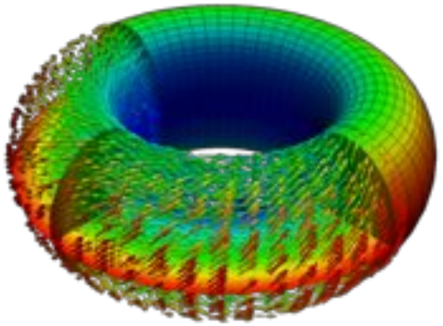
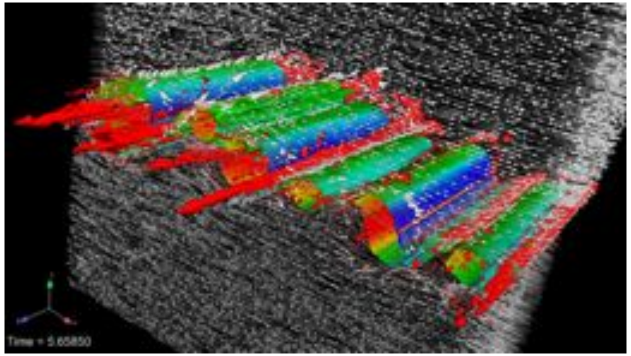
Building Drekar

Building a large-scale CFD / electromagnetics code

Drekar: The Physics

- Solves PDEs for fluids
  - Can be coupled to electromagnetics
- Low-mach compressible fluids
- Incompressible fluids
- Compressible / incompressible magnetohydrodynamics
- Multi-species plasmas interacting with magnetic fields
- Complex geometries





Drekar: The Numerics

- Massively parallel finite element code
    - Has been run on 1.3 million cores
- Written in C++
- Fully-implicit / semi-implicit algorithms
- Has automatic differentiation tools
- Built on Trilinos
    - More on this later

- Continuous finite elements
- Discontinuous finite elements
- Stabilized finite elements
- Mixed-integration finite element bases
    - Nodal, edge, face, volume
- Arbitrary Lagrangian-Eulerian capability

Trilinos

- https://trilinos.github.io/, https://github.com/trilinos/Trilinos
- A collection of packages providing solvers for scientific problems
  - Linear solvers, both direct (Amesos) and iterative(Aztec00, Belos)
  - Preconditioners including algebraic (Ifpack), multilevel (ML), block (Meros)
  - Eigenvalue solvers (Anasazi)
  - Nonlinear solvers (NOX)
  - Optimization algorithms (MOOCHO)
  - Time-integrators (Rythmos, Tempus)
  - Automatic differentiation (Sacado)
  - ...

> “ *Each Trilinos package is a self-contained, independent piece of software with its own set of requirements, its own development team and group of users. Because of this, Trilinos itself is designed to respect the autonomy of packages* ”

Building Drekar

- You will work with my fork of Drekar: https://github.com/dsondak/DrekarBase/
- Instructions on building Drekar for this class:
  - On the wiki: https://github.com/dsondak/DrekarBase/wiki
  - Getting started Part 1: https://github.com/dsondak/DrekarBase/wiki/Instructions-(Part-I):-Getting-Started
- Drekar uses CMake for the build process
- Put CMake commands in `configure-drekar.sh`
  - Specific details can be found in the Github repo and wiki
- Why Cmake?
  - Drekar has *a lot* of dependencies
  - Would like a robust and efficient way to "build" the code
  - Automatic build tools make automatic generation of Makefiles and executable code feasible

Brief Detour into Build Systems and Tools

- A simple build system is just a `Makefile`
- Sometimes, projects are very large
  - e.g. many libraries and executables
- It would be nice to automatically generate the project’s executables and libraries
- Many software implementations can do this:
  - Cmake
  - Autotools (including Autoconf and Automake)
  - configure
  - Others: Build Automation Software

Cmake

- https://cmake.org/
  - Why Cmake?
  - About Cmake
- Manages the build process
- Operating system-independent
- Compiler-independent
- Extensible
- Open source

The Basics of Cmake

- Put config files in each source directory
    - These are called `CmakeLists.txt`
- These config files are used to generate build files
    - e.g. Makefiles
- Supports *in-place* and *out-of-place* builds
    - In-place build: Compile code in the source tree
    - Out-of-place build: Compile code outside of source tree
        - Can have multiple builds without polluting source tree
    - Out-of-place builds are preferred
- Additional resources:
    - cmake documentation
    - cmake examples

Building Drekar

- You will work with my fork of Drekar: https://github.com/dsondak/DrekarBase/
- Instructions on building Drekar for this class:
  - On the wiki: https://github.com/dsondak/DrekarBase/wiki
  - Getting started Part 1: https://github.com/dsondak/DrekarBase/wiki/Instructions-(Part-I):-Getting-Started
- Drekar uses CMake for the build process
- Put CMake commands in `configure-drekar.sh`
  - Specific details can be found in the Github repo and wiki
- Why Cmake?
  - Drekar has *a lot* of dependencies
  - Would like a robust and efficient way to "build" the code
  - Automatic build tools make automatic generation of Makefiles and executable code feasible

Goals for Today

- Follow Drekar Instructions Part 1
- Minimum requirements:
  - Complete *through* Step 4: Configure
  - Start Step 5: Compile
- You will not finish Step 5 in class today

Before Thursday

- You ***must*** complete Instructions Part 1: Getting Started ***before*** lecture on Thursday!
- You will start running Drekar on Thursday and cannot proceed until Part 1 is complete