OpenFOAM structure: Linux environment, user environment variables, directory and file structure, units, etc.

Eric Paterson
egp@vt.edu

Kevin T. Crofton Department of Aerospace and Ocean Engineering
Virginia Polytechnic Institute and State University

ESOP Workshop
5-8 June 2017

Outline

1. Linux environment
2. Installation location
3. User directory organization
4. How do I use OpenFOAM?
5. Environment variables
6. Environment variables
7. OpenFOAM "settngs"
8. OpenFOAM aliases
9. Dimensional units

VirginiaTech
Invent the Future®

Linux environment

- Linux is the preferred operating system for OpenFOAM, and parallel HPC and large–scale scientific computing in general.
- Although
  - OpenFOAM can be built on Mac, since OSX is essentially a flavor of UNIX.
  - OpenFOAM on Windows is trickier, but possible through a number of VMs and dual-boot systems.
    - We are using a Docker container with a Linux VM and running OpenFOAM in that environment.
    - There are other similar options available from other sources, e.g., : `https://cfd.direct/openfoam/download/`
- Interesting recent development is cloud computing services, such as Amazon Web Services (AWS) EC2, `https://cfd.direct/cloud/aws/`

VirginiaTech
Invent the Future®

OpenFOAM Distributions

- There are a number of OpenFOAM forks. The latest versions and sources include:
  - OpenFOAM v4.1 from CFD Direct `https://cfd.direct`
  - OpenFOAM v1612+ from ESI, owners of the OpenFOAM trademark `http://www.openfoam.com`
  - OpenFOAM-extend 3.2 from a loose group of academics and industry CFD experts, `https://sourceforge.net/projects/openfoam-extend/`
  - Caelus from Applied CCM `http://www.caelus-cml.com`
  - ... plus a few more smaller players
- We will use v1612+ from ESI since it provides a Docker containerized VM for Windows users.

VirginiaTech
Invent the Future®

OpenFOAM Installation

- Installation of OpenFOAM from source is beyond the scope of this workshop.
- For most users, they will either download binaries, or have their local HPC administrators build it into a `module` that can be loaded at login. Note: OpenFOAM is on all DOD HPCMP machines.
- Advantage of building from source, is that compiler options and parallel MPI libraries can be optimized for specific hardware.
- On Virginia Tech computer CASCADES, v1612+ is loaded with the following commands:

```
module load gcc/5.2.0 openmpi/2.0.0 python OpenFOAM/v1612+
. $OPENFOAM_DIR/OpenFOAM-v1612+/etc/bashrc
```

- These are typically put in your `$HOME/.bashrc` so that OpenFOAM is loaded when you login.

VirginiaTech
Invent the Future®

Installation location

- OpenFOAM is installed at `$WM_PROJECT_INST_DIR`
- That environment variable points to:
  Cascades:
  `/groups/arc/apps/cascades/opt/apps/gcc5_2/openmpi2_0/OpenFOAM/v1612+`
  Docker VM: `/opt/OpenFOAM`
- Users do not have write privileges to this tree. Users can run compiled applications/utilities and link to compiled libraries.
- Custom code (e.g., ESOP) has to be compiled somewhere in the users `$HOME` directory structure.
- Users cannot run the tutorials in the installation directory tree since they don't have write permissions. This is the reason that we copied the tutorials to `$FOAM_RUN`, which points to a directory in the user space:
  Cascades: `/home/egp/OpenFOAM/egp-v1612+/run`
  Docker VM: `/Users/egp/OpenFOAM/docker-v1612+/run`

VirginiaTech
Invent the Future®

User directory organization

- The users OpenFOAM space often includes the following directories:

```
[17:55:32] [egp@calogin1:egp-v1612+] 50046$ ls
applications  libraries  packages  platforms  run
```

- Where these directories are defined as:
  - `applications`: location for custom application source code
  - `libraries`: location for custom library source code
  - `packages`: location for larger software suites, like ESOP and swak4Foam.
  - `platforms`: location of user binaries. `$FOAM_USER_APPBIN` and `$FOAM_USER_LIBBIN` point to this directory.
  - `run`: A place for users to organize their personal cases. `$FOAM_RUN` points to this folder.

How do I use OpenFOAM?

- First step is to get your environment set–up so that paths and variables are properly set.
- This is accomplished by sourcing the `bashrc` file in the OpenFOAM source code with the the following command (for Bash shell uses who are installing OpenFOAM in their home directory):
  `% source $HOME/OpenFOAM/OpenFOAM-v1612+/etc/bashrc`
- For users on large systems with central installations, loading a module is now the typical approach. The following commands would be in the users `$HOME/.bashrc` file.
  ```
  module load gcc/5.2.0 openmpi/2.0.0 python OpenFOAM/v1612+
  . $OPENFOAM_DIR/OpenFOAM-v1612+/etc/bashrc
  ```
- The `module load` command loads gcc, openmpi, python and OpenFOAM. Exact details can be found by using the `module spider OpenFOAM` command.

VirginiaTech
Invent the Future®

Environment variables

- If everything goes smoothly, you should now have a lot of new environement variables and aliases.

```
[18:56:53][egp@calogin1:esop]50070$ echo $WM_
$WM_ARCH              $WM_COMPILER          $WM_DIR               $WM_MPLIB             $WM_PROJECT_DIR
$WM_ARCH_OPTION       $WM_COMPILER_LIB_ARCH $WM_LABEL_OPTION      $WM_OPTIONS           $WM_PROJECT_INST_DIR
$WM_CC                $WM_COMPILER_TYPE     $WM_LABEL_SIZE        $WM_OSTYPE            $WM_PROJECT_USER_DIR
$WM_CFLAGS            $WM_CXX               $WM_LDFLAGS           $WM_PRECISION_OPTION  $WM_PROJECT_VERSION
$WM_COMPILE_OPTION    $WM_CXXFLAGS          $WM_LINK_LANGUAGE     $WM_PROJECT           $WM_THIRD_PARTY_DIR
[18:56:53][egp@calogin1:esop]50070$ echo $FOAM_
$FOAM_APP             $FOAM_INST_DIR     $FOAM_RUN          $FOAM_SITE_LIBBIN  $FOAM_USER_APPBIN
$FOAM_APPBIN          $FOAM_JOB_DIR      $FOAM_SETTINGS     $FOAM_SOLVERS      $FOAM_USER_LIBBIN
$FOAM_ETC             $FOAM_LIBBIN       $FOAM_SIGFPE       $FOAM_SRC          $FOAM_UTILITIES
$FOAM_EXT_LIBBIN      $FOAM_MPI          $FOAM_SITE_APPBIN  $FOAM_TUTORIALS
```

- Aliases can be discovered using the following command `% alias`

VirginiaTech
Invent the Future®

Environment variables

- If everything goes smoothly, you should now have a lot of new environment variables.
- These are set in this file:

```
$WM_PROJECT_INST_DIR/OpenFOAM-v1612+/etc/config.sh/settings
```

```
[18:56:53][egp@calogin1:esop]50070$ echo $WM_
$WM_ARCH              $WM_COMPILER           $WM_DIR                $WM_MPLIB              $WM_PROJECT_DIR
$WM_ARCH_OPTION       $WM_COMPILER_LIB_ARCH  $WM_LABEL_OPTION       $WM_OPTIONS            $WM_PROJECT_INST_DIR
$WM_CC                $WM_COMPILER_TYPE      $WM_LABEL_SIZE         $WM_OSTYPE             $WM_PROJECT_USER_DIR
$WM_CFLAGS            $WM_CXX                $WM_LDFLAGS            $WM_PRECISION_OPTION   $WM_PROJECT_VERSION
$WM_COMPILE_OPTION    $WM_CXXFLAGS           $WM_LINK_LANGUAGE      $WM_PROJECT            $WM_THIRD_PARTY_DIR
[18:56:53][egp@calogin1:esop]50070$ echo $FOAM_
$FOAM_APP             $FOAM_INST_DIR     $FOAM_RUN           $FOAM_SITE_LIBBIN   $FOAM_USER_APPBIN
$FOAM_APPBIN          $FOAM_JOB_DIR      $FOAM_SETTINGS      $FOAM_SOLVERS       $FOAM_USER_LIBBIN
$FOAM_ETC             $FOAM_LIBBIN       $FOAM_SIGFPE        $FOAM_SRC           $FOAM_UTILITIES
$FOAM_EXT_LIBBIN      $FOAM_MPI          $FOAM_SITE_APPBIN   $FOAM_TUTORIALS
```

OpenFOAM "settings"

- In addition to environement variables, your path will have additions which will allow your system to be aware of the location of application, utility, and library binaries.
- The important path additions include:
  - `$FOAM_APPBIN` - this points to the OpenFOAM application binaries
  - `$FOAM_LIBBIN` - this points to the OpenFOAM library binaries
  - `$FOAM_USER_APPBIN` - this points to the user's application binaries
  - `$FOAM_USER_LIBBIN` - this points to the user's library binaries

OpenFOAM aliases

- Aliases can be discovered using the following command `% alias`
- Aliases are set in this file:
  `$WM_PROJECT_INST_DIR/OpenFOAM-v1612+/etc/config.sh/aliases`
  - alias app='cd $FOAM_APP'
  - alias foam='cd $WM_PROJECT_DIR'
  - alias foamSite='cd $WM_PROJECT_INST_DIR/site'
  - alias lib='cd $FOAM_LIBBIN'
  - alias run='cd $FOAM_RUN'
  - alias sol='cd $FOAM_SOLVERS'
  - alias src="cd $FOAM_SRC'
  - alias tut='cd $FOAM_TUTORIALS'
  - alias util='cd $FOAM_UTILITIES'

VirginiaTech
Invent the Future®

Dimensional units [1]

- In general, algebraic operations require consistent dimensional units
- Specifically, addition, subtraction, and equality are only physically meaningful if each of the terms of a governing equation have the same dimensional units
- As a safeguard, OpenFOAM attaches dimensions to field data and physical properties and performs dimension checking on any scalar, vector, or tensor operation
- Dimensions are set for each field variable or property using a set of 7 scalars delimited by square brackets. For example:
  - Velocity `(0/U)`: `[0 1 -1 0 0 0 0]`
  - Kinematic viscosity `(/constant/transportProperties)`: `[0 2 -1 0 0 0 0]`
- Each scalar value within the brackets corresponds to the power of each of the base units shown in following table

Dimensional units [2]

```
[0 2 -1 0 0 0 0]
```

No.	Property	SI unit	USCS unit
1	Mass	kilogram (kg)	pound-mass (lbm)
2	Length	metre (m)	foot (ft)
3	Time	second (s)	
4	Temperature	Kelvin (K)	degree Rankine (°R)
5	Quantity	kilogram-mole (kgmol)	pound-mole (lbmol)
6	Current	ampere (A)	
7	Luminous intensity	candela (cd)	

If your units are inconsistent, either in an input file, dictionary, or the source code for an application/utility, OpenFOAM will fail with an "incompatible dimensions for operation" error message

Dimensional units [3]

- Note that OpenFOAM does require some dimensioned physical constants for certain calculations (e.g., Universal Gas Constant R, Standard Pressure and Temperature)
- If you want to check, dimensioned constants are specified in `$WM_PROJECT_INST_DIR/OpenFOAM-v1612+/src/OpenFOAM/global/constants`
- Finally, note that:
  - OpenFOAM dimension checking can be turned off in the main controlDict and is not recommended
  - It is possible to perform dimensionless calculations by appropriately constructing the governing equations and correctly setting field variable / property dimensions. However, this can be very tricky and is not recommended.
  - Dimension checking is powerful quality-control mechanism, and all simulations should be performed using proper dimensions.

VirginiaTech
Invent the Future®