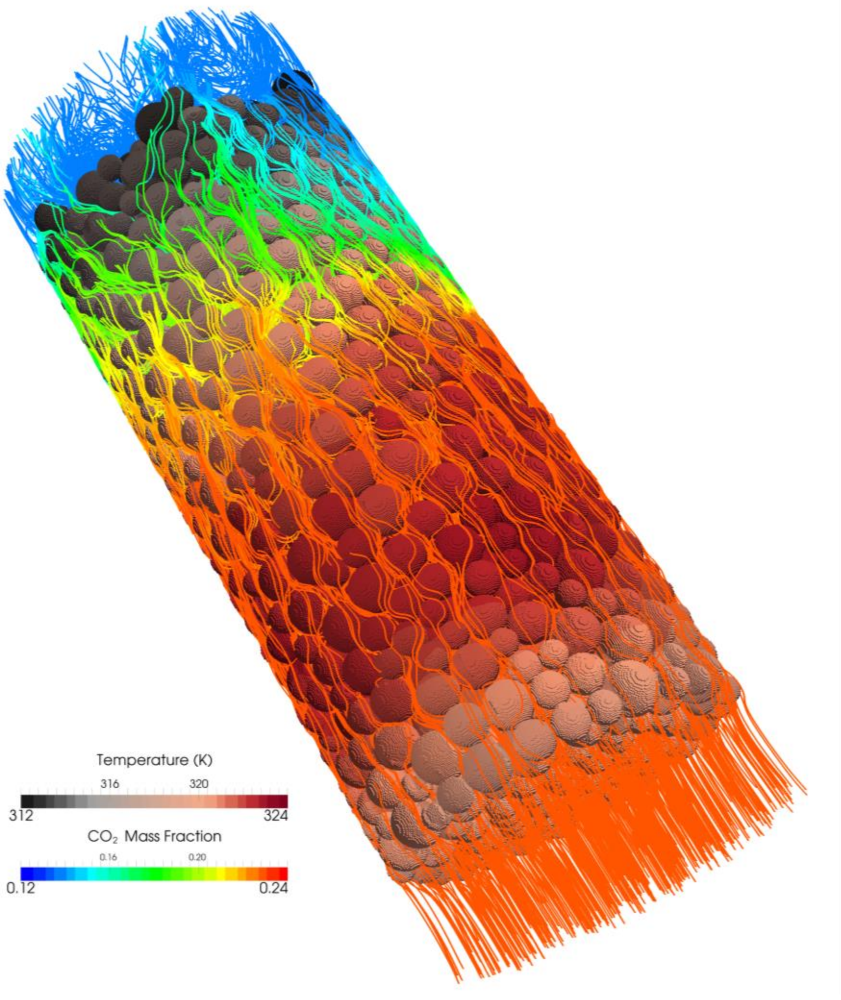
OpenFOAM® Basic Training

Tutorial Ten



3rd edition, Feb. 2015

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).

simpleFoam & scalarTransportFoam – TJunction

(Residence Time Distribution)

Simulation

Use the simpleFoam and scalarTransportFoam to simulate the flow through a square cross section T pipe and calculate RTD (Residence Time Distribution) for both inlets using a step function injection:

- Inlet and outlet cross sections: 1×1 m^2
- Gas in the system: air at ambient conditions
- Operating pressure: 10^5 Pa
- Inlet 1: 0.1 m/s
- Inlet 2: 0.2 m/s

Objectives

- Understanding RTD calculation using OpenFOAM®
- Using multiple solver for a simulation

Post processing

Plot the step response function and the RTD curve.

Step by step simulation

Copy tutorial

Copy the following tutorial to your working directory as a base case:

`~/OpenFOAM/OpenFOAM-2.3.0/tutorials/incompressible/simpleFoam /pitzDaily`

0 directory

Update p, U, nut, nuTilda, k and epsilon files with the new boundary conditions, e.g. U:

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

dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);

boundaryField
{
    inlet_one
    {
        type            fixedValue;
        value           uniform (0.1 0 0)
    }
    inlet_two
    {
        type            fixedValue;
        value           uniform (-0.2 0 0)
    }
    outlet
    {
        type            zeroGradient;
    }
    walls
    {
        type            fixedValue;
        value           uniform (0 0 0)

    }
}
// ************************************************************************* //
```

constant directory

Edit the blockMeshDict in the polyMesh directory as following for creating an appropriate geometry.

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
convertToMeters 1.0;

vertices
(
    (0 4 0) // 0
    (0 3 0) // 1
    (3 3 0) // 2
    (3 0 0) // 3
    (4 0 0) // 4
    (4 3 0) // 5
    (7 3 0) // 6
    (7 4 0) // 7
    (4 4 0) // 8
    (3 4 0) // 9
    (0 4 1) // 10
    (0 3 1) // 11
```

```
    (3 3 1) // 12
    (3 0 1) // 13
    (4 0 1) // 14
    (4 3 1) // 15
    (7 3 1) // 16
    (7 4 1) // 17
    (4 4 1) // 18
    (3 4 1) // 19

);
blocks
(
    hex (0 1 2 9 10 11 12 19) (10 30 10) simpleGrading (1 1 1)
    hex (9 2 5 8 19 12 15 18) (10 10 10) simpleGrading (1 1 1)
    hex (8 5 6 7 18 15 16 17) (10 30 10) simpleGrading (1 1 1)
    hex (2 3 4 5 12 13 14 15) (30 10 10) simpleGrading (1 1 1)
);
edges
(
);
patches
(
    patch inlet_one
    (
        (0 10 11 1)
    )
    patch inlet_two
    (
        (7 6 16 17)
    )
    patch outlet
    (
        (4 3 13 14)
    )
    wall walls
    (
        (0 1 2 9)
        (2 5 8 9)
        (5 6 7 8)
        (2 3 4 5)
        (10 19 12 11)
        (19 18 15 12)
        (18 17 16 15)
        (15 14 13 12)
        (0 9 19 10)
        (9 8 18 19)
        (8 7 17 18)
        (2 1 11 12)
        (3 2 12 13)
        (5 4 14 15)
        (6 5 15 16)
    )
);

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

Check RASProperties file for the turbulence model (`kEpsilon`).

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

RASModel        kEpsilon;

turbulence      on;

printCoeffs     on;


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

Running simulation

```
>blockMesh
```

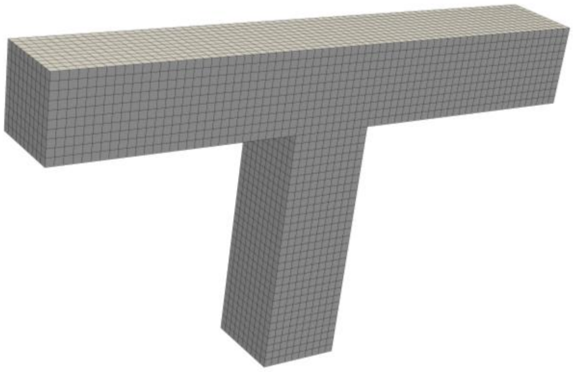


Figure 10.1 mesh created using blockMesh

```
>simpleFoam
```

Wait for simulation to converge. After convergency check the results to be sure the solution is converged (?).

```
>foamToVTK
```

The simulation results are as follows:

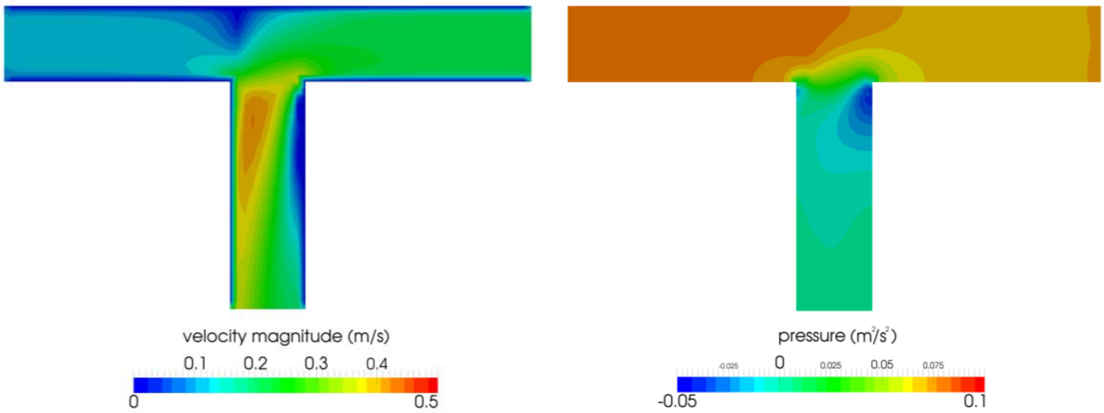


Figure 10.2 Simulation results after convergence (114 iterations)

RTD calculation

Copy tutorial

Copy following tutorial to your working directory:

```
~/OpenFOAM/OpenFOAM-2.3.0/tutorials/basic/scalarTransportFoam
/pitzDaily
```

0 directory

Delete the U file and replace it with the calculated velocity field from the first part of the tutorial (use the latest time step velocity field from previous part of simulation to calculate RTD for this geometry). There is no need to modify or change it. The solver will use this field to calculate the scalar transportation.

Update T (T will be used as an inert scalar in this simulation) file boundary conditions to match new simulation boundaries, to calculate RTD of the `inlet_one` set the `internalField` value to `0`, T value for `inlet_one` to `1.0` and T value for `inlet_two` to `0`.

constant directory

Replace the blockMeshDict file with the one from the first part of tutorial.

system directory

In the controlDict file change the `endTime` from `0.1` to `120` (approximately two times ideal resistance time) and also `deltaT` from `0.0001` to `0.1` (Courant number approximately 0.4).

Running simulation

```
>blockMesh
```

```
>scalarTransportFoam
```

```
>foamToVTK
```

Simulation results

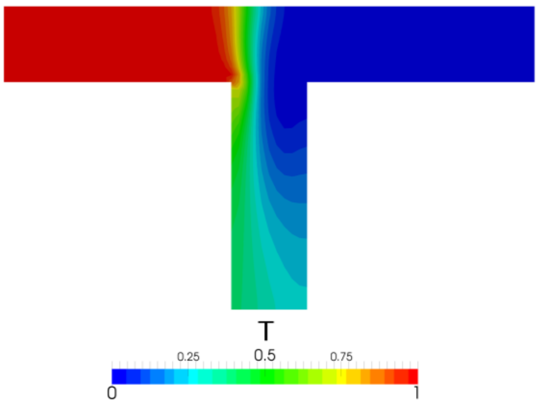


Figure 10.3 Contour plots scalar T at 120 s

Calculating RTD

To calculate RTD the average T value at the outlets should be calculated first. The “integrate variables function” of ParaView can be used for this purpose.

```
>foamToVTK
```

Load the outlet VTK file into paraview using following path:

File > Open > VTK > outlet > outlet_..vtk > OK > Apply

Select T from variables menu, and then integrate the variables on the outlet:

Filters > Data Analysis > Integrate Variables > Apply

The values given in the opened window are integrated values in this specific time step. By changing the time step values for different time steps are displayed. As mentioned before, the average value of the property is needed. Therefore, these values should be divided by outlet area to get average values (1m × 1m).

The same procedure should be followed for calculating RTD of `inlet_two`, except T value for `inlet_one` should be `0` and for `inlet_two` it should be `1.0`.

Calculating RTD

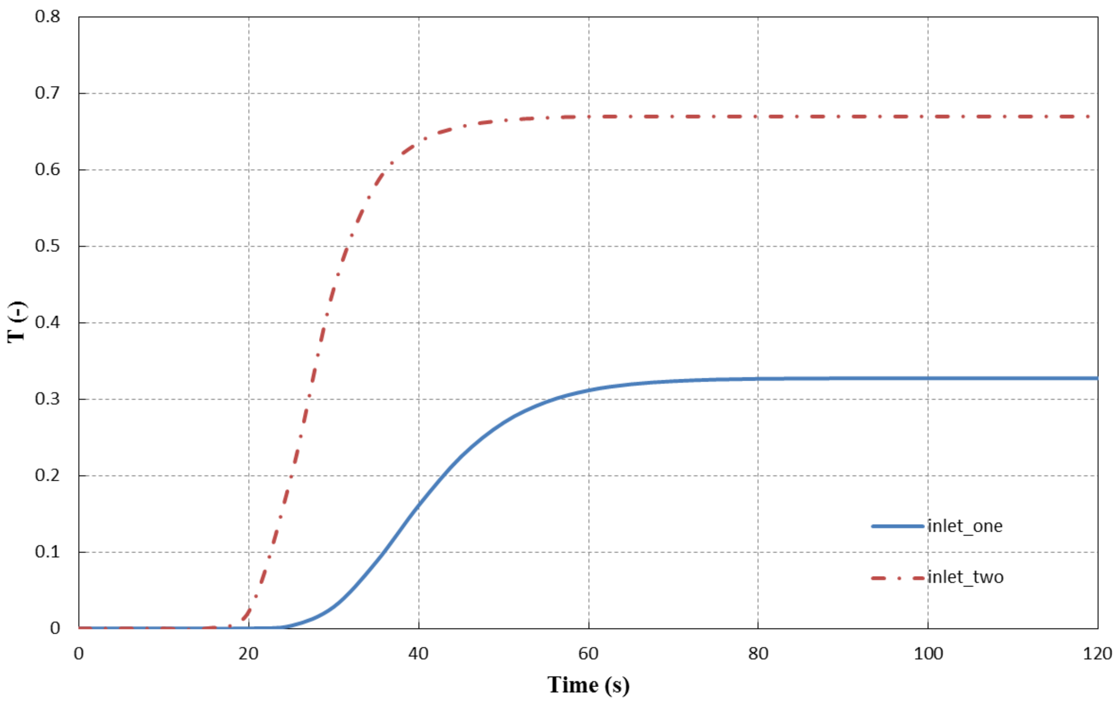


Figure 10.4 Average value of T on the outlet for two inlets versus time

The average value of T for each outlet approaches a certain constant value, which is the ratio of that scalar mass inlet to the whole mass inlet.

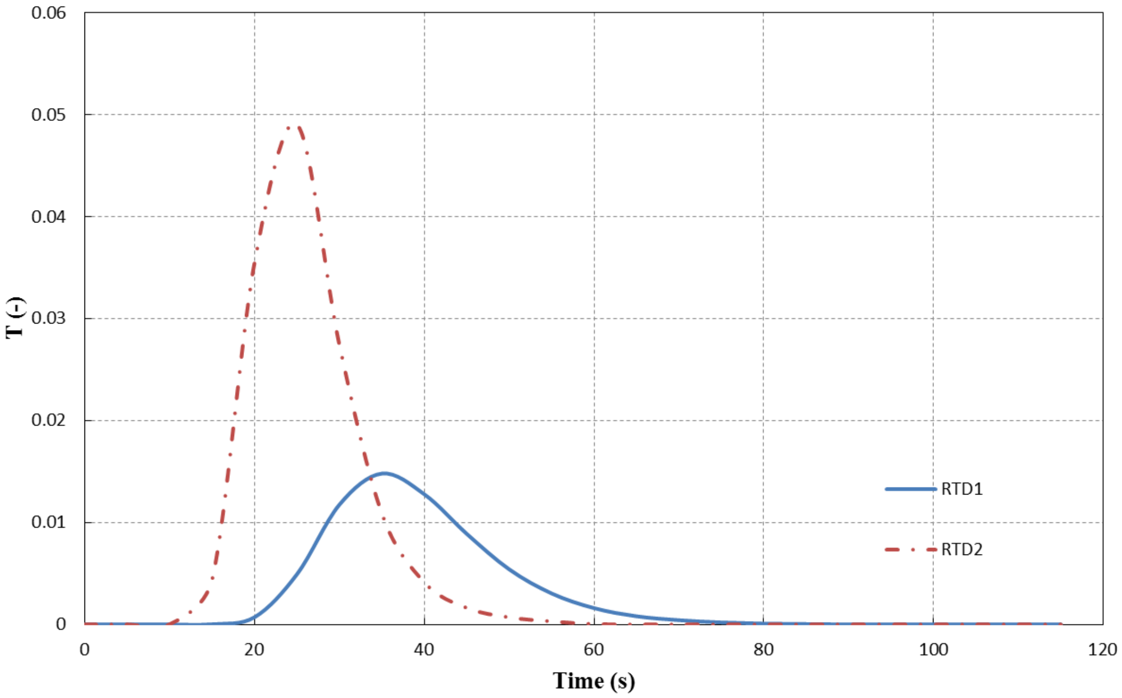


Figure 10.5 RTD of two inlets