

SAPAC's OpenFOAM Manual

Felicia Tristante
School of Mechanical Engineering
The University of Adelaide¹

April 4, 2007

¹Summer Research Scholarship 2006/07 candidate supported by SAPAC

Contents

1	Introduction	5
1.1	About this User's Manual	5
1.2	About OpenFOAM	6
2	Getting Started	7
2.1	Installing OpenFOAM	7
2.1.1	On serial computers	7
2.1.2	On Parallel computers	8
2.2	Running Tutorials	8
3	Using OpenFOAM	9
3.1	On Serial Computers	9
3.1.1	Pre-Processing	10
3.1.2	Execution	10
3.1.3	Post-Processing	11
3.2	On Parallel computers	11
4	Application-Laminar Boundary Layer	13
4.1	Problem Specifications	13
4.2	The Blasius' Solution	13
4.3	OpenFOAM Application	14
4.3.1	Choosing a Solver	14
4.3.2	Mesh Generation	15
4.3.3	Physical properties	19
4.3.4	Initial and Boundary Conditions	19
4.3.5	Time and Convergence Control	20
4.4	Results	21
4.5	Conclusion and Discussion	22
4.6	Future Outlook	24
5	Submitting Jobs on Hydra	25
5.1	Logging into Hydra	25
5.2	File Management	25
5.3	Serial Computation	26
A	Useful Unix/Linux Commands	29

B Useful Links **31**

C Serial Installation Manual **33**

Chapter 1

Introduction

1.1 About this User's Manual

The purpose of this manual is to provide a quick reference on how to install and operate OpenFOAM Version 1.3 in a Unix/Linux environment. This Manual also documents the OpenFOAM code validation by comparing results obtained from its application to viscous flow over a flat plate with the analytical solution. This manual was written as the deliverable for a summer scholarship with SAPAC (South Australian Partnership for Advanced Computing); it is heavily based on SAPAC's expectation to investigate the performance of OpenFOAM in parallel mode.

Detailed explanations of OpenFOAM, its structure, coding and utilization can be found in both the OpenFOAM User Guide [3] and Programmers Manual [4] which can be found respectively at;

<http://foam.sourceforge.net/doc/Guides-a4/UserGuide.pdf>

<http://foam.sourceforge.net/doc/Guides-a4/ProgrammersGuide.pdf>

Note that 'User's Manual' will refer to OpenFOAM User's Manual [3], whereas 'Programmer's Manual' will refer to OpenFOAM Programmer's Manual [4].

Although OpenFOAM can run in a Windows environment using Cygwin (Linux/Unix interphase for Windows), it is not as widely used and popular as the conventional Linux/Unix environment; because of this, support for the Cygwin version is limited and is beyond the scope of this manual. When running in a Unix/Linux environment, OpenFOAM can be run using a Graphical User Interface (GUI) or a shellscript. This manual describes the fundamental steps involved in operating OpenFOAM using both the GUI as well as command line settings.

Users are encouraged to familiarize themselves with C++ libraries although an in-depth knowledge is not essential when using the precompiled applications. The user should be able to understand details of the C++ libraries by modifying the existing codes.

Please note that this document is a work in progress, and it will be updated whenever further progress is made. Any feedback, comments and corrections are welcomed from readers and should be forwarded to:

felicia_tristante@yahoo.co.jp or fellisia.tristante@student.adelaide.edu.au

1.2 About OpenFOAM

OpenFOAM (or Open Field Operation and Manipulation) is an open source CFD code, programmed and designed using a series of C++ libraries. It is a powerful tool in simulating various sorts of complex problems from fluid dynamics, turbulence and heat transfer to structural mechanics, electromechanics and finance. It implements the Finite Volume Method to solve the associated partial differential equations.

The benefit of using OpenFOAM over commercial CFD packages (e.g. CFX and Fluent) is that the user can have total control over data input, the numerical methods incorporated, the numerical models used, etc. On the downside, ignorance in coding details may lead to catastrophic simulation disasters.

Chapter 2

Getting Started

2.1 Installing OpenFOAM

2.1.1 On serial computers

OpenFOAM can be installed on various operating systems: Unix, Linux, Mac and Windows. The trick in installing OpenFOAM in a Windows environment is that users must download and install the cygwin software. This software can be found using the following link:

<http://www.cygwin.com/mirrors.html>

This manual describes the OpenFOAM installation on a Unix operating system only. Details on how to install OpenFOAM on any other operating system is beyond the scope of this manual and readers are referred to the OpenFOAM discussion board using the following link:

<http://openfoam.cfd-online.com/cgi-bin/forum/discus.cgi>

To download OpenFOAM according to your computer specifications, please refer to the following link:

<http://www.opencfd.co.uk/openfoam/download.html#download>

The downloaded OpenFOAM comes with FoamX and ParaView software. The use and description of these software is presented in this manual. All versions of OpenFOAM for all operating systems come with a README file containing a step by step guide to install OpenFOAM.

Please note that there are several unresolved issues regarding the installation of FoamX on a Mac, as FoamX is not currently available for a Mac. With the new Intel Mac, users might face some problems installing ParaView. These problems are under review and solutions will be made available at the following link:

<http://openfoam.cfd-online.com/cgi-bin/forum/show.cgi?1/2311>

A brief guide to installing OpenFOAM in a Unix environment, extracted directly from the README file found in `$Home/OpenFOAM/OpenFOAM-1.3` directory, is included here as Appendix C.

2.1.2 On Parallel computers

Attempts are being made at the present to install OpenFOAM on a parallel computer cluster and installation details will be made available as soon as positive results are achieved.

The newly released Intel Pentium Duo processor contains a dual processor, which enables users to run OpenFOAM in parallel using the dual processor on their own personal computer instead of a cluster system.

2.2 Running Tutorials

It is recommended that the new user work through the tutorials in Chapter 2 of the OpenFOAM User Guide, then the example cases in Chapter 3 of the Programmer's Guide for more advanced examples.

Please note there are some errors and inconsistencies in the tutorials in the User Guide. These errors are listed below:

1. Page U-31, Section 2.1.4.3, plotting streamlines. The trick is to select *Extract Parts* in Filter Utilities first, then choose *Internal flows only* in the Parameter Tabs. Ensure *volPointInterpolate(U)* is chosen in both *cavity.foam* and *extract parts Display* tab and click *Accept*. The instructions written in the second paragraph then can be followed.
2. Page U-36, Section 2.1.5.7, last paragraph. To use this *Probe* menu, the user has to *Extract Parts* as described above.
3. Page U-44, Section 2.1.10. The one paragraph in this section is a little misleading. It is true that the velocity graph can be generated at a specified time, yet the method described in this paragraph is very vague. To generate a velocity vector at a specified time, select the desired time from *cavity.foam*. Then create a *dummy.foam* file using the *touch* command and *open*. Select *Cell Centers* from the *Filter* utility menu, then click *Accept*.

All tutorial files are contained within OpenFOAM and can be found in the following folder:

```
$FOAM_TUTORIALS
```

When running your own simulation and considering the solver you intend to use, users are encouraged to copy all the main files, namely directory `0`, `constants`, `systems` and `samples`, from the tutorial directory and modify them accordingly. Note that the directory name listed in the tutorial directories identifies the type of solver used within the example.

Chapter 3

Using OpenFOAM

3.1 On Serial Computers

Many types of *solvers* exist within OpenFOAM, and these are described in detail in the OpenFOAM User Guide on page U-84. Table 3.1 lists several basic standard solvers that are described in the User's Manual and the Programmers Guide. The underlining procedures in using these *solvers* are similar in general. They start with pre-processing routines such as mesh generation, setting initial and boundary conditions, then execution, followed by post-processing routines such as mesh viewing, vector plots, graphs, etc. or extracting data from the simulation results.

Solver Name	Functions
icoFoam	for solving transient, incompressible, laminar flow of Newtonian fluids.
simpleFoam	for solving steady-state, incompressible, turbulent flow of non-Newtonian fluids.
potentialFoam	for solving simple potential flow, generating Navier-Stokes codes, starting fields.
turbFoam	for solving transient, incompressible, turbulent flow.
sonicFoam	for solving trans-sonic/supersonic, laminar flow of a compressible gas.
stressedFoam	for solving transient or steady-state linear-elastic, stress-strain solid body deformation.

Table 3.1: Several OpenFOAM Standard Solvers

Both the GUI and Unix phases of OpenFOAM can be used, as the GUI provides the user with information related to simulation choices, rather than searching the User Guide or the Programmer's Manual to find specific information. The tutorials contained in the User Guide demonstrate how to execute and use the GUI as well as the command line format. Basically the command needed to execute the GUI environment is:

FoamX

Please refer to the User Guide, pages U-(116-132) for a comprehensive explanation of how to use FoamX.

3.1.1 Pre-Processing

The pre-processing steps are listed below, noting that the text in italics indicates the filename with its corresponding directory:

1. Mesh Creation - *constant/polyMesh/blockMeshDict* (User Guide, pages U-(132-158))
2. Set IC and BCs - *0/U .. p*, etc.
3. Define physical properties - *constant/transportProperties .. turbulenceProperties*
4. Time control - *systems/controlDict*
5. Choosing numerical methods - *systems/fvSchemes .. fvSolution*

When defining the mesh, the ICs and the BCs, users must be careful of the units used. OpenFOAM does not require specified units in the simulation, and for this reason, users must be consistent with how units are incorporated.

3.1.2 Execution

The solver is initiated by entering:

```
<solverName>Foam_SOURCE_CASE
```

For example, if we are running the cavity case in icoFoam, we would type:

```
icoFoam . cavity
```

which means 'execute cavity' in the current directory.

Note that it is useful to keep a log of all simulation activities for pre-processing analysis purposes, e.g. for investigating simulation convergence using initial residuals, as can be seen in page U-170. This can be done using the following command:

```
<SolverName> . <Source> > log
```

For example

```
icoFoam . cavity >logFile&
```

Note that the symbol "&" means to run the simulation and write the processes in the file called logFile in the background. In other words, the user cannot monitor the simulation as it proceeds, but a log of the processes can be viewed by typing `fg` in the command line. Alternatively, the command `foamJob` produces the same effect (see the User Guide, page U-172). The command

```
foamJob <solver> <root> <case>
```

produces a comprehensive log file named `log` automatically. A log file that contains useful information can be extracted using the following command:

```
foamLog <root> <case> <logfileName>
```

3.1.3 Post-Processing

Graphical View

Using `paraFoam` (refer to the User Guide pages U-29,44,53,61,159-165). Grid mesh, vector velocity plots, streamlines plot, pressure distributions, etc. can be viewed using the command line:

```
paraFoam <root> <case>
```

It is highly recommended to **always** view the mesh grid to ensure its appropriateness using `paraFoam`.

Graphical data

Sampling data at a particular time can be achieved by using the `sample` command shown on page U-169 of the User Guide. The overall simulation output can be analyzed by using the techniques identified on page U-170 of the User Guide.

3.2 On Parallel computers

The setting up and pre-processing of problems on parallel computers is the same as for a serial computer. Execution and post-processing, however, is done differently. Parallel simulation reduces (clock) computation time significantly. Turbulence simulations, for example, would normally take weeks on serial computers. Parallel computers, however, depending on the number of processors or clusters used, would be able to significantly reduce the computational time. Details on how to run simulations in parallel can be found in the OpenFOAM User's Manual on page U-61.

Chapter 4

Application-Laminar Boundary Layer

4.1 Problem Specifications

OpenFOAM was initially applied to the problem of flow past a thin flat plate of length 1m with a uniform upstream velocity of $1ms^{-1}$ with $Re = 10^5$ and the growth of the boundary layer observed. The flow was assumed to be steady and incompressible. A graphical representation of this problem can be seen in Figure 4.1. A 0.2m non-viscous calculation domain is inserted before the viscous calculation domain, which represents the flat-plate, in order to avoid a singularity problem at the leading edge of the flat plate.

This problem represents the classical Blasius' problem, and solutions using OpenFOAM have been compared with those obtained from published solutions to Blasius' problem, in order to verify Openfoam.

4.2 The Blasius' Solution

The Blasius' solution solves the Navier-Stokes and continuity equations for a steady-state, two dimensional, incompressible, laminar fluid, which can be reduced to the following:

$$u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \nu \frac{\partial^2 u}{\partial z^2} \quad (4.1)$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \quad (4.2)$$

We assume that the pressure is zero everywhere in the domain and we apply the following boundary conditions on the plate:

$$u = v = 0 \quad (4.3)$$


```

laminarCoeffs
{
}
...

// ***** //

```

4.3.2 Mesh Generation

Mesh generation is the most important and time consuming step in setting up the problem as it is difficult to predict the appropriate refinement. Generally mesh coarseness is estimated based on experience. The rule of thumb in creating a mesh is to have refinement where flows vary significantly, and a coarse mesh where flow variation is much less. In this case, as we are interested in comparing the velocity profile of the flat-plate boundary layer at $Re = 10^5$, it is wise to estimate the boundary layer thickness first. Using equation 4.5, a value of $\delta = 16mm$ is obtained.

$$\frac{\delta}{x} = \frac{5}{\sqrt{Re_x}} \quad (4.5)$$

δ = Boundary layer thickness
 x = Length of x domain (flat-plate)
 Re_x = Reynolds number calculated based on length of x-domain

$$Re_x = \frac{\rho U(x) x}{\mu} = \frac{U_x x}{\nu} \quad (4.6)$$

ρ = Flow density ($kg.m^{-3}$)
 μ = Viscosity ($kg.m^{-1}.s^{-1}$)
 ν = Kinematic viscosity ($kg.m^2.s^{-1}$)

As a result, we can confidently conclude that the y-domain of interest is between 0-16mm. For $y > 16mm$, the flow velocity is effectively uniform and equal to U. In order to avoid singularity problems, a slightly larger refined region has been chosen.¹ These considerations have resulted in the following entries being written into the `blockMeshDict` file:

```

// * * * * * //

convertToMeters 1e-03;

vertices
(
    //reference node number
    (0 0 0) //0
    (200 0 0) //1
    (1200 0 0) //2
    (1200 0 1) //3

```

¹A potential area of research.

```

(200 0 1) //4
(0 0 1) //5
(0 20 0) //6
(200 20 0) //7
(1200 20 0) //8
(1200 20 1) //9
(200 20 1) //10
(0 20 1) //11
(0 300 0) //12
(200 300 0) //13
(1200 300 0) //14
(1200 300 1) //15
(200 300 1) //16
(0 300 1) //17
);

blocks
(
    //reference block number
    hex (0 1 7 6 5 4 10 11) (100 10 1) simpleGrading (1 10 1) //1
    hex (1 2 8 7 4 3 9 10) (500 10 1) simpleGrading (1 10 1) //2
    hex (6 7 13 12 11 10 16 17) (100 20 1) simpleGrading (1 10 1) //3
    hex (7 8 14 13 10 9 15 16) (500 20 1) simpleGrading (1 10 1) //4
);

edges
(
);

patches
(
    patch inlet
    (
        (6 0 5 11)
        (12 6 11 17)
    )
    patch outlet
    (
        (2 8 9 3)
        (8 14 15 9)
    )
    wall upperWall
    (
        (13 12 17 16)
        (14 13 16 15)
    )
    wall lowerWall

```

```
(
    (1 2 3 4)
)
symmetryPlane down
(
    (0 1 4 5)
)
);
```

```
mergePatchPairs
(
);
```

```
// ***** //
```

Note that there is a 5% (0.2m) region in front of the simulated domain to reduce flow inlet effects. This value of 0.2m is arbitrary.

The following command will generate the mesh when you are inside the folder, say Bllaminar:

```
blockMesh . .
```

which means run blockMesh in folder Bllaminar.

It is recommended to **always** view the mesh after generation using paraFoam to make sure the spacing between the mesh and the resulting mesh shapes are appropriate. Note that OpenFOAM uses the Finite Volume Method, so that the shape of the mesh elements is not necessarily square or cubical.

The mesh can be viewed through ParaView, which is included in paraFOAM. The following command allows the mesh to be reviewed:

```
paraFoam . <source>
```

Figure 4.1 summarizes these blockMeshDict entries. The numbers 0-17 shown in figure 4.1 indicate reference node numbers, whereas the ellipsed numbers 1-4 indicate the block number. The mesh configuration used in this simulation can be seen in figure 4.2. Note how the mesh is set up. As mentioned, sufficient refinement is required to model the mesh over the y domain of [0,20mm].

In this simulation, three types of meshes, referred to as Coarse, Medium and Fine, are simulated and compared with the Blasius' Solution. The types of meshes are summarised in Table 4.2;

Mesh Name	Mesh Size (mm)	
	Block 1 and 2	Block 3 and 4
Coarse	20 x 0.2 x 1	20 x 11.2 x 1
Medium	10 x 0.1 x 1	10 x 5.6 x 1
Fine	5 x 0.05 x 1	5 x 2.8 x 1

Table 4.2: Mesh type summary



Figure 4.1: Computation Domain layout

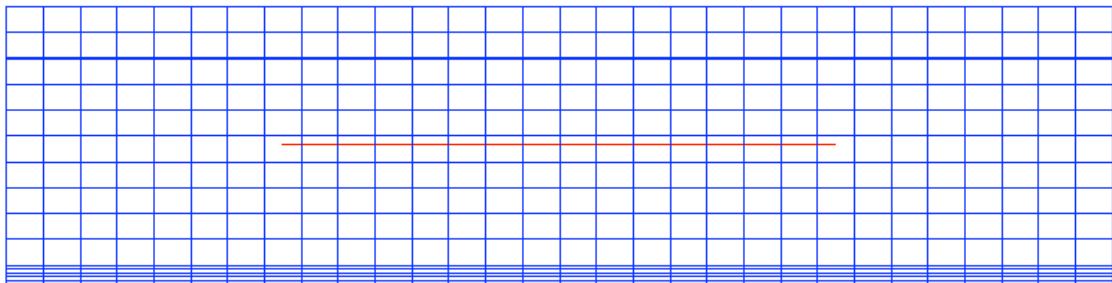


Figure 4.2: Mesh Layout(the line in the centre identifies the cursor location)


```

writePrecision 6;

writeCompression uncompressed;

timeFormat      general;

timePrecision 6;

runTimeModifiable yes;

// ***** //

```

Simulation convergence can be determined by monitoring residual data from the `log` file using the following:

```
foamLog <root> <case> <logFileName>
```

or the following:

```
foamLog . . log
```

which means to run `foamLog` in the current directory (e.g. `Bllaminar`) and extract the log-file called "log" that is created automatically if the `foamJob` command was utilised earlier. This produces a directory named `logs`, which contains sub directories such as `countCumulative_0`, `Ux_0`, `plters_0`, `Uy_0`, etc. The only data that we are interested in is `Ux_0`, which represents the initial residuals of the x-velocity component (U_x). This file contains two columns of data: the first column indicates the computation time, while the second column indicates the initial residuals of $U(x)$. Figure 4.3 shows an example of time convergence.

In theory, convergence is reached when the error (in this case the residuals) stops growing with time. This phenomenon can be seen in Figure 4.3 where, for the Coarse Mesh, the residuals fluctuate about a straight line. In most scientific applications, residuals of the order of 10^{-6} or 10^{-7} provide acceptable numerical convergence.

4.4 Results

Figure 4.3 shows that the time for convergence is directly proportional to the number of grid points in the solution domain or inversely proportional to the mesh size. As can be seen in this figure, the coarse mesh takes approximately 4,000 seconds to converge whereas the medium and fine mesh take more than 10,000 seconds. Obviously, looking at the behaviour of the residuals in Figure 4.3, the fine mesh would take a much longer time to converge compared to the medium mesh.

If residuals of the order of 10^{-7} are used to determine numerical convergence, Figure 4.3 indicates that convergence for the Coarse, Medium and Fine mesh occurs at approximately 2,000 seconds, 6,000 seconds and over 10,000 seconds respectively.

Figure 4.4 presents comparisons between the Blasius solution and solutions from the Coarse, Medium and Fine meshes with no grading at time 5000 seconds, 10,000 seconds and 10,000 seconds respectively. The Medium mesh velocity profile, which was sampled at 10,000 seconds when the residuals are of the order of 10^{-7} , coincides with the Coarse mesh profile. However, the residuals of the Fine mesh at 10,000 seconds are only of the order of 10^{-5} ; clearly, the velocity profile at this point would be less accurate compared to the Coarse and Medium mesh.

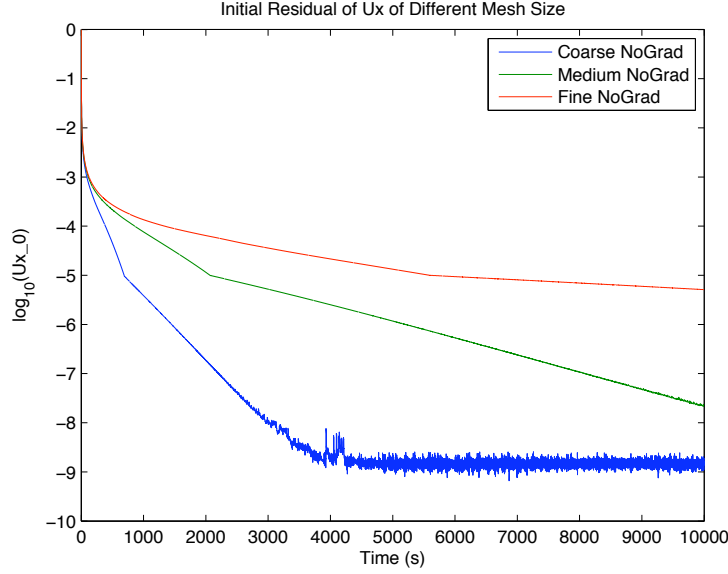


Figure 4.3: Comparison of residuals for U for the three mesh sizes

4.5 Conclusion and Discussion

Section 4.4 suggests that the Coarse mesh with no grading seems to be the most promising of the three examined. However, investigations of the skin friction coefficient (Table 4.4) suggests that the Medium mesh is marginally more accurate compared to the Blasius skin friction. Equations 4.7 and 4.8 are used to produce the tabulated shear stress τ_w and skin friction C_f results shown in Table 4.4. This table also shows the significant increase in computation time for the three meshes. Looking at the trend in this table, and figures 4.3 and 4.4, it can be concluded that the Coarse mesh is the best for the present and future simulations, as it is accurate (very close to the Medium mesh) and uses the least amount of computing time.

$$\tau_w = \mu \left. \frac{\partial u}{\partial y} \right|_{y=0} \quad (4.7)$$

$$C_f = \frac{\tau_w}{\frac{1}{2}\rho U^2} \quad (4.8)$$

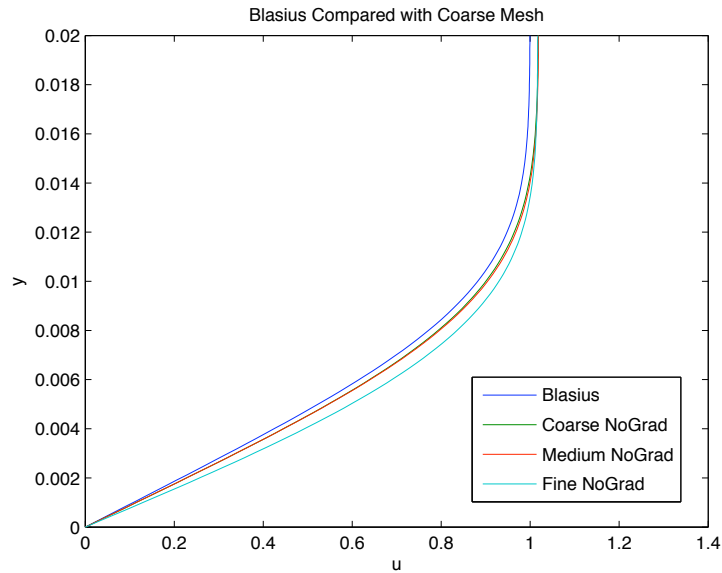


Figure 4.4: All mesh comparisons with the Blasius solution

	Shear Stress τ_w	Skin Friction C_f	%error	Simulation (clock) Time (seconds)
Blasius	1.07715×10^{-3}	2.15430×10^{-3}	N/A	N/A
Coarse	1.15200×10^{-3}	2.30401×10^{-3}	6.949%	526
Medium	1.14865×10^{-3}	2.29730×10^{-3}	6.638%	5,575
Fine	1.130418×10^{-3}	2.60837×10^{-3}	21.077%	59,000

Table 4.4: Skin Friction of different mesh compared to the Blasius' Solution

4.6 Future Outlook

It is anticipated that a thorough grid convergence analysis will be conducted using the ICs and BCs shown in Table 4.3. When this verification process is completed, the case will be extended to three dimensions, followed by a two dimensional turbulent simulation using the Large Eddy Simulation (LES) solver. Ultimately, it is planned to extend the investigation to a three dimensional turbulent boundary layer problem using LES.

Chapter 5

Submitting Jobs on Hydra

Hydra is a cluster system maintained by SAPAC. In order to be able to submit jobs on Hydra, an account needs to be set up. Information regarding Hydra and how to set up an account can be found from the SAPAC website:

<http://www.sapac.edu.au/>

5.1 Logging into Hydra

Once an account is set up, the account holder can log into Hydra by entering the following command:

```
ssh username@hydra.sapac.edu.au
```

followed by the user's password.

5.2 File Management

All files required to solve a specific problem must be copied into the Hydra account. For example, if we want to simulate cavity flow (the sample case contained in the tutorial), which is located in the following folder:

```
TUTORIAL/icoFoam
```

we can use the following command to copy the tutorial sample across to Hydra:

```
scp -r $TUTORIAL/icoFoam/cavity username@hydra.sapac.edu.au:.
```

Note that the `-r` is a recursive command which enables all files contained in the cavity directory to be copied. The user will then be asked for a password, and this will cause the files to be copied.

5.3 Serial Computation

Instead of typing,

```
blockMesh . .
```

or

```
simpleFoam . . >log&
```

etc, a .pbs file containing all the required commands needs to be set up. The following commands are written in typical .pbs form (where the file name is called filename.pbs) to submit a job to Hydra:

```
#!/bin/sh

#PBS -V

### Job name
#PBS -N OpenFOAM

### Join queuing system output and error files into a single output file
#PBS -j oe

### Send email to user when job ends or aborts
#PBS -m ae

### email address for user
#PBS -M fellisia.tristante@student.adelaide.edu.au

### Queue name that job is submitted to
#PBS -q hydra

### Request nodes NB THIS IS REQUIRED
#PBS -l nodes=1,walltime=20:00:00

# This job's working directory
echo Working directory is $PBS_O_WORKDIR
cd $PBS_O_WORKDIR
echo Running on host `hostname`
echo Time is `date`

# Run the executable
blockMesh . .

simpleFoam . .>log

foamLog . . log
```

To submit a batch job, the following command is entered:

```
qsub filename.pbs
```

Once this is done, the job number will appear on the command window.

Appendix A

Useful Unix/Linux Commands

bg	stands for background. It is normally used in combination with Ctrl+Z to put the process in the background.
cd <directory>	stands for change directory. This command allows the user to change to the specified directory.
cat <filename>	prints the contents of the specified filename into the screen.
cd ..	to go back to the previous directory.
cd	to go back to the home directory.
<command> help]	finding help with the corresponding command.
cp file SOURCE DIRECTORY]	copy file from the specified source to the specified directory.
Ctrl+Z	stops an active process.
fg	stands for foreground, putting the job back into the foreground.
kate <filename>	one kind of text editor.
kill -g PID	will end the active task with the Process ID number (PID#)
latex	open L ^A T _E X. Latex is the Unix/Linux document preparation system.
less <filename>	display contents of specified filename page by page.
ls	will list the contents of the current directory.
lyx	open lyx. It is the graphic interface of L ^A T _E X.
man	provides information on the specified command.
<command>	
man man	provides information on how to use the manual.
more	to look at the contents of specified filename.
<filename>	
matlab	will open MATLAB.
ps	this command is the same as the Ctrl+Alt+Del command in Windows. It shows the task manager.
pwd	stands for ' <i>present working directory</i> ', and shows what directory we are currently working in
rm	remove the specified files

Appendix B

Useful Links

OpenFOAM Official Website. All sorts of information, such as downloads, how to install, the OpenFOAM User Guide and the Programmer's Manual as well as explanations about OpenFOAM, are available at

<http://www.open CFD.co.uk/openfoam/>

OpenFOAM Discussion Link. To post discussion questions and answers, a one-off user registration (free) is required and login is required for each session.

<http://openfoam.cfd-online.com/cgi-bin/forum/discus.cgi>

The following websites are developed by Wiki.Ltd. They provide detailed technical information.

http://www.mfix.org/mwiki/index.php/Main_Page

http://openfoamwiki.net/index.php/Main_Page

Appendix C

Serial Installation Manual

1. Downloading

~~~~~

Download and unpack the files in the \$HOME/OpenFOAM directory from;  
<http://www.openfoam.org/download.html>

### 2. Setting the Environment

~~~~~

The environment variable settings can be found at the following directory;
\$HOME/OpenFOAM/OpenFOAM-1.3/.OpenFOAM-1.3

type 'echo \$SHELL' if you are unsure to what type of shell you are running.

a) For bash or ksh

source the .OpenFOAM-1.3/bashrc file by adding the following line to
the end of \$HOME/.bashrc file:

```
. $HOME/OpenFOAM/OpenFOAM-1.3/.OpenFOAM-1.3/bashrc
```

Then update the environment variables by typing:

```
. $HOME/.bashrc
```

b) For tcsh or csh

source the .OpenFOAM-1.3/cshrc file by
adding the following line to the end of your \$HOME/.cshrc file:

```
source $HOME/OpenFOAM/OpenFOAM-1.3/.OpenFOAM-1.3/cshrc
```

Then update the environment variables by typing in the terminal:

```
source $HOME/.cshrc
```

3. Testing Installation

~~~~~

foamInstallationTest' script can be found in:  
\$HOME/OpenFOAM/OpenFOAM-1.3/bin/foamInstallationTest  
to check whether or not the environment variables have been set properly  
or not. If no problems are reported, proceed to getting started with  
OpenFOAM; otherwise, go back and check you have installed the  
software correctly and/or contact your system administrator.

#### 4. Getting Started

~~~~~

Create a project directory within the \$HOME/OpenFOAM directory named
<userName>-1.3 (e.g. 'chris-1.3' for user chris) and create a
directory named 'run' within it, e.g. by typing:

```
mkdir -p $HOME/OpenFOAM/${LOGNAME}-1.3/run
```

Copy the 'tutorial' examples directory in the OpenFOAM distribution to the
'run' directory. If OpenFOAM environment variables are set correctly, then
the following command will be correct:

```
cp -r $WM_PROJECT_DIR/tutorials $HOME/OpenFOAM/${LOGNAME}-1.3/run
```

Run the first example case of incompressible laminar flow in a cavity:

```
cd $HOME/OpenFOAM/${LOGNAME}-1.3/run/tutorials/icoFoam  
blockMesh . cavity  
icoFoam . cavity
```

NOTE: the above document was extracted and modified from the README script
which can be found in '\$HOME/OpenFOAM/OpenFOAM-1.3' directory.

Bibliography

- [1] American Institute of Aeronautics and Astronautics (AIAA), *Guide for the Verification and Validation of Computational Fluid Dynamics Simulation*, AIAA, USA, 1998.
- [2] Munson, B. R., Young, D. F., Okiishii, T. H., “Prandtl/Blasius Boundary Layer Solution” *Fundamentals of Fluid Mechanics*, 5th ed., Chapter 9, pp.497–501. John Wiley & Sons, inc., USA, 2006.
- [3] OpenCFD Limited *OpenFOAM: The Open Source of CFD Toolbox - User’s Guide* November 2001, Boston. [Online]<http://foam.sourceforge.net/doc/Guides-a4/UserGuide.pdf>
- [4] OpenCFD Limited *OpenFOAM: The Open Source of CFD Toolbox - Programmer’s Guide* November 2001, Boston. [Online]<http://foam.sourceforge.net/doc/Guides-a4/ProgrammersGuide.pdf>