All that begins . . .

السلام عليكم

peace be upon you

Starting OpenFOAM on Linux

for the uninitiated

Abu Hasan 'ABDULLAH

October 2018

Outline

- This guide provides information and example terminal commands for Linux, relevant to users of OpenFOAM.
- You will be introduced to commands which refer to OpenFOAM and OpenFOAM Linux environment variables.
- Those commands that refer to OpenFOAM will only function as stated, if they are executed on a machine on which OpenFOAM is installed and the user's environment variables are set up for OpenFOAM

- Linux uses environment variables that specify a set of values that affect the way the computer runs.
- The OpenFOAM configuration sets environment variables that begin with
  - FOAM_ to provide short-cuts in the use of OpenFOAM and
  - WM_ to help with compilation of OpenFOAM.

Table 1: env command

Command	What it does
`env`	List all enviroment variables in the shell (terminal)
`env | grep ^FOAM_`	List environment variables beginning `FOAM_`
`echo $FOAM_SRC`	Return the value (denoted by $...) of the `FOAM_SRC` environment varaible

- **OpenFOAM 4**: To source/load its environment variables type

```
$ . /opt/openfoam4/etc/bashrc
```

at the command prompt. Watch the dot at the beginning of line!!

- **OpenFOAM 5**: To source/load its environment variables type

```
$ . /opt/openfoam5/etc/bashrc
```

at the command prompt. Again, watch the dot at the beginning of line!!

Commands to Print to Terminal

- Scroll through the `Allwmake` file in terminal; type <SPACE> to scroll, `Q` to quit:

```
$ less $WM_PROJECT_DIR/Allwmake
```

- Print the first 10 lines of `Allwmake`

```
$ head -10 $WM_PROJECT_DIR/Allwmake
```

- Print the last 10 lines of `Allwmake`

```
$ tail -5 $WM_PROJECT_DIR/Allwmake
```

Expression Matching

- -h: Prints lines of file Allwmake that contain the expression build

```
$ grep -h build $WM_PROJECT_DIR/Allwmake
```

- -i: Prints lines of file Allwmake that contain BuIlD, ignoring upper/lower case

```
$ grep -h -i BuIlD $WM_PROJECT_DIR/Allwmake
```

- -l: Prints the filename Allwmake to terminal if it contains the expression build

```
$ grep -l build $WM_PROJECT_DIR/Allwmake
```

- -H: Prints both filename and lines of a file that contain an expression build

```
$ grep -H build $WM_PROJECT_DIR/Allwmake
```

Finding Files/Directories

- Prints all files, directories and links in the OpenFOAM `src` directory

```
$ find $FOAM_SRC
```

- Prints files and links (or directories) named `fvMesh.H` in `FOAM_SRC`

```
$ find $FOAM_SRC -name fvMesh.H
```

- Prints files only named `fvMesh.H` in `FOAM_SRC`

```
$ find $FOAM_SRC -name fvMesh.H -type f
```

- Prints links only named `fvMesh.H` in `FOAM_SRC`

```
$ find $FOAM_SRC -name fvMesh.H -type l
```

- Prints files only ending `.C` or `.H` in `FOAM_SRC` (`*` means “any characters”)

```
$ find $FOAM_SRC -name "*.[CH]" -type f
```

Searching for an expression in a large number of files

- Combining `find` and `grep` allows us to search for an expression in large number of files.
- E.g. search through all OpenFOAM `.C` source files to find one containing the expression `kepsilon` (case insensitive):

```
$ find $FOAM_SRC -name "*.C" | xargs grep -l -i kepsilon
```

- An alternative syntax, that executes slower is:

```
$ find $FOAM_SRC -name "*.C" -exec grep -l -i kepsilon {} \;
```

sample case from tutorials

- **OpenFOAM 4**: To source/load its environment variables type

```
$ . /opt/openfoam4/etc/bashrc
```

at the command prompt. Watch the dot at the beginning of line!!

- **OpenFOAM 5**: To source/load its environment variables type

```
$ . /opt/openfoam5/etc/bashrc
```

at the command prompt. Again, watch the dot at the beginning of line!!

- Check the directory where your OpenFOAM cases are run from:

```
$ echo $FOAM_RUN
```

If it doesn't exist, create one and list it contents immediately, just to check!

```
$ mkdir -p $FOAM_RUN
$ ls -1 $FOAM_RUN
```

`$FOAM_RUN` is a (parent) general working directory containing working directories of your cases. Change directory there:

```
$ cd $FOAM_RUN
```

- Copy sample `cavity` case subdirectory from the OpenFOAM tutorials

```
$ cp -r    $FOAM_TUTORIALS/incompressible/icoFoam/cavity/cavity .
```

then change into `cavity` working directory

```
$ cd cavity
```

you are now in the `cavity` case working directory. Optionally, you can check to see subdirectories under this case using the command:

```
$ tree
```

- A description on how to discretize computational domain into cells i.e. to mesh the geometry of study is kept in a "dictionary" file `blockMeshDict` which is in the `system` subdirectory of your case working directory. Use `gedit` editor to view or modify it:

```
$ gedit system/blockMeshDict
```

- Once you are done editing `blockMeshDict`, save it and invoke `blockMesh` mesher to process your computing domain

```
$ blockMesh
```

when `blockMesh` is done without error, you can, optionally, check your mesh through

```
$ checkMesh
```

- At this stage you may, if you wish, preview the meshing of your computational domain by calling

```
$ paraFoam &
```

- Our `cavity` case can be described by incompressible laminar Navier-Stokes equations and solved using the PISO algorithm.
- We therefore invoke the appropriate `icoFoam` to solve it

```
$ icoFoam
```

- To post-process results from the solver, invoke

```
$ paraFoam &
```

1. Sourcing OpenFOAM environment variables

```
$ . /opt/openfoam4/etc/bashrc
```

2. Housekeeping & Navigating Working Directories

```
$ echo $FOAM_RUN
$ mkdir -p $FOAM_RUN
$ ls -1 $FOAM_RUN
$ cd $FOAM_RUN
$ cp -r $FOAM_TUTORIALS/incompressible/icoFoam/cavity/cavity .
$ cd cavity
$ tree
```

3. Pre-Processing (PREP)

```
$ gedit system/blockMeshDict
$ blockMesh
$ checkMesh
$ paraFoam &
```

4. Solving (SOLV)

```
$ icoFoam
```

5. Post-Processing (POST)

```
$ paraFoam &
```

simple case from scratch

- **OpenFOAM 4**: To source/load its environment variables type

```
$ . /opt/openfoam4/etc/bashrc
```

at the command prompt. Watch the dot at the beginning of line!!

- **OpenFOAM 5**: To source/load its environment variables type

```
$ . /opt/openfoam5/etc/bashrc
```

at the command prompt. Again, watch the dot at the beginning of line!!

- The first step to start the OpenFOAM simulation is to create the working directories which will contain all the required files:
  - some of them are going to be created by the user to setup the pre-processing,
  - some of them will be automatically generated by the solver, and
  - some of them will be necessary during post-processing.
- Check the directory where your OpenFOAM cases are run from:

```
$ echo $FOAM_RUN
```

- If it doesn't exist, create one and list it contents immediately, just to check!

```
$ mkdir -p $FOAM_RUN
$ ls -1 $FOAM_RUN
```

- To start preparing for your case, change directory there:

```
$ cd $FOAM_RUN
```

- Create the case working directory `ppWall`, where all the files for the Couette flow simulation will be stored,

```
$ mkdir -p ppWall
```

and change directory there

```
$ cd ppWall
```

- Within this case working directory, there must be three main subdirectories: `0` , `constant` and `system`. To create them,

```
$ mkdir -p 0 constant system
```

`0`	controls the initial field data ($t = 0$),
`constant`	contains the physical properties for the case concerned, and
`system`	contains a full description of the case mesh and controls the setting parameters associated with the solution procedure itself.

- Optionally, to see how these three subdirectories are structured under your case working directory, type

```
$ tree
```

- Create and edit the mesh *dictionary* file `system/blockMeshDict` containing a description on how to discretize computational domain into cells i.e. to mesh the geometry of case under study. Use `gedit` editor:

```
$ gedit system/blockMeshDict
```

- Create and edit files containing boundary conditions and initial values for the case in the `0` subdirectory contains two files, `p` representing the pressure (p) field:

```
$ gedit 0/p
```

- ... and `U`, representing the velocity (U) field:

```
$ gedit 0/U
```

- Create and edit files containing physical properties of the case in the `constant` subdirectory containing *dictionary* files, whose names are given the suffix ...`Properties`.

  The `ppWall` case will be solved using `icoFoam` solver, which solves transient, incompressible, laminar flows of Newtonian fluids and it needs these physical properties described in `transportProperties`:

```
$ gedit constant/transportProperties
```


- Create and edit several files located in `system` subdirectory:

1. `controlDict` dictionary file containing input data related to the control of time and reading and writing of the solution data,

```
$ gedit system/controlDict
```

2. `fvSchemes` file which specifies the choice of finite volume discretization schemes (for each one of the terms of the differential equations governing the problem),

```
$ gedit system/fvSchemes
```

3. `fvSolution` file containing the specifications of the linear equation solvers and tolerances and other algorithm controls,

```
$ gedit system/fvSolution
```

- By now you have already created, edited and saved `blockMeshDict`. Invoke `blockMesh` mesher to process your computing domain

```
$ blockMesh
```

- Optionally, you may check your meshing:

```
$ checkMesh
```

and/or preview the meshing of your computational domain:

```
$ paraFoam &
```

- Our `ppWall` case can be solved by `icoFoam` solver, which solves transient, incompressible, laminar flows of Newtonian fluids using the PISO algorithm.

  We therefore invoke `icoFoam` to solve it:

```
$ icoFoam
```

- To post-process results from the solver, invoke

```
$ paraFoam &
```

1. Sourcing OpenFOAM environment variables

```
$ . /opt/openfoam4/etc/bashrc
```

2. Housekeeping & Navigating Working Directories

```
$ echo $FOAM_RUN
$ mkdir -p $FOAM_RUN
$ ls -1 $FOAM_RUN
$ cd $FOAM_RUN
$ mkdir -p ppWall
$ cd ppWall
$ mkdir -p 0 constant system
$ tree
```

3. Pre-Processing (PREP)

```
$ gedit system/blockMeshDict
$ gedit 0/p
$ gedit 0/U
$ gedit constant/transportProperties
$ gedit system/controlDict
$ gedit system/fvSchemes
$ gedit system/fvSolution
```

4. Solving (SOLV)

```
$ icoFoam
```

5. Post-Processing (POST)

```
$ paraFoam &
```

exploring ship resistance case

1. Sourcing OpenFOAM environment variables

```
$ . /opt/openfoam4/etc/bashrc
```

2. Housekeeping & Navigating Working Directories

```
$ echo $FOAM_RUN
$ mkdir -p $FOAM_RUN
$ ls -1 $FOAM_RUN
$ cd $FOAM_RUN
$ cp -r $FOAM_TUTORIALS/multiphase/interFoam/ras/DTCHull .
$ cd DTCHull
$ mv 0.orig 0
$ cd ..
$ tree DTCHull
```

You will notice that there are many more files, Figure **??**, related to this case compared with the previous two cases.

```
DTCHull
├── 0
│   ├── alpha.water
│   ├── k
│   ├── nut
│   ├── omega
│   ├── pointDisplacement
│   ├── p_rgh
│   └── U
├── Allclean
├── Allrun
├── constant
│   ├── g
│   ├── hRef
│   ├── transportProperties
│   ├── triSurface
│   │   └── README
│   └── turbulenceProperties
├── README
└── system
    ├── blockMeshDict
    ├── controlDict
    ├── decomposeParDict
    ├── fvSchemes
    ├── fvSolution
    ├── meshQualityDict
    ├── refineMeshDict
    ├── setFieldsDict
    ├── snappyHexMeshDict
    ├── surfaceFeatureExtractDict
    ├── topoSetDict.1
    ├── topoSetDict.2
    ├── topoSetDict.3
    ├── topoSetDict.4
    ├── topoSetDict.5
    └── topoSetDict.6
```

Figure 1: Subdirectories and files under `DTCHull` case.

3. Pre-Processing (PREP)
You shall use snappyHexMesh which is a mesh generator that takes an already existing mesh (usually created with blockMesh) and refines it into the mesh you want.

```
$ gedit system/blockMeshDict
$ blockMesh
$ checkMesh
$ gedit system/snappyHexMeshDict
$ snappyHexMesh
$ checkMesh
$ paraFoam &
```

Surf

- https://openfoamwiki.net/index.php/SnappyHexMesh
- https://openfoamwiki.net/images/f/f0/Final-AndrewJacksonSlidesOFW7.pdf
- http://www.calum-douglas.com/openfoam-tutorial-snappyhexmesh/
- http://www.training.prace-ri.eu/uploads/tx_pracetmo/snappyHexMesh.pdf

for helps, tips & tricks.

4. Solving (SOLV)
   You will need to use a solver different from previous two exercises; this time you shall use the RAS (Reynolds Averaged Stress) model of `interFoam` solver:

```
$ interFoam
```

Surf

- https://publications.lib.chalmers.se/records/fulltext/146569.pdf
- http://www.personal.psu.edu/dab143/OFW6/Training/maki_slides.pdf
- https://eprints.soton.ac.uk/345877/1/windennutts12.pdf
- https://www.shipjournal.co/index.php/sst/article/view/150/438
- http://opus.bath.ac.uk/38309/1/UnivBath_PhD_2013_G_Morgan.pdf
- www.training.prace-ri.eu/uploads/tx_pracetmo/MarineCFDApplications.pdf

for helps, tips & tricks.

5. Post-Processing (POST)

```
$ paraFoam &
```

Bibliography

1. https://openfoam.org/
2. https://www.openfoam.com/documentation/tutorial-guide/
3. http://the-foam-house5.webnode.es/
4. http://www.foamacademy.com/wp-content/uploads/2016/11/2017-03-29_tutorial_shipresistance.pdf
5. https://openfoamwiki.net/index.php/SnappyHexMesh
6. http://www.tfd.chalmers.se/ hani/kurser/OS_CFD_2007/HassanHemida/Hassan_Hemida_VOF.pdf
7. http://infofich.unl.edu.ar/upload/3be0e16065026527477b4b948c4caa7523c8ea52.pdf
8. http://www.training.prace-ri.eu/uploads/tx_pracetmo/MarineCFDApplications.pdf

- . . . and I end my presentation with two supplications

رَّبِّ زِدْنِي عِلْمًا

my Lord! increase me in knowledge

(TAA-HAA (20):114)

اَللّٰهُمَّ اِنَّا نَسْئَلُكَ عِلْمًا نَافِعًا

O Allah! We ask You for knowledge that is of benefit

(IBN MAJAH)