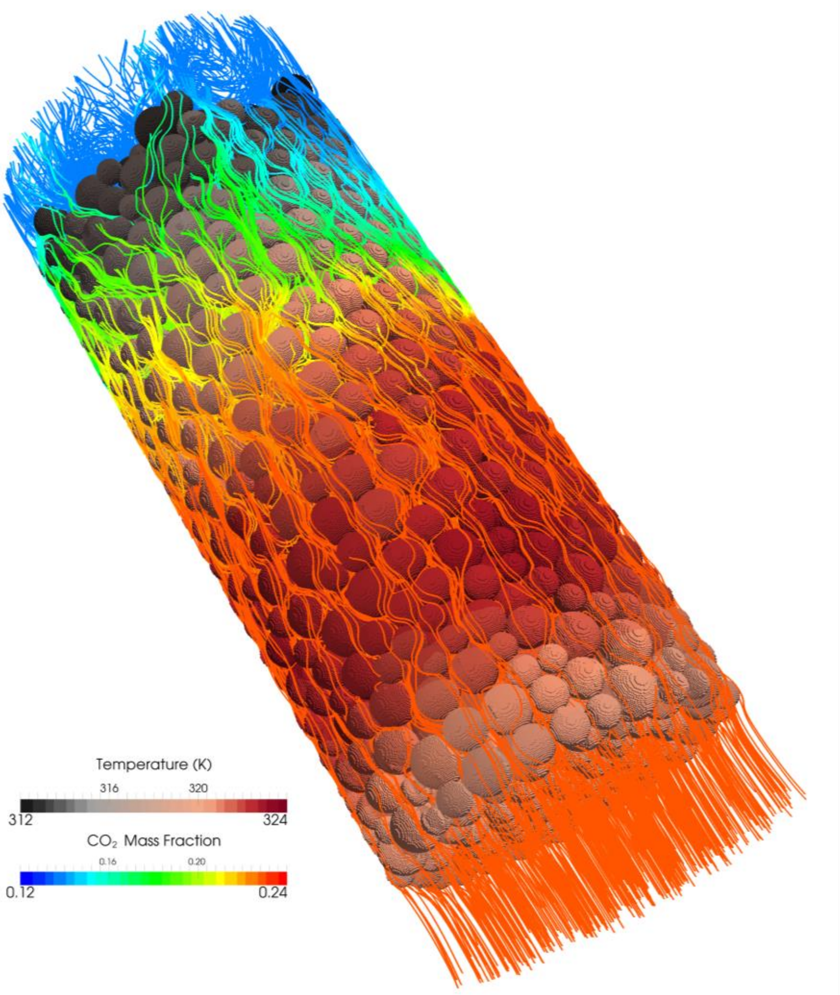
OpenFOAM® Basic Training

Tutorial Two



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).

sonicFoam – forwardStep

Simulation

Using sonicFoam solver, simulate 10 s of flow over a forward step.

Objectives

- Understand blockMesh
- Define vertices via coordinates as well as surfaces and volumes via vertices.

Post processing

Import your simulation into ParaView, and examine the mesh and the results in detail.

Step by step simulation

Copy tutorial

Copy the tutorial from the following folder to your working directory:

`~/OpenFOAM/OpenFOAM-2.3.0/tutorials/compressible/sonicFoam/`

`laminar/forwardStep`

0 directory

The file T includes the initial temperature values. Internal pressure and temperature fields are set to 1, and the initial velocity in the domain is set to zero except at the inlet boundary, where it is 3.

Note: As it can be seen, the p unit is the same as the pressure unit, because the sonicFoam is a compressible solver.

Note: Do not forget that, this example is a purely numeric example (you might have noticed this from pressure values).

constant directory

On checking thermophysicalProperties file, different properties of a compressible gas can be set:

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
thermoType
{
    type            hePsiThermo;
    mixture         pureMixture;
    transport       const;
    thermo          hConst;
    equationOfState perfectGas;
    specie          specie;
    energy          sensibleInternalEnergy;
}
mixture
{
    specie
    {
        nMoles          1;
        molWeight       11640.3;
    }
    thermodynamics
    {
        Cp              2.5;
        Hf              0;
    }
    transport
    {
        mu              0;
        Pr              1;
    }
}

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

In the `thermoType`, the models for calculating thermo physical properties of gas are set:

- `mixture`: Is the model which is used for the mixture, whether it is a pure mixture, a homogeneous mixture, a reacting mixture or ….

- `transport`: Defines the used transport model. In this example a constant value is used.

- `thermo`: It defines the method for calculating heat capacities, e.g. in this example constant heat capacities are used.

- `equationOfState`: Shows the relation which is used for the compressibility of gases. Here ideal gas model is applied by selecting `perfectGas`.

- `energy`: This key word lets the solver decide which type of energy equation it should solve, enthalpy or internal energy.

After defining the models for different thermo physical properties of gas, the constants and coefficients of each model are defined in the sub-dictionary `mixture`. E.g. `molWeight` shows the molecular weight of gas, `Cp` stands for heat capacity and `mu` for dynamic viscosity as `Pr` shows the Prandtl number.

By opening the turbulenceProperties appropriate turbulent mode can be set (in this case it is laminar):

```
simulationType   laminar;
```

There are two files in the polyMesh directory: blockMeshDict and boundary. In this example the mesh is not imported from other programs (e.g. GAMBIT). It will be created inside OpenFOAM®. For this purpose the blockMesh tool is used. The blockMesh reads the geometry and mesh properties from blockMeshDict file:

```
>nano blockMeshDict
```

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
convertToMeters 1;
vertices
(
    (0 0 -0.05)
    (0.6 0 -0.05)
    (0 0.2 -0.05)
    (0.6 0.2 -0.05)
    (3 0.2 -0.05)
    (0 1 -0.05)
    (0.6 1 -0.05)
    (3 1 -0.05)
    (0 0 0.05)
    (0.6 0 0.05)
    (0 0.2 0.05)
    (0.6 0.2 0.05)
    (3 0.2 0.05)
    (0 1 0.05)
    (0.6 1 0.05)
    (3 1 0.05)
);
blocks
(
    hex (0 1 3 2 8 9 11 10) (25 10 1) simpleGrading (1 1 1)
    hex (2 3 6 5 10 11 14 13) (25 40 1) simpleGrading (1 1 1)
    hex (3 4 7 6 11 12 15 14) (100 40 1) simpleGrading (1 1 1)
);
edges
(
);
boundary
(
    inlet
```

```
    {
        type patch;
        faces
        (
            (0 8 10 2)
            (2 10 13 5)
        );
    }
    outlet
    {
        type patch;
        faces
        (
            (4 7 15 12)
        );
    }
    bottom
    {
        type symmetryPlane;
        faces
        (
            (0 1 9 8)
        );
    }
    top
    {
        type symmetryPlane;
        faces
        (
            (5 13 14 6)
            (6 14 15 7)
        );
    }
    obstacle
    {
        type patch;
        faces
        (
            (1 3 11 9)
            (3 4 12 11)
        );
    }
);

mergePatchPairs
(
);
// ************************************************************************* //
```

As noted before units in OpenFOAM® are SI units. If the vertex coordinates differ from SI, they can be converted with the `convertToMeters` command. The number in the front of `convertToMeters` shows the constant, which should be multiplied with the dimensions to change them to meter (SI unit of length). For example:

```
convertToMeters    0.001
```

shows that the dimensions are in millimeter, and by multiplying them by 0.001 they are converted into meters.

In the `vertices` part, the coordinates of the geometry vertices are defined, the vertices are stored and numbered from zero, e.g. vertex `(0 0 -0.05)` is numbered zero, and vertex `(0.6 1 -0.05)` points to number 6.

In the `block` part, blocks are defined. The array of numbers in front each block shows the block building vertices, e.g. the first block is made of vertices `(0 1 3 2 8 9 11 10)`.

After each block the mesh is defined in every direction. e.g. `(25 10 1)` shows that this block is divided into:

- 25 parts in x direction
- 10 parts in y direction
- 1 part in z direction

As it was explained before, even for 2D simulations the mesh and geometry should be 3D, but with one cell in the direction, which is not going to be solved, e.g. here number of cells in z direction is one and it's because of that it's a 2D simulation in x-y plane.

The last part, `simpleGrading (1 1 1)` shows the size function.

In the `patches` part each boundary is defined by the vertices it is made of, and also its `type` and `name` are defined.

Note: For creating a face the vertices should be chosen clockwise when looking at the face from inside of the geometry.

Running simulation

Before running the simulation the mesh has to be created. In the previous step the mesh and the geometry data were set. For creating it the following command should be executed from case main directory (e.g. forwardStep):

```
>blockMesh
```

After that, the mesh is created in the polyMesh folder. For running the simulation, type the solver name form case directory and execute it:

```
>sonicFoam
```

Exporting simulation

The mesh is presented in the following way in ParaView, and you can easily see the three blocks, which were created.

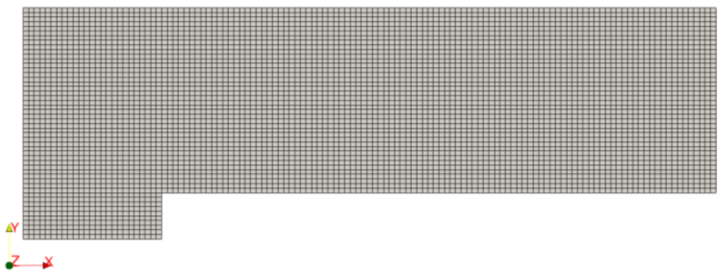


Figure 2.1 Mesh generated by blockMesh

Note: When a cut is created by default in ParaView, the program shows the mesh on that plane as a triangular mesh even if it is a hex mesh. In fact, ParaView changes the mesh to a triangular mesh for visualization, where every square is represented by two *triangles. For skipping this when creating a cut in ParaView in the Slice properties window, uncheck "Triangulate the Slice".*

The simulation results are as follows:

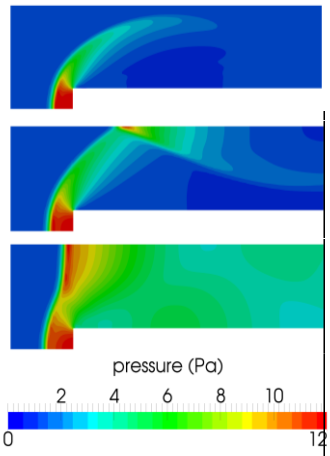
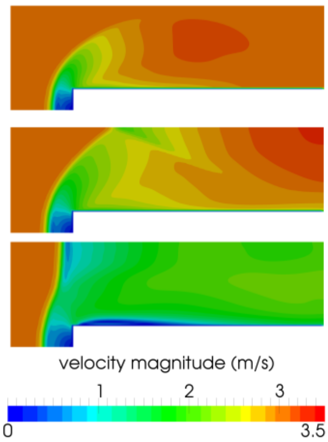
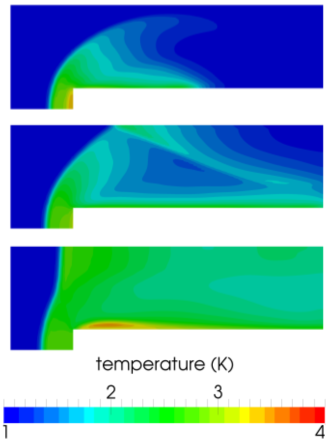
Time	Pressure	Velocity	Temperature
0.5 s			
1 s			
10 s			

Figure 2.2 Pressure, velocity and temperature contours at different time steps