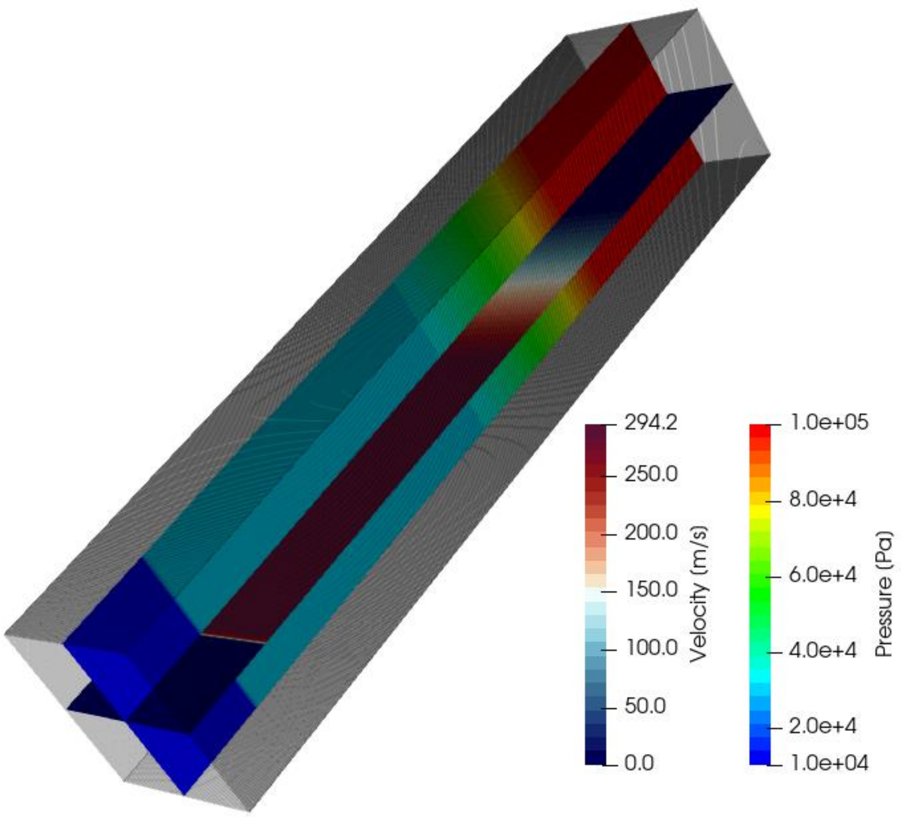
Tutorial Three

Patching Fields



Bahram Haddadi

ENGINEERING FLUID DYNAMICS

7th edition, March 2025

Contributors:

- Bahram Haddadi
- Christian Jordan
- Michael Harasek
- Clemens Gößnitzer
- Sylvia Zibuschka
- Yitong Chen
- Vikram Natarajan
- Jozsef Nagy

Technische Universität Wien
Institute of Chemical, Environmental
& Bioscience Engineering

Available from: www.fluiddynamics.at

Background

1. Initial and Boundary Conditions

Before running a numerical simulation, it is crucial to correctly define initial and boundary conditions for the problem. Poorly defined boundary conditions can lead to non-convergence or incorrect results. Understanding and applying these conditions properly ensures that the simulation behaves realistically and produces accurate and reliable results.

Initial conditions define the starting state of the simulation. These values are (usually) assigned to the center of every cell in the computational domain before the solver begins calculations. As the simulation progresses, the solver updates these values at each iteration based on the governing equations.

Key Points about Initial Conditions:

- **Importance:** They provide a reference point for the solver and influence how quickly the solution converges.
- **Transient problems:** The solution evolves over time from the specified initial state.
- **Steady-state problems:** The solver will iterate until a stable solution is found, regardless of the initial values, but the initial value might affect the speed of convergence and solution stability.

In OpenFOAM®, non-uniform initial conditions can be specified using the setFields utility, which allows defining non-uniform distributions of properties in the computational domain. This approach is especially useful when different regions of the domain require different starting values.

Boundary conditions define how the simulation domain interacts with its surroundings. They specify fixed values or behavior at the domain's boundaries, ensuring that the flow variables (such as velocity, pressure, or temperature) remain well-defined at these locations and represent the real physics at these boundaries. Types of Boundary Conditions:

- **Dirichlet Boundary Conditions (Fixed Value):** the variable (e.g., velocity or temperature) is assigned a fixed value at the boundary, e.g. specifying a constant temperature at a heated wall.
- **Neumann Boundary Conditions (Fixed Gradient):** instead of a fixed value, the gradient of the variable is specified at the boundary, e.g. a heat flux condition at a surface where the temperature gradient is controlled.
- **Mixed Boundary Conditions:** a combination of both Dirichlet and Neumann conditions, often used for heat transfer and fluid dynamics problems.
- **Periodic Boundary Conditions:** the solution at one boundary is linked to the opposite boundary, creating a repeating or cyclic condition useful for modeling infinite domains.
- **Symmetry**: used when a boundary behaves like a mirror, preventing flow normal to it.

In OpenFOAM®, boundary and initial conditions are defined in configuration files located in the 0 directory, where users can specify different types of conditions depending on the physical problem being modeled.

Setting appropriate boundary and initial conditions ensures that the simulation correctly represents the physical problem and achieves accurate, stable, and realistic results.

2. Courant-Friedrichs-Lewy (CFL) Condition

When performing numerical simulations, stability is a key concern. The Courant-Friedrichs-Lewy (CFL) condition is an essential mathematical criterion that ensures numerical schemes remain stable and that information propagates correctly within the computational domain.

Numerical solvers work by advancing the solution through a series of small time steps. However, for the solution to be physically meaningful, the information carried by waves or particles within the fluid must not move faster than the numerical grid can capture it. If this condition is violated, the simulation may become unstable, leading to numerical errors or divergence. To achieve this CFL condition is used which is mathematically expressed as:

$$Co = \frac{u\Delta t}{\Delta x} \leq 1$$

Where:

- u = Velocity magnitude in the considered direction
- Δt = Simulation time step size
- Δx = Mesh cell size in the corresponding direction

How the CFL Condition Affects Simulations:

- **Co > 1**: The simulation becomes unstable because the numerical scheme cannot properly capture the flow information moving between cells.
- **Co ≤ 1**: The information is correctly captured within the computational grid, ensuring stability and accuracy.

A finer mesh (smaller Δx) requires a smaller time step (Δt) to maintain stability, while higher velocity (larger u) also requires a smaller Δt to satisfy the CFL condition.

One of the most effective ways to determine a suitable Δt is to maintain the Courant number close to 1 (to have the biggest stable time step), using the maximum velocity in the domain and the smallest cell size. By using this approach, the solver will run efficiently while maintaining stability.

fluid Solver – shockTube

Tutorial outline

Use the “fluid” solver; simulate 0.007 s of flow inside a shock tube, with a mesh with 100, 1000 and 10000 cells in one dimension, for initial values 1 bar/0.1 bar and 10 bar/0.1 bar.

Objectives

- To understand the setFields utility
- Learn how to specify initial and boundary conditions
- Investigate effect of grid resolution.

Data processing

Import your simulation into ParaView, and compare results.

1. Pre-processing

1.1. Copying tutorial

Copy the tutorial from the following directory to your working directory

```
$FOAM_TUTORIALS/fluid/shockTube
```

1.2. Mesh and setting fields

Looking at the *blockMeshDict* file (in system directory), it is obvious that it is a 1D mesh, because of the number of mesh cells in y and z directions is one, and also in `boundary` section, plates vertical to these directions are defined as `empty`. The mesh density can be set in the `blocks` part by changing x direction mesh size (e.g. change it from `1000` to `100` or `10000`).

Another important file is *setFieldsDict* (in the system directory), which is used by the tool `setFields` for patching (assigning an amount to a region) in the simulation. For example, here a pressure of 10^5 Pa is set as the default value for the entire region (in the `defaultFieldValues`), then half of the region (from 0 to 5) is patched with a pressure of 10^4 Pa.

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

defaultFieldValues ( volVectorFieldValue U ( 0 0 0 ) volScalarFieldValue T
348.432 volScalarFieldValue p 100000 );

regions         ( boxToCell { box ( 0 -1 -1 ) ( 5 1 1 ) ; fieldValues (
volScalarFieldValue T 278.746 volScalarFieldValue p 10000 ) ; } );

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

In the `defaultFieldValues`, a value is assigned to the whole domain, for example here, the velocity has been set everywhere to zero, the temperature to 348.432 K, and the pressure to 100000 Pa. In the `regions`, a specific value is patched to a certain region of the domain. In this example the region is defined as a cube, by the coordinates of one of its diagonals in `boxToCell`.

After choosing the region, the new values are assigned to the parameters (e.g. temperature at 278.746 K and pressure at 10000 Pa).

2. Running simulation

First the mesh needs to be created:

```
>blockMesh
```

In order to assign the values which were set in the setFieldDict:

```
>setFields
```

Then run:

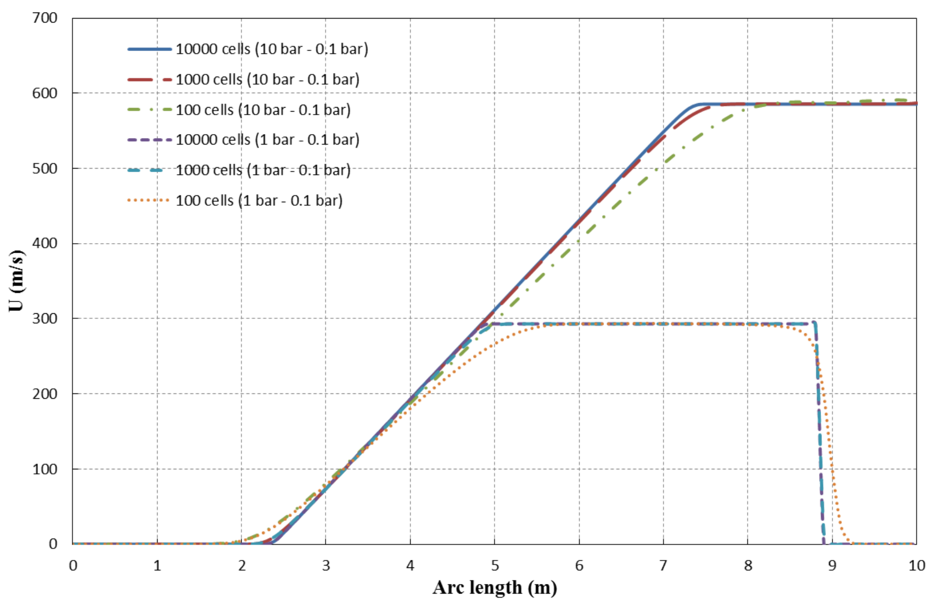
```
>foamRun -solver fluid
```

Note: Checking the information printed to the terminal (or log file) you can see how by decreasing the cell size (e.g. by increasing the number of cells) or by increasing the velocity (changing the pressure values) the Co number is increasing at a constant time step. In case the Co is getting bigger than one, the `deltaT` needs to be adopted accordingly to keep the Co below one.

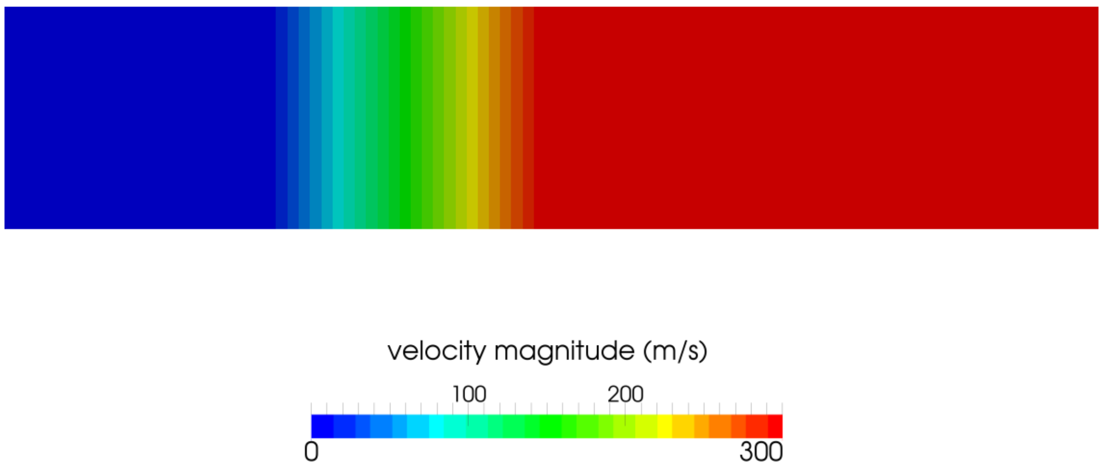
Note: After running setFields for the first time, the files in the 0 directory are overwritten. If the mesh is changed these files are not compatible with the new mesh and the simulation will fail. To solve this problem replace the files in the 0 directory with the files in the 0.orig or the files with suffix ".orig", e.g. p.orig in the 0 directory. In the OpenFOAM® files or directories with suffix ".orig" ("original") usually contain the backup files. If a command changes the original files these files can be replaced.

3. Post-processing

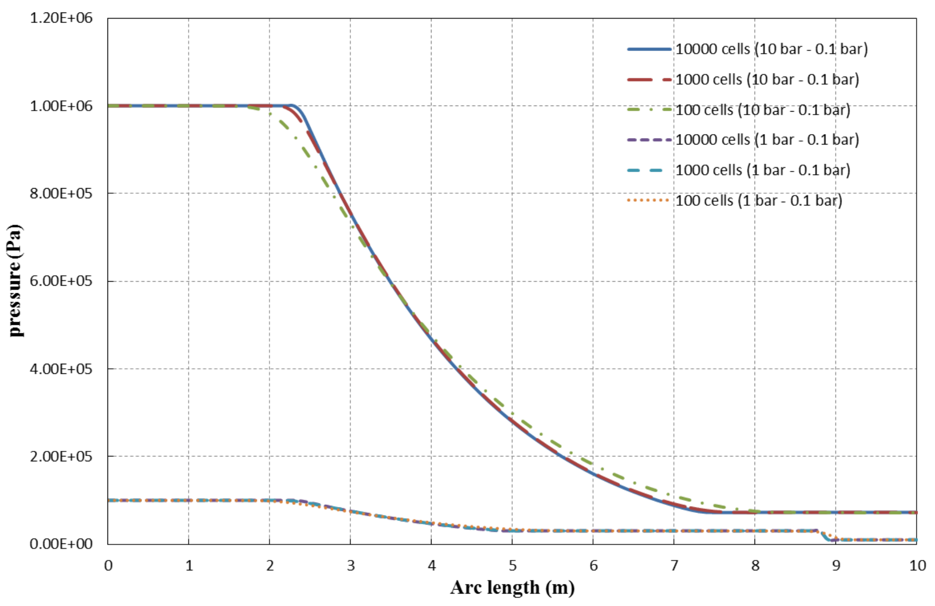
The simulation results are as follows:



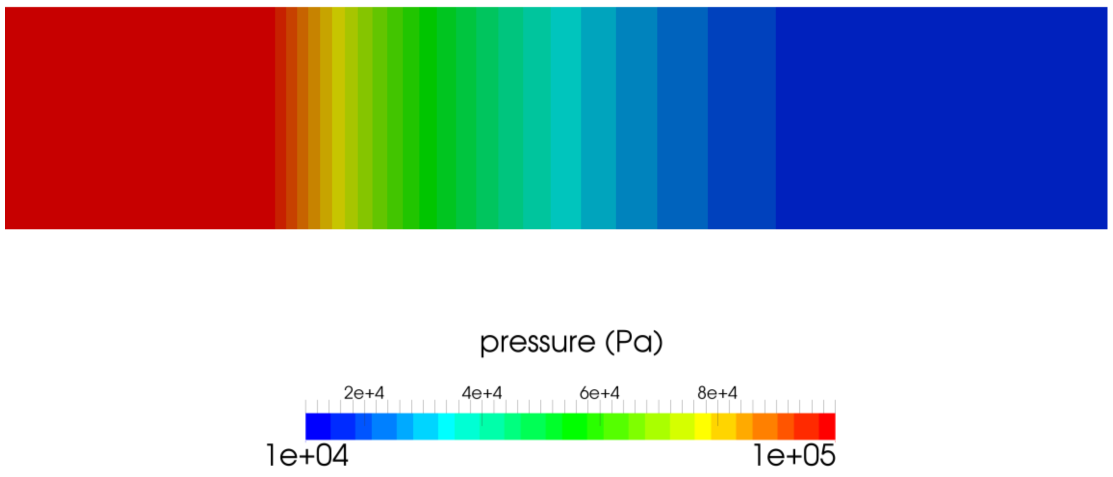
Velocities for different configurations along tube at t = 0.007 s



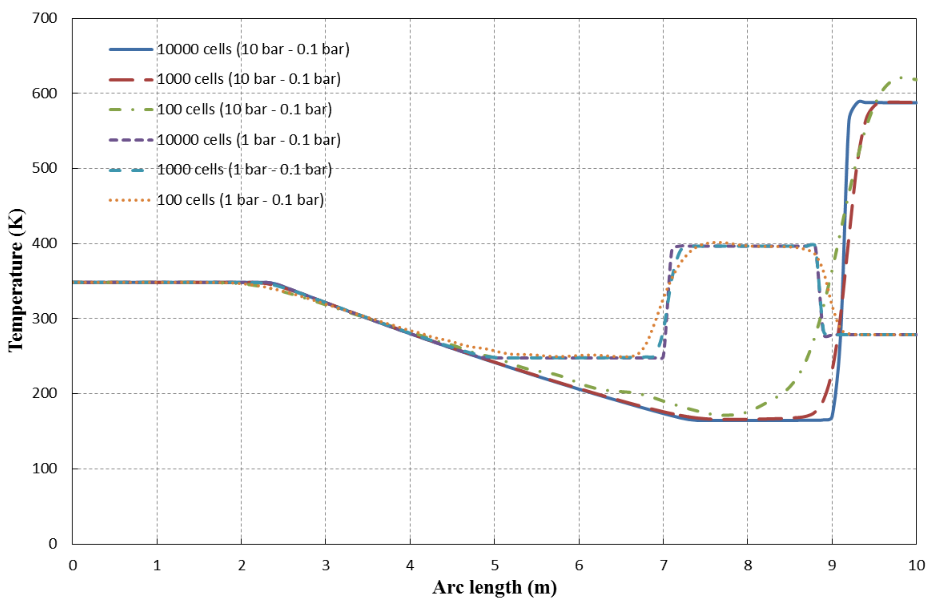
Velocity along tube axis for 10 bar/0.1bar and 10000 cells case at t = 0.007s



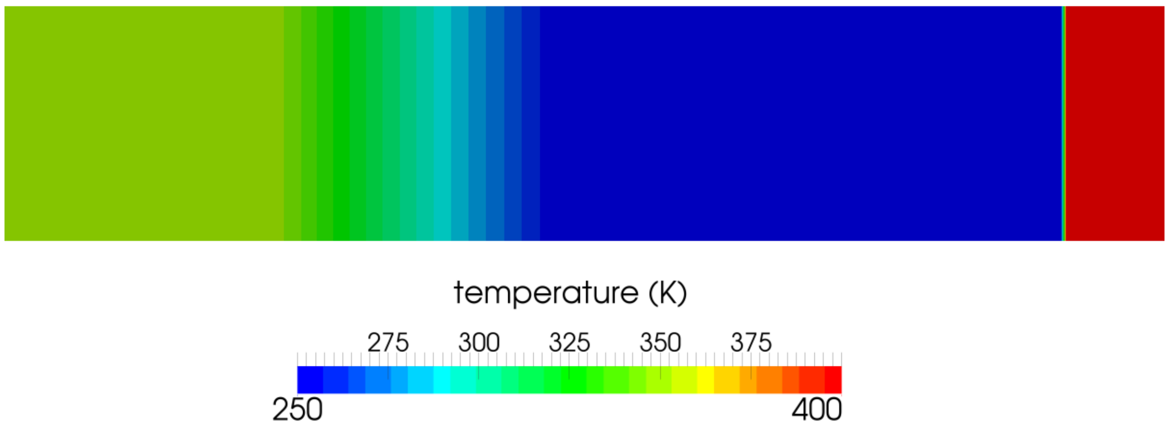
Pressures for different configurations along tube at t = 0.007 s



Pressure along tube axis for 10 bar/0.1bar and 10000 cells case at
t = 0.007s



Temperature for different configurations along tube at t = 0.007 s



Temperature along tube axis for 10 bar/0.1bar and 10000 cells case at t = 0.007 s