EMAN2 Reconstruction Tutorial
Using the new Project Manager interface

Note that this tutorial was updated on 07/06/2012. It should not be used with versions of EMAN2 prior to 2.06 (or snapshots very close to the 2.06 release).

- ➡ EMAN2 documentation is largely provided via the Wiki at: http://blake.bcm.edu If you wish to edit the Wiki, first, create an account for yourself, then send email to sludtke@bcm.edu and we will adjust permissions so you can edit. Previously this was an ‘open’ wiki, but we had serious spam problems, and now have to approve new users. You don’t need an account to view the contents, of course.
- ➡ There are also two Google Groups for discussing EMAN2/SPARX. These can be used basically as mailing lists with a permanent archive of past discussions. New feature announcements, bug reports, user questions, etc. are all made through the Google Groups. You don’t have to join the group to search the archive, only if you wish to receive emails. Links to the groups and more details are provided on the front page of the Wiki.

- ➡ **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 display windows (plots, 2-D images and 3-D volume display), 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. Note that it is also possible to enter values which are outside the current slider range.

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 new Project Manager interface, which replaces the older Workflow interface. EMAN2 is now at a bit of a crossroads, as we prepare for EMAN2.1 in the next couple of months. EMAN2.1 will bring a number of significant changes to the conventions for how images and information is handled. The Project Manager interface can be used for the full reconstruction process in EMAN2.06, but some user-friendly features, like the Wizard interface, are not yet complete. This tutorial will explain how to use the current version of the Project Manager to complete a full single particle reconstruction.

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 10,000 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. Since this workshop is making use of Windows PC's, the detailed instructions will focus on that platform, but there may still be some Windows-specific problems. Please call them to my attention if something unexpected happens.

Overview/Quickstart

For those who never read manuals, and hate to be told too many details, we begin with an outline of the overall process. This will be followed by the detailed tutorial. This is the 'quickstart' version of the entire process:

1. Open a Windows command prompt (use 'Console' or other enhanced command line if available)
2. **cd <project directory>**
3. **e2projectmanager.py**
4. *Project menu -> edit projects*
    a. *Mass* = 800, *Cs* = 1.6, *voltage* = 300, *apix* =2.1
5. *Raw Data -> Evaluate & Import Micrographs*
    a. Browse for images in orig_micrographs, select all HDF files
    b. *ac* = 10, *box* = 384
    c. *launch*
6. Evaluate images and import good ones
    a. Set Annotate to None, zoom in a bit on 2D power spectrum
    b. Zoom image display so most of image visible
    c. Deselect any squares with contamination or c-film in image display
    d. Adjust CTF parameters if necessary, though defocus is only important parameter
    e. If image is 'good', uncheck *Invert* and *X-ray Pixels*, then press *Import*
    f. That will move the image to micrographs directory and record CTF parameters
    g. repeat for each image
    h. Exit by closing 'e2evalimage - Control Panel' window when done
7. *Particles -> Interactive Boxing* - Only need to do this for one or two images to get the experience
    a. Select all micrographs with 'Browse'
    b. *Boxsize* = 168, *Launch*
    c. Arrange windows so micrograph fills most of display
    d. use Current Boxing Tool = "Erase" to mark bad regions of the image
    e. In manual mode left click/drag to select a particle, shift-left-click to erase
    f. In Swarm mode, select templates until most particles are accurately selected
    g. go to manual mode and clean up results
    h. Black boxes -> swarm references, Green boxes -> swarm auto, white -> manual
    i. Shift-click can be used in 'Particles' window as well
    j. When finished with an image, do NOT 'write output', just select the next image from the 'Thumbnails' window
8. *Particles -> Coordinate Import -e2import.py*, this will load particle box locations we provide
    a. Use Browse and select all .box files in orig_box
    b. *extension* -> hdf
    c. *Launch*
    d. Very fast, don't expect to wait for it.
9. *Particles -> generate output*
    a. Default settings
10. *CTF -> automated fitting*
    a. Make sure all particle files selected
    b. *Oversamp* = 2, other options should be ok
    c. *Launch*
11. *CTF->interactive tuning*

   a. Fits should be good, but check to make sure defocus is right. Can also tweak b factors if you like

12. *CTF->generate structure factor*
    a. Use all images
13. *CTF->automated fitting*
    a. Run again with same options
14. *CTF->interactive tuning*
    a. Just to observe the improved fit. Tweak if you like
15. *CTF->generate output*
    a. Select *refinebysnr* and *phasefliphp*
16. Particle sets-> build particle sets
    a. Press browse
    b. Double -click on the first image in the browser. Single clicking on each particle will mark it as 'bad'
    c. Repeat for each image you want to mark bad particles in.
    d. When done, select all images in the browser and press ok.
    e. *excludebad* = checked
    f. *setname* = all
    g. *Launch*
    h. repeat for *setname* = small, but pick only the best 6 images
17. Shrink some particles for class averaging
    a. Open file browser (top button on right)
    b. Browse to the 'small_ctf_filt_hp' file you created in the sets folder
    c. Select this set, and press the *filtertool* button
    d. Replace *default* with *shrink*
    e. Below *shrink* select *math* in the left popup menu
    f. Select *meanshrink* in the right popup
    g. Enter 2 for the *n* parameter that just appeared
    h. Check the checkbox next to math. The particles in the image window should shrink.
    i. On the *file* menu select *save processed stack*
    j. Enter 'all-s2.hdf' in the window that appears, and press ok
    k. Close the filtertool window
18. *Reference free class averages -> generate classes*
    a. *Input* = small-s2.hdf
    b. *Ncls* = 24, *normproj* = checked, *iter* = 6, *naliref* = 5, *nbasisfp* = 5
    c. Other options defaults ok
    d. Repeat for all-s2.hdf with *Ncls* = 48
19. Use the browser
    a. Enter the "r2d_01" directory
    b. Look at "allrefs_06" (double click)
    c. Contains some good and some bad class-averages
    d. Double-click on a class-average to see the particles that went into it. Red marks are 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. *save* in control panel: good_classes.hdf
    i. Close browser
20. *Initial model-> make model - e2initialmodel*
    a. *Input* = good_classes.hdf
    b. *Sym* =d7, iterations = 12
    c. Other options defaults

   d. When initial model generator is done, open browser and look in “initial_models”
   e. “model_01_01” will hopefully be good. If not, run again
21. While that runs, *particle sets -> evaluate particle sets*
   a. In new window, select “bdb:r2d_02#allrefs_06”
   b. 3 windows appear, 2 blank, 1 with averages
   c. Click on bad average to see included and excluded particles
   d. Double click on any bad class averages (not particles). Blue mark appears.
   e. Click 'mark as bad', then respond yes
   f. This marks the particles from the bad classes as bad. Not always a good idea...
   g. Build particle sets again, with a new name, and you will see that the new set is smaller. Use either the old or new set for 3d refinement. Your choice.
22. *3D refinement -> run e2refine*
   a. *See the options in the detailed tutorial below*

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 high resolution 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.

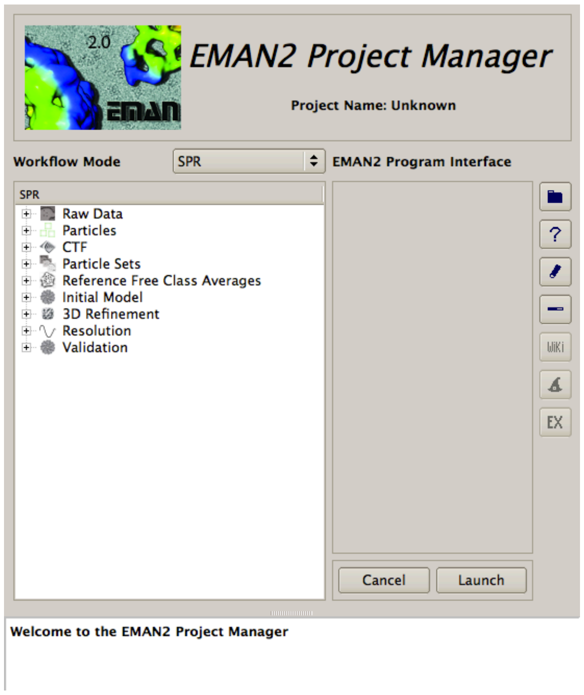
➡ Text you see in *italics* will generally be labels you will see in the GUI, such as buttons to press. Text you see in **bold**, are commands to be typed in. Items like: <param> are parameters you should fill in (without the < >). Items like: [param] are optional parameters (again, fill without the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

➡ 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.

You will find the demo data already unzipped and available on your hard drive in a directory called workshop_singapore. At the command-line **cd <path to workshop_singapore>**. If you type **dir** you should see: 0README.txt, orig_boxes, and orig_micrographs. If not, try again.

The directory/folder you are now in is called the ‘project directory’. You will run virtually all commands from this directory. All files associated with this project will then live in this directory or subdirectories of this directory. 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 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.

➡ PLEASE read this web page: http://blake.bcm.edu/emanwiki/EMAN2/DatabaseWarning

4. Setup project

The workflow 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

5-6. Evaluate images and import data

- Expand *Raw Data* (by pressing on the little + to the next of it).
- select *Evaluate & Import Micrographs*
- press the *Browse* button
  - This should cause a browser window to open
  - Browse to the *orig_micrographs* directory 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, but no easy way to undo once you press it). 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.
- **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.
- Close the *Control Panel* window. Should cause the other windows to close as well.

7. 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. If you feel short of time, you can skip this section entirely and jump directly to section 8 for now, and return to this section later while you're waiting for something else to complete.

➡ The displays on the workshop PCs are marginal for this task. To box particles easily, you really need to have a large display with at LEAST 1080p resolution. Dual displays or 30" displays are even better. Some laptops have adequate displays for this, but many do not.

- In the Project Manager, expand *Particles*.
- select *Interactive Boxing*
  - Use Browse to select a few (4-8) of the micrographs.
    - Note: If you are using a computer with a fair bit (4+ GB) of RAM, you can select all of the images at once. This doesn't really gain you very much, though.
  - Set the *box size* to 168.
  - Press *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
      - the 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.

➡ Also note that the Swarm picker works well on some projects, and less well on others. If you prefer to use some other tool, such as EMAN1's popular boxer tool, it is possible to import box coordinates into EMAN2, and have them appear in EMAN2's boxer as *Manual* mode boxes. At the moment, this mechanism only works for EMAN1 style .box files, but we'd be happy to add other formats upon request.

- 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

➡ Note that this isn't your only opportunity to eliminate bad particles. Try not to be overly liberal, but realize that you will have two more opportunities later to remove bad particles (after CTF processing).

➡ When picking, 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 processing.

- 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.

8. 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 see the mechanisms which can be used to clean them up.

- In the Project Manager: *Particles -> Coordinate Import -e2import.py*
  - 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 immediate 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.

9. 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 (one image file containing many individual particle images).
- *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.
- You should have the *write_ptcls* as the only checkbox selected, box_size = 168, *norm* = *normalize.edgemean*, and *format* = *bdb*. 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 non-Windows machines, you can open the Task Manager (4th button from the top in the Project Manager), and it will let you monitor running jobs to see when they have completed. Unfortunately this capability does not yet exist on windows, though we expect it to in the 2.1 release.

- This process should take only a few seconds. If you select the Particles item from the Project Manager after it's complete, it will show you a list of all of the micrograph names and the number of boxed out particles for each.

➡ Assuming you chose the 'bdb' format your boxed particle output you'll note that the filenames that appear aren't simply "1160_ptcls", but much longer and complicated looking 'bdb:particles#1160_ptcls'. The specifications for bdb database access are fairly straightforward:
  - 'bdb:dbname' which refers to the database (think of it as an image file) in the local directory named 'dbname'. 'dbname' can contain individual images, numbered sets of images, named images and other named metadata
  - 'bdb:/path/to#dbname' allows you to specify a database residing in a different directory. Note that '#' is used for the final separation between the path and the database name
  - files in the EMAN2DB directory should NEVER be renamed, individually deleted, moved, etc. It can cause the system to become confused and produce a range of seemingly unrelated errors.
  - If you really insist on deleting files in the BDB directory, make sure you aren't running any EMAN2 programs, run **e2bdb.py -c** and only then erase the files.
  - For more info on the database please see: http://blake.bcm.edu/emanwiki/Eman2DataStorage

➡ 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.

10-15. CTF Correction

- CTF correction is broken into a sequence of steps. While the fitting is completely automated for most projects, you need 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.
- Here is the basic overview. Detailed instructions follow:
  - Run automatic fitting (no structure factor available) with 2x oversampling
  - Run manual fitting. Do a quick check to make sure determined defocuses and B-factors are ok.
  - Run structure factor determination.
  - Run automatic fitting (may slightly 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 accurate 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, and is generally considered a reasonable lower limit for a solved structure. SSNR makes this measurement at every resolution. Naturally SSNR is higher at lower resolutions, and lower at higher resolutions. SSNR is additive if proper weighting is applied. 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.

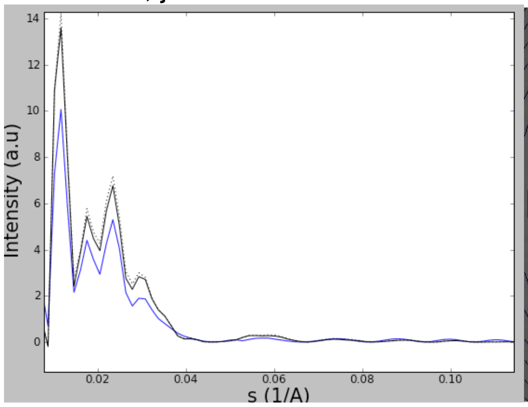
➡ Automatic B-factor determination doesn't work all that well, but it doesn't really matter. Unlike EMAN1, the B-factors really aren't used in EMAN2, with the sole exception of determining structure factor, and its influence there is fairly minor in most cases. Just insure that the B-factors aren't completely unreasonable and don't vary very widely among images.

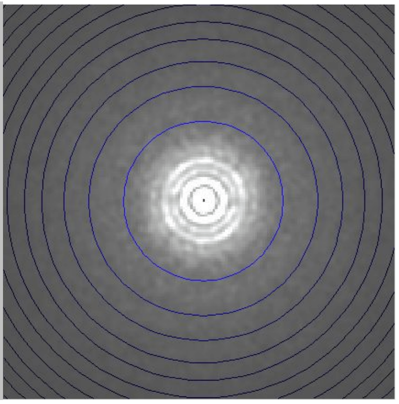
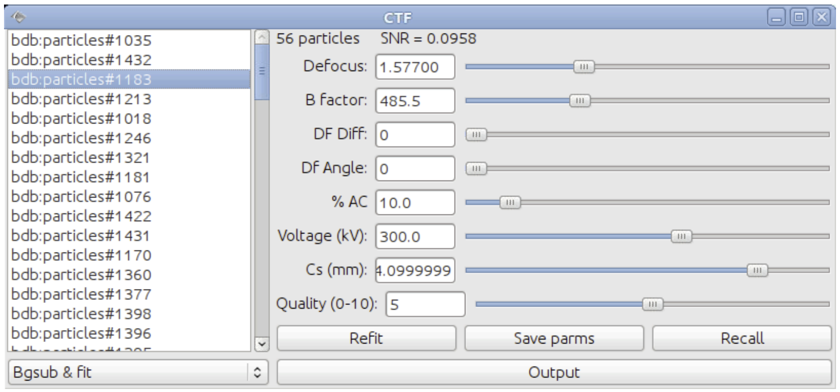
➡ When working with your own data, we do not recommend using CTF corrected (ie- phase-flipped) particles from EMAN1 or from any other software package. While it is possible to reconstruct phase-flipped particles from other packages, by disabling all of EMAN2's CTF correction and SNR weighting features, you will be throwing away much of what makes EMAN2 unique. No other packages will determine the SSNR and particle-based background parameters EMAN2 requires for many of its algorithms.

- Select *CTF -> Automated Fitting*
  - Use *Browse* to select all of your data
  - 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 may consider leaving this at 1.

- Check *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 a fraction of 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.
- Check *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.
- The other options should be correct as they are.
- *Launch*.

- When it's done (watch the console for the output to stop), *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.



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 the odd error. 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*.

➡ 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. 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 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*.

- 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 SNR at >0.01 should peak at at least 0.5. Images with peak low resolution SNR'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.
  - If you decide an 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 save after adjusting this specific slider.
  - The actual values you use for Quality are arbitrary. They have no 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, run CTF->generate output
    - If you are wondering, no, 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 - As it sounds, this will produce unfiltered phase-flipped particles with embedded CTF parameters
      - 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 very 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 are very useful in some cases for visualizing your raw data, and identifying bad particles.
  - Check all 3 of the output options, then *Launch*.

16. Marking bad particles and Building sets

- *Particle sets-> build particle sets*
  - Press *Browse*
    - The interface for manually marking bad particles for exclusion in further analysis is a bit counterintutitve. Sorry about that.
      - When you press browse, a file browser will appear. If you double-click one of the image files, it will open a display of all of the particles in that image stack.
      - To mark a specific particle as bad, double click on the particle in this image display window. While it will look just like any other tiled image display, it will be in a special mode configured for marking bad particles.
      - Note that in this case, manually marking bad particles probably isn't critical. The box set that came with the demo data has only ~10% bad particles, and there are other automated methods we can use to eliminate many of these later.

➡ 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 rough structures we are doing at the workshop, in general, you should get rid of bad particles here, when picking or later, semi-automatically.

    - When you've finished marking any bad particles you wish to mark, we're ready to build our particle sets. We want to make 2 sets. One with all of the micrographs (or all that you thought had good quality), and another with only ~1000 particles for faster initial model generation.
    - Before you press OK in the open file browser window, select your 4-5 best images.
  - setname = small
  - excludebad = checked (otherwise it will ignore your 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 (you don't need to manually select bad particles again. It will permanently remember the ones you already marked)
  - 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 'virtual stack file' which points to the image data in the original image file. So, feel free to make as many stacks as you like.

17. Shrink 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 will be even faster if we don't use full-scale particle data, since these 2-D averages won't have very high resolution anyway. So, we first 'shrink' our small particle stack to make the process faster.

- Open the file browser (top button on the right, looks like folder)
- Browse to the “small_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 *small-s2.hdf* in the window that appears, and press ok
  - Close the filtertool window
- Repeat the above process for “all_ctf_filt.hp”, but save the output as *all-s2.hdf*. When you do this, note you can just select shrink from the pop up menu rather than enter it in the text box.

➡ **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.

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

- Select *Reference free class averages -> generate classes*
  - *Input* = small-s2.hdf
  - *Ncls*=24, *normproj* = checked, *iter* = 6, *naliref* = 5, *nbasisfp* = 5
  - Other options defaults ok, *Launch*
- On Windows, that job will only use one processor. For a separate purpose, we’ll use another processor to compute a different set of class-averages at the same time. Repeat the above, but with:
  - Input = all-s2.hdf, Ncls = 48

➡ For most projects, this is an ideal point at which to look for structural heterogeneity in your data (not an issue for the workshop sample data). 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.

19. 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 first class-averaging job finishes).
- Use the browser
  - Enter 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.
    - *save* in control panel and enter: “good_classes.hdf”

- Close browser

20. 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:

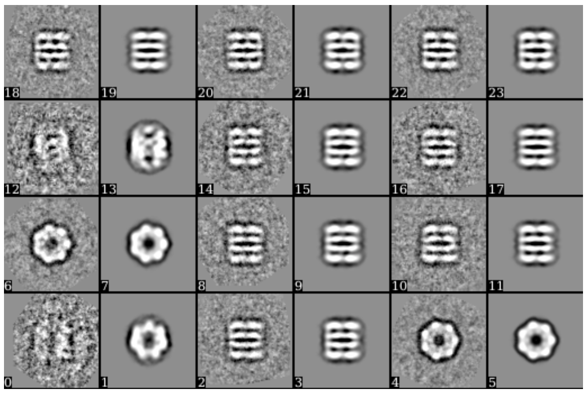
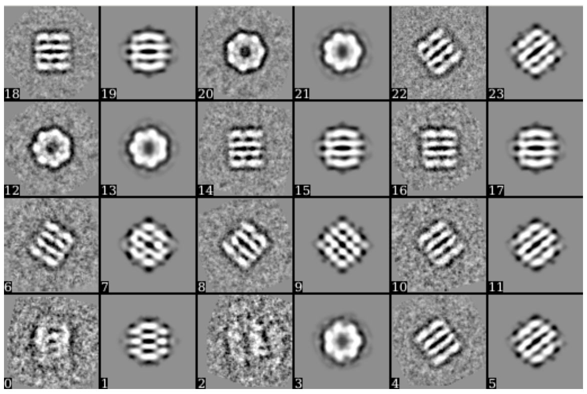
- ➡ When you are uncertain about the quaternary structure of your molecule, tilt validation is a critical test. You collect pairs of images at 0 degrees and typically 10-20 degrees tilt, box out tilt pairs of particles, then run them through a tilt validation procedure against your final 3-D map. This 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 e2refinmulti.py and e2classifyligand.py for purposes of processing such data sets using normal single particle data.
- ➡ Poor angular distribution. If your particles have a 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, discussed at the workshop on Wed 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* = 10, *models* = 8
- 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 agree fairly well (but not as well as the correct structure) with the input data. That is, 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. 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 a couple of 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 the first 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. This time you will get a sequence of model_02_YY files.

21. Removing (some) bad particles

- While the initial model generator runs, or if you prefer, while (22) Refinement runs, we can take a look at one provided utility for reducing the number of bad particles in your sets. The command is e2evalparticles.py, but we will use it through the project manager.
- Select: *particle sets -> evaluate particle sets*
  - In the new window that opens, select bdb:r2d_02#allrefs_06
  - 3 windows will appear, 2 blank, 1 with averages
  - Click on a bad average and it will show the particles included in the average in one window and the particles excluded from the average in the other.
  - Double click on any bad classes (not particles). A blue mark will appear. The buttons for operating on groups of particles (*mark as bad*, etc.) will operate on particles associated with any blue tagged class-averages. There is no way to mark individual particles in this interface. You have to use the set building interface for that.
  - Click '*mark as bad*', then respond yes
  - This marks the particles from the bad classes as bad. Make sure you don't throw away TOO many good particles with the bad.
- Build particle sets again, with a new name, and you will see that the new set is smaller. Use either the old or new set for 3d refinement. Your choice.

22. **3D Refinement**

- We are finally ready to do our first (coarse) 3-D refinement. This is the place where we have a lot of options to set and understand (particularly when we're going for higher resolution). In EMAN2.1, there will be a functional Wizard interface to guide you step by step through all of these options. For the moment, we'll start with a simple initial refinement targeting improving our initial model:
- Select: *3D Refinement -> Run e2refine*
  - *input* = The set containing the particles you plan to use. I would suggest using a ctf_flip_hp set.
  - *model* = The initial model to seed the refinement. Normally bdb:initial_models#model_01_01
  - *iter* = 4
  - *Auto Mask 3D* = checked, default options are a reasonable start.
  - *orientgen*, change params delta=3 instead of 5
  - *sym* = d 7
  - *shrink* = 2
  - *classiter* = 5
  - *classaverager* = ctfw.auto
  - *pad* = 196
  - *recon* = wiener_fourier
  - Launch
- This will take some time to complete on a single processor, even with these basic options. If you continue the tutorial on your own, however, you would normally follow this basic refinement with one targeting higher resolution. Use the same options as before with the following changes:
  - *input* = same as before
  - *model* = the final map from the first refinement, threed_filt_04 (browse for it)
  - *orientgen*, delta=2
  - *shrink* = 1
  - *twostage* = 2
  - *simralign* = refine
  - *classiter* = 1
  - *sep* = 3
  - *classkeep* = 1.5
  - *classkeepsig* = checked
  - *classralign* = refine
  - *m3dsetsf* = checked
  - *m3dpostprocess* = filter.lowpass.gauss
  - *params*: cutoff_freq=0.08
  - If you are running this later on a linux box or mac, you can use threaded parallelism to improve speed. If, for example, you have 4 cores, you would put *thread:4* in the parallel box.

23. Resolution assessment

- We really won't have time for this at the workshop, but once you have a final structure you're happy with, you need to assess its resolution. The original program for doing this is e2eotest.py, and is available from the project manager. Thernew program, e2refine_evenodd.py has not been integrated yet. In EMAN2.1, we plan to have a new refinement program which integrates the capabilities of e2refine_evenodd into the normal refinement process, but this is not done yet.
- The easiest way to run e2eotest.py is through the command-line. Use your operating system browser to look at the .eman2log.txt file in the project directory, and search it for e2refine.py. It should look something like this :

```
e2refine.py --iter=2 --input=bdb:sets#all_ctf_flip_hp --model=bdb:./refine_03#threed_filt_04 --
mass=800.0 --apix=2.1 --automask3d=0.8,30,5,5,30 --sym=d7 --projector=standard --
orientgen=eman:delta=2.0:inc_mirror=0:perturb=1 --simalign=rotate_translate_flip --simaligncmp=ccc --
simralign=refine --simraligncmp=ccc --simcmp=frc:zeromask=1:snrweight=1 --twostage=0 --sep=1 --
classkeep=1.5 --classkeepsig --classiter=1 --classalign=rotate_translate_flip --classaligncmp=ccc --
classralign=refine --classraligncmp=ccc --classaverager=ctfw.auto --classcmp=frc:snrweight=1 --
classnormproc=normalize.edgemean --pad=196 --recon=wiener_fourier --m3dkeep=0.8 --m3dsetsf --
m3diter=2 --m3dpreprocess=normalize.edgemean --m3dpostprocess=filter.lowpass.gauss:cutoff_freq=.
08
```

- To run e2eotest.py, run exactly the same command, but replace ‘e2refine.py’ with ‘e2eotest.py’, and add one more option **--path=refine_02** (or whatever the refine directory you want to test the resolution of is called).
- Running **e2refine_evenodd.py** is exactly the same, except you also need to add **--randomres=18** . This 20 is a resolution somewhat worse than the resolution you are expecting to achieve. ie- if you think you have a 10 Å resolution map, fill in a larger number than 10, say 15.
- The results from both tests can be observed using *e2plotFSC* in the project manager. Note that for **e2refine_evenodd.py**, it may be justifyable to use a threshold of 0.143 rather than the 0.5 we typically use with **e2eotest.py**.
- The test run by **e2refine_evenodd.py** is much more robust than **e2eotest.py**, though for GroEL it may make little difference, reviewers are beginning to ask for this better test in published manuscripts.