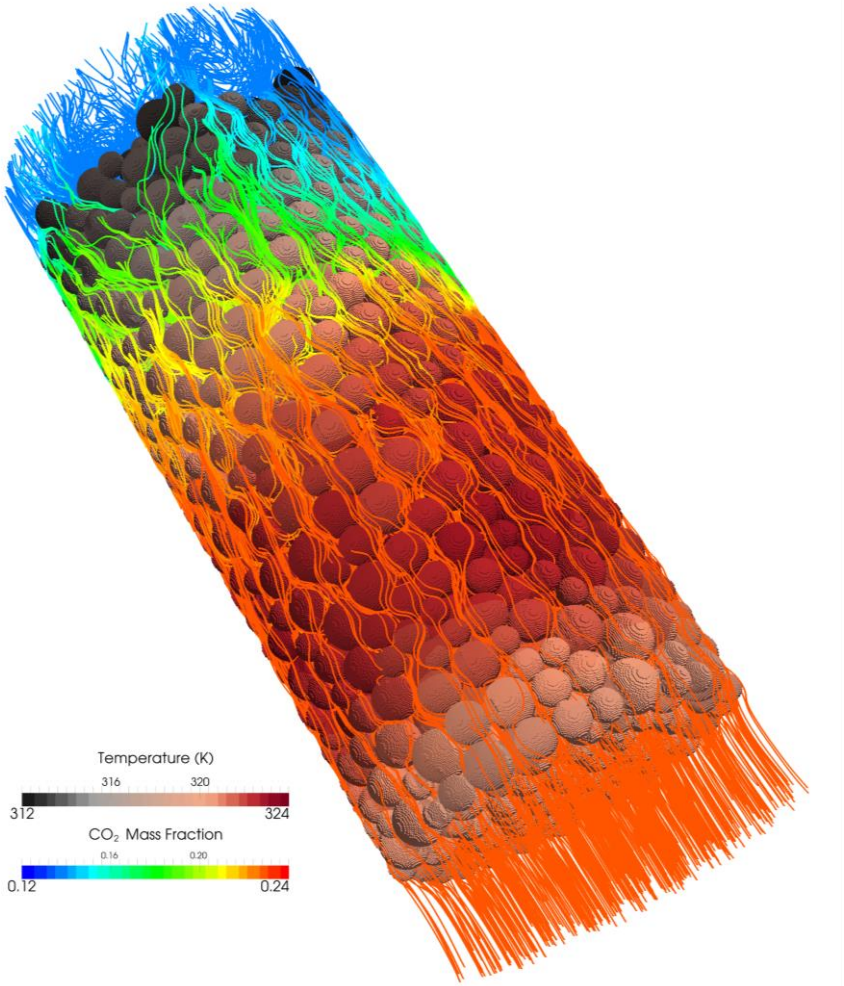
OpenFOAM® Basic Training

Tutorial One



3rd edition, Feb. 2015

www.thvt.at

Editors and Contributors:

- Bahram Haddadi (TU Wien)
- Christian Jordan (TU Wien)
- Jozsef Nagy (JKU Linz)
- Clemens Gößnitzer (TU Wien)
- Vikram Natarajan (TU Wien)
- Sylvia Zibuschka (TU Wien)
- Michael Harasek (TU Wien)

Cover picture from:

- Bahram Haddadi, The image presented on the cover page has been prepared using the Vienna Scientific Cluster (VSC).

icoFoam – elbow (mesh)

Simulation

Using icoFoam solver, simulate 75 s of flow in an elbow for following GAMBIT meshes:

- Tri-mesh (comes with OpenFOAM® tutorial)
- Hex-mesh coarse (check GAMBIT "elbow 2D" tutorial)
- 2 times finer hex-mesh (refine previous step mesh)

Objectives

- Looking at the initial values for p and U
- Make sure all boundaries in GAMBIT are called accordingly
- Make sure after importing the mesh no additional surfaces are introduced
- Make sure the surfaces in the file constant/polyMesh/boundary are defined correctly.

Post processing

Import your simulation to ParaView, using extracted data from paraview make two diagrams (using spreadsheet calculators) of pressure and velocity magnitude along the line between two tubes, do the same for all three simulations.

Step by step simulation

Setting system environment

Make sure your system environment is set correctly, check Appendix B.

Copying tutorial

Open a terminal and copy the elbow tutorial from the following path to your working directory (see Appendix A for using a terminal in Linux):

```
~/OpenFOAM/OpenFOAM-2.3.0/tutorials/incompressible/icoFoam/

elbow
```

Converting mesh

The mesh which is produced by GAMBIT is not directly compatible with OpenFOAM®. First, the mesh needs to be converted to an OpenFOAM® mesh, using following tool:

```
>fluentMeshToFoam elbow.msh
```

If the mesh was created in mm and is converted using the mentioned command it will convert the mesh with wrong dimensions, since all the units in OpenFOAM® are SI[1] Units. There are different flags included with most of OpenFOAM® tools, for checking them use the flag `-help` after the command, e.g.:

```
>fluentMeshToFoam -help
```

The output gives an overview of available options of the tool and also a short description on how to use it:

```
Usage: fluentMeshToFoam [OPTIONS] <Fluent mesh file>
options:
  -case <dir>       specify alternate case directory, default is the cwd
  -noFunctionObjects
                    do not execute functionObjects
  -scale <factor>   geometry scaling factor - default is 1
  -writeSets        write cell zones and patches as sets
  -writeZones       write cell zones as zones
  -srcDoc           display source code in browser
  -doc              display application documentation in browser
  -help             print the usage

Using: OpenFOAM-2.3.0 (see www.OpenFOAM.org)
Build: 2.3.0-f5222ca19ce6
```

The `-scale` flag is used for converting the mesh dimensions from other units to SI units, e.g. if the mesh was created in mm it will be converted to meter by using `-scale 0.001` and if the flag is omitted, uses `1`:

```
>fluentMeshToFoam elbow.msh -scale 1.0
```

Note: The mesh which is imported to OpenFOAM® should be a three dimensional mesh. For carrying out 2D (also 1D) simulations a three-dimensional mesh should be

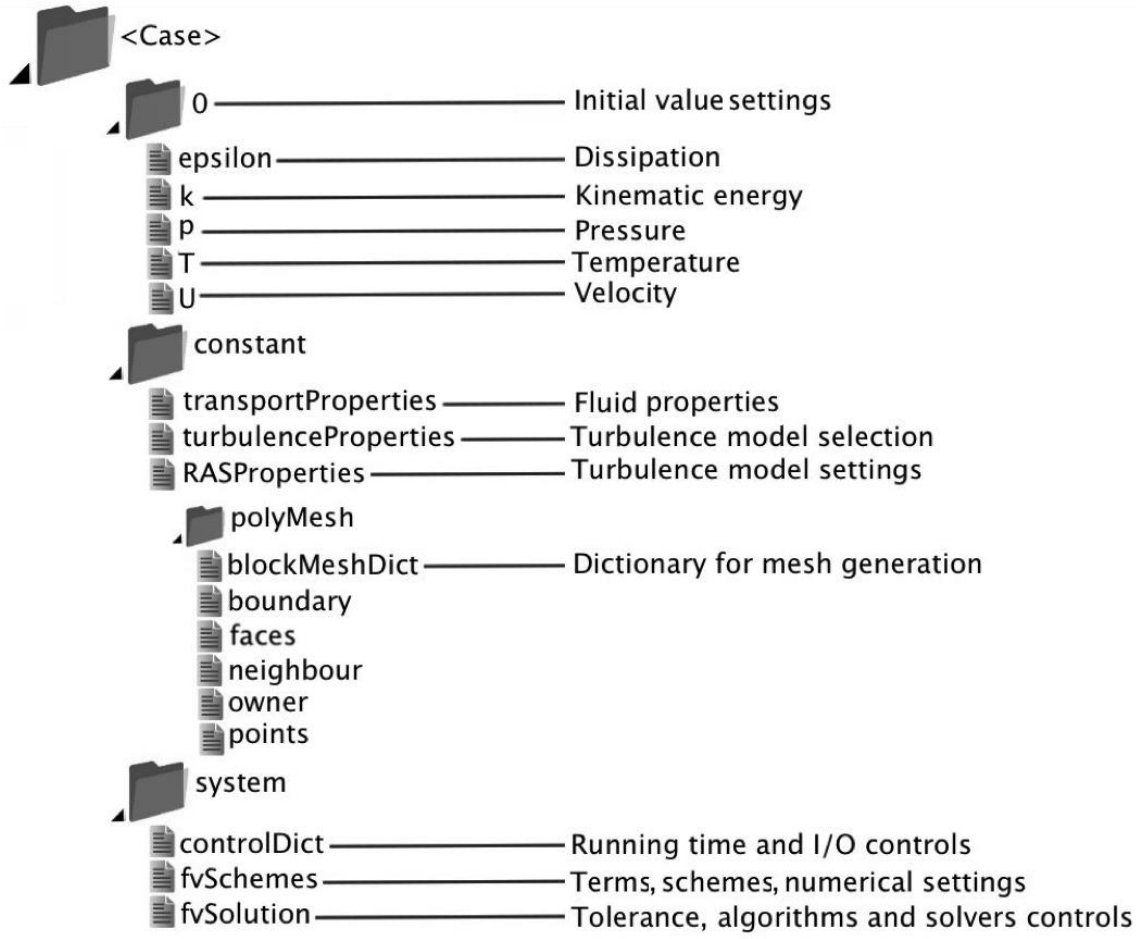
[1] International System of Units

created with just one cell in the third dimension (For 1D, one cell in the second and also one cell in the third direction).

Note: If there are internal boundaries in the mesh, there is another tool, fluent3DMeshToFoam. Using this tool, the internal boundaries will be kept during conversion.

Case structure

Most of the cases in OpenFOAM® have the following basic case structure (directory tree):



There are three main directories (0, constant, system) in each case foloder:

0 directory

The 0 directory includes the initial conditions for running the simulation. In each file in this folder the initial conditions for one property can be set. The files are named after the property they are standing for, e.g. usually T file includes temperature initial conditions. In the elbow example there are only two files inside the 0 directory, p and U. p stands for pressure and U stands for velocity. Checking p:

```
>nano[1] p
```

It will be like this:

[1] nano is a text editor used in Linux OS (for closing and saving: ctrl+x)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.com                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    wall-4
    {
        type            zeroGradient;
    }

    velocity-inlet-5
    {
        type            zeroGradient;
    }

    velocity-inlet-6
    {
        type            zeroGradient;
    }

    pressure-outlet-7
    {
        type            fixedValue;
        value           uniform 0;
    }

    wall-8
    {
        type            zeroGradient;
    }

    frontAndBackPlanes
    {
        type            empty;
    }
}
// ************************************************************************* //
```

In the `dimensions` the physical dimension according to SI base units of the quantity is defined, for example here it shows that the p dimension is $(m/s)^2$.

Note: As you can see the p unit is not the pressure unit (Pa). It is due to the fact that in incompressible solvers in OpenFOAM®, p is defined as "reduced" pressure divided by density.

Note: All units are SI units in OpenFOAM®.

Note: In the dimension matrix the first number presents mass unit power, the second one the length, the third one time, the forth one the temperature and the fifth one the quantity (mole).

The `internalField` sets the initial field of a specific quantity in the solution domain.

The type of each of our boundaries as well as the value of this quantity on the boundaries is defined in the `boundaryField`. There are different types of boundary conditions in OpenFOAM®:

- `zeroGradient`: Applies a zero gradient boundary type to this boundary (Neumann boundary condition).
- `fixedValue`: Applies a fixed value to this boundary (Dirichlet boundary condition).
- `empty`: It is for sides, which are vertical to the direction which is not going to be considered (e.g. in 2D simulations these boundaries are vertical to the third dimension). In this boundary type both of the sides vertical to one dimension should be selected together and named as one boundary.

Note: If a `fixedValue` *boundary condition with* `value` *equals* `$internalField` *is used, it is equal to using* `zeroGradient`*, except* `zeroGradient` *applies the boundary condition implicitly, but* `fixedValue` *with* `$internalField` *value applies the boundary condition explicitly.*

Note: In some mesh creation softwares like GAMBIT, empty boundary condition do not exist. All the faces perpendicular to the direction which is not going to be considered should be defined as a new boundary with type `wall`*. After converting the mesh to OpenFOAM® mesh, modify that boundary in the file constant/polyMesh/boundary, and change its* `type` *from* `wall` *to* `empty`*, and also change* `inGroups` *from* `wall` *to* `empty`*.*

The U file has to be defined via three components (since velocity is a vector): first one stands for the x component, second one for the y component, and the third one for the z component. For this case setup the z component is always zero because it is a 2D simulation and no calculations will be done in the z direction. The boundaries vertical to z direction have been already set to empty.

constant directory

The constant directory usually consists of a subdirectory and some files. The files (usually) include material properties, simulation physics and chemistry. In the directory "polyMesh" the mesh data are stored (in this case the data for converted mesh). The boundary file in this polyMesh directory includes the mesh boundary data, e.g. type, number of faces on this boundary and also starting face number (unique face IDs) for this boundary (for the sake of space, the dictionary headers will not be included in this scope any more):

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

6
(
    wall-4
    {
        type            wall;
        nFaces          100;
```

```
        startFace       1300;
    }
    velocity-inlet-5
    {
        type            patch;
        nFaces          8;
        startFace       1400;
    }
…
    frontAndBackPlanes
    {
        type            empty;
        nFaces          1836;
        startFace       1454;
    }
)

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

Comparing the boundary names with the ones set in GAMBIT, they should be the same, and also the boundary types (walls should be wall, inlet and outlets should be patch, empties should be empty). Starting cell number and also number of each face cells can also be checked here.

By opening the transportProperties file, properties dimensions and also the property value can be found and edited, e.g.:

```
nu              nu [ 0 2 -1 0 0 0 0 ] 0.01;
```

`nu` is the fluid kinematic viscosity, which is 0.01 m^2/s for this example.

system directory

Solver and finite volume methods settings can be found and changed in this directory. There are three main files in this directory:

- fvSchemes: The discretization scheme which is used for each term of the equations are set in this file.
- fvSolution: Contains the settings to the coupling method of pressure and velocity, the numerical methods, which are used for solving different quantities, and also the final tolerance for convergence of that quantity.
- controlDict: The time, when simulation starts (`startFrom`), the time when the simulation finishes (`stopAt`), the time step (`deltaT`), the data saving interval (`writeInterval`), the saved data file format (`writeFormat`), the saved file data precision (`writePrecision`), and also if changing the files during the run can affect the run or not (`runTimeModifiable`) are set in this file.

Note: If the write format is `ascii`*, then the simulation data which is written to the file can be opened and read using any text editor. If the format is* `binary`*, the data will be written in binary style and is not readable by text editors. The advantage of* `binary` *over* `ascii` *is the smaller file size, and consequently faster conversion and writing to disk, for big simulations.*

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

application     icoFoam;
```

```
startFrom       latestTime;

startTime       0;

stopAt          endTime;

endTime         75;

deltaT          0.05;

writeControl    timeStep;

writeInterval   20;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable yes;

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

Note: This simulation continues from the last time step data which is saved (`latestTime`). If there was no saved data it will start from start time (`startTime`), which is zero in this case.

Running simulation

The simulation can be run by typing the solver's name and executing it:

```
>icoFoam
```

Note: For running the simulation the solver command (e.g. icoFoam) should be executed inside the copy of the tutorial main folder. For example: The command should be executed in the elbow folder, if it was run at some subfolders or somewhere else, the simulation will fail.

Exporting simulation data

The data files created by OpenFOAM® should be exported (converted) by the appropriate tools, to the post processing tools data format. For ParaView:

```
>foamToVTK
```

where VTK is the ParaView data format. This command should be also executed in the case main directory, e.g. elbow. Here, ParaView is used as the post-processing tool, for running it

```
>paraview &
```

Note: There is also another option to open the OpenFOAM® simulation results with ParaView without converting them to VTK; Create an empty text file in the main case directory, name it <someName>.foam (e.g. foam.foam), and execute the following

command. This method is good for fast evaluation of the data in the middle of the simulation or with a decomposed case in parallel simulations:

```
>paraview foam.foam &
```

Note: By putting & at the end of command, the command line will remain active and ready for further inputs while that program is running.

Examining different meshes

Do the same for the other two meshes. Just the mesh for the first simulation is included in the elbow example of OpenFOAM®. For the other two simulations the mesh should be provided by the user. For finding the tutorials on how to create the geometry and the mesh, search the internet for "GAMBIT elbow mesh 2D". The dimensions and also the mesh info are provided in that tutorial. Try to create it by using GAMBIT. When you are done you have to convert it into a 3D mesh with 1 cell in the z-direction.

The comparisons of all three cases results and charts are shown below.

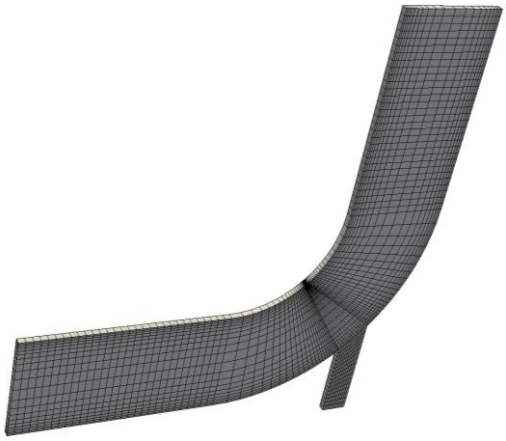


Figure 1.1 The Hex Fine mesh created using GAMBIT

Note: As mentioned before after converting the mesh, change the boundaries perpendicular to the direction which is not going to be considered (the z direction) from `wall` *to* `empty` *(just replace* `wall` *with* `empty` *for this boundary).*

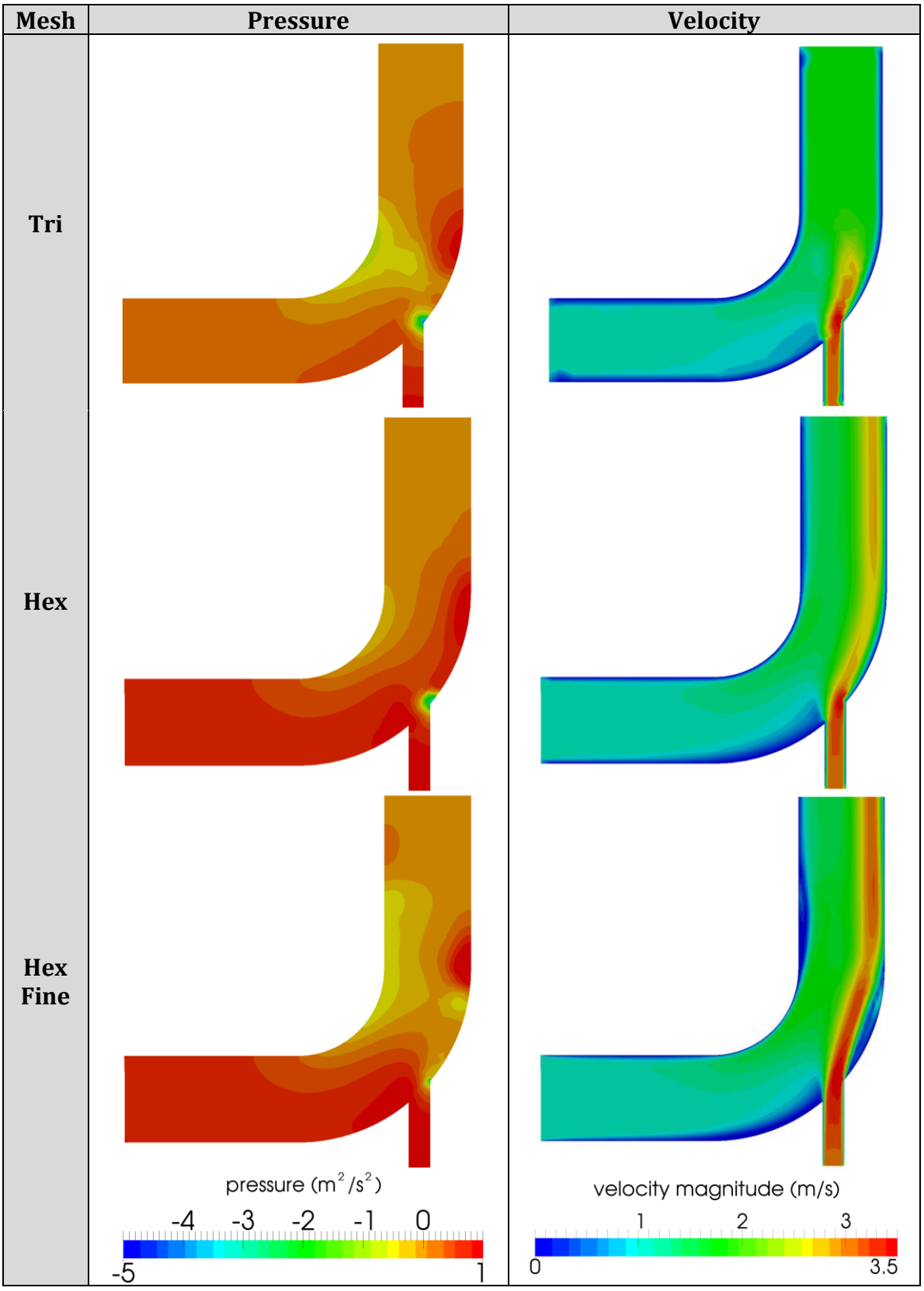


Figure 1.2 Comparison of different mesh type results at t=75 s

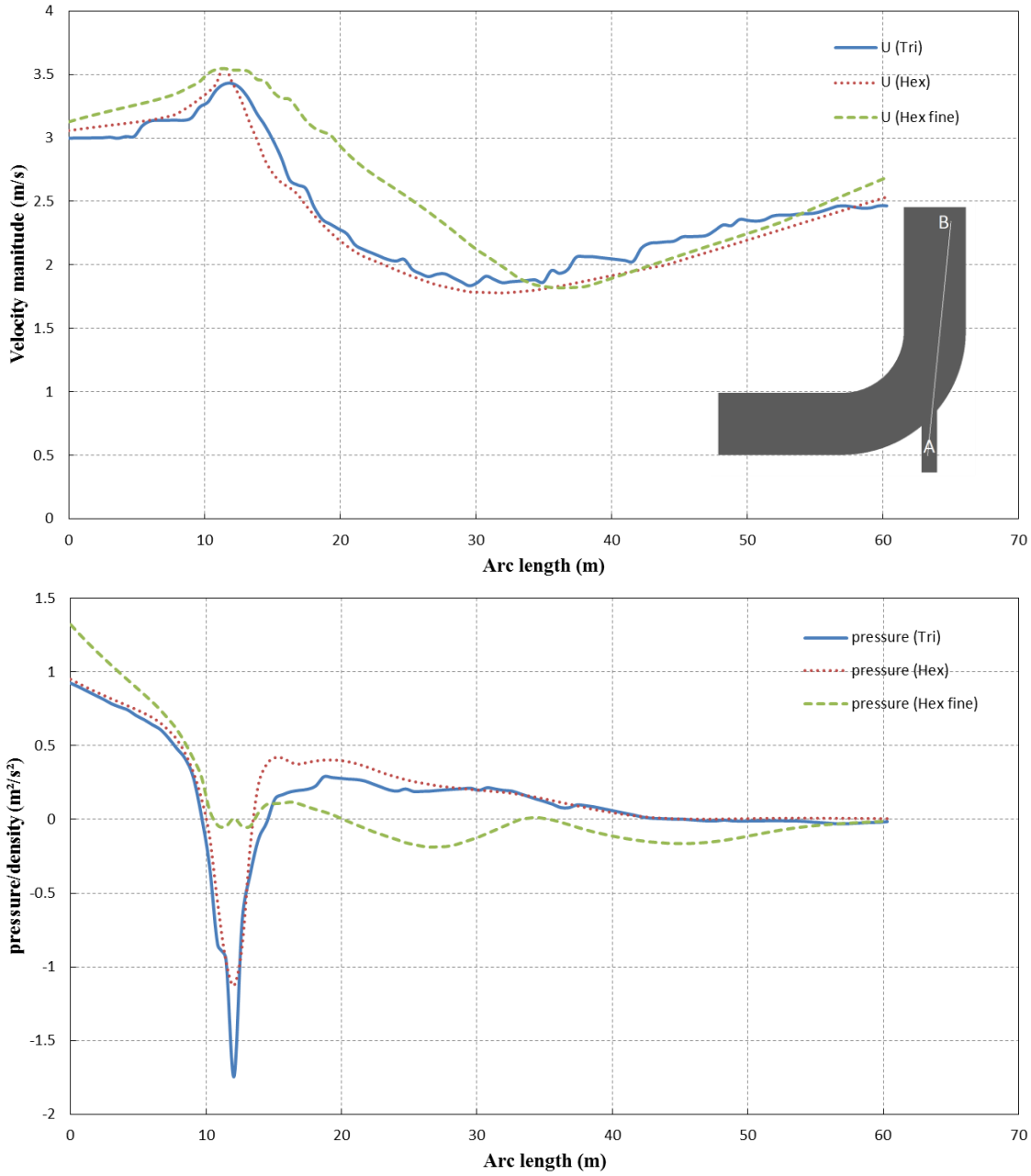


Figure 1.3 Pressure and velocity for different meshes at t=75 s, along the arc shown

The comparison plots are along the line between points A (54 0 0) at the small tube entrance and B (60 60 0) at the large tube exit part (length units are in meter).

Note: For extracting data over a line, the line should be defined in ParaView using "Plot Over Line", then the data over this line can be exported by choosing Save Data from File menu in ParaView.