Single-particle processing in RELION-1.3

Sjors H.W. Scheres
MRC Laboratory of Molecular Biology
Francis Crick Avenue, Cambridge Biomedical Campus
Cambridge CB2 0QH, UK
scheres@mrc-lmb.cam.ac.uk

July 22, 2014

Abstract

This tutorial provides an introduction to the use of RELION (release-1.3) for cryo-EM structure determination. RELION is a software package that performs an empirical Bayesian approach to (cryo-EM) structure determination by single-particle analysis. This tutorial covers the entire single-particle analysis workflow in RELION-1.3: CTF estimation; automated particle picking; particle extraction; 2D class averaging; 3D classification; high-resolution 3D refinement; the processing of movies from direct-electron detectors; and final map sharpening and local-resolution estimation.

Carefully going through this tutorial should take less than a day if you copy the precalculated results where indicated. After that, you should be able to run RELION on your own data. *If RELION is useful in your work, please cite our papers and tell your colleagues about it.*

Contents

1 Getting prepared

1.1 Recommended reading

The theory of RELION is described in detail in:

- S.H.W. Scheres (2012) "RELION: Implementation of a Bayesian approach to cryo-EM structure determination" *J. Struc. Biol.*, 180, 519-530.
- S.H.W. Scheres (2012) "A Bayesian view on cryo-EM structure determination" *J. Mol. Biol.*, 415, 406-418.

More general background and practical recommendations for maximum-likelihood methods is described in:

- F.J. Sigworth, P. Doerschuk, J.M. Carazo, S.H.W. Scheres (2010) "An introduction to maximum-likelihood methods in cryo-EM" *Meth. Enzym.*, 482, 263-294.
- S.H.W. Scheres (2010) "Classification of structural heterogeneity by maximum-likelihood methods" *Meth. Enzym.*, 482, 295-320.

This tutorial will follow the "Recommended Procedures" on the RELION wiki.

1.2 Install MPI

Note that in order to run RELION within reasonable amounts of time you'll need a computing cluster (or a multi-core desktop machine may suffice for small data sets) with an MPI (message passing interface) installation. To compile RELION, you'll need a mpi-devel package. The exact flavour (openMPI, MPICH, LAM-MPI, etc) or version will probably not matter much. If you don't have an mpi-devel installation already on your system, we recommend installing openMPI.

1.3 Install RELION

RELION is open-source software. Download it for free from the RELION wiki, and follow the installation instructions. If you're not familiar with your job submission system (e.g. Sun Grid Engine, PBS/TORQUE, etc), then ask your system administrator for help in setting up the qsub.csh script as explained in the installation instructions. Note that you will probably want to run so-called hybridly-parallel jobs, i.e. calculations that use both MPI for distributed-memory parallelization AND pthreads for shared-memory parallelization. Your job submission queueing system may require some tweaking to allow this. Again, ask your sysadmin for assistance.

1.4 Install CTFFIND3

CTF estimation is not part of RELION. Instead, RELION provides a wrapper to Niko Grigorieff's CTFFIND3 [4]. Please download it from Niko's website and follow his installation instructions.

1.5 Install ResMap

Local-resolution estimation is also not part of RELION. Instead, RELION provides a wrapper to Alp Kucukelbir's ResMap [3]. Please download it from Alp's website and follow his installation instructions.

1.6 Download the test data

This tutorial uses a test data set that is a subset of the micrographs used to calculate various beta-galactosidase reconstructions [9, 2, 13]. To reduce the computational load, these data were 2x down-sampled. The data may be downloaded and unpacked using the following commands:

```
wget ftp://ftp.mrc-lmb.cam.ac.uk/pub/scheres/relion13_tutorial.tar.gz
gunzip relion13_tutorial.tar.gz
tar -xf relion13_tutorial.tar
```

If you are using your own computer system to follow this tutorial, note that in order to run on a cluster, you'll need a `qsub.csh` script that has been configured for your particular queueing system. Ask your system administrator if you don't know whether you have the correct script.

2 Preprocessing

Although, in principle, RELION can use particles that have been extracted by a different program, this is NOT the recommended procedure. Many programs change the particles themselves, e.g. through phase flipping, band-pass or Wiener filtering, masking etc. All these are sub-optimal for subsequent use in RELION. Moreover, gathering all the required metadata into a correctly formatted STAR file may be prone to errors. Because re-extracting your particles in RELION is straightforward and very fast, the procedure outlined below is often a much easier (and better) route into RELION.

Also, several implementations of wrappers around RELION have now been reported (e.g. in EMAN2, XMIPP3 and APPION). Although we try to be helpful when others write these wrappers, we have absolutely no control over them and do not know whether their final product uses RELION in the best way. Therefore, in case of any doubt regarding results obtained with these wrappers, we would recommend following the procedures outlined in this tutorial.

2.1 Getting organised

We recommend to create a single directory per project, i.e. per structure you want to determine. We'll call this the project directory. **It is important to always launch the RELION graphical user-interface (GUI) from the project directory.** Inside the project directory you should make a separate directory to store all your raw (i.e. unprocessed by other software!) micrographs in MRC format. We like to call this directory `Micrographs/` if all micrographs are in one directory, or `Micrographs/15jan13/` and `Micrographs/23jan13/` if they are in different directories (e.g. because the micrographs were collected on different dates). If for some reason you do not want to place your micrographs inside the RELION project directory, then inside the project directory you can also make a symbolic link to the directory where your micrographs are stored. When you unpacked the tutorial test data, the Project directory (`betagal/`), and the corresponding `betagal/Micrographs` directory with all the recorded images were already created.

Go to the project directory, and see how many micrographs there are by typing

```
cd relion13_tutorial/betagal
ls Micrographs/*.mrc
```

We will start by making a so-called STAR-file, which contains a list of the micrographs we want to work on. RELION uses the STAR-file format to store any type of metadata (i.e. everything apart from the images, for which RELION uses MRC format). See the RELION wiki for a description of the STAR format.

Make sure you are inside the project directory, and launch the GUI by typing:

```
relion &
```

The white rectangle on the left contains many different *job-types*, each of which is dedicated to a different functionality of RELION. The first job-type is a special one called `General` and the information given here is used by many of the other job-types. The magnified pixel size for the down-sampled images in this tutorial is 3.54 Angstrom. A suitable size for the mask diameter (which should not exclude any density of the particle, but not include too much of the surrounding solvent either) is 200 Angstroms.

The next job-type `Micrograph inspection` may be used to generate the initial STAR file with the information about which micrographs to process. On the `I/O` tab, set:

- `Input micrographs: Micrographs/*.mrc`
- `Output STAR file: all_micrographs.star`
- `Picking rootname: manual`

On the `Display` tab, the best scale of the micrograph will depend on the size of your screen. We like to display micrographs at a contrast of 3 sigma (see the DisplayImages entry on the RELION wiki for more details), and low-pass filtered to for example 20 Angstroms. Ignore the `Colors` tab for now, and on the `Running` tab just press the `Run!` button. (Note that the `Run!` button decomes de-activated upon clicking it. To re-activate, press `ALT-R`; select the `Reactivate Run` option from the `File` menu, or click the job-type on the left-hand side.) This will open up a new GUI with a line for each micrograph. If a micrograph is selected, it has a checked selection box at the beginning of each line. You can now visualise each micrograph by clicking its `pick` button, and if necessary unselect bad micrographs by unchecking the selection box. Let's keep all micrographs, and save a STAR file with all of them using the `Save selection` option from the File menu on the top left of the window. This will create a file called `all_micrographs.star` in your project directory. Use the linux command `less` to look at it.

2.2 Estimating CTF parameters

We will use the RELION GUI as a wrapper to submit multiple CTFFIND3 jobs in parallel on your cluster. Click on the `CTF estimation` job-type on the side bar, and on the `I/O` tab, set:

- `Input micrographs for CTF: all_micrographs.star` contains all micrographs on which to estimate the CTF
- `Output STAR file: all_micrographs_ctf.star` is the name of the output STAR file with the CTF parameters for all micrographs
- `CTFFIND3 executable: XXX`. Or wherever you ResMap executable is. Note you can make your location the default by defining the environment variable RELION_CTFFIND3_EXECUTABLE in your shell.

- `Estimate CTF on window size (pix): -1` (which will skip windowing the micrographs before CTF estimation)

On the `Microscopy` tab you set the experimental conditions:

- `Spherical aberration (mm): 2`
- `Voltage (kV): 300`
- `Amplitude contrast: 0.1` (although amplitude contrast is known to be less, giving values of around 10% has been shown to yield better results for many structures. This may be because of unmodelled low-frequency inelastic scattering.)
- `Physical pixel size on detector (um): 14`

And on the `CTFFIND3` tab, you provide the parameters for Niko's program (see his documentation for their exact meaning):

- `FFT box size (pix): 512`
- `Minimum resolution (A): 100`
- `Maximum resolution (A): 7`
- `Minimum defocus cvalue (A): 5000`
- `Maximum defocus cvalue (A): 50000`
- `Defocus step size (A): 500`
- `Amount of astigmatism (A): 0`

On the running tab you may specify on how many processors you would like to run the CTF estimation. On our 3GHz Xeon CPUs, each micrograph takes approximately 5 minutes. If this takes too long for you to wait for, you can also copy our precalculated results into your Micrographs directory by typing:

```
cp PrecalculatedResults/Micrographs/*.log Micrographs/.
cp PrecalculatedResults/Micrographs/*.ctf Micrographs/.
cp PrecalculatedResults/Micrographs/*.com Micrographs/.
cp PrecalculatedResults/all_micrographs_ctf.star .
```

Otherwise, you can monitor your job's progress through the `stdout` output. Once it has finished there will be additional files inside your `Micrographs` directory: the `.ctf` file contains an image in `MRC` format with the computed power spectrum and the fitted CTF model; the `_ctffind3.log` file contains the output from CTFFIND3; and the `_ctffind3.com` file contains the script that was used to launch CTFFIND3.

Use the `Display` button on the main GUI to select the `all_micrographs_ctf.star` file in your Project Directory. Make sure the `rlnCtfImage` is selected in the

"Display" menu, and hit the red `Display!` button on the pop-up window. You may want to play with the option `Sort images on:` and choose either `rlnDefocusU` or `rlnCtfFigureOfmerit`. If you see CTF models that are not satisfactory, you can tweak the CTFFIND3-parameters in the corresponding `_ctffind3.com` file, and run the script again for a particular micrograph. Alternatively, if many models are wrong, you could also make a new STAR file with those, and then re-run the estimation of all of them by changing the CTFFIND3-parameters in the RELION GUI.

2.3 Manual particle picking

As of release 1.3, RELION has a functionality to manually and automatically pick particles. The automated particle-picking is reference-based, and references are typically generated by manually picking some particles, calculating 2D class averages from those particles, and then use 2D class averaging to generate the references for the auto-picking procedure.

Picking particles manually is a personal experience! If you don't like to pick particles in RELION, we also support coordinate file formats for Jude Short's Ximdisp [11] (with any extension); for XMIPP-2.4 [10] (with any extension); and for Steven Ludtke's e2boxer.py [12] (with a `.box` extension). If you use any of these, make sure to save the coordinate files as a text file with the same root-name, but a different extension as the micrograph, e.g. `Micrographs/006.box` for micrograph `Micrographs/006.mrc`, etc.

To manually pick particles in RELION, select the `Micrograph inspection` job-type again, and this time input the `all_micrographs_ctf.star` file. As the input STAR file contains the CTF estimation results, this time the pop-up window with all the micrographs will also contain a `CTF` button, which display the CTF model for each micrograph. Pick the first 2-3 micrographs in order to get approximately 1,000 particles. If you are too impatient to do this, you may also copy the ones we picked previously by typing:

```
cp PrecalculatedResults/Micrographs/*_manual.star Micrographs/.
```

Note that one does not really need the CTF parameters to pick particles (perhaps other than for discarding bad micrographs). Therefore, in real-life applications, we often first submit the CTF estimation job to the cluster, and while that runs we manually pick a few micrographs to get enough particles to calculate references for the auto-picking.

2.4 Extracting and normalising particles

Once you have a coordinate file for every micrograph that you want to pick particles from, you can extract the corresponding particles and gather all required metadata through the `Particle extraction` job-type on the RELION GUI. On the corresponding `I/O` tab, set:

- `Micrograph STAR file: all_micrographs_ctf.star`

- `Coordinate-file suffix: _manual.star`
- `Extract rootname: particles_manual`

On the `extract` tab you set the parameters for the actual particle extraction:

- `Particle box size: 100` This should always be an even number!
- `Invert contrast?: Yes` makes white instead of black particles
- `Rescale particles?: No`. Unless computation becomes a problem, we typically do not downscale our particles. (Note that when downscaling, also the down-scaled image size should be an even number!)
- `Normalize particles?: Yes`. We always normalize.
- `Stddev for white dust removal: -1`
- `Stddev for black dust removal: -1` We only remove very white or black outlier pixels if we actually see them in the data. In such cases we would use stddev values of 5 or so. In this data set there are no outlier pixels, so we don't correct for them, and leave the default values at -1 (i.e. don't do anything).

On the `movies` tab we indicate these are not movies we're extracting from, and we submit the job using a single MPI processor on the `Running` tab. This will extract all manually picked particles from the micrographs, invert their contrast, normalise them, and gather all metadata in a file called `particles_manual.star`, which will be placed in your Project directory. We prefer to always keep STAR files with (different selections of) our particles in the Project directory, although this is optional. You can look at your STAR file by typing:

```
less particles_manual.star
```

Your particles will be extracted into MRC stacks (which always have an `.mrcs` extension in RELION) in a new directory called `Particles/Micrographs/`. It's always a good idea to quickly check that all has gone OK by opening one of them using the `Display` button on the main GUI. Alternatively, you can also select the `particles_manual.star` from the `Display` pop-up window to see all of them (e.g. sorted on defocus value?).

2.5 Getting templates for auto-picking

To calculate templates for the subsequent auto-picking of all micrographs, we will use reference-free 2D class averaging, which can be found under job-type `2D classification` on the main GUI. We will use the same job-type later on to throw away bad particles. On the `I/O` tab, set:

- `Input images STAR file: particles_manual.star`

- `Output rootname: Class2D/manual`, where the Class2D directory will be created upon submitting the ob from the GUI.
- `: Number of classes: 10`. For cryo-EM data we like to use on average at least approximately 100 particles per class. For negative stain one may use fewer particles per class.

On the `CTF` tab, set:

- `Do CT-correction?: Yes`, which will perform full phase+amplitude correction inside the Bayesian framework.
- `Have data been phase-flipped?: No`. This option is only useful if you pre-processed your data outside RELION.
- `Ignore CTFs until first peak?: No`. This option is only occasionally useful, when amplitude correction gives spuriously strong low-resolution components.

On the `Optimisation` tab, set:

- `Number of iterations: 25`. We often don't change this.
- `Regularisation parameter T: 2`. For the exact definition of T, please refer to [7]. For cryo-EM 2D classification we typically use values of T=1-2, and for 3D classification values of 3-4. For negative stain sometimes slightly lower values are better. In general, if your class averages appear noisy, then lower T; if your class averages remain too-low resolution, then increase T. The main thing isto be aware of overfitting high-resolution noise.
- `Mask individual particles with zeros?: Yes`.
- `Limit resolution E-step to (A): -1`. If a positive value is given, then no frequencies beyond this value will be included in the alignment. This can also be useful to prevent overfitting. Here we don't really need it, but it could have been set to 10-15A anyway.

On the `Sampling` tab we hardly ever change the defaults. Five degrees angular sampling is enough for most projects, although some large icosahedral viruses may benefit from finer angular samplings. In that case, one could first run 25 iterations with a sampling of 5 degrees, and then `Continue old run` (button on top of main GUI) for an additional five iteration (by setting `Number of iterations: 30` on the `Optimisation` tab) with a sampling of say 2 degrees. For this data set, this is NOT necessary at all. It is useful to note that the same Continue button may also be used to resume a job that has somehow failed before, in which case one would not change any of the parameters.

On the `Running` tab, specify the `Number of MPI processors` and `threads` to use. The total number of requested CPUs, or cores, will be the product of the two values. Threads offer the advantage of more efficient RAM usage, whereas

MPI parallelization scales better than threads. Often, for 3D classifications and refinements you will probably want to use many threads in order to share the available RAM on each (multi-core) computing node. 2D classification is less memory-intensive, so you may not need so many threads. However, the points where communication between MPI processors (the bottle-neck in scalability there) becomes limiting in comparison with running more threads, is different on many different clusters, so you may need to play with these parameters to get optimal performance for your setup.

Wait until the job finishes, or just copy our precalculated results:

```
mkdir Class2D
cp -r PrecalculatedResults/Class2D/manual* Class2D/.
```

Then, have a look at the resulting 2D class averages by clicking the `Display` button on the main GUI and selecting the file `Class2D/manual_it025_model.star`. On the pop-up window, you may want to look at the class averages in a specific order, e.g. based on `rlnClassDistribution` (in reverse order) or on `rlnAccuracyRotations`.

From the 10 class averages, select as many good-quality, representative views as deemed necessary for auto-picking. Don't repeat very similar views, and don't include bad class averages. Selection is done by left-mouse clicking on the class averages. Note that you can have a look at the aligned particles of certain classes by right-mouse clicking and selecting `Show particles from selected classes`. You can save your selection of class averages from the same pop-up menu using the `Save STAR with selected classes` option. Alternatively, you can copy our selection:

```
cp -r PrecalculatedResults/manual_7classes.star .
```

2.6 Auto-picking

We will now go back to all micrographs, and use the selected 2D class averages as templates in the `auto-picking` job-type. However, before we will run the auto-picking on all micrographs, we will need to optimise two of its main parameters (on the `autopicking` tab: the `Picking threshold` and the `Minimum inter-particle distance`. This will be done on only a few micrographs in order to save time. Therefore, we will need a STAR file with the CTF information of only 2 or 3 micrographs. One should use representative micrographs for the entire data set, e.g. a high and a low-defocus one, and/or with thin or thick ice. These few micrographs should be saved in a STAR file. One can use a text editor to copy lines from the `all_micrographs_ctf.star` file, or use the `Micrograph inspection` job-type and save a STAR file with the selected micrographs from the GUI. You can also copy our selection by typing:

```
cp -r PrecalculatedResults/2mics_ctf.star .
```

Then, on the `I/O` tab, set:

- `Input micrographs for autopick: 2mics_ctf.star`
- `Autopick rootname: autopick`

On the `References` tab, set:

- `References: manual_7classes.star` or however you called your own STAR file with the selected 2D class averages.
- `Lowpass filter references (A): 20`. It is very important to use a low-pass filter that is significantly LOWER than the final resolution you aim to obtain from the data, in order to prevent the "Einstein-from-noise" effect.
- `Angular sampling (deg): 5`
- `References have inverted contrast?: Yes`. Because we have black particles in the micrographs, and the references we will use are white.
- `Are References CTF corrected?: Yes`. Because we performed 2D class averaging with the CTF correction.
- `Ignore CTFs until first peak: No`. This is not often used. Only in cases with suprious low-frequency components in the images this option is sometimes useful.

On the `autopicking` tab, set:

- `Picking threshold: 0.8`
- `Minimum inter-particle distance (A): 200`
- `Write FOM maps?: Yes` See below.
- `Read FOM maps?: No`

On the `Running` tab, use only a single MPI processor (see below). Using the 2kx2k micrographs in this tutorial, these calculations take about 4-5 minutes per micrograph on our computer. When using more common 4kx4k micrographs, this is typically approximately 5 minutes *per reference* and per micrograph.

The expensive part of this calculation is to calculate a probability-based figure-of-merit (related to the cross-correlation coefficient between each rotated reference and all positions in the micrographs. This calculation is followed by a much cheaper peak-detection algorithm that uses the threshold and minimum distance parameters mentioned above. Because these parameters need to be optimised, the program will write out so-called FOM maps as specified on the `References` tab. These are two large (micrograph-sized) files per reference. To avoid running into hard disc I/O problems, the autopicking program can only be run sequentially (hence the single MPI processor above) when writing out FOM maps.

Once the FOM maps have been written to disc they can be used to optimise the picking parameters much faster. First, examine the auto-picked particles with the current settings using the `Micrograph inspection` job-type. On the I/O tab there, change the rootname to `autopick`, and optionally on the `Colors` tab you may choose to color each particle according to the figure-of-merit for the autopicking by setting:

- `Blue<>red color particles?: Yes`
- `MetaDataLabel for color: rlnAutopickFigureOfMerit:`
- `STAR file with color label:` Leave this empty.
- `Blue value: 1`
- `Red value: 0`

Executing the job on the `Running` tab will produce a similar GUI with green `pick` and `CTF` buttons as before. The former will launch a display of the micrographs with the particles colored from red (very low FOM) to blue (very high FOM).

Open both micrographs and decide whether you would like to pick more or less particles (i.e. decrease or increase the threshold) and whether they could be closer together or not (fot setting the minimum inter-particle distance). You can leave the display windows for these micrographs open.

In the `Auto-picking` job-type, *leaving everything else the same* you can now change these two parameters. Also, change:

- `Write FOM maps?: No`
- `Read FOM maps?: Yes`

This will then re-read the previously written FOM maps from disc instead of re-doing all FOM calculations. The subsequent calculation of the new coordinates will then be done in a few seconds. Afterwards, you can right-click in the micrograph display windows and select `Reload coordinates` from the pop-up menu to read in the new set of coordinates. This way you can quickly optimise the two parameters.

Once you know the parameters you want to use for auto-picking of all micrographs, you replace `2mics_ctf.star` on the `I/O` tab for `all_micrographs_ctf.star`, and on the `autopicking` tab set:

- `Picking threshold: 0.4`
- `Minimum inter-particle distance (A): 110`
- `Write FOM maps?: No`
- `Read FOM maps?: No`

This time, the job may be run in parallel. On the `Running` tab, specify the number of cores to run auto-picking on. The maximum useful number of MPI processors is the number of micrographs in the input STAR file. Alternatively, you can copy our auto-picked coordinates using:

```
cp PrecalculatedResults/Micrographs/*_autopick.star Micrographs/.
```

You could again check these using the `Micrograph inspection` job-type, specifying `all_micrographs_ctf.star` on the `I/O` tab. Middle-mouse clicking in the micrograph display windows will delete particles. Often, we use this to go manually over the auto-picking results to check for obvious, high-contrast false positives (which the program seems to produce). The `Save STAR with coordinates` option from the right-mouse pop-up menu will over-write the STAR file with the new set of coordinates from which the false positives were removed.

Also, once you're happy with the overall results, in order to save disc space you may want to delete the FOM-maps that were written in the optimisation step by typing (after making sure there are no other important ".spi" files there...):

```
rm Micrographs/*.spi
```

Once you are happy with the auto-picked coordinates, you will need to re-run the `Particle extraction` job-type, keeping everything as before, but change:

- `Micrograph STAR file: all_micrographs_ctf.star`
- `Coordinate-file suffix: _autopick.star`
- `Extract rootname: particles_autopick`

2.7 Particle sorting

As of release 1.3, RELION also has a new functionality to quickly sort particles based on the difference between each particle and their aligned, CTF-corrected reference. The sorting program is accessible from the `Particle sorting` job-type and can be run whenever references are available for each particle, i.e. after auto-picking, 2D or 3D classification, or 3D auto-refinement. To demonstrate its use, we will now run it after the auto-picking. On the `I/O` tab, set:

- `Input particles to be sorted: particles_autopick.star`. The file that was generated by the extraction program after the autopicking. In case of 2D or 3D classification or 3D auto-refinement, the `_data.star` from the last iteration should be given.
- `References: manual_7classes.star`. This STAR file should contain the references used for the auto-picking in this case. In case of 2D or 3D classification or 3D auto-refinement, the `_model.star` from the last iteration should be given.

- `Are References CTF corrected?: Yes`
- `Ignore CTFs until first peak?: No`

This program runs very quickly (often in minutes) so does not need to be run with many MPI processors. It will not produce any new files, but will add a new (or update a) column called `rlnParticleSelectZScore` in the input particle STAR file. You can then use the `Display` button on the main GUI to visualise this STAR file and sort the particles based on this column. There will (should) then be a strong tendency to have "good" particles at the top and "bad" particles at the bottom of the display. One could scroll up from the bottom and decide a given point from which below all particles need to be discarded and use the right-mouse pop-up menu to select `Select all above` and/or (de-)select particles on a one-by-one basis. A new STAR file with all selected (red-boxed) particles can then be saved by choosing `Save STAR with selected images` from using the same pop-up menu. Alternatively, copy our selection by typing:

```
cp PrecalculatedResults/particles_autopick_sort.star .
```

3 Reference-free 2D class averaging

We always use reference-free 2D class averaging as a great tool to throw away bad particles. Although we always try (very hard!) to only include good particles for the particle extraction step in the previous section (manually supervising the auto-picking results and sorting the particles), most of the times there are still particles in the data set that do not belong there. Because they do not average well together, they often go to relatively small classes that yield ugly 2D class averages. Throwing those away then becomes a good way of cleaning up your data.

3.1 Running the job

Most options will remain the same as explained when we were generating templates for the auto-picking in the previous section, but on the `I/O` tab of the `2D classification` job-type, set:

- `Input images STAR file: particles_autopick_sort.star`
- `Output rootname: Class2D/autopick_sort`
- `Number of classes: 100`. Again, use at least approximately 100 cryo-EM particles per class. However, to save computational resources, for very large data sets we typically do not use more than 200-300 classes.

This job is much more expensive than the previous one (the algorithm scales linearly with the number of particles and the number of classes), so run it in parallel on your cluster. Alternatively, copy our results by typing:

```
cp -r PrecalculatedResults/Class2D/autopick_sort* Class2D/.
```

Again, after the job has finished, display the `_model.star` file from the last iteration from the `Display` button on the main GUI. Reverse-order sorting the classes on `rlnClassDistribution` is often useful. Now select all nice-looking classes by clicking on them (and/or using the right-mouse pop-up menu option `Select all classes above`). At this point, if you would have used a low threshold in the auto-picking procedure, you should be very wary of "Einstein-from-noise" classes, which look like low-resolution ghosts of the templates used to pick them, on which high-resolution noise may have accumulated. Avoid those in the selection. After all good classes have been selected use the pop-up menu option `Save STAR with particles from selected classes` to save a new STAR file called `particles_autopick_sort_class2d.star`, or copy ours by typing:

```
cp PrecalculatedResults/particles_autopick_sort_class2d.star .
```

Note that this procedure of selecting good classes may be repeated several times. Also, note that after the 2D classification one could re-run the sorting algorithm to identify remaining outliers in the data.

3.2 Analysing the results in more detail

If you are in a hurry to get through this tutorial, you may skip this sub-section. It contains more detailed information for the interested reader.

For every iteration of 2D or 3D classification RELION performs, it writes out a set of files. For the last iteration of our 2D class averaging calculation these are:

- `Class2D/autopick_sort_it025_classes.mrcs` is the MRC stack with the resulting class averages. These are the images that will be displayed in the RELION GUI when you select the `_model.star` file from the `Display` button on the main GUI. Note that RELION performs full CTF correction (if selected on the GUI), so your class averages are probably white on a black background. If the data is good, often they are very much like projections of a low-pass filtered atomic model. The quality of your 2D class averages are a very good indication of how good your 3D map will become.

- `Class2D/autopick_sort_it025_model.star` contains the model parameters that are refined besides the actual class averages (i.e. the distribution of the images over the classes, the spherical average of the signal-to-noise ratios in the reconstructed structures, the noise spectra of all groups, etc. Have a look at this file using the `less` command. In particular, check the distribution of particles over each class in the table `data_model_classes`. If you compare this with the class averages themselves, you will see that particles with few classes are low-resolution, while classes with many particles are high-resolution. This is an important feature of the Bayesian approach, as averaging over fewer particles will naturally lead to lower signal-to-noise ratios in the average. The estimated spectral signal-to-noise ratios for each class are stored in the `data_model_class_N` tables, where N is the number of each class. Likewise, the estimated noise spectra for each group are stored in the tables called `data_model_group_N`. The table `data_model_groups` stores a refined intensity scale-factor for each group: groups with values higher than one have a stronger signal than the average, relatively low-signal groups have values lower than one. These values are often correlated with the defocus, but also depend on accumulated contamination and ice thickness.

- `Class2D/autopick_sort_it025_data.star` contains all metadata related to the individual particles. Besides the information in the input `particles_autopick_sort.star` file, there is now additional information about the optimal orientations, the optimal class assignment, the contribution to the log-likelihood, etc. Note that this file can be used again as input for a new refinement, as the STAR file format remains the same.

- `Class2D/autopick_sort_it025_optimiser.star` contains some general information about the refinement process that is necessary for restarting

an unfinished run. For example, if you think the process did not converge yet after 25 iterations (you could compare the class averages from iterations 24 and 25 to assess that), you could select `Continue old run` from the GUI, on the `I/O` tab select this file for `Continue from here`, and then set `Number of iterations: 40` on the `Optimisation` tab. The job will then restart at iteration 26 and run until iteration 40. You might also choose to use a finer angular or translational sampling rate on the `Sampling` tab. Another useful feature of the optimiser.star files is that it's first line contains a comment with the exact command line argument that was given to this run.

- `Class2D/autopick_sort_it025_sampling.star` contains information about the employed sampling rates. This file is also necessary for restarting.

3.3 Making groups

If you are in a hurry to get through this tutorial, you may skip this sub-section. It contains more detailed information for the interested reader.

RELION groups particles together to do two things: estimate their average noise power spectrum and estimate a single-number intensity scale factor that describes differences in overall signal-to-noise ratios between different parts of the data, e.g. due to ice thickness, defocus or contamination.

The default behaviour is to treat all particles from each micrograph as a separate group. This behaviour is fine if you have many particles per micrograph, but when you are using a high magnification, your sample is very diluted, or your final selection contains only a few particles per micrograph, then the estimation of the intensity scale factor (and the noise spectra) may become unstable. We generally recommend to have at least 10-20 particles in each group, but do note that initial numbers of particles per group may become much smaller after 2D and 3D classification.

In cases with few particles per micrograph we recommend to group particles from multiple micrographs together. For this purpose, the new GUI in RELION-1.3 implements a convenient functionality: when selecting a `_model.star` file from the `Display` button on the main GUI, the pop-up window will have a line at the bottom that will allow to "Regroup seelcted particles in number of groups". Setting any value larger than 0, will result in an output STAR-file with particles from the selected particles that will (approximately) have the number of requested groups. This way, complicated grouping procedures in previous releases of RELION may be avoided. As the micrographs in this tutorial do contain sufficient particles, we will not use this procedure now.

Please note that the groups in RELION are very different from defocus groups that are sometimes used in other programs. RELION will always use per-particle (anisotropic) CTF correction, irrespective of the groups used.

4 Unsupervised 3D classification

All data sets are heterogeneous! The question is how much you are willing to tolerate. RELION's 3D multi-reference refinement procedure provides a powerful unsupervised 3D classification approach. However, you will still need a single, low-resolution consensus model to start with. We will use a low-resolution beta-galactosidase model that was generated from an early crystal structure (PDB-ID 3I3E). It is already in your project directory and is called `3i3e_lp50A.mrc`. In a typical project you may not have a good initial model. In that case you would need to generate one in a different program, as RELION as yet does not do initial model generation.

We already made the reference model with the correct pixel and box size. If you have a model for which this is not the case, then you may use the program `relion_image_handler` from the command line to change the pixel size (using options `--angpix` and `--rescale_angpix`) and the box size (`--new_box`).

4.1 Running the job

Unsupervised 3D classifcation may be run from the `3D classification` job-type. On the `I/O` tab set:

- `Input images STAR file: particles_autopick_sort_class2d.star`
- `Output rootname: Class3D/run1`
- `Number of classes: 4`

On the `Reference` tab set:

- `Reference map: 3i3e_lp50.mrc`
- `Ref. map is on absolute greyscale: No`. Any map that is not reconstructed from the same data in RELION or XMIPP should be considered as not being on the correct greyscale.
- `Initial low-pass filter (A): 50`. One should NOT use high-resolution starting models as they may introduce bias into the refinement process. As also explained in [6], one should filter the initial map as much as one can. For ribosome we often use 70Å, for smaller particles we typically use values of 40-60Å.
- `Symmetry: C1`. Although we know that this sample has D2 symmetry, it is often a good idea to perform an initial classification without any symmetry, so bad particles can get separated from proper ones.

On the `CTF` tab set:

- `Do CTF correction?: Yes`

- `Has reference been CTF-corrected?: Yes`. As this model was made from a PDB file, it's amplitudes have not been affected by any CTF. This means it could be considered as "CTF-corrected". Models made from most other EM packages should be considered NOT to be CTF-corrected, as not many packages perform a proper low-frequency amplitude-correction (although some may do so: therefore one could check slices through the model and if they look white on black the model was probably properly CTF-corrected; if they look grey-ish with a black halo then the model probably wasn't correced well). Models previously made in RELION with CTF-correction are properly corrected.

- `Have data been phase flipped?: No`

- `Ignore CTFs until first peak?: No`

On the `Optimisation` tab set:

- `Number of iterations: 25`. We often don't change this.

- `Regularisation parameter T: 2`. For the exact definition of T, please refer to [7]. For cryo-EM 2D classification we typically use values of T=1-2, and for 3D classification values of 2-4. For negative stain sometimes slightly lower values are better. In general, if your class averages appear noisy, then lower T; if your class averages remain too-low resolution, then increase T. The main thing isto be aware of overfitting high-resolution noise. We happened to use a value of 2 in our pre-calculated results. Probably a value of 4 would have worked equally well...

- `Mask individual particles with zeros?: Yes`.

- `Reference mask (optional):` . Leave this empty. This is the place where we for example provided large/small-subunit masks for our focussed ribosome refinements (also see section 9). If left empty, a spherical mask with the particle diameter given on the `General` job-type will be used. This introduces the least bias into the classification.

- `Limit resolution E-step to (A): -1`. If a positive value is given, then no frequencies beyond this value will be included in the alignment. This can also be useful to prevent overfitting. Here we don't really need it, but it could have been set to 10-15A anyway.

On the `Sampling` tab one usually does not need to change anything. However, because these are downsampled images (with a pixel size of 3.54 Å, we decided to limit the `Offset search range(pix)` to 3 in our precalculated results in order to speed things up. Again, on the `Running` tab, one may specify the `Number of MPI processors` and `threads` to use. As explained for the `2D classification` job-type, 3D classification takes more memory than 2D classification, so often more threads are used. However, in this case the images are rather small and RAM-shortage may not be such a big issue.

Instead of running the job yourself, you can also copy our precalculated results by typing:

```
cp -r PrecalculatedResults/Class3D .
cp PrecalculatedResults/particles_autopick_sort_class2d_class3d.star .
```

When analyzing the refined reconstructions, it is extremely useful to also look at them in slices, not only as a thresholded map in for example UCSF Chimera. In the slices view you will get a much better impression of unresolved heterogeneity, which will show up as fuzzy or streaked regions in the slices. Slices also give a good impression of the flatness of the solvent region. Use the `Display` button and select any of the reconstructions from the last iteration (e.g. `Class3D/run1_it025_class001.mrc`) to open a slices-view in RELION.

When looking at your rendered maps in 3D, e.g. using UCSF Chimera, it is often a good idea to fit them all into the best one, as maps may rotate slightly during refinement. In Chimera, we use the `Tools -> Volume Data -> Fit in Map` tool for that. For looking at multiple maps alongside each other, we also like the `Tools -> Structure Comparison -> Tile Structures` tool, combined with the `independent` center-of-rotation method on the `Viewing` window.

As was the case for the 2D classification, one can again evaluate the results from the `Display` button on the main GUI and selecting the `_model.star` file from the last iteration. This will show central slices through the 4 refined models, and generating STAR files with particles from the best classes can again be done through the right-mouse pop-up menu. We selected class 1 and class 3 and saved a new particle STAR-file called `particles_autopick_sort_class2d_class3d.star`.

4.2 Analysing the results in more detail

Again, if you are in a hurry to get through this tutorial, you may skip this sub-section. It contains more detailed information for the interested reader.

The output files are basically the same as for the 2D classification run (we're actually using the same code for 2D and 3D refinements). The only difference is that the map for each class is saved as a separate MRC map, e.g. `run1_it025_class00?.mrc`, as opposed to the single MRC stack with 2D class averages that was output before.

As before, smaller classes will be low-pass filtered more strongly than large classes, and the spectral signal-to-noise ratios are stored in the `data_model_class_N` tables (with $N = 1, \ldots, K$) of the `_model.star` files. Perhaps now is a good time to introduce two handy scripts that are useful to extract any type of data from STAR files. Try typing:

```
relion_star_printtable Class3D/run1_it025_model.star
  data_model_class_1 rlnResolution rlnSsnrMap
```

It will print the two columns with the resolution (`rlnResolution`) and the spectral signal-to-noise ratio (`rlnSsnrMap`) from table `data_model_class_1` to

the screen. You could redirect this to a file for subsequent plotting in your favourite program. Alternatively, if `gnuplot` is installed on your system, you may type:

```
relion_star_plottable Class3D/run1_it025_model.star
  data_model_class_1 rlnResolution rlnSsnrMap
```

To check whether your run had converged, (as mentioned above) you could also monitor:

```
grep _rlnChangesOptimalClasses Class3D/run1_it???_optimiser.star
```

As you may appreciate by now: the STAR files are a very convenient way of handling many different types of input and output data. Linux shell commands like `grep` and `awk`, possibly combined into scripts like `relion_star_printtable`, provide you with a flexible and powerful way to analyze your results.

5 High-resolution 3D refinement

Once a subset of sufficient homogeneity has been selected, one may use the 3D auto-refine procedure in RELION to refine this subset to high resolution in a fully automated manner. This procedure employs so-called *gold-standard* Fourier Shell Correlation (FSC) calculations to estimate resolution, so that overfitting may be completely avoided [9]. Combined with a novel procedure to estimate the accuracy of the angular assignments [8], it automatically determines when a refinement has converged. Thereby, this procedure requires very little user input, i.e. it remains objective, and has been observed to yield excellent maps for "normal" data sets (i.e. not extremely difficult ones, also see section 9). Another advantage is that one typically only needs to run it once, as there are hardly any parameters to optimize.

5.1 Running the job

Select the `3D auto-refine` job-type on the RELION GUI.

The input parameters on the GUI largely remain the same as for the 3D classification run type, although some of them are no longer available. The orientational sampling rates on the `Sampling` tab will only be used in the first few iterations, from there on the algorithm will automatically increase the angular sampling rates until convergence. Therefore, for all refinements with less than octahedral or icosahedral symmetry, we typically use the defaults of `Angular sampling interval: 7.5 degrees`, and `Local searches from auto-sampling: 1.8 degrees`. Only for higher symmetry refinements, we use 3.7 degrees and perform local searched from 0.9 degrees.

Just select the `particles_autopick_sort_class2d_class3d.star` as the input STAR file, and we could use one of the refined map from the 3D classification run (e.g. `Class3D/run2_it025_class003.mrc`) as initial model. Alternatively, we could use the same `3i3e_lp50A.mrc` model we used before; it would probably make little difference. Again, it is best to low-pass filter the initial map strongly to prevent bias towards high-frequency components in the map, and to maintain the "gold-standard" of completely independent refinements at resolutions higher than the initial one. Let's again use 50Å. As the MPI nodes are divided between one master (who does nothing else than bossing the others around) and two sets of slaves who do all the work on the two half-sets, it is most efficient to use an odd number of MPI processors. Memory requirements may increase significantly at the final iteration, as all frequencies until Nyquist will be taken into account, so for larger sized images than the ones in this test data set you may want to run with as many threads as you have cores on your cluster nodes.

Again, if you cannot run the procedure yourself, you can copy our precalculated results. In that case, just type:

```
mkdir Refine3D
cp PrecalculatedResults/Refine3D/run2* Refine3D/.
```

5.2 Analysing the results

Also the output files are largely the same as for the 3D classification run. However, at every iteration the program writes out two `run2_it0??_half?_model.star` and two `run2_it0??_half?_class001.mrc` files: one for each independently refined half of the data. Only upon convergence a single `run2_model.star` and `run2_class001.mrc` file will be written out (without `_it0??` in their names). Because in the last iteration the two independent half-reconstructions are joined together, the resolution will typically improve significantly in the last iteration. Because the program will use all data out to Nyquist frequency, this iteration also requires more memory and CPU.

Note that the automated increase in angular sampling is an important aspect of the auto-refine procedure. It is based on signal-to-noise considerations that are explained in [8], to estimate the accuracy of the angular and translational assignments. The program will not use finer angular and translational sampling rates than it deems necessary (because it would not improve the results). The estimated accuracies and employed sampling rates, together with current resolution estimates are all stored in the `_optimiser.star` and `_model.star` files, but may also be extracted from the stdout file. If your queueing system stores that in a file called `Refine3D/run2.out`, you can use the following command to extract this information:

```
grep Auto Refine3D/run2.out
```

Can you now use the `relion_star_printtable` command to make a plot of the FSC curve for your paper?

6 Postprocessing

After performing a 3D auto-refinement, the map needs to be sharpened. Also, the gold-standard FSC curves inside the auto-refine procedures only use unmasked maps to strictly prevent overfitting. However, this also means that the actual resolution is under-estimated somewhat because a noisy solvent area will lower the FSC curve. RELION's procedure for B-factor sharpening and calculating masked FSC curves [2] is accessible through the `Post-processing` job-type.

6.1 Running the job

On the `I/O` tab, set:

- `One of the 2 unfiltered half-maps: Refine3D/run2_half1_class001_unfil.mrc`
- `Output rootname: post_run2`

On the `Mask` tab, set:

- `Perform auto-masking?: Yes`. This will automatically create a soft mask around your protein density. Alternatively, provide your own mask. In that case, make sure it has a generaous soft edge and is not too tight.
- `Initial binarisation threshold: 0.02`. This should be a threshold at which rendering in for example Chimera shows ANSOLUTELY NO noisy spots outside the protein area. Move the threshold up and down to find a suitable spot. Often these around 0.01-0.04.
- `Extend binary map this many pixels: 2`. The threshold above is used to generate a black-and-white mask. The white volume in this map will be "grown" this many pixels in all directions. Use this to make your initial binary mask less tight.
- `Add a soft-edge of this many pixels: 3`. This will put a cosine-shaped soft edge on your mask. This is important, as the correction procedure that measures the effect of the mask on the FSC curve may be quite sensitive to too sharp masks. As the mask generation is relatively quick, we often play with the mask parameters to get the best resolution estimate.
- `Provide your own mask?: No`.

On the `Sharpen` tab, set:

- `MTF of the detector (STAR file): mtf_falcon2_300kV.star`. The manufacturer may provide you with a suitable MTF curve for your own detector, which you will need to store in the same format as the `mtf_falcon_300kV_down4.star` file that is stored in your project directory.

- `Estimate B-factor automatically: Yes`. This procedure is based on the [5] paper and will therefore need the final resolution to extend significantly beyond 10 Å. If your own map does not reach that resolution, you may want to use your own "ad-hoc" B-factor instead.
- `Lowest resolution for auto-B fit (A): 10`. This is usually not changed.
- `Use your own B-factor?: No`

On the `Filter` tab, set:

- `Skip FSC-weighting?: No`. This option is sometimes useful to analyse regions of the map in which the resolution extends *beyond* the overall resolution of the map. This is not the case now.

As this job will run very quickly, we typically run it locally instead of submitting it to a queue. If you cannot run this, you may also copy our precalculated results by typing:

```
cp PrecalculatedResults/Refine3D/post_run2* Refine3D/.
```

If you open the output map `Refine3D/post_run2.mrc` in Chimera or in RELION's display program, you will see that it is much easier to see where all the alpha-helices are then in the converged map of the 3D auto-refine procedure. This is a result of the increased resolution estimate due to the masking AND of the sharpening of the map. It is therefore *strongly recommended* to always run the post-processing. In case of strange resolution estimates, one could analyse the FSC curves of the masked, unmasked and randomised-phased reconstructions, which are all explain in [2]. In particular too tight or too sharp masks may cause trouble.

7 Local-resolution estimation

The estimated resolution from the post-processing program is a global estimate. However, a single number cannot describe the variations in resolution that are often observed in reconstructions of macromolecular complexes. Alp Kucukelbir and Hemant Tagare wrote a nifty program to estimate the variation in resolution throughout the map [3]. RELION-1.3 now implements a wrapper to this program. On the `Local-resolution` job-type, on the `I/O` tab set:

- `One of the two unfiltered half-maps: Refine3D/run2_half1_class001_unfil.mrc`
- `P-value: 0.05`
- `Highest resolution (A): 7`
- `Lowest resolution (A): 10`
- `Resolution step size (A): 0.5`
- `User-provided mask (optional): Refine3D/post_run2_automask.mrc`
- `ResMap executable: XXXX`. Or wherever you ResMap executable is. Note you can make your location the default by defining the environment variable RELION_RESMAP_EXECUTABLE in your shell.

If you cannot run ResMap, copy our results by typing:

```
cp PrecalculatedResults/Refine3D/*resmap* Refine3D/.
```

The resulting `Refine3D/run2_half1_class001_unfil_resmap.mrc` may be used in UCSF Chimera to color the `Refine3D/post_run2.mrc` map according to local resolution (using `Tools` - `Volume data` - `Surface color`, and then select `by volume data value` and browse to the `Refine3D/run2_half1_class001_unfil_resmap.mrc` file. In this particular case, because the data were downsampled, the local-resolution variation isn't very exciting to look at, but it's good to know this anyway!

8 Movie-processing

8.1 Getting organised

In the `Micrographs` directory, make sure that your movies are called with the same name as the averaged micrograph from which you picked your particles before; MINUS its `.mrc` extension; PLUS an underscore; PLUS a common `movie rootname` ("movie" in this case) PLUS a `.mrcs` extension. For example, the movie for `mic001.mrc` should be called `mic001_movie.mrcs`. You can check that this is indeed the case for our test data by typing:

```
ls Micrographs/*mrc*
```

Also, it is assumed later on in the particle normalisation that each of the original micrographs was the average of its corresponding movie. If this is somehow not the case for your own data, then you can use the program `relion_image_handler` to calculate average micrographs from any given number of movie frames, before starting the entire image processing procedure.

8.2 Extracting the movie-particles

On the `Particle extraction` job-type change nothing with respect to the job where you extracted the autopicked particles from all micrographs, except for the `movies` tab, where you set:

- `Extract from movies?: Yes`
- `Rootname of movies files: movie`
- `First movie frame to extract: 1`
- `Last movie frame to extract: 16`

Running the job will again not take very long (although longer than before). The output particles will again be stored in MRC stacks in the `Particles/Micrographs` directory and a new STAR file with all movie-particle frames will be saved in your Project directory as `particles_autopick_movie.star`.

8.3 Refinement with the movie-particles

If you have somehow changed the `3D auto-refine` job-type GUI after you ran `Refine3D/run` you can reload the original settings of that run using the `File -> Load settings` options and select `Refine3D/run2.gui_auto3d.settings`. Then, click the `Start new run` button at the top of the GUI so it display `Continue old run`. On the `I/O` tab, for the option `Continue from here` select the `_optimiser.star` file from the last iteration of the `Refine3D/run2`. Change nothing else on any of the tabs, except for the `Movies` tab, where you set:

- `Realign movie frames?: Yes`
- `Input movie frames STAR file: particles_autopick_movie.star`
- `Running average window: 7`
- `Stddev on the translations (pix): 1`
- `Also include rotational searches?: No`. Including the rotations becomes computationally much more expensive. And the particle polishing procedure below will disregard any beam-induced rotations anyway. Only for very large objects may including the beam-induced rotations (and skipping the polishing step) be useful. Still, even for ribosomes we now prefer the polishing procedure outlined below.

This job will again need to be run on the cluster. It will take at least as much memory as the last iteration of the normal refinement, and probably more CPU time. If you do not want to wait for the results, copy our precalculated ones by typing:

```
cp PrecalculatedResults/Refine3D/run2_ct21* Refine3D/
```

The only useful output file will be a new movie-frame particle STAR file called `Refine3D/run2_ct21_data.star`.

8.4 Particle polishing

The particle polishing algorithm takes the (possibly very noisy) movement tracks from the `Refine3D/run2_ct21_data.star` and fits straight lines through it. In order to further reduce the noise in these tracks, it also exploits the observation that neighbouring particles on a micrograph often move in similar directions. It does so by evaluating the tracks of multiple neighbouring particle together, where the contribution of each particle to it's neighbours tracks is determined by a Gaussian function of the inter-particle distance. The standard-deviation of this Gaussian (in pixels) is determined by the user. In a second step, the polishing algorithm estimates a B-factor and a linear intensity factor to describe the resolution-dependent power of the signal in each individual movie frame. This way, resolution-dependent radiation damage can be modelled by performing a weighted average of all aligned movie frames for each particle. The resulting particles are called "polished" or "shniy", as they have increased signal-to-noise ratios.

On the `Particle polishing` job-type, on the `I/O` tab set:

- `Input STAR file with aligned movies: Refine3D/run2_ct21_data.star`
- `Output rootname: shiny`

On the `Movement` tab set:

- `Linear fit particle movements?: Yes`

- `Running average window: 7`. Just set the same value as you used in the movie-refinement above!
- `Stddev on particle distance (pix): 300`. (The larger the particle, the less noisy the movement tracks from the movie-refinement become and therefore the smaller value can be used here. For ribosomes we often use values like 100 pixels. Smaller values allow to describe ore complicated movement patterns.

On the `Damage` tab set:

- `Perform B-factor weighting?: Yes`
- `Highres-limit per-frame maps (A): 6`. This option is merely to save computational resources. As individual-frame reconstructions tyopically do not extend out to Nyquist, you can give this value here to limit the frequencies taken into account when performing the single-frame reconstructions.
- `Lowres-limit B-factor estimation (A): 20`. The algorithm will estimate relative Guinier-plots between individual-frame reconstructions and a reconstruction from all movie-frames averaged together. These plots may be used to fit a straight line, and we have seen that often lower frequencies may be included in these plots than in the original Guinier-plots as proposed by [5].
- `Mask for the reconstruction: Refine3D/post_run2_automask.mrc`. Use the mask that was determined automatically during the post-processing of the reconstruction before the movie-processing.
- `Symmetry: D2`

The maximum useful number of MPI processors for this job is the number of movie frames in the data times two. If your particle boxes are relatively large, then the memory requirements may be significant, although probably not larger than during the last iteration of the 3D auto-refine runs.

You can copy our precalculated results again by typing:

```
cp PrecalculatedResults/Refine3D/*shiny* Refine3D/.
```

The fitted linear movement tracks are in a file called `Refine3D/run2_ct21_data_shiny.star`. It is often insightful to look at them and compare them to the original tracks from the movie-refinement in the previous subsection. To help you with that, this script on the RELION wiki, which is also stored in your Project directory may be executed as follows:

```
cd Refine3D
../AnalyseMovement.csh run2_ct21_data.star
../AnalyseMovement.csh run2_ct21_data_shiny.star
cd ..
```

This will create two new directories with a text file with the movement tracks (exaggerated 50 times) for each micrograph. We use the `Grace` plotting program to visualise them by typing:

```
xmgrace Refine3D/offsets_run2_ct21_data/Falcon_2012_06_12-16_55_40_0_movie.mrcs.offsets
xmgrace Refine3D/offsets_run2_ct21_data_shiny/Falcon_2012_06_12-16_55_40_0_movie.mrcs.offset
```

The original movement tracks are very noisy, but the fitted straight lines may have extracted the underlying movement pattern.

The result of the B-factor estimation is stored in a file called `Refine3D/run2_ct21_data_shiny_bfactors.star`. Have a look at the estimated B-factors and the linear intensity scales by typing:

```
relion_star_printtable Refine3D/run2_ct21_data_shiny_bfactors.star data_perframe_bfactors \
  _rlnMovieFrameNumber _rlnBfactorUsedForSharpening > bfac.dat
relion_star_printtable Refine3D/run2_ct21_data_shiny_bfactors.star data_perframe_bfactors \
  _rlnMovieFrameNumber _rlnFittedInterceptGuinierPlot > intc.dat
xmgrace bfac.dat
xmgrace intc.dat
```

We often observe relatively high B-factors for the first 1-2 frames where the sample moves a lot, and then also in the later frames where radiation damage sets in. These movies were recorded with 1 electron per squared Angstrom per frame, so the total dose of 16 electrons per squared Angstrom isn't very high, and radiation damage only plays a minor role. Often the intercept of the fitted line with the Y-axis of the Guinier-plot (i.e. the linear intensity scale factor) decreases with dose, although also here the first few frames may be relatively bad.

If somehow the estimated B-factors or scale factors look weird, then one could examine the linear fits through the Guinier plots, which are stored in files called `Refine3D/run2_ct21_data_shiny_frame0??_guinier.star`. Plotting these may sometimes indicate that the frequency range over which a linear line will be fitted must be adapted. Also, if single-frame reconstructions do not extend beyond the 20 Angstrom frequency as defined above, then no line can be fitted and the program will not work (it will give NaN for the estimated B-factors). In such cases, one may choose to skip the B-factor weighting, or to average multiple frames together to estimate the B-factors. The latter may be achieved by specifying `--bfactor_running_avg 3` (always use odd numbers!) on the additional arguments line on the `Running` tab of the particle polishing job-type.

It is probably not a bad idea to visualise the STAR file with the output polished particle (`shiny.star`) in the display program to make sure everything has gone all right.

One final word on the particle polishing job-type. The `Continue old run` button on top has a special function. When you click it, nothing on the GUI will change (e.g. no options are (de-)activated. However, when the program is run with the `Continue old run` button, then the program will look for the

following files, and if they are there it will skip that step of the calculation and move to the next step:

- `Refine3D/run2_ct21_data_shiny_frame0??_half?_class001_unfil.mrc`
- `Refine3D/run2_ct21_data_shiny_bfactors.star`
- `shiny.star`

So, if any of the single-frame reconstrctions was already calculated before, and the `_bfactors.star` file isn't, then the program will jump straight to the B-factor estimation from those. This is useful is for example you would like to change the lower-resolution limit of the frequencies used in the linear fit of the relative Guinier plot. Likewise, if the single-frame reconstructions and the `_bfactors.star` files are already on disc, but the `shiny.star` file isn't, then the program will read in the `_bfactors.star` file and will just perform the weighted averaging of the particles with those values. This is useful if you would manually like to modify the B-factors and intercept, e.g. to remove noise from the `bfac.dat` and `intc.dat` plots mentioned above.

8.5 Re-refine the polished particles

The polished particles in `shiny.star` have increased signal-to-noise ratios compared to the original ones. This may be useful in further classifications. At the very least we ALWAYS use the polished particles to perform a final 3D auto-refine run, as the reconstruction from the polishing itself only positions each particle in its most-likely orientation instead of performing the ML-specific probability-weighted angular assignment that yields higher resolution maps. Therefore, repeat the 3D auto-refine run with the `shiny.star` as input. After this refinement, don't forget to perform the post-processing again in order to calculate the effects of the masks on the FSC curve and to sharpen the map. Additionally, you may also re-run the local-resolution estimation in ResMap. You can copy our precalculated results by typing:

```
cp -r PrecalculatedResults/Refine3Dshiny/ .
```

In this case, the reconstruction already went to the Nyquist frequency of the (down-sampled) data and the differences between before and after movie-processing are negligible. However, hopefully this demonstration was still useful to learn how to do the movie-processing.

You can have a look at your final map before and after movie-processing, together with an X-ray structure by typing:

```
chimera 3i3e_ali.pdb  Refine3D/post_run2.mrc Refine3Dshiny/post_run1.mrc
```

An alternative strategy to the one we followed above, might have been to use all the particles that were selected after 2D classification (i.e. the ones in `particles_autopick_sort_class2d.star` for a 3D auto-refine (so skip 3D

classification at this point); run the movie-frame alignment and the subsequent polishing on all those particles; and only then perform the 3D classification with the polished particles and perform a final 3D auto-refine on the particles in the best classes. This reflects a general change between RELION-1.2 and RELION-1.3: before particle polishing was available (i.e. in the 1.2 release) all classifications were performed on the original, averaged particles; and only after a homogeneous subset was obtained was a final (computationally expensive) movie-frame refinement performed. Now, with the particle polishing algorithm in RELION-1.3, it may be better to include more particles in the (cmoputationally cheaper) movie-processing and then look for structurally homogeneous subsets exploiting the increased SNR in the polished particles.

9 More advanced topics

The following topics are not a part of this tutorial, but describe other tricks we apply to deal with problematic data sets, or to get to even higher resolutions.

9.1 Classifications with finer angular samplings

The 3D classification approach described in section 4 becomes very expensive for fine angular samplings. However, some small conformational differences cannot be classified using coarse samplings. To still be able to classify such small changes, we have successfully used the following procedure:

We first perform a 3D auto-refine with all particles. The resulting reconstruction will be a consensus between the different conformations. However, if the conformational changes are relatively small, then probably the optimal orientations for each particle are not far off their correct values. Therefore, we may use the `_data.star` file upon convergence as input for a 3D classification run. This classification run is then started from an intermediate-resolution low-pass filter, e.g. 40 Å, on the `Optimisation` tab, and on the `Sampling` tab, we select a relatively fine angular sampling, e.g. 1.8 or 0.9 degrees. However, the crucial difference with the 3D classification procedure described in section 4 is that we will use `Perform local angular searches? Yes`, and select a relatively small search range, e.g. 5 degrees. In addition, as the optimal center has already been established for each particle, we can use a smaller `Offset search range`, and possibly a finer `Offset search step`. (The latter may depend on the estimated accuracy as observed for the offsets in the 3D auto-refine run.). The limited orientational search ranges will make this classification much faster, while the fine sampling rates may allow classification of smaller conformational differences.

9.2 Refining very difficult cases

The 3D auto-refine has been developed as a one-click, fully automated procedure for typical cryo-EM data sets. It works well for homogeneous complexes of sizes say larger than 0.5 MDa. There are cases however, where the auto-refine option does not work very well. If the complex is much smaller, the particles become ever harder to align. Also, in some cases the particles are very long and thin, so that one needs a large box to fit the reconstruction in. Because the gold-standard FSC is calculated on unmasked maps, there is then a relatively large amount of noise inside the box, which may lead to severe under-estimation of resolution.

Another problem we have observed is that for very noisy data, or particles that are otherwise very difficult to align, extremely broad probability distributions for the angular assignments lead to (to some extent) spherically-averaged reconstructions. In such cases, we have observed that the gold-standard FSC can still over-estimate the true resolution, which has a detrimental effect on the convergence.

In these difficult cases, one might get better refinement results with the 3D classification run type. The classification approach uses the resolution estimates as explained in [7], i.e. not the gold-standard FSCs as described in [8]. Although this places the responsibility of keeping overfitting at bay again in the hands of the user, it does provide more flexibility in squeezing the most out of difficult data. To that purpose, just use a single class in the 3D classification run type and continue an initial run with relatively coarse angular samplings by multiple runs with increasingly finer samplings (using the `continue old run` on the GUI.

As the 3D classification procedure does not have a convergence criterion you will have to monitor convergence yourself, and stop when the map no longer improves. Again, be careful with using too high regularisation parameters T, as high-resolution overfitted noise will impede the correct interpretation of your map.

9.3 Focussed refinements

In some cases there is a large degree of (continuous) conformational heterogeneity that is hard to separate by dividing the data into a discrete number of subsets. One example of this is ratchet-like movement of the small ribosomal subunit with respect to the large one. In such cases, we have found it useful to mask out either the large or the small subunit at every iteration. Thereby, the reference projections tahat are created internally will only contain density for one of the subunits, while the experimental particles of course have density for both. This additional density in the particles is just considered as "noise" and as long as the proportion of this density isn't too large, refinement may still converge well. For example, masking away the small subunit works very well and leads to greatly improved density of the large subunit. Masking the large subunit away is a bit more difficult (because it corresponds to more than half of the density in the particles), but still works quite well to improve the density in the small subunit. For these focussed refinements, it is important to make the masks with a generously soft edge, as very hard edges may result in artefacts. In any case of masking during the refinement, one should actually be wary of artefactual densities accumulating near the edge of the mask. Examples of successful application of focussed refinement are the large subunit of the yeast mitochondrial ribosome [1], and both subunits of the cytoplasmic ribosome of the *Plasmodium falciparum* parasite [15] and the mammalian one in complex with the translocon Sec61 [14].

10 Wrapping up

That's it! Hopefully you enjoyed this tutorial and found it useful. If you have any questions about RELION, please first check the FAQ on the RELION Wiki and the CCPEM mailing list. If that doesn't help, use the CCPEM list for asking your question. If RELION turns out to be useful in your research, please do cite our papers and tell your colleagues about it.

References

[1] Alexey Amunts, Alan Brown, Xiao-chen Bai, Jose L. Llcer, Tanweer Hussain, Paul Emsley, Fei Long, Garib Murshudov, Sjors H. W. Scheres, and V. Ramakrishnan. Structure of the yeast mitochondrial large ribosomal subunit. *Science*, 343(6178):1485–1489, March 2014.

[2] Shaoxia Chen, Greg McMullan, Abdul R. Faruqi, Garib N. Murshudov, Judith M. Short, Sjors H. W. Scheres, and Richard Henderson. High-resolution noise substitution to measure overfitting and validate resolution in 3D structure determination by single particle electron cryomicroscopy. *Ultramicroscopy*, 135:24–35, December 2013.

[3] Alp Kucukelbir, Fred J Sigworth, and Hemant D Tagare. Quantifying the local resolution of cryo-EM density maps. *Nature methods*, 11(1):63–65, January 2014.

[4] Joseph A. Mindell and Nikolaus Grigorieff. Accurate determination of local defocus and specimen tilt in electron microscopy. *Journal of Structural Biology*, 142(3):334–347, June 2003.

[5] Peter B Rosenthal and Richard Henderson. Optimal determination of particle orientation, absolute hand, and contrast loss in single-particle electron cryomicroscopy. *Journal of Molecular Biology*, 333(4):721–745, October 2003.

[6] Sjors H W Scheres. Classification of structural heterogeneity by maximum-likelihood methods. In *Cryo-EM, Part B: 3-D Reconstruction*, volume 482 of *Methods in Enzymology*, pages 295–320. Academic Press, 2010.

[7] Sjors H W Scheres. A bayesian view on cryo-EM structure determination. *Journal of Molecular Biology*, 415(2):406–418, January 2012.

[8] Sjors H W Scheres. RELION: implementation of a bayesian approach to cryo-EM structure determination. *Journal of Structural Biology*, 180(3):519–530, December 2012.

[9] Sjors H W Scheres and Shaoxia Chen. Prevention of overfitting in cryo-EM structure determination. *Nature Methods*, in press, 2012.

[10] Sjors H W Scheres, R. Nunez-Ramirez, C. O. S Sorzano, J. M Carazo, and R. Marabini. Image processing for electron microscopy single-particle analysis using XMIPP. *Nature Protocols*, 3(6):977–90, 2008.

[11] J M Smith. Ximdisp–a visualization tool to aid structure determination from electron microscope images. *Journal of structural biology*, 125(2-3):223–228, May 1999.

[12] Guang Tang, Liwei Peng, Philip R Baldwin, Deepinder S Mann, Wen Jiang, Ian Rees, and Steven J Ludtke. EMAN2: an extensible image processing suite for electron microscopy. *Journal of Structural Biology*, 157(1):38–46, January 2007.

[13] Kutti R. Vinothkumar, Greg McMullan, and Richard Henderson. Molecular mechanism of antibody-mediated activation of -galactosidase. *Structure*, 22(4):621–627, April 2014.

[14] Rebecca M. Voorhees, Israel S. Fernndez, Sjors H. W. Scheres, and Ramanujan S. Hegde. Structure of the mammalian ribosome-sec61 complex to 3.4 resolution. *Cell*, 0(0), 2014.

[15] Wilson Wong, Xiao-chen Bai, Alan Brown, Israel S. Fernandez, Eric Hanssen, Melanie Condron, Yan Hong Tan, Jake Baum, and Sjors HW Scheres. Cryo-EM structure of the plasmodium falciparum 80S ribosome bound to the anti-protozoan drug emetine. *eLife*, June 2014.