This list serves as an initial outline for a more comprehensive documentation of the workflow for the emdrp pipeline. The starting point is aligned raw data in Knossos format (the alignment and writing of data in Knossos format is done exclusively by Kevin).

See also EM\_pipeline\_time\_mem\_storage.xls for details of run time, memory and storage requirements and also abbreviated version of workflow dependencies (schematic).

30 March 2017, Paul Watkins

1. create raw hdf5 from Knossos cubes
   1. make\_hdf5\_from\_knossos\_raw.m run on blue
   2. copy raw hdf5 to nas
2. create downsampled version in xy to get closer to isotropic voxels
   1. dpResample.py run on blue
   2. copy ds raw dhf5 to nas
3. manually label at least 3 representative volumes of size 256x256x32 if not using ortho networks OR 128x128x128 if using ortho networks, avoid edges because networks need context outside labeled area
   1. for watkinspv, 3 volumes takes about 40 hours of full time work, varying depending on the complexity of the neurites being traced (areas with large dendrites and cell bodies go much faster)
   2. label ECS with label 1 and label all other neurites / cell bodies with unique label ids, trace up to but not including the membrane that defines boundaries
   3. decide on smoothing and minsize parameters and run label\_checker.sh (dpCleanLabels.py steps) on blue
   4. create label hdf5 label\_maker.sh run on blue
   5. copy label hdf5 to nas
4. train convnets on all training data
   1. setup and test new .ini / architecture combo and test on blue, need to either sample or comprehensively iterate over some portion of dataset to estimate overall mean and std used by preprocessing in parseEMdata.py
   2. decide on combination of networks (typically 4)
   3. train MORE networks than needed incase of gpu errors or changing mind about using ortho networks, etc
   4. train networks on gpu clones
   5. copy trained convnets to all clones (so far found that best division of labor is each gpu clone processes 1/number\_of\_clones of the dataset, and each of 4 gpus on each clone uses a different trainined network)
5. train convnets using x-fold cross validation (needed for agglomeration training)
   1. use same number of networks per cross validation
   2. may have to modify x-validation if more than about 8-10 training volumes (either by sampling of leave out one, or some other leave out combinations)
6. copy trained convets to all clones
7. export trained convnets over all of dataset or some pre-defined portion of dataset