EMAN2.1 Reconstruction Tutorial
Using the new Project Manager interface

This tutorial was updated in July 2013. It should not be used with versions of EMAN2 prior to 2.1. Many things have changed, so it is important not to use 2.0x with this tutorial.

➡ The main source for EMAN2 documentation is the Wiki at: http://blake.bcm.edu . There is also a Google Group for software support and discussions: http://groups.google.com/group/eman2 You don't have to join the group or Wiki to view existing documentation.

➡ **GUI Tips:** EMAN2 will work best with a 3-button scroll mouse, though there are alternatives using keyboard modifiers for people using one button mice on Macs.

- In most EMAN2 windows (2-D images, 3-D volumes, plots, etc.), the middle mouse button will open a 'control panel' for the widget with many options to control the display
- The right mouse button is used for panning in 2-D or 3-D image windows, and can be used to zoom (by shift+dragging), and to reset the zoom (clicking) in plot windows.
- The left mouse button has various purposes in various contexts.
- The scroll-wheel will generally act as a zoom. Use the control-panel for more precise control
- If you have a one button mouse, one of the modifier keys (depending on platform) combined with a mouse click will serve the same role as a middle-click. You may need to try them (alt, command, ctrl, shift) to discover which works on your machine.
- In the control panels, and other places in the EMAN2 interface you may encounter 'Value Sliders'. A slider is attached to a text-box with a number in it. Dragging the slider controls the number, and entering a number will change the slider. In addition, the text-box can be used to control the range of the slider and get more precise control. By typing '<value' or '>value' in the text box you can change the limits of the slider.

➡ Text you see in *italics* will generally be labels you will see in the GUI, such as buttons to press. Text you see in **bold**, are commands to be typed in. Items like: <param> are parameters you should fill in (without the < >). Items like: [param] are optional parameters (again, fill without the [ ]).

Introduction
EMAN2 can be used at many different levels ranging from high-level task-based workflow interface to writing code in C++ or Python. In this tutorial, we will be focusing primarily on the Project Manager interface (the older Workflow interface has been removed in EMAN 2.1). The Project Manager is relatively complete, but we have temporarily removed some items which have not been finished in 2.1 yet. More items will reappear as the final 2.1 release approaches.

Technically, EMAN2/SPARX supports all 3 of the major platforms: Linux, Mac and Windows. That said, development is almost entirely done on Linux and Mac, and few of the developers even have a Windows PC available for routine testing. Why ? All of the image processing we will discuss is very computationally intensive, and virtually any structure aiming at eventual publication will need at least 1,000 CPU-hr to complete, and many published structures at high resolution have used 100,000+. This sort of computing pretty much demands use of a Linux cluster to complete, thus the developers develop on the platform they must use for most of their work. We strive to make sure that Windows support is stable, at least for the major release versions, but encourage use of Linux/Mac when possible.

While we do have other tutorial datasets available for download, we use GroEL for this tutorial because:
- The entire tutorial can be completed in an afternoon on a reasonable desktop machine. Something with lower symmetry like a ribosome would involve more waiting.
- This data is sufficient to achieve a resolution of ~8 Å, at which alpha-helices are clearly visible.
- The structure is 'obvious' from the data, without any confusing projections. This makes it easier for novices to understand what's going on, though technically, ribosomes can be even easier to refine.

Overview/Quickstart

What follows is just an outline of the actual tutorial. Detailed descriptions for each point in the outline are later in this document. If you get stuck, please look up the corresponding section later in the document. Note that this methodology is appropriate for small-medium projects with up to a few hundred micrographs. It may still be reasonable for 1000-2000, but for extremely large projects, you may want to consider more automated methods even at the cost of accuracy:

Commands should be typed in a terminal window (not e2.py):

1. **tar xvzf workshop_beijing.tgz** (on Windows you will need to use an extraction program)

2. cd workshop_beijing

3. e2projectmanager.py
   a. NOTE - on Linux, the 2.1alpha1 release has a bug which will produce a lot of "X Error: BadDrawable (invalid Pixmap or Window parameter)" messages. They are just warnings and can be ignored. They should go away in alpha2.

4. *Project → edit project* (on the menu bar, not the task list)
   a. *Mass* = 800, *Cs* = 1.6, *voltage* = 300, *apix* =2.1 (for the GroEL data we are using)

➡ There is a choice: you can manually evaluate and import each micrograph, or just assume they are all good for now, and import the whole set. Select only one of step 6 or step 7

5. Raw Data → Import Micrographs
   a. <u>This option will just import all of the images into the project without any evaluation.</u>
   b. Browse for images in orig_micrographs, select all micrographs
   c. uncheck x-ray pixels and invert (the demo images are scanned film, not CCD/DDD)
   d. uncheck moverawdata
   e. Launch

** OR **

6. Raw Data → Evaluate & Import Micrographs
   a. <u>This option will allow you to examine the 2-D power spectrum of each image to decide whether to include it in the project or not.</u>
   b. Browse for images in orig_micrographs, select all micrographs
   c. *ac* = 10, *box* = 384

d. Launch → several windows will appear
e. Set Annotate to None, zoom in a bit on 2D power spectrum (right mouse drag)
f. uncheck *Invert* and *X-ray Pixels*
g. Zoom image display (wheel) so most of image visible
h. You may want to deselect squares (left click) with contamination or c-film
i. Adjust CTF parameters so defocus is roughly correct, and decide if image is 'good'
j. If image is 'good' press *Import*
k. That will move the image to micrographs folder and record estimated defocus
l. evaluate each image in the list this way
m. Exit by closing 'e2evalimage - Control Panel' window when done

7. *Particles → Coordinate Import*
a. This will load files containing already selected particle box locations for each micrograph. <u>We are skipping manual particle selection for now due to time constraints</u>
b. Use Browse and select all .box files in orig_box
c. *extension* → hdf
d. Launch → Very fast, don't expect to wait for it.

8. Particles → generate output
a. *Browse* and select all images with *stored boxes*
b. Box size 140
c. Launch - This may take a minute to finish. You can use the process monitor tool: , to tell when its done (may only work properly on Linux). Alternatively, use the browser: to monitor the particles folder (use to refresh the browser).

9. *CTF → automated fitting*
a. Check the 'allparticles' box. This will use everything in the 'particles' folder. No need to browse for files (leave that box blank)
b. *Oversamp* = 2, ac=10, other options should be ok
c. Launch

10.CTF→interactive tuning
a. leave 'allparticles' selected. Check the 'sortdefocus' box.
b. Fits should be good, but check to make sure defocus is right.

➡ The first time you run a GUI program like CTF→interactive tuning, the windows will appear in random locations on the screen. If you position them nicely, the next time you run the same program in this project, it will remember your preferred window locations.

11.CTF→generate structure factor
a. Everything should be checked correctly. Just launch the job. Takes a few seconds.

12.CTF→automated fitting
a. Run again with same options above (this time it will use the structure factor). You only need to perform this process once. Iterating between 12 and 13 is not useful.

13.CTF→interactive tuning
a. Observe the improved fit.
b. You can adjust B-factors if you like, but they aren't really used for anything at present.
c. Remember to press 'save' or 'refit' on each image if you adjust anything.
d. You may also want to put the plot in SSNR mode to observe data quality. Data is only useful when above ~0.02 (2% contrast, 50x more noise than signal !)

14. *CTF→generate output*
    a. Select *refinebysnr* and *phasefliphp*
    b. Other options default values
    c. Launch, will take a minute or two to run

15. *Particle sets→ build particle sets*
    a. Press browse and select all of the particle stacks
    b. *excludebad* = checked (now, only a few particles have automatically been marked bad. You can improve results by going back later and eliminating more bad particles)
    c. *setname* = all
    d. *Launch*
    e. If you have a slower computer, you may wish to repeat this step for "*setname* = small", but pick only the best 6 images. You can use the 'small' set instead of the 'all' set below. For making initial models, it doesn't really matter, and will run faster. If you use 'small' for the final refinement, you will still get a good structure, the resolution will just be a bit worse.

➡ For users familiar with EMAN2.0x and a long delay waiting for this process to complete, note that in EMAN2.1 set building is virtually instantaneous. It will be done almost as soon as you press the Launch button. 'sets' are now stored in text files with a '.lst' extension, similar to EMAN1.

16. *Reference free class averages → generate classes*
    a. *Input* = sets/all__ctf_flip_hp.lst
    b. *Ncls* = 24, iter=6, *naliref* = 3
    c. parallel =   thread:N   where N is the number of cores on your computer
    d. Other options: defaults ok
    e. *Launch*. This may take ~15 min to finish (less if you have a faster computer, or used a smaller set of particles)

17. Use the browser
    a. Enter the "r2d_01" folder
    b. Look at "allrefs_06" (double click)
    c. Contains some good and some bad class-averages. Quality would have been better with more iterations, but these should be good enough for our purposes.
    d. You can double-click on a class-average to see the particles that went into it, but this isn't necessary. Colored marks indicate particles excluded from the average.
    e. Middle-click on the image viewer to get a control panel
    f. Select the 'del' button
    g. Delete all but the best 10-15 classes. Keep representatives of all views.
    h. Use *save* in control panel: "good_classes.hdf"
    i. Close browser

18. *Initial model→ make model*
    a. *Input* = good_classes.hdf
    b. *Sym* = d7, tries = 8
    c. parallel: set as above
    d. Launch
    e. When initial model generator is done, open browser and look in "initial_models"
    f. "model_01_01" will hopefully be good. If not, check 01_02. If still not, run again.
    g. The details for this section below describe how to check whether an initial model is good

19. *Inverting handedness*
    a. You may not know GroEL well enough to know whether the handedness is correct or not. In general, handedness can only be determined by a tilt experiment, so you have a 50% chance of having the correct handedness in your starting model. This explains how to manually invert the handedness if you decide you wish/need to.
    b. Use the browser and hilight the map you wish to flip the handedness on.
    c. Do not double-click to open it. Instead press the *FilterTool* button.
    d. Two windows will open. In the one that says 'default', select 'xform' in the left empty popup menu, then select 'flip' in the right popup.
    e. A box saying 'axis' will appear (you may need to make the window larger). Enter 'z'
    f. Check the box to the left of 'xform'. You should see the handedness of your map invert.
    g. On the *File* menu, you can then 'save processed map'.

20. *Coarse refinement, 3D refinement → run e2refine_easy*
    a. This new program has FAR fewer required parameters than the old e2refine.py
    b. input → select all__ctf_flip_hp.lst
    c. model → select your good starting model (model_00_01.hdf usually, or flipped map)
    d. targetres → 15, sym → d7, iter → 3, mass → 800
    e. fill in parallel as above.
    f. Launch
    g. e2refine_easy creates a logfile as it runs explaining the various decisions it makes and the parameters it uses, as well as summarizing some results. In the 2.1alpha1 release, this file is still fairly simple, but it will improve as the final 2.1 release approaches. Browse into the refine_01/report folder and select index.html. You can open this in a web browser or press the 'Info' button to display this file. The file is updated continuously while the refinement is running.
    h. Once at least one iteration is complete, you can select the *Resolution* item from the projectmanager (not *e2plotFSC*, but *Resolution* itself). The resolutions presented in this table (unlike EMAN2.0x) can be directly interpreted as true gold-standard values. Double click to get a plot.

➡ 3D refinement is the most computationally intensive step of the process. This initial coarse refinement will take ~30 minutes on a 4-core machine if the full data set is used. You can use the time for other tasks.

21. Final refinement, *3D refinement → run e2refine_easy*
    a. This will be similar to the last run, but we will continue the refinement we started rather than starting from scratch.
    b. input and model should both be empty text boxes now
    c. type "refine_01" in the *startfrom* box.
    d. change targetres to 8.0
    e. Launch. This will take longer to run, but will hopefully get through 1 or 2 iterations before the end of the tutorial session. You should start seeing helices in the first iteration.

➡ Note that setting *startfrom* to an existing refine_xx folder is NOT equivalent to setting *model* to the final map from the same refine_xx folder. If you specify a single input *model*, it will be phase-randomized two times to relatively low resolution as starting models for the even/odd half refinements (that is, you take a step backwards if you do this). If you use startfrom, the existing even/odd references are used, and the refinement can progress naturally, with no model bias.

##### Additional mini-tutorials available below:

- Boxing particles
- Shrinking particles using e2filtertool
- Manually marking bad particles

Detailed tutorial for each step above

Getting started

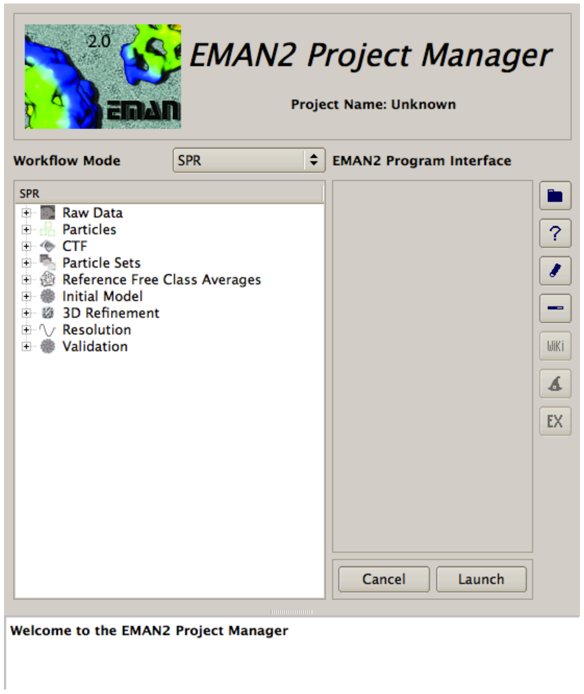
- Due to the limited time available for the tutorial at this workshop, we are all focusing on a single data set: a subset of the data used in our 2008 structure of GroEL. For those reading this tutorial outside the context of the workshop, there are additional data sets available from the 2011 EMAN2 tutorial, and, additionally, raw data for a range or projects is available from: http://ncmi.bcm.edu/publicdata/db/home. This tutorial and data has the advantage that the entire process can be completed by a knowledgable person in just a few hours, and produce a very satisfying structure. If, for example, you were to use Ribosomes, you would have to wait longer for the computation, and the resolution wouldn't be as good (not a large enough set). So, I encourage people to do GroEL once, first, before branching out to other specimens.

➡ Windows users - We strongly suggest using the windows command-line to start any programs rather than trying to click on icons. You will be able to more easily monitor error messages, and things are less likely to go wrong. Installation of an 'enhanced command-line' tool for windows, such as Console 2.x, will make your life somewhat easier.

If you are using a workshop computer, you should find the demo data already unzipped and available on your hard drive in a folder called workshop_beijing. At the command-line **cd <path to workshop_beijing>**. If you type **ls** (**dir** on windows) you should see: orig_boxes, and orig_micrographs. If not, try again. If you are on your own computer, you will need to download the data and **tar xvzf workshop_beijing.tgz** (or use an untarring tool on Windows).

The folder you are now in is called the 'project folder'. You will run virtually all commands from this location. All files associated with this project will then live in this folder or subfolders inside this folder. The system is designed so all of your data and results are self contained, if you use the GUI in the normal way. The goal is that at the end of the project you will be able to trace back any step you performed to produce any of your results without ambiguity. While you could use the lower-level command-line programs and produce reconstructions without this restriction, a year later, when you're publishing, and have to submit your model to the EMDB, you may appreciate the organization provided by the project system.

Type **e2projectmanager.py**. A window should appear that looks roughly like this:



Welcome to the EMAN2 Project Manager

➡ EMAN2 uses a lot of disk space. Disk storage is so inexpensive now, we err on the side of keeping intermediate files, on the theory that they may be useful later. You can always delete things you don't need, but it's very hard to get something back that you didn't keep in the first place.

Each item below is related to the corresponding number in the outline above:

4. Setup project

The project manager interface is an expandable tree. Each level of the tree has a form with associated information or parameters, even the levels which just appear to be containers for the levels below them.

- Begin by setting the overall properties of the project, by using the *Project → Edit Project* menu item.
- Enter the following parameters: mass = 800, Cs = 1.6, voltage = 300, A/pix=2.1

➡ A note on image file formats: There are many file formats used in the CryoEM community, including familiar formats like TIFF and PNG, cryo-EM specific formats like IMAGIC and SPIDER, and proprietary formats like DM4 and SER. EMAN2 supports virtually all file formats used in the community (http://blake.bcm.edu/emanwiki/Eman2DataStorage). People often ask "how do I convert file X to work with EMAN2". In most cases, the answer is "you don't have to !" Any EMAN2 program should transparently read any supported file format. A few specific programs may have some specific restrictions, but by and large this is the case. Similarly, when specifying output files, simply use the correct extension and EMAN2 will write to almost any format as well. There are a few restrictions on this, for example, the much abused trick of storing sets of 2-D images as a 3-D image in MRC files requires using the **e2proc2d.py** program with the --twod2threed or --threed2twod options. In general, **e2proc2d.py** and **e2proc3d.py** can be used for general file-format conversion as well as generic image processing operations (--process option).

5-6. Evaluate images and import data

You have two choices. You can simply trust all of the data is good, or you can go through the micrographs one at a time and briefly evaluate each (with the side-effect of estimating the defocus of each frame). There are two overall strategies:

1) Import all frames → Pick particles in all frames → Evaluate frames based on particles
2) Import only good frames → Pick particles in good frames → Particle based CTF

The advantage to method 2 is, if you have a significant fraction of 'bad' frames (and most people do), then you don't have to box particles from them because you've already eliminated them. This can save a lot of effort. However, even if you have evaluated the CCD frames, you'll still most likely want to look at the per-particle CTF fitting as well, meaning you will be looking at the power spectrum of each image twice (in two different ways).

If you have a really huge number of frames (say >5000), you may just give up on the manual approach entirely and decide you have to trust automatic picking and automatic CTF determination. The fully automated approach won't really be covered in this particular tutorial, because it is really only appropriate for the sort of large scale project that takes a lot of time (both computer and human) to complete.

5. Import only

If you simply wish to import the data, use the *Raw Data → Import Micrographs* item. Press the *Browse* button and highlight all of the image files in orig_micrographs. Since these images are scanned micrographs rather than CCD frames (they were recorded ~8 years ago), you will want to uncheck *X-ray pixels* and *Invert*.

The X-ray pixels filter is designed to remove the very high intensity pixels from CCD images. Scanned images rapidly become saturated when exposed to such, and don't really have this problem.

The Invert button changes the contrast in the image by multiplying by -1. EMAN2 uses the convention that your particles should be 'white' on a dark background. That is, the particle density to be reconstructed should have higher values than the surrounding solvent. Cryo-images on CCD or DDD detectors will have dark protein on a light background. The use of negative stain reverses this. In any case, you need to manually look at one of your images, and if the particles look dark, check the 'invert' box. If you forget to do this here, there will be another opportunity later in the process where inversion is possible.

When you press "Launch" it will process each micrograph as specified, and convert to HDF format in the "micrographs" folder.

***6.* Evaluate and Import**

Unless you have a simply huge number of frames and are confident that all of the images are good, I strongly recommend going through your images by hand to identify problems and assess quality. Bad frames == low quality reconstruction. More frames is not a solution to most problems.

- Raw Data → *Evaluate & Import Micrographs*
- press the *Browse* button
  - This should cause a browser window to open
  - Browse to the *orig_micrographs* folder and select all images. Press 'OK'
  - Change the *Box Size* to 384.
  - If you change to the *Command* tab, you can see (and edit) the exact EMAN2 command that will be executed when you press *Launch*.
- Press *Launch*.
  - Four windows will appear: Control Panel, Micrograph View, Plot and 2D FFT. You will need to arrange these windows so you can see them all. Make the micrograph window as large as you can, without obscuring the others. Use the mouse-wheel to zoom the micrograph window so you can see the entire micrograph and its pattern of green boxes. Select the first image file in the *Control Panel*.
  - In the *Control Panel* window, change *Ctf Zeroes* to *None*. This will remove the pattern of green rings obscuring the 2-D FFT.
  - Unselect the *Invert* and *X-ray Pixels* check boxes in the *Control Panel*. Particles should be white on a darker background. If the particles were darker, you would leave *Invert* selected. This data is scanned film, if the data was collected on CCD, you would leave *X-ray Pixels* checked.
  - In the micrograph window, clicking on any green box will toggle it on/off. Note that it may not be obvious that a single 'off' box in the center of 8 other 'on' boxes is actually off, since the lines overlap. The 'on' boxes define the region used for the power spectrum calculation. If there is contamination or some other artifact present in the image, turn 'off' the boxes in these regions.

- You can then go through the images one at a time and decide which ones are appropriate to further process. Most of the provided images are good. Things to consider:
  - Drift: If the image has too much directional falloff in the 2-D power spectrum, you should consider excluding it.
  - Astigmatism: None of the included images should have significant astigmatism, but when processing your own data, you should exclude images with too much astigmatism.
  - Particle concentration: If the particle concentration is so high you don't believe you will be able to isolate a lot of particles well separated from others, you may consider excluding it.
- For each image you decide is good, press the Import button (harmless to do it more than once). This will copy the image to the micrographs folder, and store the preliminary CTF parameters you've determined. Make sure you have unchecked the *Invert* and *X-ray Pixels* check boxes.
- If you accidentally import an image you didn't mean to, you can simply delete the hdf file from the micrographs folder. In EMAN2.0 this would have caused major problems. In EMAN2.1, you can manually move files around at pretty much any time.
- **e2evalimages.py <image> <image> ...** (the program you're using now) has many more capabilities you can explore, but for purposes of this tutorial, this is all you need.
- When you have imported all of the good frames, close the *Control Panel* window. This should cause the other windows to close as well.

7. Importing box coordinates

- As mentioned above, we don't have time in this tutorial for you to go through all of the images and select particles. This is usually the most (human) time consuming step of the entire single particle reconstruction process. Instead, we will use .box files which contain preselected particle locations. These box files are (intentionally) not perfect. They include some bad particles, so you can experiment with the mechanisms which can be used to clean them up.
- In the Project Manager: *Particles → Coordinate Import*
  - Use the *Browse* button, and select all of the .box files in the orig_box folder. It is ok to select .box files for micrographs you previously excluded. This will not cause those micrographs to be included in the project unless you import them later.
  - *Extension* = hdf
  - Press *Launch*. This import process takes only a second or so to complete, and there is no progress display indicating that anything happened.
  - In the next step of the process we will double check to see that this process successfully imported all of the particle locations.

8. Extracting selected particles

- We are now ready to actually extract the images of the particles from the micrographs and save them into particle stack files.

➡ A stack file is the common name for a single image file containing many images of the same size, and derives from the fact that in 'MRC format', sets of 2-D images are stored as a single large 3-D image. In other formats it is possible to have sets of 3-D images in a single file, and each 2-D or 3-D image in the file can have its own header information (not possible in MRC files).

- *Particles → Generate Output - e2boxer*
- Press the Browse button and select all of the micrographs. Note that the Stored Boxes column should now display the number of selected particles in each image (typically 100-300). If it doesn't then something went wrong with step 8, and you should try again or seek assistance. Note that a .box file was not provided for one of the images, so you could practice boxing by hand if you like.
- You should have *write_ptcls* as the only checkbox selected, box_size = 140, *norm* = *normalize.edgemean*, and *format* = *hdf*. Then press *Launch*.

➡ It is critical for accurate CTF correction that the box size be substantially larger than the particle, ideally almost 2x. That is, if a box that just barely contains your particle has a size of 128, you should use a final box size in the 192-256 range. This is larger than typically used in EMAN1, but there is a good reason for it, as you will see later. See: http://blake.bcm.edu/emanwiki/EMAN2/BoxSize for information on 'good' sizes to use to optimize processing speed. Since we are just practicing here, it doesn't matter much, but if this were your own data, picking an appropriate box size at this stage is critical !

➡ On Linux machines, you can open the Task Manager( ), and it will let you monitor running jobs to see when they have completed. In 2.1alpha1, it isn't fully functional on the other platforms, but you can try it...

- This process should take only a minute. You can use the browser to look in the 'particles' directory while it's running or after it's complete to monitor progress if the Task manager doesn't work.

➡ When working on your own data, if you have already selected your particles using another program, or prefer to use a different program, it is much preferable to import particle coordinates rather than simply import particles using *Particles → Particle Import*. This will preserve the particle location in the header of each particle, which could potentially be useful down the road.

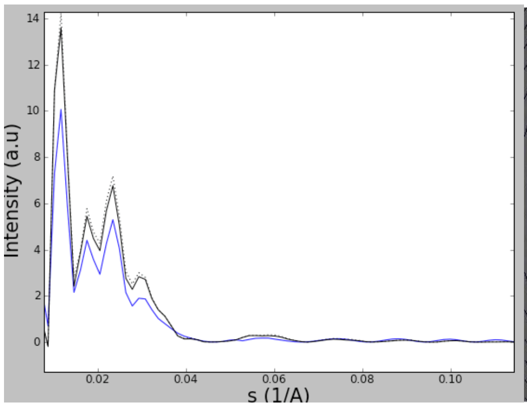
9-14. CTF Correction

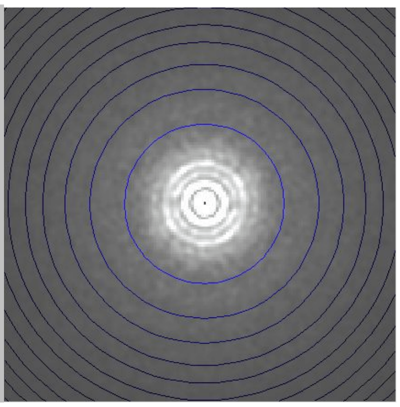
- CTF correction is broken into a sequence of steps. While the fitting is completely automated for most projects, it is a good idea to review the results manually. This is an important step in quality control of your data, as it gives you another opportunity to eliminate images with low quality particles which could actually damage your reconstruction. If the project simply has too many frames to manually review them all, you may consider just reviewing some representative images in each range of defocuses, but do not be tempted to completely skip the manual review process.

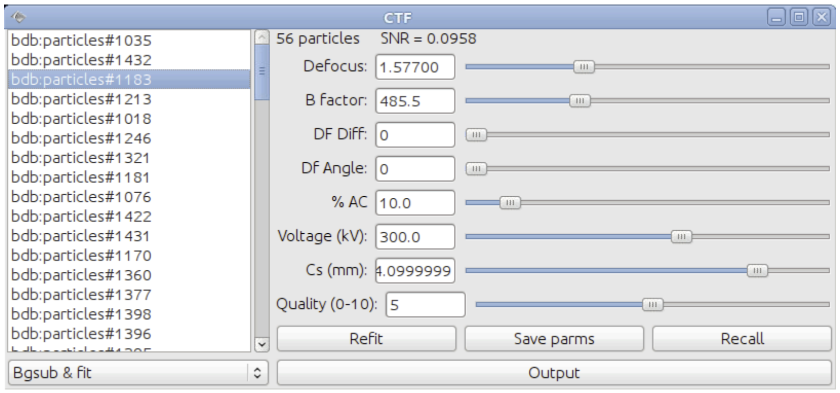
- Here is a basic overview. Detailed instructions follow:
  - Run automatic fitting (no structure factor yet) with 2x oversampling
  - Run manual fitting on at least a few images. Do a quick check to make sure determined defocuses and B-factors are ok.
  - Run structure factor determination on the images you just hand-checked.
  - Run automatic fitting (may improve results using structure factor)
  - Run manual fitting. Evaluate your data critically. Adjust quality of any images you suspect may not be as good.
  - Generate output with refinebysnr. (Oversampling should always be 1 here). This:
    - Fine tunes defocus based on high resolution SSNR optimization (optional)
    - Performs phase-flipping correction, and optional filters
    - Stores CTF parameters including SSNR in particle headers.

- ➡ SSNR = Spectral Signal to Noise Ratio. This is a measure of data quality, and its estimation from the particles themselves is one of EMAN2's most unique features. A Signal to Noise Ratio of 1.0 means there is an equal amount of signal and noise. SSNR is a measurement of noise level as a function of resolution. Naturally, SSNR is higher at lower resolutions, and lower at higher resolutions. SSNR is additive if proper weighting is applied when averaging particles together. Meaning, if the SSNR were 0.05 at 10Å resolution, it should take roughly 20 particles in that orientation to achieve a minimal average SSNR of 1. In theory you could use this method to achieve arbitrary resolutions, but in practice SSNR values below ~0.02 have empirically proven difficult to recover, regardless of the number of particles.
- ➡ Automatic B-factor determination isn't working very well in 2.1alpha1, but it doesn't matter much. Unlike EMAN1, the B-factors are really only used in structure factor determination, and generally only have a minor impact even there. It is a good idea to adjust them to reasonable values in the images you use for structure-factor determination, but don't waste a lot of time on it.
- ➡ When working with your own data, you may be tempted to take "CTF corrected particles" from some other software and use them in EMAN2. NOT a good idea. When particles say they have "CTF corrected particles", it generally means they have been phase-flipped only, or phase flipped and filtered with some sort of per-particle Wiener filter. This won't give EMAN2 the information it needs for accurate amplitude correction, classification or alignment. It will still do as good a job as it can, but this sort of data will not allow EMAN2 to use its best algorithms. Even if it's unsatisfying, we strongly recommend importing uncorrected particles and performing CTF correction within EMAN2.

- Select *CTF → Automated Fitting*
  - check the 'allparticles' box. This is an alternative to manually selecting all of the individual images in your particles folder.
  - 'minptcl' allows you to skip any frames with fewer than the specified number of particles. Leave this at 0 for the sample GroEL data.
  - 'minqual' allows you to skip any frames where the user has set the quality below a specified value. Again, you can leave this at 0 for now.
  - Set oversample to 2. This will provide better fitting in most cases. If you are working with your own data and the box size is large (256+) you can consider leaving this at 1.
  - Select *autohp*. This will modify the SNR curve to eliminate the first sharp peak that appears in virtually all single particle data due to ice gradients and other issues. This peak isn't completely removed, but is just heavily downfiltered, and in most cases it will noticeably improve alignments. However, in some projects, the structure factor is such that this option will filter out too much low resolution information. If you see strange low-resolution artifacts, you may try disabling this.
  - Select *curdefocushint*. This will make use of any existing defocus information during fitting. Only uncheck this if you have bad defocus values from earlier fitting attempts and want to start from scratch.
  - *Launch*. (watch the console for the output to stop,1-5 seconds/image)
- When it's done , *CTF→interactive tuning*
  - Options should all be right. Just *Launch*.
  - You will get 4 windows. 3 are shown here. The 4th shows the first 20 particles from the image stack, just as a reference.







CTF

bdb:particles#1035
bdb:particles#1432
bdb:particles#1183
bdb:particles#1213
bdb:particles#1018
bdb:particles#1246
bdb:particles#1321
bdb:particles#1181
bdb:particles#1076
bdb:particles#1422
bdb:particles#1431
bdb:particles#1170
bdb:particles#1360
bdb:particles#1377
bdb:particles#1398
bdb:particles#1396

56 particles SNR = 0.0958

Defocus:	1.57700
B Factor:	485.5
DF Diff:	0
DF Angle:	0
% AC	10.0
Voltage (kV):	300.0
Cs (mm):	4.0999999
Quality (0-10):	5

- This interface will look very similar to the earlier image evaluation interface, but there are some fundamental (and important) differences. The primary difference is that this analysis is based on the particles rather than regions of the micrograph. This permits us to do particle-based SSNR analysis.
- While there are many interesting things we can do with this program, at this point we mainly need to insure that the defocus values are correct. The automatic fitting is quite good, but with some specimens, or with particularly low quality data, it will make occasional errors. Generally if it's wrong, it is significantly wrong. If you find an image with a significantly incorrect defocus, adjust it so it's approximately correct, and hit the *refit* button (you can use it several times if necessary).
- If this doesn't solve the problem, you can fit manually, then press the *save parms* button instead.
- You may also wish to look at the B-factors and make sure that the general falloff of the fit curve at high resolution roughly matches the data. If you adjust the B-factor, remember to press *save parms*.
- This project is small enough that you can take the time to look at all of the images, and use the full set for structure factor determination. If you have 1000 images, and the automatic fitting seems to work fairly well, you may just select a subset of 10-20 images to fine tune and use for str

➡ Amplitude contrast can only be determined experimentally through some rather tricky experiments, and frankly, slightly different values will not have a strong impact in most reconstructions. Values of 10-20% generally work best for cryo data. However, for negative stain data, if you try CTF correction, you will likely want to use a much larger value than the default 10% (likely ~60-80% will work better).

    - You'll note that on your computer, the blue (theoretical) curve doesn't match the black (data) curve very well at low resolution. That's because we don't (yet) have a 1-D structure factor for our data. This shouldn't prevent you from doing a good job fitting the defocus and B-factor, so just ignore the poor low resolution fit for now. It will be improved in the next steps.

- CTF→generate structure factor
    - Options should all be right. Just *Launch*. (takes a few seconds)

- Rerun *CTF → Automated Fitting*
    - Default options should be correct.

- Now *CTF→interactive tuning, again*
    - This time, you should find that the blue fit curve actually matches the black curve fairly well. You may find in some cases that the overall fit matches well, but there is an apparent shift or scaling difference at low resolution. This isn't a problem. If you manually adjust parameters at this stage, you should focus on a good fit at high resolution, and not be overly concerned about a poor low-resolution match.
    - To assess data quality, rather than looking at the CTF fit, looking at SSNR curves is generally much more useful. In the pop-up menu under the list of images, you'll find *SNR* and *Smoothed SNR*. For good data, the SSNR between 0.01 and 0.03 should peak well above 0.5. Images with peak low resolution SSNR's much below this level are unlikely to align successfully, and will represent a much higher risk of contributing to noise bias in your final structure.
    - You may also look at high resolution and consider whether there is strong enough signal in the image to help your reconstruction at the desired resolution. Once the resolution falls much below 0.02 it usually won't contribute much to the reconstruction, even if you have a lot of particles.
    - If you decide one image is worse than the others, you may use the quality slider, and set it to a lower than the default (5) value. Similarly if you see any unusually good images, you may consider increasing this slider. It is not necessary to press *save* after adjusting the quality slider.
    - The actual values you use for quality are arbitrary. They have no direct meaning to EMAN2, but will be displayed to you later in the process when you are deciding which images to include in the reconstruction.
    - When you are happy with your CTF fitting, close the manual fitting program, and run CTF→generate output

➡ There is no point in repeating the process, and determining a new structure factor again. It is very unlikely to make a measurable improvement.

    - The refinebysnr checkbox should normally be checked. While in some cases (low resolution data) it won't do any good, it will rarely do anything harmful either. This option will make a final pass at making very small adjustments to the defocus value to optimize the measured SSNR of your data at high resolution.
    - The other options are used to select what types of output files to generate.
        - phaseflip - this will produce unfiltered phase-flipped particles
        - phasefliphp - This will produce phase flipped particles which have been high-pass filtered to eliminate any very low resolution peak present in the data (ice gradients,

etc.). Normally this is a good thing. Sometimes it is not, and can end up filtering out important low-resolution signal. You won't really know until you try.
    - wiener - This will produce Wiener filtered particles. Note that these particles are not normally useful for reconstructions (other than at very low resolution), as Wiener filters MUST be applied only once, and when particles are being averaged. Wiener filtered particles used in a reconstruction will be massively over-filtered. However, they can be useful in visualizing individual particles, and identifying bad particles.
  - Check all 3 of the output options, then *Launch*.

15. Building sets

- *Particle sets→ build particle sets*
  - Press *Browse* and select your best 5-6 images.
    - The interface for manually marking bad particles is not working well yet in EMAN2.1alpha1. It should be improved again in the next alpha release. Some bad particles will have been automatically selected during CTF output generation.

➡ In general, one 'bad' particle can do far more harm than a single 'good' particle would do to aid the reconstruction. While not absolutely necessary for the structures we are doing at the workshop, in general, you should get rid of bad particles here, when picking or later, semi-automatically.

  - setname = small
  - excludebad = checked (otherwise it will ignore any bad particle selections)
  - *Launch*
- Select *Particle sets→ build particle sets* again
  - This time select all of the images you thought were good before pressing OK
  - setname = all
  - *Launch*

➡ Particle sets allow you to try doing reconstructions with various subsets of your data without taking large amounts of disk space, and keeping track of exactly what you're doing. For example you may ask 'do I get a better reconstruction if I use only the 10 best images, or if I use all 20 images'. Particle sets do not make a copy of the particle data, but rather make a text file in LST format which points to the image data in the original image file. So, feel free to make as many stacks as you like. EMAN2 will treat LST files as if they are actual image files, so you can use them with any program, even though the actual data is in other files.

➡ In EMAN2.0, set building took some time to complete. The LST files in EMAN2.1 are created almost instantaneously, so do not expect to wait for anything after pressing Launch.

16. Generating reference free class averages (2D refinement)

- Select *Reference free class averages → generate classes*
  - *Input* = sets/small.lst
  - *Ncls* = 24, iter=6, *naliref* = 3
  - parallel= "thread:N" where N is the number of cores on your computer
  - Other options defaults ok, *Launch*

➡ For most projects, this is an ideal point at which to look for structural heterogeneity in your data (not a signficant issue for the workshop sample data except for 'bad' particles). If you see several class averages apparently in near-identical orientations, but with subtly different internal features, this may be a sign that your particle is moving in solution. For example, a molecule like mammalian fatty acid synthase can be observed to move as much as 50 A in solution. It can be difficult in some cases to distinguish between variability and differences in orientation, so you may not get a definitive answer to this question at this stage. If you do observe what appears to be structural variability, keep it in mind as you move on to subsequent steps.

17. Select good averages for making an initial model

- Our next goal is to select a subset of the class-averages we just generated (you'll have to wait until the class-averaging job finishes, perhaps 10-30 min depending on your computer).
- Use the browser
    - Look in the r2d_01 folder (results of the first run of reference-free class-averaging)
    - Double-click allrefs_06
        - You will see some good and some bad class-averages. It should be pretty obvious which are which.
        - Middle-click on the image viewer to get a control panel
        - Select the 'del' button
        - Delete all but the best 10-15 classes. Keep representatives of all views.
        - Press *save* in control panel and enter: "good_classes.hdf"
- Close browser

18. Making an initial model

There is a lot of controversy in the cryoEM community on this point. Some people feel that initial model generation is the most critical step in refinement, and you need to use difficult and time-consuming experimental methods to get an initial model before you can proceed. In the vast majority of cases, we have shown that the simple approach used in EMAN2 can give a completely reliable initial model with no additional experiments required. However, there are a few important caveats here:

➡ When you are uncertain about the quaternary structure of your molecule, tilt validation is a critical test. You collect pairs of images at 0 degrees and typically 10-20 degrees tilt, box out tilt pairs of particles, then run them through a tilt validation procedure against your final 3-D map. This method is fully implemented in EMAN2, but we will not cover it during this workshop (there is a separate tutorial for this in the Wiki).

➡ Heterogeneity is a potential issue. If you have a particle that is highly heterogeneous, the EMAN initial model strategy is likely to fail to produce a unique answer (since there isn't one). Single particle tomography may offer the best solution towards understanding the heterogeneity in your specimen. Once you understand the heterogeneity, EMAN2 includes e2refinemulti.py and e2classifyligand.py for purposes of processing such data sets using normal single particle data. (note that these two programs are not ported to EMAN2.1 yet, but should be in the next alpha release)

➡ Poor angular distribution. If your particles have a single, strongly preferred orientation, especially if this is combined with a low symmetry, there may not be enough information to produce an unambiguous starting model. However, it is also important to note that in this situation, even if you get a good starting model, refinement will also tend to degrade rather than improving the model. To perform a proper 3-D reconstruction, you must have a reasonable number of particles in orientations covering at least one *great circle* around the unit sphere.

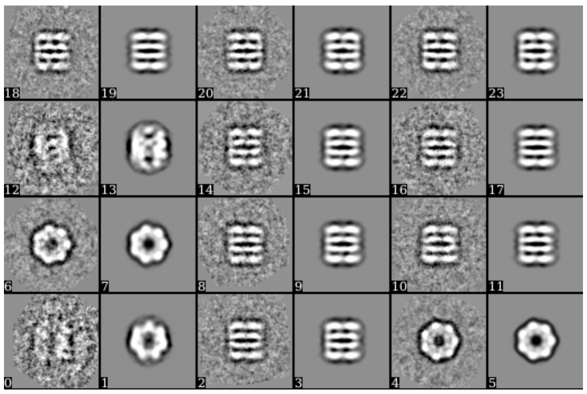
➡ If you do have a difficult structure, single particle tomography is one good solution. Random Conical Tilt is another possibility. Both methods are fully supported in EMAN2.

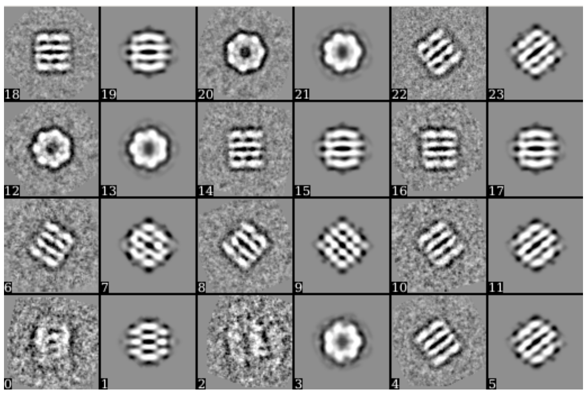
- Despite all of the lengthly caveats and descriptions, the actual initial model generation program is quite easy to use. Just select: *Initial model→make model - e2initialmodel*
- *Input* = good_classes.hdf, *sym* = d7, *iterations* = 8, *models* = 8
- *parallel*= “thread:N” (fill this in wherever you see a *parallel* box)
- Use the default values for the other options, and *Launch*

➡ This program will take a few minutes to run. What it does is fairly straightforward. It treats the class-averages you provided as input as particles, generates a randomized blob as a starting model, and runs a highly optimized version of the normal EMAN2 refinement procedure to refine a structure. It does this *models* times, and in the end sorts the various output maps in order of apparent quality.

➡ For most structures, there are a number of ‘local minima’ in the energy space. What that means is, there are a number of incorrect structures which can still appear to agree fairly well (but not as well as the correct structure) with the input data. So, some fraction of the answers you get out are likely to be bad starting models. On the bright side, such bad starting models are usually quite obvious. The severity of this problem varies considerably with the shape of the molecule and the amount of orientation coverage you have. GroEL, with it’s strongly preferred orientation and nearly square shape in the side view, is actually among the most difficult structures to produce a good starting model for, particularly if you shrink your data heavily. Interestingly, particles like ribosomes, generally viewed as ‘difficult’ have virtually no local minima, and will produce a usable starting model almost every time.

- For each output, 4 files will be produced in the initial_models folder. We will only look at two of them in the tutorial: ‘model_NN_MM’ and ‘model_NN_MM_aptcl’. In this case, we will start by looking at ‘model_01_01_aptcl’, which should theoretically represent the best of the 8 produced starting models. Use the browser to view this file. Here are two examples of possible aptcl files:





- Each of these files contains pairs of images. Image 0,2,4,... will be the class-averages you provided as input to the program. Image 1,3,5,... will be projections of the 3-D model in the orientation which should match the projection. Can you tell which of these two sets of images represents a correct structure ?

- If you compare carefully, you'll notice that aside from one pair, the set on the left has a very good match between the class-average inputs and the map projections. The set on the right, however, has a very poor match between most of the pairs. Look at one or more of the _aptcl files in your initial_models folder and see if any of them seem to look good. If so, double click on the corresponding model_01_01 to see if the structure looks reasonable.
- Hopefully the first model will be good. If not, check the other models. If none are good, you need to run the initial model generation process again (it is, after all, random). This time you will get a sequence of model_02_YY files.

19. Inverting handedness

The outline tutorial above discusses how to use e2filtertool.py to invert the handedness of a structure. Here will will describe the alternative command-line approach to do the same thing. You may or may not actually need to do this at this point. There is a 50% chance that the initial model you have will have the correct handedness.

e2proc3d.py (and e2proc2d.py) can be used to perform generic image processing operations as well as file format conversions. We can use sequences of --process options to apply any of EMAN2's 180 different categorized image filters.

- Let's start by looking at the available processors. Go to the terminal window and type **e2help.py processors**
- This will produce a concise (one line per filter) list of all of the available algorithms in this category. There are filters, masks, transformations, and a host of other operations available, each of which takes a set of parameters. To get more detail on the processors and the parameters, you can enter:
- **e2help.py processors -v 2**
- You can use these processors with e2proc2d.py or e2proc3d.py very easily:
- **e2proc3d.py infile.hdf outfile.mrc --process procname:parm1=value1:parm2=value2**
- procname is the name of any of the available processors, and the parm/value pairs are the associated parameters. You may also put multiple --process options on a single command-line. Also note that the command above will convert an HDF input file to an MRC format output file after performing the requested processing.
- Here is an example of how to flip the handedness of an initial model:

e2proc3d.py initial_models/model_01_01.hdf model_flipped.hdf --process=xform.flip:axis=z

- You may find the filtertool approach described in the outline easier for general processing, as it provides sequential choices rather than making you read through the e2help.py output, but e2proc2d/3d are also very useful commands.

20. 3D Refinement

- We are finally ready to do our first (coarse) 3-D refinement.

➡ Previously in EMAN2.0, there were a LOT of options to fill in and explain when running 3-D refinements. In EMAN2.1 there is a new command called e2refine_easy.py. In normal use, this command takes only a handful of parameters, and all of the other options are automatically selected by looking at the parameters of your project and your stated (to the program) goals. It will produce an HTML (for web browsers) document describing all of the decisions it made while running, and detailing all of the steps of refinement. All of the options in the original e2refine.py command are still available from the command-line for users wishing very detailed control over the process, but most users will never need to use them or even be aware that they exist.

- The new e2refine_easy.py program now computes a true 'gold standard' resolution as part of the refinement process, for each iteration of the refinement algorithm, and even uses this information to properly filter the final 3-D maps. Additionally, optimizations in parameter selection have reduced the time required to run a refinement even further.
- 3D refinement → run e2refine_easy
- input→ all__ctf_flip_hp.lst
- model → select your good starting model (model_00_01.hdf usually, or the flipped map)
- targetres→15,sym→d7,iter→3,mass→800
- fill in *parallel* as above
- Launch
- e2refine_easy creates a logfile as it runs explaining the various decisions it makes and the parameters it uses, as well as summarizing some results. In the 2.1alpha1 release, this file is still fairly basic, but it will improve as the final 2.1 release approaches. Browse into the refine_01/report folder and select index.html. You can open this in a web browser or press the 'Info' button to display this file. The file is updated continuously while the refinement is running.
- Once at least one iteration is complete, you can select the Resolution item from the projectmanager (not e2plotFSC, but Resolution itself). The resolutions presented in this table (unlike EMAN2.0x) can be directly interpreted as true gold-standard values. Double click to get a plot.
- 3D refinement is the most computationally intensive step of the process. This initial coarse refinement will take ~30 minutes on a 4-core machine if the full data set is used. You can use the time for other tasks.
- The refined map will be called refine_01/threed_03.hdf when the job finishes.

21. Final refinement, 3D refinement → run e2refine_easy

- This will be similar to the last run, but we will continue the refinement we started rather than starting from scratch.
- *input* and *model* should both be empty text boxes now
- type "refine_01" in the *startfrom* box.
- change *targetres* to 8.0
- Launch. This will take longer to run, but will hopefully get through 1 or 2 iterations before the end of the tutorial session. You should already start seeing helices in the first iteration (refine_02/threed_01.hdf).

➡ Note that setting startfrom to an existing refine_xx folder is NOT equivalent to setting model to the final map from the same refine_xx folder. If you specify a single input model, it will be phase-randomized two times to relatively low resolution as starting models for the even/odd half refinements (that is, you take a step backwards if you do this). If you use startfrom, the existing even/odd references are used, and the refinement can progress naturally, with no model bias.

22. You're done

Look at the final refined map (refine_02/threed_03.hdf). You can even open it in Chimera and compare it to PDB map 1SS8 (crystal structure of a GroEL heptamer). You should see that the helices are quite clear and faithfully reproduced.

In EMAN2.0 we would have also needed to run e2refine_evenodd.py or e2eotest.py to compute resolution curves. In EMAN2.1, the "gold standard" FSC curves are produced as part of the refinement, and are available as fsc_*.txt in each refine_XX directory. You can double-click on any of these plots in the browser to display them. These FSC curves should always be "healthy". That is, they will start near 1.0 at low resolution then fall off fairly sharply to zero, and then oscillate noisily around zero until Nyquist. These curves can be safely interpreted using the 0.143 threshold for assessing resolution. The values shown in the Resolution table in the project manager represent 0.143 values when the FSC curve has been filtered with a 5-point running average (to smooth it out).

Extra Mini-tutorials

Interactive particle selection (boxing)

We won't be interactively boxing all of the images due to time constraints, but we will play around with the available boxing tools to get a feel for them. Obviously in your own project, you will have to actually box out particles using EMAN2 or some other available tool. If you use some other tool to do this, you just import the boxed out particle stacks using *Particles → Particle Import*. If you want to use EMAN2's semi-automatic boxing, continue with this section.

- To avoid interfering with the existing refinement, create a new project folder, **cd** into it, then run **e2projectmanager.py**
- Follow step 6 above to import the micrographs into this new project, but do not proceed with step 8. We want to start from scratch.
- *Particles → Interactive Boxing*
- Select all micrographs with 'Browse'
- *Boxsize* = 140, *Launch*
- 4 windows should open. The windows are:
  - Control panel - it has an assortment of buttons and text entry boxes
  - Micrograph display window - shows the full micrograph
  - particle display window - (with nothing in it) will eventually show the selected particles.
  - micrograph thumbnails - This will only appear if you selected 2 or more micrographs in the previous step. Clicking on one of these will select the current image to be boxed.
- You'll need to reposition these windows so you can use them effectively. The micrograph display window is the most important, and should take most of the screen. The particle window should also be visible, though it can be much smaller. At least a piece of the control panel should be visible so you can easily bring it to the front when you need it. The thumbnails window can be safely hidden most of the time.
- The particle picker (e2boxer.py when started from the command line) has 4 modes of operation:
  - Manual - You manually select each particle
  - Erase - Mark large regions of the display which should be excluded from automatic picking
  - Swarm - A fast, interactive autopicker. Select a few reference particles, and it will select others. The more references you select, generally the better it will do.
  - Gauss - Programmed autoboxer. Set parameters and it will find particles automatically.

➡ Note that particles the different modes can be combined. If you use *Swarm* to select most of the particles in the image, but it has missed some, you can use *Manual* mode to pick up the missing particles without impacting the *Swarm* results.

- Start with the *Swarm* picking mode. Select this mode, and start by manually selecting 2-3 particles. You should see many other boxes appear automatically. The more manual references you pick, the better the automatic selection should become. Some particles may cause the picker to become too liberal. In this case, generally picking a few more references manually will help.
- The *Particle Diameter* parameter is quite important. This is not the same as box-size. If you are having troubles getting good Swarm results, try changing this parameter. You can also change the method to 'More Selective' to get fewer particles.

➡ You can delete 'bad particles' by holding down shift and clicking on them. This includes particles that were automatically selected. If you delete one of the references you selected, it will update the autopicking results.

➡ Bad particles or boxes that contain just noise can be damaging to your reconstruction, as they permit noise/model bias to become stronger. It is much better to miss a few good particles if it permits you to exclude obvious 'bad' particles.

➡ One caveat to the above, if the good particles that get excluded are all in one orientation, that would be bad

➡ When picking particles, it can sometimes help for purposes of more accurately centering and identifying particles, to use a box size smaller than the final size you plan to use for processing. This is absolutely fine. You can use whatever box size you like during the picking process. When you get to 'generate output' later, you will be given a chance to select the final box size to be used for extracting particles.

- When you finish picking the particles in one image, DO NOT press the *Write Output* button (don't worry, all of your particle locations are stored automatically as you work). Instead, just select the next image in the thumbnail display. If you have used *Swarm* mode at all, the references from the first image will be used on the subsequent images as well.
- When you finish your interactive picking session just close the *Control Panel* window or press the *Done* button.

Shrinking some particles for faster class-averaging

- Making reference free class-averages in EMAN2 is fairly fast (certainly MUCH faster than EMAN1 or other competing methods). However, it would be even faster if we don't use full-scale particle data, since these 2-D averages won't have very high resolution anyway.
- Open the file browser (top button on the right, looks like folder)
- Browse to the "all__ctf_filt_hp" file you created in the sets folder
- Select this set, and press the *filtertool* button
  - Where it says the word default, enter shrink
  - Below shrink select *math* in the left popup menu
  - Select *meanshrink* in the right popup
  - Enter 2 for the *n* parameter that just appeared
  - Check the checkbox next to math. The particles in the image window should become 1/2 size
  - On the *file* menu select *save processed stack*
  - Enter *all-s2.hdf* in the window that appears, and press ok
  - Close the filtertool window

➡ **e2filtertool.py** is a powerful program which allows you to assemble sequences of image processing operations in both 2-D and 3-D, and interactively adjust their parameters. In essence it is a graphical version of the command-line **e2proc2d.py** and **e2proc3d.py** generic image processing programs. In fact, when you build a set of processors in **e2filtertool.py**, it stores them in text files called "filtertool_<name>.txt". If you look at one of these files you will find the command-line parameters you would use with **e2proc2d.py** or **e2proc3d.py** to accomplish the same results.

Manually mark bad particles

- EMAN2.1 will automatically detect particles which are off the edge of the micrograph and mark them as bad (these particles can cause a lot of problems).

➡ We are still working on a new tool for integrated 'bad particle' marking in EMAN2. The description here is a temporary solution, which takes more effort than it should.

- Open the browser and go into the 'particles' folder. Select one of the non-phase-flipped particle stacks, and double-click it. You should see a tiled view appear.
- Open the control-panel (middle click)
- There is a *Sets* button and a *Sets* tab in this window. Select both of them.
- If there was already a bad particle in this image, you will see "bad_particles" in the sets list. If there wasn't the list will be empty.
- Scroll through the particles and click on a bad one. "bad_particles" will appear if it wasn't there previously. Highlight "bad_particles" in this list.
- Click on other bad particles in this file. Click a second time to unmark a marked particle.
- Continue the process with the other images in the "particles" folder. Any images marked as bad can be excluded when you build a set of particles (step 16 above). Note that this exclusion happens when the set is created. If you mark additional particles as bad after creating a set, they will only be excluded from future sets, not already created sets.