# Justification

After improving the hot three loops of the first few iterations of ML2D, we discovered that the remaining iterations were dominated by the BackProjection effort.

This document tries to describe the ML2D algorithm in a manner that is suitable for reducing and attacking the time spent in the later iterations

# Background material

<http://www2.mrc-lmb.cam.ac.uk/relion/index.php/PreProcessing>

<http://www.sciencedirect.com/science/article/pii/S0022283611012290>

# The Procedural Flow

The MKProgram() does three phases, not in a loop

MLOptimizer::readImages();

MLOptimizer::prepare();

MLOptimizer::iterate();

## readImages

MLOptimizer::readImages() fills

double\* images\_data;

with consecutive nr\_local\_images images, each ori\_size\*ori\_size double values. No attempt is made to align them, but they are likely to be on addresses that are multiples of 8\*4\*4 because the side lengths (ori\_size) are often divisible by four.

## prepare

Allocates a lot of data structures needed to hold the shared state and the temporaries that we don’t want to repeatedly allocate and deallocate for performance reasons

For each image, calls

* checkNormalizedImage, which tests that the area outside a central circle has an average in -0.5..0.5 and a std dev between 0.5 and 2.
* calculateSumOfPowerSpectraAndAverageImage, which produces
  + an sum of the images in Mavg (later divided to get average)
  + the fft for each image, which is promptly discarded
  + wsum\_sigma2\_noise and wsum\_sumw\_group

For each class

* sets each class’s initial image to be the average image (yes, all the same!)
  + yes, duplicate copies of it
* replaces them from the reference file if one was provided
  + currently only supports one class

Calls

* setSigmaNoiseEstimates
  + model\_sigma2\_noise is set to wsum\_sigma2\_noise[i] / ( 2. \* wsum\_sumw\_group )
* applyLowPassFilter does some FFT improve inv-FFT to each class
  + since all the same, seems redundant
* initialiseDataVersusPrior
  + updates the per-class tau info.

## iterate

MLOptimizer::iterate() is the main loop.

Loops calling

* updateImageSizeAndResolutionPointers()
* prepareExpData()
* expectation()
* -- here is where it shares the data across the MPI nodes
* maximization()
* updateCurrentResolution()
* endExpData()

These are described in the following sections

### updateImageSizeAndResolutionPointers

Decides various sizes and fills in Mresol\_fine and Mresol\_coarse.

### prepareExpData

For every class, call

* computeFourierTransformMap
  + makes a copy of the class with 0’s padding around it
  + centerFFT which is shifting an image in both directions
    - odd giving we just padded the image but not performance critical
  + FourierTransform is applied to the shifted padded class image.
  + The power of each of the concentric rings is computed

Clear lots of large temps

### expectation

This loops. Each iteration

* Chooses which classes and which images this node will be doing
* Copies the images for this node, and their metadata
* getImagesFourierTransformsAndCtfs() does the following to all assigned images.
  + Copies, normalizes, translates, and FFT’s it
  + Various noise and contrast transfer functions get applied to it
* expectationSomeParticles()
  + tries two different granularities of translation and rotation, doing the following
    - there are slight random variations
  + getShiftedImagesCtfsAndInvSigma2s() shifts all the images by the translations in fourier space
    - this is not much worse than a copy
  + getReferenceAllOrientations() rotates all the class images in fourier Space
  + getAllSquaredDifferences() for every image and class image
    - applies the ctfs to the frefs
    - does the diff calculation
  + convertSquaredDifferencesToWeights() for every image and class image
    - processes diffs into weights
  + findAllSignificantPoints()
    - for every image
      * determine which classes it fits best
  + for the finest granularity
    - updateOtherParams()
      * TBD what this is doing
    - getShiftedImagesNomask()
      * shifts the images – unclear by how much
    - backProjection()
      * I’m missing the undo of the rotation…
    - storeWeightedSums()
      * ?
* monitorHiddenVariableChanges() checks what has changed
  + TBD where it was recorded
* update\_metadata(my\_first\_image, exp\_metadata, exp\_nr\_images);
  + Copies the data from the exp temp to the result

### maximization

For each class

* calls reconstruct()

then maximizationOtherParameters and updateOverallChangesInHiddenVariables

### updateCurrentResolution

### endExpData

# The Data Flow

images\_data is read by calculateSumOfPowerSpectraAndAverageImage, prepare, expectation.

They are sometimes masked, then they are summed into Mavg and FFT’ed and feed into wsum\_sigma2\_noise by calculateSumOfPowerSpectraAndAverageImage.

Some copied into exp\_imgs, by expectation.