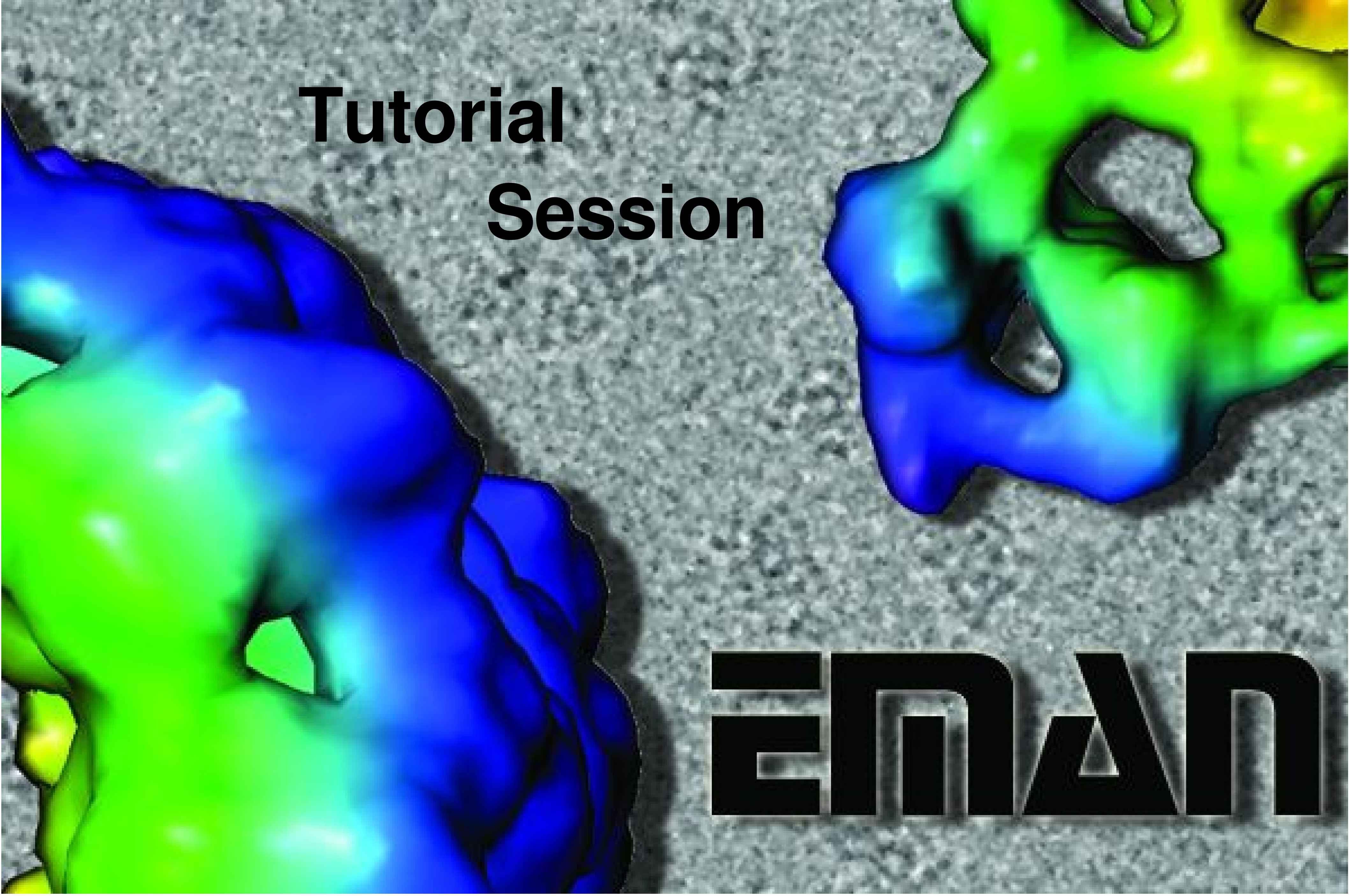


Tutorial Session

National Center for Macromolecular Imaging

EMAN1 Documentation

- <> denotes a parameter to fill in, ie <input file>
- [] denotes an optional parameter, ie [mask=<radius>]
- *italics* denote something to be typed into the computer

example:

proc2d <input file> <output file> [mask=<r>] [mrc] [spider] [pgm]

proc2d file.hed file.spi spider mask=22

- *<program> help* (won't work with python scripts yet)
- *<program>* (shows usage information)
- A few python programs use EMAN2 style options

EMAN2 Documentation

- All programs begin with 'e2' to avoid conflicts
- Standard unix-style arguments with '--' or '-'
- e2proc2d infile.hdf outfile.hdf --edgenorm --apix=2.3
- e2proc2d --help
- e2help.py (doesn't do a lot yet)

GUI

(eman browser)

- *cd ~/data/eman.demo/samples*
- *eman*

GUI Programs

- eman
- boxer
- helixboxer
- ctfit
- qsegment(/chimera)
- qindex
- glmatrix
- v2
- v4
- qplot

Non-EMAN Programs

- chimera

EMAN2 GUI

- e2display.py
- e2boxer.py
- many more to come...

EMAN2 uses PyQt4 + OpenGL for all GUI programs

Canonical EMAN Refinement

- **Image Acquisition**
  - **e2scannereval.py, ctfit**
- Particle Picking
- 2-D Analysis
- Symmetry/Preliminary Model
- Determine CTF Parameters
- High Resolution Refinement
- Post-processing
- Dynamics

Evaluate Data

- *cd ~/data/eman.demo/samples/boxer*
- *ls*
- *e2scannereval.py jj0880f.mrc --norm*
- *e2display jj0880f.eval.mrc*

also

- *box out particles then*
- *ctfit jj5339.f.hed*

Canonical EMAN Refinement

- Image Acquisition
- **Particle Picking**
  - **boxer, batchboxer, e2boxer.py, helixboxer**
- 2-D Analysis
- Symmetry/Preliminary Model
- Determine CTF Parameters
- High Resolution Refinement
- Post-processing
- Dynamics

Particle Picking (Boxing)

- *cd ~/data/eman.demo/GroEL/stage1*
- *ls*
- *cat 000script*
- *boxer <one of the mrc files>*

  try *Boxes->Autobox* after picking 3-4 particles
- *e2boxer.py <mrc file> --box=128 --gui*

  When you get bored ...
- *./000script*

  particles will now be in ../stage2

Particle Picking (Boxing)

- *cd ~/data/eman.demo/samples/boxer*
- *boxer <image>*

Tricks:

- *makeboxref.py groel.mrc sym=d7 invert ang=15*
  (creates a set of reference images from a 3D model for autoboxing)
- *batchboxer input=jj5339.f.mrc auto=.3,.7,.1 dbout=5339.box output=5339.ptcl.hed refimg=best.hed*
  (autoboxes a single micrograph, parameters determined in boxer, refimg specifies reference particles for boxing)

Particle Picking (Boxing)

- *cd /tmp/USERNAME/groel/stage1*
- *batchboxer input=jj5337.f.mrc dbbox=5337.box output=../stage2/5337.hed invert*
  (takes a list of box locations and chops out each particle into a single image stack)
- There are a lot of other particle picking programs out there as well...

Canonical EMAN Refinement

- Image Acquisition
- Particle Picking
- **2-D Analysis**
  - **refine2d.py**
- Symmetry/Preliminary Model
- Determine CTF Parameters
- High Resolution Refinement
- Post-processing
- Dynamics

2-D Analysis

- *cd ../stage1a*
- *cat 000script* *(on a laptop you might remove --iter=2)*

This script will run a 2-D refinement on the data from one micrograph. After running it, look at :

iter.final.img

cls*.lst files contain the particles in individual classes, BUT not in the same order as iter.final.img

Canonical EMAN Refinement

- Image Acquisition
- Particle Picking
- 2-D Analysis
- **Symmetry/Preliminary Model**
  - **startcsym, symsort**
- Determine CTF Parameters
- High Resolution Refinement
- Post-processing
- Dynamics

First Preliminary Model

Skip stage2 for the moment

- *cd ../stage3*
- *cat 000script*

This will make an initial model from the class-averages assuming C7 symmetry

- *source 000script*
- *v4 threed.0a.mrc*

Note 'fixrot' version of startcsym, try that

First Preliminary Model

Look at:

- *avg.img, ali.img, cls0000.lst, cls0001.lst*
- *classes.img, sym.img, threed.0a.mrc*

Reliable Preliminary Model

- *cd ../stage4*
- *cat 000script*

This will refine the raw data vs an initial model we just determined. Either copy the stage3 model, or use makeinitialmodel.py

While it runs...

- stage4a - refine from a 'makeinitialmodel.py' that looks kind of like GroEL
- stage4b - D7 symmetrized random model
- stage4c - Single Gaussian blob
- stage4x - refine from refine2d class averages

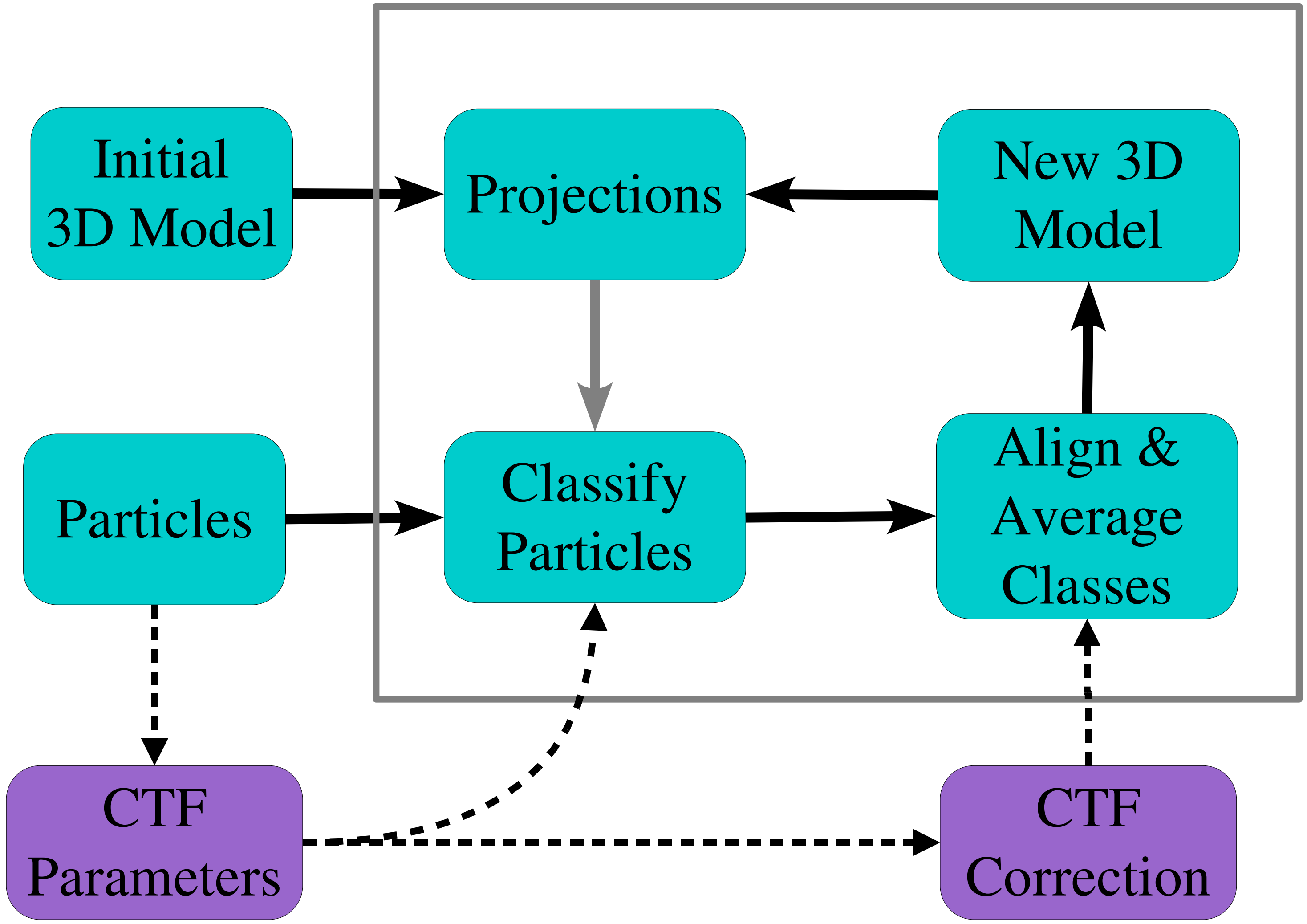
Basic Image Processing

- *iminfo <file>   [all] [stat]*
  (image information)
- *proc2d <infile> <outfile> [options]*
  (generic 2d image processing)
- *proc3d <infile>*
- *proc3d <infile> <outfile> [options]*
  (generic 3d image processing)
- *procpdb.py <infile> <outfile> [options]*
  (simple PDB file processing)

Canonical EMAN Refinement

- Image Acquisition
- Particle Picking
- 2-D Analysis
- Symmetry/Preliminary Model
- **Determine CTF Parameters**
  - **fitctf, ctfit, applyctf**
- High Resolution Refinement
- Post-processing
- Dynamics

CTF Correction - EMAN



Possible Corrections

- Phase flipping (+)
  - Astigmatism (-)
- Amplitude contrast correction (+)
- Envelope function correction (+)
  - Drift (-)
- Inter-micrograph weighting (+)
- Wiener filtration (+)

+ Corrections in EMAN1

CTF Parameter Determination

- *fitctf* - only if you have a real or simulated structure factor curve
- *ctfit* - check results, or fit if you don't have a structure factor, evaluate micrograph quality

- ctfit used with 3+ micrographs can be used to produce a pseudo-structure factor for fitting

CTF Parameters

- *cd ../stage2*
- *cat 000script*
- *./000script*
- *ctfit *.tnf*

Use ctfit to fix the automatic fitting...

- *./000script2*
- *eman*
- *look at files in ../raw and ../raw.w*

Canonical EMAN Refinement

- Image Acquisition
- Particle Picking
- 2-D Analysis
- Symmetry/Preliminary Model
- Determine CTF Parameter
- **High Resolution Refinement**
  - **refine**
- Post-processing
- Dynamics

High Resolution Refinement

- `cd ../stage5`
- *cat 000script*
- *precomputed (looong job)*
- *eman*
- *look at:*
- *classes.*.img, x.*.mrc*
- *analysis -> convergence*

Refine Options

- Projection:

mask=<r>
proc=<n>
sym=<cn,dn,etc>
ang=<dang>
tree=<2,3>
perturb

- Classification:

dfilt, phasecls, fscls
refine
sep=<n>
tree=<2,3>
shrink=<n>
maxshift=<rad>
precen, usefilt, slow
mask=<r>,imask=<r>
amask=<r>,<thr>,<iter>
proc=<n>
ctfc=<res>, ctfcw=<SF> or median

- Class Averages:

dfilt, phasecls, fscls
refine
classiter=<n>
classkeep=<sig mult, but not 0>
mask=<r>,imask=<r>
amask=<r>,<thr>,<iter>
proc=<n>
ctfc=<res>, ctfcw=<SF> or median
maxshift=<r>
precen,slow
euler2=<oversmp>

Refine Options

- 3D Reconstruction:

hard=<phase err>
pad=<size>
sym=<cn,dn,etc>
3dit=<n>,3dit2=<n>
mask=<r>,imask=<r>
proc=<n>
collapse=<alt thr>

- Postprocessing (betw iter):

xfiles=<A/pix>,<mass in kd>,<ali to>
amask=<r>,<thr>,<iter>
filt3d=<lp rad>

- Typical options to use for a high-resolution refinement (EMAN 1.6):

mask
proc (for clusters)
hard
ctfcw (requires a SF file)
sym (exclude if no symmetry)
ang
pad
classkeep
classiter (3 or 0 at the end, 1-2 invalid)
xfiles
amask (must also use xfiles)
refine
dfilt (must also use ctfcw)
usefilt (start.filt.hed must exist)
sep (with oversampled ang=)

Refine Options

REQUIRED:

- <total iter> - Final number of iterations you wish to be complete in this directory. ie - if 5 are complete and you specify 6, 1 more iteration will run
- mask=<rad> - A circular mask to be applied virtually everywhere. Should be a few pixels larger than the largest radius of your particle
- ctfc=, ctfcw= OR median - CTF correction options. ctfc takes a filter resolution in Angstroms, but ctfcw is far superior. ctfcw takes the name of a 1D structure factor file used when fitting the data (MUST be the same file). median does no CTF correction at all. Naturally the data must be properly preprocessed for this option to function, otherwise crashes or invalid results are likely.
- hard=<phase err> - Rather obscure. This option determines when a class-average should be excluded from the 3D reconstruction process. 25 is generally good. see 'make3d' for more info.
- sym=<c*n*,d*n*,oct,icos> - For asymmetric objects, either omit this specification, or use 'c1'. cn denotes a single n-fold rotational symmetry (about the z-axis), dn denotes n-fold dihedral symmetry (cn with n 2-folds in the x-y plane), oct is octahedral (2-3-4, symmetry of a cube), icos is icosahedral (2-3-5).
- ang=<dang> - Angular spacing between projections. Smaller numbers produce more projections, usually between 1-10. Usually 90/n where n is any integer.

Refine Options

REQUIRED:

- pad=<size> - This is used to reduce artifacts in Fourier reconstruction. Should be about 25% larger than your model in most cases, and have small prime factors, ie - model size 48 -> pad=64, 64-> pad=96, 100-> pad=128
- classkeep=<sig mult> - This determines how many raw particles are discarded for each class-average. This is defined in terms of the standard-deviation of the self-similarity of the particle set. A value close to 0 (should not be exactly 0) will discard about 50% of the data. 1 is a typical value, and will typically discard ~10-20% of the data.
- refine – Almost always specified. This will increase the alignment accuracy (with a speed penalty). May be omitted for early rounds of refinement.
- classiter=<n> - Generation of class-averages is an iterative process. Rather than just aligning the raw particles to a reference, they are iteratively aligned to each other to produce a class-average representative of the data, not of the model. ie - this eliminates initial model bias, typically 8 in the early rounds and 3 in later rounds, 0 may be used at the end, but model bias may result.
- dfilt, fscls, phasecls – Mutually exclusive options that determine similarity criteria used for classification and quality comparisons. dfilt requires ctfcw, but generally produces the best results. phasecls uses mean phase error. fscls uses Fourier ring correlation. dfilt is an optimized variance with a matched filter.

Refine Options

OPTIONAL:

- filt3d=<rad> - Applies a lowpass filter to the 3D model between iterations. This can be used to correct problems that may result in high resolution terms being upweighted. <rad> is the same as for the 'lp=' option in proc3d.
- sep=<n> - This interesting option causes each particle to be assigned to the n best classes, not just the single best class. This may be used to smooth and improve fine details in the final stages of a high resolution refinement. Generally used with oversampled ang= values.
- tree=<2,3> - This can be a risky option, but it can produce dramatic speedups in the refinement process. Rather than comparing each particle to every reference, this will decimate the reference population to 1/4 or 1/9 of its original size, classify, then locally determine which of the matches is best. Is is safest in conjunction with very small angular steps, ie - large numbers of projections. The safest way to use this is for the initial iterations of refinement (then turn it off for the last couple of iterations). May not work on certain cluster configurations.

Refine Options

SUGGESTED (not required):

- xfiles=<A/pix>,<mass>,<ali to> - This is a convenience option. For each 3D model it will produce a corresponding x-file: threed.1a.mrc -> x.1.mrc. Based on A/pix and mass (in kd), the x-file will be scaled so an isosurface threshold of 1 will contain ~ the specified mass. 'ali to' is an iteration number. ie - if 'ali to' is 4, then x.7.mrc would be aligned in 3D to x.4.mrc. x.3.mrc would not be aligned at all. Often this is set to a large value, like 99.
- 3dit=<iter>, 3dit2=<iter> - **No longer suggested.** They apply a real-space iterative reconstruction technique to the 3D model to clean up artifacts caused by Fourier techniques. Typically 3dit=1 and 3dit2=2. Rarely necessary, and can produce other artifacts.
- amask=<r>,<threshold>,<iter> - This option applies an automatically generated 'form fitting' soft (Gaussian) mask to the model after each iteration. The mask generation is generally quite good. See proc3d option automask2 for details. This option can only be used in conjunction with xfiles=, since selection of the threshold requires proper volume normalization. This option can have a profound effect on proper convergence, but should be used with caution. Too tight a mask will produce artificially inflated resolution values. To use properly, insure that the mask does not cut through any significant density in the final model, and that the mean value is already ~0 at the mask position. This is somewhat akin to solvent flattening.

Refine Options

TYPICAL (not required):

- shrink=<n> - Another option that can produce dramatic speed improvements. In some cases, this option can actually produce an improvement in classification accuracy. This option scales the particles and references down by a factor of n before classification. Since data is often heavily oversampled, and classification is dominated by low resolution terms, this can be both safe, and actually improve classification by 'filtering' out high resolution noise. Generally shrink=2 is safe and effective especially for early refinement. In cases of extreme oversampling, larger values may be ok. This option should not be used for the final rounds of refinement at high resolution.
- euler2=<oversmp factor> - This option should produce improvements in convergence and reconstruction quality, but has produced mixed results in the past. *Not recently tested*. It adds an additional step to the refinement process in which class-averages orientations are redetermined by projection-matching. The parameter allows you to decrease the angular step (ang=) used to generate projections. ie - 2 would produce projections with angular step of ang/2. It may be worth trying, but use it with caution on new projects. Untested with some new refine options.

Refine Options

EXPERIMENTAL (not required):

- usefilt - This flag allows one to use arbitrarily filtered raw particles for classification purposes, but still use the unfiltered data when generating the actual reconstruction. To use this option, apply filter the data from start.hed into start.filt.hed. This option is now recommended for most reconstructions where ctfcw= is used. The particles should be filtered with proc2d wiener=.
- collapse=<alt max> - This option is used to correct for a problem wherein side views of particles with Dn symmetry can iteratively 'drift' from alt=90 up to smaller altitudes. **This problem should no longer exist if dfilt, phasecls or fscls are used.**
- perturb - This is a very new option, which is potentially useful, and at worst should be harmless. However, it has not been well characterized yet. Rather than generating Euler angles at evenly spaced positions, it adds some randomness to the positions. This should produce a more uniform distribution of data in 3D Fourier space and reduce Fourier artifacts.
- imask=<rad> - This option was designed to improve classification for virus particles containing non-icosahedral DNA/RNA. It has not been tested in a long time and may or may not function.

Canonical EMAN Refinement

- Image Acquisition
- Particle Picking
- 2-D Analysis
- Symmetry/Preliminary Model
- Determine CTF Parameter
- High Resolution Refinement
- **Post-processing**
  - **procpdb.py, pdb2mrc, foldhunterP, masksym.py**
- Dynamics

Simple Docking and Segmentation

- *cd ../stage6*
- *./000script*
- *chimera fh.0.ent shrunk.mrc ../stage5/threed.8a.mrc*
- *./000script2*
- *use qsegment*
- *use chimera*

Easier and better ways of doing this will be discussed later in the workshop.

Canonical EMAN Refinement

- Image Acquisition
- Particle Picking
- 2-D Analysis
- Symmetry/Preliminary Model
- Determine CTF Parameter
- High Resolution Refinement
- Post-processing
- **Dynamics / Heterogeneity**

2-D Analysis Step 1

- Run refine2d.py on a large particle set
- Use a relatively small number of classes, so you have a large number of particles per-class
- Do not use --finalsep
- The resulting classes should mainly represent different orientations rather than dynamics

2-D Analysis Step 2
(particle extraction)

We need to extract particles in a single view:

- *for i in cls* lst*
- *do proc2d $i avgs.hed average*
- *done*

Identify particles from avgs.img (rather than iter.final.img), then for each class:

- *~/EMAN/python/lstsub.py cls####.lst myptcl.hed start.hed*

This will copy the original particles associated with the selected classes into myptcl.hed. Assumes the file you ran refine2d.py on was called start.hed.

2-D Analysis Step 3
(more refinement)

- Run refine2d.py on myptcl.hed in a new subdirectory.
- e2stacksort.py iter.final.hed similar.hed
- Manually edit similar.hed and delete 'bad' particles (wrong orientation or low quality)
- e2stackanim.py similar.good.hed similar.gif
  (this will produce similar.gif, a gif animation showing the level of motion/heterogeneity in your data.

multirefine

- No demo data for this:
- create heterogeneous start.hed/img file
- Say we need to do a 3-model multirefine
- Make this directory structure:

```
./start.hed            # all particle data
./start.img
./0/threed.0a.mrc      # Initial model 1
./1/threed.0a.mrc      # Initial model 2
./2/threed.0a.mrc      # Initial model 3
```

If you have a structure factor file a copy should be in all 4 directories.

multirefine

- Initial models may be almost identical, ie - the same model with a tiny amount of added noise.
- For faster convergence, make different models based on the refine2d.py results
- multirefine takes almost the same options as refine, with a second fixed parameter indicating the number of models (ie - 'multirefine 6 3 ...')
- multirefine does not store class information. If you wish to use it to split the particle data, you must let an iteration run to completion, not interrupt it early

multirefine

- When multirefine completes, usually you follow-up with a single refine on the split particles
- From the directory containing start.hed:

```
mkdir r0
for i in 0/cls*lst
do proc2d $i r0/start.hed first=1
done
cd r0
refine ...
```

Parallel Processing

- Uses SSH
- Multicore (SMP) single machines
  - use 'proc=<n>'
  - use the non-cluster compile of EMAN
- Clusters or sets of workstations
  - Use the 'cluster' compile of EMAN
  - Must have the data directory cross-mounted to all nodes in the same place
  - Must be able to SSH between nodes without a password (ssh-agent or proper config)
  - Best if the 'head' node is physically connected to the disk
- 'runpar test'
- Works with OpenPBS or LoadLeveler

Generating a Hybrid Structure Factor

- ctfit <3 or 4 particle sets>
- set micrograph parameters
- load existing structure factor (crystb.sm for demo)
- use alternate background fitting mode to make it easier
- fit each data set independently
- turn on the predicted structure factor plot, zoom in below 1st zero
- 'tweak' the parameters to make the low resolution structure factors match
- Write down the cutoff spatial frequency (before the 1st zero)
- enable all plots, then Advanced --> Save 1 Column. Save column 11
- Run 'sfmerge.py <ctfit file> <other SF file> <cutoff> mysf.txt'
- Edit mysf.txt. If the first line is 'nan' remove it
- Refit your data using mysf.txt as a structure factor