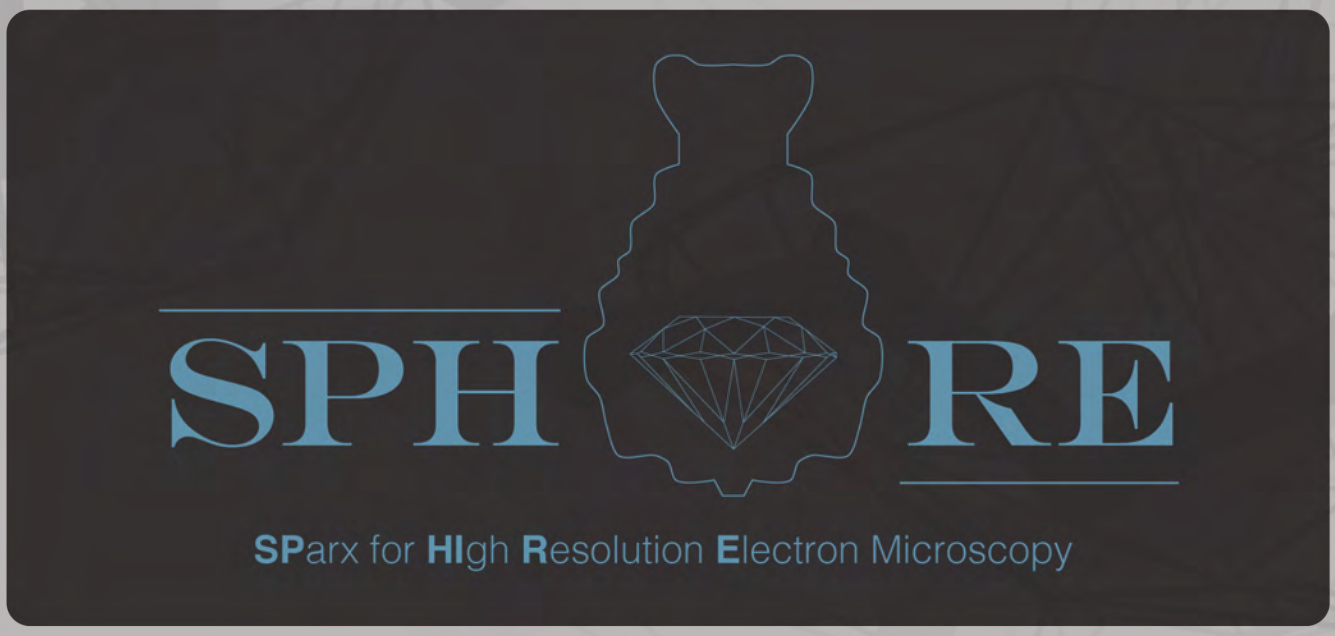


structure determination with SPHIRE

a practical guide

Info?
Turn here!

March 2018

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 the best results from your own data. Optional steps, performed outside the GUI, as well as known issues of the silver-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

Software Installation

Install SPHIRE and MPI support

To follow this guide and run SPHIRE you will need to properly install SPHIRE, either on a Linux computing cluster or a Linux/Mac multi-core desktop. In both case an MPI installation is required. Download SPHIRE at

http://sphire.mpg.de/wiki/doku.php?id=howto:download_latest

and follow the instructions regarding SPHIRE and MPI installation. It should be noted that SPHIRE and **EMAN2** (Tang et al. 2007) are installed jointly, i.e., after installation of SPHIRE, **EMAN2** is also available 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 completed successfully, open a terminal, type ***sphire***, hit the **Return** key and check if the SPHIRE GUI pops up. To make sure the MPI installation finished successfully, type ***sx.py*** in the terminal window and then type ***import mpi***. If there are no error messages, exit the interactive mode by typing ***quit***. Otherwise, it is advisable to contact either the local IT support or the SPHIRE team.

Installation of associated EM Software Packages

Alignment of direct detector movie frames is currently not included in SPHIRE. However, within the framework of the GUI we provide as utility 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 use the molecular graphics program **UCSF CHIMERA** (Pettersen et al. 2004b) (https://www.cgl.ucsf.edu/chimera/download.html). An extensive 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 determine a 3D cryo-EM structure 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 and the particle is a pentameric assembly with C5 point group symmetry.

First, create a **project directory** named **SPHIRE-demo**.

```
mkdir SPHIRE-demo
```

The **project directory SPHIRE-demo** will be created. As it is necessary to **always start the SPHIRE GUI and run all project-associated commands from the project directory**, change to this directory:

```
cd SPHIRE-demo
```

Download the tutorial data. At the terminal type (one long command):

```
wget --no-check-certificate
https://ftp.gwdg.de/pub/misc/sphire/test_dataset/sphire_testdata_movies.tar
```

(Alternatively, you can use your file browser to download the data from here:

https://ftp.gwdg.de/pub/misc/sphire/test_dataset/sphire_testdata_movies.tar)

In order to extract the downloaded file, type:

```
tar -xvf sphire_testdata_movies.tar
```

To launch SPHIRE GUI, type:

```
sphire &
```

When SPHIRE is started for the first time, the main window will appear.

The column on the left contains several pictograms that allow you to select a particular step of 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 for 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.*

Click the ***PROJECT*** pictogram and fill out the following fields in the GUI interface that will appear:

NOTE: ***Clicking*** *the gray button, next to the respective input field, retrieves a default value. After the settings are registered here, they will become default values in the subsequent steps.*

- **Protein name:** TcdA1
- **Micrograph pixel size [A]:** 1.14

  TIP: *Do not forget to adjust the pixel size, in case you binned your micrographs. For example, micrographs recorded with the K2 in super-resolution mode have a very small pixel size and usually we bin them before we start image processing in SPHIRE. To bin a micrograph 2 times (ratio of new to old image size=0.5), you can use following SPHIRE command at the terminal:*

  ```
  sxprocess.py input.mrc output.mrc --changesize --ratio=0.5
  ```

  You can also bin all micrographs using bash batch processing:

  ```
  mkdir binned_images
  ```

  and afterwards (one long command)

  ```
  for file in $(ls *.mrc);do sxprocess.py ${file} binned_images/${file}.mrc
  --changesize --ratio=0.5; done
  ```

- **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 × to 2 × larger than the longest axis of the particle. The long axis of the tutorial particle is 25 nm (250 Å) and the pixel size is 1.14 Å). Thus, the long axis of the particle is 250 Å / 1.14 Å/pixel = 220 pixel. Here we set the box size to 352 pixel, which corresponds to 1.6 × the size of our particle.*

  NOTE: *The window size has to be an even number due to Fast Fourier Transform (FFT) requirements of some programs.*

  TIP: *In case images you recorded images at rather high defocus values (i.e., > 3 µm), you might want to 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 for data processing, both in 2D as well as in 3D. Therefore, at the beginning of the project and especially if the center of the particle does not coincide with its center of mass, it might be preferable to use a slightly larger particle radius. For globular complexes that can be easily centered, such as ribosomes, you should use a rather tight radius closely corresponding to the particle radius right from the beginning.*

- **Point-group symmetry:** C5

  NOTE: *If the point group symmetry of the particle is uncertain, do not assume symmetry and enter C1.*

- **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

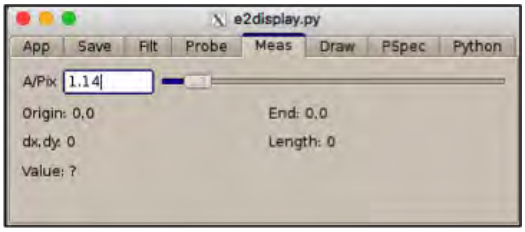
  NOTE: *Your microscope name/configuration. It is optional and is kept for the record only.*

Click the ***Register settings*** button, to register these values as defaults for all subsequent steps of the workflow, and then click the ***Save settings*** button, to save them in a text file. This text file may be useful for entering information into your lab book or can be used to reload the settings if necessary. A detailed description of the options is provided on our wiki page. Even if you do not save these settings now, registered parameters will be automatically displayed every time you start the GUI from the **project directory**.

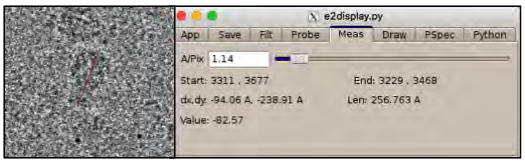
Measuring the size of your particle

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

1. Click the button ***UTILITIES*** on the left and then the Button ***Display Data*** in the middle to launch **e2display.py**.
2. Open an overview image with sufficient contrast.
3. Click 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 possibly 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 angstrom will be reported in the e2display.py pop up window (**Len**).



6. To get the particle radius in pixels, one needs to divide the result by the pixel size:

$$\text{Len in pixels} = \frac{\text{Length in angstrom}}{\text{Pixel size}} = \frac{256}{1.14} \approx 112$$

Movie

Micrograph Movie Alignment

Modern direct detectors record high-resolution images as movie frames. Usually, we motion-correct the micrograph movies on the fly during image acquisition. If motion-correction has not been performed yet, the first step in the workflow is the alignment of these frames for each image to correct the overall motion. SPHIRE provides as a utility, a wrapper of the program **Unblur**, developed by the Grigorieff lab.

> **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 respect and it is possible 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.*

> **TIP:** *If you are not processing movies, you can directly copy the digital micrographs to a folder within the **project directory** and proceed directly to the **CTER** (CTF Estimation) Section of the Tutorial.*

> **TIP:** *If you already used **Unblur** or **MotionCor2** to align your movies, copy all results (including the logfiles) to the output folder **CorrectedSums** within the **project directory** and proceed to the next step.*

> **TIP:** *If you are processing negative stain data, you can directly proceed to the **WINDOW** (Particle Extraction) section of the Tutorial.*

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
```

This folder contains 112 low-dose movies of TcdA1 in the mrc file format.

In the main window of the SPHIRE GUI click the button ***MOVIE*** on the left and then the ***Micrograph Movie Alignment*** button in the middle. An activated command button contains blue highlighted text.

Fill out the following input fields:

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

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

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

  NOTE: *Click the **Select MRC movie** 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, but not necessary, 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 executable** button to use the file browser to select it.*

- **Movie selection file:** none
- **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 have the same pixel size

- **Pixel size [A]:** 1.14
- **Microscope voltage [kV]:** 300
- **Per-frame Exposure [e/A^2]:** 2.5

  NOTE: *The 24-frame movies of this dataset have an overall dose of 60 e/Å^2. Thus, the per frame exposure is*

$$\frac{\textit{Total dose}}{\textit{Number of movie frames}} = \frac{60\,\mathrm{e}/\mathring{A}^2}{24} = 2.5\,\mathrm{e}/\mathring{A}^2\,.$$

- **Pre-exposure [e/A^2]:** 0
- **Number of threads:** 12

  TIP: *Expert users can also click 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 required about 90 minutes. 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 clicking 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 should seek help from the system IT administrator in order to prepare the correct file. Details about setting up the template file can be found on the SPHIRE wiki. (http://sphire.mpg.de/wiki/doku.php?id=howto:submissions)

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

```
tail -f name_of_sxunblur_logfile
```

```
Progress: 3.57%; Time: 0h:3m:9s/1h:28m:20s; Micrograph done:TcdA1-0012_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 -Rp 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 averages computed **without dose weighting**.
- **corrsum_dose_filtered:** Contains the motion-corrected, **dose-weighted** averages for all movie files
- **logfiles:** Contains the outputs of **Unblur** and **Summovie**
- **shift:** Contains text files with the x,y-shifts per frame calculated 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 with the **Unblur** shift file names

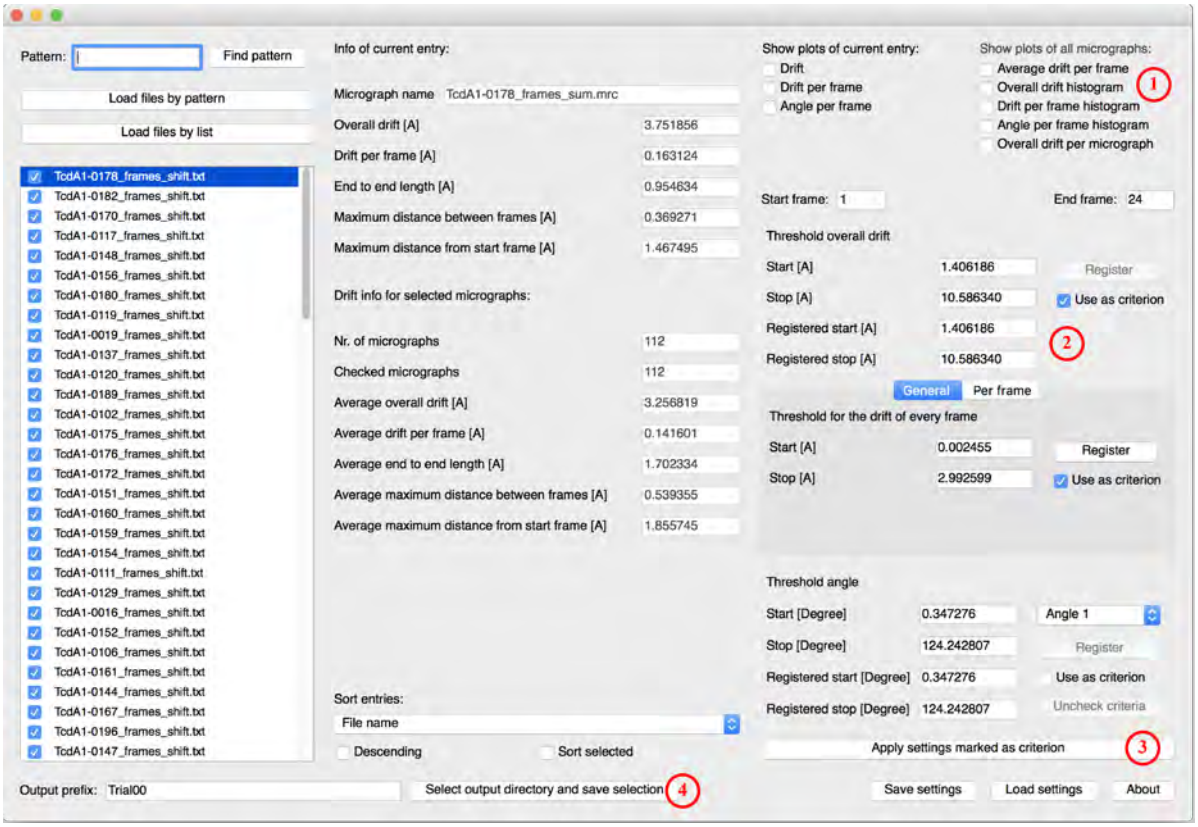
The motion-corrected averages computed without dose weighting are necessary for CTF estimation, whereas the dose-weighted averages will be used for all other steps of the workflow.

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 (**sxgui_unblur.py**) to select images with the lowest amount of drift.

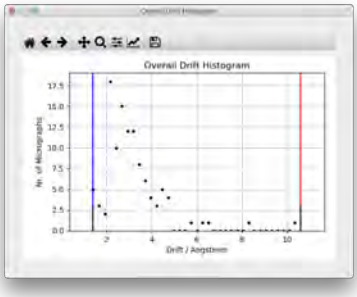
In the main window of the SPHIRE GUI click the button ***MOVIE*** on the left and then the button ***Drift Assessment*** in the middle.

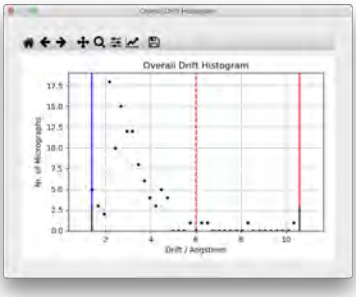
Click the ***Select drift shift params*** button next to **Shift files** and use the file browser to select one shift file (**CorrectedSums/shift/TcdA1-0001_frames_shift.txt**). Replace the micrograph number with the wildcard "*" and then click the ***Run command*** button to open the GUI.



TIP: *The **Drift Assessment GUI** also supports analysis of **MotionCor2** shift files. In this case, click the **Select drift shift params** button and use the file browser to select one **MotionCor2** shift file (*-Full.log file), replace the variable file name with the wildtype character "*" and click the **Run command** button.*

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



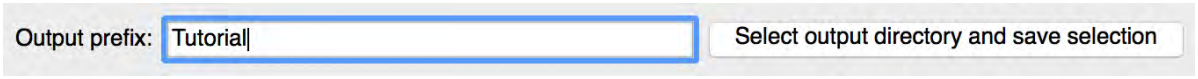


In the tutorial dataset, the average overall drift is 3.2 Å (Drift info for selected micrographs). Right click the plot to set the discard threshold to roughly 6 Å.

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

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**)



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

- **Tutorial_selected.txt:** List of selected micrographs; 108 micrographs (96 %)
- **Tutorial_discarded.txt:** List of discarded micrographs; 4 micrographs (3 %)
- **Tutorial_shift_selected.txt:** Shifts of selected micrographs
- **Tutorial_shift_discarded.txt:** Shifts of discarded micrographs

The average overall drift of the selected micrographs is now reduced to 3.07 Å.

> **TIP:** *A more advanced usage of the **Drift Assessment GUI** 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 also have a positive effect on the final resolution of the map.*

CTER

CTF Estimation

The next step is to estimate CTF parameters 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 the CTF estimation will be performed exclusively on the micrographs averaged **without** dose-weighting. In the main window of the SPHIRE GUI click the button ***CTER*** on the left and then the button ***CTF Estimation*** in the middle.

Fill out the following fields:

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

  NOTE: *Click the **Select MRC micrograph** button and use the file browser to select one of the motion-corrected averages computed **without** dose-weighting in the **corrsum** folder. Then replace the variable part of the file name (the serial number) with the wildcard character "*".*

- **Output directory:** CTFEST

- **Micrograph selection file:** DriftAssess/Tutorial_selected.txt

  NOTE: *Click the **Select micrograph list** 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 [A]:** 1.14
- **Microscope spherical aberration (Cs) [mm]:** 0

  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 resolution [A]:** 40

  NOTE: *Low-resolution cut-off.*

- **Highest resolution [A]:** 4

  NOTE: *High-resolution cut-off.*

Specify the number of processors (we used 96 for this job) and submit the job to the queuing system of the cluster using an appropriate submission file by clicking 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. At the terminal, type:

```
tail -f name_of_sxcter_logfile

Micrographs processed by main process (including percent of progress):

Processing CorrectedSums/corrsum/TcdA1-0001_frames_sum.mrc ---> 0.00%
```

On our cluster, this process finished after about 3 minutes. However, if you do not have enough time to wait for the results, copy our pre-calculated results to your project.

```
cp -r SphireDemoResults/CTFEST ./
```

Once the job has finished, the CTF results are stored in **CTFEST/partres.txt**. A detailed explanation of the ***CTER*** outputs and the format of the **partres.txt** file can be found on the SPHIRE wiki. (`http://sphire.mpg.de/wiki/doku.php?id=pipeline:cter:sxcter` and in the ***CTER*** paper (Penczek et al. 2014).

Due to internal randomization of statistical resampling procedure used to compute parameters and their standard deviation, results of different ***CTER*** runs will differ slightly.

TIP: *Open the **CTER** output using a text editor and confirm that the image defocus (column 1) is within the range used during data collection and its standard deviation (column 9) is low.*

CTF Assessment

The CTF results are analyzed using the **CTF Assessment GUI** (**sxgui_cter.py**). This tool is used to assess the overall quality of the dataset and to identify and remove outliers that might have a negative impact on the final result. This is accomplished by evaluating errors of CTF parameters estimated from the collected micrographs. Micrographs for which estimated parameters are unreliable, i.e., have large errors, can be removed as they will not contribute high-resolution information. The **CTF Assessment GUI** allows screening using multiple criteria simultaneously and this is particularly useful for the fast and effective analysis of large datasets.

Here are examples of possible micrograph selection criteria to employ:

⇒ Within a specific defocus range.

NOTE: *If you aim to reconstruct a rather "large" particle and your dataset contains sufficient number of images and image contrast is not an issue, why would you like to keep far-from-focus images in your dataset? They will not contribute high resolution information to the project.*

⇒ With low standard deviations regarding the defocus.

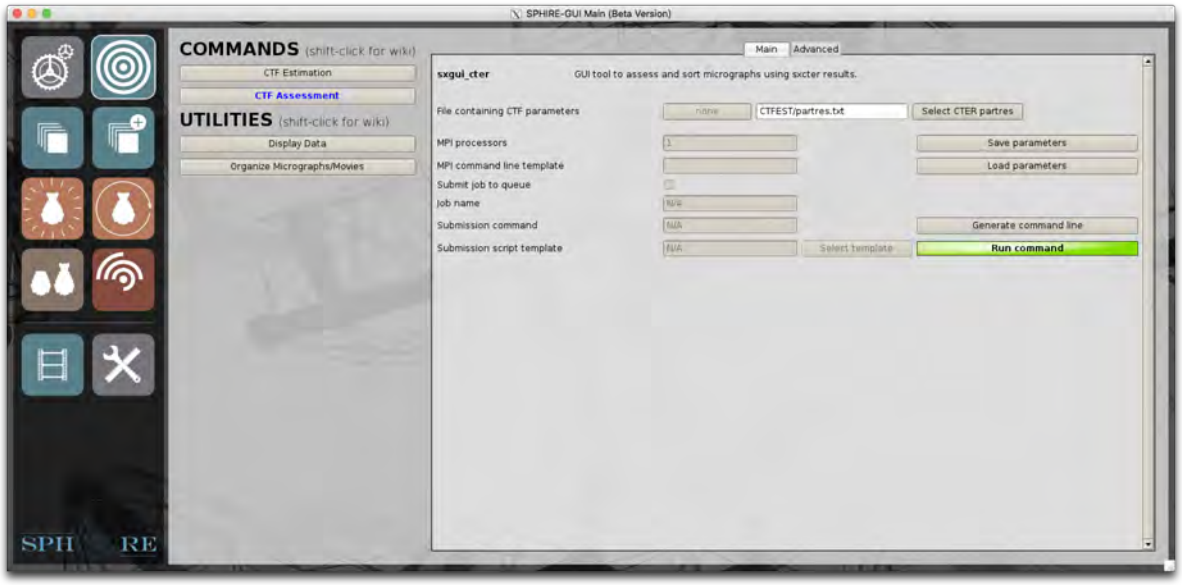
NOTE: *A uniform defocus across the micrograph field is critical for near atomic resolution structure determination.*

⇒ With a certain frequency limit.

NOTE: *Micrographs with unreliable CTF parameters and large defocus gradients, will most likely have a negative impact on the quality of the final map.*

Elimination of low-quality micrographs reduces data processing time and improves the quality and resolution of the final structure.

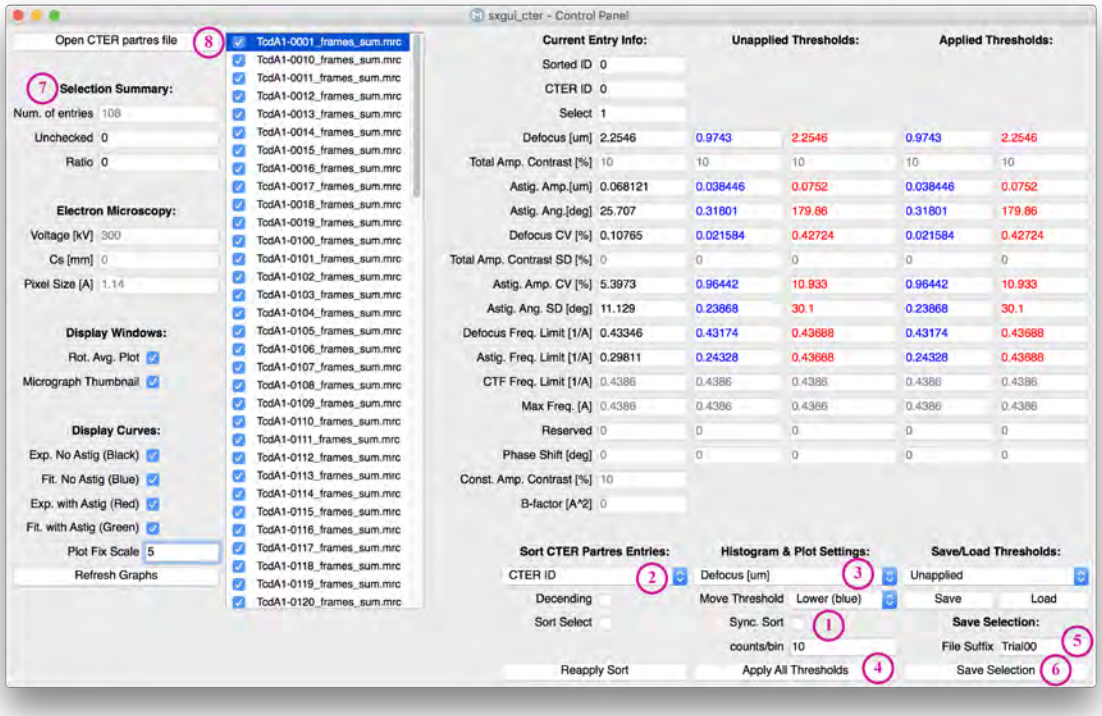
In this tutorial, we will only perform a simplified screening by setting thresholds for the defocus value and the astigmatism frequency limit (1st and 16th column in **partres.txt**). Open the main window of SPHIRE GUI and click the button ***CTER*** on the left and then the button ***CTF Assessment***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



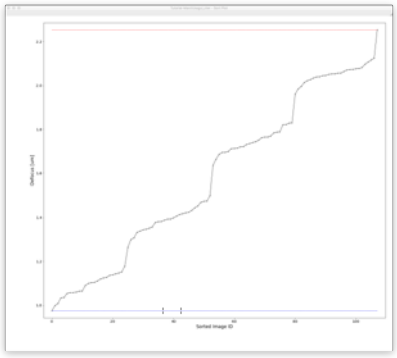
Click the ***Select CTER partres*** button and use the file browser to select the CTF parameter file **partres.txt** in the **CTFEST**-folder and click the ***Run command*** button to launch the GUI tool and six windows will pop-up:

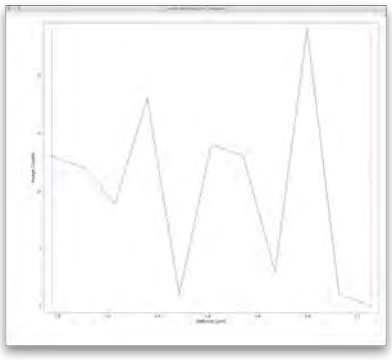
- **Control Panel**
- **Histogram**
- **Sort Plot**
- **Zoom Plot**
- **Plot**
- **Micrograph**

This is the **Control Panel** window:

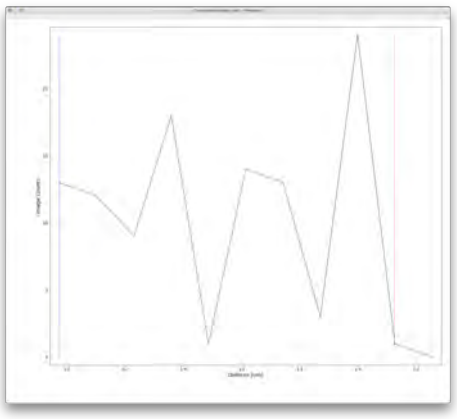


First, check the **Sync. Sort** option (1). This will synchronize selection of parameters in **Sort CTER Partres Entries** (2) and **Histogram & Plot Settings** (3) and their respective windows. Now we will analyze the defocus distribution of this tutorial dataset. Select **Defocus [um]** from the combo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





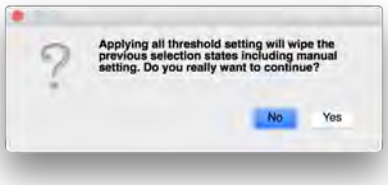
From both plots, we can see that the defocus of this dataset ranges from 0.97 µm to 2.25 µm. To demonstrate the functionality of this tool, we will now discard all micrographs with a defocus > 2.13 µm by setting the respective threshold. Left click the Histogram window while clicking the SHIFT button 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 in order to discard micrographs with a defocus lower than a specific value. In this case* ***left click*** *the* ***Histogram*** *window at the respective value and a blue dashed line will appear at the respective position.*

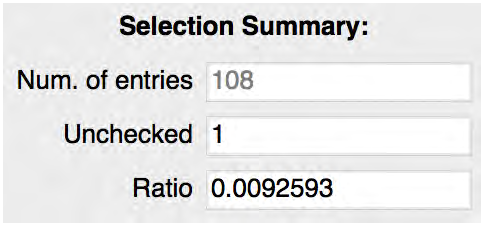
NOTE: *The* ***Sort plot*** *window supports the same mouse control. Usually, we use this approach to identify defocus "outliers" and/or select micrographs within a specific search range.*

In order to apply the user-defined defocus threshold, click the ***Apply All Thresholds*** (4) button. A message dialog will pop up. Click **yes** to apply the threshold.



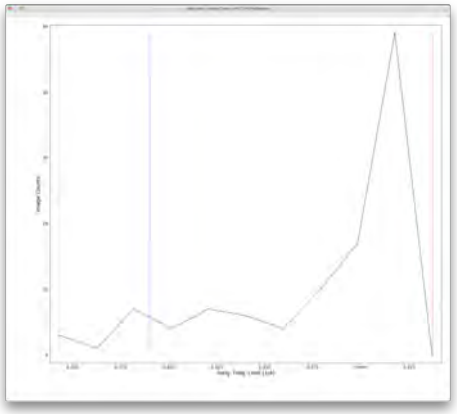
NOTE: *The respective outliers have now been unchecked in the micrograph list.*

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 Partres Entries****. All unchecked micrographs will now be placed at the top of the micrograph list.* ***Left click*** *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/A]**. This parameter provides a good criterion to identify micrographs that have little high-resolution information content. In the control panel, instead of **Defocus [um]**, select the **Astigm. Freq. Limit [1/A]** parameter from the combo box (3) and check again the **Histogram** and **Sort Plot** windows. Using the procedure described above, set a low cuf-off threshold at about 0.29 1/Å (=3.44 Å) and click 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 application of this threshold, the low value cut-off (blue dashed line) should be at the following position in the Histogram plot:



NOTE: *The value of 0.29 1/Å corresponds to a resolution limit of* $1/0.29\,Å = 3.44\,Å$*. Thus, in this case we will discard all micrographs that have an astigmatism resolution limit > 3.44 Å.*

TIP: *You can also use other parameters to screen micrographs. However, from our experience* ***Astigm. Freq. Limit [1/A]*** *is the most convenient criterion to identify bad micrographs. Thus, shifting the mean value of the* ***Astigm. Freq. Limit [1/A]*** *of your dataset to a higher value is likely to improve the final outcome. Moreover, if one would prefer to discard micrographs with high amount of astigmatism, we recommend using the* ***Astig. Amp. [um]*** *criterion. 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. Finally, you should always remove micrographs whose CTF parameters have unusually high errors, meaning high standard deviations (SD of parameters).*

At this point the number of unchecked micrographs increased from 1 to 7. Enter “Tutorial” in **File Suffix** (5) under **Save Selection:** in the **Control Panel** and click the ***Save Selection*** button (6). A message dialog will pop up and inform you that following files are saved:

- **Tutorial_micrographs_select.txt:** List of selected Micrographs
- **Tutorial_shift_discard.txt:** List of discarded micrographs.
- **Tutorial_partres_select.txt:** CTF parameters of the selected micrographs.
- **Tutorial_partres_discard.txt:** CTF parameters of discarded micrographs.
- **Tutorial_thresholds.txt:** Applied thresholds.

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

TIP: *After this first selection step with the **CTF Assessment GUI**, it is possible to perform a second round of screening if necessary. For this purpose, click the **Open CTER partres file** button (8) and load the CTF parameter file of the selected micrographs (**Tutorial_partres_select.txt**).*

Window

Particle Coordinates

In order to extract particles from the selected micrographs, we need the respective coordinate files. SPHIRE does not directly provide yet 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.

Please keep in mind that in the previous steps of the tutorial, we created 2 lists of micrographs to discard.

- **List of images with a large overall drift (DriftAssess/Tutorial_discarded.txt)**

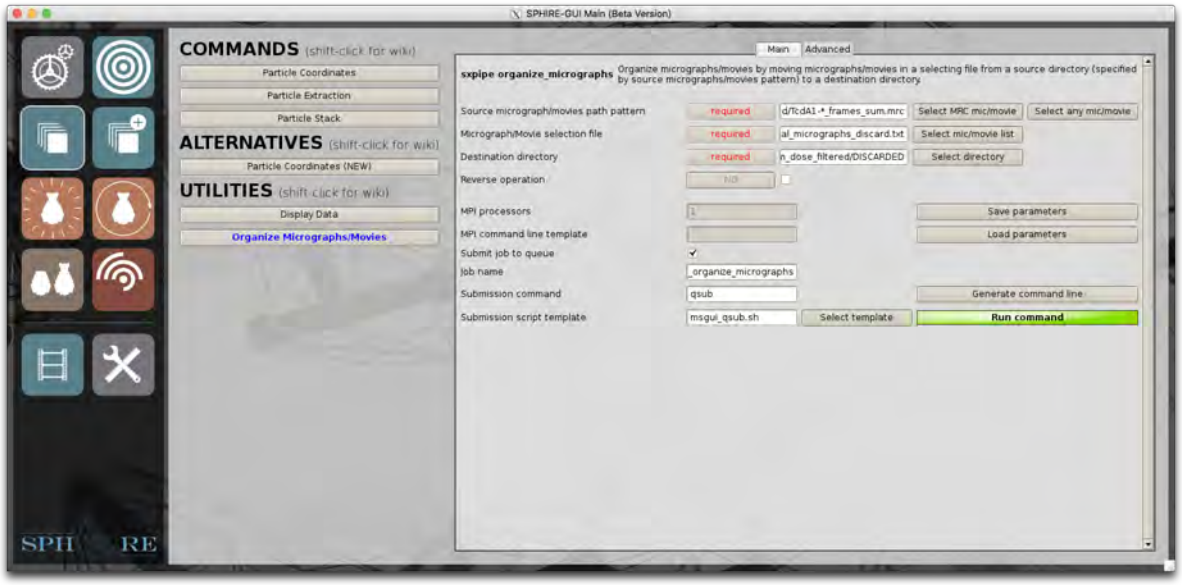
  NOTE: *This list was obtained using the **Drift Assessment GUI**.*

- **List of images with low resolution limit (CTFEST/Tutorial_micrographs_discard.txt)**

  NOTE: *This list was obtained using the **CTF Assessment GUI**.*

The first step now is to move all these micrographs included in the two “discarded” lists into a different folder. This is necessary as we want to pick particles only from the selected “good” micrographs. For this purpose, we will use the utility ***Organize Micrographs/Movies***.

In the main window of the SPHIRE GUI click button ***WINDOW*** on the left and then the button ***Organize Micrographs/Movies*** (***UTILITIES***) in the middle.



Fill out the following fields of the GUI interface:

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

  NOTE: *Click the **Select MRC mic/movie** 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**).*

  NOTE: *From now on, we will exclusively work with **motion-corrected and dose-weighted** frame averages.*

- **Micrograph/Movie selection file:** CTFEST/Tutorial_micrographs_discard.txt

  NOTE: *Click the **Select MRC mic/movie list** button and use the file browser to select the CTF Assessment list of discarded micrographs (**CTFEST/Tutorial_micrographs_discard.txt**).*

- **Destination directory:** CorrectedSums/corrsum_dose_filtered/DISCARDED

Click the ***Run command*** button and check the content of the standard output.

```
2017-11-24_16:26:02 =>Summary of dataset consistency check...
2017-11-24_16:26:02 =>Detected : 112
2017-11-24_16:26:02 =>Valid : 7
2017-11-24_16:26:02 =>Rejected by not found in both dirs : 0
2017-11-24_16:26:02 =>Rejected by already in dst dir : 0
2017-11-24_16:26:02 =>Rejected by duplicated in dst dir : 0
```

```
2017-11-24_16:26:02 =>Moving micrographs in the selecting list
(CTFEST/Tutorial_micrographs_discard.txt) from the source directory
(CorrectedSums/corrsum_dose_filtered) to the destination directory
(CorrectedSums/corrsum_dose_filtered/DISCARDED)...

2017-11-24_16:26:0 =>Summary of processing...

2017-11-24_16:26:0 =>Moved : 7
```

The program moved 7 “bad” micrographs to the destination directory:

CorrectedSums/corrsum_dose_filtered/DISCARDED, according to the entries of the ***CTF Assessment*** list.

Now we will repeat the same procedure in order to move “bad” micrographs of the ***Drift Assessment*** discarded list to the destination directory **DISCARDED**. In the ***Organize Micrographs/Movies*** window, replace the ***CTF Assessment*** discarded list with the ***Drift Assessment*** discarded list (**DriftAssess/Tutorial_ discarded.txt**) and click the ***Run command*** button.

Check again the content of the standard output.

```
2017-11-24_16:35:09 =>Summary of dataset consistency check...

2017-11-24_16:35:09 =>Detected : 105

2017-11-24_16:35:09 =>Valid : 4

2017-11-24_16:35:09 =>Rejected by not found in both dirs : 0

2017-11-24_16:35:09 =>Rejected by already in dst dir : 0

2017-11-24_16:35:09 =>Rejected by duplicated in dst dir : 0

2017-11-24_16:35:09 =>Moving micrographs in the selecting list
(DriftAssess/Tutorial_discarded.txt) from the source directory
(CorrectedSums/corrsum_dose_filtered) to the destination directory
(CorrectedSums/corrsum_dose_filtered/DISCARDED)...

2017-11-24_16:35:09 =>Summary of processing...

2017-11-24_16:35:09 =>Moved : 4
```

This time, the program moved 4 "bad" micrographs to the **DISCARDED** output directory, according to the entries of the ***Drift Assessment*** list. Thus, the output directory should now contain a total of 11 micrographs that will not be considered for the further steps of the analysis.

Now, in the main window of the SPHIRE GUI click the button ***WINDOW*** on the left and then the ***Particle Coordinates*** button in the middle.

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

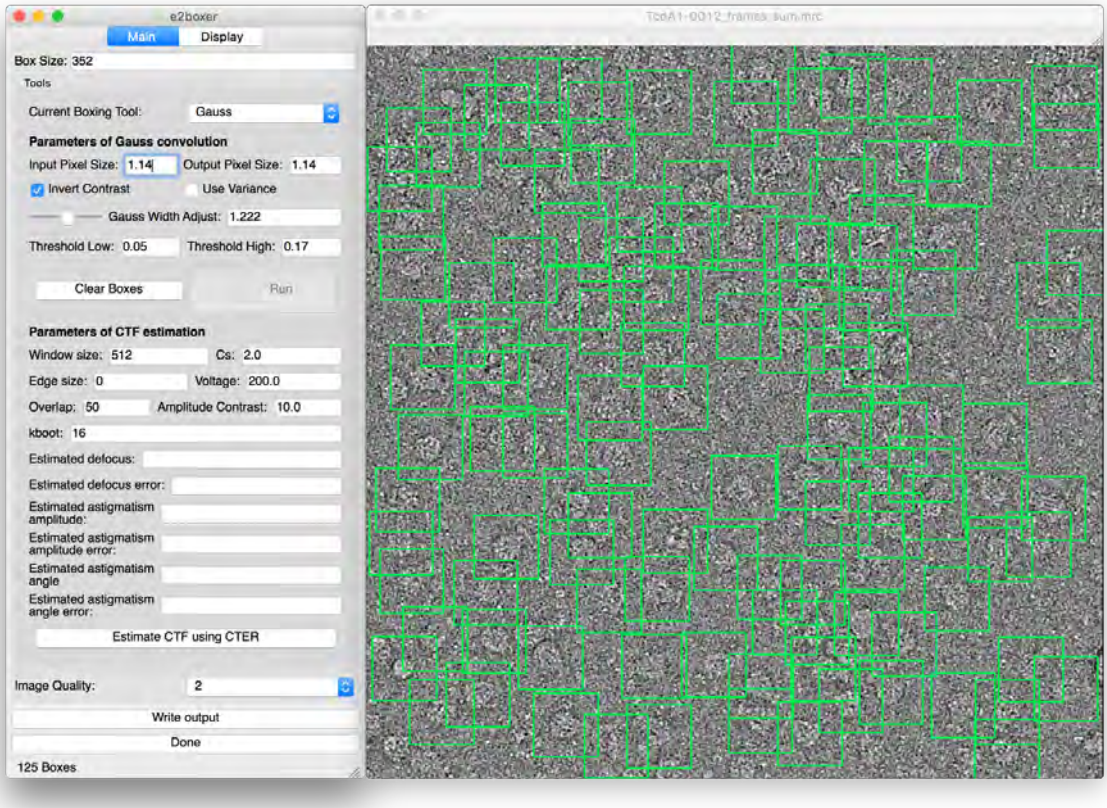
NOTE: *This folder contains only the "good" micrographs.*

Replace the variable part of the file name (in this case the serial number) with the wildcard character "*" (i.e., from **TcdA1-0010_sum.mrc** to **TcdA1-*_sum.mrc**) and click the ***Run command*** button to launch the **e2boxer_old.py** GUI. We will pick particles automatically using the **Gauss boxing tool** and store their coordinates in the *.box* file format (**EMAN1**) in a folder named Coordinates.

- In the **e2boxer** control window, select **Gauss** as the **Current Boxing Tool**.
- Select a micrograph with high contrast. Note the number of the micrograph.
- Set **Input** and **Output Pixel Size** to 1.14 Å.
- **Check** the **Invert Contrast** and **uncheck** the **Use Variance** Option.
- Set **Gauss Width Adjust** to 1.222, **Threshold Low** to 0.05, **Threshold High** to 0.17 and click the ***Run*** button.
- Control picking quality and adjust settings if necessary.

- Once finished, click the ***Done*** button.

Typical settings for the **Gauss boxer** are shown in the image below.



The settings for the selected micrograph (did you notice the micrograph number?) are automatically stored after clicking the ***Done*** button. Based on these settings, we will now automatically identify particles in all micrographs.

At the terminal type (one long command):

```
e2boxer_old.py --indir CorrectedSums/corrsum_dose_filtered/*.mrc --write_dbbox
--gauss_autoboxer CorrectedSums/corrsum_dose_filtered/TcdA1-NUMBER_frames_sum.mrc
```

The coordinates for all micrographs will be stored in the *.box* file format in the **project directory**. Make a folder named **Coordinates** and move all *.box* files into this folder. At the terminal type:

```
mkdir Coordinates

mv *.box Coordinates
```

TIP: *The **Gauss boxing tool** will work well for globular particles. For more challenging projects and intricate shapes, we recommend to use the reference- or Neural Network-based particle picking tools implemented in a more recent version of **e2boxer.py**. To launch an **e2boxer.py** version that includes these tools, in the main window of the SPHIRE GUI click the button ***WINDOW on the left and then the Particle Coordinates (NEW)*** button in the middle (**ALTERNATIVES**).*

*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 you might want to try different particle picking methods as implemented in other software packages. Our windowing utility can import various particle coordinate files, so results from other programs can be easily used (see below).*

In case there is no time to perform autopicking, you can use provided coordinate files, which were prepared with Gauss autoboxing of **e2boxer_old.py**.

In this case type:

```
cp -Rp SphireDemoResults/Coordinates ./
```

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

At the terminal, type:

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

The folder should contain 101 coordinate files.

TIP: *If you picked your micrographs with a different particle picking utility, 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 the **EMAN1** coordinate file format (.box), but SPHIRE also supports coordinates from **EMAN2** and **SPIDER** with the .json and .spi extension, respectively. In case **RELION** coordinate files are available (.star), you can easily convert them to **EMAN1** file format (.box) using the **sxrelion2sphire.py** utility. For this purpose, in the main window of the SPHIRE GUI click the pictogram **UTILITIES** on the left and then the **RELION to SPHIRE conversion** button in the middle. Detailed instructions are provided in the **Import a star file** page of the wiki. http://sphire.mpg.de/wiki/doku.php?id=howto:relion2sphire*

Particle Extraction

In the main window of the SPHIRE GUI click the button ***WINDOW*** on the left and then the **Particle Extraction** button in the middle.

Fill out the following input fields at the GUI interface:

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

  NOTE: *Click the **Select MRC micrograph** 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 the processed data is negative stain and the particle coordinates are available, one can start directly with particle extraction and skip all previous steps. In this case, one inserts here **the/path/to/your/micrographs/*.mrc** instead.*

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

  NOTE: *Click the **Select BOX coordinates** 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 "*" (i.e., from **TcdA1-0010_frames_sum.box** to **TcdA1-*_frames_sum.box**). In this demo, the folder **Coordinates** contains 101 .box files.*

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

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

NOTE: *Click the **Select CTER partres** 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 transferred to the header of all extracted particles from the respective micrograph.*

TIP: *If you process negative stain data, do not provide the CTF parameter source file here, but type the pixel size of your micrographs instead. In this case, the program will transfer an "ideal" CTF (no modulation of the signal; 90° phase shift; defocus 0 µm) to the header of the extracted particles.*

- **Output directory:** Particles
- **Micrograph selecting list:** CTFEST/Tutorial_micrographs_select.txt

  NOTE: *Click the **select micrograph list** 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
- **Invert image contrast:** YES

  NOTE: *For typical cryo-EM data (particles appear dark in brighter background), leave this flag as it is. The program will invert the contrast of your cryo-EM images automatically.*

  NOTE: *If your particles appear bright and the background dark (e.g. negative stain or inverted cryo-EM images), you will have to deactivate this flag.*

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

```
Global summary of coordinates level processing...

Detected : 11011

Processed : 11003

Rejected by out of boundary : 8
```

However, if there is no time to wait for the results, copy our precalculated results to your **project directory**.

```
cp -Rp SphireDemoResults/Particles ./
```

Once the job has finished, the extracted particles are stored in the folder Particles. For each processed micrograph, extracted particle images are stored in an independent BDB data file. On our cluster, this step completed in about 1 minute.

Particle Stack

The next step is to combine the individual per-micrograph stacks into a single "virtual" particle stack. We call this stack "virtual" as, taking advantage of **EMAN2**'s BDB format, it contains **only** metadata (header information associated with images, such as pixel size, particle micrograph origin, any alignment parameters) and links to the original images. In this way we minimize disk space usage while affording great flexibility with rapid creation of particle data stacks as necessary in subsequent data processing steps.

> **TIP:** *BDB files reside in separate directories always named **EMAN2DB**.*
> *An **EMAN2DB** directory that is not linked to any other BDB directory (through the virtual stack mechanism) can be easily archived, copied and transferred. However, directories linked through the virtual stack mechanism have to be copied jointly while preserving relative directory structure. It is very important not to modify **EMAN2DB** directories content in any way as this usually results in irreversible loss of information.*

In the main window of the SPHIRE GUI click the button ***WINDOW*** on the left and then the ***Particle Stack*** button in the middle.

Fill out the following input fields:

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

  NOTE: *This defines 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 "*" (i.e., **mpi_proc_000** to **mpi_proc_***).*

Click the ***Run command*** button to create the stack and monitor the progress of this job through the standard output. This step is finished in few seconds.

After the job finished, you can verify the number of particles in the resulting stack using **EMAN2**'s **e2iminfo.py** command.

At the terminal, type:

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

Depending on the settings you used during particle picking, the dataset processed here should contain about 8500 to 12000 single particles. The stack in the precalculated results contains 11003 particles.

You can also display the virtual stack using the utility **Display Data**. For this purpose, in the main window of the SPHIRE GUI click the ***Display Data*** button in the middle and load the stack. However, please keep in mind that display of a very large particle stack might result in a memory crash.

ISAC

2D Clustering

The 2D classification is one of the crucial steps of the workflow. It will give you fast a good idea about the quality and heterogeneity of the data and is a great tool to sort out bad particles. Thus, it is a key step towards a high-resolution structure determination. In other packages, this is mostly done during computationally expensive 3D classifications, which usually have to be performed multiple times. In contrast, we prefer to conduct 3D analysis using datasets that were pre-cleaned in 2D and, as a result, we usually need to perform heterogeneity analysis in 3D only once.

SPHIRE uses the **ISAC** method (Yang et al. 2012) to perform 2D alignment and clustering and to determine validated and homogeneous subsets of images with minimal human intervention. Due to the high number of reproducibility tests during the validation procedure, **ISAC** used to be (pre-2016 version) very time- and memory-consuming. The current, optimized **ISAC2** version delivers results for data sets of up to 700000 and it is 10 × to 15 × faster than the pre-2016 version.

The ISAC program will:

⇒ Phase-flip the 2D images in the stack, if requested.

⇒ Establish initial translation parameters using a reference-free approach, if requested.

⇒ Perform equal size K-means clustering, run a group stability test and output stable class averages, while setting aside images that could not be accounted for by the current classes. The determined class averages and their associated images, called accounted for, are returned as output. The remaining, unaccounted for images, are inputted to the next round of **ISAC** clustering (called generation). The program will terminate if no further stable class averages can be produced. Finally, the particle members of stable output class averages are returned as "Processed", while those that could not be accounted for by determined class averages are returned as "NotProcessed".

In the main window of the SPHIRE GUI click button **ISAC2** on the left and then the ***ISAC2 - 2D Clustering*** button in the middle.

Fill out the following input fields:

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

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

  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 was 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 particular, **RELION** data files (stacks and metainformation stored in .star files) can be directly converted using the **sxrelion2sphire.py** utility. To access this tool, in the main window of the SPHIRE GUI click the pictogram **UTILITIES** on the left and then the **RELION to SPHIRE conversion** button in the middle. Detailed instructions are provided in the **Import a star file** page of the wiki.*

- **Output directory:** Class2D
- **Particle radius [Pixels]:** 145
- **Images per class:** 100

  NOTE: *Our dataset contains about 8,800 particles, thus by setting this parameter to 100, the expected number of 2D averages will be about*

$$\textit{Number of classes} = \frac{\textit{Total Number of Particles}}{\textit{particles per class}} = \frac{11003}{100} \approx 100\,.$$

TIP: *Proper setting of the desired number of particles per class requires some experience and running multiple **ISAC2** runs with different settings might be necessary in order to obtain the most optimal results. 100 to 200 particles per class is a good starting point for cryo-EM datasets of high quality containing about 50 to 100000 particles, but for early exploratory analysis of very large, autopicked datasets one may want to consider using a larger number (i.e., 500 to 2000) particles per class. Moreover, depending on the appearance of the resulting class averages, one may need to adjust this value accordingly. For example, if the resulting averages are excessively noisy, one might want to repeat **ISAC2** run using a larger number of particles per class. However, this is always at the expense of the amount of detail in averages: the larger the number of members per averages the worse the amount of discernible details in averages. The computational time of **ISAC2** is a steeply raising function of the number of class averages and to obtain high-quality homogeneous (i.e., with possibly few members) class averages one has to be prepared to invest significant time and resources. It should be also emphasized that in the next step of ab initio structure determination (**VIPER**) one will need at least 100 to 150 high quality class averages to produce a reliable 3D model.*

TIP: *The final number of particles per class and the number of classes are determined by the algorithm itself based on its ability to align them and thus form stable class averages. The number does not depend on the number of input images but rather on their quality and heterogeneity of the set, i.e., the number of distinct views. Setting the number too high, which typically means more than few hundred, may result in heterogeneous (a.k.a. "fake") class averages and/or rejecting rare views by the program.*

TIP: *Use 50 to 150 members per class for negative stain datasets of*

5000 to 15000 particles.

- **CTF phase flipping:** YES

  TIP: *Deactivate this flag for negative stain data.*

- **Translation search range [Pixels]:** 1
- **Pixel error threshold [Pixels]:** 0.7

  NOTE: ***ISAC2*** *evaluates quality of class averages by performing multiple reference-free ab initio alignments of images within a class. Orientation parameters of individual images determined in these independent runs are compared and their dispersion is reported as **pixel error** of a given class (or image). Class averages whose **pixel errors** exceed the pre-set threshold are discarded and their image members are set aside for processing in the subsequent generation. **pixel error** is thus one of the most important **ISAC2** parameters.*

TIP: *For **ISAC2** runs with large datasets of excellent quality and low number of particles per class (e.g. 100), one should use a rather conservative value for this threshold (0.7). With increased number of particles per class or in case you expect a high degree of flexibility in your particle, you should set a higher threshold value (for example 1.7), as otherwise **ISAC2** might discard too many particles.*

- **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 are resized to match the user-provided particle radius and image size. The default and thoroughly tested values are 29 pixels radius and the box size 76 pixels, respectively. Thus, the resulting class averages will have a new (usually larger) pixel size and most probably high-resolution features (such as secondary structure elements) will not be visible. The higher-resolution information will be restored later in the subsequent Beautifier step. The useful conversion formula is:*

$$\textit{Target particle radius} = \frac{\textit{Target particle image size} \times \textit{Original particle radius}}{\textit{Original particle image size}}$$

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

Specify the number of processors (for this job we used 96) and submit the job to the queuing system of the cluster using an appropriate submission script by clicking the ***Run command*** button. Monitor progress of the job through the standard output. On our cluster, this job with 96 processors finished after about 30 minutes.

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

```
cp -Rp SphireDemoResults/Class2D ./
```

Once the job finished, the Class2D folder will contain the following folders and files:

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

  NOTE: *Display the total-average of your dataset after the pre-alignment (**Class2D/2dalignment/aqfinal.hdf**) using the **Display Data** utility.*

  TIP: *Confirm that the 2D total-average is properly centered and that the circular 2D mask covers the entire particle. If this is not the case, increase the particle radius (project-wide parameter) and rerun **ISAC2**.*

– **class_averages.hdf:** Contains the final stable class averages.

  TIP: *If the file is not produced or contains only few class averages, it means that given the quality of the data the value of **Pixel error threshold [pixels]** parameter was set too low and thus too many class averages were eliminated during the stability tests. In this case, you can try to increase the value of **Pixel error threshold [pixels]** and run **ISAC2** again. If this does not help either, consider recording a dataset with better contrast and/or improve the quality of the autopicking. In most cases it is not possible to determine a reliable structure from a dataset that does not yield good (i.e., with distinct features) 2D class averages. In case of a very "dirty" autopicked dataset, a second **ISAC2** run might be necessary to obtain the best results.*

  TIP: *To determine the pixel size of the **ISAC2** class averages, divide the original pixel size by the shrink ratio. The shrink ratio can be found in the file **README_shrink_ratio.txt** located in the **Class2D** folder.*

– **processed_images.txt:** Contains the particle IDs of the members of the stable class averages.

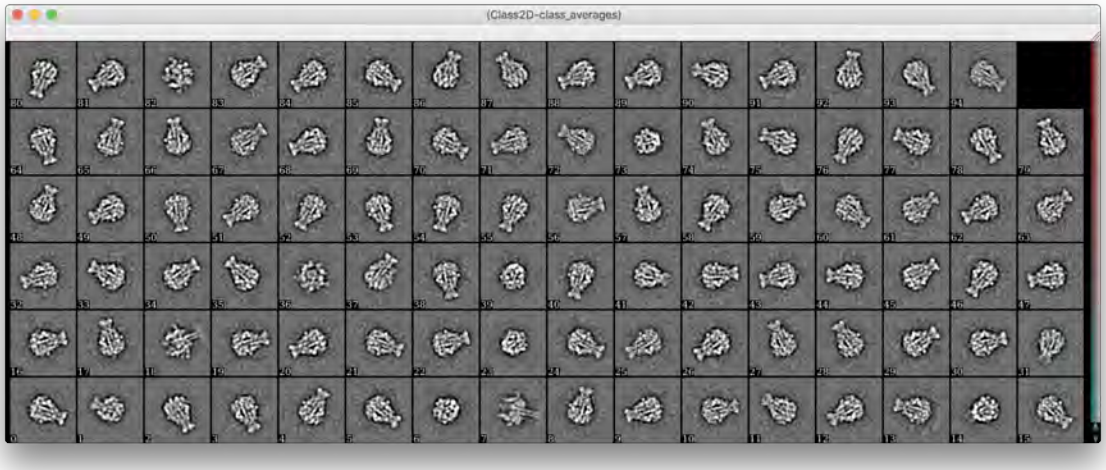
  NOTE: *At the terminal, type: **wc –l Class2D/processed_images.txt**. This command will count the number of lines in the specified file. This corresponds to the total number of particles assigned to stable class averages. For the precalculated results this file has 9821 entries.*

– **not_processed_images.txt:** Contains the particle IDs of the particles not assigned to stable class averages.

  NOTE: *At the terminal, type: **wc –l Class2D/not_processed_images.txt** The printed number corresponds to the total number of particles not assigned to class averages. For the precalculated results **ISAC2** discarded 1503 "bad" particles.*

NOTE: *In case **ISAC2** run completed at least one main iteration but crashed afterwards for some reason, it is possible to restart calculations using the "Continue mode". Open the **ISAC2 - 2D Clustering** page of the SPHIRE GUI and enter the exact same settings you used for the unfinished **ISAC2** run. Then click the "advanced settings" button, set the "Restart Run" parameter at the bottom to 0 and then click the **Run command** button to submit the job. The program will identify the last fully completed main iteration and continue from there.*

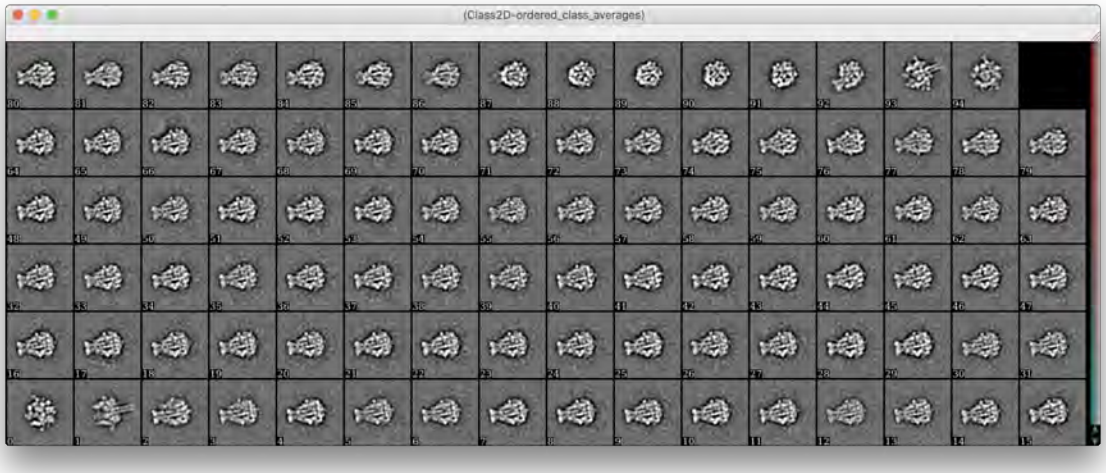
Display the final stable class averages (**Class2D/class_averages.hdf**) using the ***Display Data*** utility.



The high quality of this dataset is reflected by the impressive quality of the 2D class averages. Please note, that as mentioned above, in order to speed up the process, the particles of the input dataset are resized to a large pixel size and high-resolution features are not visible. However, the stable 2D class averages will be further processed in the subsequent **Beautifier** step and high resolution information will be restored.

To facilitate a better comparison between the stable class averages, **ISAC2** realigns them relative to each other and orders them based on pair-wise similarity. The result is stored in the file: **Class2D/ordered_class_averages.hdf**

Display the ordered and aligned final stable class averages (**Class2D/ordered_class_averages.hdf**) using the **Display Data** utility.



Known issue in the release

We optimized the parallelization of **ISAC2** and the current **ISAC2** version can now handle large datasets. On our cluster with 128 GB RAM and 24 cores per node, we managed to successfully process datasets with up to 750000 particles. However, larger datasets or large datasets of lower quality might still fail due to insufficient memory. The suggested workaround is to split the data into subsets, run **ISAC2** for each subset separately, as described above, and then combine the results. For example, to split a dataset in 4 subsets type at the terminal (it is one long line command):

```
n=4; for i in $(seq 0 $((n-1))); do e2bdb.py bdb:Particles#stack
--makevstack=bdb:Particles#stack_${i} --step=${i},${n}; done
```

The stack bdb:Particles#stack_0 contains afterwards every fourth particle of the original stack starting from 0, the stack bdb:Particles#stack_1 contains now every fourth particle of the original stack starting from 1 and so on.
This command creates 4 virtual particle stacks, but you can easily change the number of substacks by changing the 4 in the beginning of the command to the value wanted.
To combine the clean stacks obtained from 4 runs of **ISAC2** into a single virtual stack, use the following before going on to the next step VIPER (it is one long line command):

```
e2bdb.py bdb:Particles#isac_substack_1 bdb:Particles#isac_substack_2
bdb:Particles#isac_substack_3 bdb:Particles#isac_substack_4
--makevstack=bdb:Particles#stack_all
```

In case you do not have enough time time to perform the described steps, copy our precalculated **ISAC2** results and the resulting clean dataset to the **project directory**.

```
cp -Rp SphireDemoResults/Class2D ./
```

```
cp -Rp SphireDemoResults/Particles ./
```

Most class averages show the typical pineapple-like shape of the side-view of the prepore state of the TcdA1 toxin (Gatsogiannis et al. 2013). However, two class-averages (Nr. 1 and 93) differ considerably; they show an umbrella-like shape and correspond to the pore state of the complex (Gatsogiannis et al. 2013; Gatsogiannis et al. 2016). Thus, the present dataset includes a mixture of two populations corresponding to two different conformational states of the same complex in a ratio of approximately 19 to 1.

We will now delete these two class averages using the ***Display Data*** Utility. For this purpose, in the main window of the SPHIRE GUI click the **Display Data** button under **UTILITIES** and load the file **Class2D/ordered_class_averages.hdf**.

> **TIP:** *Usually, at this step we delete only "junk" class averages (e.g. blurry class-averages, ice contamination, carbon-edge etc.) and we let **RVIPER** identify and exclude outliers during the initial model generation procedure. However, in this the case the structural differences are very obvious, therefore we will delete these two classes right away.*

To delete "bad" class averages or outliers, press the mouse wheel button somewhere on the graphics window and activate the ***Del*** button (1), in the pop-up window.

Next, select the "bad" class averages by clicking on the respective images (2) and click the ***Save*** button (3) to save the remaining images under a new name: **Class2D/best.hdf**.

> **TIP:** *In the pop-up graphics window, you can also click the* ***Values*** *button and select the* ***n_objects*** *parameter in order to display the number of images for each class average.*

These class averages (**Class2D/best.hdf**) will be used to generate an initial 3D model with **RVIPER**.

Beautifier

In order to speed up calculations, we resized the particles of the input dataset during **ISAC2** to a smaller box and a larger pixel size. Most importantly, we did not apply full CTF correction; instead, images were only phase-flipped. Therefore, in most cases, because these operations remove high-resolution information from the data, possible fine details (such as secondary structure elements) will not be visible in the **ISAC2** 2D class averages.

The **Beautifier** tool extracts members of each **ISAC2** class average from the original, full-size dataset, performs local 2D refinement, applies full CTF correction and finally aligns and orders averages by pair-wise similarity. It is also possible to request a low-pass filtration to supress high-resolution noise. Thus, the outcome of the **Beautifier** tool contains full-size and averages computed at fully attained resolution.

> **TIP:** *High-resolution datasets will in most cases yield "beautified" class averages with distinct high-resolution features. Thus, the results of the current processing step provide a good checkpoint to verify that the project has the potential for high-resolution 3D structure determination.*

> **TIP:** *It is not necessary to use the Beautifier tool for negative stain data.*

In the main window of the SPHIRE GUI click the ***ISAC2*** button on the left and then the ***Beautifier - Adjust to Model*** button in the middle.

Fill out the following input fields of the GUI:

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

  NOTE: *Click the **Select BDB image stack** button and use the file browser to select the virtual stack containing all particles in the **Particles** folder. You **have to** select the stack used as input to perform the 2D Classification with **ISAC2**.*

- **ISAC2 run directory:** Class2D

  NOTE: *Click the **Select directory** button and use the file browser to select the **ISAC2** output directory.*

- **Output directory:** Beaut

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

  NOTE: *You have to use the original particle radius (145) and **not** the target particle radius used during **ISAC2** (29).*

- **CTF correction:** YES

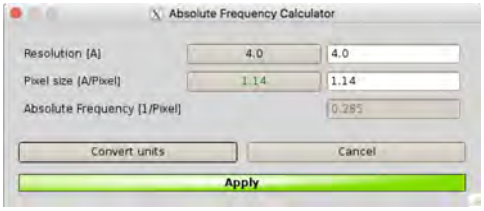
  NOTE: *Activate the checkbox to apply full CTF correction. Do not activate in case you process negative stain data.*

- **Local alignment:** YES

  NOTE: *Activate the checkbox to perform a local alignment of the **ISAC2** averages. The "beautified" class averages will improve, but also the computational time will increase.*

- **Low-pass filter cutoff resolution [1/Pixel]:** 0.285

  NOTE: *Specify the cutoff for the low-pass filter to be applied (in absolute frequency units [1/Pixel]). You can also use our calculator to convert resolution in angstrom to absolute frequency.*



  *For this purpose, click the **Use resolution[A]** button; in the pop-up window set the resolution to 4 Å and then click the **Convert units** button. A resolution of 4 Å corresponds to an absolute frequency of 0.285 (by the given pixel size of1.14 Å).*

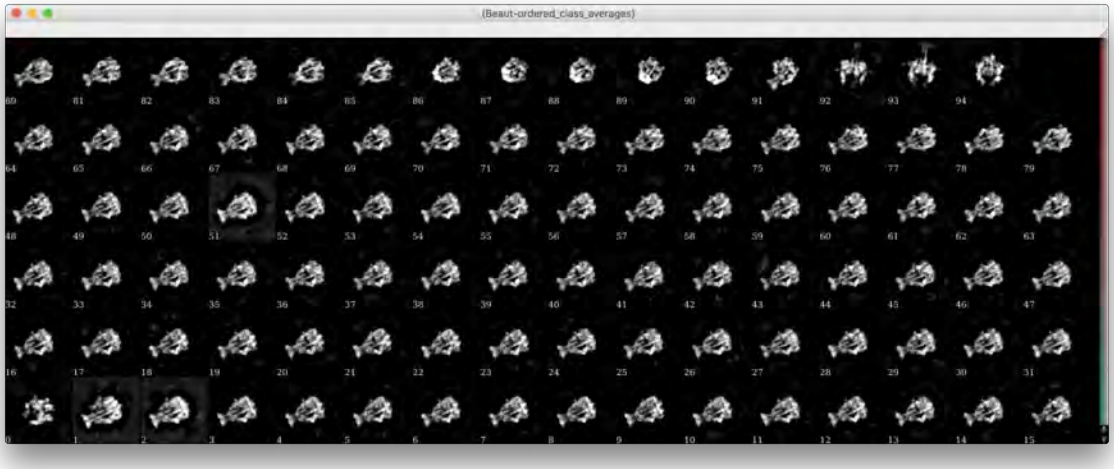
  The conversion formula is:

$$Absolute\ frequency = \frac{Resolution}{pixel\ size}$$

  *Now click the **Apply** button to set this value in the low-pass filter frequency input field.*

  NOTE: *According to the above setting, the "beautified" class averages will be low pass filtered to 4 Å [Absolute frequency 0.285]. If you do not wish to low-pass filter the averages set this parameter to 0 instead. Setting this parameter to -1, will low-pass filter to the resolution automatically determined from the **ISAC2** averages **before** the local refinement, whereas by setting it to -2, the program will low-pass filter to the resolution automatically determined from the **ISAC2** averages **after** the local refinement.*

Specify the number of processors (for this job we used 48) and submit the job to the queuing system of the cluster using an appropriate submission script by clicking the ***Run command*** button. On our cluster this job took about 12 minutes using 48 processors. Display the final stable, resized, refined, power spectrum adjusted and CTF corrected, ordered class averages (**Beaut/ordered_class_averages.hdf**) using the **Display Data** utility.



Click the **mouse wheel button** while pointing somewhere on the graphics window and adjust the **brightness** and the **contrast** at the **e2display.py** pop-up window. The high quality of this dataset is reflected again in the high level of detail of the beautified 2D class averages. Delete the class averages of the pore state of the complex, as it was described in the previous section, and store the remaining images in a new file named "**Beaut/ best.hdf**".

> **TIP:** *Direct visualization of secondary structure elements in 2D class averages requires a sufficiently large dataset and a class-size of 1000 to 2000 members.*

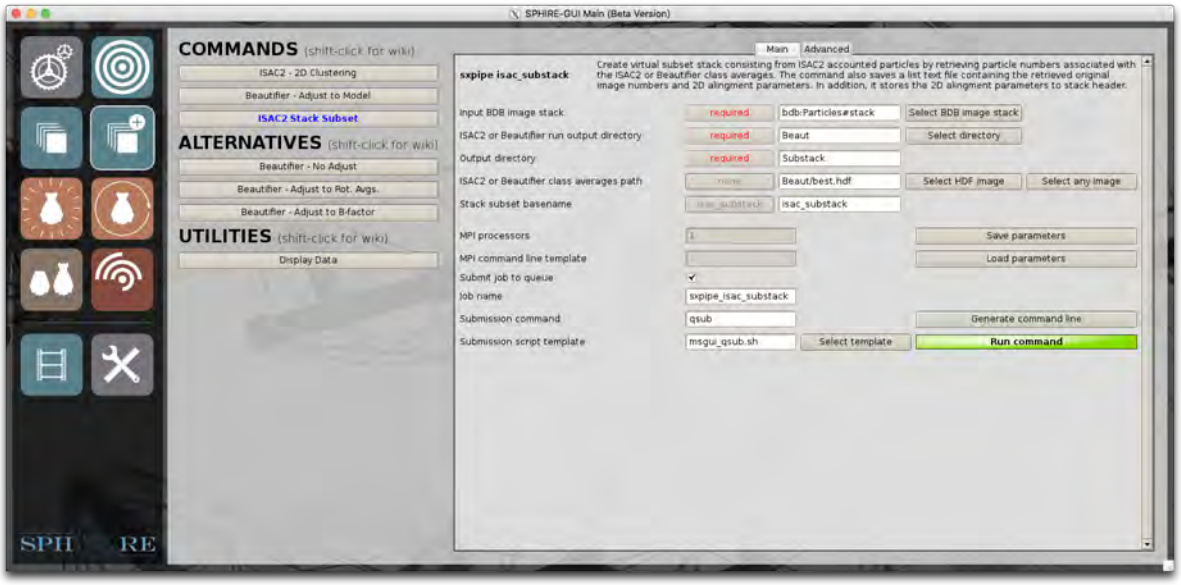
Beautified class-averages can also be used to calculate an initial model using **RVIPER**. In several cases we used them to obtain sub-nanometer structures, and thus better initial models. However, running time of **RVIPER** increases significantly with the box size, therefore we generally recommend using the down-scaled **ISAC2** class-averages as input for **RVIPER** instead.

Create Stack Subset

The initial model that we will calculate in the next step (**RVIPER**) will be refined against the particles members of the selected class averages (the "clean" stack). This will be done in our **MERIDIEN** 3D structure refinement.

Now, we will create a virtual "clean" particle stack that will contain only the members of the selected class averages (**Beaut/best.hdf**).

It should be emphasized that the **Beautifier** tool performs a local 2D refinement of the particle members of the beautified class averages and centering is in general improved. The resulting orientation parameters can be passed to the 3D **MERIDIEN** refinement helping to accomplish higher resolution of the final map at a shorter time of calculations. Therefore, we will store in the header of the particles of the "clean" stack, the 2D alignment parameters. In the main window of the SPHIRE GUI click the button ***ISAC2*** on the left and then the ***ISAC2 Stack Subset*** button in the middle.



Next, fill out the following input fields:

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

  NOTE: *Click the **Select BDB image stack** button and use the file browser to select in the **Particles** folder the virtual stack that contains all particles.*

- **ISAC or Beautifier run output directory:** Beaut

  NOTE: *Click the **Select directory** button and use the file browser to select the output directory of the **Beautifier** run. The program will import from this directory the 2D parameters of the **Beautifier** local refinement and store them to the header of the extracted particles.*

  TIP: *In case you skipped the **Beautifier** step, you should select the **ISAC2** output directory (**Class2D**) instead. However, in this case the 2D parameters stored in the header of the resulting "clean" stack will be imported from the **ISAC2** 2D alignment parameters. Keep in mind that since **ISAC2** performs the alignment and clustering on downscaled images, the x-, y-shifts will be less precise, the 3D refinement will not start from the best possible centering parameters and might need more iterations to converge.*

- **Output directory:** Substack
- **ISAC2 or Beautifier class averages path:** Beaut/best.hdf

  NOTE: *Click the **Select HDF image** button and use the file browser to select the file that contains the selected class averages.*

- **Stack subset basename:** isac_substack

  NOTE: *Particles that are members of selected class averages will be stored in a virtual BDB file named **isac_substack** in the specified output directory (**bdb:Substack#isac_substack**).*

Click the ***Run command*** button and check the output of the job.

At the terminal, type:

```
tail -f name_of_sxpipe_logfile

018-02-07_12:14:59 =>Summary of processing...

2018-02-07_12:14:59 =>Particles in fullstack : 11003

2018-02-07_12:14:59 =>Accounted particles : 9441

2018-02-07_12:14:59 =>Default class averages : 95

2018-02-07_12:14:59 =>Provided class averages : 93

2018-02-07_12:14:59 =>Extracted class members : 9259

2018-02-07_12:14:59 =>ISAC substack size : 9259
```

The “clean” stack contains 9259 particles. In case of this high-quality dataset, we eliminated about 16 % of 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 a smaller but cleaner dataset.*

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

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 **ISAC2**.

First, check the number of your selected **ISAC2** class averages.

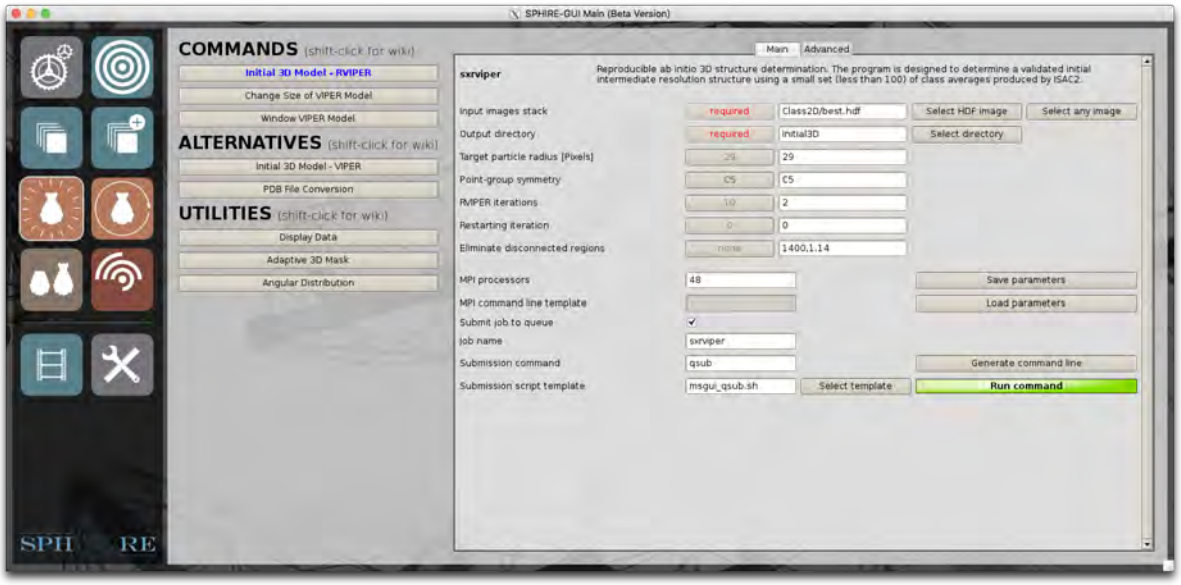
```
e2iminfo.py Class2D/best.hdf
```

In general, about 150 high quality class averages are sufficient to obtain a validated initial structure in a reasonable amount of time. Although **RVIPER** would likely provide better results with more than 150 images, computational time would increase significantly and in most cases there would be no noticeable improvement of the resulting model.

After **ISAC2**, we deleted the averages of the second population. The output file **best.hdf**, contains only 93 class averages (the exact number of class averages can slightly differ due to internal randomization used in the initialization of the **ISAC2** clustering process). However, the test complex has C5 symmetry, therefore the available number of class-averages should be sufficient to calculate a reliable 3D model with **RVIPER**.

> **TIP:** *In order to run **RVIPER** successfully for cryo data it is necessary to have at least 60 to 80 high quality averages with about 100 to 200 members each (for negative stain about 100 members). Be sure that the input class averages include as many orientations of the particle, as possible. It will not be possible to calculate a structure from images dominated by one (or few) preferred orientation. If class averages contain only few members or have a low signal to noise ratio, one might need to use more than 150 class averages for **RVIPER**, particularly in case of asymmetric complexes.*

In the main window of the SPHIRE GUI, click button **VIPER** on the left and then the Initial ***Initial3D Model - RVIPER*** button in the middle.



Fill out the following input fields:

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

  IMPORTANT: Use the same target particle radius as the one used for **ISAC2**.
- **Point-group symmetry:** C5

  TIP: *Use C1 if the symmetry of the complex is not known.*
- **RVIPER iterations:** 2

  TIP: *The optimal number of iterations depends on the starting number of class averages. Our standard settings are: 100 averages: 2 iterations; 400 averages: 5 iterations.*
- **Restarting iteration:** 0

  NOTE: *In case your **RVIPER** run crashed for some reason, set this parameter to 0 and resubmit the job. The program will then identify the latest fully completed iteration and continue from there.*
- **Eliminate disconnected regions:** 1400,1.14

NOTE: *Enter the approximate molecular weight of the complex (kilo dalton) and the pixel size (angstrom), separated by a coma. Based on these values the program will compute, after each iteration, the threshold at which the candidate structure occupies the expected volume (in voxels). Any disconnected pieces that will appear at this threshold will be eliminated and the corrected 3D structure will be used as reference for the next iteration. The intention is to force the program to favour candidate structures that have "structural integrity", as reasonably expected for physically plausible macromolecular complexes.*

NOTE: *If the molecular weight is underestimated or there is strong complex flexibility the program may eliminate valid regions of the structure, a problem particularly common with negative stain data; in this case, try to increase the molecular weight value and restart the program. If in doubt, one can also perform two runs of **RVIPER**: one with elimination of disconnected regions, the other without and compare the results.*

TIP: *To suppress the above functionality, enter "none" into the input field.*

Specify the number of processors (for this job we used 48) and submit the job to the queuing system of the cluster using the appropriate submission script and by clicking the ***Run command*** button. Monitor the progress of the job through the standard output. On our cluster this job required about 27 minutes to complete.

NOTE: *Note, that the number of processors needs to be multiplicity of the advanced settings option **GA population size** (default 4).*

Known issue

The program might crash if the number of processors exceeds the number of class averages.

RVIPER (**R**eproducible **VIPER**) performs multiple **VIPER** runs (*ab initio* 3D structure determination using a small set of ISAC2 class averages) and thus calculates multiple initial models. It performs a reproducibility test, removes unstable projections and produces a single validated initial model.

After the program finished, output **main**ITERNR folders are placed in the **Initial3D** output. The number of **main**ITERNR folders corresponds to the number of **RVIPER** iterations. Folders named **run**ITERNR within each main folder contain the results of each independent **VIPER** run. In our case there is only folder: **main001**. The final reproducible structure is **Initial3D/main001/average_volume.hdf**.

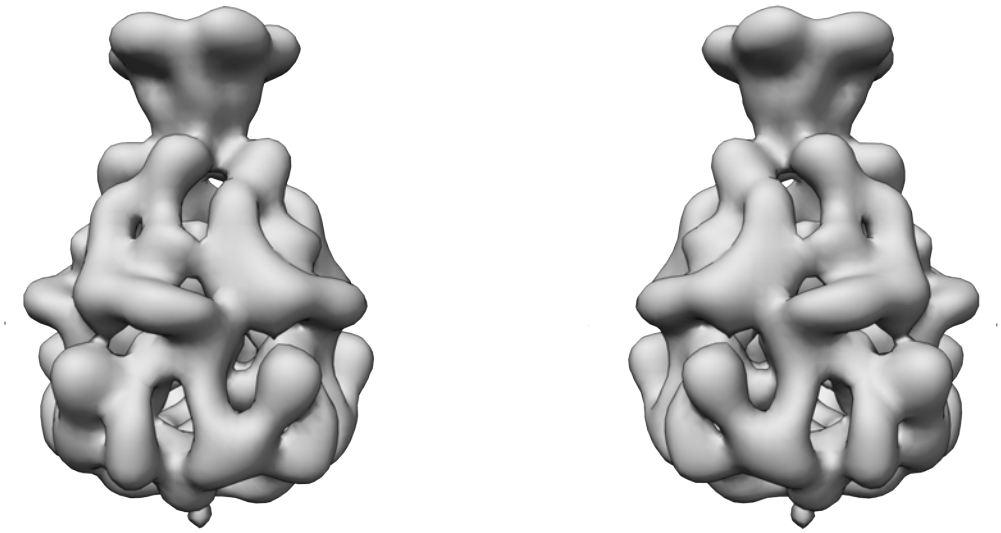
TIP: *If the internal stability criterion is not met after 10 independent **VIPER** runs, the program will stop (it will not continue to the final iteration) and no average structure will be created. However, we still recommend careful examination of all output structures from the independent runs (i.e., **volf.hdf** and **refvol2.hdf** in the subdirectories **main001/run**ITERNR) as some of them might fit the input data sufficiently well and thus might serve as suitable*

*initial models in the subsequent 3D **MERIDIEN** structure refinements. Nevertheless, it should be emphasized that these structures are not validated and therefore their quality has to be assessed visually, in particular their structural integrity, i.e., presence of "moons" (significant large densities) floating around the main body of the structure can indicate problems.*

Please keep in mind that one cannot determine hand of a 3D structure based on the set of its 2D projections alone. Therefore, **RVIPER** might produce an enantiomer (mirror structure). There is no way to establish correct handedness of low-resolution models short of performing tilt experiments.

TIP: *In case neither the handedness of the complex is known nor a homolog structure is available, you should either perform a tilt experiment or proceed with 3D refinement using the obtained **RVIPER** model. Once sufficient resolution is accomplished, you can establish the correct hand based on the visual appearance of secondary structure elements.*

In our case the hand of the test structure is known and therefore the handedness of the map obtained from the current **RVIPER** run can be set correctly at this point. For this purpose, display the **RVIPER** map using the molecular graphics program **UCSF Chimera** (Pettersen et al. 2004a).



Correct hand | Wrong hand

Compare the appearance of the obtained **RVIPER** maps with the images above and check if your map has the correct handedness. If not, it is necessary to mirror the structure about the x-y plane (that is along z-axis) using **EMAN2**'s **e2proc3d.py** command (one long command).

```
e2proc3d.py Initial3D/main001/average_volume.hdf Initial3D/main001/average_volume_flip.hdf
--process xform.flip:axis=z
```

To compare the **RVIPER** structure with the available X-ray crystallographic 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 (see Resize VIPER Model section below for details) and fit the atomic model 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 during the **VIPER** 3D reconstruction, as described on our SPHIRE wiki page. The file can also be displayed in **UCSF Chimera**.

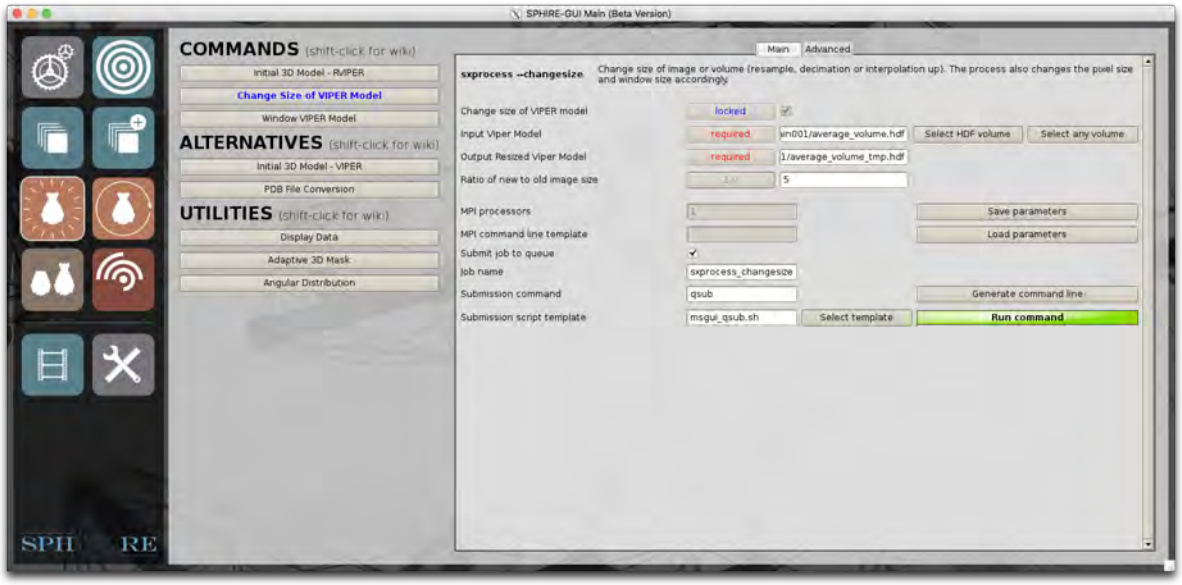
> **TIP:** *Alternatively, to the procedure described above, instead of using the downscaled **ISAC2** class-averages as input for **RVIPER**, one can also calculate an initial 3D model directly from the beautified class averages with the original pixel size. However, the computational time for **RVIPER** on averages of original image size increases with the square of their size. Depending on the quality of the averages, the advantage of running **RVIPER** on original image size is that you might be able to obtain directly from beautified class-averages initial models at 6 Å to 9 Å. For this approach however, you will have to set the **Low-pass filter frequency [1/Pixels]** (**Advanced** option) (filter to be applied to the **RVIPER** structures) to an appropriate value before submitting the job.*

Resize VIPER Model

The particles of the dataset were resized during **ISAC2** (in this case from a radius of 145 pixels and box size of 352 pixels to a radius of 29 pixels and box size of 76 pixels). Therefore, the **ISAC2** averages and the **VIPER** 3D model have now an increased pixel size of 5.7 Å. In order to use this model as a reference structure for the 3D refinement of the original dataset, we have to resize and window the **VIPER** model in order to match the dimensions and the pixel size of the original particle stack. To check the shrink ratio used to downscale the images before **ISAC2** type:

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

Now click the ***Change Size of VIPER Model*** button in the middle of the SPHIRE GUI.



Fill out the following fields:

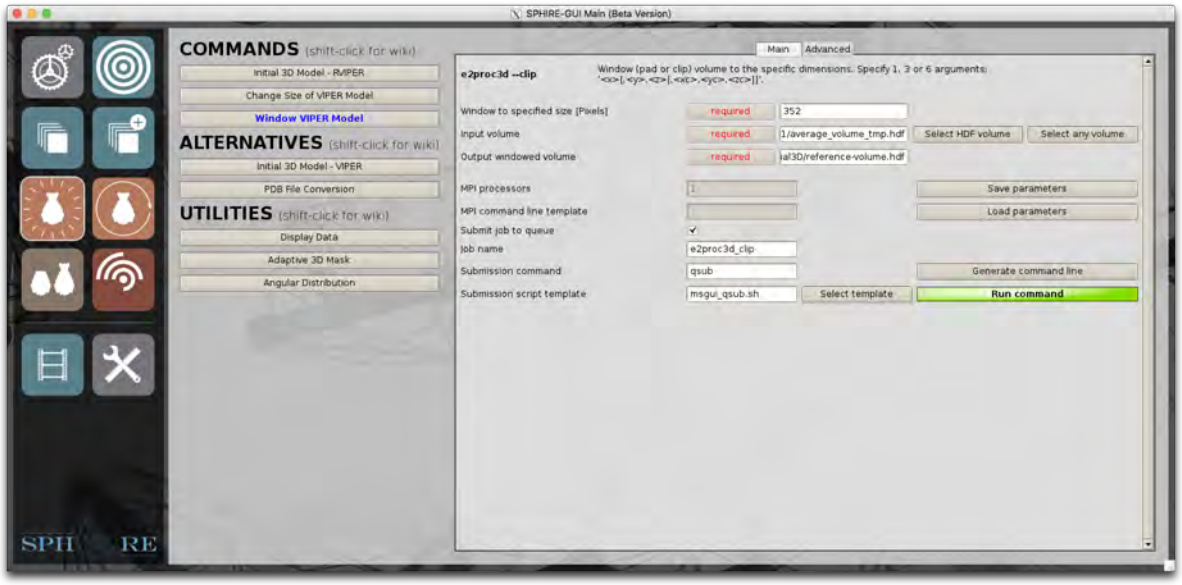
- **Input Viper Model:** Initial3D/main001/average_volume.hdf or Initial3D/main001/average_volume_flip.hdf
- **Output Resized Viper Model:** Initial3D/main001/average_volume_tmp.hdf
- **Ratio of new to old image size:** 5

  NOTE: *The shrink_ratio used in **ISAC2** is 0.2 and therefore:*

$$\textit{Ratio of new to old image size} = \frac{1}{0.2} = 5$$

Click the ***Run command*** button.

Now we will window this up-scaled volume to the original box size. In the middle of the SPHIRE GUI click the ***Window VIPER Model*** button:



Fill out the following fields:

- **Window to specified size [Pixels]:** 352
- **Input volume:** Initial3D/main001/average_volume_tmp.hdf
- **Output windowed volume:** Initial3D/reference-volume.hdf

TIP: *In case an X-ray structure of a homologous complex is available and there is a high-degree of similarity expected, one can omit the ab initio structure determination with **RVIPER** 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 density map.*

If you do not have enough time to perform these steps, you can also copy our precalculated results to the **project directory**.

```
cp -Rp SphireDemoResults/Initial3D ./
```

MERIDIEN

In the previous steps we removed all "junk" images and produced a clean subset of particles that was aligned and clustered in a stable and reproducible manner using the powerful **ISAC2** approach. Subsequently, from the validated **ISAC2** class averages we calculated a reproducible model using **RVIPER**. These results give us confidence that we can now proceed with the structure refinement using the obtained model as a starting reference (3D Refinement; **MERIDIEN**).

MERIDIEN employs a quasi-Maximum Likelihood approach (later referred to as ML). Briefly, the set of 2D input images is randomly split into half-datasets and is refined quasi-independently in order to minimize noise bias and over-refinement. During projection-matching phase the program estimates "probabilities" with which each particle image matches reference reprojections of the current approximation of 3D structure. Next, during 3D reconstruction phase, the data is backprojected using all significant angular (and translation) directions with probabilities used as weights. The two steps are repeated iteratively. Note both the maps and resolution curves (driver FSCs) generated during each iteration of the program serve only as internal approximations (i.e., the resolution reported after each iteration is usually underestimated). Usable results are obtained after the refinement finished, using the ***PostRefiner*** utility that processes the so-called unfiltered final half-volumes to produce the ultimate result, including the FSC curve.

MERIDIEN provides the user with the following functionalities:

1. The standard use is to perform full 3D structure refinement initialized with an initial user-provided reference structure. In this case the program will start from exhaustive searches without assuming any angular information, even though it is possible to provide initial translation parameters to improve the convergence.
2. Local refinement mode is meant mainly to refine subsets obtained by 3D sorting. This mode is described in detail in the subsequent sections. It can also be used to locally refine datasets obtained by other software packages, as long as orientation parameters (both angles and translations) are stored in input image headers in the expected format.
3. 3D reconstruction of full-size unfiltered maps using parameters obtained from a user-specified iteration of **MERIDIEN** refinement.
4. Restart mode, which allows user to continue refinement (either standard or local) should the program run into time limit or crash.

We will now use mode (1) and perform a high-resolution 3D refinement starting from a **RVIPER** structure as reference, but first we will create a 3D mask to be used during the refinement.

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

Adaptive 3D Mask

In the main window of the SPHIRE GUI, click the button **MERIDIEN** on the left and then the **Adaptive 3D Mask** button in the middle (**UTILITIES** section):

Fill out the following fields:

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

  NOTE: *This is the **RVIPER** final result after resizing and windowing to the original box/pixel size.*

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

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

- **Mask extension cycles:** 3

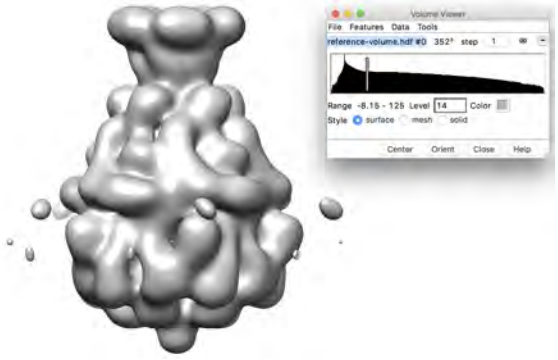
NOTE: *Numbers of cycles to extend this initial binarized structure; see the section below for instructions.*

- **Gaussian kernel size:** 11
- **Kernel standard deviation:** 9

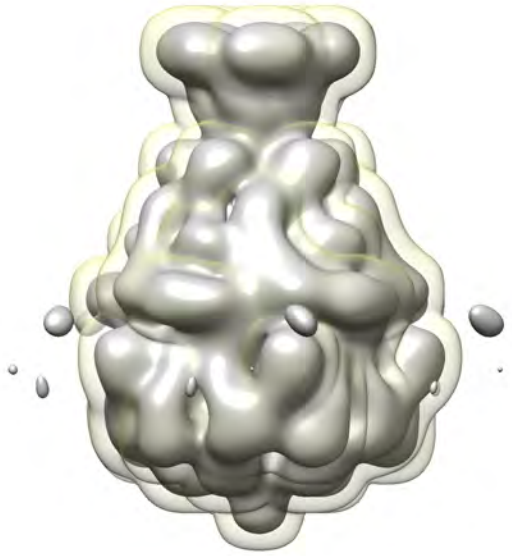
Click the ***Run command*** button. This process is usually finished after few minutes. Detailed description regarding the usage of this program is provided on our wiki page.

Density Binarization and Mask Extension

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



The generated mask should have the shape of the particle, but it should extend in all directions by a sufficient number of voxels (3 to 5) to prevent masking artifacts during 3D refinement. The amount of extension is controlled by the mask extension cycles parameter (set to 3 cycles; you may have to adjust this value). Check the size of the resulting mask relative to the initial map 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 map while excluding all satellite densities. In order to compare both maps in **UCSF Chimera**, set their voxel sizes to 1.14 Å (**Volume Viewer->Features->Coordinates->Voxel Size**: 1.14 Å).



3D Refinement

3D refinement of the structure is done using the original individual particle images. The standard default run (mode 1) starts from exhaustive searches using the resized **RVIPER** volume as initial reference structure. In the main window of the SPHIRE GUI click the button ***MERIDIEN*** on the left and then the ***3D REFINEMENT*** button in the middle and fill out the following input fields:

- **Input image stack:** bdb:Substack#isac_substack

  NOTE: *Click the **Select BDB image stack** button and use the file browser to select the unbinned subset of particles, containing the members of the selected **ISAC2** class averages.*

- **Output directory:** Refine3D

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

  NOTE: *Click the **Select HDF volume** button and use the file browser to select the resized and windowed **RVIPER** structure.*

  TIP: *Here you can also directly load any type of 3D structure as an initial reference (e.g. structure produced by another program, structure derived from X-ray crystallographic model, a sphere in case of high point group symmetry, etc.). The file that contains the structure does not have to have any header information. HDF is the primary SPHIRE format for 3D volumes, but .mrc, .spi, bdb:, .img and .dm4 are also supported and can be directly loaded to the GUI by clicking the **Select any volume** button. Keep in mind that the box and voxel size of this volume have to match the input dataset. If this is not the case, the volume has to be resized with the help of **VIPER** utilities described above.*

- **Particle radius [Pixels]:** 145
- **3D mask:** Initial3D/mask.hdf

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

- **Point-group symmetry:** C5
- **Starting resolution [A]:** 25.0

  NOTE: *The initial **VIPER** model will be low-pass filtered to this resolution to mitigate the initial model bias.*

  TIP: *In case you use an X-ray derived structure as a starting reference, a degree of caution is called for and in this case, you should apply a stronger low-pass filter (for example to 40 Å to 50 Å).*

  On the other hand, it should be emphasized that, based on the overall shape of the particle, too restrictive initial filtration may suppress discernible features and the program may fail to converge properly.

- **Initial angular sampling step [Degrees]:** 7.5

  NOTE: *Usually the default value of 7.5° is reasonable to create sufficient number of reprojections for the initial global parameter search for almost every asymmetric and/or low symmetry structures with initial resolution of 15 Å to 15 Å. Refinement of higher resolution initial models and/or structures with high point group symmetry may benefit from a smaller initial angular sampling step, but values lower/equal than 3.75° may result in excessively long time of calculations.*

- **Search range [Pixels]:** 5
- **Search step size [Pixels]:** 1

  NOTE: *Change of the two above default values 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 justified only in cases when previously determined orientation parameters are available; see option **Read shifts from header** below.*

- **Read shifts from header:** YES

  NOTE: *The 2D data should have translation parameters stored in headers, as computed during 2D alignment in **ISAC2**. Refinements that use pre-centered particles converge faster and produce better results. Therefore, this flag is set to **YES** by default and the predetermined **ISAC2** shifts are imported from the header automatically. If shifts are not available, set this flag to **NO**.*

TIP: *If the data were already subjected to 3D refinement and 3D orientation parameters are already available, you can also use the predetermined shifts (Euler angles are ignored, as we start from exhaustive search) to significantly accelerate the refinement process.*

In order to import previously determined 3D orientation parameters into the data file, use the following command (one long command):

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

- **Do 2D pre-alignment step:** NO

  NOTE: *If the data processing protocol was followed as described in this tutorial, the 2D data should have in headers translation parameters computed during 2D alignment in **ISAC2** and this option should be set to **NO**. If 2D alignment information is not provided, we recommend to set this option to **YES** in order to perform an initial reference-free pre-centering of the data. This significantly improves the performance by accelerating the convergence and improving the final resolution. Note that setting flag **Read shifts from header** (see below) to **YES** turns off the initial 2D pre-alignment.*

- **Memory per node [GB]:** 80

  NOTE: *Depends on cluster specifications. The program has to know the amount of physical memory available on each node as it uses "per node" MPI parallelization. Nodes are basic units of a cluster and each node has a number of processors. While clusters are often characterized by the amount of memory per processor, here we ask for the total amount of **Memory per node [GB]** as the program will internally adjust the number of processors it is using. For example, a cluster that has 3 GB memory per processor and 16 processors per node has 3 GB × 16 = 48 GB memory per node. It is advisable to use a number lower by about 20 % as part of the memory is not available to the program.*

IMPORTANT: Listed below are advanced options, please do not change the default values unless you understand their meaning well.

Advanced parameters:

- **Correlation peaks to be included [%]:** 99.9

  NOTE: ***MERIDIEN** will computationally match each image with template reprojections of the structure and convert the similarities into probability values. For good quality data and good agreement of data with the initial model we want to include all computed probabilities as this assures the best performance of the program. However, for lower quality data, the number of non-zero probabilities (and thus the "smear" of the correlation peak) might be excessively large and the running time of the 3D reconstruction step might become excessively long. Should that happen, it is advisable to terminate the program and lower the value of this parameter. 0.0 % value corresponds*

to usage of one orientation per image, thus to the "hard matching" refinement. While this setting will result in a very fast execution of the program, the result will be most likely disappointing. We found that for challenging cases setting of this parameter to 90 % to 95 % provides a good balance between time of calculations and quality of the result.

Specify the number of processors (on our cluster that has 24 processors per node, we used 96 processors) and submit the job to the queuing system of the cluster using an appropriate submission script by clicking the ***Run command*** button. 3D structure refinement with **MERIDIEN** is computationally demanding and the running time increases significantly with the number of particles and the box size.

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 concurrent refinements. The current resolution is then used to set main parameters of the refinement process (i.e., angular step, shift search range, maximum resolution and thus the window size) using a set of simple heuristics. In order to prevent premature termination at a suboptimal resolution stage, the program also uses heuristic randomization of the refinement process. Thus, repeated runs of the program from the same starting point will yield slightly different results, both in terms of orientation parameters and the ultimate resolution. One can monitor the progress of the job through the standard output, which will give 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.

```
Number of images : 9 48 18.8% 0.0min
```

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 script.*

```
2018-03-01_10:16:21 =>ITERATION # 1. Current state: INITIAL, nxinit: 52,
delta: 7.5000, xr: 5.0000, ts: 1.0000

2018-03-01_10:18:07 =>Resolution achieved in ITERATION # 1: 16/ 26 pixels, 25.08A/15.43A.

2018-03-01_10:18:07 =>ITERATION # 2. Current state: PRIMARY, nxinit: 120,
delta: 7.5000, xr: 5.0000, ts: 1.0000

2018-03-01_10:19:19 =>Resolution achieved in ITERATION # 2: 34/ 43 pixels, 11.80A/ 9.33A.

2018-03-01_10:19:19 =>ITERATION # 3. Current state: PRIMARY, nxinit: 96,
delta: 7.5000, xr: 5.0000, ts: 1.0000

2018-03-01_10:20:15 =>Resolution achieved in ITERATION # 3: 37/ 47 pixels, 10.85A/ 8.54A.
```

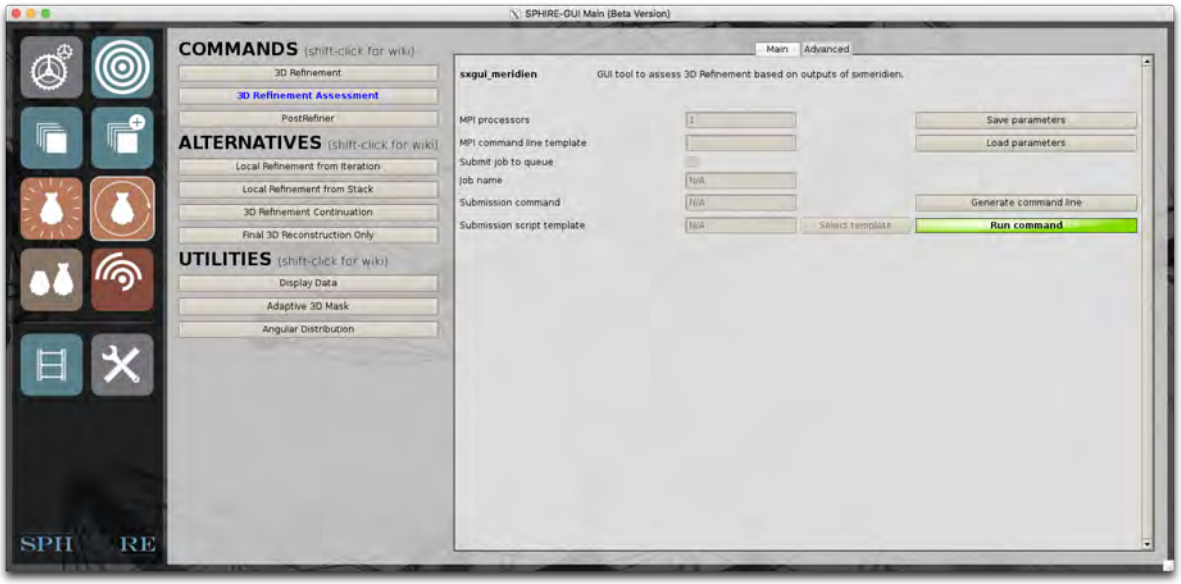
```
2018-03-01_10:20:15 =>ITERATION # 4. Current state: PRIMARY, nxinit: 104,
delta: 7.5000, xr: 5.0000, ts: 1.0000

2018-03-01_10:21:18 =>Resolution achieved in ITERATION # 4: 37/ 46 pixels, 10.85A/ 8.72A.

2018-03-01_10:21:18 =>ITERATION # 5. Current state: PRIMARY, nxinit: 104,
delta: 7.5000, xr: 5.0000, ts: 1.0000

2018-03-01_10:22:16 =>Resolution achieved in ITERATION # 5: 37/ 47 pixels, 10.85A/ 8.54A.

2018-03-01_10:22:16 =>ITERATION # 6. Current state: PRIMARY, nxinit: 104,
delta: 7.5000, xr: 5.0000, ts: 1.0000
```

The printout contains a sequence of pairs of lines for each iteration. They contain information about the time the iteration started, its number, the resolution achieved, both in Fourier pixels and in angstrom as read from driver FSC@0.5 and FSC@0.143. Note that the so-called driver FSCs computed during iterations and thus the resolutions printed are not the ultimate results but merely approximations calculated quickly for expediency to guide the program and used for adjustment of its parameters.

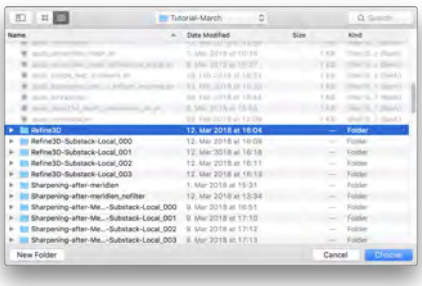
```
2018-03-01_10:24:38 =>ITERATION # 9. Current state: EXHAUSTIVE, nxinit: 104,
delta: 3.7500, xr: 4.5881, ts: 1.0518
```

For example, the line above informs us that ITERATION number 9 involved exhaustive searches and that the parameters used were: processing window size (**nxinit**) 104, angular step (**delta**) 3.75°, translational search range (**xr**) 4.588 1 pixels, and search step (**ts**) 1.051 8 pixels.

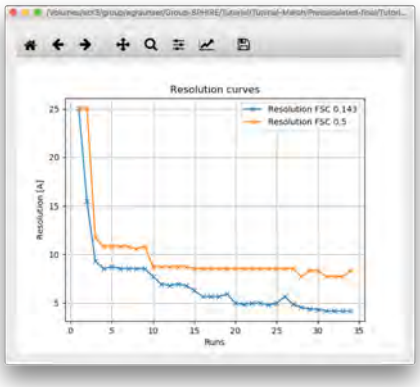
You can also follow the progress of the refinement and plot the driver FSC for each iteration using the **3D Refinement Assessment** GUI. For this purpose, click the ***Refinement Assessment*** button in the middle:



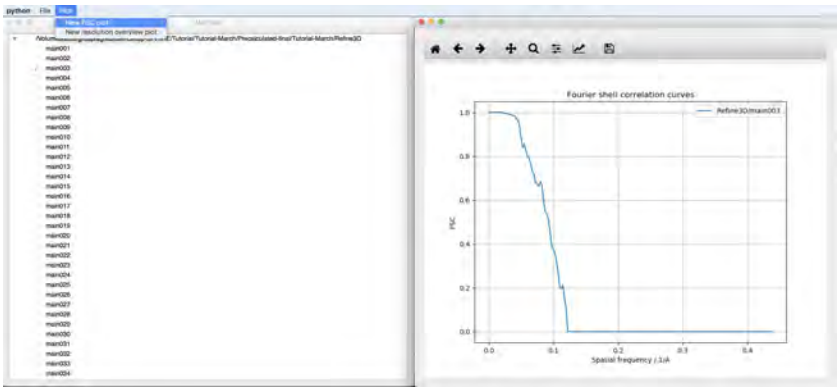
and then click the **Run command** button. In the Pop-up window select the current refinement directory (**Refine3D**):



After clicking the ***Choose*** button, the program will display a plot of two curves showing resolution as a function of iteration number, as determined using FSC@0.143 and FSC@0.5 criteria, respectively. For a currently running program, the plot will include information about already completed iterations. Once the next iteration is finished, a pop-up window will prompt you to update the plot.

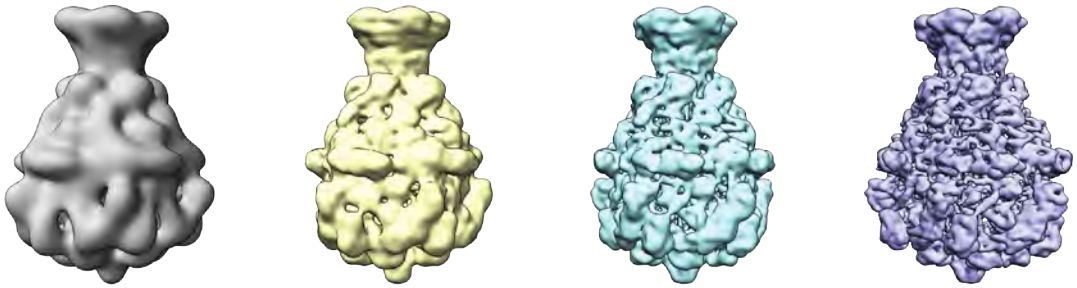


To display the driver FSC of a specific iteration, check its tick box (i.e., **main003**) at the main window of the **Refinement Assessment** GUI, click the ***Plot*** button and select the ***New FSC Plot*** option.



NOTE: *Again, please keep in mind the driver FSC@0.143 and driver FSC@0.5 computed during iterations do not reflect the ultimate results.*

TIP: *When we monitor the progress of the refinement, we usually examine the half-structures for selected iterations 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, you should not attach too much weight to their appearance and, in particular, to the apparent lack of details. Only the so-called "final" maps can be properly interpreted (see below for instructions how to compute them). If one is impatient to examine the details without waiting for the refinement to complete, one can enhance the details using the* ***PostRefiner*** *tool in single volume mode that is available in the main* ***UTILTIES*** *section of the GUI.*



Iteration	1	3	6	16
Resolution (0.5)(Å)	25	12.16	10.85	8.54

Once the program converged, it will output a statement and then calculate the final unfiltered, full size structures:

```
2018-03-01_11:19:12 =>Convergence criterion A is reached (angular step delta smaller than
3/4 changes in angles))

2018-03-01_11:19:12 =>Resolution achieved in ITERATION #34: 48/ 97 pixels, 8.36A/ 4.14A.
```

The final half-volumes are stored in:

Refine3D/vol_0_unfil_iternr.hdf and **Refine3D/vol_1_unfil_iternr.hdf**.

The final orientation parameters are stored in

Refine3D/final_params_032.txt.

Known issue in SPHIRE 1.0

The last iteration is memory intensive and the run may crash if sufficient memory is not available. check the **MEMORY ESTIMATION** entry for the final iteration in the **MERIDIEN** log file.

```
MEMORY ESTIMATION. memory per node = 80.0GB, volume size = 2.12GB, data size per node
= 0.59GB, estimated number of CPUs = 24
2017-11-15_16:30:46 =>do_final_rec3d
```

In this case Meridien estimated it required

$$24\,\mathrm{GB} \times 0.59 + 2.12\,\mathrm{GB} = 16\,\mathrm{GB}$$

and the available memory per node was set to $80\,\mathrm{GB}$, thus the program assumes there is sufficient memory to calculate the final full-size 3D reconstruction.
In case the program crashed due to insufficient memory, try to perform final reconstruction with the memory per node parameter set to a smaller value (**final 3D reconstruction only**, **ALTERNATIVES**).
In case however the program does not finalize even if it uses only one processor per node, one has two options:

1. Reduce the size of input particles (and reference volume) by scaling them down. This can be done with the following command:

   ```
   sxprocess.py bdb:inputstack bdb:outputstack --changesize --ratio=0.8
   ```

   This will reduce window size to **nx** × 0.8, but it has an undesirable associated effect of increasing the pixel size by a factor of 1.0 / 0.8 = 1.25, thus increasing the maximum frequency (Nyquist frequency) by the same factor. It is also likely that due to larger pixel size the alignment accuracy will decrease.

2. Further "clean" your dataset with an additional, more conservative **ISAC2** round, by setting the **Pixel error threshold [Pixels]** parameter of **ISAC2** to a lower value. Refinements of "cleaner" stacks have in general lower average smear, run faster and require less memory.

NOTE: *The actual number of iterations varies from run to run and depending on the number of the iteration with the highest resolution, the parameter file might have a different file name; so, please check the content of Refine3D directory and use the proper number.*

On our cluster this **MERIDIEN** job required about 50 minutes to complete using 96 processors. If there is not enough time to perform the 3D refinement, please copy our precalculated results to the **project directory**.

```
cp –r SphireDemoResults/Refine3D ./
```

NOTE: *Should a **MERIDIEN** run into time limit or crash, you can restart it. For this purpose, click in the main window of the SPHIRE GUI the button **MERIDIEN** on the left and then the **3D Refinement Continuation** button within the **ALTERNATIVES** subsection in the middle and specify the **MERIDIEN** run directory (**Refine3D**). The program will identify, within the specified directory of unfinished **MERIDIEN** run, the last fully completed iteration automatically and continue from there.*

NOTE: *It is possible to change some of the refinement parameter values upon a restart, but we do not encourage experimenting with the settings. The reason is that ML refinement is an iterative process and by changing the parameters in the middle of it we modify the state of the program and this may have unintended consequences.*

Now we can use the ***Angular Distribution*** utility to produce a graphics file (*.bild*) with the angular distribution of your particles. Click the ***Angular Distribution*** button (**UTILITIES**) in the middle of the SPHIRE GUI. Fill out the following fields:

- **Projection parameters file:** Refine3D/final_params_032.txt
- **Number precision:** 5

  NOTE: *In case your refinement reached a small **delta** or your particle has no symmetry, the resulting .bild file will become very large and might crash your **UCSF Chimera** session, depending on your system's hardware. To avoid this, set the number of precision to 0 or -1.*

- **Particle radius [Pixels]:** 175

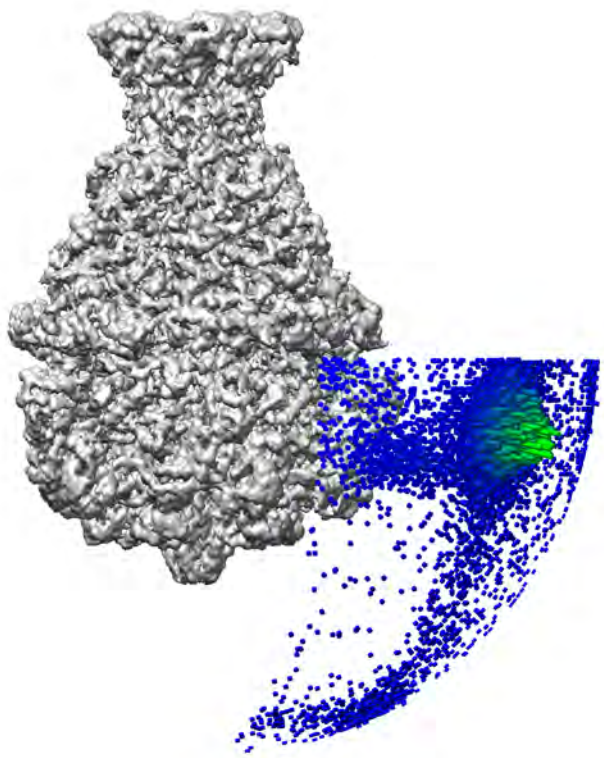
  NOTE: *Choose a value slightly larger than your actual particle radius, otherwise the 3D representation might cut your density.*

Click the ***Run command*** button and wait for the program to finish.

NOTE: *If you did not set the the **Project Settings** described in chapter PROJECT you need to set the values for **Point-group symmetry**, **Pixel size [A]** (**Advanced**) and **Box size [Pixels]** (**Advanced**) manually.*

NOTE: *Unlike previous versions, starting from version 1.0 a 3D distribution representation without mirror projections is produced, as this also reflects the information distribution. To learn more about it, visit the wiki page `http://sparx-em.org/sparxwiki/Euler_angles`. To recover the old behavior and produce a 3D distribution containing mirror projections, use C0 for **Point-group symmetry** (not recommended).*

Wait for the program to finish and then load one of the half volumes (i.e., **Refine3D/vol_0_unfil_032.hdf**) together with the resulting *.bild* file into **UCSF Chimera**.



The resulting *VRML model* in **UCSF Chimera** will illustrate the number of particle images assigned to given orientation directions within the asymmetric unit of angular space, as defined by the **Point-group symmetry**. Note this display only contains the distribution of the most probable orientations assigned to the data, as determined by the Maximum Likelihood methodology employed by **MERIDIEN**. Therefore, the picture is somewhat misleading as actual distribution of angular information may be significantly different. The full information about all orientation parameters is stored in each **Refine3D/mainITERNR/oldparamstructure**. However, at this point we do not have any utilities that would allow user to examine this information in a compact form.

Make sure that the visualized angular distribution is not be dominated by just a few directions. In such a case reconstructed structure will likely have directional artifacts (elongation in real space).

TIP: *If the data were already subjected to 3D refinement and 3D orientation parameters are available, you can use the already determined shifts and Euler angles to start a local refinement (**MERIDIEN/ALTERNATIVES/Local Refinement from Stack**). The local 3D refinement is designed to refine a structure with user-provided orientation parameters taken to initialize the iterative process. This program will do only local searches and it begins with 3D reconstruction of initial reference half-maps.*

In order to initiate the local refinement (or 3D sorting; see section below), input images should have 3D orientation parameters stored in headers. Alignment parameters from a **MERIDIEN** are imported to the header of the particle stack with following command (one long command):

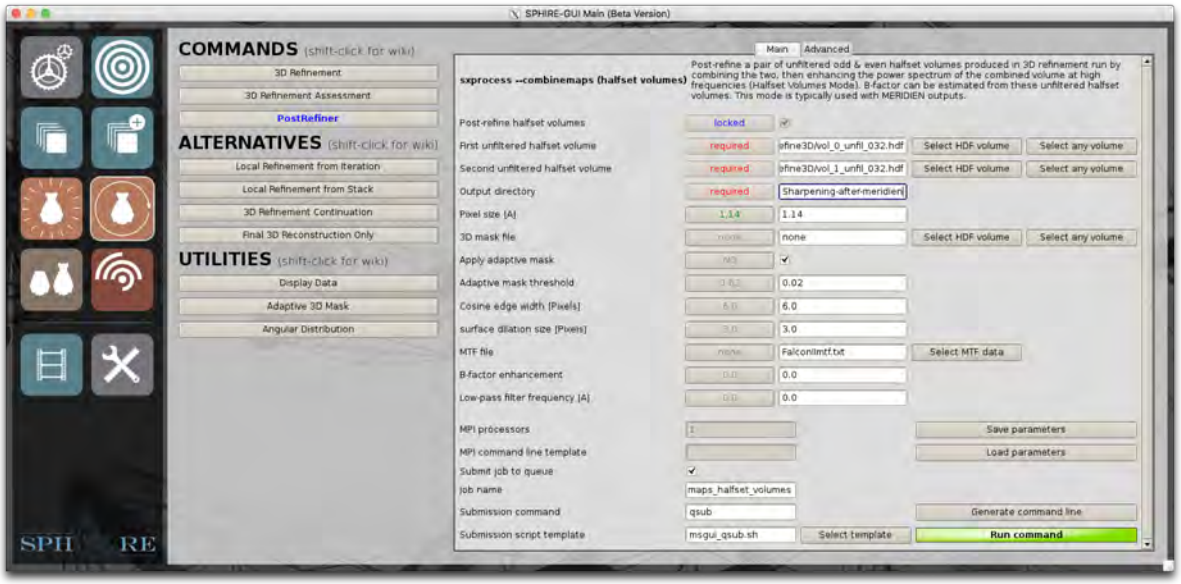
```
sxheader.py bdb:input_stack_name --params=xform.projection
--import=orientation_parameters.txt
```

TIP: *If you refined data using **RELION** you can easily convert the files and create a particle stack in BDB file format using the **UTILITIES/RELION to SPHIRE conversion** tool. The particle stack will contain alignment parameters stored in the header and you will be able to directly continue with SPHIRE and perform a local 3D refinement and/or 3D sorting using imported data.*

PostRefiner

This utility executes a sequence of operations in the following order: calculation of a soft 3D mask, estimation of the FSC using masked half-maps with FSC rescaling to account for the full-size of the dataset (Penczek 2010), merging of two half-maps, MTF compensation, filtration by the FSC, estimation of the B-factor from the merged map, high-pass filtration of the merged map using a B-factor-parametrized Gaussian function (B-factor is either estimated by the program or defined by the user), and low-pass filtration (either at the estimated resolution or at a cut-off chosen by user). Some of the operations listed are optional. The output of the utility is a power spectrum-adjusted map and also its masked version. Finally, the program reports resolution values of FSC@0.5 and FSC@0.143.

In the main window of the SPHIRE GUI click the button ***MERIDIEN*** on the left and then the ***PostRefiner*** button in the middle and fill out the following input fields:



- **First unfiltered halfset volume:** Refine3D/vol_0_unfil_032.hdf

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

- **Second unfiltered halfset volume:** Refine3D/vol_1_unfil_032.hdf

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

- **Output directory:** Sharpening-after-meridien
- **Pixel size [A]:** 1.14
- **3D mask file:** none

  NOTE: *Usually we automatically compute a 3D mask, using the 3D adaptive mask procedure (see below). In case you want to skip this and use a 3D mask obtained by a different method instead, use the file browser to load a 3D mask file and deactivate the apply adaptive mask option below.*

- **Apply adaptive mask**: YES

  NOTE: *Activation of this option will create a 3D mask directly from the combined half-maps guided by user-defined parameters listed below.*

  NOTE: *Setting of **3D mask file** to **none** and **Apply adaptive mask** to **NO**, will completely deactivate 3D masking. In this case the final output will be power spectrum adjusted but not masked and the FSC will be computed between unmasked half-volumes.*

- **Adaptive mask threshold:** 0.02

  NOTE: *Open one of the half maps in **UCSF Chimera** and establish visually a proper threshold value. The density map should not show any disconnected regions.*

- **Cosine edge width [Pixels]:** 6.0
- **surface dilation size [Pixels]:** 3.0
- **MTF file:** FalconIImtf.txt

  NOTE: *Providing an MTF file name will activate MTF compensation. Click the **Select MTF data** button and use the file browser to select the MTF file that comes along with out test data set for the Falcon II detector at 300 kV, stored in the **project directory**.*

- **B-Factor enhancement:** 0

  NOTE: *Set to 0, in order to estimate and apply B-factor.*

  *Alternative Options: **-1:** Do not apply B-factor.*

  ***Positive number:** 128 (a user specified B-factor of 128 $Å^2$ will be applied.*

NOTE: *By default, the program will estimate B-factor in 10 Å to full resolution range. You can adjust the B-factor estimation range in the advanced options.*

- **Low-pass filter frequency [A]:** 0

  NOTE: *Cut-off resolution to filter the final map. Set to 0 to filter the map according to FSC@0.143.*

  Alternative options

  Positive value: *5.8 (low pass filter to 5.8 Å.)*

Click the ***Run command*** button.

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

```
tail –f Sharpening-after-meridien/log.txt
```

The output looks like this:

```
2018-03-01_15:27:39 =>dilation :3.0

2018-03-01_15:27:39 =>------------>>>processing <<<----------------------

2018-03-01_15:27:39 =>3-D refinement combinemaps

2018-03-01_15:27:39 =>combinemaps enhances odd and even map

2018-03-01_15:27:39 =>-----------------------------------------------------

2018-03-01_15:27:39 =>------------>>>Refine3D/vol_0_unfil_032.hdf <<<----------------------

2018-03-01_15:27:39 =>the first input volume: Refine3D/vol_0_unfil_032.hdf

2018-03-01_15:27:39 =>the second input volume: Refine3D/vol_1_unfil_032.hdf

2018-03-01_15:27:41 =>create an adaptive mask, let's wait...

2018-03-01_15:27:41 =>options.mask_threshold, options.dilation,
options.consine_edge 0.020000 3.00 6.00

2018-03-01_15:30:56 =>adjust FSC to the full dataset by: 2.*FSC/(FSC+1.)

2018-03-01_15:30:56 =>fsc smoothly falls from 0.5 to 0.143

2018-03-01_15:31:00 =>MTF correction is applied

2018-03-01_15:31:00 =>MTF file is FalconIImtf.txt

2018-03-01_15:31:23 =>fsc_adj is not applied

2018-03-01_15:31:26 =>similarity between the fitted line and 1-D rotationally average power
spectrum within [41, 117] is 0.788

2018-03-01_15:31:26 =>the slope is -6.50[A^2]

2018-03-01_15:31:30 =>low-pass filter to FSC0.143 resolution (3.459310[A])

2018-03-01_15:31:33 =>the enhanced map without masking is saved as
Sharpening-after-meridien/vol_combined_nomask.hdf

2018-03-01_15:31:35 =>---------- >>>summary <<<------------

2018-03-01_15:31:35 =>resolution 0.5/0.143 are 3.90/ 3.46[A]

2018-03-01_15:31:35 =>B-factor is -26.00[A^2]
```

```
2018-03-01_15:31:35 =>FSC curves are saved in fsc_halves.txt, fsc_full.txt,
fsc_masked_halves.txt, fsc_masked_full.txt

2018-03-01_15:31:35 =>the final volume is Sharpening-after-meridien/vol_combined.hdf

2018-03-01_15:31:35 =>guinierlines in logscale are saved in
Sharpening-after-meridien/guinierlines.txt

2018-03-01_15:31:35 =>tanl low-pass filter is applied to cutoff high frequencies
from 1/ 3.46[1/A]

2018-03-01_15:31:35 =>---->>>analysis of enhancement <<<-----

2018-03-01_15:31:35 =>B_factor : 26.004211

2018-03-01_15:31:35 =>low-pass filter cutoff : 3.459310[A] (0.329545[absolute])

2018-03-01_15:31:35 =>low-pass filter falloff : 0.010000[absolute]

2018-03-01_15:31:35 =>max enhancement point : 114[pixels]

2018-03-01_15:31:35 =>max enhancement ratio : 5.031139

2018-03-01_15:31:35 =>1st zero pw spectrum point : 131[pixels]

2018-03-01_15:31:35 =>fallall width : 17[pixels]

2018-03-01_15:31:35 =>-----------------------------------------------------

2018-03-01_15:31:35 =>---------- >>>DONE<<<------------

2018-03-01_15:31:35 =>-----------------------------------------------------
```

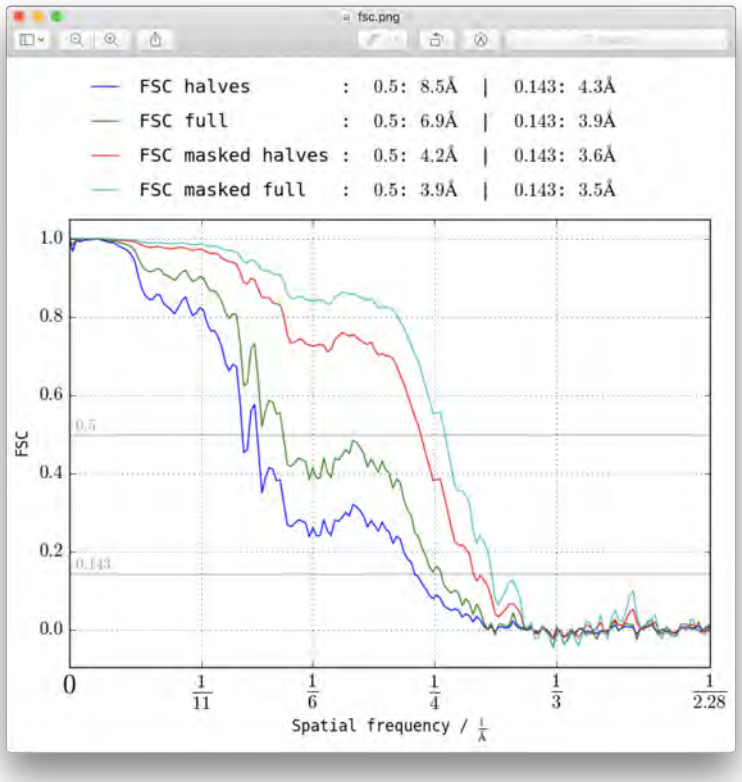
On our Linux cluster this program job finished after about 4 minutes.

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**.

In our case, according to the masked FSC@0.143, the final map has a resolution of 3.5 Å. A B-Factor of -26 Å^2 and a low-pass filter at 3.5 Å were thus applied to the map. In the output directory you can now find the following files:

- **vol_combined.hdf:** The merged, sharpened and masked final structure, filtered to the achieved resolution of 3.5 Å.
- **vol_combined_nomask.hdf:** The final structure, with no mask applied.
- **vol_adaptive_mask.hdf:** The soft 3D mask used for the FSC calculations.
- **fsc.png:** A plot with all FSC curves computed.

The program will also output text files for all FSC curves and the Guinier plot. Display the final structure using **UCSF Chimera** and check the FSC plots carefully (see note below). Compare the EM map with the available X-ray structure.

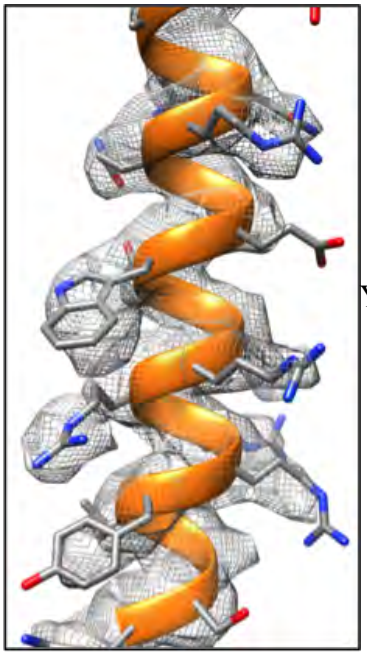


NOTE: *As is well known, application of a very tight mask may result in an overestimation of resolution. While in general the effect is impossible to avoid (short of not using any mask at all), one has to take precautions to avoid excessive impact of the mask. Examine the FSC curve carefully and in case FSC curves raise in high frequencies or it never drops below zero, repeat FSC estimation using a more generous mask. Another telltale sign is emergence of strange B-factor values estimated by the program and/or high-resolution artifacts in the map. Ideally, right after it drops to 0.143, the FSC should oscillate about the zero line. For an in-depth discussion and more examples see (Penczek 2010).*

TIP: *You should always visually inspect the resulting map and confirm that the features of the density agree with the nominal resolution (i.e., a high-resolution map should show clearly discernible side chains). You can also simply download atomic coordinates of an X-ray crystallographic structure, convert them to a pseudo-electron density map using box and pixel size of the map in question (**PDB File Conversion** in **UTILITIES**) and apply low-pass filter using the same cut-off resolution as the one determined in the post-refinement step. The appearance of surfaces, as visualized by **UCSF Chimera**, should be similar between the EM map and the converted X-ray model. In particular, spurious surface features, strange sharpness of the surface, disconnected pieces, lack of secondary features that at the same time are discernible in X-ray map set a red flag. Conversely, exceedingly smooth appearance of EM map means that either its resolution and/or the B-factor were underestimated.*

IMPORTANT: Scaling of the FSC (**FSC masked full**) is still an experimental feature under development. Check the output structure carefully – for some projects (large datasets and/or high symmetries) the scaled FSC might report overestimated resolutions. In such a case, consider also the result of the half-map FSC halves@0.143 (**FSC masked halves**) or the FSC full@0.25 and filter the output structure to the most appropriate resolution value taking into account the visual appearance of the structure.

Results of the refinement highly depend on the quality of the initial model. Therefore, we usually recommend starting a new round of refinement using the sharpened volume as an initial reference and the 3D mask obtained by the **PostRefiner** utility. This may improve resolution of the newly refined structure.



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

SORT3D

3D Variability

After the 3D structure refinement is completed, we assess possible structural variability of the complex. 3D variability might be caused by conformational changes, substoichiometric ligand binding and/or compositional heterogeneity. In general, regions with higher variability are less reliable and the data may require further analysis. In order to detect such regions, we calculate a 3D variability map using the EM image data and their orientation parameters.

> **TIP:** *Although regions with higher variability typically show fewer details, this step is not done to estimate the local resolution, but it is rather used to guide the subsequent step of 3D sorting. 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 calculations. 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 orientation parameters of the final refinement iteration (with the highest resolution) to the header of the clean particle stack (the particle stack created after **ISAC2**, which we used as input for the 3D refinement).

In the main window of the SPHIRE GUI click the button ***SORT3D*** on the left and then the ***Import Projection Parameters*** button in the middle:

Fill out the following input fields:

- **Import parameters:** Refine3D/final_params_026.txt

  NOTE: *Click the **Select parameters text** button and use the file browser to select the text file with the orientation parameters of the refinement iteration with the highest resolution.*

- **Input image stack:** bdb:Substack#isac_substack

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

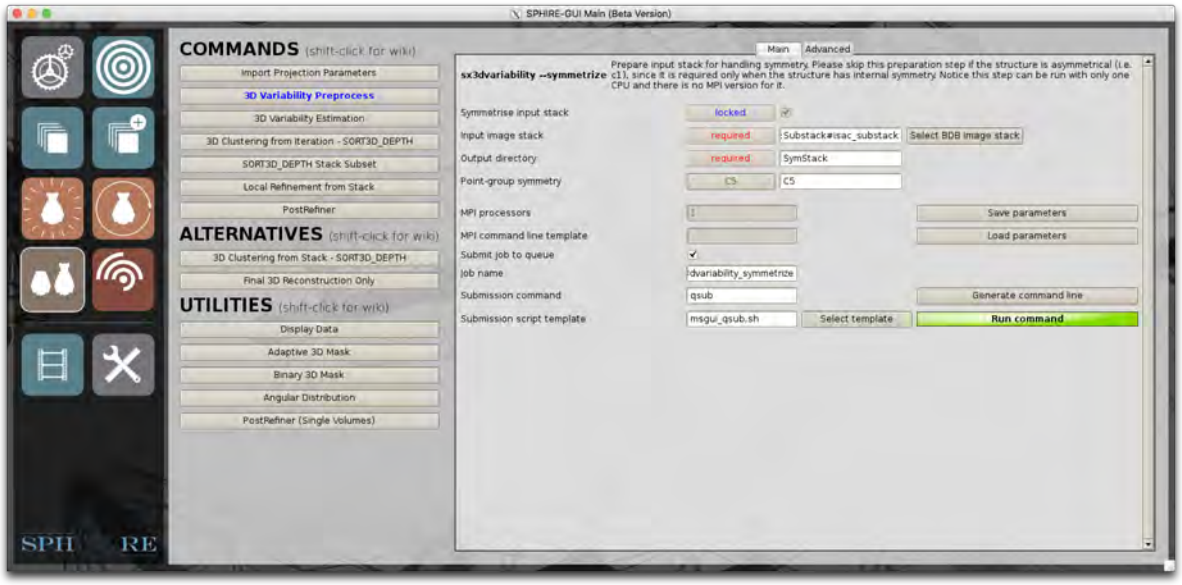
NOTE: *During the 3D refinement, **MERIDIEN** outputs and uses multiple projection directions (probability-weighted) for each particle. However, for the purpose of variability analysis, we use only the highest-probability orientations for each particle.*

Click the ***Run command*** button.

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: *For an asymmetric particle, skip this step.*

In the main window of the SPHIRE GUI click the button ***SORT3D*** on the left and then the ***3D Variability Preprocess*** button in the middle:



Fill out the following input fields:

- **Input image stack:** bdb:Substack#isac_substack

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

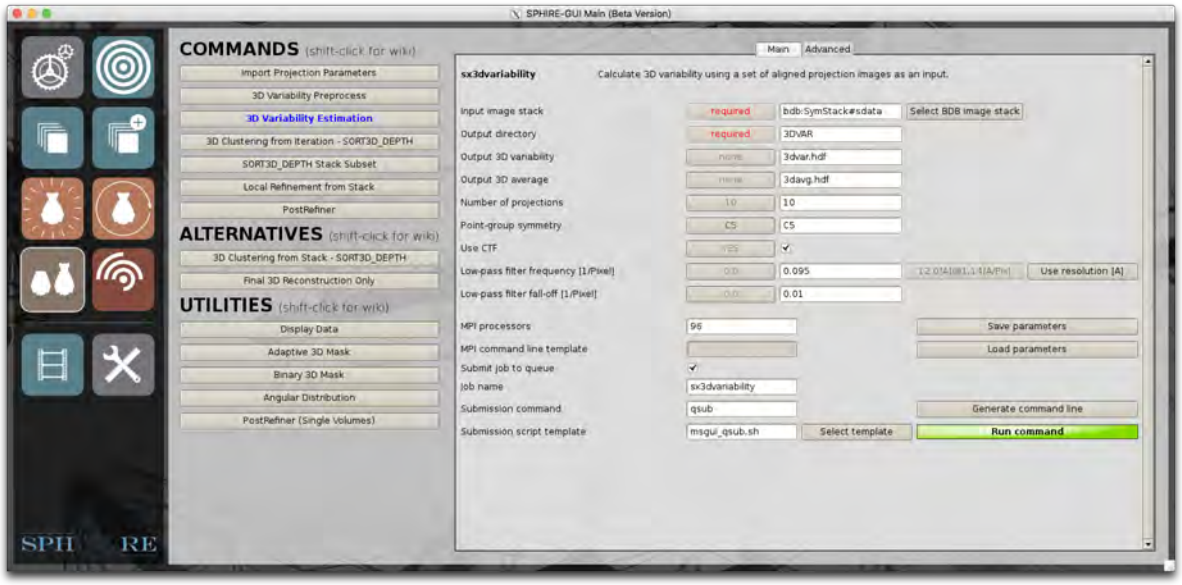
- **Output directory:** SymStack
- **Point-group symmetry:** C5

A symmetrized dataset named **bdb:SymStack#sdata** will then be stored directly in the output directory. (It is a virtual stack, so no excessive disk space is consumed). Use **EMAN2**'s **e2iminfo.py** to check the number of particles in this dataset.

```
e2iminfo.py bdb:SymStack#sdata
```

After "symmetrization", for C5 symmetry used here, this stack should contain five times more projections than your input dataset ($5 \times 9259 = 46295$ particles).

In the main window of the SPHIRE GUI click the button ***3D Variability*** Estimation in the middle:



Fill out the following input fields:

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

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

  NOTE: *After the successful 3D refinement, projection parameters are now available 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 map (see below). If you want to output the 2D intermediate results, activate the respective option in the advanced parameters.*

- **Output directory:** 3DVAR
- **Output 3D variability:** 3dvar.hdf

  NOTE: *3D variability map.*

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

  NOTE: *3D map computed from neighbor-based 2D averages.*

- **Number of projections:** 10

  TIP: *The larger the number, the less noisy the variability map but the lower the resolution. In general, 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
- **Use CTF:** YES

  TIP: *Deactivate this flag for negative-stain data. For stain data, variability analysis is challenging and should be performed only if one expects compositional heterogeneity of large-molecular-weight components or very large and well-defined conformational changes. To enhance the results, consider restricting analysis of the variability to data extracted from micrograph areas of similar stain thickness.*

- **Low-pass filter frequency [1/Pixel]:** 0.095
- **Low-pass filter fall-off [1/Pixel]:** 0.01

  NOTE: *In order to suppress noise, before the variability calculation, we will apply a low-pass filter with a cut-off resolution of 12 Å to 2D averages. This corresponds to an absolute frequency of 0.095 1/pixel. Use the resolution pop-up converter (**Use resolution [A]** button) to calculate resolution in angstrom.*

  TIP: *If you wish to detect variability of high-resolution features (e.g. movement of secondary structure elements or binding of small ligands) you will have to relax the low-pass filter.*

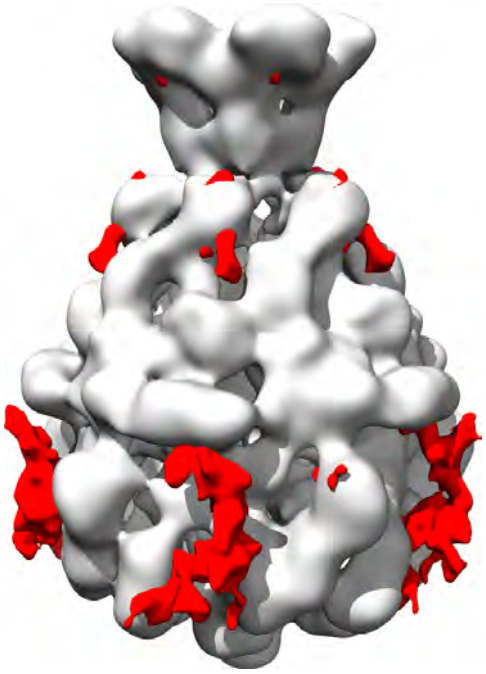
Specify the number of processors (for this job we used 96) and submit the job to the queuing system of the cluster using an appropriate submission script by clicking the ***Run command*** button. This process should finish after few minutes (or hours for larger datasets).

The 3D average and variability volumes (**3davg.hdf**, **3dvar.hdf**) are located in the output directory **3DVAR**. Display both maps in **UCSF Chimera**.

Otherwise, to save time, you can copy the precalculated variance and average maps to the **project directory** and display them:

```
cp –Rp SphireDemoResults/3DVAR ./
```

The first impression is that the variability map is noisy. Nevertheless, in order to identify higher variability regions, we will try to suppress the noise to bring up possible features in the map. For this purpose, set the threshold of the variability map to a high value (0.0006) and remove smaller blobs using the **Hide Dust** tool of **UCSF Chimera** (set parameter “size” to 32, see the outcome in the image below). The average map and the simplified variability map are shown in gray and red, respectively.



Although the simplified variability map is still noisy, it is apparent that the interfaces between the protomers and especially the lower part of the particle are 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 not as well resolved in the X-ray map as other regions and which are expected to be more flexible. Interestingly, the area with the highest variability is visible at the expected location of the His-tagged N'-terminal α-helix, which could not be resolved at all in the crystal structure. In summary, as expected, the bulk of our tutorial particle can be considered rather rigid, so the 3D variability analysis yields rather limited information.

Nevertheless, in order to demonstrate how to identify homogeneous subpopulations of particles, we will continue with a 3D heterogeneity analysis in the next section.

3D Clustering

We will now perform a 3D cluster analysis of the 3D refined dataset using the **SORT3D** program. In the protocol adopted in SPHIRE we separate 3D refinement from 3D sorting of the data. Thus, in a first step we refine the entire dataset using **MERIDIEN** (see previous section) assuming that it is relatively homogeneous, i.e., it does not contain different categories of macromolecular shapes (for example a mixture of ribosomes and proteasomes). In particular,

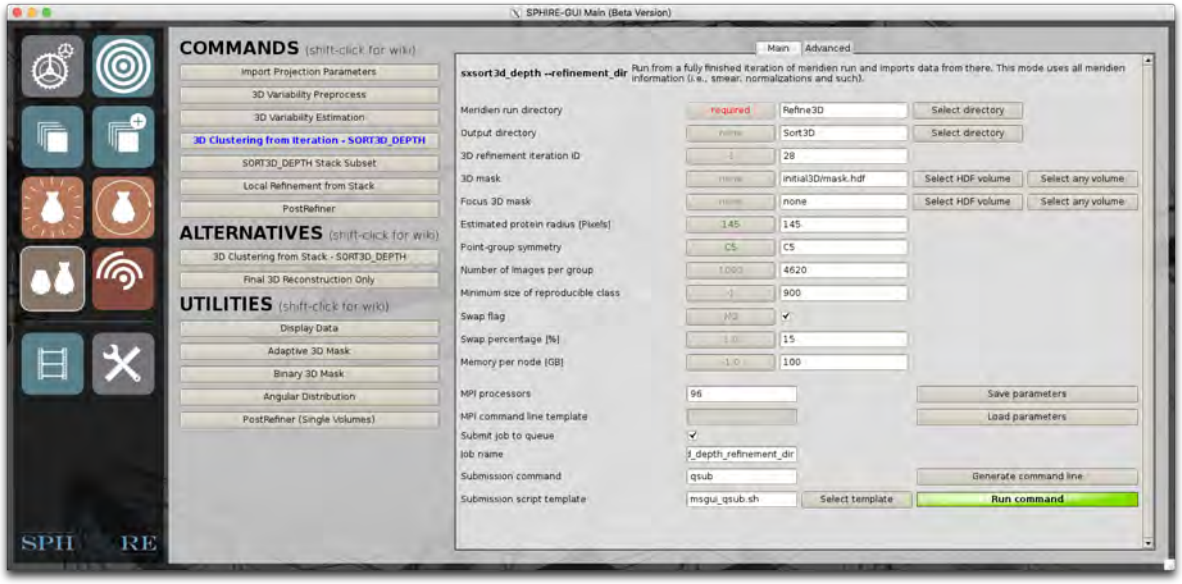
any heterogeneity or flexibility of the analyzed complex should be "secondary" with respect to the overall shape and thus it should make sense to initially treat all projections as belonging to one, quasi-homogeneous dataset. The overall structure derived from **MERIDIEN** refinement should, to a degree, "make sense" and its reprojections should match the reference-free 2D class averages of **ISAC2**.

Given this assumption, we can now proceed with the next step, the clustering, treating orientation parameters as constant (i.e., there is no alignment during sorting). In **SORT3D** we employ a variant of *K*-means algorithm in which we maintain approximately equal sizes of sorted groups. To provide validation of the outcome, the **SORT3D** program employs a strategy of multiple independent random restarts. Thus, the program will actually perform clustering of the data multiple times using different random initializations (different initial structures), compare resulting group assignments and accept results only if they are above the level expected from chance sorting. It follows that the outcome is validated in a sense that the user can be reasonably certain that the results can be reproduced and are not random artifacts. Finally, the program will output results of statistical ANOVA tests designed to verify that the outcome of sorting is not influenced by parameters that are unrelated to the structure itself and, for example, avoid results that were sorted "by defocus", which regrettably is a relatively common artifact.

TIP: *This step is computationally expensive, and the running time increases significantly with number of particles and groups.*

TIP: *In other software packages, 3D sorting is sometimes performed to remove "junk" particles. In SPHIRE, most of these particles have already been eliminated by **ISAC2** in 2D.*

In the main window of the SPHIRE GUI click the button ***SORT3D*** then button ***3D Clustering from Iteration - SORT3D_DEPTH*** in the middle and fill out the following fields:



- **Meridien run directory:** Refine3D

NOTE: *Click the **Select directory** button and use the file browser to select the 3D refinement directory.*

TIP: *The program takes advantage of all of the ML information outputted by **MERIDIEN**, i.e., in the 3D reconstruction step will use probability distributions associated with particle images.*

- **Output directory:** Sort3D
- **3D refinement iteration ID:** 28

  NOTE: *The **MERIDIEN** iteration number whose alignment parameters will be used by the program.*

  TIP: *The choice of iteration number depends on the goals set by the user. It does not have to be the final iteration and in fact to select it is usually counterproductive as last few iterations use large window size and this will result in long time of **SORT3D** calculations. For the same reason one should avoid iterations with large smear. Also, early iterations should not be used, including early **RESTRICTED** ones as the structure is not yet formed sufficiently to permit sensible sorting.*

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

  NOTE: *This is the global mask used for the sorting procedure. It should be always soft-edged. For the tutorial dataset, the mask previously used in **MERIDIEN** 3D refinement is a good choice. Click the **Select HDF volume** button and use the file browser to select the soft 3D mask used during refinement. However, we prefer to use a more generous mask here (not the 3D mask obtained during sharpening), because the heterogeneity revealed by the 3D variability analysis was not resolved in the refined map. In general, tight masks should be avoided as one may inadvertently mask out a yet undetected region of interest.*

  TIP: *When you process your own data display the 3D variability map, the refined map and the 3D mask together jointly in **UCSF Chimera**, to confirm that the soft 3D mask encloses the protein map completely (particularly the dynamic components that are not resolved in the refined map, but clearly present in the variability map).*

- **Focus 3D mask:** none

  NOTE: *A binary focus mask. It is used to perform focused clustering. It allows the user to focus sorting on a particular region of a 3D structure, for example a flexible region, as discussed earlier. Here, since the 3D variability map did not reveal any local heterogeneity (e.g. substoichiometric ligand binding, a flexible domain, etc.), we will not conduct focused classification and thus we set this input to "none".*

  TIP: *In case the variability analysis indicates a clear local flexibility in the map, use the **Binary 3D Mask** utility to binarize the 3D variability map and create a focus mask for **SORT3D**.*

- **Number of images per group:** 4620

  NOTE: *The value entered will be used to determine the initial number of groups, as given by the following relation:*

  $$\textit{Total Number of Particles} = \frac{9259}{\textit{Particles per group}} = 4620 \approx 2\,.$$

  However, the ultimate number of groups determined by the program depends on the heterogeneity of the dataset and may be either larger or smaller than the initial number. In particular, if there is no reproducible partition of the dataset given user-settings, the program will terminate without returning any groups.

  The program will try to maintain groups of the size given by this parameter. However, the ultimate group sizes will not necessarily be equal to the value used; they can differ significantly. In general, larger value will result in larger group sizes.

  IMPORTANT: The desired number of images per group has to be larger than the minimum group size set (the next parameter).

  TIP: *Adjust this parameter group size based on the size, resolution, and quality of your dataset, the available computational resources, and the goal of **SORT3D**. The computational time will increase linearly with the number of groups.*

- **Minimum size of reproducible class:** 900

  NOTE: *The clusters determined by the program will have at least the number of images determined by the value of this parameter. Smaller candidate clusters will be eliminated. A minimum group size cannot be too small (for example 10), as it is impossible to compute a valid 3D structure from such a small number of images. We recommend setting it to a possibly large value, but not to exceed the desired group size.*

- **Swap flag:** YES
- **Swap percentage [%]:** 15
- **Memory per node [GB]:** 100

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

You can monitor the progress of the program by following the information that appears in the logfile **log.txt**.

```
tail -f Sort3D/log.txt
```

On our cluster this job finished in about 75 minutes. Otherwise, to save time, copy the precalculated results to the **project directory** and continue with those. In this case, type:

```
cp –Rp SphireDemoResults/Sort3D ./
```

The **SORT3D** results are reported at the end of the logfile (**log.txt**) saved in the output directory. For each determined group the printout states group ID, number of images assigned to the group, group reproducibility, group random reproducibility, the standard deviation of the group random reproducibility, the name and the location of the group selection file, and finally the location and name of the map file of this group. The logfile also contains information about the total number of images, the number of images in all groups (called accounted for), and the number of the images not assigned to any groups (unaccounted for, their numbers are saved in a text file **Unaccounted.txt** in the output directory). What follows are results of statistical ANOVA computed for group defocus values, image norms per group, and the smearing (spread of probabilities per image) per group. These results make it possible to determine whether the determined grouping of the data reflects factors other than structural heterogeneity. Of particular interest is defocus. Should its average values differ significantly between groups, it is quite likely that the apparent structural differences are due to differences in group defocus values, not due to actual differences in structures.

At the terminal type:

```
cat Sort3D/log.txt

2018-03-05_18:13:39 =>Final results saved in Sort3D3

2018-03-05_18:13:39 =>-----------------------------

2018-03-05_18:13:39 =>Group ID size determined in generation reproducibility random
reproducibility std selection file map file

2018-03-05_18:13:42 =>0 3370 0 55.4 37.0 3.2 Cluster_000.txt vol_cluster000.hdf

2018-03-05_18:13:46 =>1 2651 0 49.7 31.2 3.3 Cluster_001.txt vol_cluster001.hdf

2018-03-05_18:13:50 =>2 1380 1 75.2 34.9 3.4 Cluster_002.txt vol_cluster002.hdf

2018-03-05_18:13:53 =>3 1226 1 75.4 32.3 3.5 Cluster_003.txt vol_cluster003.hdf

2018-03-05_18:13:53 =>Images 9259 accounted for images: 8627
unaccounted for images: 632

2018-03-05_18:13:53 =>Unaccounted images saved in Unaccounted.txt
```

From the above table we learn that out of 9259 images, the program assigned 8627 images (accounted for) to four groups (numbered 0 to 3, respectively), containing 3370, 2651, 1380, and 1226 images, respectively. 632 images were not assigned to any groups (unaccounted for images). Due to intrinsic randomness of the sorting algorithm, numbers of images per group may slightly differ from one run to another.

The final results table contains results of reproducibility tests. As the program performs independent clustering it is possible to compute how reproducible are groups obtained in these independent runs, i.e., compute the percentage of image that were assigned to the same group in different runs. Here they are 55.4 %, 49.7 %, 75.2 %, and 75.4 %, respectively. In order to assess whether these reproducibility levels are above what one would expect if the assignments were entirely random, we perform **Monte Carlo** simulation using group sizes as input. The random reproducibility levels are 37.0 %, 31.2 %, 34.9 %, and 32.3 %, while their standard

deviations are 3.2, 3.3, 3.4, and 3.5, respectively. We set the group rejection threshold at random reproducibility plus its two standard deviations (which approximately corresponds to 5 % significance level) and remove those groups whose reproducibility is below this level. Please note that at this level (and even at three σ level) all obtained groups are significantly more reproducible than what would obtain if groups were due to chance assignments, i.e., if in reality there were no groups in the data.

The output logfile also contains results of statistical analysis of possible influence of external parameters that are a source of data heterogeneity on the output of the sorting. We will discuss analysis of variance of average defocus values per group, i.e., we would want to know whether obtained groups differ by defocus. If that was the case, the sorting result would be questionable. The relevant part of the logfile is:

```
2018-03-05_18:13:58 =>=====================
2018-03-05_18:13:58 =>ANOVA analysis
2018-03-05_18:13:58 =>-------------------------
2018-03-05_18:13:58 =>ANOVA of defocus
2018-03-05_18:13:58 =>ANOVA F-value p_value
2018-03-05_18:13:58 =>ANOVA 7.09 0.00
2018-03-05_18:13:58 =>
2018-03-05_18:13:58 =>ANOVA: defocus mean of all clusters: 1.613500
2018-03-05_18:13:58 =>ANOVA: Group averages
2018-03-05_18:13:58 =>ANOVA GID N mean std
2018-03-05_18:13:58 =>ANOVA 0 3370 1.5937 0.3490
2018-03-05_18:13:58 =>ANOVA 1 2651 1.6344 0.3343
2018-03-05_18:13:58 =>ANOVA 2 1380 1.6172 0.3473
2018-03-05_18:13:58 =>ANOVA 3 1226 1.6184 0.3481
2018-03-05_18:13:58 =>
2018-03-05_18:13:58 =>ANOVA Pair-wise tests
2018-03-05_18:13:58 =>ANOVA A B avgA avgB P_value f-value
2018-03-05_18:13:58 =>ANOVA 0 1 1.5937 1.6344 20.863 0.0000
2018-03-05_18:13:58 =>ANOVA 0 2 1.5937 1.6172 4.434 0.0353
2018-03-05_18:13:58 =>ANOVA 0 3 1.5937 1.6184 4.487 0.0342
2018-03-05_18:13:58 =>ANOVA 1 2 1.6344 1.6172 2.332 0.1268
2018-03-05_18:13:58 =>ANOVA 1 3 1.6344 1.6184 1.868 0.1718
2018-03-05_18:13:58 =>ANOVA 2 3 1.6172 1.6184 0.008 0.9307
```

According of the results of the overall ANOVA of defocus the differences are not significant, as further confirmed by pair-wise tests, therefore we can reject the hypothesis that defocus influenced results of 3D sorting. (Significance should be
textless= 0.05 in order to reject null hypothesis of equality of defocus means).

In the output directory, you can find following files:

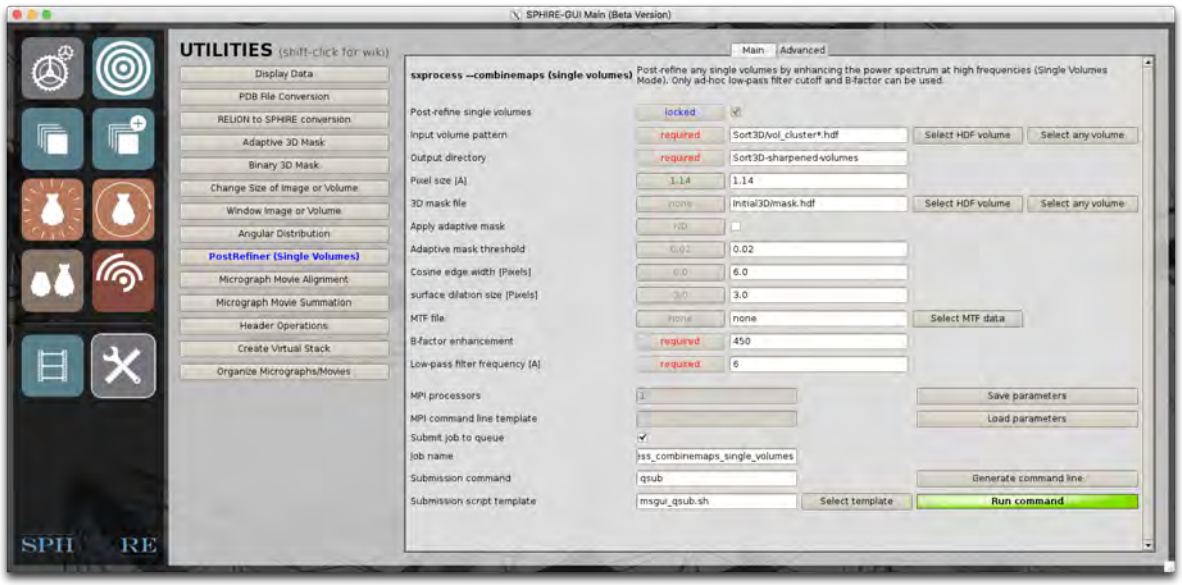
– **Cluster_ITERNR.txt**

> **NOTE:** *The numbered ClusterITERNR.txt files contain the IDs of the particles assigned to the respective cluster.*

– **vol_clusterITERNR.hdf**

These are the volumes calculated from each cluster of images. Note however these are internal **SORT3D** maps (similar to half-maps stored for every iteration during **MERIDIEN** run in mainITERNR directories), i.e., during 3D reconstruction they were truncated at the maximum resolution of the refined input data. As such, they will be featureless if displayed in **UCSF Chimera**. If you wish to bring up some details, you can use the **PostRefiner** utility and apply a large B-factor to these volumes. However, keep in mind that these volumes are truncated and the final resolution can be obtained only after running a local refinement on the respective substack of particles (see below).

To apply a B-factor on these volume, click the button ***UTILITIES in the main window of the SPHIRE GUI and then button PostRefiner (Single Volumes)*** in the middle and fill out the following fields:



- **Input volume pattern:** Sort3D/vol_cluster*.hdf

NOTE: *Click the **Select HDF Volume** button and use the file browser to select a cluster volume. Then replace the variable part of the file name with the wildcard character "*". The program will batch process all cluster volumes using the settings below.*

- **Output directory:** Sort3D-sharpened-volumes
- **Pixel size [A]:** 1.14
- **3D mask file:** Initial3D/mask.hdf

  NOTE: *This is the global mask used for the sorting procedure.*

- **Apply adaptive mask:** NO
- **MTF file:** none
- **B-factor enhancement:** 450

  TIP: *The choice of B-factor is somewhat arbitrary, but it is not all that important as enhanced maps are only used for visual assessment of the sorting outcomes so user can decide which, if any, groups should be subjected to subsequent local refinement. The value of B-factor should be adjusted using visual inspection of the outcome: the enhanced map should retain "structural integrity" and histogram of densities should have "ice peak" (routinely set to zero by the reconstruction program) at about 0.25 of the histogram range.*

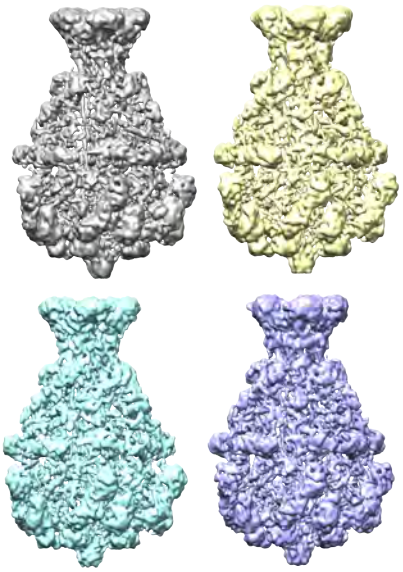
- **Low-pass filter frequency [A]:** 6

  NOTE: *This should be resolution used by the **SORT3D** program.*

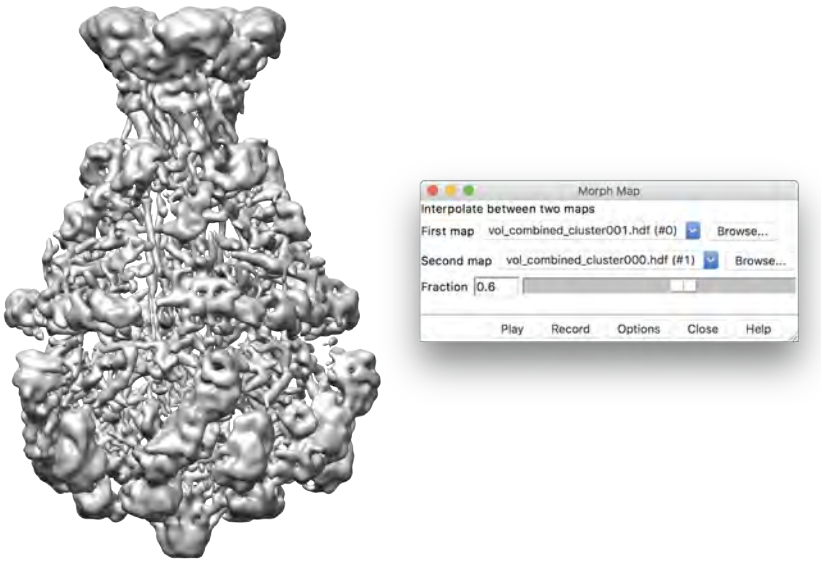
Start the **PostRefine*r (Single Volumes)*** job by clicking the ***Run command*** button.

NOTE: *This step (and **SORT3D** in general) does not output any resolution or FSC curves.*

Display the resulting maps with **UCSF Chimera**.



At first glance, the maps appear similar. Now use **UCSF Chimera**'s **Morph Map** utility to compare them.



In the morphed structures, we observe a rather quasi-continuous but clear flexibility of several domains of the protein and overall movement of secondary structure elements. The flexibility is localized in the upper-region of the complex: The TcB-binding site and adjacent domains. Further studies revealed that binding of component TcB to TcA and formation of the holotoxin complex stabilizes this structural region. Such an in-depth analysis of such heterogeneity requires however a much larger dataset and would therefore be beyond the scope of this tutorial.

Keep in mind that the orientation parameters used during sorting relate particle image to the average structure, thus, due to superposition of different conformer, the fine detail differences might not be fully resolved at this point. Therefore, the full potential and resolution limit of each determined group can be only revealed after within-group 3D local refinement (see next section Local Subset Refinement and Final Reconstruction).

However, we usually perform a local refinement only for groups whose visual appearance promises delivery of interesting (and biologically interpretable) results. This is the basic outline and, needless to say, in practice the procedure can be more complex, for example sorting maybe applied again within selected groups after their local refinement. In addition, we expect users to compute and analyze 3D variability maps at all stages as well as examine carefully results of statistical tests done in **SORT3D** program in order to avoid acceptance of chance groups.

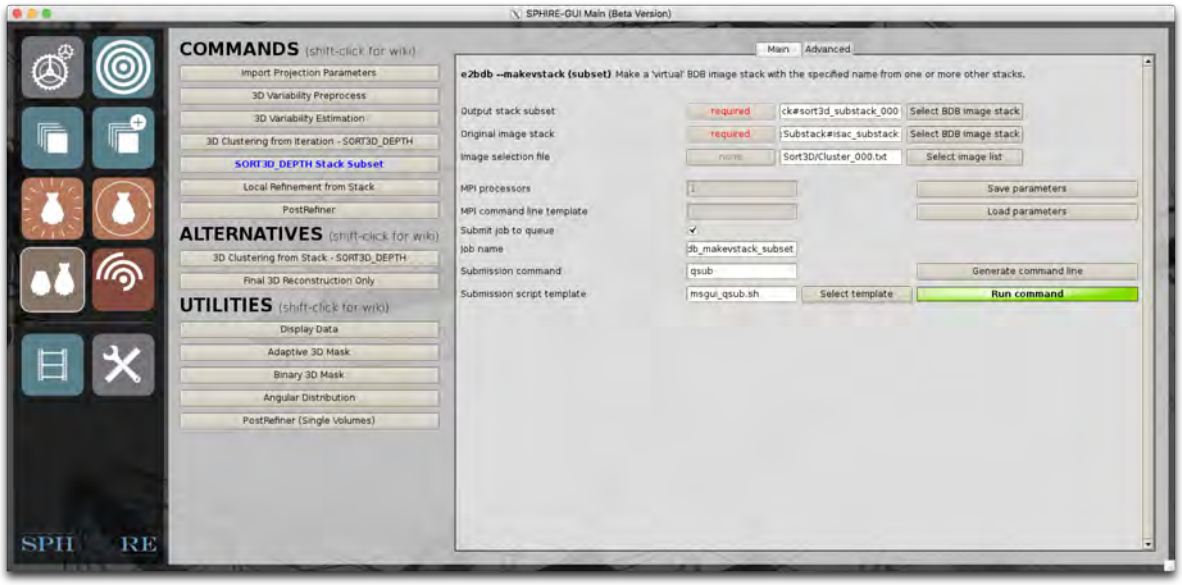
To complete the protocol, it is necessary to use **MERIDIEN** in the local mode to perform a local refinement of all groups of interest independently with possible additional gain of improved resolution, as now the data subsets have improved homogeneity.

> **TIP:** *In case **SORT3D** reveals large conformational changes, every group should be refined independently and in such a case one might want to consider starting the **MERIDIEN** runs of each group with exhaustive searches (mode 1) (instead using the local search mode).*

Now, we will create virtual "clean" particle stacks for each sorted cluster that will contain only the particles with IDs included in **Cluster_ITERNR.txt** files outputted by **SORT3D**.

> **TIP:** *It is advisable to verify at this point that the stack header is properly populated with the orientation parameters of the **MERIDIEN** run. At the terminal, type:*
>
> ```
> sxheader.py bdb:Substack#isac_substack --params=xform.projection
> --export=try_orientation_parameters.txt
> ```
>
> *Should the command crash or produce an empty text file, it means there are no parameters in the input stack and one should carefully check all steps of the analysis to find out what went wrong. The parameters should have been imported after **MERIDIEN** run was completed and **SORT3D** was initialized using the **Import Projection Parameters** tool.*

In the main window of the SPHIRE GUI click the button ***SORT3D*** and then the ***SORT3D_DEPTH Stack Subset*** in the middle:



Next, fill out the following input fields:

- **Output stack subset:** bdb:Substack#sort3d_substack_000
- **Original image stack:** bdb:Substack#isac_substack

  NOTE: *Click the **Select BDB image stack** button and use the file browser to select in the **Substack** folder the virtual stack used as input for the previous **SORT3D** run.*

- **Image selection file:** Sort3D/Cluster_000.txt

  NOTE: *Click the **Select image list** button and use the file browser to select the file with the IDs of all accounted particles by **SORT3D**.*

  Click the ***Run command*** button.

  This will create a virtual stack containing all particles of **Cluster_000.txt**.

  TIP: *The above command has to be **repeated for all groups we intend to do local refinement on independently**. One has to input selected **Cluster_ITERNR.txt** files and set number **bdb:Substack#sort3d_substack_ITERNR** accordingly. For the results described here, one should create four different substacks.*

  – *bdb:Substack#sort3d_substack_000*

  – *bdb:Substack#sort3d_substack_001*

  – *bdb:Substack#sort3d_substack_002*

  – *bdb:Substack#sort3d_substack_003*

  *After the job has finished, you can verify the number of particles in the resulting stack using **EMAN2**'s **e2iminfo.py** command. At the terminal, type:*

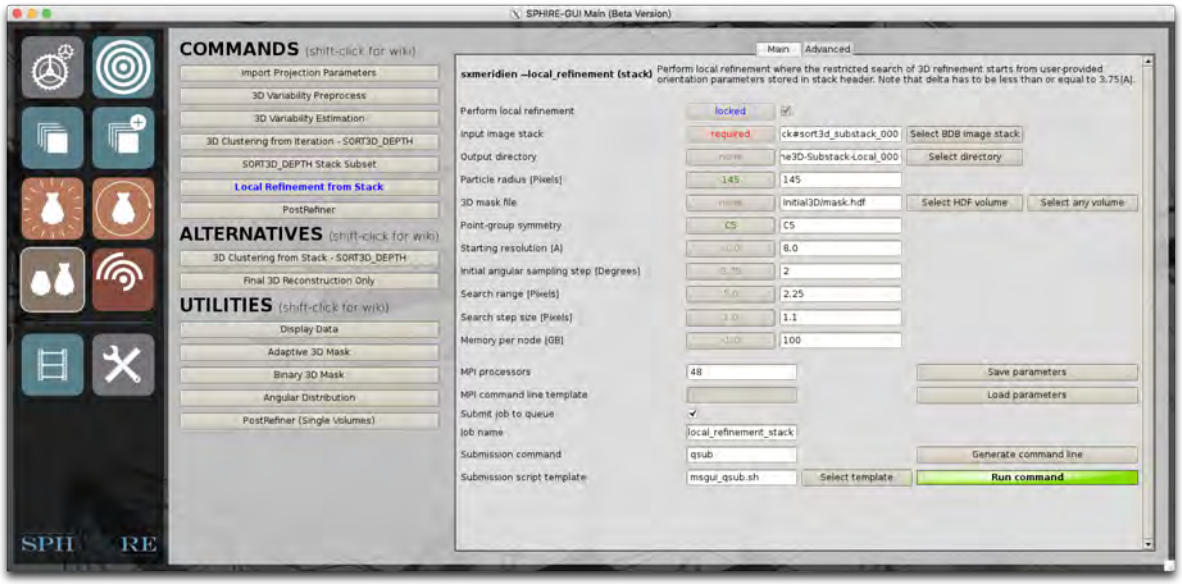
```
e2iminfo.py bdb:Substack#sort3d_substack_000
e2iminfo.py bdb:Substack#sort3d_substack_001
e2iminfo.py bdb:Substack#sort3d_substack_002
e2iminfo.py bdb:Substack#sort3d_substack_003
```

The substacks in the precalculated results contain 3370, 2651, 1380 and 1226 particles, respectively.

Local Subset Refinement and Final Reconstruction

The group substacks created in the previous section contain 3D orientation parameters (recall they were imported to the full stack at the beginning of sorting protocol, and thus will be inherited by any virtual substack derived from it), so they have the correct format to serve as inputs for a local 3D refinement by **MERIDIEN**.

In the main window of the SPHIRE GUI click the button ***SORT3D*** on the left and then the ***Local Refinement from Stack*** in the middle and fill out the following input fields:



- **Input image stack:** bdb:Substack#sort3d_substack_000

  NOTE: *Click the **Select BDB image stack** button and use the file browser to select the substack created after **SORT3D**.*

- **Output directory:** Refine3D-Substack-Local_000
- **Particle radius [Pixels]:** 145
- **3D mask file:** Initial3D/mask.hdf

NOTE: *Click the **Select HDF volume** button and use the file browser to select the soft-edge 3D mask used during the refinement step. However, if the 3D sorting reveals large conformational changes, you should create a different mask from each **SORT3D** volume to be used as reference for the local refinement. For this purpose, you can use the **Adaptive 3D Mask** utility.*

- **Point-group symmetry:** C5
- **Starting resolution [A]:** 8.0

  NOTE: *Enter the resolution that the program is likely to estimate using initial orientation parameters provided. The starting resolution here has limited impact on the refinement process and it is only used to accelerate the initial 3D reconstruction, whose size will be limited to correspond to the resolution given. If the parameter is omitted, the program will compute full size half-maps to estimate initial resolution, which most likely will be time consuming and is in general unnecessary.*

- **Initial angular sampling step [Degrees]:** 2.0

  NOTE: *The initial angular step has to be smaller than 3.75°.*

- **Search range [Pixels]:** 2.25
- **Search step size [Pixels]:** 1.1

  NOTE: *The values of the three above sampling parameters depend on many factors, such as resolution of the initial structure, what kind of refinement was already performed, where the orientation parameters originated from and, importantly, what is the goal of the local refinement to be performed. Therefore, we cannot provide default values and it is more than likely that it will be necessary to experiment with initial settings and adjust them based on performance of the program and the quality of the result. As a rule of thumb, the initial angular step and search range should be 1.5 × of the values of the **MERIDIEN** iteration used as an input to **SORT3D**.*

- **Memory per node [GB]:** 100

Specify the number of processors (we used 48) and submit the job to the queuing system of the cluster using an appropriate submission script by clicking the ***Run command*** button.

Monitor the progress of the job as discussed in the standard 3D refinement section. On our cluster, this job with 48 processors finished in about 3 hours.

However, if there is no time to wait for the results, copy the precalculated results to the **project directory**.

```
cp -Rp SphireDemoResults/Refine3D-Substack-Local_* ./
```

NOTE: *The above local refinement has to be done for all selected groups. In this case we will refine all 4 groups. Please change **Input file stack** name and **do not forget** to change **Output directory** name accordingly. If you have sufficient computational resources, you can run all local refinements simultaneously.*

After all, four local refinements completed, we apply the **PostRefiner** tool to obtain the power spectrum-adjusted map and report the resolution after the local refinement. Again, this step has to be done independently for each refined group.

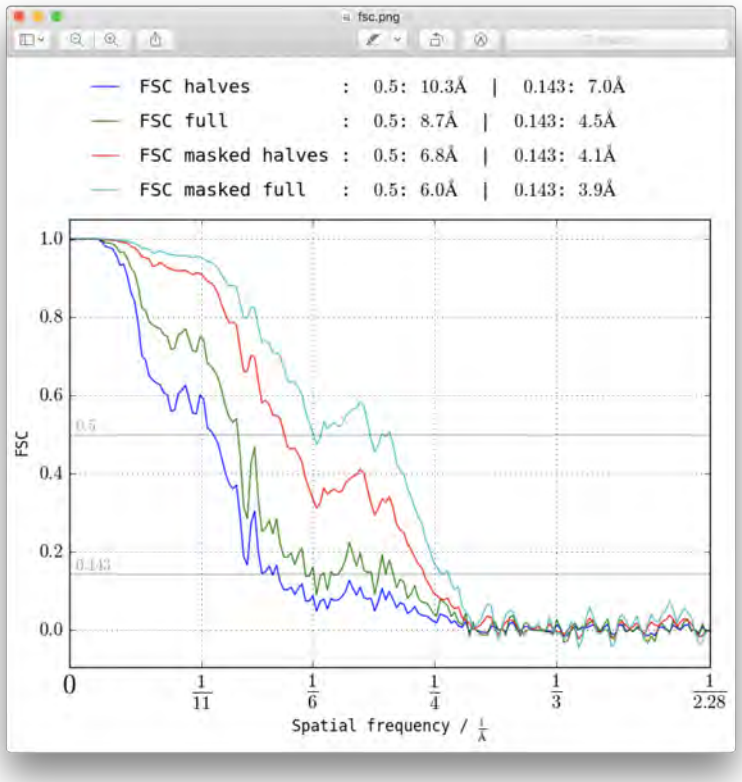
In the main window of the SPHIRE GUI click the **SORT3D** button and then **PostRefiner** button in the middle.

1. Specify the two final half maps (**Refine3D-Substack-Local_000/vol_0_unfil_ITERNR.hdf** and **Refine3D-Substack-Local_000/vol_1_unfil_ITERNR.hdf**).
2. Define **Sharpening-after-Meridien-Substack-Local_000** as output directory for **PostRefiner** results of local refinement of **Cluster_000.txt**.
3. Activate the **Apply adaptive mask** Flag.
4. Set the appropriate **Adaptive mask threshold** for the adaptive mask.
5. Use the exact same parameters as in the previous run of **PostRefiner**.
6. Click the ***Run command*** button.

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

```
tail -f Sharpening-after-Meridien-Substack-Local_000/log.txt

2018-03-09_16:49:28 =>---->>>analysis of enhancement <<<-----
2018-03-09_16:49:28 =>B_factor : 113.609749
2018-03-09_16:49:28 =>low-pass filter cutoff : 3.858461[A] (0.295455[absolute])
2018-03-09_16:49:28 =>low-pass filter falloff : 0.010000[absolute]
2018-03-09_16:49:28 =>max enhancement point : 103[pixels]
2018-03-09_16:49:28 =>max enhancement ratio : 570.998864
2018-03-09_16:49:28 =>1st zero pw spectrum point : 117[pixels]
2018-03-09_16:49:28 =>fallall width : 14[pixels]
2018-03-09_16:49:28 =>-----------------------------------------------------
2018-03-09_16:49:28 =>---------- >>>DONE<<<-----------
2018-03-09_16:49:28 =>-----------------------------------------------------
```

Now check the final FSC plot.



The resolution of locally refined group 000 is 4.1 Å according to FSC@0.143 between the two masked half-maps.

Now run the **PostRefiner** tool for the remaining 3 groups and check the respective output files. For the precelculated results, the reported resolution of the four groups of 3370, 2651, 1380 and 1226 particles is 4.1 Å, 4.1 Å, 4.7 Å and 4.8 Å, respectively.

Thus although we improved the homogeneity of each substack by 3D clustering, the achieved resolution after local refinement is rather limited by the low number of particles.

You can now display the maps with **UCSF Chimera**, morph them and compare them with the available X-ray structure. However, due to the localized continuous heterogeneity, detailed analysis of this flexibility at near-atomic resolution level of detail would require a much larger dataset.

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 highest resolution we achieved so far, was 3.5 Å FSC@0.143, and was obtained after 3D refinement, using a dataset cleaned in 2D by **ISAC2**. This value, however, represents an average resolution for the entire map. However, due to the intrinsic flexibility, as revealed by 3D clustering using **SORT3D**, and image processing artifacts, the resolution may vary locally. Using the **LOCALRES** tool, we will now compute the local resolution of our final map (the map we obtained so far with the highest average resolution).

In the main window of the SPHIRE GUI click the button ***LOCALRES*** on the left and then the ***Local Resolution*** button in the middle.

Fill out the following fields:

- **First half-volume:** Refine3D/vol_0_unfil_032.hdf

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

- **Second half-volume:** Refine3D/vol_1_unfil_032.hdf

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

- **3D mask:** Sharpening-after-meridien/vol_adaptive_mask.hdf

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

  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 region outlines by the mask. This will speed up the process.*

- **Output volume:** localres.hdf
- **FSC window size [Pixels]:** 7

  NOTE: *Real space FSC will be performed within small window areas. This parameter defines their size in pixels. The correct size depends on the resolution of the map. Setting the window size too small will result in inaccurate estimations that will vary widely from location to location, whereas too large windows will produce a coarsely sampled resolution map, which most likely will not reflect local resolution variations sufficiently well. 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 the pixel size is 1 Å, the window size should be at least 7 pixels. For a map with nominal resolution of 12 Å and pixel size 2 Å, the window size should be at least 5 pixels.*

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

  NOTE: *Thickness of the Fourier shell used to focus resolution estimation in reciprocal space. Setting a larger step size will speed up the process, but produce a less precise resolution map.*

- **Overall resolution [1/Pixel]:** 0.325

  NOTE: *Using a small window size might result in inaccurate and usually underestimated FSC values. This parameter allows calibrating the average value of all measured local resolutions to the overall resolution, which was estimated between the two half-maps in the previous step (in our case 3.5 Å). Use the resolution converter (**Use resolution [A]** button) to convert resolution in angstrom (3.5) into absolute frequency units.*

Advanced option:

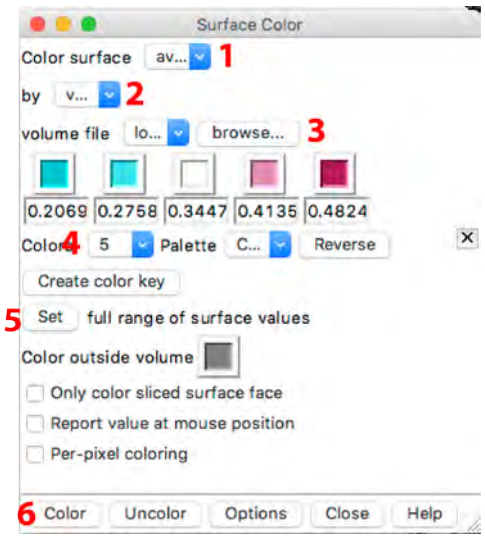
- **Save Angstrom local resolution:** YES

  NOTE: *The program outputs by default a local resolution map with resolution values given in absolute frequency units. Activate this option to obtain a second local resolution map with resolution values given in angstrom. This map will be used only for visualization in Chimera (see section below).*

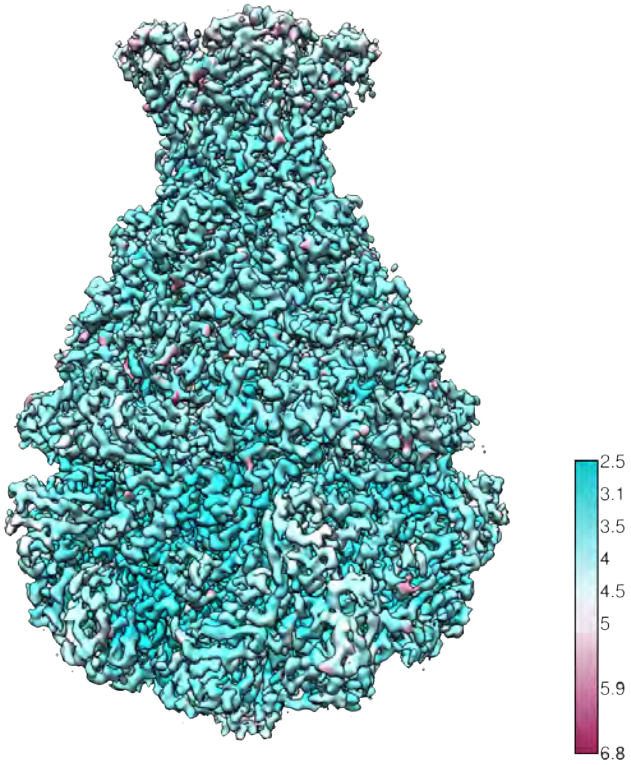
Specify the number of processors and submit the job to the queuing system of the cluster using an appropriate submission script by clicking the ***Run command*** button.

On our cluster, this job with 96 processes finished in about 4 minutes.

Load the resulting angstrom resolution volume (**localres_ang.hdf**) and the sharpened map (**Sharpening-after-meridien/vol_combined.hdf**) into **UCSF Chimera**. Open the **Surface Color** tool available in **UCSF Chimera**.



- (1) Click the **Options Button**. Color surface of volume: **vol_combined.hdf**.
- (2) By volume data value.
- (3) of volume file **localres_ang.hdf**.
- (4) Set number of colors to 5. Select the **Palette Cyan-Maroon**.
- (5) Click the **Set** button.
- (6) and then the **Color** button.



TIP: *The respective resolution values should be 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 the **project directory**.

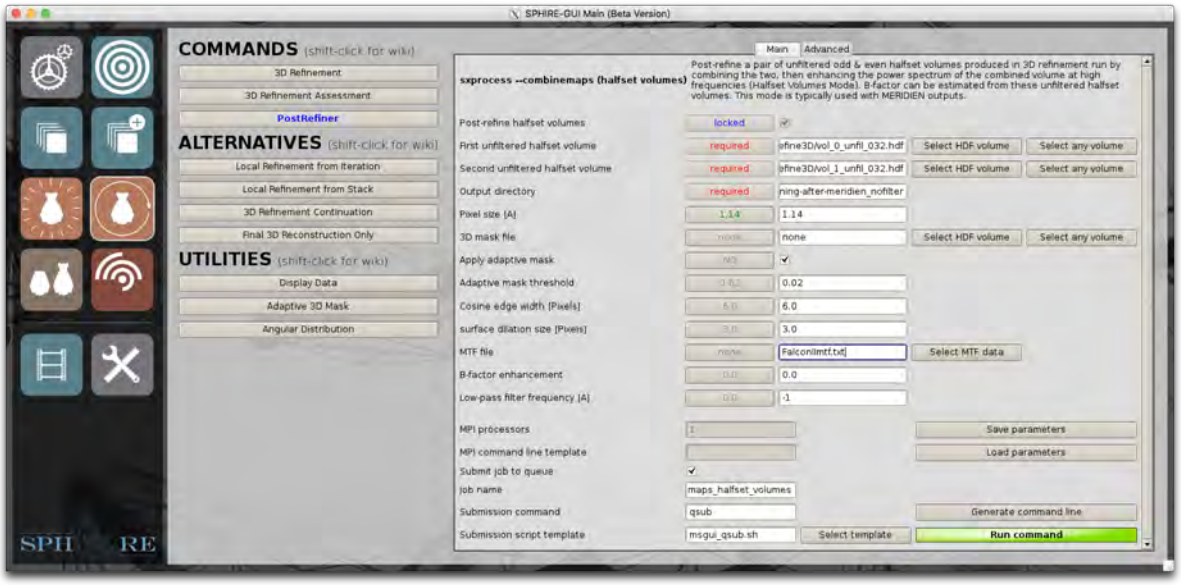
```
cp -Rp SphireDemoResults/localres*.hdf ./
```

Local Filtering

After calculating the local resolution map of our final structure, we will now filter the map accordingly. Such a locally filtered map is more suitable for interpretation and can be used for further analysis (i.e., model building) without the risk of over-interpreting individual features. In the locally filtered poorly resolved regions (corresponding to flexible domains) will be filtered to their respective local resolutions 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 in these regions.

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

In the main window of the SPHIRE GUI click the button ***MERIDIEN*** and then the ***PostRefiner*** button in the middle.



Fill out the following fields:

- **First unfiltered halfset volume:** Refine3D/vol_0_unfil_032.hdf
- **Second unfiltered halfset volume:** Refine3D/vol_1_unfil_032.hdf
- **Output directory:** Sharpening-after-meridien_nofilter
- **3D mask file:** none
- **Apply adaptive mask:** YES
- **MTF file:** FalconIImtf.txt
- **B-factor enhancement:** 0
- **Low-pass filter frequency [A]:** -1

  NOTE: *-1 means no low-pass filtration.*

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

```
tail -f Sharpening-after-meridien_nofilter/log.txt
```

The output will look like this:

```
2018-03-12_13:34:23 =>---------- >>>summary <<<------------

2018-03-12_13:34:23 =>resolution 0.5/0.143 are 3.90/ 3.46[A]

2018-03-12_13:34:23 =>B-factor is -26.00[A^2]

2018-03-12_13:34:23 =>FSC curves are saved in fsc_halves.txt, fsc_full.txt,
fsc_masked_halves.txt, fsc_masked_full.txt

2018-03-12_13:34:23 =>the final volume is Sharpening-after-meridien_nofilter/vol_combined.hdf

2018-03-12_13:34:23 =>guinierlines in logscale are saved in
Sharpening-after-meridien_nofilter/guinierlines.txt

2018-03-12_13:34:23 =>the final volume is not low-pass filtered.

2018-03-12_13:34:23 =>------------------------------------------------------

2018-03-12_13:34:23 =>---------- >>>DONE<<<------------

2018-03-12_13:34:23 =>------------------------------------------------------
```

Now we will apply the local filter to the output volume (**Sharpening-after-meridien_nofilter/vol_combined.hdf**).

In the main window of the SPHIRE GUI click the button ***LOCALRES*** and then the ***3D Local Filter*** button in the middle.

Set the following input fields:

- **Input volume:** Sharpening-after-meridien_nofilter/vol_combined.hdf

  NOTE: *Click the **Select HDF volume** button and use the file browser to select the sharpened but unfiltered map created in the previous step.*

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

  NOTE: *Click the **Select HDF volume** button and use the file browser to select local resolution volume with resolution values given in **absolute frequency** units.*

- **3D mask:** Sharpening-after-meridien_nofilter/vol_adaptive_mask.hdf

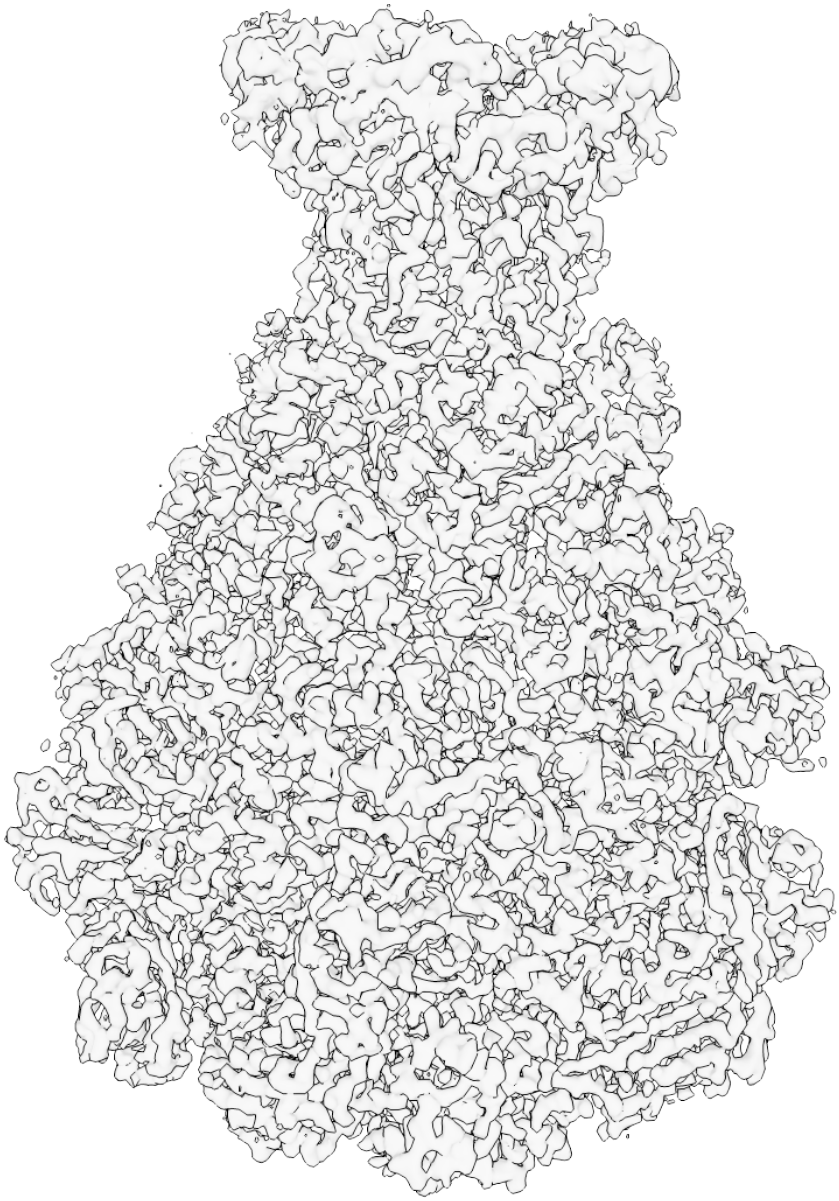
  NOTE: *Click the **Select HDF volume** button and use the file browser to select the soft-edge 3D mask obtained in the previous step.*

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

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

On our cluster this job finished in about 3 minutes.

Display the final map with **UCSF Chimera**. Compare the density with the available X-ray structure.



Congratulations

You have finished the tutorial successfully.

Now it is time to process your own data.

Acronyms

ANOVA	analysis of variance
BDB	Berkley Data Bank
cryo-EM	cryo-electron microscopy
Cs	Spherical Aberration
CTF	Contrast Transfer Function
EM	Electron Microscopy
FSC	Fourier shell correlation
GUI	graphical user interface
HDF	Hierarchical Data Format
ML	Maximum Likelihood
MPI	Message Passing Interface
MTF	modulation transfer function
PDB	protein data bank
SD	standard deviation
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 is in preparation.. Until it becomes available, please cite our recent SPHIRE JoVE video protocol:

Moriya, T., Saur, M., Stabrin, M., Merino, F., Voicu, H., Huang, Z., et al. High-resolution Single Particle Analysis from Electron Cryo-microscopy Images Using SPHIRE. J. Vis. Exp. (123), e55448, doi:10.3791/55448 (2017).

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 Photorhabdus luminescens toxins". In: *Nature* 495, pp. 520–523. DOI: 10.1038/nature11987.

Gatsogiannis, Christos, Felipe Merino, Daniel Prumbaum, Daniel Roderer, Franziska Leidreiter, Dominic Meusch, and Stefan Raunser (2016). "Membrane insertion of a Tc toxin in near-atomic detail". In: *Nature Structural and Molecular Biology* 23, pp. 884–890. DOI: 10.1038/nsmb.3281.

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.

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.

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.

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.

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.

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.

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.