ddtFoam

F. Ettner, T. Sattelmayer
Lehrstuhl für Thermodynamik, Technische Universität München
July 17, 2013

1 What is ddtFoam?

ddtFoam is an OpenFOAM solver developed to simulate the deflagration-to-detonation transition.

Highlights:

- computationally efficient as both deflagration and detonation are described using a reaction progress variable
- tabulated chemistry (ignition delay times)
- sharp resolution of gasdynamic effects as convective fluxes are solved using the HLLC scheme
- applicable to unstructured grids
- works on relatively coarse grids

Scientific publications describing the code:

- F. Ettner: Effiziente numerische Simulation des Deflagrations-Detonations-Übergangs. *Ph.D. thesis (in German), Technische Universität München, 2013.*
- F. Ettner, K.G. Vollmer, T. Sattelmayer: Numerical assessment of the deflagration-to-detonation transition in mixtures with vertical concentration gradients. *Submitted to Journal of Hydrogen Energy*

The source code can be downloaded freely from `http://sourceforge.net/projects/ddtfoam/`. When using the code or any part of it in publications, please cite one of the publications mentioned above.

We do not provide any individual support on the code.

2 Installation

Prerequisite: A *working* installation of OpenFOAM version 2.1.1. Please follow the procedure and checks described on `http://www.openfoam.org`.

The solver has been developed and checked in an OpenSuSE 12.1 environment (kernel: Linux 3.1.10-1.19-desktop x86_64). The code might work on other Linux and other OpenFOAM versions as well. However, we have not checked this.

Download the code from `http://sourceforge.net/projects/ddtfoam/` and put it into an arbitrary directory (we recommend your local `OpenFOAM/user-2.1.1` directory).

To compile, run the `Allwmake` script in the main directory. This will compile all the libraries, boundary conditions, flux schemes and solvers included in ddtFoam.

3 Running the tutorials

3.1 Shock tube tutorial

This tutorial demonstrates the shock-capturing capability of the HLLC scheme in inert flow. The tutorial computes the flow in a one-dimensional shock tube. You can first examine the initial conditions, then run the code using the `Allrun` script in the `tutorials/shocktube` directory. When the run has completed, the `Allrun` script automatically samples the variables `(p T U rho)` from the result file. Use any plot program (gnuplot, matlab, paraview, ...) to visualize the result. The results at $t = 0.1$ms should look like this:

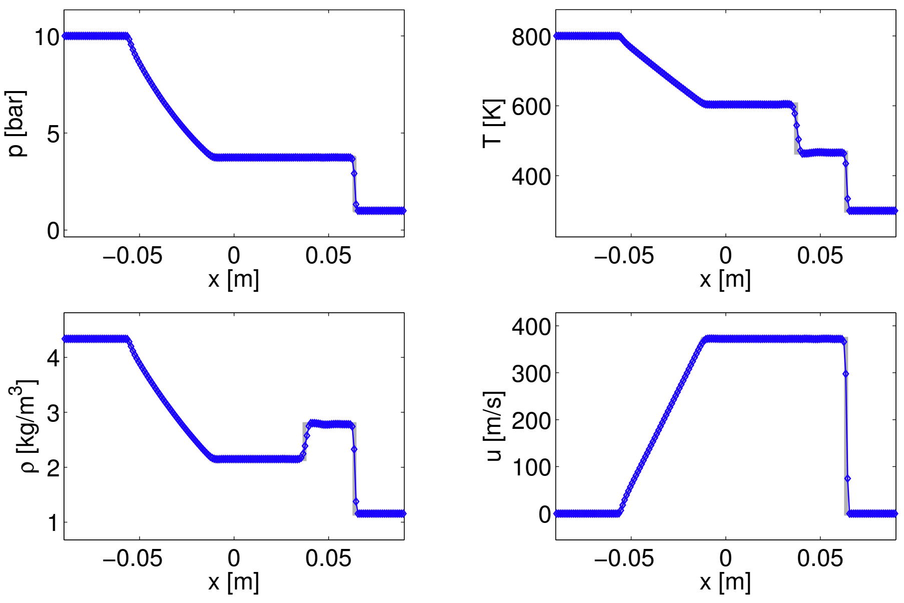

Abbildung 1: Result of the shock tube tutorial. The grey line represents the analytical solution. Take from [1].

Repeat the computation with your favourite OpenFOAM solver and compare the results.

3.2 DDT tutorial

In this tutorial, flame acceleration and the transition to detonation in a 2D channel is computed. The rectangular channel is equipped with some obstacles to promote flame acceleration. The initial hydrogen distribution varies in vertical direction. The initial flow velocity is zero everywhere. The flow is ignited by a small patch where the reaction progress variable $c = 1$ which represents the ignition kernel.

As the HLLC scheme is part of a density-based solver, it cannot be used in absolutely stagnant flow, where no coupling between density and pressure exists. Therefore, an additional pressure-based solver, *pddtFoam*, is provided to deal with low Mach number flow.

Go to the directory `tutorials/pddtFoam_Tutorial`, run the `Setup` script and examine the initial conditions. Start running the case by invoking

```
pddtFoam
```

Examine the result at $t = 8.0$ms using paraview. The temperature field shows that the flame has passed the first obstacle:

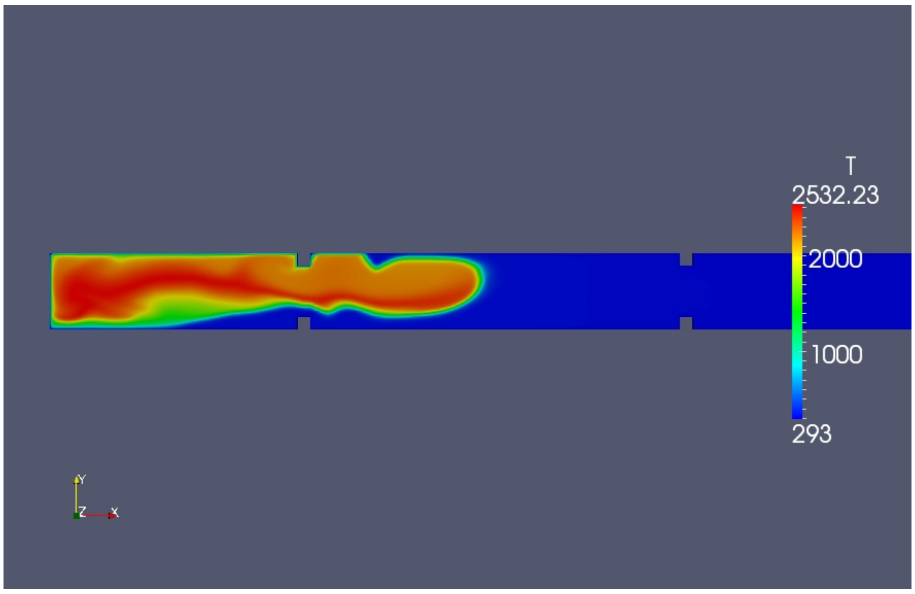

Abbildung 2: The temperature field at $t = 8.0$ms.

By now the combustion-induced flow is strong enought to switch to the density-based solver which will provide better shock-capturing in the later stage of the simulation, when compressible effects become dominant. Go to the directory `tutorials/ddtFoam_Tutorial` and run the `Setup` script. This will import the current values at $t = 8.0$ms from the `pddtFoam_Tutorial` directory. Run the case by invoking

```
ddtFoam
```

As this case requires several CPU hours, you might alternatively consider running it in parallel. Decompose the case into several domains as described on `http://openfoam.org/docs/user` and run it using

```
mpirun -np X ddtFoam -parallel
```

where `X` stands for the number of processors to run on (should be equal to the numbers of domains).

When examining the results, you will see that a DDT occurred somewhere between $t = 11.75$ms and $t = 12.00$ms.

You might try to rerun the case from $t = 11.75$ms on and save more transient files to examine the DDT process in detail.

Alternatively, you might also rerun the whole case on a finer grid and check if any differences occur.

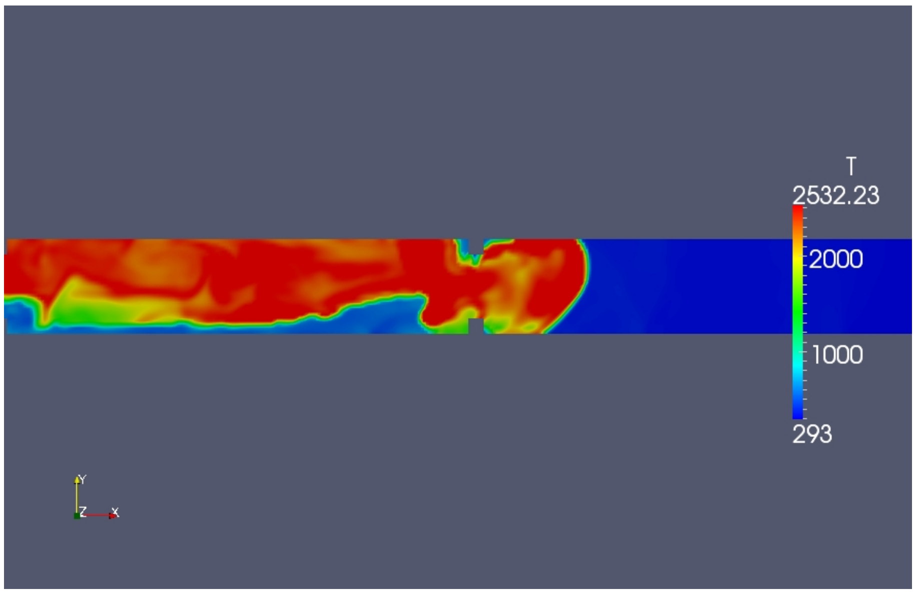

Abbildung 3: The temperature field after the occurrence of DDT at $t = 12.0$ms.

4 References

[1] F. Ettner: Effiziente numerische Simulation des Deflagrations-Detonations-Übergangs. *Ph.D. thesis (in German), Technische Universität München, 2013.*