*cis*TEM: User-friendly software for single-particle image processing

Tim Grant [1], Alexis Rohou [1,2] and Nikolaus Grigorieff [1]

[1] Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, Virginia, USA

[2] Present address: Department of Structural Biology, Genentech, South San Francisco, California, USA

Abstract

We have developed new open-source software called *cis*TEM (computational imaging system for transmission electron microscopy) for the processing of data for high-resolution electron cryo-microscopy and single-particle averaging. *cis*TEM features a graphical user interface that is used to submit jobs, monitor their progress, and display results. It implements a full processing pipeline including movie processing, image defocus determination, automatic particle picking, 2D classification, ab-initio 3D map generation from random parameters, 3D classification, and high-resolution refinement and reconstruction. Some of these steps implement newly-developed algorithms; others were adapted from previously published algorithms. The software is optimized to enable processing of typical datasets (2000 micrographs, 200k – 300k particles) on a high-end, CPU-based workstation in half a day or less, comparable to GPU-accelerated processing. Jobs can also be scheduled on large computer clusters using flexible run profiles that can be adapted for most computing environments. *cis*TEM is available for download from cistem.org.

Introduction

The three-dimensional (3D) visualization of biological macromolecules and their assemblies by single-particle electron cryo-microscopy (cryo-EM) has become a prominent approach in the study of molecular mechanisms (Cheng et al., 2015; Subramaniam et al., 2016). Recent advances have been primarily due to the introduction of direct detectors (McMullan et al., 2016). With the improved data quality, there is increasing demand for advanced computational algorithms to extract signal from the noisy image data and reconstruct 3D density maps from them at the highest possible resolution. The promise of near-atomic resolution (3 – 4 Å), where densities can be interpreted reliably with atomic models, has been realized by many software tools and suites (Frank et al., 1996; Hohn et al., 2007; Lyumkis et al., 2013; Scheres, 2012; Tang et al., 2007; van Heel et al., 1996). Many of these software tools implement a standard set of image processing steps that are now routinely performed in a single particle project. These typically include movie frame alignment, contrast transfer function (CTF) determination, particle picking, two-dimensional (2D) classification, 3D reconstruction, refinement and classification, and sharpening of the final reconstructions.

We have written new software called *cis*TEM to implement a complete image processing pipeline for single-particle cryo-EM, including all these steps, accessible through an easy-to-use graphical user interface (GUI). Some of these steps implement newly-developed algorithms described below; others were adapted from previously published algorithms. *cis*TEM consists of a set of compiled programs and tools, as well as a wxWidgets-based GUI. The GUI launches programs and controls them by sending specific commands and receiving results via TCP/IP sockets. Each program can also be run manually, in which case it solicits user input on the command line. The design of *cis*TEM, therefore, allows users who would like to have more

control over the different processing steps to design their own procedures outside the GUI. To adopt this new architecture, a number of previously existing Fortran-based programs were rewritten in C++, including Unblur and Summovie (Grant and Grigorieff, 2015b), mag_distortion_estimate and mag_distortion_correct (Grant and Grigorieff, 2015a), CTFFIND4 (Rohou and Grigorieff, 2015), and Frealign (Lyumkis et al., 2013). Additionally, algorithms described previously were added for particle picking (Sigworth, 2004), 2D classification (Scheres et al., 2005) and ab-initio 3D reconstruction (Grigorieff, 2016), sometimes with modifications to optimize their performance.

*cis*TEM currently does not support computation on graphical processing units (GPUs). Benchmarking of a hotspot identified in the global orientational search to determine particle alignment parameters showed that an NVIDIA K40 GPU performs approximately as well as 16 Xeon E5-2687W CPU cores after the code was carefully optimized for the respective hardware in both cases. Since CPU code is more easily maintained and more generally compatible with existing computer hardware, the potential benefit of GPU-adapted code is primarily the lower cost of a high-end GPU compared with a high-end CPU. We chose to focus on optimizing our code for CPU.

Results

Movie alignment and CTF determination

Movie alignment and CTF determination are based on published algorithms previously implemented in Unblur and Summovie (Grant and Grigorieff, 2015b), and CTFFIND4 (Rohou and Grigorieff, 2015), respectively, and these are therefore only briefly described here. Unblur

determines the translations of individual movie frames necessary to bring features (particles) visible in the frames into register. Each frame is aligned against a sum of all other frames that is iteratively updated until there is no further change in the translations. The trajectories along the x- and y-axes are smoothed using a Savitzky–Golay filter to reduce the possibility of spurious translations. Summovie uses the translations to calculate a final frame average with optional exposure filtering to take into account radiation damage of protein and maximize its signal in the final average. *cis*TEM combines the functionality of Unblur and Summovie into a single panel and exposes all relevant parameters to the user (Figure 1). Both programs were originally written in Fortran and have been rewritten entirely in C++.

CTFFIND4 fits a calculated two-dimensional CTF to Thon rings (Thon, 1966) visible in the power spectrum calculated from either images or movies. The fitted parameters include astigmatism and, optionally, phase shifts generated by phase plates. When computed from movies, the Thon rings are often more clearly visible compared to Thon rings calculated from images (Figure 2; (Bartesaghi et al., 2014)). When selecting movies as inputs, the user can specify how many frames should be summed to calculate power spectra. An optimal value to amplify Thon rings would be to sum the number of frames that correspond to an exposure of about 4 electrons/$Å^2$ (McMullan et al., 2015).

Since our original description of the CTFFFIND4 algorithm (Rohou and Grigorieff, 2015), several significant changes were introduced. (1) The initial exhaustive search over defocus values can now be performed using a one-dimensional version of the CTF (i.e. with only two parameters: defocus and phase shift) against a radial average of the amplitude spectrum. This search is much faster than the equivalent search over the 2D CTF parameters (i.e., four parameters: two for defocus, one for astigmatism angle and one for phase shift) and can be

expected to perform well except in cases of very large astigmatism (Zhang, 2016). Once an initial estimate of the defocus parameter has been obtained, it is refined by a conjugate gradient minimizer against the 2D amplitude spectrum, as done previously. In *cis*TEM, the default behavior is to perform the initial search over the 1D amplitude spectrum, but the user can revert to previous behavior by setting a flag in the "Expert Options" of the "Find CTF" Action panel. (2) If the input micrograph's pixel size is smaller than 1.4 Å, the resampling and clipping of its 2D amplitude spectrum will be adjusted so as to give a final spectrum for fitting with an edge corresponding to 1/2.8 $Å^{-1}$, to avoid all of the Thon rings being located near the origin of the spectrum, where they can be very poorly sampled. (3) The computation of the quality of fit (CC_{fit} in (Rohou and Grigorieff, 2015)) is now computed over a moving window, similar to (Sheth et al., 2015), rather than at intervals delimited by nodes in the CTF. (4) Following background subtraction as described in (Mindell and Grigorieff, 2003), a radial, cosine-edged mask is applied to the spectrum, and this masked version is used during search and refinement of defocus, astigmatism and phase shift parameters. The cosine is 0.0 at the origin, and 1.0 at a radius corresponding to 1/4 $Å^{-1}$, and serves to emphasize high-resolution Thon rings, which are less susceptible to artefacts caused by imperfect background subtraction. For all outputs from the program (diagnostic image of the amplitude spectrum, 1D plots, etc.), the background-subtracted, but non-masked, version of the amplitude spectrum is used. (5) Users receive a warning if the box size of the amplitude spectrum and the estimated defocus parameters suggest that significant CTF aliasing occurred (Penczek et al., 2014).

Particle picking

Putative particles are found by matching to a soft-edged disk template. The use of a soft-edged disk template as opposed to structured templates has two main advantages. It greatly speeds up calculation, enabling picking in 'real time', and alleviates the problem of templates biasing the result of all subsequent processing towards those templates (Henderson, 2013; Subramaniam, 2013; van Heel, 2013). Any bias that is introduced will be towards a featureless "blob" and will likely be obvious if present.

The picking is performed using an algorithm adapted from (Sigworth, 2004). Rather than describing it fully, we will emphasize here where we deviated from this algorithm. The user must specify three parameters: the radius of the template disk, the maximum radius of the particle, which sets the minimum distance between picks, and the detection threshold value, given as a number of standard deviations of the (Gaussian) distribution of scores expected if no particles were present in the input micrograph. Values of 1.0 to 6.0 for this threshold generally give acceptable results. All other parameters mentioned below can usually remain set to their default values.

Prior to matched filtering, micrographs are resampled by Fourier cropping to a pixel size of 15 Å (the user can override this by changing the "Highest resolution used in picking" value from its default 30 Å), and then filtered with a high-pass cosine-edged aperture to remove very low-frequency density ramps caused by variations in ice thickness.

The background noise spectrum of the micrograph is estimated by computing the average rotational power spectrum of 50 areas devoid of particles, and is then used to "whiten" the background (shot + solvent) noise of the micrograph. Normalization, including CTF effects, and matched filtering are then performed as described (Sigworth, 2004), except using a single reference image and no principal components' decomposition.

One difficulty in estimating the background noise spectrum of the micrograph is to locate areas devoid of particles without a priori knowledge of their locations. Our algorithm first computes a map of the local variance and local mean in the micrograph (computed over the area defined by the maximum radius given by the user (Roseman, 2004; van Heel, 1982)) and the distribution of values of these mean and variance maps. The average radial power spectrum of the 50 areas of the micrograph with the lowest local variance is then used as an estimate of the background noise spectrum. Optionally, the user can set a different number of areas to be used for this estimate (for example if the density of particles is very high or very low) or use areas with local variances closest to the mode of the distribution of variances, which may also be expected to be devoid of particles.

Matched-filter methods are susceptible to picking high-contrast features such as contaminating ice crystals or carbon films. (Sigworth, 2004) suggests subtracting matched references from the extracted boxes and examining the remainder in order to discriminate between real particles and false positives. In the interest of performance, we decided instead to pick using a single artificial reference (disk) and to forgo such subtraction approaches. To avoid picking these kinds of artifacts, the user can choose to ignore areas with abnormal local variance or local mean. We find that ignoring high-variance areas often helps avoid edges of problematic objects, e.g. ice crystals or carbon foils, and that avoiding high- and low-mean areas helps avoid picking from areas within them, e.g. the carbon foil itself or within an ice crystal (Figure 3). The thresholds used are set to $Mo + 2\,FWHM$ for the variance and $\pm\,(Mo + 2\,FWHM)$ for the mean, where Mo is the mode and $FWHM$ the full width at half-maximum of the distribution of the relevant statistic. For micrographs with additional phase plate phase shifts between 0.1 and 0.9 π, where much higher contrast is expected, the variance threshold is increased to $Mo + 8\,FWHM$. We have found that

158 in favorable cases many erroneous picks can be avoided. Remaining false-positive picks are
159 removed later during 2D classification.

160 Because of our emphasis on performance, our algorithm can be run nearly instantaneously on a
161 typical ~4K image, using a single processor. In the Action panel, the user is presented with an
162 "Auto preview" mode to enable interactive adjustment of the picking parameters (Figure 3). In
163 this mode, the micrograph is displayed with optional and adjustable low-pass and high-pass
164 filters, and the results of picking using the currently selected parameters are overlaid on top.
165 Changing one or more of the parameters leads to a fast re-picking of the displayed micrograph,
166 so that the parameters can be optimized in real-time. Once the three main parameters have been
167 adjusted appropriately, the full complement of input micrographs can be picked, usually in a few
168 seconds or minutes.

169 A possible disadvantage of using a single disk template exists when the particles to be picked are
170 non-uniform in size or shape (e.g. in the case of an elongated particle). In this case, it may be
171 expected that a single template would have difficulty in picking all the different types and views
172 of particles present, and that in this case using a number of different templates would lead to a
173 more accurate picking. In practice, we found that with careful optimization of the parameters,
174 elongated particles and particles with size variation (Figure 3) were picked adequately.

175 The underlying implementation of the algorithm supports multiple references as well as
176 reference rotation. These features may be exposed to the graphical user interface in future
177 versions, for example enabling the use of 2D class averages as picking templates, should the
178 need arise.

179

2D classification

2D classification is a relatively quick and robust way to assess the quality of a single-particle dataset. *cis*TEM implements a maximum likelihood algorithm (Scheres et al., 2005) and generates fully CTF-corrected class averages that typically display clear high-resolution detail, such as secondary structure. Integration of the likelihood function is done by evaluating the function at defined angular steps $d\alpha$ that are calculated according to

$$d\alpha = R/D \tag{1}$$

where R is the resolution limit of the data and D is the diameter of the particle (twice the mask radius that is applied to the iteratively-refined class averages). *cis*TEM runs a user-defined number of iterations n defaulting to 20. To speed up convergence, the resolution limit is adjusted as a function of iteration cycle l ($0 \leq l < n$):

$$R = R_{start} + l\left(R_{finish} - R_{start}\right)/(n-1) \tag{2}$$

where R_{start} and R_{finish} are user-defined resolution limits at the first and last iteration, defaulting to 40 Å and 8 Å, respectively. The user also sets K, the number of classes to calculate. Depending on this number and the number of particles N in the dataset, only a percentage p of the particles are included in the calculation. These particles are randomly reselected for each iteration and p is typically small, for example 0.1, in the first 10 iterations (p_{0-9}), then increases to 0.3 for iteration 10 to 14 (p_{10-14}) and finishes with five iterations including all data (p_{15-19}):

$$p_{0-9} = \begin{cases} 300K/N, & 300K/N < 1 \\ 1, & 300K/N \geq 1 \end{cases}$$

$$p_{10-14} = \begin{cases} 0.3, & p_{0-9} < 0.3 \\ p_{0-9}, & p_{0-9} \geq 0.3 \end{cases} \tag{3}$$

$$p_{15-19} = 1$$

For example, for a dataset containing $N = 100{,}000$ particles, $p_{0-9} = 0.15$, i.e. 15% of the data will be used to obtain $K = 50$ classes. Apart from speeding up the calculation, the stepwise increase of the resolution limit and the random selection of subsets of the data also reduce the chance of overfitting (see also the calculation of ab-initio 3D reconstructions and 3D refinement below) and, therefore, increase the convergence radius of the 2D classification algorithm.

For the calculation of the likelihood function, the particle images $\mathbf{X}_i$ are noise-whitened by dividing their Fourier transforms $\mathcal{F}\{\mathbf{X}_i\}$ by the square root of the radially average noise power spectrum, NPS:

$$\mathcal{F}\{\tilde{\mathbf{X}}_i\}(\mathbf{g}) = \mathcal{F}\{\mathbf{X}_i\}(\mathbf{g})/\sqrt{NPS(g)} \quad (4)$$

where $\mathbf{g}$ is the 2D reciprocal space coordinate and $g = |\mathbf{g}|$ its magnitude. The noise power spectrum is calculated from the boxed particle images using the area outside the circular mask set by the user according to the expected particle size. To increase accuracy, it is further averaged across 2000 randomly selected particles. The background (density outside the mask) is further normalized by adding a constant to each particle that yields a background average of zero.

Finally, at the beginning of each iteration, noise features in the class averages $\mathbf{A}_i$ are suppressed by resetting negative values below a threshold t_i to the threshold:

$$t_i = -0.3 \max_j A_{i,j} \quad (5)$$

where j runs over all pixels in average $\mathbf{A}_i$.

221 *3D refinement (FrealignX)*

222 The refinement of 3D reconstructions in *cis*TEM uses a version of Frealign (Lyumkis et al.,
223 2013) that was specifically designed to work with *cis*TEM. Most of Frealign's control
224 parameters are exposed to the user in the "Manual Refine" Action panel (Figure 4). The "Auto
225 Refine" and "Ab-Initio" panels also use Frealign but manage many of the parameters
226 automatically (see below). Frealign's algorithm was described previously (Grigorieff, 2007;
227 Lyumkis et al., 2013) and this section will mostly cover important differences, including a new
228 objective function used in the refinement, different particle weighting used in reconstructions,
229 optional likelihood-based blurring, as well as new masking options.

230 **Matched filter** To make Frealign compatible with *cis*TEM's GUI, the code was completely
231 rewritten in C++, and it will be referred to here as Frealign v10, or FrealignX. The new version
232 makes use of a matched filter (McDonough and Whalen, 1995) to maximize the signal in cross
233 correlation maps calculated between particle images and reference projections. This requires
234 whitening of the noise present in the images and resolution-dependent scaling of the reference
235 projections to match the signal in the noise-whitened images. Both can be achieved if the spectral
236 signal-to-noise ratio (SSNR) of the data is known. As part of a 3D reconstruction, Frealign
237 calculates the resolution-dependent $PSSNR$, the radially averaged SSNR present in the particle
238 images before they are affected by the CTF (Sindelar and Grigorieff, 2012). Using $PSSNR$ and
239 the CTF determined for a particle, the SSNR in the particle image can be calculated as

240 $$SNR(\mathbf{g}) = PSSNR(g) \times CTF^2(\mathbf{g}) \quad (6)$$

(as before, $\mathbf{g}$ is the 2D reciprocal space coordinate and $g = |\mathbf{g}|$). Here, SNR is defined as the ratio of the variance of the signal and the noise. The Fourier transform $\mathcal{F}\{\tilde{\mathbf{X}}_i\}$ of the noise-whitened particle image $\tilde{\mathbf{X}}_i$ can then be calculated as

$$\mathcal{F}\{\tilde{\mathbf{X}}_i\}(\mathbf{g}) = \frac{\mathcal{F}\{\mathbf{X}_i\}(\mathbf{g})}{\sqrt{|\mathcal{F}\{\mathbf{X}_i\}|_r^2(g)}}\sqrt{1 + SNR(\mathbf{g})} \tag{7}$$

where $\mathcal{F}\{\mathbf{X}_i\}$ is the Fourier transform of the original image $\mathbf{X}_i$, $|\cdot|$ is the absolute value, and $|\mathcal{F}\{\mathbf{X}_i\}|_r^2$ is the radially averaged spectrum of the squared 2D Fourier transform amplitudes of image $\mathbf{X}_i$. To implement Eq. (7), a particle image is first divided by its amplitude spectrum, which includes power from both signal and noise, and then multiplied by a term that amplifies the image amplitudes according to the signal strength in the image. The reference projection $\mathbf{A}_i$ can be matched by calculating

$$\mathcal{F}\{\tilde{\mathbf{A}}_i\}(\mathbf{g}) = \frac{\mathcal{F}\{\mathbf{A}_i\}(\mathbf{g})}{\sqrt{|\mathcal{F}\{\mathbf{A}_i\}|_r^2(g)}}\sqrt{SNR(\mathbf{g})} \tag{8}$$

Eq. (8) scales the variance of the signal in the reference to be proportional to the measured signal-to-noise ratio in the noise-whitened images. The main term in the objective function $O(\phi)$ maximized in FrealignX is therefore given by the cross-correlation function

$$CC(\phi) = \frac{Re\left(\mathcal{F}_{R1,R3}\{\tilde{\mathbf{A}}_i(\phi)\}^* \mathcal{F}_{R1,R3}\{\tilde{\mathbf{X}}_i\}\right)}{\|\mathcal{F}_{R1,R3}\{\tilde{\mathbf{A}}_i(\phi)\}\|\|\mathcal{F}_{R1,R3}\{\tilde{\mathbf{X}}_i\}\|} \tag{9a}$$

where ϕ is a set of parameters describing the particle view, x,y position, magnification and defocus, $Re(\cdot)$ is the real part of a complex number, $\|\cdot\|$ is the Euclidean norm, i.e. the square root of the sum of the squared pixel values, and $\mathcal{F}_{R1,R3}\{\cdot\}^*$ is the conjugate complex value of the Fourier transform $\mathcal{F}_{R1,R3}\{\cdot\}$. The subscripts $R1$ and $R3$ specify the low- and high-resolution

limits of the Fourier transforms included in the calculation of Eq. (9a), as specified by the user. To reduce noise overfitting, the user has the option to specify also a resolution range in which the absolute value of the cross terms in the numerator of Eq. (9a) are used (Stewart and Grigorieff, 2004), instead of the signed values (option "Signed CC Resolution Limit" under "Expert Options" in the "Manual Refine" Action panel). In this case

$$CC(\phi) = \frac{Re\left(\mathcal{F}_{R1,R2}\{\widetilde{\mathbf{A}}_i(\phi)\}^* \mathcal{F}_{R1,R2}\{\widetilde{\mathbf{X}}_i\}\right) + \left|Re\left(\mathcal{F}_{R2,R3}\{\widetilde{\mathbf{A}}_i(\phi)\}^* \mathcal{F}_{R2,R3}\{\widetilde{\mathbf{X}}_i\}\right)\right|}{\|\mathcal{F}_{R1,R3}\{\widetilde{\mathbf{A}}_i(\phi)\}\| \|\mathcal{F}_{R1,R3}\{\widetilde{\mathbf{X}}_i\}\|} \quad (9b)$$

where $R2$ is specified by the "Signed CC Resolution Limit." The objective function also includes a term $R(\phi|\Theta)$ to restrain alignment parameters (Chen et al., 2009; Lyumkis et al., 2013; Sigworth, 2004), which currently only includes the x,y positions:

$$R(\phi|\Theta) = -\frac{\sigma^2}{M}\left(\frac{(x-\bar{x})^2}{2\sigma_x^2} + \frac{(y-\bar{y})^2}{2\sigma_y^2}\right) \quad (10)$$

where σ is the standard deviation of the noise in the particle image and Θ represents a set of model parameters including the average particle positions in a dataset $\bar{x}$ and $\bar{y}$, and the standard deviations of the x,y positions from the average values, σ_x and σ_y, and M is the number of pixels in the mask applied to the particle before alignment. The complete objective function is therefore

$$O(\phi) = CC(\phi) + R(\phi|\Theta) \quad (11)$$

The maximized values determined in a refinement are converted to particle scores by multiplication with 100.

CTF refinement FrealignX can refine the defocus assigned to each particle. This may be useful when particles have a size of about 400 kDa or larger. Depending on the quality of the sample and images, these particles may generate sufficient signal to yield per-particle defocus values

that are more accurate than the average defocus values determined for whole micrographs by CTFFIND4 (see above). Refinement is achieved by a simple one-dimensional grid search of a defocus offset applied to both defocus values determined in the 2D CTF fit obtained by CTFFIND4. FrealignX applies this offset to the starting values in a refinement, typically determined by CTFFIND4, and evaluates the objective function, Eq. (11), for each offset. The offset yielding the maximum is then used to assign refined defocus values. In a typical refinement, the defocus offset is searched in steps of 50 Å, in a range of ± 500 Å. In the case of β-galactosidase (see below), a single round of defocus refinement changed the defocus on average by 60 Å; the RMS change was 80 Å. The refinement produced a marginal improvement of 0.05 Å in the Fourier Shell Correlation (FSC) threshold of 0.143, suggesting that the defocus values determined by CTFFIND4 were already close to optimal. In a different dataset of rotavirus double layered particles, a single round of defocus refinement changed the defocus on average by 160 Å; the RMS change was 220 Å. In this case the refinement increased the resolution from ~3.0 Å to ~2.8 Å.

Masking FrealignX has a 3D masking function to help in the refinement of structures that contain significant disordered regions, such as micelles in detergent-solubilized membrane proteins. To apply a 3D mask, the user supplies a 3D map that will be binarized by setting to zero all voxel values less than or equal to zero, and all others to 1 to indicate that they are inside the masked region. A soft cosine-shaped falloff of specified width (e.g. 10 Å) is then applied to soften the edge of the masked region and avoid sharp edges when the mask is applied to a 3D reconstruction. Voxels of the masked reconstruction that fall outside the mask can be set to zero, or to a low-pass filtered version of the original density, optionally downweighted by multiplication by a scaling factor set by the user. At the edge of the mask, the low-pass filtered

density is blended with the unfiltered density inside the mask to produce a smooth transition. Figure 5 shows the result of masking the reconstruction of an ABC transporter associated with antigen processing (TAP, (Oldham et al., 2016)). The mask was designed to contain only density corresponding to protein and the outside density was low-pass filtered at 30 Å resolution and kept with a weight of 100% in the final masked reconstruction. The combination of masking and low-pass filtering in this case keeps a low-pass filtered version of the density outside the mask in the reconstruction, including the detergent micelle. Detergent micelles can be a source of noise in the particle images because the density represents disordered material. However, at low, 20 to 30 Å resolution, micelles generate features in the images that can help in the alignment of the particles. In the case of TAP, this masking helped obtain a reconstruction at 4 Å resolution (Oldham et al., 2016).

3D reconstruction In Frealign, a 3D reconstruction $\mathbf{V}_k$ of class average k and containing N images is calculated as (Lyumkis et al., 2013; Sindelar and Grigorieff, 2012)

$$\mathbf{V}_k = \mathcal{F}^{-1}\left\{\frac{\sum_{i=1}^{N}\frac{q_{ik}}{\sigma_i^2}\mathcal{R}(\phi_i, w_{ik}\cdot CTF_i\cdot\mathcal{F}\{\hat{\mathbf{X}}_i\})}{\sum_{i=1}^{N}\frac{q_{ik}}{\sigma_i^2}\mathcal{R}(\phi_i, w_{ik}\cdot CTF_i^2)+1/PSSNR_k}\right\} \quad (12)$$

where q_{ik} is the probability of particle i belonging to class k, σ_i is the standard deviation of the noise in particle image i, ϕ_i are its alignment parameters, w_{ik} the score-based weights (Eq. (14), see below), CTF_i the CTF of the particle image, $\mathcal{R}(\phi_i, \cdot)$ the reconstruction operator merging data into a 3D volume according to alignment parameters ϕ_i, $PSSNR$ the radially averaged particle SSNR derived from the FSC between half-maps (Sindelar and Grigorieff, 2012), $\hat{\mathbf{X}}_i$ noise-whitened image i, and $\mathcal{F}^{-1}\{\cdot\}$ the inverse Fourier transform. For the calculation of the 3D reconstructions, as well as 3D classification (see below) the particle images are not whitened

according to Eq. (7). Instead, they are whitened using the radially- and particle-averaged power spectrum of the background around the particles:

$$\mathcal{F}\{\hat{\mathbf{X}}_i\}(\mathbf{g}) = \frac{\mathcal{F}\{\mathbf{X}_i\}(\mathbf{g})}{\sqrt{|\mathcal{F}\{B(\mathbf{X}_i)\}|^2_r(g)}} \quad (13)$$

where $B(\mathbf{X}_i)$ is a masked version of image $\mathbf{X}_i$ with the area inside a circular mask centered on the particle replaced with the average values at the edge of the mask, and scaled variance to produce an average pixel variance of 1 in the whitened image $\hat{\mathbf{X}}_i$. Using the procedure in Eq. (13) has the advantage that whitening does not depend on the knowledge of the SSNR of the data, and reconstructions can therefore be calculated even when the SSNR is not known.

Score-based weighting In previous versions of Frealign, resolution-dependent weighting was applied to the particle images during reconstruction (the Frealign parameter was called "PBC", (Grigorieff, 2007)). The weighting function took the form of a B-factor dependent exponential that attenuates the image data at higher resolution. FrealignX still uses B-factor weighting but the weighting function is now derived from the particle scores (see above) as

$$w(score, \mathbf{g}) = e^{-\frac{BSC}{4}(score-\overline{score})g^2} \quad (14)$$

BSC converts the difference between a particle score and the average particle score, $\overline{score}$, into a B-factor. Setting BSC to zero will turn off score-based particle weighting. Typical values for BSC that produce reasonable discrimination between high-scoring and low-scoring particles are between 2 and 10 Å^2.

3D Classification FrealignX uses a maximum-likelihood approach for 3D classification (Lyumkis et al., 2013). Assuming that all images were noise-whitened according to Eq. (13), which scales the variance of each image such that the average standard deviation of the noise in a

pixel is 1, the probability density function (PDF) of observing image $\mathbf{X}_i$, given alignment parameters ϕ_i and reconstruction $\mathbf{V}_k$, is calculated as (Lyumkis et al., 2013)

$$\Gamma(\mathbf{X}_i|\phi_{ik},\mathbf{V}_k) = \left(\frac{1}{2\pi}\right)^{\widetilde{M}} \exp\left[-\frac{\|\hat{\mathbf{X}}_i - \wp(\mathbf{V}_k,\phi_{ik})\|^2_{\widetilde{M}}}{2}\right]\gamma(\phi_{ik}|\Theta_k). \quad (15)$$

As before, ϕ_{ik} are the alignment parameters (usually just Euler angles and x,y shifts) determined for image i with respect to class average k, $\wp$ is the projection operator producing an aligned 2D projection of reconstruction $\mathbf{V}_k$ according to parameters ϕ_{ik}, $\left\|\hat{\mathbf{X}}_i - \wp(\mathbf{V}_k,\phi_{ik})\right\|^2_{\widetilde{M}}$ is the sum of the squared pixel value differences between whitened image $\hat{\mathbf{X}}_i$ and the reference projection inside a circular mask defining the area of the particle with user-defined diameter, $\widetilde{M}$ is the number of pixels inside this mask, and $\gamma(\phi_{ik}|\Theta_k)$ is a hierarchical prior describing the probability of observing alignment parameters ϕ_{ik} given model parameters Θ_k (see Eq. (10)). Given the joint probability, Eq. (15), determined in a refinement, the probability q_{ik} of particle i belonging to class k can be updated as (Lyumkis et al., 2013)

$$q_{ik} = \frac{\Gamma(\mathbf{X}_i|\Theta_{ik},\mathbf{V}_k)\pi_k}{\sum_{k=1}^{K}\Gamma(\mathbf{X}_i|\Theta_{ik},\mathbf{V}_k)\pi_k} \quad (16)$$

where the summation in the denominator is taken over all classes and the average probabilities π_k for a particle to belong to class k are given by the average values of q_{ik} determined in a prior iteration, calculated for the entire dataset of N particles:

$$\pi_k = \frac{1}{N}\sum_{i=1}^{N} q_{ik} \quad (17)$$

An example of 3D classification is shown in Figure 6 for F_1F_O-ATPase, revealing different conformational states of the γ subunit (Zhou et al., 2015).

Focused classification 3D classification can be improved by focusing on conformationally- or compositionally-variable regions of the map. To achieve this, a mask is applied to the particle images and reference projections, the area of which is defined as the projection of a sphere with user-specified center (within the 3D reconstruction) and radius. This 2D mask is therefore defined independently for each particle, as a function of its orientation. When using focused classification, $\widetilde{M}$ in Eq. (15) is adjusted to the number of pixels inside the projected mask and the sum of the squared pixel value differences in Eq. (15) is limited to the area of the 2D mask. By applying the same mask to image and reference, only variability inside the masked region is used for 3D classification. Other regions of the map are ignored, leading to a "focusing" on the region of interest. The focused mask also excludes noise contained in the particle images outside the mask and therefore improves classification results that often depend on detecting small differences between particles and references. A typical application of a focused mask is in the classification of ribosome complexes that may exhibit localized conformational and/or compositional variability, for example the variable conformations of an IRES (Abeyrathne et al., 2016) or different states of tRNA accommodation (Loveland et al., 2017).

Likelihood-based blurring In some cases, the convergence radius of refinement can be improved by blurring the reconstruction according to a likelihood function. This procedure is similar to the maximization step in a maximum likelihood approach (Scheres, 2012). The likelihood-blurred reconstruction is given by

$$\mathbf{V}_k^n = \frac{\sum_{i=1}^{N} \frac{1}{\sigma_i^2} \int_{\phi_{axy}} \Gamma\left(\mathbf{X}_i \middle| \phi_i, \mathbf{V}_k^{n-1}\right) \mathcal{R}(\phi_i, w_i \cdot CTF_i \cdot \mathbf{X}_i) d\phi_{axy}}{\sum_{i=1}^{N} \frac{q_{ik}}{\sigma_i^2} \mathcal{R}(\phi_i, w_i \cdot CTF_i^2) + 1/PSSNR_k} \quad (18)$$

where, in the case of FrealignX, $\phi_{\alpha xy}$ only includes the x,y particle positions and in-plane rotation angle α, which are a subset of the alignment parameters ϕ_i, and $\mathbf{V}_k^{n-1}$ is the reconstruction from an earlier refinement iteration. As before, $\Gamma(\mathbf{X}_i|\phi_i, \mathbf{V}_k^{n-1})$ is the probability of observing image i, given alignment parameters ϕ_i and reconstruction V_k^{n-1}. Integration over these three parameters can be efficiently implemented and, therefore, does not produce a significant additional computational burden.

Resolution assessment The resolution of reconstructions generated by FrealignX is assessed using the FSC criterion (Harauz and van Heel, 1986). FSC curves in *cis*TEM are calculated using two reconstructions ("half-maps") calculated either from the even-numbered and odd-numbered particles, or by dividing the dataset into 100 equal subsets and using the even- and odd-numbered subsets to calculate the two reconstructions (in the *cis*TEM GUI, the latter is always used). The latter method has the advantage that accidental duplication of particles in a stack is less likely to affect the FSC calculation. All particles are refined against a single reference and, therefore, the calculated FSC values may be biased towards higher values (Grigorieff, 2000; Stewart and Grigorieff, 2004). This bias extends slightly beyond the resolution limit imposed during refinement, by approximately $2/D_{mask}$, where D_{mask} is the mask radius used to mask the reconstructions (see above). During auto-refinement (see below), the resolution limit imposed during refinement is carefully adjusted to stay well below the estimated resolution of the reconstruction and the resolution estimate is therefore unbiased (Scheres and Chen, 2012). However, users have full control over all parameters during manual refinement and will have to make sure that they do not bias the resolution estimate by choosing a resolution limit that is close to, or higher than, the estimated resolution of the final reconstruction. Calculated FSC curves are

406 smoothed using a Savitzky–Golay cubic polynomial that reduces the noise often affecting FSC
407 curves at the high-resolution end.

408 The FSC calculated between two density maps is dependent on the amount of solvent included
409 inside the mask applied to the maps. A larger mask that includes more solvent background will
410 yield lower FSC values than a tighter mask. To obtain an accurate resolution estimate in the
411 region of the particle density, one possibility is to apply a tight mask that closely follows the
412 boundary of the particle. This approach bears the risk of generating artifacts because the particle
413 boundary is not always well defined, especially when the particle includes disordered domains
414 that generate weak density in the reconstruction. The approach in Frealign avoids tight masking
415 and instead calculates an FSC curve using generously masked density maps, corrected for the
416 solvent content inside the mask (Sindelar and Grigorieff, 2012). The corrected FSC curve is
417 referred to as $Part\_FSC$ and is calculated from the uncorrected FSC_{uncor} as (Oldham et al.,
418 2016)

419 $$Part\_FSC_{half-maps} = \frac{fFSC_{uncor}}{1+(f-1)FSC_{uncor}}, \quad (19)$$

420 where f is the ratio of mask volume to estimated particle volume. The particle volume can be
421 estimated from its molecular mass M_w as $\frac{\text{Å}^3}{0.81\text{Da}} M_w$ (Matthews, 1968). Eq. (19) assumes that
422 both maps have similar SSNR values, as is normally the case for the two reconstructions
423 calculated from two halves of the dataset, indicated by the subscript $half - maps$. If one of the
424 maps does not contain noise from solvent background, for example when calculating the FSC
425 between a reconstruction and a map derived from an atomic model, the solvent-corrected FSC is
426 given as

$$Part\_FSC_{model-map} = \sqrt{\frac{fFSC_{uncor}^2}{1+(f-1)FSC_{uncor}^2}} \quad (20)$$

Speed optimization FrealignX has been optimized for execution on multiple CPU cores. Apart from using optimized library functions for FFT calculation and vector multiplication (Intel Math Kernel Library), the processing speed is also increased by on-the-fly cropping in real and reciprocal space of particle images and 3D reference maps. Real-space cropping reduces the interpolation accuracy in reciprocal space and is therefore limited to global parameter searches that do not require the highest accuracy in the calculation of search projections. Reciprocal-space cropping is used whenever a resolution limit is specified by the user or in an automated refinement (ab-initio 3D reconstruction and auto-refinement). For the calculation of in-plane rotated references, reciprocal-space padding is used to increase the image size four-fold, allowing fast nearest-neighbor resampling in real space with sufficient accuracy to produce rotated images with high fidelity.

Ab-initio 3D reconstruction

Ab-initio reconstruction offers a convenient way to proceed from single particle images to a 3D structure when a suitable reference is not available to initialize 3D reconstruction and refinement. Different ab-initio methods have been described (Hohn et al., 2007; Punjani et al., 2017; Reboul et al., 2018) and *cis*TEM's implementation follows a strategy published originally by (Grigorieff, 2016). It is based on the premise that iterative refinement of a reconstruction initialized with random angular parameters is likely to converge on the correct structure if overfitting is avoided and the refinement proceeds in small steps to reduce the chance of premature convergence onto

an incorrect structure. The procedure is implemented as part of *cis*TEM's GUI and uses FrealignX to perform the refinements and reconstructions.

After initialization with random angles, *cis*TEM performs a user-specified number of global alignment parameter searches, recalculating the reconstruction after each search and applying an automatic masking procedure to it before the next global search. Similar to 2D classification (see above), only a randomly selected subset of the data is used in each iteration and the resolution limit applied during the search is increased with every iteration. The number of iterations n defaults to 40, the starting and final resolution limits R_{start} and R_{finish} default to 20 Å and 8 Å, respectively, and the starting and final percentage of included particles in the reconstruction, p_{start} and p_{finish} default to $2500K/N$ and $10{,}000K/N$, respectively (results larger than 1 are reset to 1), with K the number of 3D classes to be calculated as specified by the user, and N the number of particles in the dataset. If symmetry is being applied, N is replaced by NO_{sym} where O_{sym} is the number of asymmetric units present in one particle. The resolution limit is then updated in each iteration l as in Eq. (2), and the percentage is updated as

$$p = p_{start} + l\left(p_{finish} - p_{start}\right)/(n-1), \tag{21}$$

again resetting results larger than 1 to 1. *cis*TEM actually performs a global search for a percentage $3p$ of the particle stack, i.e. three times as many particles as are included in the reconstructions for each iteration. The particles included in the reconstructions are then chosen to be those with the highest scores as calculated by FrealignX.

The global alignment parameters are performed using the "general" FrealignX procedure with the following changes. Firstly, the $PSSNR$ is not directly estimated from the FSC calculated at each round. Instead, for the first 3 iterations, a default $PSSNR$ is calculated based on the

molecular weight. From the 4th iteration onwards, the $PSSNR$ is calculated from the FSC, however if the calculated $PSSNR$ is higher than the default $PSSNR$, the default $PSSNR$ is taken instead. This is done in order to avoid some of the overfitting that will occur during the refinements. Secondly, during a normal global search the top h (where h defaults to 20) results of the grid search are locally refined, and the best locally refined result is taken. In the ab-initio procedure, however, the result of the global search for a given particle image is taken randomly from all results that have a score which lies in the top 15% of the difference between the worst score and the best score.

During the reconstruction steps, the calculated σ for each particle is reset to 1 prior to 3D reconstruction and score weighting is disabled. This is done because the σ and score values are not meaningful until an approximately correct solution is obtained.

The reconstructions are automatically masked before each new refinement iteration to suppress noise features that could otherwise be amplified in subsequent iterations. The same masking procedure is also applied during auto-refinement (see below). It starts by calculating the density average $\bar{\rho}$ of the reconstruction and resetting all voxel values below $\bar{\rho}$ to $\bar{\rho}$. This thresholded reconstruction is then low-pass filtered at 50 Å resolution and turned into a binary mask by setting densities equal or below a given threshold t to zero and all others to 1. The threshold is calculated as

$$t = \bar{\rho}_{filtered} + 0.03\left(\bar{\rho}_{\max\_500} - \bar{\rho}_{filtered}\right) \quad (22)$$

where $\bar{\rho}_{filtered}$ is the density average of the low-pass filtered map and $\bar{\rho}_{\max\_500}$ is the average of the 500 highest values in the filtered map. The largest contiguous volume in this binarized map is then identified and used as a mask for the original thresholded reconstruction, such that all

voxels outside of this mask will be set to $\bar{\rho}$. Finally, a spherical mask, centered in the reconstruction box, is applied by resetting all densities outside the mask to zero.

The user has the option to repeat the ab-initio procedure multiple times using the result from the previous run as the starting map in each new run, to increase the convergence radius if necessary. In the case of symmetric particles, the default behavior is to perform the first 2/3^{rds} of the iterations without applying symmetry. The non-symmetrized map is then aligned to the expected symmetry axes and the final 1/3rd of the iterations are carried out with the symmetry applied. This default behavior can be changed by the user such that symmetry is always applied, or is never applied.

Alignment of the model to the symmetry axes is achieved using the following process. First, 3 direction are chosen. These directions are kept constant through the rest of the process. Next, a brute force grid search over rotations around the x, y and z axes is set up. At each position on the grid the 3D map is rotated using the current x, y and z parameters, and then its projection along the 1st direction is calculated. The symmetry-related projections for this direction are then calculated, and for each one a cross-correlation map is calculated using the original projection as a reference, and the peak within this map is found. This process is then repeated for the other 2 directions and the sum of all of the peak heights across all directions is calculated. The x,y,z rotations that result in the largest sum of all peaks is taken as the final rotation result. Shifts for this rotation are then calculated based on the found 2D x,y shifts between the initial and symmetry related projections, with the z shift being set to 0 for C symmetries. The symmetry alignment is also included as a command-line program, which can be used to align a volume to the symmetry axis when the ab-initio is carried out in C1 only, or when using a reference obtained by some other means.

Automatic refinement

Like ab-initio 3D reconstruction, auto-refinement makes use of randomly selected subsets of the data and of increasing resolution limit as refinement proceeds. However, unlike the ab-initio procedure, the percentage of particles p_l and the resolution limit R_l used in iteration l depend on the resolution of the reconstructions estimated on iteration $l-1$. When the estimated resolution improved in the previous cycle,

$$p_l = \max[p_R, p_{l-1}] \tag{23}$$

with

$$p_R = 8000Ke^{75/R_{l-1}^2}/N \tag{24}$$

where K is the number of 3D classes to be calculated and N the number of particles in the dataset. As before, if the particle has symmetry, N is replaced by NO_{sym} where O_{sym} is the number of asymmetric units present in one particle. If the calculated p_l exceeds 1, it is reset to 1. The resolution limit is estimated as

$$R = FSC_{0.5} - 2/D_{mask} \tag{25}$$

where $FSC_{0.5}$ is the point at which the FSC, unadjusted for the solvent within the mask (see above) crosses the 0.5 threshold and D_{mask} is the user-specified diameter of the spherical mask applied to the 3D reference at the beginning of each iteration, and to the half-maps used to calculate the FSC. The term $2/D_{mask}$ accounts for correlations between the two half-maps due to the masking (see above). When the resolution did not improve in the previous iteration,

$$p_l = 1.5 p_{l-1} \quad (26)$$

(reset to 1 if resulting in a value larger than 1). At least five refinement iterations are run and refinement stops when p_l reaches 1 (all particles are included) and there was no improvement in the estimated resolution for the last three iterations.

If multiple classes are refined, the resolution limit in Eq. (25) is set independently for each class, however the highest resolution used for classification is fixed at 8 Å. At least nine iterations are run and refinement stops when p_l reaches 1, the average change in the particle occupancy in the last cycle was 1% or less, and there was no improvement in the estimated resolution for the last three iterations.

In a similar manner to the ab-initio procedure, σ values for each particle are set to 1 and score weighting is turned off. This is done until the refinement resolution is better than 7 Å, at which point it is assumed the model is of a reasonable quality.

Map sharpening

Most single-particle reconstructions require some degree of sharpening that is usually achieved by applying a negative B-factor to the map. *cis*TEM includes a map sharpening tool that allows the application of an arbitrary B-factor. Additionally, maps can be sharpened by whitening the power spectrum of the reconstruction beyond a user-specified resolution (the default is 8 Å). The whitening amplifies terms at higher resolution similar to a negative B-factor but avoids the over-amplification at the high-resolution end of the spectrum that sometimes occurs with the B-factor method due to its exponential behavior. Whitening is applied after masking of the map, either with a hollow spherical mask of defined inner and outer radius, or with a user-defined mask

557 supplied as a separate 3D volume. The masking removes background noise and makes the
558 whitening of the particle density more accurate. Both methods can be combined in *cis*TEM,
559 together with a resolution limit imposed on the final reconstruction. The whitened and B-factor-
560 sharpened map can optionally be filtered with a figure-of-merit curve calculate using the FSC
561 determined for the reconstruction (Rosenthal and Henderson, 2003; Sindelar and Grigorieff,
562 2012).

563

564 *GUI design and workflow*

565 *cis*TEM's GUI required extensive development because it is an integral part of the processing
566 pipeline. GUIs have become more commonplace in cryo-EM software tools to make them more
567 accessible to users (Conesa Mingo et al., 2018; Desfosses et al., 2014; Punjani et al., 2017;
568 Scheres, 2012; Tang et al., 2007). Many of the interfaces are designed as so-called wrappers of
569 command-line driven tools, i.e. they take user input and translate it into a command line that
570 launches the tool. Feedback to the user takes place by checking output files, which are also the
571 main repository of processing results, such as movie frame alignments, image defocus
572 measurements and particle alignment parameters. As processing strategies become more
573 complex and the number of users new to cryo-EM grows, the demands on the GUI increase in
574 the quest for obtaining the best possible results. Useful GUI functions include guided user input
575 (so-called wizards) that adjust to specific situations, graphical presentation of relevant results,
576 user interaction with running processes to allow early intervention and make adjustments, tools
577 to manipulate data (e.g. masking), implementation of higher-level procedures that combine more
578 primitive processing steps to achieve specific goals, and a global searchable database that keeps
579 track of all processing steps and result. While some of these functions can be or have been

implemented in wrapper GUIs, the lack of control of these GUIs over the data and processes makes a reliable implementation more difficult. For example, keeping track of results from multiple processing steps, some of them perhaps repeated with different parameters or run many times during an iterative refinement, can become challenging if each step produces a separate results file. Communicating with running processes via files can be slow and is sometimes unreliable due to file system caching. Communication via files may complicate the implementation of higher-level procedures, which rely on the parsing of results from the more primitive processing steps.

The *cis*TEM GUI is more than a wrapper as it implements some of the new algorithms in the processing pipeline directly, adjusting the input of running jobs as the refinement proceeds. It enables more complex data processing strategies by tracking all results in a single searchable database. All processing steps are run and controlled by the GUI, which communicates with master and slave processes through TCP/IP. *cis*TEM uses a SQL database to store all results (except image files), offers input functions that guide the user or set appropriate defaults, and implements more complex procedures to automate processing where possible. *cis*TEM's design is flexible to allow execution in many different environments, including single workstations, multiple networked workstations and large computer clusters.

User input and the display of results is organized into different panels that make up *cis*TEM's GUI, each panel dedicated to specific processing steps (for examples, see Figure 1, 3, 4). This design guides users through a standard workflow that most single particle projects follow: movie alignment, CTF determination, particle picking, 2D classification, 3D reconstruction, refinement and classification, and sharpening of the final reconstructions. Three types of panels exist, dealing with Assets, Actions and Results. Assets are mostly data that can be used in processing

steps called Actions. They include Movies, Images, Particle Positions and Volumes. One type of Asset, a Refinement Package, defines the data and parameters necessary to carry out refinement of a 3D structure (or a set of structures if 3D classification is done), it contains a particle stack, as well as information about the sample (e.g. particle size and molecular weight) along with parameters for each particle (e.g. orientations and defocus values). Actions comprise the above mentioned workflow steps, with additional options for ab-initio 3D reconstruction, as well as automatic and manual 3D refinement to enable users to obtain the best possible results from their data. The results of most of these Actions are stored in the database and can be viewed in the related Results panels, which display relevant data necessary to evaluate the success of each processing step. Movie alignment, 3D refinement and reconstruction also produce new Image and Volume Assets, respectively.

Importing or generating new Assets is accomplished by wizards that solicit the necessary user input and perform checks were possible to avoid nonsensical input. In the more complex case of creating a new Refinement Package Asset, the wizard allows the specification of input data, for example based on particle picking results or the selection of 2D and 3D classes. Once an Action has been launched, results are displayed as soon as they become available, together with an overall progress bar, giving users an estimate of how long a processing step will take and of whether the results are as expected. If desired, an Action can be aborted and restarted with a different set of parameters, or the Action can be run again after regular termination to test different parameters. In the latter case, all prior results remain accessible and users can specify those to be used for the next step in the workflow. This provides users with the flexibility to pick and choose the best results in cases where different parts of a dataset require different settings to yield optimal results.

626

627 *Parallelization*

628 *cis*TEM uses a home-grown scheme to accelerate processing in parallel environments. Image
629 processing of single-particle data is an embarrassingly parallel problem, i.e. the parallelization of
630 most tasks can be achieved simply by dividing the data to be processed into smaller chunks that
631 are each processed by a separate program thread, without the need for inter-process
632 communication. Only certain steps require merging of data, such as the calculation of a 3D
633 reconstruction from the entire dataset. *cis*TEM parallelizes processing steps by running multiple
634 instances of the same program, each dealing with a subset of the data, then directly
635 communicating with the launched processes over TCP/IP sockets. This enables the *cis*TEM GUI
636 to distribute jobs and receive results in real time. Communication is directed through a
637 "manager" process, which enables jobs to be run on a cluster, while the GUI itself can run on a
638 local workstation

639 Another advantage of using a home-grown scheme over existing schemes (e.g. MPI) occurs
640 when jobs are run on a multi-node computing cluster. In this case, jobs will complete even if the
641 full number of requested processors is not available. For example, if a user requests 300 CPUs
642 for a processing step but only 100 are available, *cis*TEM launches 300 jobs of which 200 will
643 remain in the job scheduler's queue. Data processing starts immediately with the 100 jobs that
644 are allowed to run and will complete even if the remaining jobs never run. In such a scenario, an
645 MPI-based job could only run when 300 CPUs become available, potentially delaying execution.
646 In the few cases were a step requires merging of an entire dataset, for example in a 3D
647 reconstruction, parallelization is achieved by calculating multiple intermediate 3D
648 reconstructions for subsets of the data, dumping the intermediate reconstructions to disk and

649 merging them after all reconstruction jobs have finished. It can therefore help to designate a fast
650 disk as a scratch disk to allow rapid dumping and reading of the relatively large files (200 MB –
651 5 GB).

652

653 *Benchmarking with β-galactosidase*

654 A high-resolution dataset of β-galactosidase (Bartesaghi et al., 2015) has been used to
655 benchmark Relion 2 (Kimanius et al., 2016) and is also used here to illustrate the workflow of
656 *cis*TEM and assess the time for the completion of different processing steps. The entire dataset
657 was downloaded from the EMPIAR database (Iudin et al., 2016) and consists of 1539 movies
658 containing 38 frames, recorded at super-resolution on a K2 Summit camera (Gatan, Inc.,
659 Pleasanton, CA) and stored locally as tif files using LZW compression (the conversion to tiff and
660 compression was performed by mrc2tif (Mastronarde and Held, 2017)). The pixel size of the
661 super-resolution frames was 0.3185 Å, and images were binned to a pixel size of 0.75 Å after
662 movie processing. For 2D classification and ab-initio 3D reconstruction, particles were boxed
663 using 384 x 384 pixel boxes. For auto- and manual refinement, the particles were re-boxed into
664 648 x 648 pixel boxes (boxing is part of the creation of Refinement Packages, see above). For all
665 processing steps, a Dell Precision T7910 workstation containing two E5-2699 v4 Xeon
666 processors with a total of 44 cores was used. Processing parameters were left on default settings
667 except for the CTF determination, which was performed at 3.5 Å resolution using the movie
668 averages instead of the frames, and particle picking, which used optimized parameters based on
669 previewing a few selected images (Figure 3). The data were stored on a local SSD Raid 0 disk
670 for fast access. Table 1 lists the timings of the different processing steps using all 44 CPU cores.
671 Results obtained at different points in the workflow are shown in Figure 7.

672

Processing step	**Details**	**Time (hours)**
Movie processing	1539 movies, 38 frames, super-resolution	1.1
CTF determination	Using aligned movie average as input	0.1
Particle picking	131,298 particles	0.1
2D classification	50 classes, 28 selected with 119,523 particles	0.8
Ab initio 3D reconstruction	40 iterations	0.8
Auto refinement	8 iterations, final resolution 2.2 Å	1.4
Manual refinement	1 iteration (incl. defocus), final resolution 2.2 Å	0.4
Total		**4.7**

673

674 **Table 1** Benchmarking of *cis*TEM using a high-resolution dataset of β-galactosidase (Bartesaghi
675 et al., 2015).

676

677 **Discussion**

678 The implementation of a complete image processing workflow in *cis*TEM offers users a uniform
679 experience and guarantees smooth transitions between processing steps. It also helps developers
680 maintain the software as all the tools and algorithms are developed in-house.

The main focus of *cis*TEM is on the processing of single-particle cryo-EM data and high-resolution 3D reconstruction. Future releases of *cis*TEM may include particle-based movie alignment, support for helical particles, improved 3D masking tools, more reliable resolution and quality indicators, as well as miscellaneous tools such as the determination of the detective quantum efficiency of electron detectors.

Since *cis*TEM does not rely on third-party libraries, such as Python, MPI or CUDA, that usually have to be installed and compiled separately on the target system, ready-to-run binaries can be made available for download that are optimized for different architectures. Using the wxWidgets library also means that *cis*TEM can be compiled for different operating systems, including Linux, Windows and OSX. Using a configure script, different options for the fast Fourier transforms (FFTs) can be specified, including the FFTW (http://www.fftw.org) and Intel MKL (http://software.intel.com/en-us/mkl) libraries. The downloadable binaries are statically linked against the MKL library as this exhibits superior speeds compared to the FFTW library on Intel-based CPUs.

While the lack of support for GPUs simplifies the installation and execution of *cis*TEM, it can also be a limitation on workstations that are optimized for GPU-accelerated code. These workstations often do not have many CPU cores and, therefore, *cis*TEM will run significantly more slowly than code that can take advantage of the GPU hardware. Users who would like to run both CPU and GPU-optimized software may therefore have to invest in both types of hardware. One advantage of a CPU-optimized workstation, for example a 44-core Dell Precision workstation, is that it runs significantly more quietly under load than a GPU-optimized workstation, making it easy to locate it in regular office space.

Materials and Methods

*Development of cis*TEM

The entire *cis*TEM image processing package was written in C++ using the wxWidgets toolkit (http://wxwidgets.org) to implement the GUI, as well as the libtiff library (http://www.libtiff.org) to support the tiff image format, the SQLite library (https://sqlite.org) to implement the SQL database, and Intel's MKL library (http://software.intel.com/en-us/mkl) for the calculation of Fourier transforms and vector products. Optionally, *cis*TEM can also be linked against the FFTW library (http://www.fftw.org) to replace the MKL library. The code was written and edited using Eclipse (http://www.eclipse.org) and GitHub (http://github.com) was used for version control.

Image and data formats

*cis*TEM stores all image data using the MRC format (Crowther et al., 1996). Additionally, particle parameters can be imported from, and exported to Frealign (Grigorieff, 2016) and Relion (Scheres, 2012).

Acknowledgements

The authors are grateful for feedback from early testers of *cis*TEM, including Ruben Diaz-Avalos, Sarah Loerch, Priyanka Abeyrathne, Peter Rickgauer, Ben Himes, Andrei Korostelev, Anna Loveland, Gabriel Demo, Jue Chen, Dmitry Lyumkis, Hiro Furukawa, Wei Lu and Juan Du.

725

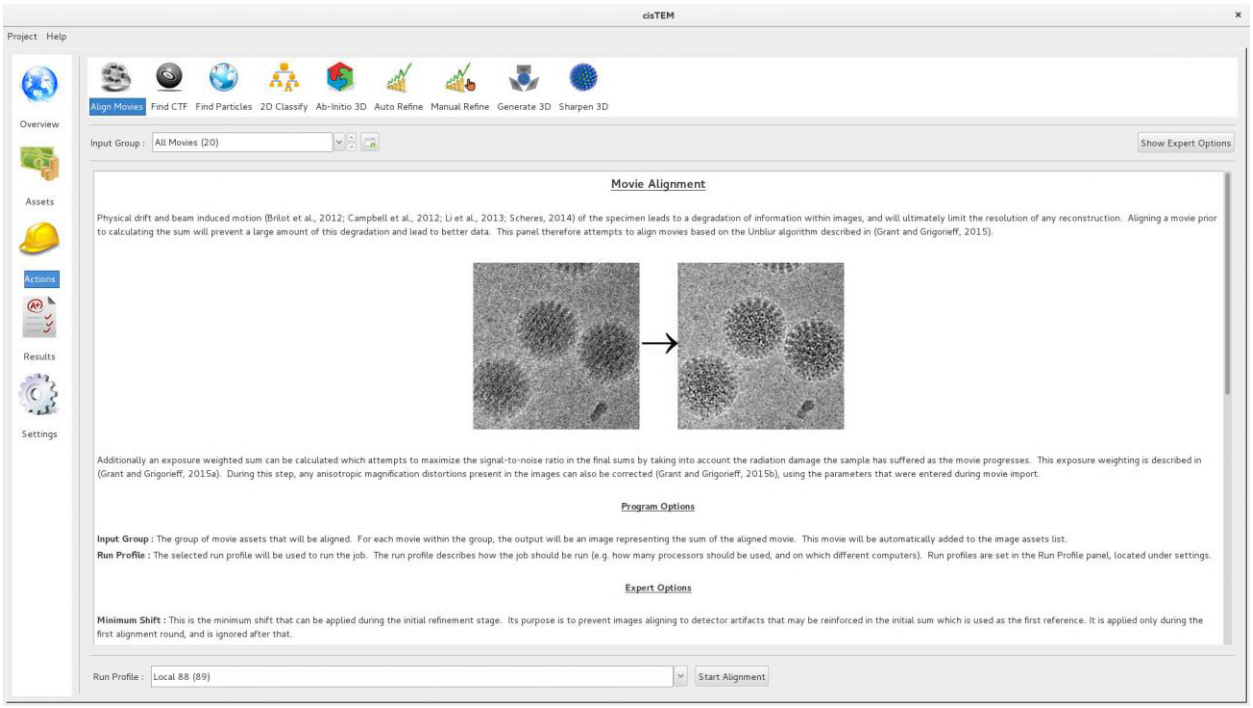
726 **Competing Interests**

727 The authors declare no competing interest.

728

729

730



731

732 **Figure 1** Movie alignment panel of the *cis*TEM GUI. All Action panels provide background

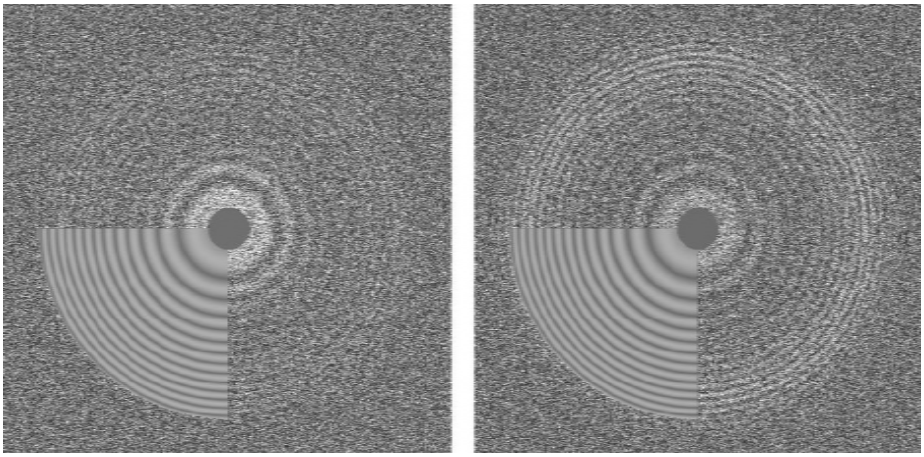
733 information on the operation they control, as well as a section with detailed explanations of all

734 user-accessible parameters. All Action panels also have an Expert Options section that exposes

735 additional parameters.

736

737



738

739 **Figure 2** Thon ring pattern calculated for micrograph "0000" of the high-resolution dataset of β-
740 galactosidase (Bartesaghi et al., 2015) used to benchmark *cis*TEM. The left pattern was
741 calculated from the average of aligned frames while the right pattern was calculated using the
742 original movie with 3-frame sub-averages. The pattern calculated using the movie shows
743 significantly stronger rings compared to the other pattern.

744

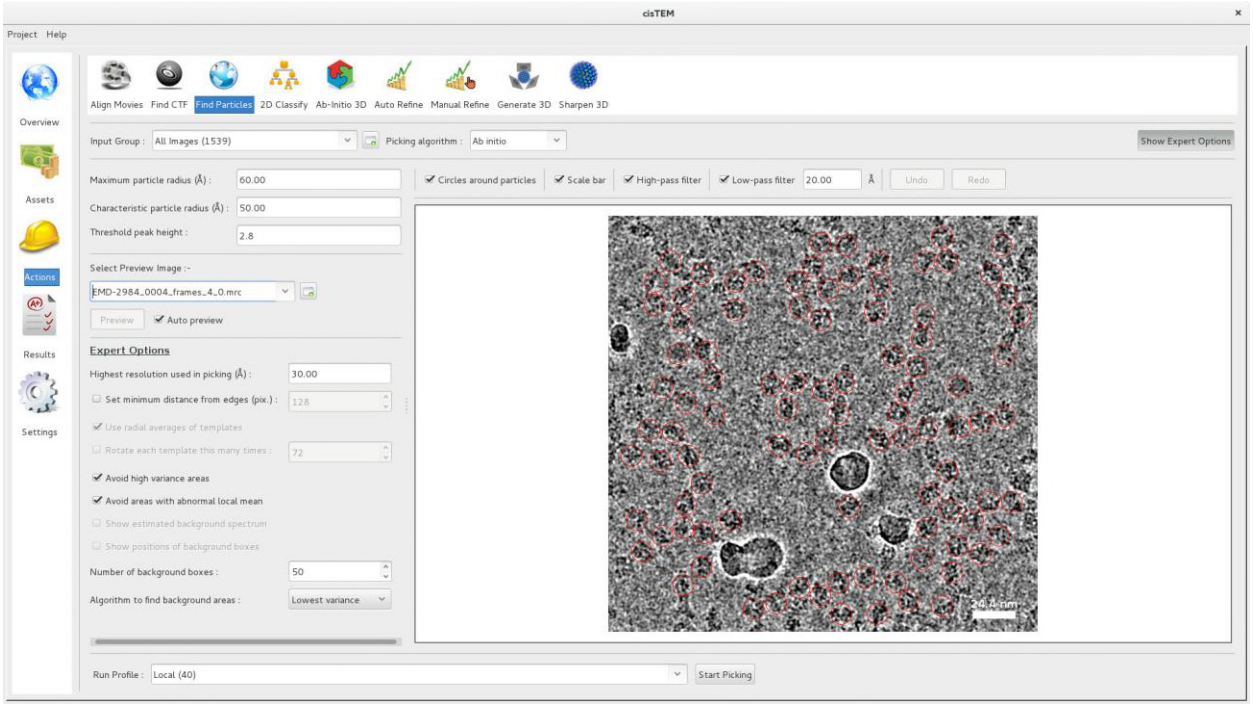
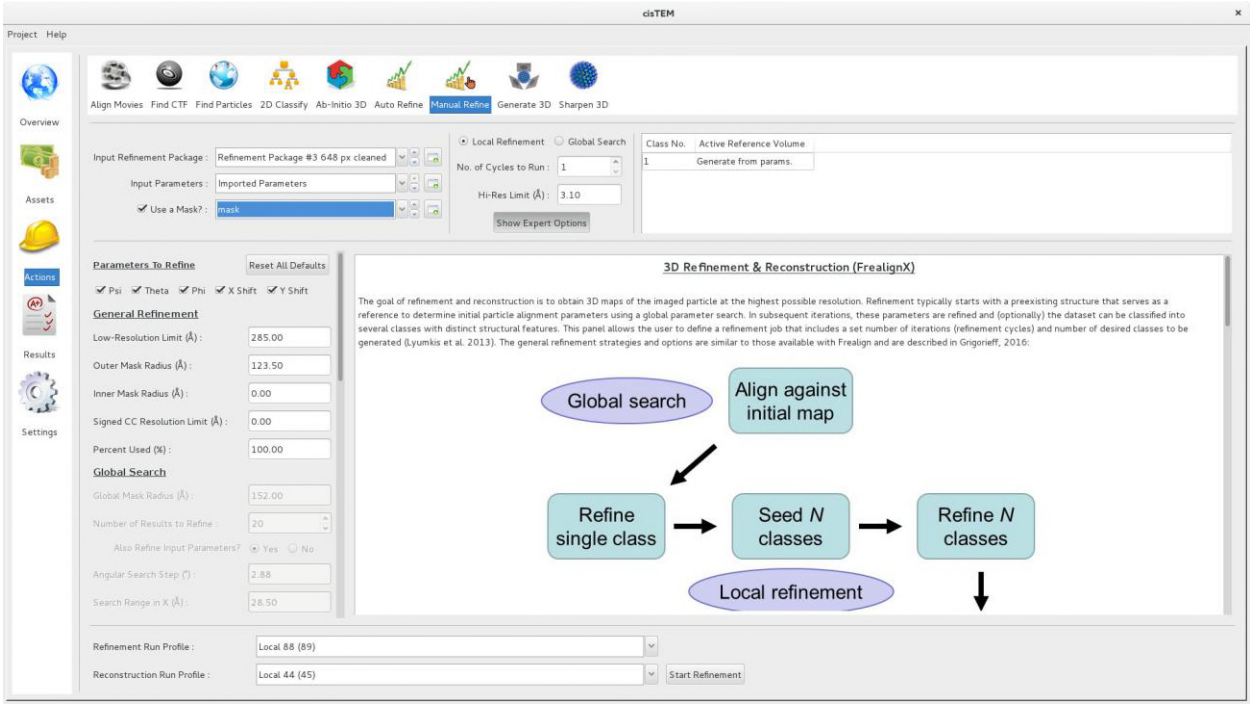


Figure 3 Particle picking panel of the *cis*TEM GUI. The panel shows the preview mode, which allows interactive tuning of the picking parameters for optimal picking. The red circles overlaying the image of the sample indicate candidate particles. The picking algorithm avoids areas of high variance, such as the ice contamination visible in the image.

752



753

754 **Figure 4** Manual refinement panel with Expert Options exposed. Most of the parameters needed

755 to run FrealignX can be accessed on this panel. The panel also allows application of a 3D mask,

756 which can be imported as a Volume Asset.

757

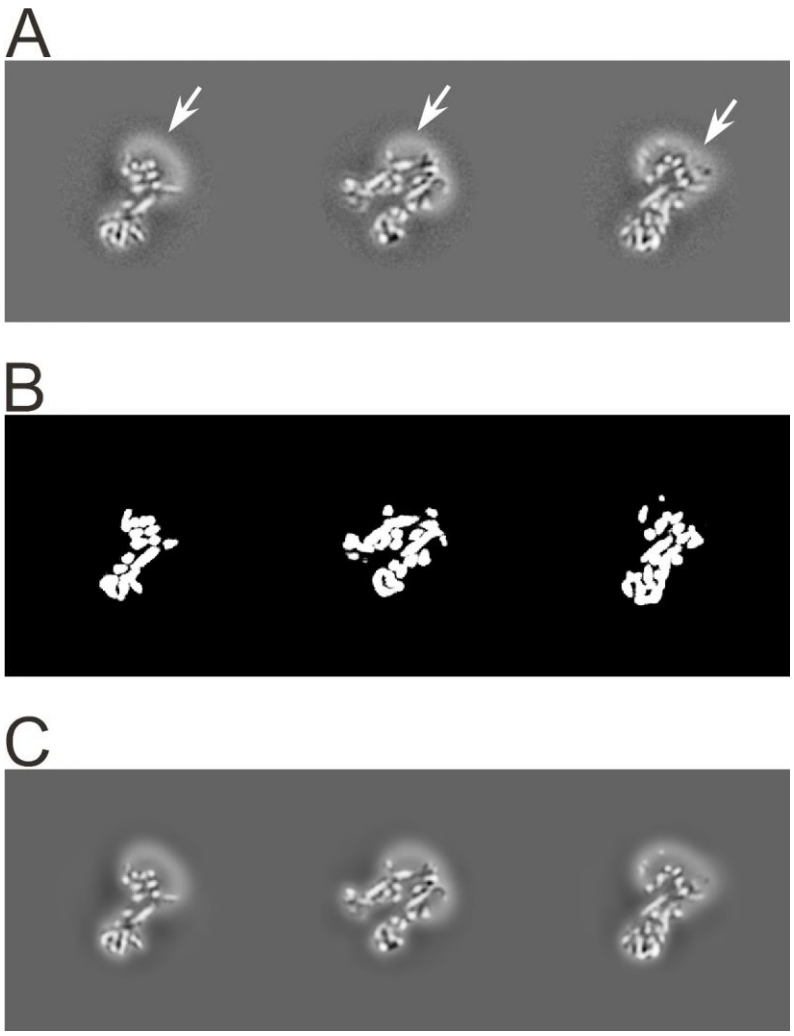
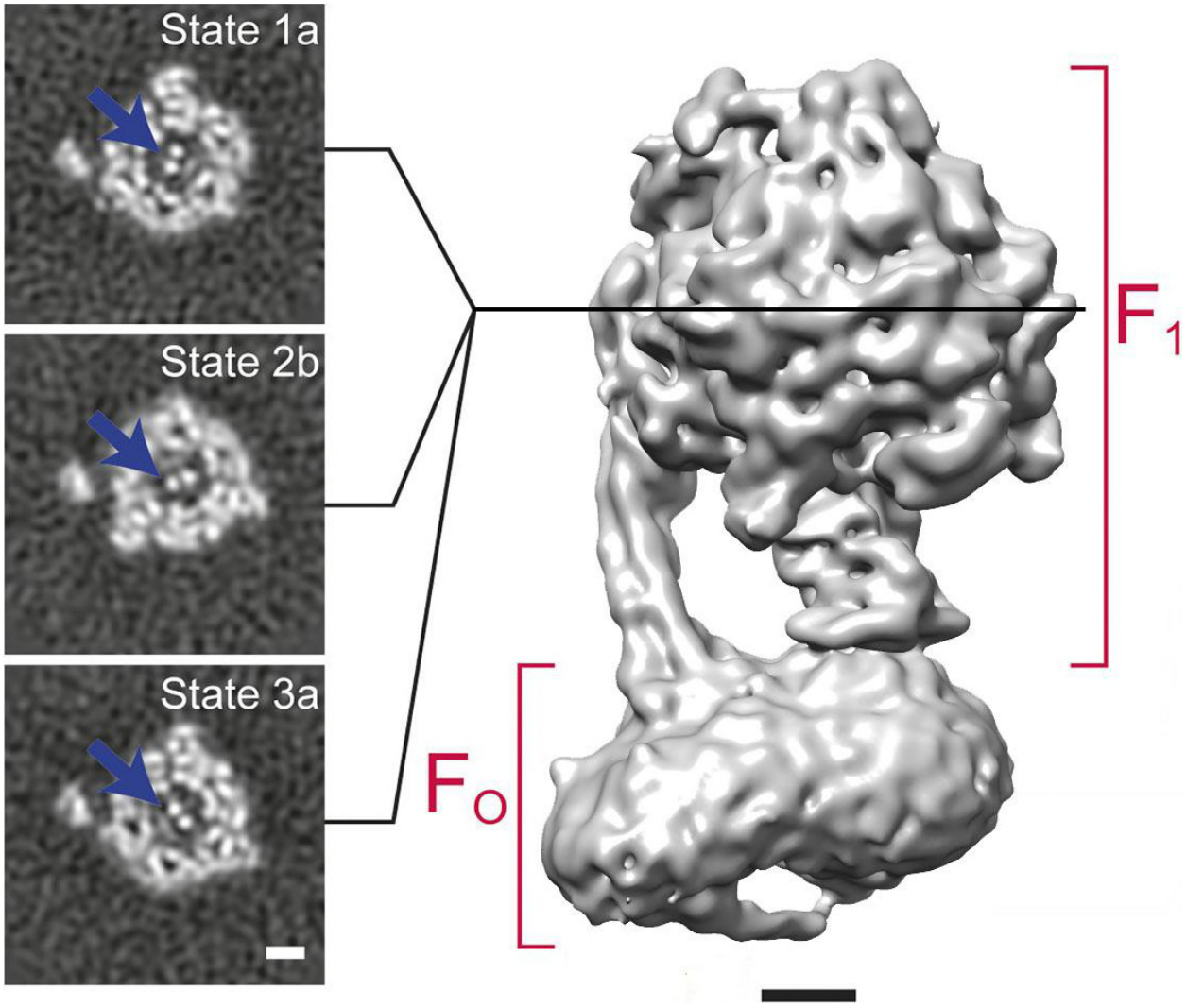


Figure 5 3D masking with low-pass filtering outside the mask. A) Orthogonal sections through the 3D reconstruction of the transporter associated with antigen processing (TAP), an ABC transporter (Oldham et al., 2016). Density corresponding to the protein, as well as the detergent micelle (n-Dodecyl b-D-maltoside; highlighted with arrows), is visible. B) Orthogonal sections through a 3D mask corresponding to the sections shown in A). The sharp edges of this mask are smoothed before the mask is applied to the map. C) Orthogonal sections through the masked 3D reconstruction. The regions outside the mask are low-pass filtered at 30 Å resolution to remove high-resolution noise from the disordered detergent micelle, but keeping its low-resolution signal to help particle alignment.

768



769

770 **Figure 6** 3D classification of a dataset of F_1F_O-ATPase, revealing different conformational states

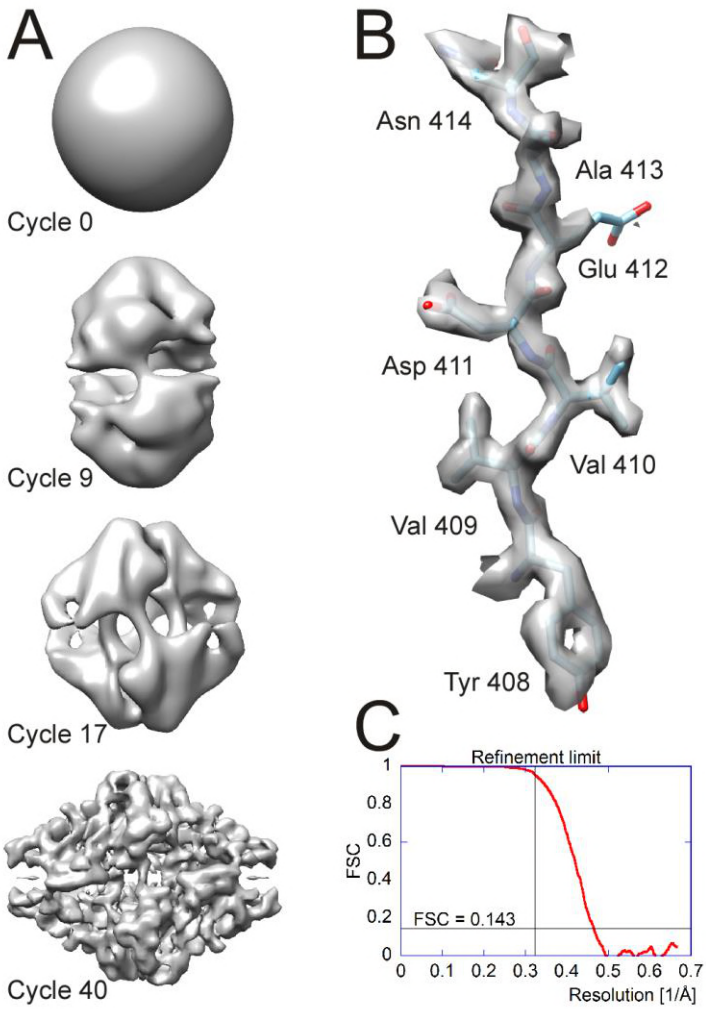
771 (Zhou et al., 2015). Sections through the F1 domain showing the γ subunit (arrows) in three

772 different states related by 120° rotations are shown on the left. A surface rendering of the map

773 corresponding to State 1a is shown on the right. Scale bars, 25 Å.

774

775



776

777 **Figure 7** Processing results of the β-galactosidase dataset (Bartesaghi et al., 2015) used to
778 benchmark *cis*TEM. A) Different stages of the ab-initio reconstruction procedure, starting from a
779 reconstruction from randomly assigned Euler angles. The process takes less than an hour to
780 complete on a high-end CPU-based workstation. B) High-resolution detail of the refined β-
781 galactosidase reconstruction with an average resolution of 2.2 Å, showing sidechain details for
782 most amino acids. C) FSC plot for the refined β-galactosidase reconstruction, adjusted for the
783 solvent background within the spherical mask applied to the half maps (Eq. (19)), and indicating
784 the resolution limit that was not exceeded during refinement, as well as the FSC = 0.143
785 threshold.

References

Abeyrathne, P.D., Koh, C.S., Grant, T., Grigorieff, N., and Korostelev, A.A. (2016). Ensemble cryo-EM uncovers inchworm-like translocation of a viral IRES through the ribosome. Elife *5*.

Bartesaghi, A., Matthies, D., Banerjee, S., Merk, A., and Subramaniam, S. (2014). Structure of beta-galactosidase at 3.2-A resolution obtained by cryo-electron microscopy. Proc Natl Acad Sci U S A *111*, 11709-11714.

Bartesaghi, A., Merk, A., Banerjee, S., Matthies, D., Wu, X., Milne, J.L., and Subramaniam, S. (2015). 2.2 A resolution cryo-EM structure of beta-galactosidase in complex with a cell-permeant inhibitor. Science *348*, 1147-1151.

Chen, J.Z., Settembre, E.C., Aoki, S.T., Zhang, X., Bellamy, A.R., Dormitzer, P.R., Harrison, S.C., and Grigorieff, N. (2009). Molecular interactions in rotavirus assembly and uncoating seen by high-resolution cryo-EM. Proc Natl Acad Sci U S A *106*, 10644-10648.

Cheng, Y., Grigorieff, N., Penczek, P.A., and Walz, T. (2015). A primer to single-particle cryo-electron microscopy. Cell *161*, 438-449.

Conesa Mingo, P., Gutierrez, J., Quintana, A., de la Rosa Trevin, J.M., Zaldivar-Peraza, A., Cuenca Alba, J., Kazemi, M., Vargas, J., Del Cano, L., Segura, J., *et al.* (2018). Scipion web tools: Easy to use cryo-EM image processing over the web. Protein Sci *27*, 269-275.

Crowther, R.A., Henderson, R., and Smith, J.M. (1996). MRC image processing programs. J Struct Biol *116*, 9-16.

805 Desfosses, A., Ciuffa, R., Gutsche, I., and Sachse, C. (2014). SPRING - an image processing
806 package for single-particle based helical reconstruction from electron cryomicrographs. J Struct
807 Biol *185*, 15-26.

808 Frank, J., Radermacher, M., Penczek, P., Zhu, J., Li, Y., Ladjadj, M., and Leith, A. (1996).
809 SPIDER and WEB: processing and visualization of images in 3D electron microscopy and
810 related fields. J Struct Biol *116*, 190-199.

811 Grant, T., and Grigorieff, N. (2015a). Automatic estimation and correction of anisotropic
812 magnification distortion in electron microscopes. J Struct Biol *192*, 204-208.

813 Grant, T., and Grigorieff, N. (2015b). Measuring the optimal exposure for single particle cryo-
814 EM using a 2.6 A reconstruction of rotavirus VP6. Elife *4*, e06980.

815 Grigorieff, N. (2000). Resolution measurement in structures derived from single particles. Acta
816 Crystallogr D Biol Crystallogr *56*, 1270-1277.

817 Grigorieff, N. (2007). FREALIGN: high-resolution refinement of single particle structures. J
818 Struct Biol *157*, 117-125.

819 Grigorieff, N. (2016). Frealign: An Exploratory Tool for Single-Particle Cryo-EM. Methods
820 Enzymol *579*, 191-226.

821 Harauz, G., and van Heel, M. (1986). Exact Filters for General Geometry 3-Dimensional
822 Reconstruction. Optik *73*, 146-156.

823 Henderson, R. (2013). Avoiding the pitfalls of single particle cryo-electron microscopy: Einstein
824 from noise. Proc Natl Acad Sci U S A *110*, 18037-18041.

825 Hohn, M., Tang, G., Goodyear, G., Baldwin, P.R., Huang, Z., Penczek, P.A., Yang, C., Glaeser,
826 R.M., Adams, P.D., and Ludtke, S.J. (2007). SPARX, a new environment for Cryo-EM image
827 processing. J Struct Biol *157*, 47-55.

828 Iudin, A., Korir, P.K., Salavert-Torres, J., Kleywegt, G.J., and Patwardhan, A. (2016). EMPIAR:
829 a public archive for raw electron microscopy image data. Nat Methods *13*, 387-388.

830 Kimanius, D., Forsberg, B.O., Scheres, S.H., and Lindahl, E. (2016). Accelerated cryo-EM
831 structure determination with parallelisation using GPUs in RELION-2. Elife *5*.

832 Loveland, A.B., Demo, G., Grigorieff, N., and Korostelev, A.A. (2017). Ensemble cryo-EM
833 elucidates the mechanism of translation fidelity. Nature *546*, 113-117.

834 Lyumkis, D., Brilot, A.F., Theobald, D.L., and Grigorieff, N. (2013). Likelihood-based
835 classification of cryo-EM images using FREALIGN. J Struct Biol *183*, 377-388.

836 Mastronarde, D.N., and Held, S.R. (2017). Automated tilt series alignment and tomographic
837 reconstruction in IMOD. J Struct Biol *197*, 102-113.

838 Matthews, B.W. (1968). Solvent content of protein crystals. J Mol Biol *33*, 491-497.

839 McDonough, R.N., and Whalen, A.D. (1995). Detection of Signals in Noise, 2nd edn (San
840 Diego: Academic Press).

841 McMullan, G., Faruqi, A.R., and Henderson, R. (2016). Direct Electron Detectors. Methods
842 Enzymol *579*, 1-17.

843 McMullan, G., Vinothkumar, K.R., and Henderson, R. (2015). Thon rings from amorphous ice
844 and implications of beam-induced Brownian motion in single particle electron cryo-microscopy.
845 Ultramicroscopy *158*, 26-32.

Mindell, J.A., and Grigorieff, N. (2003). Accurate determination of local defocus and specimen tilt in electron microscopy. J Struct Biol *142*, 334-347.

Moriya, T., Saur, M., Stabrin, M., Merino, F., Voicu, H., Huang, Z., Penczek, P.A., Raunser, S., and Gatsogiannis, C. (2017). High-resolution Single Particle Analysis from Electron Cryo-microscopy Images Using SPHIRE. J Vis Exp.

Oldham, M.L., Grigorieff, N., and Chen, J. (2016). Structure of the transporter associated with antigen processing trapped by herpes simplex virus. Elife *5*.

Penczek, P.A., Fang, J., Li, X., Cheng, Y., Loerke, J., and Spahn, C.M. (2014). CTER-rapid estimation of CTF parameters with error assessment. Ultramicroscopy *140*, 9-19.

Punjani, A., Rubinstein, J.L., Fleet, D.J., and Brubaker, M.A. (2017). cryoSPARC: algorithms for rapid unsupervised cryo-EM structure determination. Nat Methods *14*, 290-296.

Reboul, C.F., Eager, M., Elmlund, D., and Elmlund, H. (2018). Single-particle cryo-EM-Improved ab initio 3D reconstruction with SIMPLE/PRIME. Protein Sci *27*, 51-61.

Rohou, A., and Grigorieff, N. (2015). CTFFIND4: Fast and accurate defocus estimation from electron micrographs. J Struct Biol *192*, 216-221.

Roseman, A.M. (2004). FindEM--a fast, efficient program for automatic selection of particles from electron micrographs. J Struct Biol *145*, 91-99.

Rosenthal, P.B., and Henderson, R. (2003). Optimal determination of particle orientation, absolute hand, and contrast loss in single-particle electron cryomicroscopy. J Mol Biol *333*, 721-745.

866 Scheres, S.H. (2012). RELION: implementation of a Bayesian approach to cryo-EM structure
867 determination. J Struct Biol *180*, 519-530.

868 Scheres, S.H., and Chen, S. (2012). Prevention of overfitting in cryo-EM structure determination.
869 Nat Methods *9*, 853-854.

870 Scheres, S.H., Valle, M., Nunez, R., Sorzano, C.O., Marabini, R., Herman, G.T., and Carazo,
871 J.M. (2005). Maximum-likelihood multi-reference refinement for electron microscopy images. J
872 Mol Biol *348*, 139-149.

873 Sheth, L.K., Piotrowski, A.L., and Voss, N.R. (2015). Visualization and quality assessment of
874 the contrast transfer function estimation. J Struct Biol *192*, 222-234.

875 Sigworth, F.J. (2004). Classical detection theory and the cryo-EM particle selection problem. J
876 Struct Biol *145*, 111-122.

877 Sindelar, C.V., and Grigorieff, N. (2012). Optimal noise reduction in 3D reconstructions of
878 single particles using a volume-normalized filter. J Struct Biol *180*, 26-38.

879 Stewart, A., and Grigorieff, N. (2004). Noise bias in the refinement of structures derived from
880 single particles. Ultramicroscopy *102*, 67-84.

881 Subramaniam, S. (2013). Structure of trimeric HIV-1 envelope glycoproteins. Proc Natl Acad
882 Sci U S A *110*, E4172-4174.

883 Subramaniam, S., Earl, L.A., Falconieri, V., Milne, J.L., and Egelman, E.H. (2016). Resolution
884 advances in cryo-EM enable application to drug discovery. Curr Opin Struct Biol *41*, 194-202.

885 Tang, G., Peng, L., Baldwin, P.R., Mann, D.S., Jiang, W., Rees, I., and Ludtke, S.J. (2007).
886 EMAN2: an extensible image processing suite for electron microscopy. J Struct Biol *157*, 38-46.

887 Thon, F. (1966). Zur Defokussierungsabhängigkeit des Phasenkontrastes bei der
888 elektronenmikroskopischen Abbildung. Z Naturforschung *21a*, 476–478.

889 van Heel, M. (1982). Detection of Objects in Quantum-Noise-Limited Images. Ultramicroscopy
890 *7*, 331-341.

891 van Heel, M. (2013). Finding trimeric HIV-1 envelope glycoproteins in random noise. Proc Natl
892 Acad Sci U S A *110*, E4175-4177.

893 van Heel, M., Harauz, G., Orlova, E.V., Schmidt, R., and Schatz, M. (1996). A new generation
894 of the IMAGIC image processing system. J Struct Biol *116*, 17-24.

895 Zhang, K. (2016). Gctf: Real-time CTF determination and correction. J Struct Biol *193*, 1-12.

896 Zhou, A., Rohou, A., Schep, D.G., Bason, J.V., Montgomery, M.G., Walker, J.E., Grigorieff, N.,
897 and Rubinstein, J.L. (2015). Structure and conformational states of the bovine mitochondrial
898 ATP synthase by cryo-EM. Elife *4*, e10180.

899