emClarity v1.5.3.10 tutorial

June 4, 2021

Contents

List of Figures

List of Tables

1 How to use this guide

1.1 Run the jobs

Our main objective in writing this tutorial is to help you get started using emClarity for processing sub-tomogram data as quickly as possible. To begin, we will not introduce all of the methods that emClarity puts at your disposal, instead focusing on core features and concepts. If at any point you are confused, or something seems to not work as you expect, you might find more information on the wiki; feel free to also search the mailing list archive, or post new questions to the community forum, hosted on google groups, should you have any questions you cannot resolve on your own.

Tip 1: *To display every procedure available, run* `emClarity help` *from the command line.*

1.2 Algorithms

The emClarity source code is available on github, and we encourage you to go through the code to look at the algorithms directly. Because emClarity is frequently being updated, this can be a great way to see what's going on behind the scenes.

Section 16 contains descriptions of the algorithms for each section presented in this tutorial. Please keep in mind that these are simplified descriptions of what emClarity is actually doing, as we often don't mention the details that were implemented to make the code more efficient.

1.3 Installation and Requirements

Information about the installation and the software and hardware requirements is available online.

1.4 Parameter files

emClarity is currently using a parameter file to manage inputs. You can find an example here.

1.5 System parameters

Table 1: GPU and CPU parameters. Your parameter files should have the following parameters. * indicates the required parameters.

– `nGPUs`*	The number of visible GPUs. emClarity will try to use them in parallel as much as possible. If this number doesn't correspond to the actual number of GPUs available, emClarity will ask you to either adjust this number to match the number of GPUs, or modify the environment variable `CUDA_VISIBLE_DEVICE` to make some GPUs invisible to MATLAB.

– `nCpuCores`*	The maximum number of processes to run simultaneously. In most emClarity programs, the number of processes launched in parallel on a single GPU is equal to `nCpuCores/nGPUs`. If your devices run out of memory, it is likely that you will have to decrease the number of processes per device, thus decreasing this parameter.
- `fastScratchDisk`*	Path of the optional temporary cache directory, used by `ctf 3d` and `tomoCPR`. This directory is only temporary and is moved back inside the project directory at the end of the execution. We recommend setting this to the fastest storage you have available. If left empty, `ctf 3d` and `tomoCPR` will use directly the project cache directory (c.f. `<projectDir>/cache` from section 2).

2 The project directory

emClarity should be run from the "project directory", referred to as `<projectDir>`. Every output will be saved in this directory, ignoring the temporary cache set by the `fastScratchDisk` parameter (table 1). As we go along, we will present in more detail each sub-directory and their content.

- `<projectDir>`:
  - **description**: it contains every input file and input directories emClarity needs and every outputs. As most of the emClarity programs are project based, you should run emClarity from this directory.
  - **name**: anything you like.
  - **created by**: you.
- `<projectDir>/fixedStacks`:
  - **description**: it contains the raw (*not* aligned) tilt-series (`*.fixed`) and the initial tilt-series alignment files (`*.xf`, `*.tlt` and optionally `*.local` and `*.erase`). See section 5.3 for more details.
  - **name**: fixed.
  - **created by**: you or emClarity `autoAlign` (5.2).
- `<projectDir>/fixedStacks/ctf`:
  - **description**: created by `ctf estimate` (section 6) and updated after tilt-series refinements by `ctf update` (section 13). It contains the radial averages (`*_psRadial1.pdf`) and stretched power spectrum (`*_PS2.mrc`) computed by `ctf estimate`, as well as the tilt-series metadata (`*_ctf.tlt`), used throughout the entire workflow and containing in particular the dose-scheme and defocus estimate of each view.
  - **name**: fixed.
  - **created by**: emClarity `ctf estimate`.
- `<projectDir>/aliStacks`:
  - **description**: created by `ctf estimate` (section 6) and updated after tilt-series refinement by `ctf update` (section 13). It contains the aligned, bead-erased tilt-series. These stacks are mostly used by `ctf 3d` to compute the tomograms at different binning.
  - **name**: fixed.
  - **created by**: emClarity `ctf estimate`.
- `<projectDir>/cache`:
  - **description**: created and updated by emClarity when needed, usually during `ctf 3d`. Store any stack or reconstruction for the current binning. If a reconstruction (`*.rec`) is present at the current binning, `ctf 3d` will skip its reconstruction. This also means that if a reconstruction is aborted, you may need to manually delete a partially complete reconstruction. For older versions of emClarity ($<$ 1.5.1 versions), if the sampling functions are already calculated (`*.wgt`), emClarity will re-use them.

  - **name**: fixed.
  - **created by**: emClarity.

- `<projectDir>/convmap`:
  - **description**: When creating a project with `init` (section 9), emClarity will look in this directory to grab outputs from `templateSearch` (section 8). If you pick your particles with emClarity, the content of this directory is generated by `templateSearch`.
  - **name**: fixed.
  - **created by**: you / emClarity.

- `<projectDir>/recon`:
  - **description**: It holds the information for each reconstructed sub-region in a given tilt-series (section 7). The `*_recon.coords` files are read into the metadata created by `init` and is used whenever a tomogram is made or whenever the coordinates of a sub-region is needed. Any change to these files is ignored after `init`.
  - **name**: fixed.
  - **created by**: `recScript2.sh`

- `<projectDir>/<binX>`:
  - **description**: emClarity does not directly use this directory, but it is used by `recScript2.sh` to define the sub-regions boundaries (section 7) and create `<projectDir>/recon`.
  - **name**: whatever is defined in `recScript2.sh`. By default, `bin10`.
  - **created by**: `recScript2.sh`

- `<projectDir>/FSC`:
  - **description**: created and updated during subtomogram averaging (section 11). It contains the spherical and conical FSCs for each cycle (`*fsc_GLD.txt` and `*fsc_GLD.pdf`), as well as the Figure-Of-Merit used for filtering (`*cRef_GLD.pdf`) and the CTF-corrected volume used for FSC calculations (`*noFilt_EVE.mrc` and `*noFilt_ODD.mrc`). For a full description of the FSC related outputs, see section 16.8. If you run `emClarity fsc`, the molecular masks are saved in this directory as well.
  - **name**: fixed.
  - **created by**: emClarity `avg`.

- `<projectDir>/alignResume`:
  - **description**: Contains the results of the subtomogram alignments, for each cycle. emClarity will look at this directory before aligning the particles from a given sub-region. If the results for this sub-region, at the current cycle, are already saved, it will skip the alignment. See section 12.3 for more details on why it is useful.
  - **name**: fixed.
  - **created by**: emClarity `alignRaw`

3 Get your data ready

In this tutorial, we will use the 12 tilt-series of the fast-incremental single-exposure (FISE) data deposited on EMPIAR-10304 from [1]. You should be able to get a ∼7Å map from this tutorial. To download the motion-corrected tilt-series, EMPIAR recommends to use the 'Aspera Connect' option but there are options as well.

– **Sample**:	70S ribosomes, 10nm colloidal gold fiducials.
– **Tilt-scheme**:	Dose-symmetric starting from 0°, 3°increment, ±60°, 120e/Å^2 total exposure.
– **Instruments**:	Krios with single-tilt axis holder equipped with a Gatan K3 direct electron detector. 2.1Å/pix. ∼175° image rotation.

Table 2: Tutorial data-set

Note 1: *For this tutorial, we don't necessarily recommend to align the 12 tilt-series manually, as it can be quite redundant, but for beginners, we do recommend to at least try aligning one tilt-series with ETomo. In any case, you should be able to use* `emClarity autoAlign` *for this dataset.*

4 Workflow

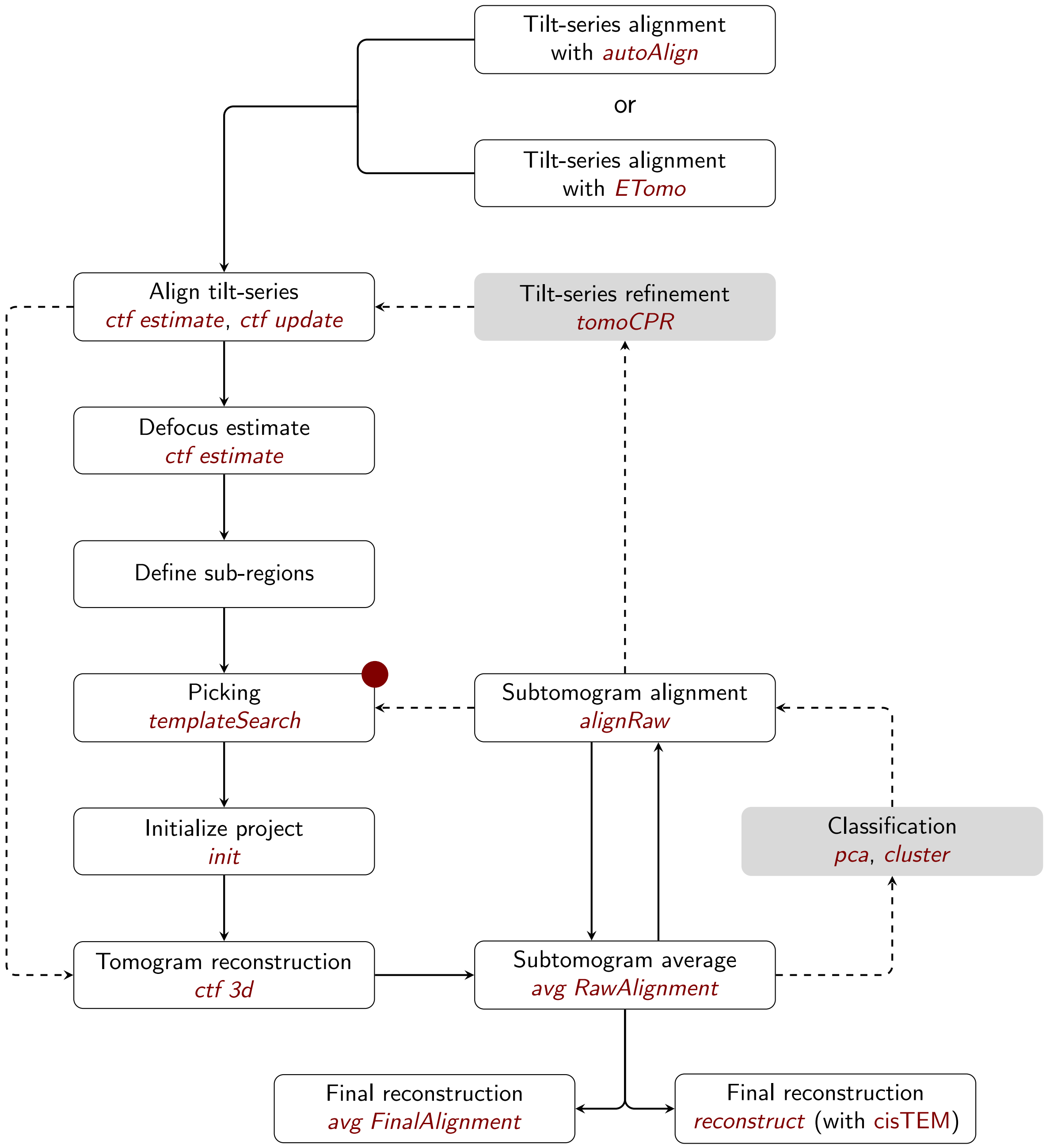
Tilt-series alignment with *autoAlign*
or
Tilt-series alignment with *ETomo*
Align tilt-series *ctf estimate*, *ctf update*
Tilt-series refinement *tomoCPR*
Defocus estimate *ctf estimate*
Define sub-regions
Picking *templateSearch*
Subtomogram alignment *alignRaw*
Initialize project *init*
Classification *pca*, *cluster*
Tomogram reconstruction *ctf 3d*
Subtomogram average *avg RawAlignment*
Final reconstruction *avg FinalAlignment*
Final reconstruction *reconstruct* (with cisTEM)

Figure 1: emClarity workflow. The classification and tilt-series refinement steps are optional. The red marker indicates the possibility to import data from other software (section 8.5 for more details).

You will often find that it is much easier to organize every emClarity calls into one script. This script has two main purposes. First, it keeps track of the jobs that have been run (you can also find this information into the `logFile` directory). This is often useful to visualize the global picture and it might help you to remember how you got your final reconstruction. Second, it is a script, so you can use it directly to run emClarity, making the workflow much simpler.

In the case of this tutorial, here is one example of such script. To make things easier, each jobs has its own parameter file. This is just an example and you can modify the number of cycles and which sampling to use, the angular searches, the number of tilt-series refinement and the number of classification, etc.

```bash
#!/bin/bash

# Simple function to stop on *most* errors
check_error() {
    sleep 2
    if tail -n 30 ./logFile/emClarity.logfile |\
       grep -q "Error in emClarity" ; then
        echo "Critical error found. Stopping the script."
        exit
    else
        echo "No error detected. Continue."
    fi
}

# Change binning with tomoCPR
run_transition_tomoCPR() {
    emClarity removeDuplicates param${i}.m ${i}; check_error
    emClarity tomoCPR param${i}.m ${i}; check_error
    emClarity ctf update param$((${i}+1)).m; check_error
    emClarity ctf 3d param$((${i}+1)).m; check_error
}

# Basic alignment cycle
run_avg_and_align() {
    emClarity avg param${i}.m ${i} RawAlignment; check_error
    emClarity alignRaw param${i}.m ${i}; check_error
}

# autoAlign
# ctf estimate
# templateSearch

# Create metadata and reconstruct the tomograms
emClarity init param0.m; check_error
emClarity ctf 3d param0.m; check_error

# First reconstruction - check if that looks OK.
emClarity avg param00.m 0 RawAlignment; check_error
emClarity alignRaw param0.m 0; check_error

# Bin 3
for i in 1 2 3 4; do run_avg_and_align; done

# Run tomoCPR at bin3 using cycle 4 and then switch to bin2
run_transition_tomoCPR

# Bin 2
for i in 5 6 7 8 9; do run_avg_and_align; done

# Run tomoCPR at bin2 using cycle 9 and then switch to bin1
run_transition_tomoCPR

# Bin 1
for i in 10 11 12 13 14; do run_avg_and_align; done
```

```
# Last cycle: merge the datasets
emClarity avg param15.m 15 RawAlignment; check_error
emClarity avg param15.m 15 FinalAlignment; check_error
emClarity reconstruct param15.m 15;
```

This example doesn't have a classification, but as explained in section 14, classifications are encapsulated in their own cycles, so you can run them anytime you want between two cycles.

In our experience, it is usually good practice to keep a close eye on how the half-maps and FSC evolves throughout the workflow, specially before deciding to change the sampling. Moreover, as mentioned in Figure 1, the tilt-series refinement in completely optional and you can simply change the binning by running `ctf 3d`, as opposed to `run_transition_tomoCPR`.

The following sections of the tutorial will give you more details on the command, on the parameters you need to run them and on the generated outputs.

Finally, if everything is set correctly and once the picking is done, you can technically run this script and wait for emClarity to output the final reconstruction. If you are new with emClarity, we do recommend you to run the commands manually, at least for the first few cycles, in order to get a better grasp of the software.

> ***Tip 2:*** *It is best practice to work the whole way through the workflow with the smallest data-set as possible and once you checked that everything holds, then process your full data. The same approach may be taken with this tutorial; it should be possible to obtain a low-resolution but recognizable 70S ribosome with only two or three of the tilt-series. This will confirm that everything "runs" (producing some output) in your hands and on your gear. Only after this it does make sense to then confirm you are able to produce the correct output, i.e. a 5.9Å map.*

5 Initial tilt-series alignment

5.1 Objectives

The first step of the workflow consists into finding an initial alignment for the raw tilt-series, that is the tilt, rotation and shift for each image within the series. After the alignment, the tilt-axis must be parallel to the y-axis. This alignment can be refined later on using the particles positions (section 13).

5.2 With emClarity

emClarity can align the tilt-series for you using its `autoAlign` procedure. This procedure is based on the IMOD programs tilt and tiltalign and offers an automatic way of aligning tilt-series, with or without gold beads.

5.2.1 Parameters

Table 3: `autoAlign` parameters. Your parameter file should have the following parameters. * indicates the required parameters. The other parameters are optional.

– `PIXEL_SIZE`*	Pixel size in meters per pixel (e.g. 1.8e-10). Should match the header of the tilt-series.
– `beadDiameter`*	Bead diameter in meters (e.g. 10e-9). If 0, the beads are ignored during the alignment, but the `.erase` file is still generated using findbeads3d with `-BeadSize`=10.5nm.
– `autoAli_max_resolution`	Low-pass cutoff, in Å, used in alignment. An additional median filter is applied to the images before alignment. The filtered series are only used for alignment. Default=18
– `autoAli_min_sampling_rate`	Maximum pixel size used for alignment, in Å per pixel. If you have a pixel size of 2, then the alignment will start at bin 5 using the default. Default=10.
– `autoAli_max_sampling_rate`	Minimum pixel size used for alignment, in Å per pixel. If you have a pixel size of 2, then the alignment will end at bin 2 using the default. Default=3.
– `autoAli_iterations_per_bin`	The number of patch tracking iterations, for each bin. Default=3.
– `autoAli_n_iters_no_rotation`	The number of patch tracking iterations, for each bin, before activating local alignments. Default=3.
– `autoAli_patch_size_factor`	Sets the size of the patches used for patch tracking. Making this larger will result in more patches, and more local areas in later iterations, but may also decrease accuracy. Default=4

– `autoAli_patch_tracking_border`	Number of pixels to trim off each edge in X and in Y. Corresponds to `-BordersInXandY` from tiltxcorr. Default=64.
– `autoAli_patch_overlap`	Fractional overlap in X and Y between patches that are tracked by correlation. This influences the number of patches. Corresponds to `-OverlapOfPatchesXandY` from tiltxcorr. Default=0.5.
– `autoAli_max_shift_in_angstroms`	Maximum shifts allowed, in Å, for the patch tracking alignment. Default=40.
– `autoAli_max_shift_factor`	The maximum shifts allowed are progressively reduced with the iterations i, such as [1]. Default=1.
– `autoAli_refine_on_beads`	Whether or not the patch tracking alignment should be refined using the gold beads. This refinement makes the alignment significantly slower, but can substantially improve the quality of the alignment. This refinement is automatically turned-off if the number of beads detected is less than 5 and local alignments are only activated if there is more than 10 beads. Default=false.

5.2.2 Run

As with every emClarity programs, you should run the next commands in the project directory. The `autoAlign` routine has the following signature:

```
>> emClarity autoAlign <param> <stack> <rawtlt> <rot>
```

where `<param>` is the name of the parameter file, `<stack>` is the tilt-series to align (e.g. `tilt1.st`), `<rawtlt>` is a text file containing the raw tilt-angles (e.g. `tilt11.rawtlt`), in degrees (one line per image, in the same order as in the `.st` file). `<rot>` is the image rotation (tilt-axis angle from the vertical), in degrees, as specified in ETomo. This angle will vary based on the microscope and magnification, but for this tutorial it is about 175°.

For example, to run `autoAlign` on the first tilt-series of the tutorial:

```
>> emClarity autoAlign param.m tilt1.st tilt1.rawtlt 175
```

You may have noticed that all of the tilt-series have their first image blank. Fortunately, `autoAlign` can remove images from the series before alignment. For instance, to remove the first view of the first tilt-series:

```
>> emClarity autoAlign param.m tilt1.st tilt1.rawtlt 175 1
```

and to remove the first and last view of tilt11, run:

```
>> emClarity autoAlign param.m tilt11.st tilt11.rawtlt 175 [1,41]
```

> ***Note 2:*** *For the tutorial dataset, you should remove the first view of tilt1 to tilt10 and the first and last views from tilt11 and tilt12. Also, the refinement using fiducial beads doesn't produce satisfying results for tilt5, tilt6 and tilt8. As such, the* `autoAli_refine_on_beads` *was turned off for these tilt-series.*

[1] $\texttt{int}(\texttt{autoAli\_max\_shift\_in\_angstroms}/i^{\texttt{autoAli\_max\_shift\_factor}}) + 1$

5.2.3 Outputs

emClarity creates and organizes the necessary files it needs to run the next step of the workflow. You can find a description of these files in section 5.3. The goal here is to check whether or not the alignment is good enough to start with and the easiest way is to look at `fixedStacks/<prefix>_3dfind.ali` or `fixedStacks/<prefix>_binX.ali`. If you are familiar with ETomo, then you can of course also look at the log files saved in `emC_autoAlign_<prefix>`. For instance, to visually check the fiducial beads:

```
>> 3dmod \
emC_autoAlign_<prefix>/<prefix>_X_3dfind.ali \
emC_autoAlign_<prefix>/<prefix>_X_fitbyResid_X.fid
```

If for some reason you want to reconstruct the aligned tilt-series now, you can run the following command:

```
>> newstack -xform <prefix>.xf <prefix>.fixed <prefix>.ali
```

5.3 With ETomo

If you don't want to use `autoAlign`, we do recommend using the (fiducial) alignment procedure from the ETomo pipeline (autofidseed, tiltalign, etc.). One powerful option of this pipeline is to be able to solve for a series of local alignments using subsets of fiducial points, which can then be used by emClarity, via the IMOD tilt program, to reconstruct the tomograms.

During the alignments, you should keep an eye at the ratio of known measurements (fiducials) and unknown parameters (shifts, rotations, tilts and magnifications) you try to solve for. If this ratio is too low, it means the program is likely to accommodate random errors to solve for the model you are asking for. In face of such low ratio, you can simplify the alignment model during the refinement by grouping and/or fixing the variables (see restrictalign "OrderOfRestrictions" for more details).

You are free to use another software for the initial alignment, but it is likely to require more efforts to integrate your data into emClarity (see the next 5.3 section). We do not plan to support imports from tomography software other than IMOD in the near future.

> ***Note 3:*** *The **IMOD batchruntomo** interface encapsulates the **ETomo** pipeline and runs the operations required to align the tilt-series. To deal with low ratio, **batchruntomo** is applying successive restrictions to the alignment. The default behavior is to group the rotations, then group the magnifications, then fix the tilt angles, then solve for only one rotation and then fix the magnification.*

> ***Tip 3:*** *You might have noticed that some images from the tutorial data-set are (mostly) blank. These images can be ignored during alignment (**ETomo**: "view to skip" option). We don't recommend to remove these images from the stack as it is possible to remove them with **emClarity** while accounting for the cumulative electron dose (section **6**).*

Once the tilt-series are aligned, you have to transfer the alignment files to emClarity. Please pay careful attention to the naming conventions in this section, as there are used throughout the pipeline.

For each tilt-series, emClarity needs:

- `<prefix>.fixed`: the raw (*not* aligned) tilt-series. These should *not* be exposure-filtered nor phase flipped. They correspond to the original `tilt1.mrc` to `tilt12.mrc` stacks available in EMPIAR. If you did any other preprocessing (X-ray removal, etc.), you should use these.

  Note 4: *While it will produce substandard results, if you have no option but to use images which are already exposure filtered, please add the following to your parameter file:* `applyExposureFilter=0`

- `<prefix>.xf`: the file with alignment transforms to apply to the `<prefix>.fixed` stacks. This file should contain one line per view, each with a linear transformation specified by six numbers. The first 4 numbers is the 2x2 rotation matrix (in-plane rotation, scaling/magnification) and the last 2 numbers are the X and Y shifts (in un-binned pixels). See section 16.5 for more details.

  If you did the alignment with ETomo, it corresponds to "OutputTransformFile" from tiltalign which is set by default to `<prefix>_fid.xf` (fiducial alignment) or `<prefix>.xf` (fiducialess alignment).

- `<prefix>.tlt`: the file with the solved tilt angles. One line per view, angles in degrees.

  If you did the alignment with ETomo, it corresponds to the "OutputTiltFile" from tiltalign, which is by default `<prefix>_fid.tlt` (fiducial alignment) or `<prefix>.tlt` (fiducialess alignment).

- **(optional)** `<prefix>.local`: the file of local alignments. This file is similar to the `<prefix>.xf` file, but contains one transformation per view and per patch, plus an additional header per patch. See the tiltalign documentation for more details.

  If you did the alignment with ETomo, it corresponds to the "OutputLocalFile" from tiltalign, which is by default `<prefix>local.xf`.

- **(optional)** `<prefix>.erase`: the file with the coordinates (in pixel) of the fiducial beads to erase before ctf estimation. These coordinates must correspond to the aligned stack. One line should contain the x, y and z (view) coordinates of the beads you wish to remove. Alternatively, you can remove the beads from the raw `<prefix>.fixed` before importing them to emClarity.

  If you did the alignment with ETomo, it corresponds to the `<prefix>_erase.fid` file.

These files should be copied to `<projectDir>/fixedStacks` (see section 2).

Tip 4: *You don't necessarily need to copy the tilt-series to the* `fixedStacks` *directory; use soft links:* `ln -s <...>/<prefix>.mrc <...>/fixedStacks/<prefix>.fixed`

6 Defocus estimate

6.1 Objectives

There are two main objectives. First, create the aligned, optionally bead-erased, weighted stacks. Weighted refers to the per-view weighting applied by emClarity to take into account the frequency dependent drop in SNR due to radiation damage, an isotropic drop in SNR due to increased thickness with the tilt-angle causing inelastic scattering losses and optionally also for the cosine dose-scheme, also referred as Saxton scheme. These stacks will be then used to compute the tomograms at later stages. The second objective is to estimate the defocus of each view of the stack (two defoci and the astigmatism angle, per view).

6.2 Parameters

Table 4: `ctf estimate` parameters. Your parameter file should have the following parameters. * indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Microscope settings	
– `VOLTAGE`*	Accelerating voltage of the microscope in Volts (e.g. 300e3).
– `Cs`*	Spherical aberration of the microscope in meters (e.g. 2.7e-6).
– `AMPCONT`*	Percent amplitude contrast ratio (e.g. 0.09).
– `PIXEL_SIZE`*	Pixel size in meters per pixel (e.g. 1.8e-10). Must match the header of the stacks in `fixedStacks/*.fixed`.
– `SuperResolution`*	Whether the stacks are super-sampled. If `1`, emClarity will Fourier crop by a factor of 2 and set the actual pixel size to `2 * PIXEL_SIZE`. Note that this is not tested anymore, so it is preferable to Fourier crop the stacks beforehand and set it to 0.
Fiducials	
– `beadDiameter`	Diameter of the beads to erase, in meters (e.g. 10e-9). This parameter is used if fiducial beads need to be erased, thus only for stacks with a `fixedStacks/*.erase` file.
– `erase_beads_after_ctf`	Whether or not the fiducial beads should be removed on the raw tilt-series (now) or on the CTF multiplied tilt-series computed during the tomogram reconstruction (section 10). Do not change this option between `ctf estimate` and `ctf 3d`. Default=`false`.
Tilt-scheme	
– `CUM_e_DOSE`*	Total exposure in e/Å^2.
– `doseAtMinTilt`*	The exposure each view receive (should be about `CUM_e_DOSE` / nb of views), in e/Å^2.
– `oneOverCosineDose`*	Whether or not it is a Saxton scheme (dose increase as 1/cos(α), α being the tilt angle); this will scale `doseAtMinTilt` according to the tilt angle (e.g. 0).
– `startingAngle`*	Starting angle, in degrees (e.g. 0).

– startingDirection*	Starting direction; should the angles decrease or increase (neg or pos).
– doseSymmetricIncrement*	The number of tilts before each switch in direction. 0=false, 2="normal" dose symmetric. The original dose symmetric scheme included 0 in the first group. For this, specify the number as a negative number.
Defocus estimate	
– defCutOff*	The power spectrum used by ctf estimate is considered from slightly before the first zero past the first zero to this cutoff, in meter (e.g. 7e-10).
– defEstimate*	Initial rough estimate of the defocus, in meter. With defWindow, it defines the search window of defoci.
– defWindow*	Defocus window around defEstimate, in meter; e.g. if defEstimate = 2.5e-6 and defWindow = 1.5e-6, try a range of defocus between 1e-6 to 4e-6.
– deltaZtolerance*	Includes the tiles with defocus equal to that at the tilt-axis $\pm\Delta Z$, in meters. See section 16.5 for more details. Default=50e-9.
– zShift*	Used for the handedness check. Shift the evaluation region ($Z_{tilt-axis}$ ± deltaZtolerance) by this amount. See section 16.5 for more details. Default=150e-9.
– ctfMaxNumberOfTiles*	Limits the number of tiles to include in the power spectrum. The more tiles, the stronger the signal but the longer it takes to compute the power spectrum. Default=4000.
– ctfTileSize*	Size of the (square) tiles, in meters. Default=680e-10.
– paddedSize*	The tiles are padded to this size, in pixel, in real space before computing the Fourier transform. Should be even, large (compared to the tiles), and preferably a power of 2. Default=768.

6.3 Run

The ctf estimate routine has the following signature:

```
>> emClarity ctf estimate <param> <prefix>
```

<param> is the name of the parameter file (e.g. param_ctf.m), and <prefix> is the base-name of the tilt-series in <projectDir>/fixedStacks you wish to process.

For example, to run ctf estimate on the first tilt-series of the tutorial:

```
>> emClarity ctf estimate param_ctf.m tilt1
```

If you have many tilt-series and you don't want to run all of them individually, you can do the following. This will select every available stack to emClarity and run ctf estimate on each one of them.

```
#!/bin/bash
for stack in fixedStacks/*.fixed; do
    prefix=${stack#fixedStacks}
    emClarity ctf estimate param_ctf.m ${prefix%.fixed}
done
```

If you didn't remove the bad images by now, `ctf estimate` can remove images from the stack. For instance, to remove the first view of tilt1, run:

```
>> emClarity ctf estimate param_ctf.m tilt1 1
```

and to remove the first and last view of tilt11, run:

```
>> emClarity ctf estimate param_ctf.m tilt11 [1,41]
```

> ***Note 5:*** *You should remove the first view from the tilt-series tilt1 to tilt10, and the first and last views from tilt11 and tilt12.*

6.4 Outputs

You should make sure the tilt-series "looks aligned" and the average defocus (at the tilt axis) was correctly estimated. The best way to check:

1. Open `aliStacks/<prefix>_ali1.fixed` and go through the views. The views should be aligned to the tilt-axis, which must be parallel to the Y axis (so vertical if you use 3dmod). If an `*.erase` file was available for this tilt-series, the beads should be removed.

2. Open `fixedStacks/ctf/<prefix>_ali1_psRadial_1.pdf` and check that the theoretical CTF estimate (green) matches the radial average of the power spectrum of the tilt-series (black). Note that the amplitude doesn't matter here, the phase on the other hand, does.

3. If they don't match, it is likely that you will need to adjust the `defEstimate` and `defWindow` parameters. Open `fixedStacks/ctf/*_ccFIT.pdf`, which plots the cross-correlation score as a function of defocus. There is often an obvious correct peak, smoothly rising and falling. If you don't see this peak, try to change the sampled defoci with `defEstimate` ±`defWindow` and re-run `ctf estimate`

> ***Note 6:*** *For a full description of the outputs generated by* `ctf estimate`*, you should refer to section* ***16.5****.*

7 Select sub-regions

7.1 Objectives

Quite often, the regions of interests don't take up the entire field of view. To speed things up and save memory, you can select the sub-regions you would like emClarity to focus on. These sub-regions can be changed until the project is initialized (section 9).

You may notice as we go through this tutorial that we often thinks in term of sub-regions and not in terms of the full tomograms. For example, during picking (section 8), each sub-regions is processed independently from the others and if you decide to ignore one sub-regions from a tomogram, you can.

> ***Note 7:*** *When the tilts-series alignment is later refined using tomogram constrained projection refinement (tomoCPR) the full area imaged is reconstructed on the fly, and all sub-tomograms from all sub-regions are inserted into this reconstruction to generate references. This allows a weighted contribution from all particles of all species to be considered, related to what "M" [2] refers to as multi-particle refinement.*

7.2 Create the boundaries for each sub-regions

- Download from the repository the recScript2.sh script and copy it inside the project directory.
- Prior to selecting the sub-regions, we need to create the tomograms containing the entire field of view. Running the following command from the project directory will do exactly that.

  ```
  ./recScript2.sh -1
  ```

  By default, this will create a new directory, called `<projectDir>/bin10`, which will have a bin 10 tomogram for every stack created by `ctf estimate` (i.e. a tomogram for every `aliStacks/*_ali1.fixed`).
- Now that we have the tomograms, we can start thinking about the sub-regions we want to select. The goal is to define 6 points which defines the boundaries of one sub-regions (x_{min}, x_{max}, y_{min}, y_{max}, z_{min} and z_{max}). So if you have 3 sub-regions you are interested in for a particular tomogram, you will need to define $6 \times 3 = 18$ points.

  This is most easily done by creating an IMOD model:

  - Go in `<projectDir>/bin10` and open the first tomogram with 3dmod:

    ```
    3dmod tilt1_bin10.rec
    ```

  - Select the "Model" mode and select the 6 points in the order specified above. Each point must be in its own contour. For the tutorial data-set, as the ribosome are homogeneously spread across the entire tomogram, we recommend to divide the tomograms into 2 sub-regions of equal size. In this case, you'll need to create a sequence of 12 points.
  - Save the model (`File` → `Save model`) with the same name as the tomogram but with the `.mod` extension.
  - Repeat for each tomogram.

  At the end of this step, you should have in `<projectDir>/bin10`, one `*.mod` file per tilt-series you wish to process.

> ***Note 8:*** *Each sub-region is kept on disk, and only individual sub-tomograms are read in from them, reducing the amount of GPU memory needed. emClarity parallelizes jobs on a sub-region level. As such, dividing a tomogram into multiple smaller sub-regions might be more efficient down the line. This is especially true for sub-tomogram alignment (section **12**) and tilt-series refinement (section **13**) since emClarity can distribute jobs across multiple servers.*

- This `bin10` directory is not directly used by emClarity. You'll need to convert the `*.mod` files you just created into something emClarity understands. You can do this on the first stack by running:

  ```
  ./recScript2.sh tilt1
  ```

  Or to convert every sub-regions of every tomograms, run:

  ```
  #!/bin/bash
  for stack in bin10/*.mod; do
      prefix=${stack#bin10} ; ./recScript2.sh ${prefix%.mod}
  done
  ```

  This will create a directory called `<projectDir>/recon` with the file emClarity needs to extract the sub-regions you selected:

Table 5: `recon/<prefix>_recon.coords`

Line	tilt parameter	Description
1		`<prefix>`; stack prefix.
2		Number of sub-regions within this stack.
3	`WIDTH`	Width in X of the first sub-region, in pixel.
4	`SLICE` 1	Starting Y coordinate of the first sub-region. Starts from 0.
5	`SLICE` 2	Ending Y coordinate of the first sub-region. Starts from 0.
6	`THICKNESS`	Thickness in Z of the first sub-region, in pixels.
7	`SHIFT` 1	Shift in X the reconstructed slice, in pixel. If it is positive, the slice will be shifted to the right and the output, the first sub-region, will contain the left part of the whole potentially reconstructable area.
8	`SHIFT` 2	Shift in Z the reconstructed slice, in pixel. If it is positive, the slice is shifted upward.
...	...	Same as line 3 to 8, but for the next sub-regions, if any.

> ***Note 9:*** *If you change the coordinates of the sub-regions, you must do it before running* `init` *(section **9**) and you always need to refresh the* `<projectDir>/recon` *directory by re-running* `./recScript2.sh <prefix>`.

> ***Note 10:*** *In this last step,* `recScript2.sh` *needs Python2 in the PATH.*

8 Picking

8.1 Objectives

It's time to pick the particles, i.e. the subtomograms. There are many ways to pick particles, but they usually all rely on the tomograms. Each particle can be described by its x, y, z coordinates and ϕ, θ, ψ Euler angles. emClarity has a template matching routine that can pick the subtomograms for you, but it requires a template.

8.2 Parameters

Table 6: templateSearch parameters. Your parameter file should have the following parameters. * indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Sampling	
– PIXEL_SIZE*	Pixel size in meter per pixel (e.g. 1.8e-10). Should match the header of the stacks in fixedStacks/*.fixed.
– SuperResolution*	Whether or not the fixedStacks/*.fixed are super-sampled. Not that this should be the same value you used for ctf estimate in section 6.
– Tmp_samplingRate*	Sampling (i.e. binning) at which the sub-region should be reconstructed to perform the template matching (1 means no binning). The sampling rate should be chosen to give a running pixel size between 8 and 12Å/pix.
Tomogram reconstruction	
– erase_beads_after_ctf	Whether or not the fiducial beads should be removed before or after CTF multiplication. Default=0.
– beadDiameter*	Bead diameter in Å.
– applyExposureFilter	Whether or not the exposure filter should be applied. If you turn it off, make sure it is turned-off during subtomogram averaging as well.
– super_sample*	Compute the back projection in a slice larger by the given integer factor (max=8) in each dimension, by interpolating the projection data at smaller intervals ("super-sampling"). This corresponds to the SuperSampleFactor entry from tilt. Default=0.

– `expand_lines`*	If `super_sample` is greater than 0, expand projection lines by Fourier padding (sync interpolation) when super-sampling, which will preserve higher frequencies better but increase memory needed. This corresponds to the `ExpandInputLines` entry from tilt. Default=`0`.
– `whitenPS`*	TODO. Default=`[0,0,0]`
Particle	
– `particleRadius`*	Particle radii, in Å. Format is $[R_X,\ R_Y,\ R_Z]$. In this context, it defines a region around a cross-correlation peak to remove from consideration after a particle is selected. See `Peak_mRadius` for more details.
– `Ali_mRadius`*	Alignment mask radii, in Å. Format is $[R_X,\ R_Y,\ R_Z]$. In this case, it is used to pad/trim the template to this size.
– `Peak_mType`	Type (i.e. shape) of the cross-correlation peaks. Can be sphere, cylinder or rectangle. See section 16.6 for more details. Default=sphere.
– `Peak_mRadius`	Radius of the cross-correlation peaks, in Å. Format is $[R_X,\ R_Y,\ R_Z]$. See section 16.6 for more details. Default= $0.75\ \times$ `particleRadius`.
– `diameter_fraction_for_local_stats`	TODO. Default=`1`.
Template matching	
–`symmetry`*	Symmetry of the partiles, used to restrict the angular search to the strict necessary and helps to reduce wedge bias by randomizing detected angles to one withing the symmetry group. E.g. C1, C2, .., CX, O, I) (see section 16.6.4.
– `Tmp_angleSearch`*	Angular search, in degrees. Format is $[\Theta_{out},\ \Delta_{out},\ \Theta_{in},\ \Delta_{out}]$. For example, $[180,\ 15,\ 180,\ 12]$, specifies a ±180° out of plane search (polar and azimuth angles) with 15° steps and ±180° in plane search (planar angles) with 12° steps. Depending on the `symmetry`, the search will be restrained if the specified range is outside of the unique set of angles.
–`Tmp_threshold`*	Estimate of the number of particles. From this value a score threshold will be calculated that should result in fewer false positives (estimated to 10%) and allows to pick for up to 2x this estimate. If there are significant departures from Gaussian noise (e.g. carbon edge) this may fail.
–`Override_threshold_and_return_N_peaks`	Overrides `Tmp_threshold` and select the specified highest scoring peaks.

-Tmp_targetSize	Size, in pixel, of the chunk to process. If the sub-region is too big, the processing will be split into individual chunks. Format is $[X, Y, Z]$. Default=[512,512,512].
-Tmp_bandpass	A band-pass filter is applied to the template and the tomogram. This parameter defined the band-pass filter. Should be vector of 3 values, aka [1,2,3]. (1) is the filter value at the DC (i.e. zero) frequency, between 0 and 1. (2) is the high-pass cutoff (where the pass is back to 1) in Å. (3) is the low-pass cutoff (where the pass starts to roll off) in Å. (2) should be larger than (3). Default=[1e-3,600,28].
-rescale_mip	TODO. Default=1.

8.3 Run

8.3.1 Preparing the template

Before running templateSearch, you need to prepare a template. This template should have the same pixel size as the PIXEL_SIZE parameter. It doesn't need to be low-pass filter, as emClarity will do it internally. If you want to re-scale a map, you can run:

```
>> emClarity rescale <in> <out> <inPixel> <OutPixel> <method>
```

<in> and <out> are the name of your template and the output name of the re-scaled template, respectively. <inPixel> is the pixel size of your template and <OutPixel> is the desired pixel size. <method> can be "GPU" or "cpu".

***Note 11:** We do provide a template for this tutorial, but any 70S ribosome map should work.*

8.3.2 Generating the tomograms

The tomograms use for the template matching are CTF multiplied as described in section 10 and 16.7. To generate them, simply run:

```
>> emClarity ctf 3d <param> templateSearch
```

This will generate a tomogram for every subregion defined in the recon/*.coords files (table 5).

8.3.3 Template matching

The templateSearch routine has the following signature:

```
>> emClarity templateSearch <param> <prefix> <region> <template>
   <symmetry> <GPU>
```

<param> is the name of the parameter file, <prefix> is the base-name of the tilt-series in <projectDir>/aliStacks to process. <region> is the number of the sub-region to process (see section 7). <symmetry> is not used but kept for backward compatibility. The symmetry parameter specified in the parameter file overrides whatever is entered on the command line. <GPU> is the GPU ID to use (starting from 1).

For example, to run templateSearch on the first tilt-series of the tutorial, where we defined 2 sub-regions:

```
# First region
>> emClarity templateSearch param.m tilt1 1 template.mrc C1 1
# Second region
>> emClarity templateSearch param.m tilt1 2 template.mrc C1 1
```

8.4 Outputs

The primary goal now is to remove false positives due to strong homogeneous features like carbon edges, membranes, or residual gold beads. The template matching produces a "cumulative correlation" map, which can be opened alongside a 3d model of the selected peaks.

In <projectDir>/convmap_wedgeType_2_bin<X>, with <X> equal to Tmp_sampling:

- To look at the 3d IMOD model containing the coordinates of the selected particles, you can open the model with 3dmodv. However, if your particles aren't organized in a lattice, it is quite difficult to do anything with this.
- Open <prefix>_<region>_bin<X>_convmap.mrc overlapped with <prefix>_<region>_bin<X>.mod to look and remove particles. Another possibility is to use the binned tomogram in cache/<prefix>_<region>_bin<X>.rec.

  ```
  3dmod <*>_convmap.mrc <*>.mod
  # or
  3dmod ../cache/<*>.rec <*>.mod
  ```
- You can always change the sampling, the angular search, the template or the threshold and re-run templateSearch, if you are not satisfied with the results. If you want to change the sub-region coordinates, change the boundaries in ../bin10/<prefix>_bin10.mod and re-run the recScrip2.sh script as explained in section 7.
- TODO Remove neighbours
- TODO Constrains

This directory cannot be seen by emClarity as it currently is. You will need to rename it to <projectDir>/convmap before going to the next step.

> ***Note 12:*** *For a full description of the outputs generated by* templateSearch*, you should refer to section* ***16.6****.*

> ***Note 13:*** *Changing the contours coordinates or adding new contours in the* .mod *file has no effect. You can only remove contours. See section* ***9*** *for more details.*

Note 14: *The `wedgeType_2` prefix comes from the an old version of emClarity and indicates the type of missing wedge mask to apply to the tomogram and the template before computing the cross-correlation scores. This parameter is no longer used.*

8.5 Import particles from another software

If you decide to use another software to pick your particles, you can still import them into emClarity. In this case, you would need to create your own `.csv` and `.mod` files (see table 15). These files are relative to the sub-regions, which should still be defined as described in section 7. If your coordinates are relative to the entire field of view, you can either define the entire field of view as your sub-regions or you can subtract the x_{min}, y_{min} and z_{min} of the sub-regions you defined in section 7 to your coordinates. As specified in table 15, the coordinates origin is at the lower left corner (at least as visualized in `3dmod`). Of course, you should still run `ctf estimate` (section 6) before going to the next section.

9 Initialize the project

9.1 Objectives

This step is creating the project metadata that will be used throughout the processing. There are three main things that emClarity will do. First, it will grab the sub-region coordinates in `/recon/<prefix>.coords`. Second, it will grab the tilt-series CTF estimate stored in `fixedStacks/ctf/<prefix>_ali1_ctf.tlt`. Lastly, it will grab the particle coordinates from `convmap/<prefix>_<nb>_<bin>.csv`. As explained in the last section, peaks can be removed from the `.csv` files using the corresponding `.mod` file.

> ***Note 15:*** *Once ran, these files are ignored and will not be used again. If one needs to modify some of the above-mentioned information, this step must be re-run for the modification to be effective.*

9.2 Parameters

Table 7: `init` parameters. Your parameter file should have the following parameters.
* indicates the required parameters, the other parameters are optional.

– `subTomoMeta`*	Project name. Most output files will have the project name as prefix and the metadata is saved in a MATLAB file called `<subTomoMeta>.mat`.
– `Tmp_samplingRate`*	Sampling (i.e. binning) at which the template matching was ran. It should corresponds to the binning registered in the filename of the .csv and .mod file.
– `fscGoldSplitOnTomos`*	Whether or not the particles from the same subregions should be kept in the same half-set or distributed randomly. Two sub-regions are overlapping, you can set this parameter to 1/True to prevent the duplicates to be in the same half-set. However, we do recommend to keep the sub-regions from overlapping and always keep this parameter to 0/False.
– `lowResCut`	This correspond to a rough estimate of the initial resolution of the data-set, directly coming out from the picking. The default value is set to 40Å and it can be lowered for data-sets collected at low defocus. In most situation, the default value is fine to start with. As the processing goes, emClarity will progressively lower this resolution estimate using the FSC.

9.3 Run

The `init` routine has the following signature:

```
>> emClarity init <param>
```

where `<param>` is the parameter file name.

9.4 Outputs

The main output is of course the output file `<subTomoMeta>.mat`. This step should only take a few seconds to run and it will output to the terminal, the total number of particles and the number of particles before and after cleaning, for each sub-region.

10 Reconstruct the tomograms

10.1 Objectives

The objective here is to reconstruct the tomograms that will be later used to extract the particles and calculate the references (i.e the half-maps). For high-resolution sub-tomogram averaging, we must account for a defocus gradient perpendicular to the til-axis, as well as a gradient through the thickness of the sample due to Ewald sphere curvature. The former is handled by multiplying the Fourier transform of each projection by their measured CTFs. In addition to correcting for contrast inversions, multiplication (rather than phase-flipping) also helps to supress noise near the CTF zeros. Previous approaches multiplied the CTF on thin strips or small tiles, which can lead to significant aliasing (see [2] Figure S2 for a nice explanation). In emClarity, the full tilt image is multiplied by the CTF for a given defocus, inverse Fourier transformed and then pixels corresponding to that defocus are extracted. This is repeated over the range of defoci in the image, thus avoiding any aliasing. Each view is also weighted according to the cumulative electron dose as described in [3]. emClarity then uses the IMOD program tilt to reconstruct the tomograms, preserving the full context of the tomogram while simultaneously considering local movement and anisotropic magnification due to microlensing from charge accumulation on the specimen [4].

10.2 Parameters

Table 8: `ctf 3d` parameters. Your parameter file should have the following parameters.
* indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Microscope settings	
– `PIXEL_SIZE`*	Pixel size in meter per pixel (e.g. 1.8e-10). Must match the header of the stacks in `fixedStacks/*.fixed`.
– `SuperResolution`*	Whether or not the `fixedStacks/*.fixed` are super-sampled. Not that this should be the same value you used for `ctf estimate` in section 6.
– `Ali_samplingRate`*	Binning factor (1 means no binning) of the output reconstruction.
CTF correction	
– `useSurfaceFit`*	Whether or not the spatial model should be calculated as a function of x, y coordinates. If 0, the spatial model is a plane (constant center-of-mass). See section 16.7.3 for more details.
– `flg2dCTF`*	Whether or not the CTF correction should correct for the defocus gradients along the electron beam (thickness of the specimen). If 1, only one z section is used. See section 16.7.2 for more details.
Others	
– `erase_beads_after_ctf`	Whether or not the fiducial beads should be removed before or after CTF multiplication. Default=`false`.
– `applyExposureFilter`	Whether or not the exposure filter should be applied. If you turn it off, make sure it is turned-off during subtomogram averaging as well.

– super_sample*	Compute the back projection in a slice larger by the given integer factor (max=8) in each dimension, by interpolating the projection data at smaller intervals ("super-sampling"). This corresponds to the SuperSampleFactor entry from tilt. Default=0.
– expand_lines*	If super_sample is greater than 0, expand projection lines by Fourier padding (sync interpolation) when super-sampling, which will preserve higher frequencies better but increase memory needed. This corresponds to the ExpandInputLines entry from tilt.Default=0

10.3 Run

To run it, you just need to specify your parameter file and emClarity will look at the project metadata to extract the current alignment cycle. If your `cache` directory already contains reconstructions at this binning (`Ali_samplingRate`), emClarity will not override them. As such, you should remove any previous reconstructions at that binning from your `cache`.

```
>> emClarity ctf 3d <param.m>
```

10.4 Outputs

The binned tilt-series and CTF-corrected reconstruction are saved in the `cache` directory.

11 Subtomogram averaging

11.1 Objectives

Once the tomograms are generated, emClarity can calculate the subtomogram averages, one for each half-set. A volume-normalized single-particle Wiener (SPW) filter [5] will minimize the reconstruction error of the particle density by applying an "optimal" (low-pass) filter and correcting for all of the systematic changes to the signal imposed by the microscope and image processing algorithms on the reconstructions.

> ***Note 16:*** *This filter is optimum as long as we correctly estimate the Signal-to-Noise Ratio of the two subtomogram averages (one for each half-set). This estimate depends on the conical Fourier shell correlation described in* ***16.8****.*

11.2 Parameters

Table 9: `avg` parameters. Your parameter file should have the following parameters. * indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Sampling	
– `PIXEL_SIZE`*	Pixel size in meter per pixel (e.g. 1.8e-10). Must match the header of the stacks in `fixedStacks/*.fixed`.
– `SuperResolution`*	Whether or not the `fixedStacks/*.fixed` are super-sampled. Not that this should be the same value you used for `ctf estimate` in section 6.
– `Ali_samplingRate`*	Current bin factor (1 means no binning). The sub-region tomograms at this given binning must be already reconstructed in the `cache` directory. If they aren't, you'll need to run `ctf 3d` before running this step.
Mask	
– `Ali_mType`*	Type of mask; "cylinder", "sphere", "rectangle".
– `particleRadius`*	[x, y, z] particle radius, in Å. It should be the smallest values to contain particle. For particles in a lattice, neighboring particles can be used in alignment by specifying a larger mask size, with `Ali_Radius`, but this parameter must correspond to the central unit.
– `Ali_mRadius`*	[x, y, z] mask radius, in Å. The mask size must be large enough to contain the entire particle (i.e. larger than `particleRadius`), the delocalized signal, proper apodization, and to avoid wraparound error in cross-correlation.
– `Ali_mCenter`*	[x, y, z] shifts, in Å, relative to the center of the reconstruction. Positive shifts translate the `Ali_mType` mask to the right of the axis.
– `scaleCalcSize`*	Scale the box size (section 16.8.1) by this number. Default=1.5.

Symmetry	
– `Raw_classes_odd`*	Parameter used to control the C symmetry of the first half set. It should be “`[0; <C>.*ones(2,1)]`”, where `<C>` is the central symmetry. This is equivalent to `[0; <C>; <C>]`.
– `Raw_classes_eve`*	Parameter used to control the C symmetry of the second half set. It should be identical to `Raw_classes_odd`.
– `symmetry`	New parameter used to control the symmetry. CX, I/I2, O, and DX are supported. The old default will be used if this parameter is not specified.
Fourier shell correlation	
– `flgCones`*	Whether or not emClarity should calculate the conical FSCs. This will greatly impact the calculated half-maps if you have preferential orientation in the data-set. We recommend to leave this to 1 (true) throughout the workflow.
– `minimumParticleVolume`*	Defines a minimum value for the $f_{mask}/f_{particle}$ ratio (section 16.8.4 and 16.8.5). Default=0.1
– `flgFscShapeMask`*	Apply a very soft mask based on the particle envelope before calculating the FSC (section 16.8.4). We highly recommend to not turn this off. Default=1.
– `shape_mask_test`	Exit after saving the shape mask in `FSC/`. This is useful when testing the following two parameters. Default=0.
– `shape_mask_lowpass`	Low-pass cutoff, in Å, to apply to the median filtered raw subtomogram averages. Default=14.
– `shape_mask_threshold`	Initial threshold used for the dilation. If you have extra "dust" outside your particle, it can be helpful to decrease the resolution used in the inital thresholding, i.e. increase `shape_mask_lowpass`, or to increase this threshold (or both). Default=2.4.
Others	
– `subTomoMeta`*	Project name. At this step, emClarity excepts to find the metadata `subTomoMeta`.mat in the project directory. Most output files will have the project name as prefix.
– `Raw_className`*	Class ID for subtomogram averaging and alignment. You should leave it set to zero.
– `Fsc_bfactor`*	Global B-factor applied to both references. See 16.8.8 for more detail. It can be a vector. In this case, if it is an intermediate reconstruction (i.e. reconstructing half-maps with `RawAlignment`) only the first value is used, if it is a final reconstruction (i.e. `FinalAlignment` introduced in section 15), emClarity will calculate one reconstruction per value registered in this vector.
– `flgClassify`*	Whether or not this cycle is a classification cycle. It must be 0 if subtomogram alignment is the next step. More information on this in 14.

– flgCutOutVolumes*	Whether or not each transformed particle (rotated and sifted) should be saved to cache directory. Note that the subtomogram have an extra padding of 20 pixel. This makes the reconstruction much slower if activated. Default=0.
– flgQualityWeight*	The particles with an alignment score (Constrain Cross-Correlation score, CCC) below the average CCC score of the entire data-set are down-weighted before being added to the reference. This parameter regulates the strength of the weighting.
– flgCCCcutoff*	Particles with an alignment score (Constrain Cross-Correlation score, CCC) below this value are ignored from the reconstruction. Default=0.
– use_v2_SF3D*	Whether or not the new per-particle sampling function procedure should be used, as opposed to the older "grouped" sampling functions. This is the default since emClarity 1.5.1.0. Default=1.
– mtfVal*	Type of Modulation Transfer Function (MTF) of the detector. This is applied on top of the volume-normalized SPA Wiener filter (see section 16.8.8. Set it to 0 to turn it off, 1 for a stronger version. You can also set your own (see code and code) but this is not tested. Default=2.

11.3 Run

To call the subtomogram averaging procedure:

```
>> emClarity avg <param.m> <cycle_nb> RawAlignment
```

where `<param.m>` is the parameter file you want to use for this cycle, `<cycle_nb>` is the cycle number, starting from 0. Each cycle starts with this step and is usually followed by the subtomogram alignment procedure. `RawAlignment` indicates to emClarity that we want to calculate the subtomogram average of the entire half-set. As we'll see later, it is also possible to compute one average (one cluster) for each class.

If you suspect some densities are being omitted by the molecular mask or if you see extra 'dust' outside your particle, you can change with the `shape_mask_*` parameters. If you want to only calculate the mask, you can run the following, with `shape_mask_test=1`:

```
>> emClarity fsc <param.m> <cycle_nb> RawAlignment
```

The same molecular mask is used for averaging and alignment. The values that you use for the averaging will be used during the alignment.

11.4 Outputs

The two subtomogram averages (half-maps) are saved into the project directory, as:
`cycleXXX_projectName_class0_REF_EVE.mrc`, and
`cycleXXX_projectName_class0_REF_ODD.mrc`.

The FSC is saved in `FSC/cycleXXX_projectName_Raw-1-fsc_GLD.pdf`. The corresponding `.txt` file constains the FSC values. The first column is the resolution, in Å^{-1}. The second column contains the Correlation Coefficients (CCs). The other 36 columns contain the CCs for each FSC cones.

For more details about the outputs, see 16.8.

12 Subtomogram alignment

12.1 Objectives

Once the reconstructions are generated, we can estimate the ϕ, θ, ψ rotations and x, y, z translations between the references and each particle, or in other words, we can maximise the constrained cross-correlation (CCC) between the references and each particle (more details in section 16.9). As this procedure directly compares the references with the particles and because the references result from the average of the transformed particles, the subtomogram averaging and alignment steps should be run until convergence of the references.

12.2 Parameters

Table 10: alignRaw parameters. Your parameter file should have the following parameters. * indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Sampling	
– PIXEL_SIZE*	Pixel size in meter per pixel (e.g. 1.8e-10). Must match the header of the stacks in fixedStacks/*.fixed.
– SuperResolution*	Whether or not the fixedStacks/*.fixed are super-sampled. Not that this should be the same value you used for ctf estimate in section 6.
– Ali_samplingRate*	Current bin factor (1 means no binning). The sub-region tomograms at this given binning must be already reconstructed in the cache directory. You must use the same sampling than during avg.
Mask	
– Ali_mType*	Type of mask; "cylinder", "sphere", "rectangle".
– particleRadius*	[x, y, z] particle radius, in Å. It should be the smallest values to contain particle. For particles in a lattice, neighboring particles can be used in alignment by specifying a larger mask size, with Ali_Radius, but this parameter must correspond to the central unit.
– Ali_mRadius*	[x, y, z] mask radius, in Å. The mask size must be large enough to contain the entire particle (i.e. larger than particleRadius), the delocalized signal, proper apodization, and to avoid wraparound error in cross-correlation.
– Ali_mCenter*	[x, y, z] shifts, in Å, relative to the center of the reconstruction. Positive shifts translate the Ali_mType mask to the right of the axis.
– Peak_mRadius	Further restrict the translations to this radius. By default (0), the translations are limited to the particleRadius.

– `flgCenterRefCOM`	Whether or not the references should be shifted to their center-of-mass before starting the alignment[2]. Default=1.
– `scaleCalcSize`*	Scale the box size (section 16.8.1) by this number. Default=1.5.
Symmetry	
– `Raw_classes_odd`*	Parameter used to control the C symmetry of the first half set. It should be "`[0; <C>.*ones(2,1)]`", where `<C>` is the central symmetry. This is equivalent to `[0; <C>; <C>]`.
– `Raw_classes_eve`*	Parameter used to control the C symmetry of the second half set. It should be identical to `Raw_classes_odd`.
– `symmetry`	New parameter used to control the symmetry. CX, I/I2, O, and DX are supported. The old default will be used if this parameter is not specified.
Angular search	
– `Raw_angleSearch`*	Angular search, in degrees. Format is $[\Theta_{out},\ \Delta_{out},\ \Theta_{in},\ \Delta_{out}]$. For example, $[180,\ 15,\ 180,\ 12]$, specifies a ±180° out of plane search (polar and azimuth angles) with 15° steps and ±180° in plane search (planar angles) with 12° steps.
Others	
– `subTomoMeta`*	Project name. At this step, emClarity excepts to find the metadata `subTomoMeta`.mat in the project directory. Most output files will have the project name as prefix.
– `Raw_className`*	Class ID for subtomogram averaging and alignment. You should leave it set to zero.
– `Cls_className`*	Class ID for classification. You should leave it set to zero for now. For more information, see section 14.
– `Fsc_bfactor`*	Global B-factor applied to both references. Although it can be a vector, at this stage emClarity will only use the first number as B-factor. See 16.8.8 for more detail.
– `flgClassify`*	Whether or not this cycle is a classification cycle. It must be 0 if subtomogram alignment is the next step. More information on this in 14.
– `use_v2_SF3D`*	Whether or not the new per-particle sampling function procedure should be used, as opposed to the older "grouped" sampling functions. This is the default since emClarity 1.5.1.0. Default=1.

12.3 Run

To call the subtomogram alignment procedure:

```
>> emClarity alignRaw <param.m> <cycle_nb>
```

[2] For membrane proteins, or phage particles for example, this parameter should be shut off either from the beginning, or after a cycle or two.

where `<param.m>` is the parameter file you want to use for this cycle and `<cycle_nb>` is the cycle number. As the alignment needs the reconstructions, one must first run the subtomogram averaging procedure.

The sub-regions are loaded and processed independently from one another during the alignment. Once the particles from a sub-region are aligned, the results from the alignment are saved in the `alignResume` directory. emClarity will not re-run the alignment of a particular sub-region if the results for this sub-region are saved in `alignResume`. This creates the possibility to run multiple instances of emClarity, each one working on a different set of sub-regions. This is especially useful if you have multiple machines connected to the same storage. emClarity offers two possibilities:

- **Reverse order**: You can run two instances of emClarity, the first one, as usual, will process the sub-regions in a particular order and save the results on a sub-region basis.

  ```
  >> emClarity alignRaw <param.m> <cycle_nb>
  ```

  Then, you can run the same job, but with a negative cycle number:

  ```
  >> emClarity alignRaw <param.m> -<cycle_nb>
  ```

  This second run will not update the metadata and will process the data in the reverse order. The first instance of emClarity will use the results from this run to update the metadata.

- **Divide the dataset**: If you want to run more than two instances of emClarity, it is also possible using the following signature.

  ```
  >> emClarity alignRaw <param.m> [<cycle_nb>, <idx>, <jobs>]
  ```

  where `<cycle_nb>` is the cycle number (as usual), `<jobs>` is the number of total jobs to split into, which should be less than the number of sub-regions, of course. `<idx>` is the index of the current job, from 1 to `<total>`. For instance, if you want to share the alignment across 4 GPU nodes and your cycle number is 8, you have to run the following commands.

  ```
  >> emClarity alignRaw <param.m> [8, 1, 4]  # @node1
  >> emClarity alignRaw <param.m> [8, 2, 4]  # @node2
  >> emClarity alignRaw <param.m> [8, 3, 4]  # @node3
  >> emClarity alignRaw <param.m> [8, 4, 4]  # @node4
  ```

  Then, run the usual command as shown below. This should only take a few seconds as it only extracts the results saved in `alignResume` by the different jobs and update the metadata.

  ```
  >> emClarity alignRaw <param.m> 8
  ```

12.4 Outputs

The results (relative rotation and shifts, plus the new associated CCC score, per particle) from the alignment are saved in `alignResume` and the metadata is updated.

Usually at this point of the workflow, you can either start a new cycle and reconstruct the subtomogram average (section 11), the final reconstruction (section 15) or run a classification, or you can stay in the same cycle and run a tilt-series refinement (section 13).

13 Tilt-series refinement

13.1 Objectives

Colloidal gold beads have two key characteristics: they are highly contrasted and they are mostly round. As such, it is quite easy to define the (x, y) center of each bead over every projection, making them suitable fiducial markers for tilt-series alignments. Unfortunately, we need a lot of beads in the field of view to have an accurate description of the local "movements" in the specimen. Fortunately, subtomogram averaging provides accurate estimates of both particle positions and high SNR reconstructions, making them excellent fiducial markers. It is thus possible to leverage this information for improving the alignment of a tilt-series.

To do so, the density corresponding to the particles in the original tomogram is replaced with a copy of the subtomogram average at the appropriate orientation. This augmented tomogram is then re-projected into a tilt-series using the current global and local geometric model. Tiles around each subtomograms projected origin are convoled with the local CTF (phase correction) and aligned locally to the corresponding tile from the raw data. This defines a new fiducial model that can be used by tiltalign to refine the tilt-series alignment.

> ***Note 17:*** *As the entire tomogram is projected, the reference projections generated to refine the location of the subtomogram fiducial markers in the raw images include information from neighboring reprojected particles, as well as any other local densities.*

> ***Note 18:*** *tiltalign forces the fiducial markers to behave similarly within a given projection (spatial restrictions), while also requiring them to vary smoothly as a group from projection to projection (spatiotemporal restrictions).*

13.2 Parameters

Table 11: `tomoCPR` parameters. Your parameter file should have the following parameters. * indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Sampling	
– `PIXEL_SIZE`*	Pixel size in meter per pixel (e.g. 1.8e-10). Must match the header of the stacks in `fixedStacks/*.fixed`.
– `SuperResolution`*	Whether or not the `fixedStacks/*.fixed` are super-sampled. Not that this should be the same value you used for `ctf estimate` in section 6.
– `Ali_samplingRate`*	Current binning factor (1 means no binning). The sub-region tomograms at this given binning must be already reconstructed in the `cache` directory. If they aren't, you'll need to run `ctf 3d` before running this step.

Fiducial alignment	
– `Ali_mType`*	Type of mask; “cylinder”, “sphere”, “rectangle”.
– `particleRadius`*	$[x,\ y,\ z]$ particle radius, in Å. It should be the smallest values to contain particle. For particles in a lattice this parameter must correspond to the central unit.
– `Ali_Radius`*	$[x,\ y,\ z]$ mask radius, in Å. The mask size must be large enough to contain the entire particle (i.e. larger than `particleRadius`), the delocalized signal, proper apodization, and to avoid wraparound error in cross-correlation.
– `Ali_mCenter`*	$[x,\ y,\ z]$ shifts, in Å, relative to the center of the reconstruction. Positive shifts translate the `Ali_mType` mask to the right of the axis.
– `peak_mask_fraction`	Fraction of the `particleRadius` used for the alignment. This has no effect if `Peak_mType` and `Peak_mRadius` are defined. Default=0.4.
– `Peak_mType`	Further restrict the translations by applying an additional mask; “cylinder”, “sphere”, “rectangle”.
– `Peak_mRadius`	Further restrict the translations to this radius. By default (0), the translations are limited by `particleRadius` and `peak_mask_fraction`. This has no effect if `Peak_mType` is not defined and it will switch back to the default behavior.
– `tomoCprLowPass`	Low-pass filter cutoff applied to the tiles, in Å. By default, it corresponds to $1.5\times$ the current resolution and it is forced to be between 10Å and 24Å.
– `tomoCprDefocusRefine`*	Refine the defocus value during alignment. This feature is experimental and we don’t recommend you to use it at the moment. Default=false.
– `tomoCprDefocusRange`*	Range of defocus to sample around the current defocus estimate, in meter.
– `tomoCprDefocusStep`*	Defocus step, in meter.
– `min_res_for_ctf_fitting`*	Low-pass filter cutoff applied to the tiles, in Å. It replaces `tomoCprLowPass` in case of a defocus search. If $\sqrt{2}$`PIXEL_SIZE` $<$ `min_res_for_ctf_fitting` the defocus search is turned off. Default=10.
Tilt-series alignment	
– `particleMass`*	Rough estimate of the particle mass, in MDa. This is used to set the number of particles per patch. Smaller particles give less reliable alignments, so more of them are included in each patch, creating a smoother solution.

– tomoCPR_randomSubset	The maximal number of particle which are going to be used as fiducials. If there is more particles available than this, a random subset of particles will be selected. Default=-1, all particles.
– k_factor_scaling	KFactorScaling from tiltalign. Default=$10/\sqrt{\text{tomoCPR\_randomSubset}}$.
– rot_option_global	RotOption from tiltalign. Default=1.
– rot_option_local	LocalRotOption from tiltalign. Default=1.
– rot_default_grouping_local	LocalRotDefaultGrouping from tiltalign. Default=3.
– mag_option_global	MagOption from tiltalign. Default=1.
– mag_option_local	LocalMagOption from tiltalign. Default=1.
– mag_default_grouping_global	MagDefaultGrouping from tiltalign. Default=5.
– mag_default_grouping_local	LocalMagDefaultGrouping from tiltalign. Default=5.
– tilt_option_global	TiltOption from tiltalign. Default=5.
– tilt_option_local	LocalTiltOption from tiltalign. Default=5.
– tilt_default_grouping_global	TiltDefaultGrouping from tiltalign. Default=5.
– tilt_default_grouping_local	LocalTiltDefaultGrouping from tiltalign. Default=5.
– min_overlap	MinSizeOrOverlapXandY from tiltalign. Default=0.5.
– shift_z_to_to_centroid	If true/1, use the ShiftZFromOriginal entry from tiltalign and set AxisZShift to 0. Default=true.
Others	
– subTomoMeta*	Project name. At this step, emClarity excepts to find the metadata <subTomoMeta>.mat in the project directory.
– Raw_className*	Class ID. You should leave it set to zero.
– Raw_classes_odd*	This is only used to set the number of references. As the multi-reference alignment is currently disabled, you can leave it as whatever you used for the subtomogram average or set it to [0;0].
– Raw_classes_eve*	It should be identical to Raw_classes_odd.

13.3 Run

13.3.1 Tilt-series refinement

The tilt-series refinement should be ran after avg or alignRaw. To run it, run the following command:

```
>> emClarity tomoCPR <param.m> <cycle_nb>
```

where <param.m> is the parameter file you want to use for this cycle and <cycle_nb> is the cycle number.

This create a new directory, called mapBack<n>, where <n> is the number of tomoCPR that you have ran before. This directory contains all the outputs from tiltalign. See section 16.10 for more information about the outputs.

Similarly to the subtomogram alignment, tomoCPR can be divided into multiple jobs. To do so, you will have to use the following signature:

```
>> emClarity tomoCPR <param.m> [<cycle_nb>, <idx>, <jobs>]
```

where `<cycle_nb>` is the cycle number (as usual), `<jobs>` is the number of total jobs to split into, which should be less than the number of sub-regions, of course. `<idx>` is the index of the current job, from 1 to `<total>`. For instance, if you want to share the refinement across 4 GPU nodes and your cycle number is 8, you have to run the following commands.

```
>> emClarity tomoCPR <param.m> [8, 1, 4]  # @node1
>> emClarity tomoCPR <param.m> [8, 2, 4]  # @node2
>> emClarity tomoCPR <param.m> [8, 3, 4]  # @node3
>> emClarity tomoCPR <param.m> [8, 4, 4]  # @node4
```

Then, run this final command to update the metadata:

```
>> emClarity tomoCPR <param.m> [8,0,0]
```

13.3.2 Update the aligned tilt-series

`tomoCPR` does not update the aligned the tilt-series, so to make the new tilt-series alignment effective, you have to run:

```
>> emClarity ctf update <param.m> <cycle_nb>
```

This will create a new aligned tilt-series in `aliStacks/<prefix>_ali<n+1>.fixed`, using the updated tilt-series alignment from `tomoCPR`, as well as a new `*_ctf.tlt` file (table 14) in `fixedStacks/ctf/<prefix>_ali<n+1>_ctf.tlt`.

14 Classification

14.1 Objectives

The heterogeneity of the data-set can be analysed by comparing individual particles with the current subtomogram average. To correct for differences in sampling between the reference and an individual particle, the current subtomogram average is distorted by the particle's 3D-sampling function it is being compared to. This effectively estimates what the average particle should look like at that subtomogram position, allowing to only compare meaningful differences. The dimensionality of these differences is reduced by principal component analysis, using singular-value decomposition (SVD). Features of a given length scale (e.g. α-helices, small protein domains, etc.) can be focused on and considered simultaneously by band-pass filtering the reconstructions and computing the SVD for each length scale. The singular vectors describing the greatest variance for each length scale are then concatenated into feature vectors and clustered with k-means or other clustering algorithms.

14.2 Parameters

Table 12: `pca` parameters. Your parameter file should have the following parameters. * indicates the required parameters, * indicates expert parameters. Expert parameters should not be changed except if you know what you are doing. The other parameters are optional.

Sampling	
– `PIXEL_SIZE`*	Pixel size in meter per pixel (e.g. 1.8e-10). Must match the header of the stacks in `fixedStacks/*.fixed`.
– `SuperResolution`*	Whether or not the `fixedStacks/*.fixed` are super-sampled. Not that this should be the same value you used for `ctf estimate` in section 6.
– `Cls_samplingRate`*	Current binning factor (1 means no binning). The sub-region tomograms at this given binning must be already reconstructed in the `cache` directory. If they aren't, you'll need to run `ctf 3d` before running this step.
– `Ali_samplingRate`*	Binning factor (1 means no binning) of the half-maps of the current cycle. If this is different from `Cls_samplingRate`, the reconstructions will be resampled to match `Cls_samplingRate`.
Masks	
– `Ali_mType`*	Type of mask used for the reconstruction; "cylinder", "sphere", "rectangle".
– `Cls_mType`*	Type of mask to use for the PCA; "cylinder", "sphere", "rectangle".
– `Ali_Radius`*	[x, y, z] mask radius, in Å, used for the reconstruction.
– `Cls_Radius`*	[x, y, z] mask radius, in Å to use for the PCA.
– `Ali_mCenter`*	[x, y, z] shifts, in Å, used for the reconstruction. These are relative to the center of the reconstruction. Positive shifts translate the `Ali_mType` mask to the right of the axis.

– `Cls_mCenter`*	[x, y, z] shifts, in Å to use for the PCA. These are relative to the center of the reconstruction. Positive shifts translate the `Ali_mType` mask to the right of the axis.
– `flgPcaShapeMask`	Calculate and apply a molecular to the difference maps. This molecular mask is calculated using the combined reference. Default=true.
– `test_updated_bandpass`	By default (0/false), low-pass filters are used to calculate the length scales. If true, use band-pass filters. See section 16.11.2 for more details.
PCA	
– `pcaScaleSpace`*	Length scales, i.e. resolution bands, in Å. If this is a vector, the PCA will be performed for each length scales and you will need to select the principal axes for each length scale.
– `Pca_randSubset`*	For very large data sets (tens of thousands) the principal axes describing the important variation can be obtained with a subset of the data, which saves a lot of computation. If 0, use the entire dataset, otherwise specify the number of particle that should be randomly selected to perform the decomposition. Usually 25% or at least 3000-4000 is a good way to go.
– `Pca_maxEigs`*	Most of the variance is usually explained within the first 20 to 30 directions, so it is usually not useful to save all of the directions. Use this parameter to select the number of principal directions to save.
Clustering	
– `Pca_coeffs`*	The selected principal axes, for each length scale. Each length scale is a row. You can select as many (or few) from any resolution as you want, but the number of entries in each row must be constant. Use zeros to fill empty places.
– `Pca_clusters`*	The number of clusters to find. If this is a vector, the clustering will be calculated for each registered value.
– `Pca_distMeasure`	This corresponds to the `Distance` entry of the `kmeans` function of MATLAB. By default, the squared Euclidean distance metric, i.e. each centroid is the mean of the points in that cluster.
– `Pca_nReplicates`	This corresponds to the `Replicates` entry of the `kmeans` function of MATLAB. By default, the number of replicates is set to 128, i.e. the number of times to repeat clustering using new initial cluster centroid positions.
Others	
– `PcaGpuPull`*	The decomposition is calculated on the CPU, but the difference maps are calculated on the GPU, which is much more efficient. This parameters controls how many difference maps should be held on the GPU at any given type.
– `flgClassify`*	Specify that the this cycle is a classification cycle. Must be set to 1/true.

– subTomoMeta*	Project name. At this step, emClarity excepts to find the metadata subTomoMeta.mat in the project directory.
– flgCutOutVolumes	Whether or not each transformed particle (rotated and sifted) used to calculate the difference maps should be saved to cache directory. Note that the subtomogram have an extra padding of 20 pixel. This makes the pre-processing for the PCA much slower if activated. Default=0.
– scaleCalcSize*	Scale the box size used to calculate the difference maps by this number. Default=1.5.
– use_v2_SF3D	Whether or not the new per-particle sampling function procedure should be used, as opposed to the older "grouped" sampling functions. This is the default since emClarity 1.5.1.0. Default=1.

14.3 PCA

14.3.1 Run

Before running the PCA, you have to start a new cycle by running the following command:

```
>> emClarity avg <param.m> <cycle_nb> RawAlignment
```

Then, you can run the PCA on the entire dataset:

```
>> emClarity pca <param.m> <cycle_nb> 0
```

For very large data sets, i.e. tens of thousands particles, the principal directions describing the important variation can be obtained with a subset of the data, which saves a lot of computation. In this case, use the Pca_randSubset parameter and run the same command shown above. Then, to calculate the principal components of the rest of the dataset, run this second command:

```
>> emClarity pca <param.m> <cycle_nb> 1
```

14.3.2 Outputs

The goal now is to select, for each length scale, the principal directions that are going to be used to reproject the data onto. There are a few files that are here to help you decide, all of which are introduced in section 16.11.4.

For example, let's say you have 3 length scales and want to select the principal directions 3 to 5 for the first scale, 4, 5, 8, 9 for the second scale, and 3 and 5 for the last scale, you would have to write in your parameter file:

Pca_coeffs=[3:5,0 ; 4,5,8,9 ; 3,5,0,0], or equivalently
Pca_coeffs=[3,4,5,0 ; 4,5,8,9 ; 3,5,0,0]

In some cases, it is not obvious which principal directions to choose and it might require some trials before successfully clustering the dataset into meaningful clusters. Moreover, it might not be clear if the classification was successful or not until the reconstruction step.

14.4 Clustering

14.4.1 Run

Once that you have selected the principal axes, for each length scale, we can select the corresponding principal components and cluster them.

You can specify how many clusters you want with the `Pca_clusters` parameters. For example, say you want to run 3 clustering, one with 2 clusters, one with 4 and one with 6:

```
Pca_clusters=[2,4,6]
```

Then, run the following command:

```
>> emClarity cluster <param.m> <cycle_nb>
```

14.4.2 Outputs

For each requested number of clusters (`Pca_clusters`), the class populations are printed into a text file in the project directory called `<projectName>_cycleXXX_ClassIDX.txt`.

14.5 Reconstruction

To create a montage of your class averages, we need to call the `avg` procedure. As such, your parameter file should contains the parameters necessary to run `avg` (table 9), plus an additional 3 parameters. Indeed, you have to specify which clustering to use. Say you ran 3 clustering, one of them with 6 clusters, like in the previous example. To reconstruct the subtomogram average of these 6 classes, you have to specify the following parameters in your parameter file:

`Cls_className=6`
`Cls_classes_odd=[1:6,1.*ones(1,6)]`
`Cls_classes_eve=[1:6,1.*ones(1,6)]`, or equivalently
`Cls_classes_eve=[1:6, 1,1,1,1,1,1]`

Each entry in the second row (everything after `1:6,`) is a CX symmetry, which may not be the same for each class. For example, you first ran the reconstruction without symmetry and realize that you have a C3 symmetry for all the classes, except in the last 2 classes where there is a C6 symmetry, you could then rerun the reconstruction with the following parameters:

```
Cls_classes_odd=[1:6, 3,3,3,3,6,6]
Cls_classes_eve=[1:6, 3,3,3,3,6,6]
```

> ***Note 19:*** *If you use the new* `symmetry` *parameter, the second row is not used but is still required for compatibility reasons.*

To call the `avg` function, run the following command:

```
>> emClarity avg <param.m> <cycle_nb> Cluster_cls
```

This can be done for each of your clustering specified in the `Pca_clusters` parameter.

14.6 Select the classes to use as references

At this point, there are two possibilities. If any of the clustering was successful and you would like to cancel the classification and switch back to averaging and alignment, you could either use and rename the metadata back-up from the previous cycle and continue as nothing happened, or you could skip the current cycle by running the following command:

```
>> emClarity skip <param.m> <cycle_nb>
```

On the other hand, if you have identified a good clustering, with some classes you would like to keep and some classes you would like to exclude, you will have to:

1. Make sure the last reconstruction was done using the clustering you want to use. For example, if you ran the clustering with `Pca_clusters=[2,4,6]`, then reconstructed the montage for all of these clustering, starting from 2 to 6, but want to use the second clustering with 4 classes, you would have to rerun the reconstruction of the second clustering before continuing.

2. Save a copy of your metadata. This is useful for two reasons. First, the next step will remove some particles from further consideration, so if something goes wrong, it's better to keep a copy of the current state of the metadata. Second, if you want to split your dataset in multiple parts to process some classes independently from the others, the easiest way is to use this copy to repeat the next steps for each (group of) classes you want to analyze.

3. **Select the classes you want to exclude**: Open your class montage in `3dmod`, in "Model" mode and create a new model point for each class you want to delete. Save the model file, for example as `classes2ignore.mod`.

4. **Exclude these classes**: Run the following command to exclude them; the particles that belongs to this/these class/classes will be tagged with a `-9999` in the metadata.

   ```
   >> emClarity geometry <param.m> <cycle_nb> Cluster_cls
      RemoveClasses <classes2ignore.mod> STD
   ```

   emClarity will tell you how many particles were removed and how many are left. This should correspond exactly to the class populations from the clustering (section 14.4.2). If it doesn't, stop and make sure you followed the instructions from the first step. Replace the metadata with your copy from step 2 and try again.

5. **End this cycle of classification**: Finally, you will need to "skip" the currently disabled class average alignment, and update the metadata so you may proceed to the next cycle of averaging/alignment.

   ```
   >> emClarity skip <param.m> <cycle_nb>
   ```

At this point, you can then turn off classification in your next parameter file (`flgClassify=0`) and proceed to the next cycle.

15 Final reconstruction

15.1 Objectives

This step consist into reconstructing the final reconstruction by combining the two half-set. emClarity offers two possibilities.

The first possibility is to calculate the final reconstruction from the two subtomogram averages (one per FSC group), within emClarity. Briefly, the half-maps are reconstructed as described in section 16.8 and the FSC is calculated, as well as the transformation between the two maps. Then, the particles from the second FSC groups are aligned to the first group, using the aforementioned transformation, while they are being re-extracted. The two aligned maps are then combined and filtered using the FSC calculated during the first reconstruction.

The second possibility is to calculate the reconstruction using cisTEM. In this case, emClarity reprojects the 3D coordinates of the particles in 2D, similarly to section 16.10.2. A cisTEM STAR file is created, containing, for each particle and for each view of the tilt-series, its x and y position, rotation, defocus, pre- and post-exposure, etc. cisTEM will then calculate an initial reconstruction using its `reconstruct3d` program, then refine it using `refine3d` (note that the ϕ, θ, ψ angles are not refined) and then finally calculates with `reconstruct3d` the final reconstruction using this refinement.

15.2 Parameters

This step requires the exact same parameters as the subtomogram averaging step (table 9, from section 11). The only difference is in the `fsc_bfactor`, as explained in table 9.

If you use cisTEM to reconstruct the final map, only the following parameters are used: `subTomoMeta`, `PIXEL_SIZE`, `particleRadius`, `VOLTAGE`, `Cs`, `AMPCONT`, `Ali_mRadius` and `particleMass`.

> ***Note 20:*** *This step assumes* `SuperResolution=0`*. If you set it to* `1` *during* `ctf estimate` *(which is not recommended), you'll have to enter the 'real' (aka Fourier cropped) pixel size for* `emClarity reconstruct`*.*

15.3 Run

15.3.1 With emClarity

First, you should start a new cycle by reconstructing the two half-maps and calculate the FSC:

```
>> emClarity avg <param.m> <cycle_nb> RawAlignment
```

Then, the second FSC group will be re-extracted while being aligned to the first group. Finally, the two maps are combined and filtered using the FSC we just calculated.

```
>> emClarity avg <param.m> <cycle_nb> FinalAlignment
```

This generates one filtered final reconstruction for every B-factor specified in `fsc_bfactor`.

15.3.2 With cisTEM

To calculate the final reconstruction with cisTEM, run the following command:

```
>> emClarity reconstruct <param.m> <cycle_nb> <prefix> <symmetry>
   <max_exposure>
```

where `<prefix>` is the prefix that will be added to every output generated by cisTEM. `<symmetry>` is the symmetry to use for the reconstruction and should corresponds to the `symmetry` parameter in table 9. `<max_exposure>` is the maximum exposure, in e/Å^2. Any images with more exposure that this value will be excluded from the reconstruction.

> ***Note 21:*** *The current implementation is quite limiting since it needs to store then entire dataset to RAM (see issue). While this is being fixed, one can reduce the* `<max_exposure>` *to 30-40 to reduce the dataset size.*

The scripts used to call cisTEM are saved in the project directory:

1. `<prefix>_ref.sh` is calculating the first reconstruction (`<prefix>_refFilt.mrc`) using the series of extracted particles (`<prefix>.mrc`) and the corresponding STAR file (`<prefix>.star`), both generated by emClarity. The FSC is saved in `<prefix>_stats.txt`.

2. `<prefix>_ref.sh` is then refining the alignment (only translations) and the statistics using the first reconstruction. The updated STAR file is saved as `<prefix>_refined.star`.

3. `<prefix>_ref2.sh` is calculating the final reconstruction (`<prefix>_refFilt_refined.mrc`) using the updated STAR file. The FSC is saved in `<prefix>_stats_refined.txt`.

16 Algorithms

16.1 Naming conventions

There is a lot of things to cover and it is often easier to use abbreviations (CTF, FSC, CCC, etc.) and symbols to refer to something.

Table 13: Symbols often used

Symbol	Description
$\boldsymbol{S}$	A reference, i.e. a subtomogram average or a template.
$\boldsymbol{s}$	A particle in 3D, i.e. a subtomogram.
$\boldsymbol{V}$	A tomogram. This usually refers to the full tomogram or a sub-region tomogram.
$\boldsymbol{I}$	An image. This can be an entire image, a strip or a tile.
$\boldsymbol{W}$	A weight in Fourier space. This can be a low- high- band-pass filter, a total 3D sampling function, 1D or 2D CTFs, an exposure or "B-factor" filter, etc. 3D sampling functions are referred as $\boldsymbol{w}$.
$\boldsymbol{M}$	Real space masks. This can be any shape mask, molecular masks, evaluation masks, etc.
$\mathbf{z}$	a defocus value
$\boldsymbol{\Delta}\mathbf{z}$	a defocus shift. $\boldsymbol{\Delta}\mathbf{z}_{ast}$ is the astigmatic shift.
$\boldsymbol{\phi}$	the azimuthal angle. $\boldsymbol{\phi}_{ast}$ is the astigmatic angle.
$\boldsymbol{R}$	a rotation matrix
$\boldsymbol{T}$	a translation
$\boldsymbol{\alpha}$	a tilt angle

Indexes are subscripts, e.g. the p^{th} subtomogram is referred as $\boldsymbol{s}_p$. This works with multiple indexes, e.g. the p^{th} subtomogram rotated by the r^{th} rotation is referred as $\boldsymbol{s}_{p,r}$. Labels are subscripts as well, e.g. if we want to specify that the subtomograms are in the reference frame, we would write $\boldsymbol{s}_{ref}$. On the other hand, if the symbol is labeled *and* an index is needed, the labeled symbol is placed between square brackets. For instance, if we want to specify that the p^{th} subtomogram is in the reference frame, we would write $[\boldsymbol{s}_{ref}]_p$.

16.2 Euler angles conventions

The ϕ, θ, ψ Euler angles used by emClarity describe a z-x-z active intrinsic rotations of the particles coordinate system. That is to say, to switch the particles from the microscope frame to the reference frame, the basis vectors of the subtomograms are rotated (positive anti-clockwise) around z, the new x, and the new z axis.

The microscope frame defines the coordinate system of the microscope, where the electron beam is the z axis. When the subtomograms are extracted from their tomogram, they are in the microscope frame. The reference frame is the coordinate system attached to the reconstruction, i.e. the subtomogram average and is usually set during the particle picking.

16.3 Linear transformations in Fourier space

Linear transformations are often applied in Fourier space directly. It might be useful to write down a few useful properties of the Fourier transforms.

- **Shift**: Shifting an image in real space is equivalent to applying a complex phase shift to its frequency spectrum, such as:

$$g(x,y) = f(x-\boldsymbol{a}, y-\boldsymbol{b}) \overset{\mathcal{F}}{\leftrightarrow} G(u,v) = F(u,v) \times \boldsymbol{e}^{\boldsymbol{-2\pi i\left(\frac{au}{N} + \frac{bv}{M}\right)}} \tag{1}$$

  where N and M are the number of u and v frequencies, respectively. The complex phase shift is just a complex number on the unit circle, so the magnitude of the Fourier transform is unchanged, so $|G(u,v)| = |F(u,v)|$.

- **Magnification**: Magnifying an image by a factor a is equivalent to magnifying its frequency spectrum by $1/a$, such as:

$$g(x,y) = f(\boldsymbol{a}x, \boldsymbol{b}y) \overset{\mathcal{F}}{\leftrightarrow} G(u,v) = \frac{\mathbf{1}}{|\boldsymbol{ab}|} \times F\left(\frac{u}{\boldsymbol{a}}, \frac{v}{\boldsymbol{b}}\right) \tag{2}$$

- **Rotation**: Rotating an image by an angle Θ in real space is the same as rotating its frequency spectrum by the same angle Θ.

16.4 Tilt-series alignment

16.4.1 Pre-processing

16.4.2 Patch tracking

16.4.3 Refining on beads

16.5 Defocus estimate

The defocus estimate is used at almost every step of the workflow to compute the theoretical CTFs. This procedure, called by `ctf estimate` is divided into 3 main steps. First, the raw tilt-series is transformed to compute the aligned tilt-series. Second, the average defocus of the entire aligned stack is estimated. Finally, the defocus is refined for each view.

16.5.1 Transform the tilt-series

1. **Pre-processing**: For each view of the input stack, "hot" pixels, i.e. pixels 6 standard deviations away from the mean of the view, are replace by random scaled values. Moreover, frequencies after Nyquist are removed by low-pass filtering.

2. **Transformation**: Each view of the pre-processed stack is transformed using the alignment files in `fixedStacks`, which effectively align the tilt-axis to the center of the images and parallel to the y axis.

   (a) Load `fixedStacks/<prefix>.xf`, decompose the rotation matrix into magnification and rotation $\boldsymbol{R}$, and get the $\boldsymbol{T}_x$ and $\boldsymbol{T}_y$ shifts.

   (b) Pad the view with its mean by a factor of 2, compute the forward Fourier transform and apply the magnification, rotation and shifts to the transform with bilinear interpolation. The spectrum is then Fourier cropped by a factor of 2 and inverse Fourier transformed to switch it back to real space. Increasing the sampling for the transformation reduces noise due to aliasing. Non-sampled regions are set to the mean, which is set to 0.

   (c) Apply the same transformation to a mask of "ones" of the same size as the view. This mask is used to track down non-sampled regions of the view. The stack of masks (1 mask per view) is saved as `aliStacks/<prefix>_ali1.samplingMask` and will be used later in the workflow.

3. **Erase the beads**: If `fixedStacks/<prefix>.erase` exists, spherical masks of diameter $1.2\times$ `beadDiameter` and positioned at the coordinates specified by this file, are used to replace the beads by random scaled values. The coordinates are applied directly on the aligned stack, not on the raw stack.

4. **Post-processing**:

   (a) In case "hot" pixels were introduced during transformation, these pixels are removed as in step **1** from the sampled regions (un-sampled pixels are delimited by the sampling mask from step **2.c**).

   (b) To keep track of the increasing thickness with the tilt angle, the views i are divided by their fraction of inelastic scattering events, $\boldsymbol{f}_{inelastic}$ defined as:

$$[\boldsymbol{f}_{inelastic}]_i = \exp\left(\frac{-\boldsymbol{t}}{\cos(\boldsymbol{\alpha}_i)\times\boldsymbol{t}_{inelastic}}\right) / \exp\left(\frac{-\boldsymbol{t}}{\cos(\boldsymbol{\alpha}_{min})\times\boldsymbol{t}_{inelastic}}\right) \tag{3}$$

   where $\boldsymbol{t}$ is the thickness estimate of the tomogram, which is set to 75nm, $\boldsymbol{\alpha}_i$ is the tilt-angle of the i^{th} view, $\boldsymbol{\alpha}_{min}$ is the lowest tilt-angle for the tilt-series and $\boldsymbol{t}_{inelastic}$ is the mean-free path of the inelastic scattering events, which is set to 400nm.

(c) Similarly, if a Saxton/cosine dose-scheme is used, the views i are divided by $1/\cos(\boldsymbol{\alpha}_i)$.

5. **Save tilt-series**: Save the aligned, fraction weighted, bead-erased tilt-series as `aliStacks/<prefix>_ali1.fixed`. These stacks are used by `templateSearch` (section 8) and `ctf 3d` (section 10) to reconstruct the tomograms.

16.5.2 Average power spectrum

We estimate the defocus by fitting theoretical CTFs, with varying defoci, to the power spectrum of the image. The quality of the fitting vastly depends on the magnitude of the Thon rings relative to the background. In a tilted image, the defocus ramp decreases the magnitude of the Thon rings, ultimately reducing the number of Thon rings in the power spectrum, thus reducing the accuracy of the defocus estimate. To reduce the negative interference between regions with different defoci (i.e. to reduce the effect of the defocus ramp), we exclude from the calculation of the power spectrum the regions that are "too high" or "too low" (defined by `deltaZtolerance`) from the tilt axis. So, if the tilt axis is parallel to the y axis, the selected coordinates should satisfy:

$$-\texttt{deltaZtolerance} < x\tan(\boldsymbol{\alpha}) < +\texttt{deltaZtolerance} \tag{4}$$

where $\boldsymbol{\alpha}$ is the tilt angle and the x coordinates are centered, meaning that the center of the axis (i.e. where the tilt axis is) is equal to 0.

For each image, the tiles within the selected regions are extracted, padded by a factor of 2 and the average 2D power spectrum of these tiles $|\boldsymbol{W}_{exp}(q_{hk}, \phi)|$ is calculated. By summing all of the average 2D power spectra of the tilt-series, we now have one 2D average power spectrum gathering the signal of only the regions "close" to the tilt axis in z.

16.5.3 Average defocus

This first search consists into finding a first rough estimate of the average defocus of the tilt-series, at the tilt axis. For now, we will ignore the astigmatism and compute the radial average of $|\boldsymbol{W}_{exp}(q_{hk}, \phi)|$. We refer to this radial average as $|\boldsymbol{W}_{exp}(q_h)|$. In order to find the defocus value $\mathbf{z}$, we are going to fit a theoretical CTF curve against this radial average. To do so, a range of defocus value is going to be tested, such as for each i^{th} defocus $\mathbf{z}_i = \texttt{defEstimate} \pm \texttt{defWindow}$, 1nm increment:

1. **Get the CTF**: Calculate the 1D theoretical CTF, $\boldsymbol{W}_i(q_h)$, in 1/Å. Note that the defocus $\mathbf{z}_i$ is the only variable from one iteration to another.

$$\lambda = 1226.39 \times 10^{-2} / \sqrt{\texttt{VOLTAGE} + 0.97845 \times 10^{-6} \times \texttt{VOLTAGE}^2}$$

$$E(q_h) = \frac{e^{-20\left(q_h \times \texttt{PIXEL\_SIZE} \times 10^{10}\right)^{1.25}} + 0.1}{1.1}$$

$$\boldsymbol{W}_i(q_h) = \sin\left(\frac{\pi}{2}\ \texttt{Cs}\ \lambda^3\ q_h^4 + \pi\ \lambda\ \mathbf{z}_i\ q_h^2 - \texttt{AMPCONT}\right)\ E(q_h) \tag{5}$$

where λ is the relativistic wavelength of the electrons, in Å. E is an ad-hoc envelope, in 1/Å, down-weighting higher frequencies. `Cs` is the spherical aberration constant, in Å.

2. **Background estimate**: if $\mathbf{z}_i$ is correct, the zeros of $|\boldsymbol{W}_i(q_h)|$ should be at the background level. Therefore, we extract the values of $|\boldsymbol{W}_{exp}(q_h)|$ at the q_h frequencies where the zeros should be, i.e. where $\boldsymbol{W}_i(q_h) = 0$. A smooth curve is fitted to these sampled points and used as the background estimate $\boldsymbol{B}_{exp}(q_h)$. We refer to the background-subtracted radial average as $|\boldsymbol{W}_{exp-B}(q_h)|$. The position of the zeros varies depending on the defocus, therefore the closer we get from the true defocus, the more accurate the background estimate, the greater the correlation between $|\boldsymbol{W}_i(q_h)|$ and $|\boldsymbol{W}_{exp-B}(q_h)|$.

3. **Calculate the CC score**: Only the frequencies, referred as $q_{h'}$, from slightly before the first zero to the first zero after `defCutoff` are considered. The CTF curves are normalized and the normalized cross-correlation is calculated as follow:

$$\mathbf{CC}_i = \frac{\sum_{q_{h'}=1}^{Q_{h'}} |\boldsymbol{W}_i(q_{h'})| \; |\boldsymbol{W}_{exp-B}(q_{h'})|}{Q_{h'} \; \boldsymbol{\sigma}_i \; \boldsymbol{\sigma}_{exp-B}} \tag{6}$$

where $Q_{h'}$ is the number of total $q_{h'}$ frequencies. $\boldsymbol{\sigma}_i$ and $\boldsymbol{\sigma}_{exp-B}$ are the standard deviations of $|\boldsymbol{W}_i|$ and $|\boldsymbol{W}_{exp-B}|$ within the $q_{h'}$ frequencies.

The selected defocus, $\mathbf{z}_{best}$, is the defocus that gave the best fit, i.e. the maximal $\mathbf{CC}_i$. To help analysing the results, the following plots are saved:

- `fixedStacks/ctf/<prefix>_ali1_ccFIT.pdf`: every $\mathbf{CC}_i$ for every sampled $\mathbf{z}_i$.
- `fixedStacks/ctf/<prefix>_ali1_bgFit.pdf`: the radial average (before background subtraction) $|\boldsymbol{W}_{exp}|$ and the estimated background $\boldsymbol{B}_{exp}$ computed using $\mathbf{z}_{best}$, as a function of $q_{h'}$ frequencies. For visualization, the background curve is slightly shifted down.
- `fixedStacks/ctf/<prefix>_ali1_psRadial_1.pdf`: the background-subtracted radial average $|\boldsymbol{W}_{exp-B}|$ (in black) and the theoretical CTF $|\boldsymbol{W}_{best}|$ (in gree), that gave the best fit, as a function of $q_{h'}$ frequencies.

16.5.4 Average astigmatic defocus

Until now, we have used the 1D radial average of the 2D power spectrum, $|\boldsymbol{W}_{exp}(q_h)|$, to estimate the defocus. In order to account for astigmatism, we will now use the 2D power spectrum, $|\boldsymbol{W}_{exp}(q_{hk}, \phi)|$, computed in section 16.5.2.

One difference from the initial search is that we'll use the same background throughout the search. As there could be astigmatism in the images, this is less precise, but it is enough for now and it will be refined later anyway. We do as follow:

1. **Background estimate**: Using $\mathbf{z}_{best}$, calculated at the previous step, we compute $\boldsymbol{W}_{best}(q_h)$, as in equation 5. Then, we extract from $|\boldsymbol{W}_{exp}(q_{hk})|$ 18 lines positioned at different angles (from 2.5° to 87.5°, 5° increment). For each of the 18 lines, we extract the CTF values of $|\boldsymbol{W}_{exp}|$ at the q_{hk} frequencies where the zeros should be, i.e. where $|\boldsymbol{W}_{best}(q_h)| = 0$ and fit a smooth curve along these values. These curves are averaged and a 2D cubic spline is fitted to this average. This is the 2D background estimate, $\boldsymbol{B}_{exp}(q_{hk})$ and we refer to the background-subtracted power spectrum as $|\boldsymbol{W}_{exp-B}(q_{hk})|$.

2. $|\boldsymbol{W}_{exp}|$ and $|\boldsymbol{W}_{exp-B}|$ are saved as `<prefix>_avgPS.fixed` and `<prefix>_avgPS-bgSub.fixed`, in `fixedStacks/ctf`. Only the $q_{h'k'}$ frequencies are shown, as they are the only ones to be considered during the search.

Once the 2D-background-subtracted power spectrum is calculated, we can start the defocus search. The defocus value $\mathbf{z}_{best}$ will stay unchanged during this search. The goal here is to find the defocus shift $\mathbf{\Delta z}_{ast}$ and the astigmatic angle $\boldsymbol{\phi}_{ast}$. For this search, $\mathbf{\Delta z}_{ast}$ is a range of i values, from -200nm to 200nm, with 15nm steps and $\boldsymbol{\phi}_{ast}$ is a range of j values, from -45° to 45°, with 10° step:

1. **Get the CTF**: Calculate the 2D CTF. This is similar to equation 5, but now the defocus varies with the direction of the azimuthal angle ϕ, such as:

$$\mathbf{z}_{i,j}(\phi) = (\mathbf{z}_{best} - [\mathbf{\Delta z}_{ast}]_i)\cos(\phi - [\boldsymbol{\phi}_{ast}]_j) + (\mathbf{z}_{best} + [\mathbf{\Delta z}_{ast}]_i)\sin(\phi - [\boldsymbol{\phi}_{ast}]_j)$$
$$\boldsymbol{W}_{i,j}(q_{hk}, \phi) = \sin\left(\frac{\pi}{2}\ \text{Cs}\ \lambda^3\ q_{hk}^4 + \pi\ \lambda\ \mathbf{z}_{i,j}(\phi)\ q_{hk}^2 - \texttt{AMPCONT}\right)\ E(q_{hk}) \tag{7}$$

2. **Calculate the CC score**: Only the frequencies, referred as $q_{h'k'}$, from slightly before the first zero to the first zero after `defCutoff` are considered. The CTF curves are normalized and the normalized cross-correlation is calculated as follow:

$$\mathbf{CC}_{i,j} = \frac{\sum_{q_{h'k'}=1}^{Q_{h'k'}} |\boldsymbol{W}_{i,j}(q_{h'k'}, \phi)|\ |\boldsymbol{W}_{exp-B}(q_{h'k'}, \phi)|}{Q_{h'k'}\ \boldsymbol{\sigma}_{i,j}\ \boldsymbol{\sigma}_{exp-B}} \tag{8}$$

where $Q_{h'k'}$ is the number of total $q_{h'k'}$ frequencies. $\boldsymbol{\sigma}_{i,j}$ and $\boldsymbol{\sigma}_{exp-B}$ are the standard deviations of $|\boldsymbol{w}_{i,j}|$ and $|\boldsymbol{w}_{exp-B}|$ within the $q_{h'k'}$ frequencies.

The astigmatic defocus, referred as ($\mathbf{z}_{best}$, $\mathbf{\Delta z}_{best}$, $\boldsymbol{\phi}_{best}$), is the defocus that gave the best fit, i.e. the maximal $\mathbf{CC}_{i,j}$ score.

Another defocus search is done on the power spectrum, with a finer sampling around the current best defocus. $\mathbf{\Delta z}_{ast}$ is now a range of i' values, from $\mathbf{\Delta z}_{best}-$ 7.5nm to $\mathbf{\Delta z}_{best}+$ 7.5nm, with 3.75nm steps and $\boldsymbol{\phi}_{ast}$ is now a range of j' values, from $\boldsymbol{\phi}_{best}-$ 5° to $\boldsymbol{\phi}_{best}+$ 5°, with 0.5° step. The refined astigmatic defocus, referred as ($\mathbf{z}_{best}$, $\mathbf{\Delta z}_{best'}$, $\boldsymbol{\phi}_{best'}$), is the defocus that gave the best fit, i.e. the maximal $\mathbf{CC}_{i',j'}$ score.

16.5.5 Handedness check

By convention, the defocus is a positive value. As such, if the sample is tilted, the regions higher than the tilt axis have a smaller defocus and the regions lower than the tilt axis have a larger defocus. If it is the opposite, it means the projections were flipped or the tilt-angles are not in the correct order or the tilt axis used for alignment is 180° off. In section 16.5.2, we have selected regions at about the same defocus than the tilt axis (± `deltaZtolerance`) to reduce interference in the radial average between regions with "significantly" different defoci. Here, we do the same, but we apply a z shift to only select regions that are significantly above the tilt axis ($-$`zShift`), referred as $|\boldsymbol{W}_{-z}|$, or significantly below the tilt axis ($+$`zShift`), referred as $|\boldsymbol{W}_{+z}|$.

As described in section 16.5.3, we have an estimate of the defocus value at the tilt-axis, referred as $\mathbf{z}_{best}$, but we can also estimate the defocus for both $|\boldsymbol{W}_{-z}|$ and $|\boldsymbol{W}_{+z}|$. As such, we calculate the defocus estimate of the regions of the specimen that were imaged while being above the tilt axis (or at least we assume so), referred as $\mathbf{z}_{-z}$, and a defocus estimate of the regions of the specimen that were imaged while being below the tilt axis (again, we assume so), referred as $\mathbf{z}_{+z}$. As in section 16.5.3, the fits are saved in `fixedStacks/ctf/<prefix>_ali1_psRadial_2.pdf` and `fixedStacks/ctf/<prefix>_ali1_psRadial_3.pdf`, respectively.

The regions below the tilt axis should be farther away from the focus compared to the regions above the tilt axis, thus if $\mathbf{z}_{-z} < \mathbf{z}_{best} < \mathbf{z}_{+z}$, then the handedness is probably correct. On the other hand, if $\mathbf{z}_{-z} > \mathbf{z}_{best} > \mathbf{z}_{+z}$, the handedness is probably wrong.

16.5.6 Per-view astigmatic defocus

So far, every 2D power spectra and 1D radial averages were the result of an average across the entire tilt-series. Therefore, we could only estimate an average defocus and average astigmatism. In practice, each view can be at a different defocus and have a different astigmatism. Consequently, we should work on one projection at a time.

We have to deal mostly with tilted images, which inherently have less Thon rings than non-tilted images, making the CTF fitting more complicated. Moreover, there is considerably less signal in one single image than in the entire tilt-series. As such, excluding regions with a “significantly” different defocus, as in section 16.5.2, is not possible.

Fortunately, if an image is tilted with a known tilt, we can account for its defocus ramp and correct for it. By stretching/compressing the power spectrum of different sub-regions (i.e. tiles) of a tilted image, we can make them all constructively interfere with each other. As a result, the analysis of tilted specimens becomes almost as good as for non-tilted specimens, up to the point where the signal is getting weaker due to the increased thickness.

Padding an image in real space is effectively stretching its power spectrum. On the other hand, cropping an image in real space is effectively compressing its power spectrum. As such, with the correct padding/cropping factor, one can make two power spectra with different defoci constructively interfere. The tiles below the tilt axis have a larger defocus, so to make their power spectrum constructively interfere with the tiles at the tilt axis, it is likely that we'll need to pad them in order to stretch their power spectrum. On the other hand, the tiles above the tilt axis have a smaller defocus, so we'll need to crop the tiles in order to compress their power spectrum.

To prevent loosing some information during cropping, all the tiles are padded in advance to `paddedSize` (default=768), to make sure the cropping will not intrude actual data. As such, the goal of this section is to find, for each view i and for each tile j, the padding factor $\boldsymbol{f}_{i,j}$, which, by stretching the power spectrum of the tile, maximizes the constructive interference between the power spectrum of the tile and the power spectrum at the tilt-axis. The CTF at the tilt-axis, $\boldsymbol{W}_{tilt}$, is defined by $\mathbf{z}_{best}$, the average defocus at the tilt-axis calculated in section 16.5.3. The output size of the tile j from view i, is defined as:

$$\boldsymbol{X}_{i,j} = \text{floor}(\texttt{paddedSize} \times \boldsymbol{f}_{i,j}) \tag{9}$$

> ***Note 22:*** *If the* `PIXEL_SIZE` *is lower than 2Å, the Thon rings start to be compressed in a very small frequency window, which makes the analysis more complicated. In this case, we Fourier crop the tilt-series at 2Å, effectively stretching the power spectrum of the image and making the CTF fitting easier.*

Each image is divided into overlapping tiles. For each view i of the tilt-series and for each tile j:

1. **Defocus shift**: We can calculate the defocus shift $\mathbf{\Delta z}_{i,j}$ between a tile and the tilt-axis:

$$\mathbf{\Delta z}_{i,j} = -x_j \tan(\boldsymbol{\alpha}_i) \tag{10}$$

   where $\boldsymbol{\alpha}_i$ is the tilt-angle, x_j is the center of the tile in x. The x coordinates are centered, meaning that the position of the tilt-axis in x is 0. Of course, this assumes that the tilt-axis is parallel to the y axis and that the defocus is a positive number. The defocus of the tile is defined as $\mathbf{z}_{i,j} = \mathbf{z}_{best} + \mathbf{\Delta z}_{i,j}$ and we can calculate the CTF of the tile, $\boldsymbol{w}_{i,j}(q_h)$, as in equation 5.

2. **Padding factor estimate**: If we know $\mathbf{\Delta z}_{i,j}$ and $\mathbf{z}_{best}$, we can calculate a first estimate of the padding factor $\boldsymbol{f}_{i,j}$. One can note that if the tilt-angle is 0°, $\boldsymbol{f}_{i,j} = 1$ (i.e no padding).

$$\boldsymbol{f}_{i,j} = \sqrt{\left(1 + \frac{\mathbf{\Delta z}_{i,j}}{\mathbf{z}_{best}}\right)} \tag{11}$$

3. **Padding factor refinement**: At this point, $\boldsymbol{f}_{i,j}$ is just an estimate. The goal of this step is to refine $\boldsymbol{f}_{i,j}$ only based on how good the fit is between $\boldsymbol{W}_{i,j}(q_h)$ and $\boldsymbol{W}_{tilt}(q_h)$. As such, for each $\boldsymbol{f}_{i,j,k}$, within $\boldsymbol{f}_{i,j} - 1$ up to $\boldsymbol{f}_{i,j} + 1$, with a 0.001 k step:

   (a) Stretch $\boldsymbol{W}_{i,j}(q_h)$ by a factor of $\boldsymbol{f}_{i,j,k}$, using linear interpolation.

   (b) Calculate the normalized cross-correlation $\mathbf{CC}_{i,j,k}$ between the stretched $\boldsymbol{W}_{i,j,k}(q_h)$ and $\boldsymbol{W0}_{tilt}(q_h)$.

   The padding factor that gave the best fit, $[\boldsymbol{f}_{best}]_{i,j}$, is selected and further refined within $[\boldsymbol{f}_{best}]_{i,j} - 0.01$ up to $[\boldsymbol{f}_{best}]_{i,j} + 0.01$, with a 0.0001 k' step. The final padding factor is referred to as $[\boldsymbol{f}_{best'}]_{i,j}$

Each tile of each view is then padded, in real space, using their $[\boldsymbol{f}_{best'}]_{i,j}$, as in equation 9. For each view, the average power spectrum of the padded tiles is calculated and stacked into `fixedStacks/ctf/<prefix>_ali1-PS.mrc`. These power spectra are band-pass filtered (saved as `*ali1-PS2.mrc`), as discussed in section 16.5.3, to only include the frequencies from slightly before the first zero to the first zero after `defCutoff`.

Finally, these band-passed power spectra are sent to CTFFIND 4 for analysis, resulting into a per-view defocus value $\mathbf{z}_i$, a per-view astigmatic defocus shift $[\mathbf{\Delta z}_{ast}]_i$ and a per-view astigmatic angle $[\boldsymbol{\phi}_{ast}]_i$. The outputs for each view i is saved in the following table.

Table 14: `fixedStacks/ctf/<prefix>_ali*_ctf.tlt`

C	Description	C	Description	C	Description
1	index i	9	$\boldsymbol{R}_{1,2,i}$	17	`Cs`
2	$\boldsymbol{T}_{x,i}$ (in pixels)	10	$\boldsymbol{R}_{2,2,i}$	18	`WAVELENGTH`
3	$\boldsymbol{T}_{y,i}$ (in pixels)	11	i^{th} post-exposure	19	`AMPCONT`
4	$\boldsymbol{\alpha}_i$	12	$[\mathbf{\Delta z}_{ast}]_i$	20	pixels in X
5	empty	13	$[\boldsymbol{\phi}_{ast}]_i$	21	pixels in Y
6	90°	14	empty	22	sections Z
7	$\boldsymbol{R}_{1,1,i}$	15	$\mathbf{z}_i$	23	?
8	$\boldsymbol{R}_{2,1,i}$	16	`PIXEL_SIZE`	24	

16.6 Template matching

16.6.1 Pre-processing the tomogram

The desired sub-region tomogram $\boldsymbol{V}$ is reconstructed by weighted back-projection using tilt.

1. **Get the aligned stack**: The aligned stack saved in `aliStacks` is loaded and binned to the desired sampling (i.e. `Tmp_sampling`) as described in section 16.7.1. If the stack already exists in `cache`, it is not recalculated.
2. **Reconstruct the subregion tomogram**: The sub-region coordinates are extracted from table 5 and used to set the tilt entries `SLICE`, `THICKNESS` and `SHIFT`. The tilt angles saved in table 14 are used for the `-TILTFILE` entry. If there are local alignments, the `.local` file is used for the `-LOCALFILE` entry. The output reconstructions from tilt are oriented with the y axis in the third dimension. With trimvol `-rx` entry, we rotate by -90° around x to place the z axis in the third dimension.

   Note 23: *This step is a simpler version of the reconstruction described in section* ***16.7****.*

If the sub-region is larger than `Tmp_targetSize`, it is divided into c equal chunks. For each chunk c:

1. **Band-pass filter**: The chunk $\boldsymbol{V}_c$ is band-pass filtered by $\boldsymbol{W}_{bandpass}$, which has a high-pass cutoff at 600Å to remove ice/intensity gradients and a low-pass cutoff a `lowResCut` or if it is not defined a cutoff at the first CTF zero. The zero is estimated based on the average defocus value of the stack (table 14). The chunk is then centered and standardized.

   Note 24: *If* `Tmp_medianFilter` *is defined, the chunk* $\boldsymbol{V}_c$ *is median filtered using the specified neighborhood window.*
2. **Positive contrast**: The chunk is low-pass filtered and we assume that `lowResCut` cuts before the first zero of the CTF. As such, the negative contrast is "flipped" in real-space by simply multiplying the chunk by -1.

Finally, the variance of $\boldsymbol{V}$ is calculated and used to normalize each chunk $\boldsymbol{V}_c$.

16.6.2 Pre-processing the template

The template $\boldsymbol{S}$ is loaded, padded up to $2 \times$ `Ali_mRadius` while enforcing a squared box. It is then centered, standardized and finally resampled to `Tmp_sampling`, using linear interpolation.

16.6.3 Angular search

The in- and out-of-plane angles $[\Theta_{out},\ \Delta_{out}, \Theta_{in},\ \Delta_{in}]$ registered in `Tmp_angleSearch` are converted into a set of $r \times 3$ Euler angles $(\phi_r,\ \theta_r,\ \psi_r)$. These rotations are finally converted into r rotation matrices $\boldsymbol{R_r}$.

To keep track of things, two empty volumes of the same size as the chunks are prepared, for each chunk. $\mathbf{CC}_{best\text{-}peak}$ will store the standardized cross-correlation scores and $\mathbf{CC}_{best\text{-}rot}$ will store the index r.

For each chunk c and for each rotation r:

1. **Rotate and pad the template**: The template is rotated by $\boldsymbol{R}_r$, padded to the size of $\boldsymbol{V}_c$, band-pass filtered with $\boldsymbol{W}_{bandpass}$ and any change in power due to interpolation is corrected. We refer to this transformed template as $\boldsymbol{S}_r$.

2. **Calculate the normalized cross-correlation**: The cross-correlation between the tomogram chunk $\boldsymbol{V}_c$ and the rotated template $\boldsymbol{S}_r$ is calculated as follow.

$$\mathbf{CC}_{c,r} = \mathcal{F}^{-1}\left\{\mathcal{F}\left\{\boldsymbol{V}_c\right\}\overline{\mathcal{F}\left\{\boldsymbol{S}_r\right\}}\right\} \tag{12}$$

   Importantly, $\mathbf{CC}_{c,r}$ is then normalized by its standard deviation.

3. **Update the best score**: We only want to save the best peaks, i.e. the peaks that are higher than the previous iterations. As such, each voxel v of $[\mathbf{CC}_{\textit{best-peak}}]_c$ and $[\mathbf{CC}_{\textit{best-rot}}]_c$ are updated as follow:

$$[\mathbf{CC}_{\textit{best-peak}}(v)]_c = \begin{cases} \mathbf{CC}_{c,r}(v), & \text{if } [\mathbf{CC}_{\textit{best-peak}}]_c(v) \leqslant \mathbf{CC}_{c,r}(v) \\ [\mathbf{CC}_{\textit{best-peak}}]_c(v), & \text{otherwise} \end{cases} \tag{13}$$

$$[\mathbf{CC}_{\textit{best-rot}}]_c(v) = \begin{cases} r, & \text{if } [\mathbf{CC}_{\textit{best-peak}}]_c(v) \leqslant \mathbf{CC}_{c,r}(v) \\ [\mathbf{CC}_{\textit{best-rot}}]_c(v), & \text{otherwise} \end{cases} \tag{14}$$

   Consequently, each voxel v is assigned to the current best CC score and to the rotation r that gave this score.

   Note 25: *Of course, if it is the first iteration, i.e. if $r = 1$, then $[\mathbf{CC}_{\textit{best-peak}}]_c = \mathbf{CC}_{c,r}$ and $[\mathbf{CC}_{\textit{best-rot}}]_c = 1$.*

At the end of the angular search, the chunks are concatenated and the following files are saved in `convmap_wedgeType_2_bin<X>` (`<X>` is equal to `Tmp_sampling`):

- `<prefix>_<region>_bin<X>_convmap.mrc`: The best scores, i.e. $\mathbf{CC}_{\textit{best-peak}}$.
- `<prefix>_<region>_bin<X>_angles.mrc`: The corresponding rotation r for each best score, i.e. $\mathbf{CC}_{\textit{best-rot}}$.
- `<prefix>_<region>_bin<X>_angles.list`: The $(\phi,\ \theta,\ \psi)$ Euler angles corresponding to the rotation r. One trio per line, r lines.

16.6.4 Extract the peaks

The goal now is to select the $x,\ y,\ z$ coordinates of the p strongest peaks registered $\mathbf{CC}_{\textit{best-peak}}$, with p equal to `Tmp_threshold`. Then, for each peak, the corresponding rotation r is extracted from $\mathbf{CC}_{\textit{best-rot}}$ and converted back to the corresponding $(\phi,\ \theta,\ \psi)$ Euler angles. Therefore, for each desired peak p:

1. **Get the coordinates**: The $x,\ y,\ z$ coordinates $[\boldsymbol{T}_{x,y,z}]_p$ of the strongest peak registered in $\mathbf{CC}_{\textit{best-peak}}$ are selected. To take into account the neighbouring pixels, these coordinates are adjusted by the local center of mass ($3 \times 3 \times 3$ matrix, centered on the strongest peak). $[\boldsymbol{T}_{x,y,z}]_p$ is relative to the subregion tomogram $\boldsymbol{V}$, with the origin at the lower left corner and is unbinned.

2. **Get the rotation**: The value of $\mathrm{CC}_{best\text{-}rot}$ at the coordinates $[\boldsymbol{T}_{x,y,z}]_p$ is extracted. This value is the rotation index r and is converted back to the corresponding $(\phi_p,\ \theta_p,\ \psi_p)$ Euler angles and rotation matrix $\boldsymbol{R}_p$. If a symmetry was entered, the rotation is randomized between the symmetry related pairs to reduce missing-wedge bias.

3. **Get the CC score**: The value of the peak in $\mathrm{CC}_{best\text{-}peak}$ at position $[\boldsymbol{T}_{x,y,z}]_p$ is extracted, centered and standardized. This score is referred as CC_p.

4. **Erase the selected peak**: The selected peak at position $[\boldsymbol{T}_{x,y,z}]_p$ and its neighbouring voxels are masked-out by $\boldsymbol{M}_{peak}$. $\boldsymbol{M}_{peak}$ is set by `Peak_mType` and `particleRadius` (or `Peak_mRadius` if it is defined), and is rotated by $\boldsymbol{R}_p$ before being applied. In that way, the next iteration cannot select the same peak nor the peaks within this particle radius.

5. **Save the peak information in table 15**.

Finally, the coordinates $\boldsymbol{T}_{x,y,z}$ are binned to `Tmp_sampling` and saved into `<prefix>_<region>_bin<X>.pos` and converted into an IMOD mod file with the following command:

```
point2model -number 1 -sphere 3 -scat *.pos  *.mod
```

Table 15: `convmap_wedgeType_2_bin<X>/<prefix>_<region>_bin<X>.csv`. One line per particle p. The translations ($\boldsymbol{T}_x$, $\boldsymbol{T}_y$, $\boldsymbol{T}_z$) are in pixel, un-binned. The Euler angles (ϕ, θ, ψ) are described in section 16.2. They are actually not directly used by emClarity. As mentioned previously, the rotation matrices ($\boldsymbol{R}_{m,n}$, $m =$ rows, $n =$ columns) are meant to be applied to the particles to rotate them from the microscope frame to the reference frame. In this case, the translations are applied before the rotation.

C	Description	C	Description	C	Description	C	Description
1	CC_p	8	empty (1)	15	θ_p	22	$[\boldsymbol{R}_{32}]_p$
2	`Tmp_sampling`	9	empty (1)	16	ψ_p	23	$[\boldsymbol{R}_{13}]_p$
3	empty (0)	10	empty (0)	17	$[\boldsymbol{R}_{11}]_p$	24	$[\boldsymbol{R}_{23}]_p$
4	Unique ID, p	11	$[\boldsymbol{T}_x]_p$	18	$[\boldsymbol{R}_{21}]_p$	25	$[\boldsymbol{R}_{33}]_p$
5	empty (1)	12	$[\boldsymbol{T}_y]_p$	19	$[\boldsymbol{R}_{31}]_p$	26	Class (1)
6	empty (1)	13	$[\boldsymbol{T}_z]_p$	20	$[\boldsymbol{R}_{12}]_p$		
7	empty (1)	14	ϕ_p	21	$[\boldsymbol{R}_{22}]_p$		

16.7 Tomogram reconstruction

16.7.1 Resample the tilt-series

For efficiency and practicality, we often start working with "binned" data and progressively decrease the "binning", up to the point where the original data is used. Binning consists into reducing images by an integer multiple to ensure that every output pixel is an average of the same number of neighbouring pixels (the "bin"). There are many ways to bin an image: box filter, pixel mixing, Fourier cropping, interpolation, etc. Currently, emClarity bins the tilt-series by resampling them in a new, smaller, coordinate grid (in Fourier space), using bilinear interpolation.

Therefore, the original aligned tilt-series, saved in `aliStacks` and registered in the metadata during the project initialization (section 9), are resampled, in parallel, as follow:

1. TODO: $1/\mathrm{sinc}(gX)^2$.
2. The projections are Fourier transformed and band-pass filtered. Low frequencies are removed up to ∼600Å, which sets the mean of the projections to 0 and attenuates eventual large intensity gradients present in the images. Frequencies after the new binned pixel size $\boldsymbol{p}_{bin}$ are also removed. $\boldsymbol{p}_{bin}$ is equal to `PIXEL_SIZE` × `Ali_mSamplingRate`.
3. When resizing images, we force the output image to fit into a new grid. This operation ultimately defines a new center in pixel space and it is important to keep this new center aligned with the center of the original image, so that any operation done on the binned data can be scaled back to the original data. Depending on the binning factor and image size, we can anticipate and shift the images before scaling, to keep the new center aligned with the original grid.
4. The projection spectra are scaled by $\boldsymbol{p}_{bin}$ using bilinear interpolation and the inverse Fourier transform is calculated to switch the images back to real space. Remember that scaling the frequency spectrum of an image by n is equivalent to scaling the image by $1/n$.
5. At this point, the images are "zoomed out", but the actual size of the images are unchanged. To complete the process of resizing the image, we need to crop them to the desired binned size.

The binned stacks are saved in the `cache` directory as `<prefix>_<ali1>_bin<nb>.fixed`.

16.7.2 Defocus step

Defocus-gradient corrected back-projection, as described in [6], requires that "during the reconstruction of tomogram by [weighted] back-projection, each voxel is calculated from tilted images that were CTF-corrected with defocus values corresponding to the position of that voxel at each tilt. To achieve this, each image in the tilt-series is CTF-corrected multiple times with different defocus value. The number of different CTF corrections performed per image depends upon how finely the defocus gradient should be sampled" [7].

> ***Note 26:*** *During tilt-series alignment, the tilt axis is aligned to the y axis. As such, if we assume that the specimen is flat, the defocus only varies along the x and z axis.*

In practice, the tomograms are divided into s z-sections of equal width, also referred as z-slabs or simply sections. Each section is assigned to a CTF-corrected tilt-series with a defocus corresponding to the defocus at the center of the section. The sampling of the defocus gradient, set by the defocus step $\mathbf{\Delta z}$, is defined to keep the average CTF amplitudes, resulting from the destructive interference between all of the CTFs within a section, above the current resolution target. It is calculated as follow:

1. The specimen thickness $\boldsymbol{t}$ is defined by the z_{min} and z_{max} boundaries of the sub-regions of the current tilt-series. The sub-regions are defined manually, as described in section 7.

2. The goal of this procedure is to progressively decrease $\mathbf{\Delta z}$, up to the point where the z sampling is fine enough to achieve the resolution target. $\mathbf{\Delta z}$ is initially set to the specimen thickness $\boldsymbol{t}$. First, we calculate a theoretical CTF for each z point defined from $-\mathbf{\Delta z}/2$ to $\mathbf{\Delta z}/2$, with $0.1 \times \mathbf{\Delta z}$ increment, and average all of these CTFs together. This gives us an estimate of the average CTF resulting from the interference of many CTFs along a z section. The defocus estimate used to calculate the CTFs is the average of the defoci calculated in section 16.5.6 and stored in table 14.

3. Using this average CTF, we can estimate the maximum resolution $\boldsymbol{h}_{max}$, in 1/Å, that a subtomogram could achieve if we were to use the current $\mathbf{\Delta z}$. $\boldsymbol{h}_{max}$ corresponds to the highest frequency, up to Nyquist, where the average CTF is above 90% of contrast.

4. We define the resolution target as $\boldsymbol{h}_{cut}/2$, where $\boldsymbol{h}_{cut}$ is the frequency cutoff, in 1/Å, defined in equation 27. If it is the first cycle and the half-maps are not reconstructed yet, $\boldsymbol{h}_{cut}$ is set to 40Å. If $\boldsymbol{h}_{max} \geqslant \boldsymbol{h}_{cut}/2$, it indicates that the defocus gradient is probably sampled enough to achieve the resolution target. On the other hand, if $\boldsymbol{h}_{max} < \boldsymbol{h}_{cut}/2$, it indicates that we are likely to benefit from a finer sampling, i.e. a smaller $\mathbf{\Delta z}$. In this case, we decrease the current $\mathbf{\Delta z}$ by 90% and recalculate the average CTF and $\boldsymbol{h}_{max}$. This procedure is repeated until $\boldsymbol{h}_{max} \geqslant \boldsymbol{h}_{cut}$, up to the minimum allowed value $\mathbf{\Delta z} = 10$ nm.

   Note 27: *Since we expect the resolution to improve using this new reconstruction, the resolution of the new reconstruction must be higher than what we currently have. The value $\boldsymbol{h}_{cut}/2$ roughly balances the trade off between achievable resolution and run time during the reconstruction.*

   Note 28: *If $\boldsymbol{h}_{max} \geqslant \boldsymbol{h}_{cut}$ at the first iteration, when $\mathbf{\Delta z}$ is equal to the specimen thickness $\mathbf{t}$, the CTF correction shouldn't be considered really as "3D" as we will only use one z section. On the other hand, it also means that, given the resolution target, the specimen is thin enough to not likely benefit from a "3D" correction.*

Once the defocus step $\mathbf{\Delta z}$ and thickness of the specimen $\boldsymbol{t}$ are calculated, we can define the number s of z-sections, as the closest odd integer from $\boldsymbol{t}$ / $\mathbf{\Delta z}$.

16.7.3 Center-of-mass and spatial model

The CTF estimate, fitted from the power spectra, is considered to be the sum of CTFs from weak phase objects at varying defoci within the field of view. The center-of-mass of the specimen in z, $\mathbf{COM}_z$, where most of the signal comes from, greatly impacts this average CTF, to the point where we can assume that the defocus estimate is the distance from $\mathbf{COM}_z$ to the focal plane.

The z-sections are positioned relative to the center of the reconstruction, $\mathbf{COR}_z$, therefore to calculate their respective defocus and correctly estimate their average CTF, we must adjust the current defocus estimate to match the center of the reconstruction $\mathbf{COR}_z$. In conclusion, we need to know the defocus at the center of the specimen or in other words, the average z-offset $\bar{\boldsymbol{Z}}_{R\text{-}M} = \mathbf{COR}_z - \mathbf{COM}_z$.

Here, we make the assumption that most of the signal comes from the particles, therefore that the defocus estimate is the distance from the center-of-mass of all the subtomograms to the focal plane. As the particle positions are expressed relative to $\mathbf{COR}_z$ (i.e. $z = 0$ at the center of the reconstruction), it becomes very easy to calculate $\bar{\boldsymbol{Z}}_{R\text{-}M}$. Moreover, as we know which particle belongs to which section, we can calculate an offset for each section s. We refer to these offsets as $[\bar{\boldsymbol{Z}}_{R\text{-}M}]_s$.

Of course, the particles within a section are not necessarily into the same z plane, meaning that $[\boldsymbol{Z}_{R\text{-}M}]_s$ varies with the x and y coordinates. To take this into account, we calculate a spatial model describing the z-positions of the particles across the specimen. To do so, we extract the x, y, z coordinates of the particles of every sub-regions of the specimen and, according to their z-position, the particles are assigned to a section. If a section contains more than 6 particles, we fit a quadratic surface to the particle positions. This surface defines the "spatial model" of the section or in other words $[\boldsymbol{Z}_{R\text{-}M}]_s(x, y)$. If there is less than 6 particles, the spatial model is an horizontal plane (it is the average z-position of the particles and is invariant across the section, i.e. $[\bar{\boldsymbol{Z}}_{R\text{-}M}]_s$).

16.7.4 3D-CTF phase correction

For each section s, we are going to calculate one CTF-corrected tilt-series using the section's defocus offset. Therefore, for each section s and for each view of the tilt-series i:

1. **Exposure filter**: Calculate the Fourier transform of the view and multiply it with the exposure filter $[\boldsymbol{W}_{exposure}]_i$, as in [3]. This filter only varies with the tilt angle of the image and therefore is identical for each section.

   Note 29: *To take into account that the tilt-series, and therefore the subtomograms, are exposure filtered, the same filter will be applied to the sampling functions of the particles in section **16.8.2**.*

2. **Microscope frame**: The view is replaced within the microscope frame. To do so, the spatial model $[\boldsymbol{Z}_{R\text{-}M}]_s(x, y)$ is tilted according to the tilt angle of the current image. As described previously, because it is calculated from the z-positions of the particles, the spatial model of the section is already correctly positioned in z.

3. **Defocus ramp**: The microscope frame is then divided into n z-slabs of $\boldsymbol{\Delta}\mathbf{z}$ width, which effectively divides the view (represented by the transformed spatial model $[\boldsymbol{Z}_{R\text{-}M}]_{s,i}$) into n strips parallel to the tilt-axis. This is similar to figure 1.A.right, from [7] and forms a defocus ramp perpendicular to the tilt-axis. The strips follow a defocus ramp, defined as follow:

$$\mathbf{z}_i + z'_{min} - \boldsymbol{\Delta}\mathbf{z} \xrightarrow[up\ to]{+\boldsymbol{\Delta}\mathbf{z}} \mathbf{z}_i + z'_{max} + \boldsymbol{\Delta}\mathbf{z} \tag{15}$$

   where z'_{min} and z'_{max} are the highest and lowest z coordinates of the transformed (i.e tilted) spatial model. $\mathbf{z}_i$ is the defocus value of the current view, saved in table 14.

> ***Note 30:*** *As the spatial model is not necessarily a plane, the strips can be "curved" (i.e. with a variable width along the tilt-axis).*
>
> ***Note 31:*** *For a 0° image, $\mathbf{z}'_{max} + \mathbf{z}'_{min} \leqslant \mathbf{\Delta z}$, because the spatial model only takes into consideration the particles from within a z-section. In other words, the 0° spatial model "fits" into the central slab of $\mathbf{\Delta z}$ width and therefore has only one strip. On the other hand, as the tilt-angle increases, more strip are necessary to fully cover the spatial model.*

4. **CTF multiplication**: At this point, the spatial model is correctly positioned in the microscope frame and divided into n strips. An array is allocated in memory to hold the final CTF-corrected view and it is progressively filled, one strip at a time. For each strip n:
   (a) We calculate the 2D CTF of the current strip. The astigmatic defocus of the strip depends on the spatial model and therefore contains the z-offset of the strip, the z-offset of the section and the z-offset to take into account that $\mathbf{\Delta z}$ is the defocus at the center-of-mass of the specimen and not at the center of the reconstruction.
   (b) A copy of the Fourier transform of the current image is multiplied by this 2D CTF and inverse Fourier transformed. The pixels that belongs to the current strip are extracted from this CTF-multiplied image and added to the pre-allocated output array.

   > ***Note 32:*** *Adjacent strips can slightly overlap by 1 pixels. The values of these pixels are divided by 2 to eliminate this overlapping artefact.*

5. **Save the CTF-corrected stacks**: Once every images of the tilt-series are reconstructed, the stack of CTF-corrected images is temporarily saved in the `cache` directory.

At this point of the procedure, we have calculated one CTF-corrected tilt-series for each z-section of the specimen.

16.7.5 Tomogram reconstructions

For each sub-region, we reconstruct each z-section of the sub-region tomogram independently, using their respective CTF-corrected tilt-series. Once the z-sections are reconstructed, we concatenate them to form the final 3D-CTF corrected sub-region tomograms. For each z-section s of a given specimen:

1. For each sub-region defined for this specimen, we reconstruct with tilt the current section using the section's 3D-CTF corrected tilt-series. The tilt angles saved in table 14 are used for the `-TILTFILE` entry. If there is a `.local` file for this specimen, it will be assigned to the `-LOCALFILE` entry. The cosine stretching is turned-off (`-COSINTERP 0`), with no low-pass filtering `-RADIAL 0.5,0.05`.

   > ***Note 33:*** *The ability to reconstruct only subsets of a bigger volume heavily relies on the `SLICE`, `THICKNESS` and `SHIFT` entries of the tilt program.*

2. The output reconstructions from tilt are oriented with the y axis in the third dimension. With trimvol `-rx` entry, we rotate by -90° around x to place the z axis in the third dimension.

Once all the sections of each sub-region are reconstructed, the z-sections of the same sub-region are stacked together in z with newstack to create the final 3D-CTF corrected sub-region tomograms.

16.8 Subtomogram averaging

For each half-set, this procedure computes the CTF-corrected filtered subtomogram average $\boldsymbol{S}$, via a volume normalized single-particle Wiener (SPW) filter [5]. The goal of this filter is to minimize the reconstruction error of the particle density. Each Fourier coefficient (q_{hkl}) of the unfiltered reconstruction $\boldsymbol{S}_{raw}$ is modulated as follow, to produce the 3D-CTF amplitude corrected filtered reconstruction, $\boldsymbol{S}$.

$$\boldsymbol{S}(q_{hkl}) = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}\left\{\boldsymbol{S}_{raw}\right\}(q_{hkl})}{\boldsymbol{W}^2_{thres}(q_{hkl}) + 1\ /\ \mathbf{SNR}(q_{hkl})}\ \boldsymbol{W}_{\textit{ad-hoc}}(q_{hkl})\right\} \tag{16}$$

where:

- $\boldsymbol{W}_{thres}$ is the total thresholded sampling function, i.e. the sampling function of the reference, calculated in section 16.8.3.
- $\mathbf{SNR}$ is the spectral Signal-to-Noise ratio estimate of $\boldsymbol{S}_{raw}$. It is defined as follow:

$$\mathbf{SNR}(q_{hkl}) = \underbrace{\left(\frac{2\ \mathbf{FSC}_{\nabla cut}(q_{hkl})}{1 - \mathbf{FSC}_{\nabla cut}(q_{hkl})}\right)}_{(1)}\underbrace{\left(\frac{1}{\mathrm{G} \otimes \boldsymbol{W}^2_{thres}(q_{hkl})}\right)}_{(2)} \tag{17}$$

  where:

  - (1) defines the map interpretability (i.e. the signal to noise ratio) found in the unfiltered data, including the noise found in the solvent region. As discussed in [8], it is equal to the square of cosine of the average phase error and is equivalent to the square of the figure-of-merit (FOM) used in X-ray crystallography. It is described from section 16.8.4 to 16.8.7.
  - (2) is a Gaussian smoothed version of the total sampling function. The FOM assumes that the number of Fourier measurements is homogeneous. This condition is not met with non-homogeneously spread subtomogram defoci and with strongly preferred orientations in the data set. To correct for these cases, each Fourier component of the FOM is weighted by the sum of squared CTF values (i.e. $\boldsymbol{W}^2_{thres}(\boldsymbol{q}_{hkl})$), which can be considered as the *effective* number of Fourier measurements.
- $\boldsymbol{W}_{\textit{ad-hoc}}$ is an ad-hoc filter described in 16.8.8. It contains the B-factor sharpening and the Modulation Transfer Function (MTF) of the detector.

16.8.1 Mask and box size

Originally calculated from the desired mask size (`Ali_mRadius` and `Ali_mType`), the box size also includes additional voxels to account for the mask roll-off, the padding factor and an eventual additional padding to optimize the final box size to Fourier transforms.

16.8.2 Sampling functions

The 3D-sampling function (i.e. a weighted 3D-CTF model [9]) describes the extent of information transfer during the imaging process. It is supposed to track all of the systematic changes to the signal imposed by the microscope and image processing algorithms. As such, it should take into account the CTF modulations for each subtomogram in each image of the tilt-series, while accounting for the increasing sample thickness with tilt angle and the accumulated electron dose. Moreover, it should account for any filtering applied to the tilt-series or weighting applied during tomogram reconstruction.

emClarity is currently calculating a simplified model of the sampling function of the particles, referred as $\boldsymbol{w}$, but a per-particle model, including every weights applied during the back-projection, will be added in the future. The current model groups particles from the same tilt-series by dividing the field of view into strips parallel to the tilt-axis. The number of strips, also referred as group, can be changed by the `ctfGroupNumber` parameter and is set by default to 9. During initialization (section 9), each particle is assigned to the group it belongs to, thus, to a given sampling function $\boldsymbol{w}$. Although particles from a given group have the same $\boldsymbol{w}$, their contribution to the total sampling function $\boldsymbol{W}$, i.e. the sampling function of the reference, is different, because they have different orientations in the tomogram.

When reconstructing an image, the microscope multiplies the spectrum of the object by the CTF. Before the tomogram reconstruction, we multiplied again the images by the CTF to correct for the negative contrast (phase correction). At this point, the spectrum of the tomogram is multiplied by the square of CTF. Therefore, to correct for the CTF amplitudes, we will need to divide the reconstruction spectrum by CTF squared. As such, we calculate directly the square of the sampling functions, $\boldsymbol{w}^2$.

In conclusion, emClarity first needs to calculate the sampling function $\boldsymbol{w}^2$ of the particles. To do so, a 3D super-sampled spectrum is filled with i CTFs, once for each view i of the tilt-series. Briefly, for each tilt-series t, for each group g and for each view i:

1. **Get the CTF**: Assuming the tilt-axis is parallel to the y axis and assuming that the defocus is a positive number, we can calculate the defocus value at the center of the strip g, such as:

$$\mathbf{z}_{t,g,i} = \mathbf{z}_{t,i} - x_g \tan(\boldsymbol{\alpha}_{t,i}) \tag{18}$$

   where $\mathbf{z}_{t,i}$ is the defocus value calculated in section 16.5.6. x_g is the center of the group g in the x direction. The x coordinates are centered, meaning that $x = 0$ at the tilt-axis. $\boldsymbol{\alpha}_{t,i}$ is the tilt-angle. Note that the further away in x the group is from the tilt-axis, the more the defocus varies with the tilt-angle. We can then calculate the 2D CTF $[\boldsymbol{w}^2]_{t,g,i}$, as in equation 7, using this defocus value, the defocus astigmatic shift $\boldsymbol{\Delta}\mathbf{z}_{t,i}$ and the astigmatic angle $\boldsymbol{\phi}_{t,i}$, both calculated in section 16.5.6.

2. **z-stretch**: To take into account the thickness of the view, $[\boldsymbol{w}^2]_{t,g,i}$ is stretched in z, thus becoming a volume of 9 slices, with the total weight along the slices kept to 1.

3. **Weighting**: The stretched $[\boldsymbol{w}^2]_{t,g,i}$ is "fraction" filtered in the same way the corresponding view i in the tilt-series was. Indeed, during the alignment of the tilt-series (section 16.5.1), we kept track of the inelastic scattering events by multiplying each view of the tilt-series by its fraction of inelastic $[\boldsymbol{f}_{inelastic}]_{t,i}$ (equation 3). As such, $[\boldsymbol{w}^2]_{t,g,i}$ is also multiplied by its $[\boldsymbol{f}_{inelastic}]_{t,i}$ and similarly by the fraction of dose, if a Saxton dose-scheme was used. Additionally, an exposure

filter $[\boldsymbol{W}_{exposure}]_{t,i}$ was applied to the tilt-series before the tomogram reconstruction (section 16.7.4) and is therefore applied to $[\boldsymbol{w}^2]_{t,g,i}$ as well.

4. **From 2D to 3D**: The z-stretched and weighted $[\boldsymbol{w}^2]_{t,g,i}$ is then rotated by its corresponding tilt-angle $\boldsymbol{\alpha}_{t,i}$, using linear interpolation, and is added to the pre-allocated 3D spectrum $\boldsymbol{w}^2_{t,g}$. Once every "view" i has been added, $\boldsymbol{w}^2_{t,g}$ is finally normalized between 0 and 1.

   Note 34: *By replacing the CTFs in the 3D space, the sampling functions ultimately include the so-called "missing-wedge".*

As such, for each tilt-series t, we calculate g sampling functions, all of which are saved into the same montage `<projectDir>/cache/<prefix>_<bin>.wgt`. These functions are in the microscope frame and will be rotated into the reference frame to calculate the total sampling function $\boldsymbol{W}^2$.

16.8.3 Subtomogram average and total sampling function

This step produces the two $[\boldsymbol{S}_{raw}]_1$ and $[\boldsymbol{S}_{raw}]_2$, i.e the two subtomogram average, and the two $[\boldsymbol{W}_{thres}]_1$ and $[\boldsymbol{W}_{thres}]_2$, i.e. the two total sampling functions, from equation 16. This entire section is repeated for each one of the two half-set.

We first pre-allocate in memory $\boldsymbol{S}_{raw}$ and $\boldsymbol{W}^2$. The following steps consist into filling these empty volumes with every transformed subtomogram and every $\boldsymbol{w}^2$ weight, respectively. For each sub-regions registered into the metadata during `init`, we extract the sub-region coordinates, the particles information (x, y, z coordinates and rotation) and for each particle p:

1. **Particle extraction**: The subtomogram $\boldsymbol{s}_p$ is extracted from its sub-region tomogram. There are a few things worth to mention:

   (a) If a particle, defined by `particleRadius`, is out of the boundaries of its sub-region, it is ignored and will be ignore in the following cycles. The box size is often much larger than the particle, to ensure that all of the delocalized information is available for full CTF restoration. Ideally, this is fully represented in every particle, but as long as the particle itself is in the sub-region, we do not worry if the high resolution information is not there and extract the particle nonetheless.

   (b) The subtomogram coordinates are floats and are likely to have decimals. To preserve the center of the particle, we take the nearest integers to extract the box from the sub-region tomogram and the remaining rounded decimals (i.e. shifts) $[\boldsymbol{T}_{orig}]_p$ will be applied during the subtomogram transformation, at the next step.

2. **Get the particle in the reference frame**: Once extracted, the particle $\boldsymbol{s}_p$ is rotated into the reference frame, shifted by their $[\boldsymbol{T}_{orig}]_p$ shift resulting from the extraction and the symmetry is applied, if any. Each symmetry pair is calculated and averaged with the other ones to produce the symmetrized rotated particle.

3. **Get the sampling function in the reference frame**: The same rotation is applied to the sampling function $\boldsymbol{w}^2_p$ to correctly represent the original sampling of the subtomogram.

4. **CCC Weighting**: If $\boldsymbol{s}_p$ has an alignment score CCC_p below the average score of the entire data-set, $\overline{\mathrm{CCC}}$, both of $\boldsymbol{s}_p$ and $\boldsymbol{w}^2_p$ q_{hkl} Fourier coefficients are down-weighted by the following

filter:

$$\Delta\mathbf{CCC}_p = \cos^{-1}(\mathbf{CCC}_p) - \cos^{-1}(\overline{\mathbf{CCC}})$$
$$[\boldsymbol{w}_{\mathrm{CCC}}]_p(q_{hkl}) = \exp\left(-\left(\texttt{flgQualityWeight} \times \Delta\mathbf{CCC}_p\right)^2 q_{hkl}^2\right) \tag{19}$$

The further away the particle is from the mean, the stronger the filter. It mostly affects high frequencies. This feature is deactivated for the first cycle as it requires CCC scores, which are calculated during the subtomogram alignment procedure (section 12).

5. **Build the reference**: Now that both $\boldsymbol{s}_p$ and $\boldsymbol{w}_p^2$ are in the reference frame, we can add them to the pre-allocated volumes $\boldsymbol{S}_{raw}$ and $\boldsymbol{W}_p^2$, respectively. There are a few things worth to mention:

   - The particle $\boldsymbol{s}_p$ is centered and standardized before being added to the reference. The corners of the box, which can contain non-sampled regions, are also ignored.
   - Particles with an alignment score below `flgCCCcutoff` are ignored. By default, `flgCCCcutoff` is 0 (i.e. no particle ignored).
   - If `flgCutOutVolumes` is 1, the transformed (rotated + shifted) particles are saved to disk in the `cache` directory. Before being saved, the subtomograms are padded by 20 pixels.

The reconstruction $\boldsymbol{S}_{raw}$ is centered, standardized and saved in the project directory as `cycleXXX_<project>_class0_REF_*_NoWgt.mrc`.

As we will discussed in the next sections, our PSNR estimate directly comes from the Fourier Shell Correlation (FSC), making it less reliable for critically under-sampled frequencies. We can use the total sampling function $\boldsymbol{W}^2$, which weights the PSNR, to correct for this effect. To do so, frequencies below a given threshold are up-weighted, which effectively will down-weight these frequencies in the final reconstruction $\boldsymbol{S}$. This "thresholded" sampling function is referred as $\boldsymbol{W}_{thres}^2$ and it is calculated as follow:

1. emClarity defines critically under-sampled frequencies as the frequencies that have a weight less than $0.2 \times \mathrm{median}(\boldsymbol{W}^2)$, excluding frequencies with a value of 0 from the calculation of the median.
2. The new sampling value of the frequencies below the threshold was defined empirically and is equal to:

```
low = 0.1 * max(W2);
threshold = 1.5 * median(W2(W2 > low) - low);
```

3. Frequencies at the "edge" of critically under-sampled regions are slightly up-weighted too, to create a smooth transition between this new threshold value and the rest of the sampling function.
4. This total thresholded sampling function $\boldsymbol{W}_{thres}^2$ is saved in the project directory as `cycleXXX_<project>_class0_REF_*_Wgt.mrc`.

We now have calculated the raw subtomogram average $\boldsymbol{S}_{raw}$ and the total sampling function $\boldsymbol{W}^2_{thres}$, of equation 16. As such, we can already correct for all of the systematic changes in $\boldsymbol{S}_{raw}$ imposed by the microscope and image processing algorithms, such as:

$$\boldsymbol{S}_{corr} = \mathcal{F}^{-1}\left\{\frac{\mathcal{F}\{\boldsymbol{S}_{raw}\}}{\boldsymbol{W}^2_{thres}}\right\} \tag{20}$$

This correction does not take the SNR of the reconstructions into account and even if the total sampling function is thresholded, the noise present in $\boldsymbol{S}_{raw}$ is amplified in $\boldsymbol{S}_{corr}$. The objective of the next few sections consists into finding the best SNR estimate of the $\boldsymbol{S}_{corr}$ half-maps and apply a new correction that does take the SNR into account.

16.8.4 Molecular masks

The SNR can be most accurately estimated from a "masked" FSC, calculated from the two $\boldsymbol{S}_{corr}$ half-maps, where the environs solvent noise is suppressed with soft-edged masks. To avoid introducing spurious correlations between half-sets, two masks are created, one for each half-map, as described below:

1. The CTF-corrected reconstruction, $\boldsymbol{S}_{corr}$, is median filtered to remove "salt and pepper" noise and low-pass filtered to 14Å. The volume is centered and standardized, and the edges of the reconstruction are forced to smoothly go to zero to suppress edge artifacts.

2. The highest intensity voxels are then selected and used as seeds for an iterative dilation. The dilation progressively dilates the seeds toward their neighbour voxels that are above a gradually relaxed intensity threshold. This procedure ensure a connectivity-based expansion.

3. The binary mask formed by the selected voxels after dilation, $\boldsymbol{M}_{bin}$, is Gaussian blurred and saved as $\boldsymbol{M}_{core}$. This mask is considered as the particle volume (non-hydrated), leaving only the strongest part of the densities.

4. Additionally, $\boldsymbol{M}_{bin}$ is radially expanded by at least 10Å, Gaussian blurred and saved as $\boldsymbol{M}_{mol}$. At this point, the $\boldsymbol{M}_{mol}$ mask includes the particle density *and* its surrounding solvent to make sure flexible densities are correctly included in the FSC calculation, thereby preventing overestimation of the resolution.

5. As we'll see, the two $\boldsymbol{S}_{corr}$ spectra are masked by their corresponding $\boldsymbol{M}_{mol}$ mask before computing the FSC. As such, we will need to "scale" the FSC to compensate for the masked-out voxels and the solvent content. The ratio $\boldsymbol{f}_{mask}/\boldsymbol{f}_{particle}$ does exactly that and is applied to the FSC at the next section, in equation 23.

   - $\boldsymbol{f}_{mask}$ is the fractional volume allowed by the $\boldsymbol{M}_{mol}$ mask.
   - $\boldsymbol{f}_{particle}$ is the fractional volume occupied by the particle within the box size. It is the volume allowed by the $\boldsymbol{M}_{bin}$ mask.
   - As $\boldsymbol{M}_{mol}$ is a soft-edged mask and therefore contains values between 0 and 1, and even though the edges almost certainly contain surrounding solvent, we estimate the signal reduction in the soft-edged regions of the mask as they could cut through actual densities. This power reduction is implicitly included in $\boldsymbol{f}_{particle}$.

   As we produce two $\boldsymbol{f}_{mask}/\boldsymbol{f}_{particle}$ ratio (one for each $\boldsymbol{S}_{corr}$), we take the average of both ratio.

16.8.5 Fourier Shell Correlation

Before calculating the FSC between the two half-maps $[\boldsymbol{S}_{corr}]_1$ and $[\boldsymbol{S}_{corr}]_2$, we need to pre-process them:

1. The `Ali_mType` and `Ali_mRadius` parameters are used to compute soft-edged shape mask, $\boldsymbol{M}_{shape}$. In order to constrain the analysis to the desired region, this mask is applied to the molecular masks $\boldsymbol{M}_{core}$ and $\boldsymbol{M}_{mol}$. The soft-edged part of $\boldsymbol{M}_{shape}$ (i.e. the taper) is randomized to reduce the correlation due to the masking during FSC calculation.

2. The reconstructions need to be aligned to each other. To do so, emClarity uses the "Fit in map" procedure of Chimera. As we want to base the alignment on the core of the particle, the averages are temporally masked with $\boldsymbol{M}_{core}$ before calling Chimera. The two masked volumes are saved as `FSC/cycleXXX_<prefix>_Raw-*Ali.mrc`, then emClarity extracts the output rotation $\boldsymbol{R}_{gold}$ and translation $\boldsymbol{T}_{gold}$, and use them to align $[\boldsymbol{S}_{corr}]_1$ to $[\boldsymbol{S}_{corr}]_2$, using spline interpolation.

3. The two reconstructions are centered, standardized, and masked with their respective $\boldsymbol{M}_{mol}$ mask. Both reconstructions are saved as `FSC/fscTmp_1_noFilt_*.mrc`. These volumes are padded to a cube of at least 384 voxels of length, to oversample their Fourier spectrum before the FSC calculation.

Once the reconstructions are aligned and masked, we can calculate the spherical FSC:

1. The two aligned and masked volumes are Fourier transformed and the cross-correlation spectrum is calculated:

$$\mathbf{CC}(q_{hkl}) = \mathcal{F}\left\{[\boldsymbol{S}_{corr}]_1\right\}(q_{hkl})\ \overline{\mathcal{F}\left\{[\boldsymbol{S}_{corr}]_2\right\}}(q_{hkl}) \tag{21}$$

   where $\bar{x}$ is the conjugate of x.

2. Each axis is divided into shells. The number of shells depends on the size of the Fourier spectrum. As the reconstructions are cubic, each shell is an empty spherical mask with a given thickness and delimits the set of $\mathbf{CC}$ scores that will be averaged together. The final FSC curve is nothing more than the concatenation of the normalized $\mathbf{CC}$ scores, such as:

$$\mathbf{FSC}(shell) = \frac{\sum\limits_{q=h',k',l'}^{Q'} \mathbf{CC}(q_{h'k'l'})}{\sqrt{\sum\limits_{q=h',k',l'}^{Q'} \left|\mathcal{F}\left\{[\boldsymbol{S}_{corr}]_1\right\}\right|^2 \times \sum\limits_{q=h',k',l'}^{Q'} \left|\mathcal{F}\left\{[\boldsymbol{S}_{corr}]_2\right\}\right|^2}} \tag{22}$$

   where the $q_{h'k'l'}$ frequencies are the q_{hkl} frequencies that belongs to the given shell. $\boldsymbol{Q'}$ is the number of $q_{h'k'l'}$ frequencies within each shell. The numerator is the cross-correlation score of the shell. The denominator is the "power" of the shell or in other words, the total complex magnitude of the shell. $|x|$ is the complex magnitude of the complex number $x = a + bi$, such as $|x| = \sqrt{a^2 + b^2}$. Note that $|\bar{x}| = |x|$.

Moreover, we can calculate the FSC over conical shells, as opposed to spherical shells, making the SNR estimate via the FSC much more robust to resolution anisotropy [10]. To do so, emClarity subdivide each spherical shell into 37 overlapping cones of 36° of half angle, with a 30° increment between each cone. Then, it recalculates the FSC as described above, but for each cone. We refer to these FSCs as $\mathbf{FSC}_{\nabla}$, where as opposed to equation 22, the normalized cross-correlation is calculated over the $q_{h''k''l''}$ frequencies, which are the frequencies within the current shell *and* within the current cone. Similarly, $\boldsymbol{Q}''$ is the number of $q_{h''k''l''}$ frequencies within each shell of each cone.

Then, the spherical FSC, as well as the 37 conical FSCs, needs to be scaled by $\boldsymbol{f}_{mask}$ and $\boldsymbol{f}_{particle}$. Indeed, as discussed in section 16.8.4, these FSCs are calculated from masked volumes, where the object of interest represents only a fraction of the reconstruction. As such, each shell, whether spherical or conical, of each FSC, is scaled as follow. This scaling is applied up to the first shell that shows no correlation between the two half-maps (i.e. $\mathbf{FSC}(shell) \leqslant 0$).

$$\mathbf{FSC}(shell) = \frac{\mathbf{FSC}(shell) \times (\boldsymbol{f}_{particle}/\boldsymbol{f}_{mask})}{1 + (1 - \boldsymbol{f}_{particle}/\boldsymbol{f}_{mask}) \times |\mathbf{FSC}(shell)|} \tag{23}$$

So far, the FSCs have been a function of shells, i.e $\mathbf{FSC}(shell)$. This is also true for $\boldsymbol{Q}'$ and $\boldsymbol{Q}''$, which can be expressed as a function of spherical or conical shells, respectively. The next sections of this chapter are using an oversampled version of these curves. The oversampled curves are calculated using spline interpolation over 501 h frequency points.

16.8.6 Frequency cutoffs

The FSC directly impacts our SNR estimate, thereby our filtering. Defining a FSC threshold level at which the resolution is reproducible is therefore necessary. The one-bit and half-bit information curves indicate the resolution level at which enough information has been collected for interpretation and will be used to define the frequency cutoffs for filtering the reference.

The one-bit and half-bit curves, $\boldsymbol{T}_{1\text{-}bit}$ and $\boldsymbol{T}_{1/2\text{-}bit}$ respectively, are calculated for the spherical FSC and each of the conical FSCs, as in [11]:

$$\boldsymbol{Q}_e(h) = \boldsymbol{Q}(h) \;/\; (2 \times U) \tag{24}$$

$$\boldsymbol{T}_{1\text{-}bit}(h) = \frac{0.5 + 2.4142 \;/ \sqrt{\boldsymbol{Q}_e(h)}}{1.5 + 1.4142 \;/ \sqrt{\boldsymbol{Q}_e(h)}} \tag{25}$$

$$\boldsymbol{T}_{1/2\text{-}bit}(h) = \frac{0.2077 + 1.9102 \;/ \sqrt{\boldsymbol{Q}_e(h)}}{1.2071 + 0.9102 \;/ \sqrt{\boldsymbol{Q}_e(h)}} \tag{26}$$

As discussed in [11], the FSC is influenced with various factors, notably the number of voxels $\boldsymbol{Q}$ in a given Fourier shell, the number of repeated unit U in the reconstruction and the size of the particle, all of which are taken into account in $\boldsymbol{Q}_e$ (equation 24). Of course, for the spherical FSC, $\boldsymbol{Q}$ corresponds to $\boldsymbol{Q'}$, whereas for the conical FSCs, it corresponds to their respective $\boldsymbol{Q''}$.

For each conical FSCs, we then define a frequency cutoff $\boldsymbol{h}_{cut}$, where $\boldsymbol{h}_{1\text{-}bit}$ and $\boldsymbol{h}_{1/2\text{-}bit}$ are the frequencies at which the FSC first goes below the $\boldsymbol{T}_{1\text{-}bit}$ or $\boldsymbol{T}_{1/2\text{-}bit}$ curves, respectively.

$$\boldsymbol{h}_{cut} = (\boldsymbol{h}_{1\text{-}bit} - \boldsymbol{h}_{1/2\text{-}bit})/2 \tag{27}$$

The frequency cutoff is also calculated for the spherical FSC, as well as the fixed-valued threshold $\boldsymbol{h}_{0.5}$ and $\boldsymbol{h}_{0.143}$. These last two are only used for display and are saved in `FSC/cycleXXX_<project>_Raw-1-fsc_GLD.pdf` or in the corresponding `.txt` file.

16.8.7 Figure-Of-Merit

As discussed in [8], the figure-of-merit (FOM) in equation 17, also referred as $\boldsymbol{C}_{ref}$, defines the resolution criterion that will be used by the Wiener filter as SNR estimate. It is worth noting that "$\boldsymbol{C}_{ref}$ is equal to the cosine of the average phase error and is equivalent to the crystallographic figure-of-merit, a common measure of map interpretability in X-ray crystallography" [8]. $\boldsymbol{C}_{ref}$ is defined as follow:

$$\boldsymbol{C}_{ref}(h) = \sqrt{\frac{2\ \mathbf{FSC}(h)}{1 + \mathbf{FSC}(h)}} \tag{28}$$

To be used by the Wiener filter, $\boldsymbol{C}_{ref}$ needs to be expressed as a function of q_{hkl} frequencies, i.e. it needs to be a volume of the same size as the spectrum about to be filtered. Additionally, $\boldsymbol{C}_{ref}$ should be limited by the frequency cutoffs computed in the previous section.

1. Each one of the 37 conical $\mathbf{FSC}_{\nabla}$ are multiplied with their corresponding frequency cutoff curve. These curves are equal to 1 (no weighting), up to their frequency cutoff $\boldsymbol{h}_{cut}$ where an exponential decay starts and forces the FSC to rapidly drop to zero. The cut-offed conical FSCs are referred as $\mathbf{FSC}_{\nabla cut}$.
2. To express $\boldsymbol{C}_{ref}$ as a function of q_{hkl} frequencies, emClarity computes the "anisotropic 3D FSC". To do so, the 37 overlapping and oversampled conical $\mathbf{FSC}_{\nabla cut}$ are reshaped and expanded back into their corresponding cone in the 3D space, while taking into account that frequencies overlapped with multiple cones are appropriately down-scaled.

Once $\mathbf{FSC}_{\nabla cut}(q_{hkl})$ is calculated, we can define $\boldsymbol{C}_{ref}(q_{hkl})$ as shown below. In practice, as with the total sampling function, we calculate directly $\boldsymbol{C}^2_{ref}$.

$$\boldsymbol{C}_{ref}(q_{hkl}) = \sqrt{\frac{2\ \mathbf{FSC}_{\nabla cut}(q_{hkl})}{1 + \mathbf{FSC}_{\nabla cut}(q_{hkl})}} \tag{29}$$

16.8.8 Volume-normalized SPA Wiener filter

At this point, we have described every component of equation 16, expect the $\boldsymbol{W}_{ad\text{-}hoc}$ weighting function. This function should be thought as an external component of the Wiener filter; it is just an additional ad-hoc filtering that we apply on the final volume. $\boldsymbol{W}_{ad\text{-}hoc}$ contains two components, the global B-factor sharpening and the default Modulation Transfer Function (MTF) of the detector.

$$\boldsymbol{B}_{factor}(q_{hkl}) = \exp\left(\frac{\texttt{Fsc\_bfactor} \times q^2_{hkl}}{4}\right) \tag{30}$$

$$\mathbf{MTF}(q_{hkl}) = \frac{0.13}{\exp\left(-20\ q^{1.25}_{hkl}\right) + 0.13} \tag{31}$$

In equation 16, we clearly show that we divide each Fourier coefficient by some weight, which depends on the sampling and on our SNR estimate. As we have now seen, the SNR estimate is calculated using the CTF-corrected $\boldsymbol{S}_{corr}$ half-maps. Here's a reformulation of equation 16 with $\boldsymbol{S}_{corr}$, as opposed to $\boldsymbol{S}_{raw}$:

$$\mathrm{S}(q_{hkl}) = \mathcal{F}^{-1}\left\{\frac{\boldsymbol{W}_{thres}(q_{hkl})^2}{\boldsymbol{W}_{thres}(q_{hkl})^2 + 1\ /\ \mathbf{SNR}(q_{hkl})}\ \mathcal{F}\left\{\boldsymbol{S}_{corr}\right\}(q_{hkl})\ \boldsymbol{W}_{ad\text{-}hoc}(q_{hkl})\right\} \tag{32}$$

16.9 Subtomogram alignment

This goal of this procedure is to find the best transformation (rotation + translation) between each particle and its reference. By convention, the rotation $\boldsymbol{R}$ and translation $\boldsymbol{T}$ are meant to be applied to the particles to align them into the reference frame.

16.9.1 Masks and filters

The box size is defined as in section 16.8.1 and every volume (particles, references, filters and masks) will be padded/cropped to this size.

- **Masks**: Every voxel outside of the alignment mask $\boldsymbol{M_{ali}}$ is ignored throughout the alignment procedure. $\boldsymbol{M_{ali}}$ is defined by the `Ali_mType`, `Ali_mRadius` and `Ali_mCenter` parameters. $\boldsymbol{M_{particle}}$ is an ellipsoid of radius equal to `particleRadius` and sets the maximum translation allowed, thereby preventing the particles to drift too much from their original position. $\boldsymbol{M_{peak}}$ is an optional rectangular mask of radius `Peak_mRadius` used to further restrict the translations.
- **Filters**: To ignore the information after the current frequency cutoff, the subtomograms are going to be filtered before comparison with the reference. This filter, corresponds to the $\boldsymbol{C_{ref}}$ described in equation 29, but in this case, only the spherical shells and the frequency cutoff $\boldsymbol{h_{cut}}$ from the spherical FSC are used, whereas the conical FSCs were used during the subtomogram averaging. Additionally, the ad-hoc Modulation Transfer Function ($\mathbf{MTF}$, equation 31) will also be applied before comparison with the reference.
- **Sampling functions**: The sampling functions of the particles $\boldsymbol{w}$ should be already calculated and saved (section 16.8.2). If the box size changed between the averaging and the alignment, the sampling functions are calculated again. The sampling functions are created and saved in the microscope frame, thus we'll refer to them as $\boldsymbol{w}_{mic}$. The total sampling functions (i.e. the sampling function of the two half-maps) $\boldsymbol{W}$ are created and saved in the reference frame, thus we will refer referred to them as $\boldsymbol{W}_{ref}$.

16.9.2 References

The references (i.e. the two half-maps) are calculated and saved in the project directory during subtomogram averaging (section 16.8) and are in the reference frame. As such, we will refer to them as $\boldsymbol{S}_{ref}$. These are CTF-corrected and filtered. Before aligning the particles to these reconstructions, some pre-processing is done:

1. **Center-Of-Mass**: The references $\boldsymbol{S}_{ref}$ are masked with $\boldsymbol{M}_{ali}$ and their $\boldsymbol{M}_{core}$ mask (section 16.8.4) is calculated and applied. Then, the x, y, z center-of-mass (COM) of the two masked $\boldsymbol{S}_{ref}$ are calculated and applied to have the center of the box aligned with the COM of the particle averages.
2. **Combine low-resolution information**: We assume that the information from the DC-component (i.e. the zero frequency) to the initial resolution cutoff $\boldsymbol{h}_{cut}$ are "shared" between the two half-maps. As such, this "shared" information is averaged and the "shared" frequencies of both $\boldsymbol{S}_{ref}$ are replaced by this average. The initial resolution cutoff is set to 40Å by default.

16.9.3 Calculate the translation between a particle and a reference

This section can be seen as a function that takes as inputs a subtomogram, $\boldsymbol{s}_{mic}$, which is in the microscope frame, a reference, $\boldsymbol{S}_{ref}$, which is in the reference frame, the particle's rotation $\boldsymbol{R}$ and translation $\boldsymbol{T}$. If the reference and the particle are both in the same frame, how do we calculate the translation $\boldsymbol{T}_{S\text{-}s}$ between them?

1. **Get the reference into the microscope frame**: The reference $\boldsymbol{S}_{ref}$ is rotated by $\boldsymbol{R}^T$, translated by $-\boldsymbol{T}$, centered and standardized. To emphasize that the reference is now in the microscope frame, we will refer to it as $\boldsymbol{S}_{mic}$ from now on.

   Note 35: *By convention, $\boldsymbol{T}$ and $\boldsymbol{R}$ are meant to be applied to the particles and will switch the particles from the microscope to the reference frame. As such, if we want to align the reference to the microscope frame, we must apply the "inverse" transformation, such as $-\boldsymbol{T}$ and $\boldsymbol{R}^T$. In this case, the translations are applied after the rotation.*

2. **Get the total sampling function into the microscope frame**: The total sampling function $\boldsymbol{W}_{ref}$ is rotated by $\boldsymbol{R}^T$ as well. To emphasize that the total sampling function is now in the microscope frame, we will refer to it as $\boldsymbol{W}_{mic}$ from now on.

3. **Calculate the CCC map**: The particle $\boldsymbol{s}^{mic}$ is masked with $\boldsymbol{M}_{particle}$, filtered with $\boldsymbol{C}_{ref}$ and $\mathbf{MTF}$, and finally centered and standardized. Then, the CCC map is calculated as follow:

$$\mathrm{CCC}_{map} = \mathcal{F}^{-1}\left\{\mathcal{F}\{\boldsymbol{s}_{mic}\}\ \boldsymbol{W}_{mic}\ \overline{\mathcal{F}\{\boldsymbol{S}_{mic}\}}\ \boldsymbol{w}_{mic}\right\} \tag{33}$$

4. **Further restrict the allowed translation**: If $\boldsymbol{M}_{peak}$ is defined, apply it to CCC_{map}. This effectively removes every peak outside the allowed region, set by `Peak_mRadius`.

5. **Get highest peak position**: Extract the x, y, z coordinates of the strongest peak, $\boldsymbol{p}_{x,y,z}$. In this case, the origin $(x = y = z = 0)$ corresponds to the center of the CCC map. A 7×7×7 box is extracted around the peak and we calculate the COM of this box, $\mathrm{COM}_{x,y,z}$. The translation $\boldsymbol{T}_{S\text{-}s}$ is then defined as $\boldsymbol{T}_{S\text{-}s} = \boldsymbol{p}_{x,y,z} + \mathrm{COM}_{x,y,z}$.

16.9.4 Calculate the CCC score between a particle and a reference

This section can be seen as a function that takes as inputs a particle, $\boldsymbol{s}_{mic}$, which is in the microscope frame, a reference, $\boldsymbol{S}_{ref}$, which is in the reference frame, the particle's rotation $\boldsymbol{R}$ and translation $\boldsymbol{T}$. If the particle and the reference are both in the same frame, how do we calculate their Constrained Cross-Correlation score, CCC_{score}?

1. **Get the particle into the reference frame**: The particle is translated by $\boldsymbol{T}$ and rotated by $\boldsymbol{R}$. To increase the SNR of the transformed particle, we also apply its symmetry. To emphasize that the particle is now in the reference frame, we will refer to it as $\boldsymbol{s}_{ref}$ from now on.

2. **Get the sampling function of the particle into the reference frame**: The same $\boldsymbol{R}$ rotation is applied to the sampling function of the particle to correctly represent the original sampling of the subtomogram. To emphasize that the sampling function is now in the reference frame, we will refer to it as $\boldsymbol{w}_{ref}$ from now on.

> ***Note 36:** As explained in section **16.8.2**, emClarity is currently not computing a sampling function per particle, but divides the field of view by 9 strips parallel to the tilt-axis.*

3. **Calculate the CCC score**: The particle $\boldsymbol{s}_{ref}$ is masked with $\boldsymbol{M}_{ali}$, filtered with $\boldsymbol{C}_{ref}$ and $\mathbf{MTF}$, and finally centered and standardized. The reference $\boldsymbol{S}^{ref}$ is centered and standardized as well. Then, the CCC score is calculated, as follow:

$$\mathbf{CCC}_{score} = \frac{\sum_{q=h,k,l}^{Q} \Re\left(\mathcal{F}\{\boldsymbol{s}_{ref}\}_q \left[\boldsymbol{W}_{ref}\right]_q \overline{\mathcal{F}\{\boldsymbol{S}_{ref}\}}_q \left[\boldsymbol{w}_{ref}\right]_q\right)}{\sqrt{\sum_{q=h,k,l}^{Q} \left|\mathcal{F}\{\boldsymbol{s}_{ref}\}_q \left[\boldsymbol{W}_{ref}\right]_q\right|^2 \times \sum_{q=h,k,l}^{Q} \left|\overline{\mathcal{F}\{\boldsymbol{S}_{ref}\}}_q \left[\boldsymbol{w}_{ref}\right]_q\right|^2}} \tag{34}$$

   where $\mathcal{F}\{x\}$ is the fast Fourier transform of x. $\Re(x) = a$, where x is a complex number such as $x = a + bi$. $|x|$ is the complex magnitude of the complex number x and is defined as $\sqrt{a^2 + b^2}$. $\overline{x}$ is the complex conjugate of x.

***Note 37:** Notice that when calculating the CCC scores, the particle is masked with $\boldsymbol{M}_{ali}$, whereas the particle is masked with $\boldsymbol{M}_{particle}$ when calculating the translation (step 3. section **16.9.3**). This allows to restrict the translation within the particle, while allowing the surrounding environment (neighbouring particles in a lattice, etc.) to contribute to the alignment.*

16.9.5 Evaluate the initial alignment

First, we calculate the initial CCC score, $\mathbf{CCC}_{init}$, of each particle $\boldsymbol{s}$. We do as follow, for each particle p:

1. **Extraction**: The particle $\boldsymbol{s}_p$ is extracted it from its sub-region tomogram. This step is identical to step 3. from section 16.8.3. As the extraction is in pixel space, any remaining decimals $\left[\boldsymbol{T}_{orig}\right]_p$ from its x, y, z coordinates are saved and will be applied during the transformation, in step 3.

2. **Get the initial rotation**: At this point of the alignment, the rotation of the particle comes from the previous cycle or directly from the picking if it is the first alignment. As such, the rotation is equal to $\left[\boldsymbol{R}_{init}\right]_p = \boldsymbol{R}_{n-1,p}$, where n is the current cycle number.

3. **Get the initial translation**: As the particle was just extracted, the only translation attached to the particle is the shifts $\left[\boldsymbol{T}_{orig}\right]_p$ left from the extraction. As such, the translation is equal to $\left[\boldsymbol{T}_{init}\right]_p = \left[\boldsymbol{T}_{orig}\right]_p$.

4. **Get the CCC score**: Using the $\left[\boldsymbol{R}_{init}\right]_p$ and $\left[\boldsymbol{T}_{init}\right]_p$ defined above, calculate $\left[\mathbf{CCC}_{init}\right]_p$, as described in section 16.9.4.

Each particle p has its own CCC score. As you'll see in the next section, if we're not able to find a higher CCC score during the angular search, the particle will keep its initial rotation $\left[\boldsymbol{R}_{init}\right]_p$ and translation $\left[\boldsymbol{T}_{init}\right]_p$.

16.9.6 Angular search

In tomography, the particles are represented as 3D volumes (i.e. subtomograms). As such, we can either rotate the reference into the microscope frame *or* rotate the particle into the reference frame. The later is not possible in SPA and allows us to apply the symmetry of the particle before comparing it with the (symmetrized) reference. The difficulty here is that, the center of symmetry, or in the case of central symmetry, the position of the symmetry axis, is not necessarily correct and any error will greatly damage the particle if the symmetry is applied. To limit this effect, we first rotate the reference into the microscope frame to get a first estimate of the translation and have a better estimate of the position of the symmetry axis. Then, using this translation, we align the particle to reference to get the CCC score.

The in- and out-of-plane angles $[\Theta_{out},\ \Delta_{out}, \Theta_{in},\ \Delta_{in}]$ registered in `Raw_angleSearch` are converted into a set of $r \times 3$ Euler angles $(\phi_r,\ \theta_r,\ \psi_r)$. These rotations are finally converted into r rotation matrices $\boldsymbol{R}_r$. For each particle p and for each rotation r:

1. **Get the current rotation**: The rotation of the particle is defined as $[\boldsymbol{R}_{search}]_{p,r} = [\boldsymbol{R}_{init}]_p \boldsymbol{R}_r$. Remember, $[\boldsymbol{R}_{init}]_p = \boldsymbol{R}_{n\text{-}1,p}$.
2. **Get the current translation**: The current translation of the particle didn't change since the last cycle, it is still equal to $[\boldsymbol{T}_{search}]_{p,r} = [\boldsymbol{T}_{init}]_p = [\boldsymbol{T}_{orig}]_p$.
3. **Update the translation**: Using the $[\boldsymbol{R}_{search}]_{p,r}$ and $[\boldsymbol{T}_{search}]_{p,r}$ defined above, we calculate $[\boldsymbol{T}_{S\text{-}s}]_{p,r}$, as described in section 16.9.3. The new translation is defined as $[\boldsymbol{T}_{search}]_{p,r} = [\boldsymbol{T}_{init}]_{p,r} + [\boldsymbol{T}_{S\text{-}s}]_{p,r}$.
4. **Get the CCC score**: Using the $[\boldsymbol{R}_{search}]_{p,r}$ and $[\boldsymbol{T}_{search}]_{p,r}$ defined above, we calculate the CCC score of this transformation, $[\mathrm{CCC}_{search}]_{p,r}$, as described in section 16.9.4.

We now have, for each particle $\boldsymbol{s}_p$, the initial score $[\mathrm{CCC}_{init}]_p$ and a set of $[\mathrm{CCC}_{search}]_{p,r}$ score. The maximal $[\mathrm{CCC}_{search}]_{p,r}$ score is defined as $[\mathrm{CCC}_{search}]_p = \max_r([\mathrm{CCC}_{search}]_{p,r})$ and the corresponding rotation and translation are referred to as $[\boldsymbol{R}_{search}]_p$ and $[\boldsymbol{T}_{search}]_p$.

The current best score is defined as follow:

$$[\mathrm{CCC}_{best}]_p = \max\Big([\mathrm{CCC}_{init}]_p, [\mathrm{CCC}_{search}]_p\Big) \tag{35}$$

As such, if $[\mathrm{CCC}_{init}]_p < [\mathrm{CCC}_{search}]_p$, the particle is assigned to a new transformation, defined by the rotation $[\boldsymbol{R}_{best}]_p = [\boldsymbol{R}_{search}]_p$ and the translation $[\boldsymbol{T}_{best}]_p = [\boldsymbol{T}_{search}]_p$. Otherwise, the particle keeps its original transformation, $[\boldsymbol{R}_{best}]_p = [\boldsymbol{R}_{init}]_p$ and $[\boldsymbol{T}_{best}]_p = [\boldsymbol{T}_{init}]_p$.

16.9.7 Angular search refinement

If the angular search goes out of plane, it is refined, using a finer sampling, around the current best rotation and best translation. To do so, a new set of $r' \times 3$ Euler angles is calculated and transformed into r' rotation matrices $\boldsymbol{R}_{p,r'}$. Then, for each particle p and for each rotation r':

1. **Get the current rotation**: The rotation of the particle is defined as $[\boldsymbol{R}_{search'}]_{p,r'} = [\boldsymbol{R}_{best}]_p \boldsymbol{R}_{r'}$.

2. **Update the translation**: Using the $[\boldsymbol{R}_{search'}]_{p,r'}$ and $[\boldsymbol{T}_{best}]_p$ defined above, we calculate $[\boldsymbol{T}_{S\text{-}s}]_{p,r'}$, as described in section 16.9.3. The new translation is updated, such as $[\boldsymbol{T}_{search'}]_{p,r'} = [\boldsymbol{T}_{best}]_p + [\boldsymbol{T}_{S\text{-}s}]_{p,r'}$.
3. **Get the CCC score**: Using the $[\boldsymbol{R}_{search'}]_{p,r'}$ and $[\boldsymbol{T}_{search'}]_{p,r'}$ defined above, we calculate the CCC score of this transformation, $[\mathbf{CCC}_{search'}]_{p,r'}$, as described in section 16.9.4.

Similarly to the first angular search, we now have for each particle $\boldsymbol{s}_p$, the current best score $[\mathbf{CCC}_{best}]_p$ calculated at the previous section and a set of $[\mathbf{CCC}_{search'}]_{p,r'}$. The highest $[\mathbf{CCC}_{search'}]_{p,r'}$ score is defined as $[\mathbf{CCC}_{search'}]_p = \max_{r'}([\mathbf{CCC}_{search'}]_{p,r'})$ and the corresponding rotation and translation are referred as $[\boldsymbol{R}_{search'}]_p$ and $[\boldsymbol{T}_{search'}]_p$.

The current best score is defined as follow:

$$[\mathbf{CCC}_{best'}]_p = \max\Big([\mathbf{CCC}_{best}]_p, [\mathbf{CCC}_{search'}]_p\Big) \tag{36}$$

As such, if $[\mathbf{CCC}_{best}]_p < [\mathbf{CCC}_{search'}]_p$, the current best estimate is updated and the particle is assigned to a new transformation, defined by the rotation $[\boldsymbol{R}_{best'}]_p = [\boldsymbol{R}_{search'}]_p$ and the translation $[\boldsymbol{T}_{best'}]_p = [\boldsymbol{T}_{search'}]_p$. Otherwise, the particle keeps its transformation, $[\boldsymbol{R}_{best'}]_p = [\boldsymbol{R}_{best}]_p$ and $[\boldsymbol{T}_{best'}]_p = [\boldsymbol{T}_{best}]_p$.

16.9.8 Final translation and update

Finally, a final translation $[\boldsymbol{T}_{S\text{-}s}]_p$ will be calculated as described in section 16.9.3, using the final rotation $[\boldsymbol{R}_{final}]_p = [\boldsymbol{R}_{best}]_p$ (or if there was a refinement, $[\boldsymbol{R}_{final}]_p = [\boldsymbol{R}_{best'}]_p$) and current best translation $[\boldsymbol{T}_{best}]_p$ (or $[\boldsymbol{T}_{best'}]_p$). The final translation is defined as $[\boldsymbol{T}_{final}]_p = [\boldsymbol{T}_{best}]_p + [\boldsymbol{T}_{S\text{-}s}]_p$ (or $[\boldsymbol{T}_{final}]_p = [\boldsymbol{T}_{best'}]_p + [\boldsymbol{T}_{S\text{-}s}]_p$)

Once the alignment is over, the particles are updated with their new rotation and translation such as:

- **New rotation**: The new rotation is simply defined as $\boldsymbol{R}_{n,p} = [\boldsymbol{R}_{final}]_p$
- **New translation**: $[\boldsymbol{T}_{final}]_p$ contains the original translation $[\boldsymbol{T}_{orig}]_p$ resulting from the extraction. As the original x, y, z coordinates of the particle $\boldsymbol{T}_{n\text{-}1,p}$ implicitly contains $[\boldsymbol{T}_{orig}]_p$, we update the coordinates such as $\boldsymbol{T}_{n,p} = \boldsymbol{T}_{n\text{-}1,p} + [\boldsymbol{T}_{final}]_p - [\boldsymbol{T}_{orig}]_p$.

16.10 Tilt-series refinement

Subtomogram averaging provides accurate estimates of both particle positions and high SNR reconstructions. It is thus possible to leverage this information for improving the alignment of a tilt-series.

16.10.1 Reconstruct the synthetic tomogram

The first is to reconstruct the full tomogram $\boldsymbol{V}$ in the same way we reconstructed the sub-regions tomograms in section 16.7. Similarly to the sub-region tomograms used for subtomogram averaging and alignment, this tomogram is CTF-phase corrected. To save some precious run time and because this tomogram is only used to refine the fiducial positions, emClarity does not follow the so-called "3D CTF-correction". Indeed, the correction is forced to have only one z slab (section 16.7.2). In other words, the thickness of the specimen is not taken into account during the correction, but only the tilts. Moreover, the center of mass in z is not adjusted and the spatial model is not defined (section 16.7.3).

Once that the full tomogram is reconstructed, the subtomograms are replaced by their corresponding half-maps. For each particle i:

1. **Get the coordinates of the particle**: The x, y, z coordinates of the particle, that are saved in the metadata, corresponds to the 3D coordinates within the sub-region tomogram, with the origin at the lower left corner. In this section, we are working with the full tomogram $\boldsymbol{V}$, i.e. the entire field of view, so the coordinates must be adjusted to $\boldsymbol{V}$.
2. **Get the reference in the microscope frame**: The particle is attached to a rotation $\boldsymbol{R}_i$. Moreover, as explained in section 16.8.3 step **1.b**, the particles p are attached to a translation $[\boldsymbol{T}_{orig}]_p$. The half-map $\boldsymbol{S}$ is rotated by $\boldsymbol{R}_i^T$ and translated by $[-\boldsymbol{T}_{orig}]_i$ (note 35). The same transformation is applied to the soft-edged molecular mask $\boldsymbol{M}_{mol}$ (section 16.8.4). We will refer to the transformed reference and transformed mask as $\boldsymbol{S}_p$ and $[\boldsymbol{M}_{mol}]_p$.
3. **Replace the particle's density by the reference's density**: The voxels of the full tomogram $\boldsymbol{V}$ corresponding to the particle $\boldsymbol{s}_p$ are replaced by the masked reference, such as:

$$\boldsymbol{V}(\boldsymbol{s}_p) = \Big(1 - [\boldsymbol{M}_{mol}]_p\Big)\, \boldsymbol{s}_p + \boldsymbol{S}_p[\boldsymbol{M}_{mol}]_p \tag{37}$$

   where $\boldsymbol{V}(\boldsymbol{s}_p)$ refers to the voxels of $\boldsymbol{V}$ that corresponds to the particle $\boldsymbol{s}_p$.

16.10.2 Reproject the synthetic tilt-series

We want to reproject the "synthetic" tilted views *and* the particles coordinates. Both reprojections are calculated by tilt:

- **Reproject the coordinates**: For the tilt-series alignment, tiltalign needs the x, y coordinates of the particles, for each view of the tilt-series. Fortunately, tilt can reproject the x, y, z coordinates of the particles and if the defocus value are know, for each view, it can also calculate the defocus of each particle, for each view, while accounting for the local alignments that were used during the reconstruction.

- **Reproject the tilt-series**: The synthetic tomogram, calculated at the previous section, is reprojected into an aligned synthetic tilt-series using tilt. Of course, it takes into account the eventual local alignments that were used to reconstruct the tomogram.

Note 38: *tilt and tiltalign operates with the y axis, i.e. the* `SLICE`*, in the third dimension. So both the "synthetic" tomogram and the x, y, z coordinates are rotated by 90° around x.*

16.10.3 Refine the fiducial positions

The synthetic tomogram is now reprojected into a synthetic tilt-series. Tiles around each projected high SNR subtomogram origin are masked out, convolved with the CTF of the raw data projection at that point and aligned to the raw data.

By default, emClarity is setting the maximal number of particle which are going to be used as fiducial to 1800. This value can be changed with the `tomoCPR_randomSubset` entry. If there is more particles than the allowed number of fiducial, a random subset of particles will be selected.

We will refer to the raw aligned tilt-series as $\boldsymbol{I}_{raw}$ and synthetic tilt-series as $\boldsymbol{I}_{synt}$. Before starting the refinement, each view $[\boldsymbol{I}_{raw}]_i$ and $[\boldsymbol{I}_{synt}]_i$ are centered and standardized, first globally, then locally. Combined with the sampling mask, calculated in section 16.5.1, it allows us to define a mask $[\boldsymbol{M}_{eval}]_i$ excluding regions that are not sampled or that significantly varies from the rest of the data, like carbon, contaminants, etc.

Before refining the fiducial positions, a global shift estimate is calculated, for each view i.

1. **Calculate the cross-correlation map**: The cross-correlation between $[\boldsymbol{I}_{raw}]_i$, which is not CTF corrected, and $[\boldsymbol{I}_{synt}]_i$, which is multiplied by the CTF, is defined as follow:

$$\mathbf{CC}_i = \mathcal{F}^{-1}\left\{\boldsymbol{W}_{low\text{-}pass}\,[\boldsymbol{W}_{ctf}]_i\,\mathcal{F}\{[\boldsymbol{I}_{raw}]_i\}\,\overline{\mathcal{F}\{[\boldsymbol{I}_{synt}]_i\}}\,\left|[\boldsymbol{W}_{ctf'}]_i\right|\right\}\boldsymbol{M}_{peak\text{-}global} \tag{38}$$

   where $\boldsymbol{W}_{low\text{-}pass}$ is a low-pass filter, with a low-cutoff set by `tomoCprLowPass`. $[\boldsymbol{W}_{ctf}]_i$ is the astigmatic 2D CTF of the view i, with envelope. $|[\boldsymbol{W}_{ctf'}]_i|$ is the astigmatic 2D CTF without envelope and is used to modulate the amplitudes of the reprojected reference to better match the raw data. $\boldsymbol{M}_{peak\text{-}global}$ is a spherical mask and limits the translation to ~ 10 to 20Å.

2. **Get highest peak position**: This step is identical to the last step of section 16.9.3 and outputs a translation $[\boldsymbol{T}_{global}]_i$.

3. **Apply the global translation estimate**: $[\boldsymbol{I}_{raw}]_i$ is translated by $[\boldsymbol{T}_{global}]_i$, using linear interpolation.

Then, the x and y position of each particle, for each view i is refined as follow:

1. **Extract the particle tile**: For both $\boldsymbol{I}_{raw}$ and $\boldsymbol{I}_{synt}$, a tile of $1.5\times$ `particleRadius` is extracted at the particle reprojected coordinate, centered and standardized. If the tiles overlap with the evaluation mask $[\boldsymbol{M}_{eval}]_i$, the particle is ignored and will not be used as fiducial.

2. **Calculate the translation between the raw and the synthetic tile**: The tiles are padded 2 times in real space, masked and the CC map is calculated as follow:

$$\mathbf{CC}_{p,i} = \mathcal{F}^{-1}\left\{\boldsymbol{W}_{low\text{-}pass}\,[\boldsymbol{W}_{ctf}]_{p,i}\,\mathcal{F}\left\{[\boldsymbol{I}_{raw}]_{p,i}\right\}\,\overline{\mathcal{F}\left\{[\boldsymbol{I}_{synt}]_{p,i}\right\}}\right\}\boldsymbol{M}_{peak\text{-}local} \tag{39}$$

where $[\boldsymbol{W}_{ctf}]_{p,i}$ is the anisotropic 2D CTF at the particle position, defined by the defocus value $\mathbf{z}_{p,i}$. $\boldsymbol{M}_{peak\text{-}local}$ is a mask controlled by `Peak_mType` and `Peak_mRadius` and is used to restrict the translation. By default, it is set to $0.4 \times$ `particleRadius`.

3. **Get the translation estimate**: This step is identical to the last step of section 16.9.3 and outputs a translation $[\boldsymbol{T}_{local}]_{p,i}$.

At this end of this procedure, each fiducial is translated by $[\boldsymbol{T}_{orig}]_{p,i} + [\boldsymbol{T}_{global}]_i + [\boldsymbol{T}_{local}]_{p,i}$ and saved for the next step.

The defocus value can be refined by sampling a range of defocus value around the current estimate in order to maximize $\mathrm{CC}_{p,i}$. The defocus search is set by $\mathbf{z}_{p,i}\pm$ `tomoCprDefocusRange`, with a step of `tomoCprDefocusStep`. For each image i, the defocus shifts $\boldsymbol{\Delta}\mathbf{z}_{p,i}$ that maximize $\mathrm{CC}_{p,i}$ are averaged and this average will be added to the current the defocus value of the image.

16.10.4 Align and transform the tilt-series

The fiducials are now aligned, so we can use them to find a new geometric model with tiltalign. A bash script is saved in `mapBack<n>` and contains the parameters that are used for the alignment. Most of them are accessible directly via the emClarity parameter file (table 11).

Finally, the raw unaligned tilt-series in `fixedStacks` is transformed using this new geometric model. This last part is done via `ctf update` and is similar to section 16.5 to the exception that this new alignment is relative to the aligned tilt-series. As such, the new rotation and shifts are added to the original ones, and the tilt angles are updated. This new transformation is saved in a new table 14. The beads coordinates are updated as well and erased.

16.11 Subtomogram classification

The heterogeneity of the data-set can be analysed by comparing individual particles with the current references. This analysis can be focused to some specific features, i.e. resolutions bands, like α-helices, small protein domains, etc. Briefly, difference maps are calculated between each particle and the references, for each resolution bands. These maps are then analysed by Principal Component Analysis (PCA), using Singular Value Decomposition (SVD). Once the difference maps are reduced in dimensionality and described by some principal axes, they can be clustered with k-means or other clustering algorithms. As a result, each subtomogram is assigned to a class and a subtomogram average can be generated for each class. We will detail this entire process in the following sections.

Usually, PCA is calculated on the covariance or correlation matrix of the data. These are quite popular in tomography because we can consider the missing wedge of the particles while calculating these matrices, using constrained cross-correlation. SVD allows us to perform a PCA directly on the data, but we still need to take into account that each particle has an incomplete sampling. To solve this problem, we are going to calculate difference maps.

16.11.1 Combine the half-maps

First, we need to prepare the reference to which the data-set is being compared to; what we are going to classify is the difference between the particles $\boldsymbol{s}$ and this reference. With the goal-standard approach, we have two half-maps, $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$. For classification, these two references are aligned by spline interpolation, using the rotation $\boldsymbol{R}_{gold}$ and translation $\boldsymbol{T}_{gold}$ calculated in section 16.8.5, and averaged into one unique reference, simply referred as $\boldsymbol{S}$. Of course, this alignment is not persistent, so that extracted class averages are still independent half-sets.

> ***Note 39:*** *The classification can be performed at a different sampling than the half-maps, hence the two distinct parameters* `Cls_samplingRate` *and* `Ali_samplingRate`*. In this case, the reference $\boldsymbol{S}$ is re-scaled to the desired pixel size, using linear interpolation.*

16.11.2 Resolution bands

As mentioned previously, we can restrict the analysis of the physical differences between the data-set and the reference to some r specific length scales, also referred as resolution bands or features. These resolution bands are controlled by the `pcaScaleSpace` entry and we refer to them as $\boldsymbol{L}_r$. As we will see in more detail, the PCA is calculated for each one of these length scales. Importantly, the clustering is only calculated once and considers each length scale simultaneously to provide a richer description of the feature space.

First, we can exclude from the analysis most of the background by calculating a molecular mask $\boldsymbol{M}_{mol}$ of $\boldsymbol{S}$, as described in 16.8.4. The `Cls_mType`, `Cls_mRadius` and `Cls_mCenter` parameters are used to a compute soft-edged shape mask, $\boldsymbol{M}_{shape}$. In order to constrain the analysis to the desired region, this mask is applied to the molecular mask $\boldsymbol{M}_{mol}$. This mask is saved as `<prefix>_pcaVolMask.mrc`.

Then, for each resolution band r, either a band-pass filter or a low-pass filter $\boldsymbol{W}_r$ is prepared.

- **Low-pass filters**: By default, low-pass filters are used and includes frequencies from 400Å to $(0.9 \times \boldsymbol{L}_r)$Å. These filters do not define “resolution bands” per say and most frequencies will be shared in the resulting filtered references $\boldsymbol{S}_r$, but they have produced good results nonetheless.

Note 40: As often, frequencies before 400Å are removed to center the reference and remove large densities gradients.

- **Band-pass filters**: If `test_updated_bandpass` is true, band-pass filters are used instead and include frequencies from 400Å to 100Å and a resolution band at $\boldsymbol{L}_r$.

For visualization, the reference $\boldsymbol{S}$ is filtered with the filters $\boldsymbol{W}_r$, masked with $\boldsymbol{M}_{mol}$, centered and standardized. These volumes are saved in `test_filt.mrc`

16.11.3 Difference maps

What we want to classify is the difference between each particle p and the reference $\boldsymbol{S}$, for each length scale r, while accounting for incomplete sampling of the particles. To calculate a difference map $\boldsymbol{X}_{p,r}$, we do as follow:

1. **Get the particle in the reference frame**: The particle $\boldsymbol{s}_p$ is extracted from the CTF phase-multiplied sub-region tomogram. Because we want to use the entire data-set for the classification, the two half-sets must be aligned in the same way we aligned two half-maps $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ in section 16.11.1. As such, the first half-set is translated by $[\boldsymbol{T}_{orig}]_p + \boldsymbol{T}_{gold}$ and rotated by $\boldsymbol{R}_p\boldsymbol{R}_{gold}$, whereas the second half-set is simply translated by $[\boldsymbol{T}_{orig}]_p$ and rotated by $\boldsymbol{R}_p$.
2. **Get the sampling function of the particle in the reference frame**: The same $\boldsymbol{R}_p$ rotation (or $\boldsymbol{R}_p\boldsymbol{R}_{gold}$) is applied to the sampling function of the particle $\boldsymbol{w}_p^2$ to correctly represent the original sampling of the subtomogram.
3. **Calculate the difference map**: The difference map $\boldsymbol{X}_{p,r}$ between the reference and the p^{th} particle, for the r^{th} length scale, is then defined as:

$$\boldsymbol{X}_{p,r} = \mathcal{F}^{-1}\left\{\left(\underbrace{\mathcal{F}\{\boldsymbol{S}\}\,\boldsymbol{w}_p^2}_{\mu=0,\ \sigma=1} - \underbrace{\mathcal{F}\{\boldsymbol{s}_p\}}_{\mu=0,\ \sigma=1}\right)\boldsymbol{W}_r\right\}\boldsymbol{M}_{mol} \tag{40}$$

$\boldsymbol{X}_{p,r}$ is then centered and standardized. To save memory and time, only the voxels within $\boldsymbol{M}_{mol}$ are saved.

Note 41: The decomposition will be calculated on the host, but the difference maps are calculated on the device, which is much more efficient. The `PcaGpuPull` *entry controls how many maps $\boldsymbol{X}_{p,r}$ should be held on the device at any given time.*

16.11.4 Singular Value Decomposition

The difference maps $\boldsymbol{X}_{p,r}$ are then linearised into column vectors and stacked into a matrix $\boldsymbol{X}$, where the number of rows is the number of voxels v within $\boldsymbol{M}_{mol}$, the number of columns is the number of particles p and the number of pages (the third dimension) is the number of length scales r.

One important step when doing a PCA is to center the variables, i.e. the voxels. In our case, it means that the rows of $\boldsymbol{X}$ should be set to have a mean equal to zero. In other words, each difference map must be subtracted by the average of all the difference maps.

Once $\boldsymbol{X}$ is ready, we can then calculate the SVD, for each length scale r. This is a representation of what we have:

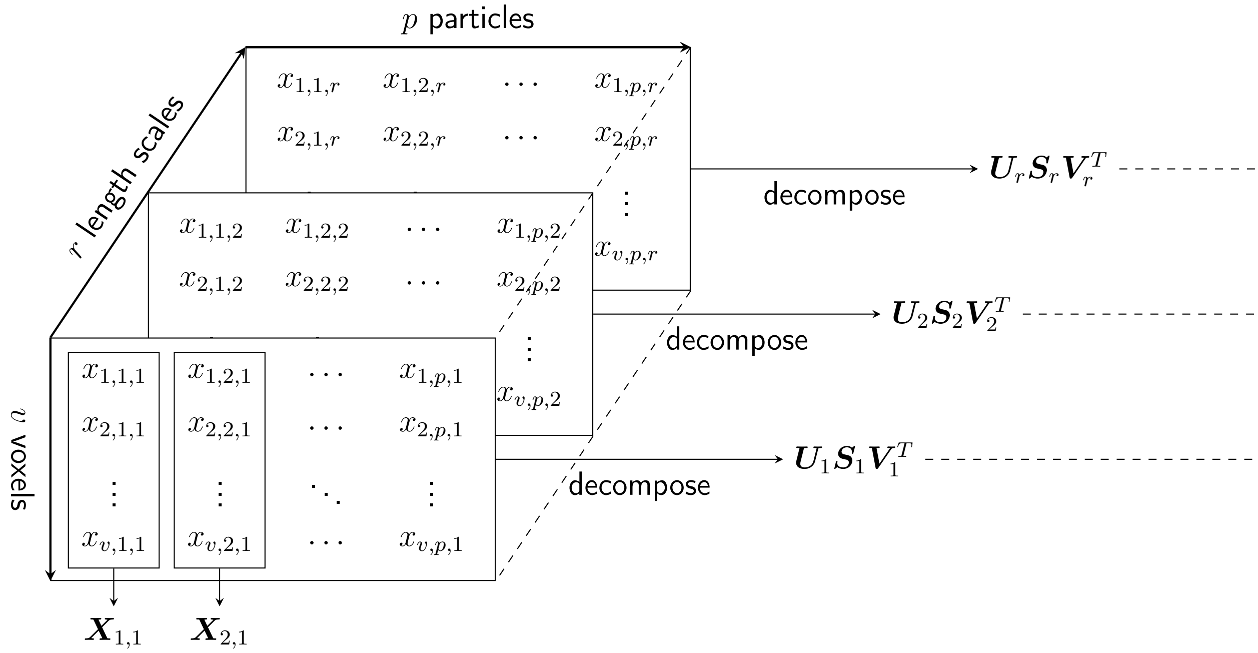
p particles
r length scales
v voxels
$x_{1,1,r}$ $x_{1,2,r}$ $\cdots$ $x_{1,p,r}$
$x_{2,1,r}$ $x_{2,2,r}$ $\cdots$ $x_{2,p,r}$
$\vdots$
$x_{v,p,r}$
$x_{1,1,2}$ $x_{1,2,2}$ $\cdots$ $x_{1,p,2}$
$x_{2,1,2}$ $x_{2,2,2}$ $\cdots$ $x_{2,p,2}$
$\vdots$
$x_{v,p,2}$
$x_{1,1,1}$ $x_{1,2,1}$ $\cdots$ $x_{1,p,1}$
$x_{2,1,1}$ $x_{2,2,1}$ $\cdots$ $x_{2,p,1}$
$\vdots$ $\vdots$ $\ddots$ $\vdots$
$x_{v,1,1}$ $x_{v,2,1}$ $\cdots$ $x_{v,p,1}$
$\boldsymbol{X}_{1,1}$ $\boldsymbol{X}_{2,1}$
decompose
$\boldsymbol{U}_r\boldsymbol{S}_r\boldsymbol{V}_r^T$
decompose
$\boldsymbol{U}_2\boldsymbol{S}_2\boldsymbol{V}_2^T$
decompose
$\boldsymbol{U}_1\boldsymbol{S}_1\boldsymbol{V}_1^T$

Figure 2: Singular Value Decomposition of the difference maps p, for each length scale r. The features, i.e. the voxels, are organized in rows. As such, $\boldsymbol{U}$ contains the right singular vectors, i.e. the principal directions/axes, $\boldsymbol{V}^T$ contains the left singular vectors. Consequently, $\boldsymbol{S}\boldsymbol{V}^T$ are the principal components.

As most of the variance is usually explained within the first 20 to 30 directions, it is usually not useful to save all of the directions. Use `Pca_maxEigs` to select the number of directions you want to save. The goal now is to select the principal directions that are going to be used to reproject the data onto. This entirely relies on the user, so the principal directions are reshaped and a few files are saved to help:

- **Variance map**: The covariance matrix of $\boldsymbol{X}$ is saved as `*_varianceMap<n>-STD-<r>.mrc`, where `<n>` corresponds to the `Pca_maxEigs` and `<r>` corresponds to the length scale number r in `pcaScaleSpace`. These may be opened on top of your averages from this cycle and should highlight regions where there is significant variability across the data set. If these maps show little blobs everywhere, there is either no significant variability (just picking up noise) or there is still substantial wedge bias at this resolution.
- **Principal axes/directions**: The principal axes $\boldsymbol{U}$ are saved in `*_eigenImage<n>-STD-mont-<r>.mrc`. The axes are organized from the lower left moving across the bottom row incrementing by one. The first image is associated with the greatest singular value, i.e. it describes the greatest portion of variance in the dataset. The second image describes the greatest portion of remaining variance, and so on. The value of the voxels can be greater than 1 or less than zero, so they should not be interpreted as a grayscale image. It might be easier to visualize them with colors (3dmod: `F12`). For visualization, the principal axes are centered and standardized.
- **Sums**: The principal axes can be difficult to interpret, so it might useful to add them to the

reference, which highlights what is being explained in each principal direction. These are saved in saved in `*_eigenImage<n>-STD-SUM-mont-<r>.mrc`.

16.11.5 Clustering

Once that the principal directions of each length scale r are selected by the user, we can reprojected the data along these axes. We refer to the principal components as $[\boldsymbol{S}_r\boldsymbol{V}_r^T]_{best}$. Before PCA, each particle i was described by v variables. Now, each particle is only described by $a+b+c$ variables (Figure 3), with $a+b+c \ll v$, i.e. the dimentionality of the dataset was reduced.

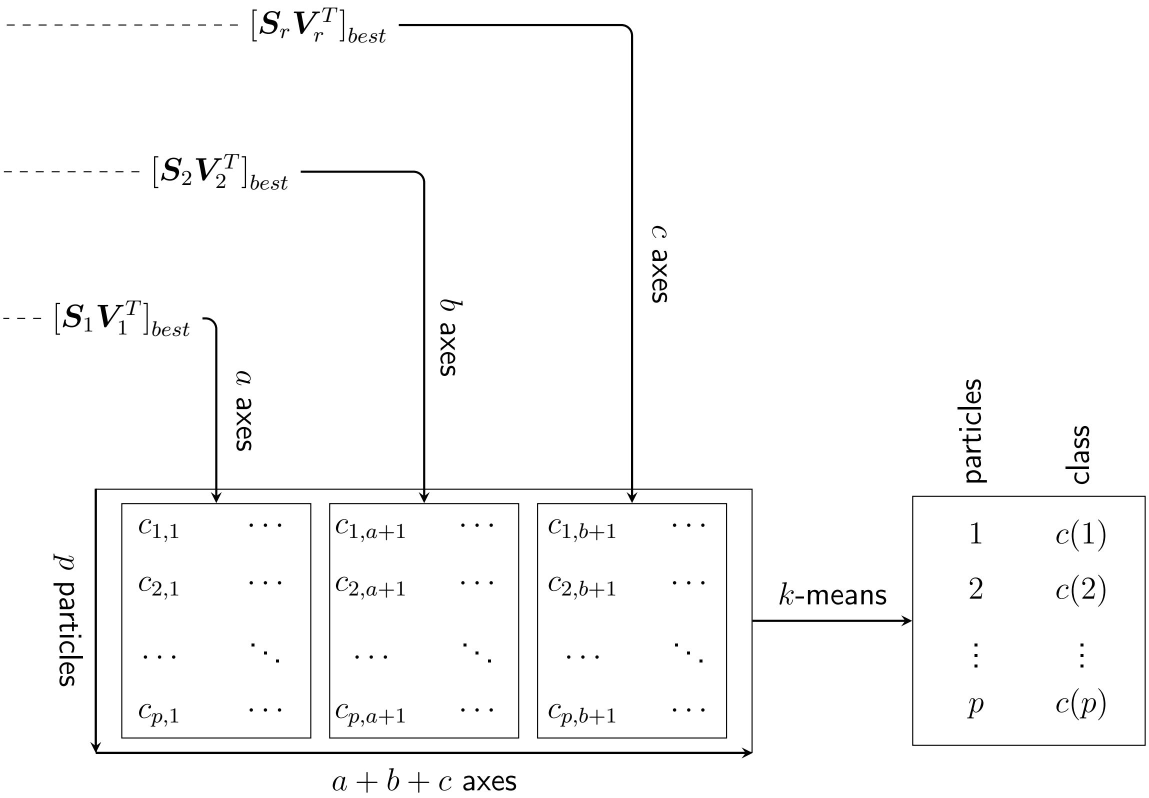
$[\boldsymbol{S}_r\boldsymbol{V}_r^T]_{best}$
$[\boldsymbol{S}_2\boldsymbol{V}_2^T]_{best}$
$[\boldsymbol{S}_1\boldsymbol{V}_1^T]_{best}$
c axes
b axes
a axes
p particles
$c_{1,1}$ $c_{2,1}$ $c_{p,1}$
$c_{1,a+1}$ $c_{2,a+1}$ $c_{p,a+1}$
$c_{1,b+1}$ $c_{2,b+1}$ $c_{p,b+1}$
$a+b+c$ axes
k-means
particles
class
1 $c(1)$
2 $c(2)$
p $c(p)$

Figure 3: The data is projected onto the principal axes, for each r length scale and stacked into one single $\boldsymbol{S}\boldsymbol{V}^T$ matrix of principal components. For visualization, the principal components are transposed to have the axes as columns and the particles as rows. The projected data is then clustered, usually with k-means. As a result, each particle is assigned to a class.

The projected data $[\boldsymbol{S}\boldsymbol{V}^T]_{best}$ is then clustered into k clusters using k-means clustering. k is set by `Pca_clusters`. The squared Euclidean distance metric is used by default, i.e. each centroid is the mean of the points in that cluster, and the number of replicates is set to 128, i.e. the number of times to repeat clustering using new initial cluster centroid positions. Both of these can be changed using the parameters `Pca_distMeasure` and `Pca_nReplicates`. The k-means++ algorithm is used for cluster center initialization and the number of maximal iteration is set to 5000.

At the end, each particle p is assigned to a cluster $c(p)$, which is saved into the metadata.

17 References

[1] Fabian Eisenstein, Radostin Danev, and Martin Pilhofer. Improved applicability and robustness of fast cryo-electron tomography data acquisition. *Journal of Structural Biology*, 208(2):107–114, November 2019.

[2] Dimitry Tegunov, Liang Xue, Christian Dienemann, Patrick Cramer, and Julia Mahamid. Multi-particle cryo-em refinement with m visualizes ribosome-antibiotic complex at 3.7Å inside cells. *bioRxiv*, 2020.

[3] Timothy Grant and Nikolaus Grigorieff. Measuring the optimal exposure for single particle cryo-em using a 2.6 Åreconstruction of rotavirus vp6. *eLife*, 4:e06980, may 2015.

[4] David N. Mastronarde and Susannah R. Held. Automated tilt series alignment and tomographic reconstruction in imod. *Journal of Structural Biology*, 197(2):102 – 113, 2017. Electron Tomography.

[5] Charles V. Sindelar and Nikolaus Grigorieff. Optimal noise reduction in 3D reconstructions of single particles using a volume-normalized filter. *Journal of structural biology*, 180(1):26–38, October 2012.

[6] Grant J Jensen and Roger D Kornberg. Defocus-gradient corrected back-projection. *Ultramicroscopy*, 84(1):57 – 64, 2000.

[7] Beata Turoňová, Florian K. M. Schur, William Wan, and John A. G. Briggs. Efficient 3D-CTF correction for cryo-electron tomography using NovaCTF improves subtomogram averaging resolution to 3.4Å. *Journal of Structural Biology*, 199(3):187–195, 2017.

[8] 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.

[9] Tanmay A. M. Bharat, Christopher J. Russo, Jan Löwe, Lori A. Passmore, and Sjors H. W. Scheres. Advances in Single-Particle Electron Cryomicroscopy Structure Determination applied to Sub-tomogram Averaging. *Structure*, 23(9):1743–1753, September 2015.

[10] C.A. Diebolder, F.G.A. Faas, A.J. Koster, and R.I. Koning. Conical fourier shell correlation applied to electron tomograms. *Journal of Structural Biology*, 190(2):215 – 223, 2015.

[11] Marin van Heel and Michael Schatz. Fourier shell correlation threshold criteria. *Journal of Structural Biology*, 151(3):250 – 262, 2005.