What is OpenFOAM

OpenFOAM is an open-source collection of numerical solvers and pre-/post-processing utilities, mainly used in Computational Fluid Dynamics. This user guide provides instructions on how to run (i) a standard OpenFOAM tutorial test case, and (ii) more extensive benchmark tests on the HPC Wales systems.

Step 1 - Log in

The example used in this guide is configured to run on the Swansea Sandy Bridge cluster. Connect to *login.hpcwales.co.uk* with your HPC Wales user credentials using your preferred method (e.g. PuTTY from a Windows machine or ssh from any Linux terminal), then **`ssh sw-sb-log-001`** to connect to the Swansea system.

The steps below involve typing commands (**`in bold font`**) in the terminal window.

Step 2 - Load an OpenFOAM module

A number of OpenFOAM binary packages are available.

- List preinstalled OpenFOAM versions:
  `module avail openfoam`
- Load your preferred version (2.1.1 is used in this tutorial on Sandy Bridge):
  `module load openfoam`
- Follow the on-screen instruction to initialise the working environment:
  `source /app/materials/openfoam/2.1.1/sb/intel-13.0/intel-4.1/OpenFOAM-2.1.1/etc/bashrc`
- Confirm the loaded modules (note dependent compiler and MPI library are loaded automatically):
  `module list`

1. The OpenFOAM Tutorial test Case

Step 3 - Create a directory

Create a directory to hold any user data files. For this example, create a directory called OpenFOAM under your home directory:

```
cd ~
mkdir -p OpenFOAM/$USER-2.1.1/run
```


Ewrop & Chymru: Buddsoddi yn eich dyfodol
Cronfa Datblygu Rhanbarthol Ewrop
Europe & Wales: Investing in your future
European Regional Development Fund
ERDF
Llywodraeth Cymru Welsh Government

This is the default location for user files, also known as *$FOAM_RUN* in official OpenFOAM documentations and tutorials. To confirm this, type:

```
echo $FOAM_RUN
```

You can navigate to this directory any time by typing the short alias '**run**'.

Step 4 - Obtain a test case

OpenFOAM has a large number of predefined test cases, which are located at:

```
/app/materials/openfoam/2.1.1/intel-13.0/intel-4.1/OpenFOAM-2.1.1/tutorials
```

This location is also known as *$FOAM_TUTORIALS* in OpenFOAM terminology.

OpenFOAM has many different solvers for different types of physical problems. For this tutorial, we are using the *cavity* case, which uses the *icoFoam* solver (incompressible laminar flow solver). For simplicity, all required files are packed in a single file:

```
run
cp /app/materials/openfoam/2.1.1/example/cavity.tar.gz .
tar xvfz cavity.tar.gz
cd cavity
```

Step 5 - Submit a serial job

Now the current directory should contain all required files to run the OpenFOAM *cavity* case. In particular, the jobscript is called *run_openfoam.SLURM.q*; the *0* directory contains the initial conditions; the *constant* directory among other things contains the mesh setting; and the *system* directory contains the solver configurations.

- Submit the job using: **`sbatch run_openfoam.SLURM.q`**
- Check the job queue using: **`bjobs`**
- When completed, output can be found in a file called *OpenFOAM.o.<JobID>* and errors, if any, can be found in *OpenFOAM.e.<JobID>* (where *<JobID>* is the unique ID generated by the queuing software).
- If your job is successful, its output should be written in directories *0.1*, *0.2*, *0.3*.... each containing solutions corresponding to a time step

Step 6 - Submit a parallel job

To re-run the cavity case as a parallel job. Repeat step 4, but work with the *cavity-parallel.tar.gz* package instead. Note the following changes from the serial case:

- *system/decomposeParDict* contains additional settings for parallel domain decomposition.

Ewrop & Chymru: Buddsoddi yn eich dyfodol
Cronfa Datblygu Rhanbarthol Ewrop
Europe & Wales: Investing in your future
European Regional Development Fund
ERDF
Llywodraeth Cymru
Welsh Government

- The extra call to *decomposePar* in the jobscript to perform the domain decomposition.
- *icoFoam* solver is now invoked using *mpirun*. Lines 5 and 6 of the SLURM jobscript read:

```
#SBATCH --ntasks=4           # number of parallel processes (tasks)
#SBATCH --ntasks-per-node=4 # tasks to run per node
```

- meaning that 4 cores are used to run this job. If you change this number, the corresponding settings in *system/decomposeParDict* need to be changed too.
- The extra call to *reconstructPar* that assembles the solutions back to serial form for further processing.
- The generation of log files for solver/utility steps.

2. The OpenFOAM Benchmark Cases

Step 3 - Create a directory

Create a directory to hold any user data files. For this example, create a directory called OpenFOAM under your home directory:

```
cd ~
mkdir -p OpenFOAM/$USER-2.1.1/run
```

This is the default location for user files, also known as *$FOAM_RUN* in official OpenFOAM documentations and tutorials. To confirm this, type:

```
echo $FOAM_RUN
```

You can navigate to this directory any time by typing the short alias '`run`'.

Step 4 - Obtain a test case

OpenFOAM has a large number of predefined test cases, which are located at:

```
/app/materials/openfoam/2.1.1/intel-13.0/intel-4.1/OpenFOAM-2.1.1/tutorials
```

This location is also known as *$FOAM_TUTORIALS* in OpenFOAM terminology.

OpenFOAM has many different solvers for different types of physical problems. For this tutorial, we are using the *cavity* case, which uses the *icoFoam* solver (incompressible laminar flow solver). For simplicity, all required files are packed in a single file:

```
run
cp /app/materials/openfoam/2.1.1/example/cavity.tar.gz .
tar xvfz cavity.tar.gz
cd cavity
```

Step 5 - Submit a serial job

Now the current directory should contain all required files to run the OpenFOAM *cavity* case. In particular, the jobscript is called *run_openfoam.SLURM.q*; the *0* directory contains the initial conditions; the *constant* directory among other things contains the mesh setting; and the *system* directory contains the solver configurations.

- Submit the job using: `sbatch run_openfoam.SLURM.q`
- Check the job queue using: `bjobs`
- When completed, output can be found in a file called *OpenFOAM.o.<JobID>* and errors, if any, can be found in *OpenFOAM.e.<JobID>* (where *<JobID>* is the unique ID generated by the queuing software).
- If your job is successful, its output should be written in directories *0.1*, *0.2*, *0.3*.... each containing solutions corresponding to a time step

Step 6 - Submit a parallel job

To re-run the cavity case as a parallel job. Repeat step 4, but work with the *cavity-parallel.tar.gz* package instead. Note the following changes from the serial case:

- *system/decomposeParDict* contains additional settings for parallel domain decomposition.
- The extra call to *decomposePar* in the jobscript to perform the domain decomposition.
- *icoFoam* solver is now invoked using *mpirun*. Lines 5 and 6 of the SLURM jobscript read:

```
#SBATCH --ntasks=4          # number of parallel processes (tasks)
#SBATCH --ntasks-per-node=4 # tasks to run per node
```

- meaning that 4 cores are used to run this job. If you change this number, the corresponding settings in *system/decomposeParDict* need to be changed too.
- The extra call to *reconstructPar* that assembles the solutions back to serial form for further processing.
- The generation of log files for solver/utility steps.

Step 7 - Advanced uses

Visualisation of the solutions is beyond the scope of this tutorial. It can be done using ParaView either on the cluster or off-line at your desktop. In the former case, *paraFoam*, a wrapper to ParaView provided by OpenFOAM can be used. Also the original SSH connection to the HPC Wales cluster has to have X forwarding enabled.

The test cases above only use the pre-built OpenFOAM solvers and utilities. Many advanced OpenFOAM applications involve in writing one's own solvers and utilities, which requires composing source codes in C++, compiling and linking them to the pre-installed OpenFOAM

distribution. Interested users may follow the many tutorials available on the Internet. A good starting point is:

https://openfoamwiki.net/index.php/How_to_add_temperature_to_icoFoam

This document demonstrates how to build a custom solver based on *icoFoam* and set up a *cavity* case that solves an additional transport equation.

References

- OpenFOAM website: http://www.openfoam.org
- Official user guide: http://www.openfoam.org/docs/user