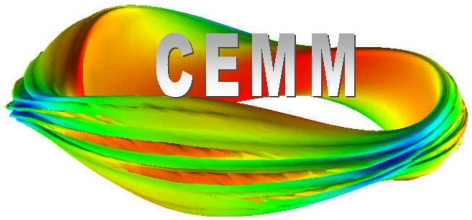
1) Results from implementing parallel I/O and interfacing NIMROD with FFTW

2) Discussion of plans for integrating FLUXGRID and NIMEQ

Jacob King, Scott Kruger
Tech-X Corporation

NIMROD Team Meeting Summer 2014

TECH-X
SIMULATIONS EMPOWERING YOUR INNOVATIONS



Part One: Results from implementing parallel I/O and interfacing NIMROD with FFTW

Parallel HDF5 I/O and FFTW overview

- Address for each topic:
  - Why and what (motivation)
  - How (implementation)
  - Results (performance comparisons)

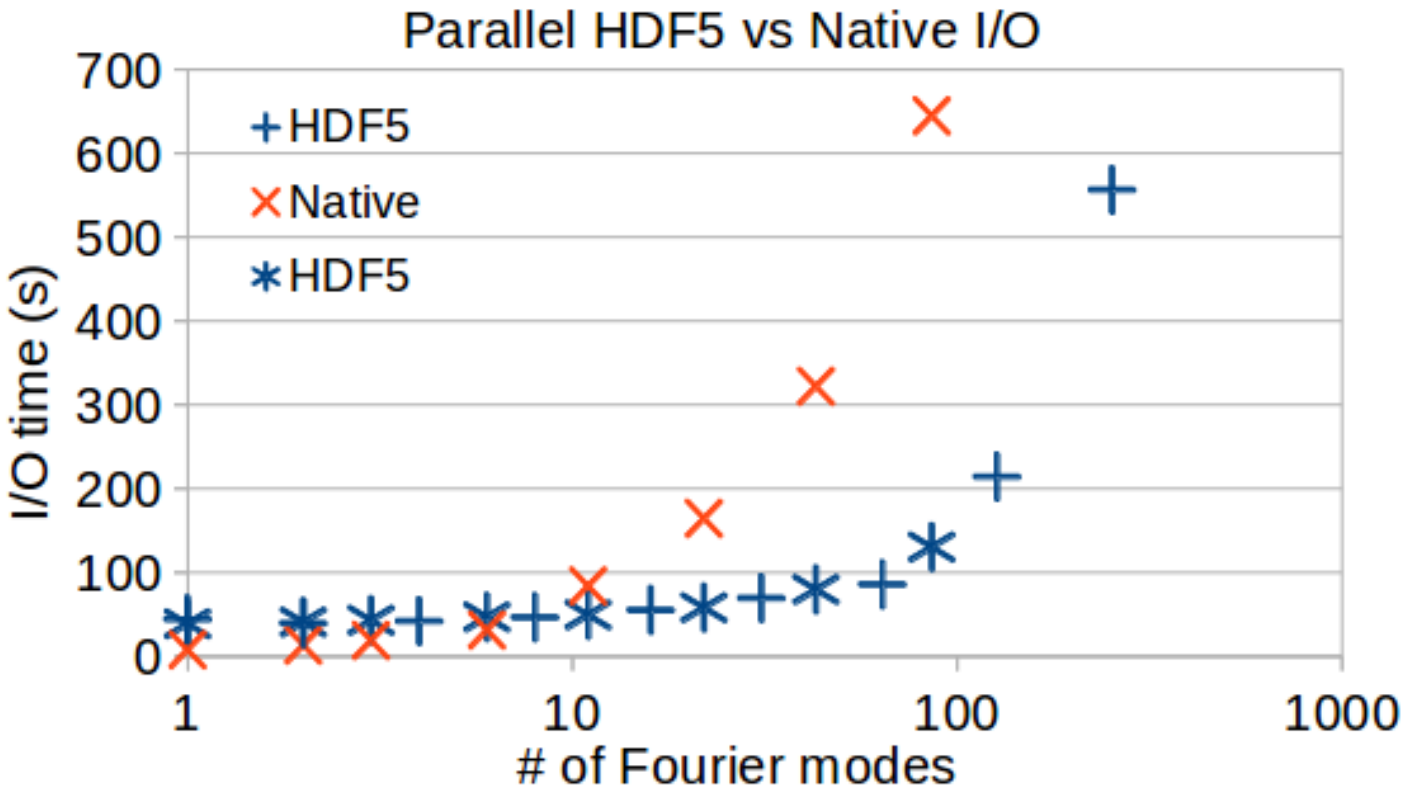
Parallel HDF5 I/O: What and Why

- HDF5 (Hierarchical Data Format) is a portable, extensible data format that stores datasets and attributes within a directory-like structure.
- The HDF5 library can do parallel I/O and typically built and optimized by the vendor on HPC machines (Hopper, Edison, Titan and Mira).
- Parallel I/O has the potential to provide better parallel performance than serial I/O
- The HDF5 format can be marked up with additional datasets and is very flexible (e.g. fluxgrid equilibrium profiles can be written to the initial dump file for diagnosis).

Parallel HDF5 I/O: implementation

- Only layer zero participates in I/O.
- Each block is written by layer zero with fields as a separate datasets.
- Every processor in layer zero must write attributes and allocate the space for the datasets, but only one process writes each dataset.
- Immediate visualization is possible through VizSchema markup which enables multidomain plots.
- File sizes are comparable to the binary format.
- HDF5 output is enabled if h5dump=T in nimrod.in (also set dump_file='dump.*****.h5').

Parallel HDF5 I/O is faster for large jobs and further optimization is likely.



256x64pd5 mesh
1 read
1 write
Nlayers = nmodes
Timings from Mira

Loop time for 40 steps varies between 400 and 700 seconds.

- Parallel HDF5 I/O is shown with default I/O environment parameters (no tuning).
  - There are a myriad of I/O tuning parameters that differ from HPC machine to HPC machine.
  - Further optimization of parallel I/O is likely possible at both the small and large scales.

Demonstration of HDF5 capabilities

- NIMROD HDF5 file generation
- Useful tools:
  - Hdfview
  - Visit

- Comments on the file format?

FFTW: What and Why

- FFTW (Fastest Fourier Transform in the West) is an open source FFT library.
- The primary motivation for linking FFTW is support for an arbitrary number of Fourier modes.
  - It is preferred that jobs on Mira (Blue Gene Q at ANL) use a node count that is a power of two.
  - This is difficult to achieve with NIMROD's layer decomposition for dealiased FFTs which seems to be an efficient prime number generator (# of modes = 2, 3, 6, 11, 22, 43, 86, 171, etc).
- FFTW is a well-written, vectorized C library with a good Fortran interface.
- The implementation is still most efficient for DFTs with numbers of Fourier modes that have small prime factorizations.

FFTW implementation

- FFTW is used when the library is linked. Instead of lphi, when FFTW is linked nphi can be specified directly (or it defaults to nphi=2^lphi if not set).
- FFTW creates a plan (the library determine the most efficient method to use for a given DFT) on the first call used in all subsequent calls, thus all DFTs must be of the same size with the current implementation.
- Data is copied and FFTW memory allocation is used to enable vectorized operation.
- FFTW's Fortran 2003 ISO_C_BINDING module Fortran bridge implementation type safe and easy to use.
- Dealiasing is enforced by zeroing sections of the arrays after the DFT to Fourier space.

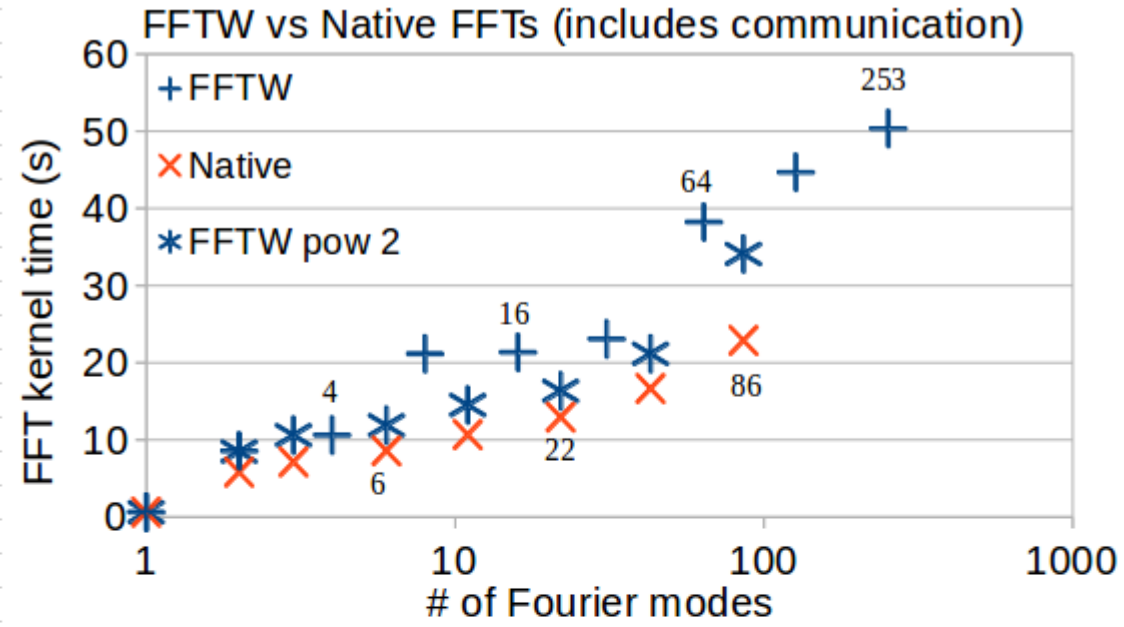
FFTW allows for better domain decomposition.

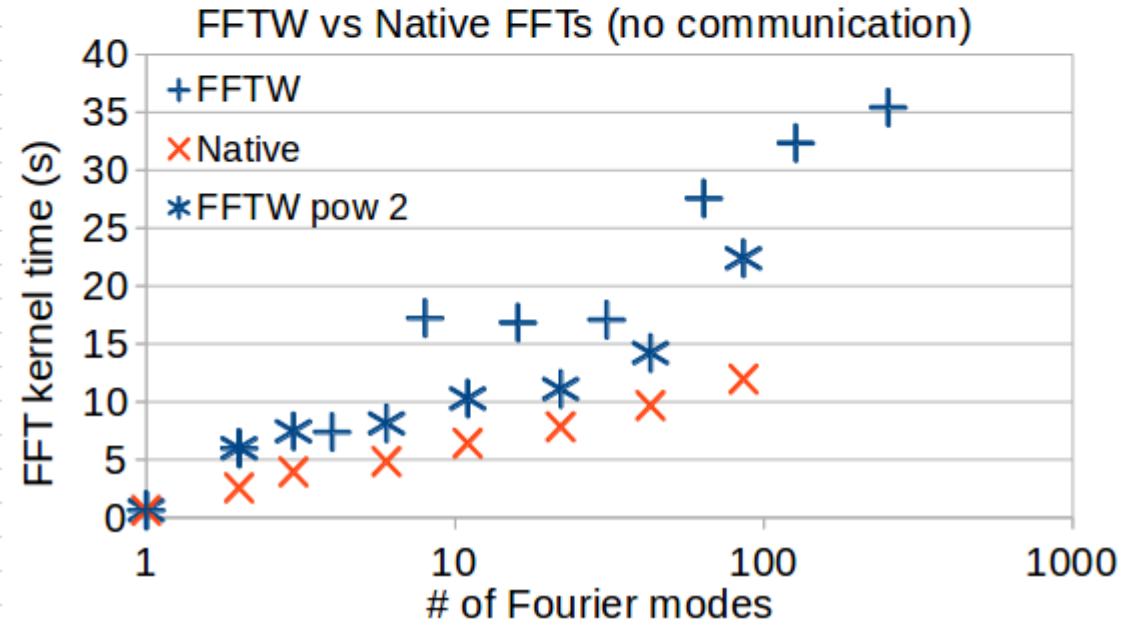
nphi	factors	Nmodes (dealiased)	Efficiency (rel pow 2)
1	1	1	1
4	2^2	2	1
9	3^2	4	1
21	3x7	8	1
45	3^2x5	16	1
90	2x3^2x5	31	31/32=0.969
189	3^3x7	64	1
378	2x3^3x7	127	0.992
756	2^2x3^3x7	253	0.988

- Nphi is chosen to generate a number of dealiased modes close to a power of two with a small prime factorization.
- We could use large prime factors and generate a number of dealiased modes that is an exact power of two, but this may not be computationally efficient.
  - Further testing is needed.

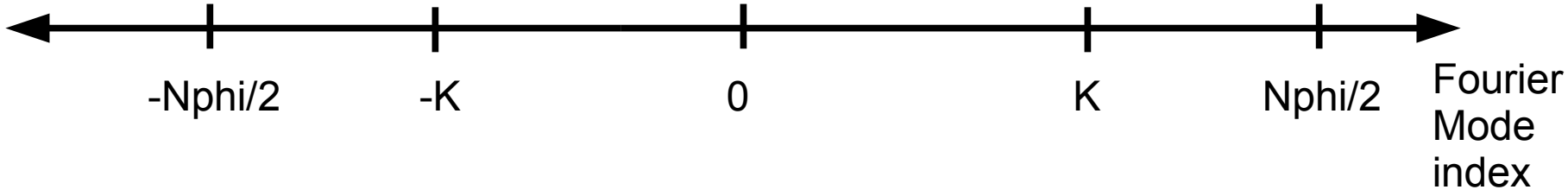
Preliminary performance comparison on Mira shows FFTW is slightly slower than NIMROD's native DFT alg.

- The best performance comparison is on cases with the same number of modes.
- The FFTW relative performance can probably be explained by either the additional copy or optimizations by the native DFTs for dealiasing.
- The DFT kernel (no communication) is 2-5% of the loop time so the overall run time impact of using FFTW is small.
- Same cases as I/O timings.





The FFTW implementation allows for new dealiasing strategies.



- If two modes i and j with i+j>Nphi/2 the modes will quadratically alias to i+j-Nphi.
- To dealiase, choose K and truncate the spectrum such that abs(i+j-Nphi)>K for all i and j where i+j>Nphi/2 → Nphi-2K>K or K=Nphi/3.
- Simliarly, one may formulate dealiasing for triple products: choose K and truncate the spectrum such that abs(i+j+k-Nphi)>K for all I, j k where i+j+k>Nphi/2 → Nphi-3K>K or K=Nphi/4.
- Since the FFTW simply truncates the spectrum, we have changed the dealiase input parameter to an integer such that K=Nphi/dealiase.
  - This needs testing, but could be an interesting test of spectral convergence for nonlinear cases.
  - The native FFT dealiase behavior is preserved.

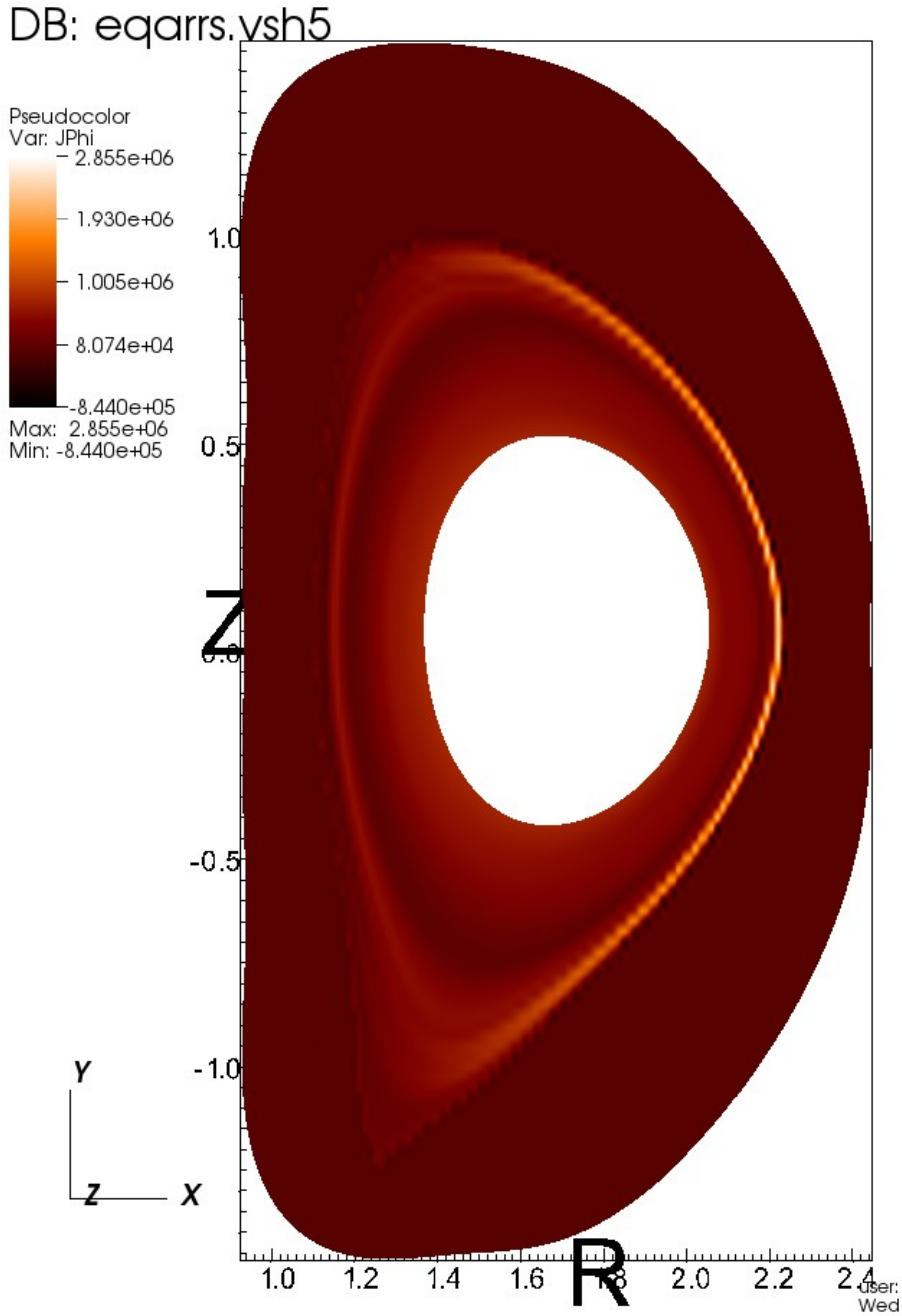
Summary

- Neither of these capabilities are enabled by default.
- HDF5 data format can be useful if either one prefers the format or for performance on large jobs.
- FFTW is useful for specific domain decompositions.
- Both implementations work but still need considerably more testing!

Part Two: Discussion of plans for integrating FLUXGRID and NIMEQ

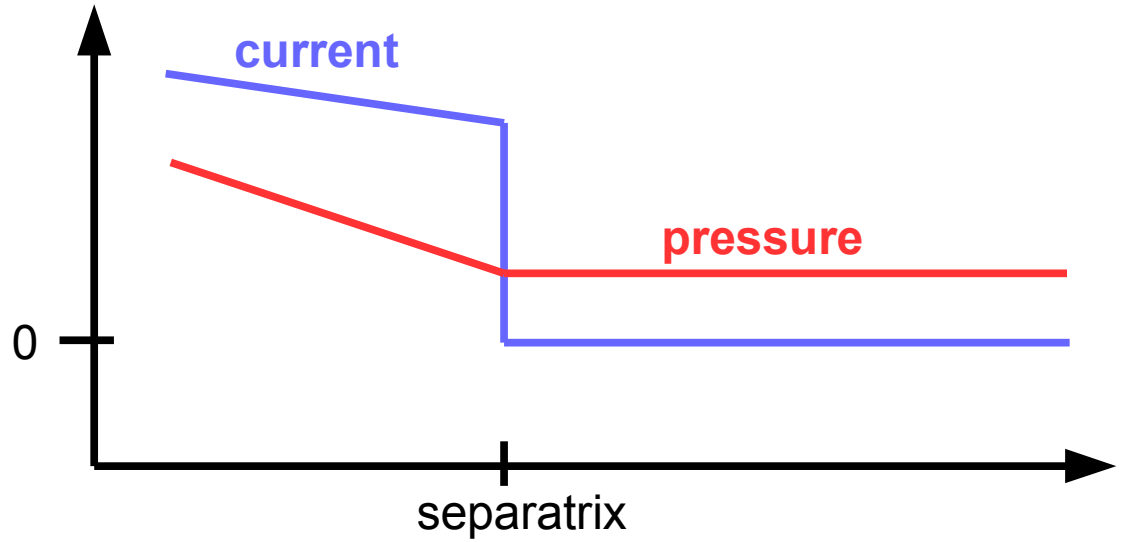
Computations with low resolution reconstructions may be corrupted by the input file resolution.

- Fluxgrid currently maps ψ from the reconstructed grid onto the NIMROD grid
- Low resolution reconstruction can lead to small scale artifacts in the equilibrium.
  - In particular, the extended-MHD operators, many of which involve high-order derivatives, seem to be sensitive to these artifacts.
- Figure: Toroidal current mapped to a 48x512pd5 NIMROD grid from a 128x128 reconstructed grid.



Additionally, reconstructions often generate discontinuous current profiles across the separatrix.

- The pressure is assumed to be constant outside the separatrix.
- Often the plasma pressure profile leads to a discontinuous pressure derivative at the separatrix.
- This can be problematic for edge cases.



We intend to couple FLUXGRID and NIMEQ

- F and p within the plasma would be determined by FLUXGRID.
- The boundary condition for ψ at the wall would also use the FLUXGRID values.
- The FLUXGRID mapping would provide an initial guess for NIMEQ.
- Multiple iterations between the two codes would be performed where the previous NIMEQ solution is used to generate a new grid and provide a new solution until a tolerance threshold on the change of the fields is met.
- Convergence properties are unknown with this approach, but convergence will likely require a decent initial guess.

As we are re-solving for the equilibrium we can tweak the profiles to eliminate the current discontinuity.

- “In mathematics, a bump function is a function $f : Rn \rightarrow R$ on a Euclidean space Rn which is both smooth (in the sense of having continuous derivatives of all orders) and compactly supported.” (from Wikipedia)
- This functionality is already implemented in fluxgrid but it does not work with the current mapping as either force balance is not enforced or the current is not the curl of the magnetic field.

Some practical details

- We will use the FLUXGRID and NIMEQ infrastructure but create a new executable with independent control flow logic.
  - The existing FLUXGRID and NIMEQ functionality would not be affected.
- The initial grid and field mapping would work as is, but subsequent mappings after NIMEQ solves would need additional development.
  - Field evaluations need to be performed with the NIMROD basis.
- Given the large resolution needed for some cases, it may be worthwhile to implement a parallel design from the start.

Summary

- We intend to integrate NIMEQ and FLUXGRID to improve the NIMROD equilibrium fields on reconstructed cases.
- Our goal is to improve the ability of NIMROD to run well independent of the quality of an equilibrium reconstruction.

- Comments, questions or suggestions?