Motivation

Phenomena with a wide range of scales are a grand challenge for theory, simulations and experiments. Simulations of such phenomena is a subject of study in diverse fields of science today. Scaling them to large node counts presents considerable challenge; however it is often essential for achieving required resolution, accuracy, complexity. Examples include:

- Cosmology/astrophysics: cosmological structure formation simulations.
- Chemistry: all-atom molecular dynamics simulations of enzymes.
- Engineering: compressible turbulence, high-speed mixing and MHD turbulence.
-\text{Challenge: resolving enough details at a high Reynolds number}

Need for a highly scalable versatile numerical library for multiscale phenomena simulations

Commonly used algorithms/frameworks (in 3D):
- Pseudospectral
- FFT
- Chebyshev
- Compact schemes
- Halo exchange etc.

Limitations of existing libraries include (in various combinations):
- Implement only some of the algorithms (for example complex-to-complex or real-to-complex FFT)
- Limited scalability (1D decomposition, scaling limited to linear grid size)
- Fixed data structure/layout (inflexible API)
- Relying on blocking MPI calls, in particular all-to-all exchange (inefficiently inadequate in the age of PetaExascale computing)

Starting Point - Prototype Software: P3DFFT, Generation 2

- Implements real-to-complex 3D FFT, Chebyshev (on one dimension, combined with 2D FFT)
- Employs 2D domain decomposition, achieves good scaling up to O(10^5) cores
- Built on top of FFTW/ESL optimized libraries for 1D FFT
- Provides interfaces for Fortran and C
- 

Limitations:
- Fixed data layout
- Only several transform types
- Relies on blocking MPI calls

Recent Work:
- Hybrid MPI/OpenMP implementation
- Multivariable version (up to x2 speedup observed)
- Investigation of nonblocking communication approaches

Progress to date and ongoing work

Our goal is to have a smooth transition from Generation 2 to Generation 3 framework. Therefore we have continued to support, update, and evolve Generation 2 (see left column) while at the same time working on developing Generation 3 framework (roughly 70% complete). In addition, a new post-processing framework for the turbulence simulation codes is being developed.

Sustainability and Portability

Existing (future) package is (will be) available as open source through Git, a commonly used interface, including a wiki and an issue tracker. Project website http://www.p3dfft.net is maintained, including documentation and mailing list. Code development emphasizes portability by employing modular components with underlying choice of mechanisms. We welcome participation of the growing user community in further code development and testing as one long-term sustainability strategy.

User engagement

In 2014-2015 P3DFFT has been cited over 60 times. Through the regularly updated project wiki and the user mailing list the users keep up-to-date with the developments. Users can contribute to project by suggesting or directly implementing new transform types or library features, as well as providing scaling and performance feedback, using self-testing and timing utilities provided within the package. Talks at HPC and domain science conferences (such as APS Fluids) and informal contacts with colleagues are also used as a means of outreach. New features will be explored and tested with the help of 5 collaborating groups we have identified, across several scientific fields.

Target Solution: P3DFFT Generation 3

Versatile library under development: performance oriented, scalable to high core counts, flexible interfaces
- Compact finite-difference schemes as well as FFT/Chebyshev in 1D, 2D, 3D; other transforms?
- \text{Flexible domain decomposition options}
- \text{User-friendly utilities: differentiation, halo exchanges, spectra, correlations}
- \text{Isolated transposes}

Performance and Scalability

The library under development will provide overlap of communication with computation. A number of mechanisms are being explored for nonblocking all-to-all exchanges, including: MPI-3 (MPI, alltoall), MPI-2 (MPI_Put or MPI_Get), OpenSHMEM, Cahay's Partition. Communication will be implemented in a modular fashion, giving the user freedom to choose the most efficient mechanism depending on their platform of choice. In addition MPI/OpenMP framework will be implemented.

References