Introduction to open source CFD

An experimental OpenFOAM course

Gerhard Holzinger

November 23, 2016

Table of contents

What is OpenFOAM?

H. Jasak [5] says:

> *[...] OpenFOAM, an Open Source object-oriented library for numerical simulations in continuum mechanics written in the C++ programming language.*

What is OpenFOAM?

OpenFOAM, an **Open Source** object-oriented library for numerical simulations in continuum mechanics written in the C++ programming language.

- Licenced under the GPL
- Basic freedoms granted:
  - freedom of charge
  - freedom to use
  - freedom to study and modify
  - freedom to redistribute

What is OpenFOAM?

OpenFOAM, an Open Source **object-oriented** library for numerical simulations in continuum mechanics written in the **C++ programming language**.

- No reason for choosing C++ given in the 1998 paper of Weller and Jasak
- Heavy use of C++'s language features throughout OpenFOAM's code base
  - operator overloading
  - inheritance
  - templates

What is OpenFOAM?

OpenFOAM, an Open Source object-oriented library for **numerical simulations in continuum mechanics** written in the C++ programming language.

- Implements the finite volume method (FVM)
  - not limited to CFD
- Offers a wide choice
  - discretisation schemes
  - linear solvers

What is OpenFOAM?

OpenFOAM, an Open Source object-oriented **library** for numerical simulations in continuum mechanics written in the C++ programming language.

- OpenFOAM is not *a* software tool
  - a large number of solvers and tools
- OpenFOAM literally consists of libraries
  - Shared concepts are outsourced into shared libaries
  - E.g. turbulence modelling library

What will we learn?

- Basics on OpenFOAM
- Run OpenFOAM cases
- Modify OpenFOAM solvers

What will we need?

- Linux
- The command line a.k.a. the Terminal
- Basic programming
- A little bit C++
  - depending on whether you want to just use, or modify OpenFOAM

Where do we find help?

- The User Guide[1]
- The internets
  - `http://www.cfd-online.com/Forums/openfoam/`
  - `https://openfoamwiki.net`
  - `http://www.tfd.chalmers.se/~hani/kurser/OS_CFD/`
  - `https://wiki.openfoam.com/Main_Page`
  - and tons of more
- Books
  - *OpenFOAM Primer*, T. Marić et al. [3]
  - *The finite volume method in computational fluid dynamics*, F. Moukalled et al. [1]
  - My manual, at `https://github.com/ParticulateFlow/OSCCAR-doc`

[1] `http://www.openfoam.org/docs/user/`

History of OpenFOAM

- Initial work at Imperial College, London
  - Release of *A tensorial approach to computational continuum mechanics using object-oriented techniques.* paper by H. Weller and H. Jasak in 1998
- Formation of Nabla Ltd. to market FOAM
- Split-up of developer community; open source release of OpenFOAM-1.0
  - Formation of OpenCFD Ltd. with H. Weller
  - Formation of Wikki Lid. with H. Jasak
- OpenCFD is aquired by SGI and later by ESI
- Split-up of OpenCFD team, formation of the OpenFOAM Foundation in 2015

Flavours of OpenFOAM

- FOAM extend
  - Maintainer: H. Jasak
  - `http://foam-extend.org/`
- Foundation release
  - Maintainer: H. Weller
  - `www.openfoam.org`
- ESI release
  - Maintainer: ESI-OpenCFD
  - `www.openfoam.com`
- Other forks and variants
  - `http://openfoamwiki.net/index.php/Forks_and_Variants`

Installation

- Differs from flavour to flavour
- Compile from sources
  - All flavours
  - On Linux, Windows and Mac
- Binary package
  - For major Linux distributions
  - Foundation release: *.deb for Ubuntu, Debian
  - FOAM extend: *.deb and *.rpm (Fedora, etc.)
- Docker image
  - ESI-OpenCFD: For Linux, Windows and Mac
- No installation at all
  - Foundation release: run on AWS EC2
    `http://cfd.direct/cloud`

Table of contents

What is Linux

- Open source operating system
  - distributed under the GPL since 1992 (MS Windows 3.1)
- UNIX-like, in fact heavily influenced by UNIX
- mostly POSIX (portable operating system interface) compliant
  - a standard written by IEEE to ensure portability of programs between different systems

What do people mean by *Linux*

- The Linux kernel
- The whole operating system (OS) (kernel, drivers, etc.)
- A whole Linux distribution (OS + graphical user interface)
  - e.g. Ubuntu, Suse, etc.

Selected features of Linux

- Free of charge, as it is open source
  - you can run as many machines as you can afford it in terms of hardware – good for building large computing clusters
- Separation of operating system and user interface (UI)
  - Graphical user interface (GUI) and textual user interface (TUI) live peacefully side by side
  - Choose the (G)UI that fits your needs
    pure command line, Gnome, KDE, Unity, etc.
- Comes with tons of useful stuff & tools for free
  - as it is developed by a worldwide community, there are plenty of development tools on-board
- Plenty of variety to choose from
  - There are lots of different Linux distributions available
  - There are lots of different tools available, e.g. text editors
- Package management, no need to roam the internets for useful tools – just use the package manager
  - automatic dependency resolution

Why should we care?

- OpenFOAM is mainly developed on/for Linux
  - It should compile on most Linuxes and UNIXes
  - There are also ports for OS X and Windows
- OpenFOAM follows in a large part the design decisions made by Linux and UNIX – more on that later
- OpenFOAM has no GUI
  - It is like the Linux kernel in a Linux distribution
  - You can add a GUI if you like, but you don't have to
- Linux is the real world[2]
  - Your local computing cluster most probably runs Linux
  - 497 of the top 500 super computers run some sort of Linux
  - The other 3 run a UNIX OS

[2] https://www.top500.org/statistics/

What about the UNIX heritage

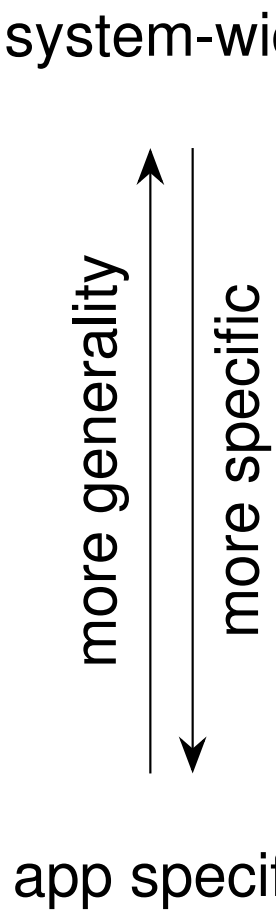
- Helps us understanding OpenFOAM (and Linux) a bit better
- See some UNIX philosophy[3] in action:
  - Create one tool for one job, don't build big monolithic monsters
  - Design for visibility to make inspection and debugging easier
- Is controlled by text files
  - Human readable ASCII files are good for setup and debugging
  - Good for scripting / automatisation

[3] See also here http://www.catb.org/esr/writings/taoup/html/

UNIX heritage

- Small is beautiful
  - There are a lot of apps and tools for very specific tasks
- Silence & repair
  - When there is nothing wrong, output to Terminal is reduced to the minimum
  - When there is something wrong, OpenFOAM fails very noisily
- Representation
  - Knowledge and all the smart stuff is woven into the data structures, the logic is rather simple

UNIX Command & control



Run-control file under `/etc`
- e.g. `/etc/fstab`

System wide environment variables
- e.g. `$PATH`

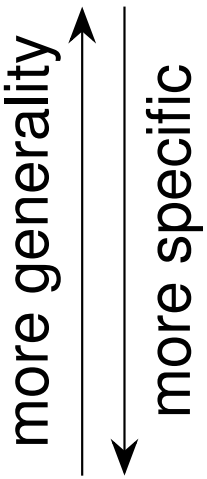
Run-control files in users home directory
- e.g. `~/.bashrc`

User set environment variables
- e.g. `$FOAM_SRC`

Switches and arguments passed to the app
- e.g. `ls -al` or `cd ~/OpenFOAM`

OpenFOAM Command & control

installation-wide



app specific

Run-control file under `OpenFOAM-x.x.x/etc`

e.g. `OpenFOAM-x.x.x/etc/bashrc`

System wide environment variables

e.g. `$FOAM_SRC`

App-specific run-control files in case directory

e.g. `blockMeshDict`

Switches and arguments passed to the app

e.g. `checkMesh -allGeometry`

UNIX heritage take away

- The equivalent of UNIX's run-control files are app-specific dictionaries
  - either system (installation) wide or app-specific

controlDict controls the solver

blockMeshDict provides input data for blockMesh

setFieldsDict provides input data for setFields

- OpenFOAM uses shell variables to define certain thing only once
  - Avoid information duplication
  - Good for maintenance
- Small is good
  - Compressible and incompressible flows are handled by different solvers
  - More than one meshing/pre-processing/etc. tool

A little C++ perspective

- OpenFOAM consists mostly of C++
  - According to github (OpenFOAM-dev)
  - C++ 52.1% C 46.6% Shell 1.2%
- Makes tremendous use of C++ language features
  - Object orientation
  - Inheritance
  - Templates
  - Operator overloading
- Other (programming related) cruelty/features
  - Pre-processor macros
  - Does not use C++'s standard template library (STL) e.g. for containers or smart pointers
    – OpenFOAM's initial developments predate the widespread use of the STL

A little C++ perspective

- Why should we care?
  - Unfortunately, there is little official documentation
  - The documentation is in the code
- Makes tremendous use of C++ language features
  - Object orientation
  - Inheritance
  - Templates
  - Operator overloading
- Other (programming related) cruelty/features
  - Pre-processor macros
  - Does not use C++'s standard template library (STL) e.g. for containers or smart pointers
    – OpenFOAM's initial developments predate the widespread use of the STL

What is CFD

- Computational Fluid Dynamics
  - There are several methods (FDM, FVM, FEM)
  - In most cases people mean FVM
- Problem:
  - Computers are good at solving linear equation systems
    $$\mathbf{A}\mathbf{x} = \mathbf{b}$$
  - However, we want to solve Navier-Stokes equations
    $$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$
    $$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \nabla \cdot \boldsymbol{\tau} - \nabla p + \rho \mathbf{g}$$
- Use a computational method to translate governing equations into a linear equation system

OpenFOAM speaks CFD

- The C++ framework of OpenFOAM is cleverly designed
  - To minimise code duplication and provide generality
  - To mimic the high-level mathematical language we all prefer
- Problem:
  - How to solve our governing equations?

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

OpenFOAM speaks CFD

- The C++ framework of OpenFOAM is cleverly designed
  - To minimise code duplication and provide generality
  - To mimic the high-level mathematical language we all prefer
- Problem:
  - How to solve our governing equations?

    $$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$
- Solution:
  - Use OpenFOAM

    ```
    solve( fvm::ddt(rho) + fvm::div(phi, rho) )
    ```
- The finite volume method *hides* behind the data structures – *Smart data structures and dumb code works a lot better than the other way around* [2].

OpenFOAM speaks CFD

- All we do is write high-level code
  - `solve( fvm::ddt(rho) + fvm::div(phi, rho) )`
- OpenFOAM takes care of the rest
  - Translating
    $$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$
    to the appropriate discretized, linear equation system
    $$\mathbf{A}\mathbf{x} = \mathbf{b}$$
    and solving it

Table of contents

What is OpenFOAM

- Solvers and models for (mainly) CFD
  - All sorts of single- and multiphase flows, Lagrangian methods for particles, etc.
  - Electrostatic and magnetic fields
  - Heat transfer
  - Solid mechanics
- Utilities for pre- and post-processing
  - Mesh creation
  - Case setup
  - Data aquisition and post-processing
- A C++ framework to write your own solvers/models/utilities

OpenFOAM case

- (Text)-file based
  - ASCII: for humans to read
  - Binary: to save disk-space
    is only available for time step data and the mesh
- Needs definitions for:
  - Initial and boundary conditions
  - Computational domain and material properties
  - Simulation and solver control

Initial and boundary conditions

- One file per field
- Initial conditions:
  - A list of per-cell values
  - A uniform value
- Boundary conditions:
  - A dictionary containing entries for all patches
  - You may also use regular expressions
    e.g. "`wall.*`" for `wallRight`, `wallLeft`, etc.

Computational domain and material properties

- Divided into a bunch of files
- The mesh, individual files for:
  - Points
  - Faces – defined by a list of points
  - Cells – defined by a list of faces
  - etc.
- Fluid properties and other stuff
  - Material constants (density, viscosity, etc.)
  - Model parameters (the turbulence model to use, etc.)

Simulation and solver control

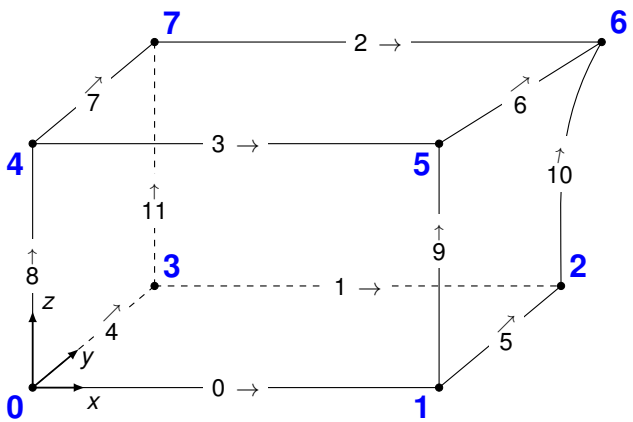
- General controls for application
  - Time step
  - When to write data
- Solver control
  - Tolerances
  - Solution control
- Controls for computational method
  - Discretisation schemes

Additional contents of a case

- New, computed time step data
  - Directory name is (roughly) the time step
- Data gathered during the simulation
  - Located in the postProcessing directory
- Data generated by post-processing tools
  - Located in the respective time step folders
- Sets of points, faces or cells

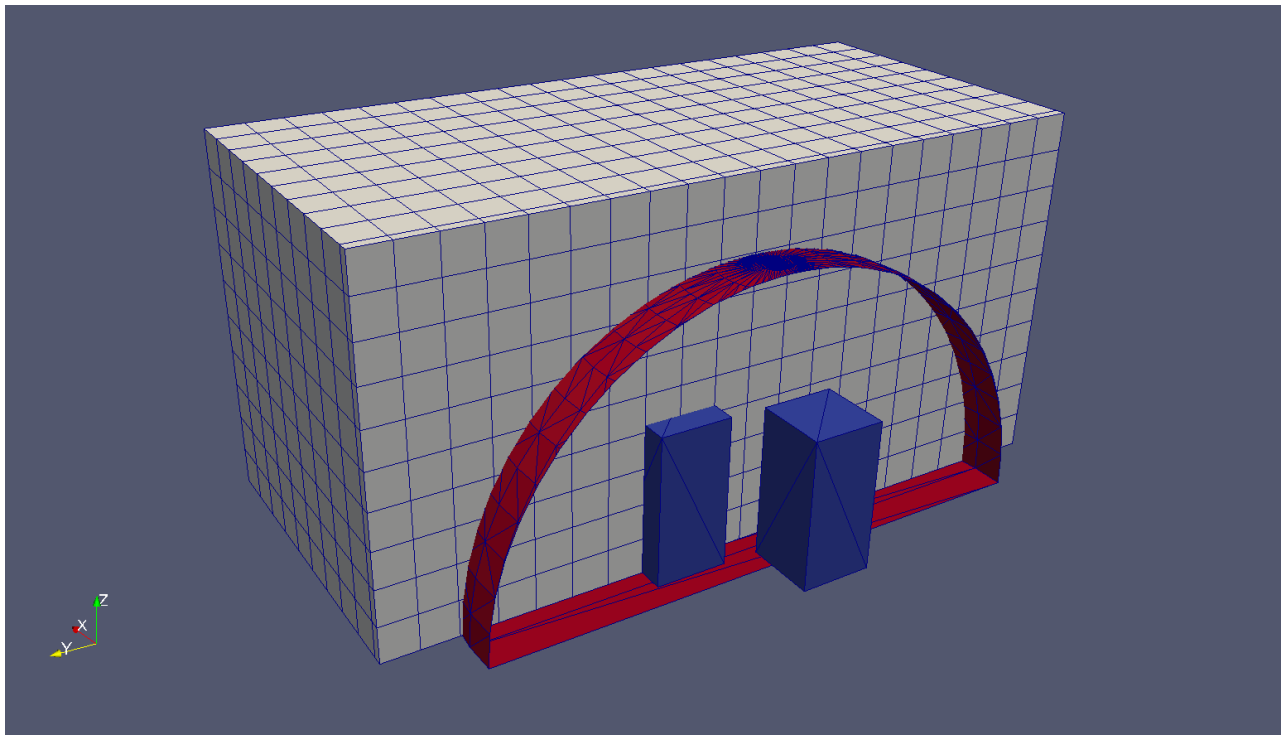
Mesh creation

- blockMesh
  - Simple meshing tool based on a single block
  - Geometry is built with blocks
  - Can be scripted



Mesh creation

- snappyHexMesh
  - Advanced meshing tool for complicated geometries
  - Geometry defined by STL file

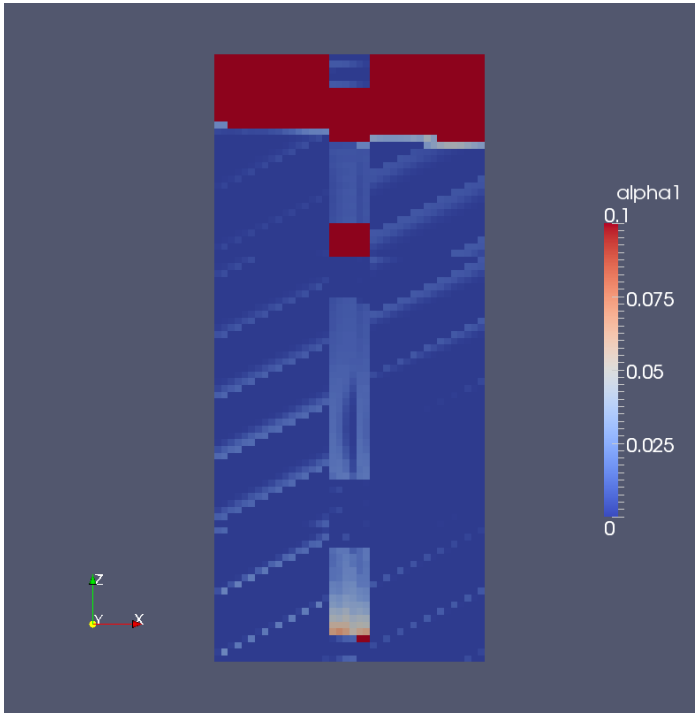


Mesh creation

- Other mesh generation tools of OpenFOAM
  - foamyMesh
  - extrudeMesh
  - converters (e.g. Fluent to OpenFOAM)
- Third-party mesh generation
  - For some an additional conversion step is needed
  - cfMesh
  - GMSH
  - Salome
  - many more

Pre-processing utilities

- Mesh manipulation
  - Define sets of cells, faces or points
  - Refine mesh, remove cells, etc.
- Manipulating fields
  - For non-trivial initial values
  - By geometric regions
  - Mapping from other case



Post-processing

- Data aquisition
  - The solution of our case is tons of per-cell values
  - We need to extract meaningful data
- Visualisation
  - Simple plotting (MATLAB, octave, gnuplot, etc.)
  - Nice images of Colourful Fluid Dynamics (ParaView)

Data aquisition

- Done by functionObjects (at run-time or afterwards) or some utilities (afterwards)
- Writes ASCII files to disk
- Provides data
  - To be interpreted directly
  - Feeding simple plotting scripts
- Operates on data
  - Simple extraction and writing to disk
  - Computing stuff (temporal averages, vorticity, etc.)

Visualisation with ParaView

- Directly reads the solution fields
  - By the power of OpenFOAM's reader plugin
  - ParaView[4] is independent of OpenFOAM
- Operates on data
  - Various modes of visualisation (plain field, streamlines, etc.)
  - Computing stuff
- Create images and animations

[4] http://www.paraview.org

Further Reading I


F. Moukalled, L. Mangani, and M. Darwish.
The finite volume method in computational fluid dynamics.
Springer, 2016.


Eric S. Raymond.
The Art of UNIX Programming.
Addison-Wesley, 2003.


T. Marić, J. Höpken, and K. Mooney.
The OpenFOAM Technology Primer.
Sourceflux, 2014.


H. G. Weller, G. Tabor, H. Jasak, and C. Fureby.
A tensorial approach to computational continuum mechanics using object-oriented techniques.
Computers in Physics, 12:620–631, 1998.

Further Reading II

H. Jasak, A. Jemcov, and Z. Tukovic.
OpenFOAM: A C++ Library for Complex Physics Simulations.
International Workshop on Coupled Methods in Numerical Dynamics, Dubrovnik, Croatia, 2007.