SIMPLE 2017 Development Document

Responsibilities: Hans, Cyril, Michael?, Susan&Hans

RANDOM STUFF

* Multi-state with different point-groups (all subgroups)
* Make distributed flow for scale that operates on split stacks so that scaling can be done in parallel
* Random selection of images for prime2D init need to take state=0 into account
* Need to have another go at unsupervised picking (massively parallel mass-centring)
* Class for memory allocation simple\_alloc. Keep track of mem exceptions and profile memory (hash table).
* Go over all random number generations and see if we can use matrices instead
* Replace integer random number generator with intrinsic one (Guide to Fortran 2008)
* mailx -s "mail from prime2D" hans.elmlund@monash.edu < from\_prime2D
* It would be useful to write out a class-average param file that contained class average identifier and number of particles in it - then the ordering by population could be done as a visualisation thing within the gui rather than a genuine reordering of the stack

STREAMING MODE (half boring)

* Relion-style input mode for substacks, one per movie (need two new programs (1) prime2Dstream in shared memory that has filetab input instead of stack and doclist, i.e. unidocs in replacement of deftab, (2) prime2Dstream in distr\_stream\_commander that monitors unidocs to identify a chunk of suitable size based on ctfres and df\_min/df\_max filter)
* unidoc: everything written to cwd
* same naming convention for picker as for others
* put in absolute path in unidoc

HIGHRES (not so boring)

* Down-scaling/LP strategy: 10, 8, 6, 4, need distributed down-scaling workflow

HETEROGENEITY

* Automatic workflow het\_from\_cavgs: (1) refine=snhc (2) refine=no X 2 resolution updates, as in ini3D
* Weighted multi-resolution state sorting by fitting B-factors

NEW DEVELOPMENTS

* Probabilistic SO(3) scatter search for high-resolution refinement
* implement tilt test

EXCEPTION HANDLING

* exception handling class that makes more sensible outputs (especially when running the code in distributed mode). We need to create a database of exceptions and then have ONE control point where we check the stack for errors and report what has failed (similar to the command line dictionary). JOB)\_FINISHED should communicate

Known bugs

None at the moment

Compilation

* port to ifort (Intel compiler)
* port to PGI (Portland group, with CUDA-FORTAN)

Documentation

Need to get a html code doc generator in place and fix the doc of every class. Will FORD source code documentation provide the solution?

<http://fortranwiki.org/fortran/show/FORD>

<https://github.com/cmacmackin/ford>

<http://jacobwilliams.github.io/json-fortran/index.html>

Books/webpages

* Structured parallel programming
* Structure and Interpretation of Computer Programs
* J-P Morrison. Flow-Based Programming: A New Approach to Application Developments. CreateSpace, 2nd ed.
* Mathematical Foundations of Imaging, Tomography and Wavefield Inversion
* Geometric Algebra for Computer Science (Revised Edition): An Object-Oriented Approach to Geometry (The Morgan Kaufmann Series in Computer Graphics) 1st Edition
* The Princeton Companion to Applied Mathematics
* Practical Machine Learning: http://www.computervisionmodels.com/
* Applied Stochastic Modelling, Second Edition (Chapman & Hall/CRC Texts in Statistical Science) 2nd Edition http://szeliski.org/Book/