.

Go back to the main window of the **SPHIRE GUI** and press the button **LocalRes** and then the **3D Local filter** button in the middle.

Adjust following parameters:



- **Input volume:** vol_for_local_filter.hdf

*NOTE: Click the **Select .hdf** button and use the file browser to select the sharpened but mask created in the previous step*

- **Local resolution file:** localres.hdf

*NOTE: Click the **Select .hdf** button and use the file browser to select local resolution volume*

- **3D mask:** vol_adaptive_mask.hdf

*NOTE: Click the **Select .hdf** button and use the file browser to select the soft-edge 3D mask obtained in the map adjustment step*

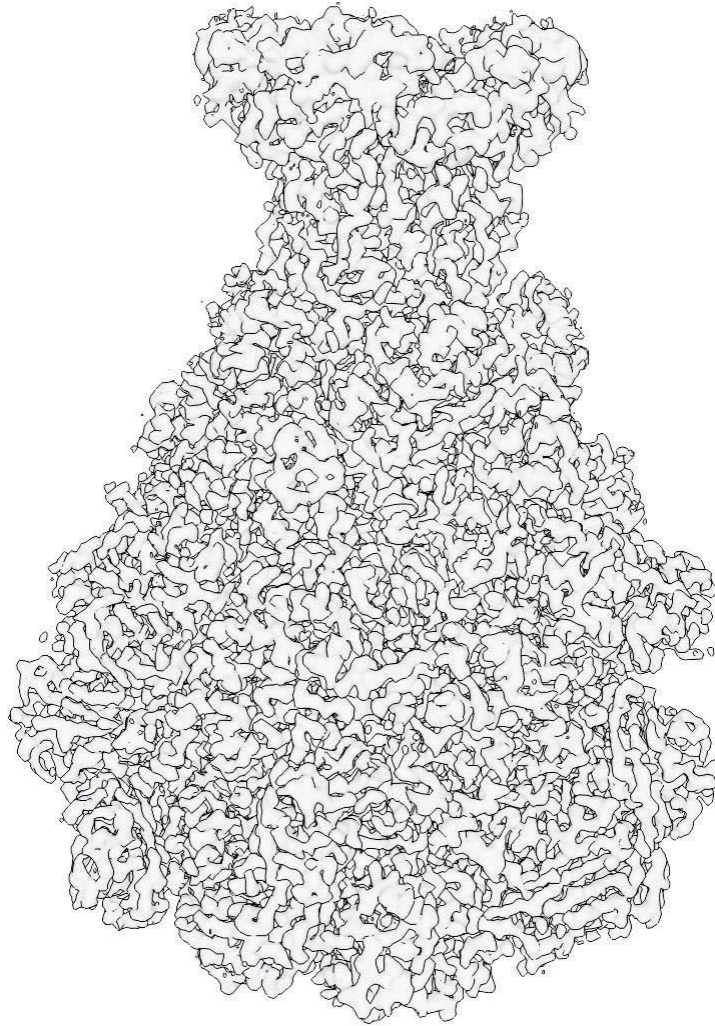
- **Output volume:** FINAL-VOLUME.hdf

NOTE: Voxels of the output local resolution volume, are assigned a resolution value (in absolute frequency units). The analysis will be restricted to the voxels within the regions specified by the 3D mask. This will speed up the process

Specify the number of processors (for this job we used 96 cores) and submit the job to the queuing system of your cluster using an appropriate submission file by pressing the **Run command** button.

On our cluster this job finished after about 3 min.

Display the final volume with **UCSF Chimera**. Compare again the density with the available X-ray Structure.



Congratulations!

You have finished the tutorial successfully.
Now it's time to process your own data.

Acronyms

BDB	Berkley Data Bank
cryo-EM	cryo electron microscopy
Cs	Spherical Abberation
CTF	Contrast Transfer Function
EM	Electron Microscopy
FSC	fourier shell correlation
GUI	graphical user interface
HDF	Hierarchical Data Format
ISAC	iterative stable alignment and clustering
MPI	Message Passing Interface
MTF	modulation transfer function
PDB	protein data bank
SD	standard deviation
SNR	Signal to noise ratio
SPA	single particle analysis
SPARX	single particle analysis for resolution extension
SPHIRE	SParx for HIgh-REsolution electron microscopy
XFEG	X-Field Emission Gun
SGE	Sun Grid Engine

Citing SPHIRE

The reference for **SPHIRE** has not been published yet. Until the first **SPHIRE** associated article becomes available, please cite **SPHIRE** as following:

Toshio Moriya, Michael Saur, Markus Stabrin, Felipe Merino, Horatiu Voicu, Zhong Huang, Christos Gatsogiannis, Pawel A. Penczek and Stefan Raunser (2016) SPHIRE:Sparx for High Resolution Electron microscopy (version alpha)[Software].

Available from <http://sphire.mpg.de>

and (Penczek et al. 2014), (Yang et al. 2012)

If you have any suggestions to improve this practical guide, do not hesitate to contact the author:
christos.gatsogiannis@mpi-dortmund.mpg.de

Bibliography

- Gatsogiannis, C., A.E. Lang, D. Meusch, V. Pfaumann, O. Hofnagel, R. Benz, K. Aktories, and S. Raunser (2013). “A syringe-like injection mechanism in *Photobacterium luminescens* toxins”. In: *Nature* 495, pp. 520–523. DOI: [10.1038/nature11987](https://doi.org/10.1038/nature11987).
- Grant, T. and N. Grigorieff (2015). “Measuring the optimal exposure for single particle cryo-EM using a 2.6 Å reconstruction of rotavirus VP6”. In: *eLife* 4, e06980. DOI: [10.7554/eLife.06980](https://doi.org/10.7554/eLife.06980).
- Hohn, Michael, Grant Tang, Grant Goodyear, P.R. Baldwin, Zhong Huang, Pawel A. Penczek, Chao Yang, Robert M. Glaeser, Paul D. Adams, and Steven J. Ludtke (2007). “SPARX, a new environment for Cryo-EM image processing”. In: *Journal of Structural Biology* 157, pp. 47–55. DOI: <http://dx.doi.org/10.1016/j.jsb.2006.07.003>.
- Meusch, D., C. Gatsogiannis, R.G. Efremov, A.E. Lang, O. Hofnagel, I.R. Vetter, K. Aktories, and S. Raunser (2014). “Mechanism of Tc toxin action revealed in molecular detail”. In: *Nature* 508, pp. 61–65. DOI: [10.1038/nature13015](https://doi.org/10.1038/nature13015).
- Penczek, P.A. (2010). “Resolution measures in molecular electron microscopy”. In: *Methods in enzymology* 482, pp. 73–100. DOI: [10.1016/S0076-6879\(10\)82003-8](https://doi.org/10.1016/S0076-6879(10)82003-8).
- Penczek, P.A., J. Fang, X. Li, Y. Cheng, J. Loerke, and C.M.T. Spahn (2014). “CTER-rapid estimation of CTF parameters with error assessment”. In: *Ultramicroscopy* 140, pp. 9–19. DOI: [10.1016/j.ultramic.2014.01.009](https://doi.org/10.1016/j.ultramic.2014.01.009).
- Pettersen, E.F., T.D. Goddard, C.C. Huang, G.S. Couch, D.M. Greenblatt, E.C. Meng, and T.E. Ferrin (2004a). “UCSF Chimera?A visualization system for exploratory research and analysis”. In: *J. Comput. Chem* 25, pp. 1605–1612. DOI: [10.1002/jcc.20084](https://doi.org/10.1002/jcc.20084).
- Pettersen, Eric F., Thomas D. Goddard, Conrad C. Huang, Gregory S. Couch, Daniel M. Greenblatt, Elaine C. Meng, and Thomas E. Ferrin (2004b). “UCSF Chimera—A visualization system for exploratory research and analysis”. In: *Journal of Computational Chemistry* 25, pp. 1605–1612. DOI: [10.1002/jcc.20084](https://doi.org/10.1002/jcc.20084).
- Tang, Guang, Liwei Peng, Philip R. Baldwin, Deepinder S. Mann, Wen Jiang, Ian Rees, and Steven J. Ludtke (2007). “EMAN2: An extensible image processing suite for electron microscopy”. In: *Journal of Structural Biology* 157, pp. 38–46. DOI: <http://dx.doi.org/10.1016/j.jsb.2006.05.009>.
- Yang, Z., J. Fang, J. Chittuluru, F.J. Asturias, and P.A. Penczek (2012). “Iterative Stable Alignment and Clustering of 2D Transmission Electron Microscope Images”. In: *Structure/Folding and Design* 20, pp. 237–247. DOI: [10.1016/j.str.2011.12.007](https://doi.org/10.1016/j.str.2011.12.007).