University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: Mechanical and Structural Engineering and Materials Science / Machine Construction	Spring semester, 2015 Open / Restricted access
Writer: Jakob Trydal	…… (Writer's signature)
Faculty supervisor: Bjørn H. Hjertager External supervisor(s): Jørgen Osenbroch	
Thesis title: CFD analysis of temperature development due to flow restriction in pipeline	
Credits (ECTS): 30	
Key words: CFD, OpenFOAM, rhoCentralFoam, chtMultiRegionFoam, highly under-expanded jet, confined supersonic jet, Mach disk, heat transfer	Pages: 89 + enclosure: 66 Stavanger, …….. Date/year

CFD ANALYSIS OF TEMPERATURE DEVELOPMENT DUE TO FLOW RESTRICTION IN PIPELINE

Jakob Trydal

Supervisor
Professor Bjørn H. Hjertager

External Supervisor
Jørgen Osenbroch

Master's thesis 30 ECTS

Faculty of Science and Technology
Department of Mechanical and Structural Engineering and Materials Science, University of Stavanger
Stavanger June 2015

Abstract

If the pressure difference in a pipe, up- and downstream of a restriction orifice, is above a critical value, the flow becomes choked and an under-expanded jet occurs. For very large pressure differences, the gas experiences a significant temperature drop, downstream of the orifice. The cold gas causes the temperature in the pipe material to decrease and its material properties to change. In the oil and gas industry, this problem typically occurs during pressurization or depressurization of gas pipelines.

The temperature distribution in a 2” pipeline, due to a flow restriction, has been investigated, using the open-source CFD code OpenFOAM. The fluid and the pipe were simulated separately using a quasi-transient approach. The natural gas was modelled as pure methane.

The results show a minimum pipe temperature of 191,8 K (-81,2°C) compared to 177,1 K (-95,9°C) in Olga simulations performed by Aker Solutions. The upstream pipe is almost unaffected by the low temperatures, while approximately 0,5 m of the downstream pipe experiences temperatures below the minimum design temperature (-46°C). After approximately 40 seconds, the temperature in the entire pipe is above the limit. In Aker’s report, this time is estimated to 228 seconds.

A grid independence test was conducted and the temperature at the fluid surface was found to vary $\pm$1%, when the number of cells was increased by 40%.

In the under-expanded jets, the locations and diameters of the normal shocks, known as Mach disks, are in agreement with experimental data, except for the cases with nozzle pressure ratios (NPR) above 80. The expressions that describe the Mach disk location and diameter are developed for free jets. In the cases with NPR above 80, the wall heavily influences the structure of the jets and thus the jets are no longer free.

Based on the validation against experimental data, as well as validation of the solver’s accuracy, it is assumed that the obtained results give a satisfactory estimate of the actual temperature distribution in the pipe.

Contents

List of Figures

Appendix

List of Tables

Nomenclature

Abbreviations

CFD	Computational Fluid Dynamics
CNG	Compressed Natural Gas
DIC	Diagonal Incomplete-Cholesky
DNS	Direct Numerical Simulation
FPSO	Floating Production and Storage Offloading
FVM	Finite Volume Method
GUI	Graphical User Interface
LES	Large Eddie Simulation
NPR	Nozzle Pressure Ratio
PCG	Preconditioned Conjugate Gradient
RANS	Reynolds-Averaged Navier-Stokes
SST	Shear Stress Transport
STL	Standard Tessellation Language
TVD	Total Variation Diminishing

Latin letters

A	Cross sectional area	[m^2]
A_n	Cross sectional area of orifice	[m^2]
A_s	Surface area with heat transfer	[m^2]
A_{su}	Constant in Sutherland's equation	
B	Constant	
c_p	Specific heat capacity at constant pressure	[$Jkg^{-1}K^{-1}$]
c_v	Specific heat capacity at constant volume	[$Jm^{-3}K^{-1}$]
C_μ	Constant	
Co	Courant number	
$\left(\frac{\partial \widehat{\boldsymbol{u}}}{\partial t}\right)_I$	Time derivative due to inviscid fluxes	
$(\frac{\partial(\rho \boldsymbol{u})}{\partial t})_V$	Time derivative due to diffusion	
$\frac{D}{Dt}$	Material derivative	
d_{hyd}	Hydraulic diameter	[m]
d_n	Nozzle diameter	[m]
$d_{12"}$	Diameter of 12" pipe	[m]
$div()$	Divergence operator ∇	
E	Energy	[Jm^3kg^{-1}]

$\hat{E}$	Total energy density	[J]
F_1	Blending function	
$\mathrm{grad}()$	Gradient operator	
h	Enthalpy	[J]
h_c	Convective heat transfer coefficient	[$Wm^{-2}K^{-1}$]
H_f	Enthalpy of fusion	[Jkg^{-1}]
i	Internal energy	[J]
I	Turbulent intensity	
k	Turbulent kinetic energy	[m^2s^{-2}]
k_c	Material conductivity	[$Wm^{-1}K^{-1}$]
l_{turb}	Turbulent length scale	[m]
l_T	Characteristic length scale of turbulence	[m]
$L_{12"}$	Length of 12" pipe upstream	[m]
m_{gas}	Mass of stagnant gas	[kg]
$\dot{m}_{gas}$	Choked mass flow	[kgs^{-1}]
n	Number of moles	
p	Pressure	[Pa]
p_c	Critical pressure	[Pa]
P_k	Rate of production of turbulent kinetic energy	
P_0	Pressure upstream of orifice	[Pa]
P_∞	Pressure downstream of orifice	[Pa]
$\dot{Q}_{cond}$	Heat flux due to conduction	[W]
$\dot{Q}_{conv}$	Heat flux due to convection	[W]
r	Radial coordinate	[m]
R	Universal gas constant	[$JK^{-1}mole^{-1}$]
S	Mean rate of strain	[s^{-1}]
S_i	Source term internal energy	
S_{ij}	Mean strain rate tensor	[s^{-1}]
S_{Mx}, S_{Mz}, S_{Mz}	Body forces in x-, y- and z-direction	[N]
Δt	Time step length	[s]
T	Temperature	[K]
T_c	Critical temperature	[K]
T_s	Surface temperature	[K]
T_{su}	Sutherland's constant	
$\mathbf{T}_{visc}$	Viscous stress tensor	[Nm^{-2}]
$T_{12"}$	Temperature in 12" pipe upstream	[K]
T_0	Reference temperature Sutherland's equation	[K]
T_∞	Temperature in fluid far from surface	[K]
u	Velocity in x-direction	[ms^{-1}]
u_l	Local velocity	[ms^{-1}]
u_{mean}	Mean flow velocity	[ms^{-1}]

u_τ	Shear velocity	[ms^{-1}]
u_T	Characteristic velocity of turbulence	[ms^{-1}]
$\boldsymbol{u}$	Velocity vector	[ms^{-1}]
$\widehat{\boldsymbol{u}}$	Inviscid momentum density	[kgm^{-2}s^{-1}]
$\overline{u_i'\varphi'}$	Result of Reynolds decomposition in scalar transport equation	
u^+	Dimensionless velocity	
U	Flow velocity	[ms^{-1}]
v	Velocity in y-direction	[ms^{-1}]
V	Volume	[m^3]
V_{gas}	Volume of upstream stagnant gas	[m^3]
V_m	Molar volume (volume of 1 mole gas)	[m^3]
w	Velocity in z-direction	ms^{-1}
Δx	Cell length	[m]
x	Coordinate	[m]
x_M	Distance to Mach disk	[m]
y	Coordinate	[m]
y_p	Distance from wall to center of first cell	[m]
y^+	Dimensionless wall coordinate	
z	Coordinate	[m]

Greek letters

α	Thermal diffusivity	[m^{2}s^{-1}]
α_1	Constant	
β	Constant	
β^*	Constant	
γ	Ratio of specific heats $c_p/\ c_v$	
ε	Dissipation of turbulent kinetic energy	[m^{2}s^{-3}]
κ	Von Karmas constant	
μ	Dynamic viscosity	[Pa·s]
μ_T	Dynamic eddy viscosity	[Pa·s]
μ_0	Dynamic viscosity at reference temperature T_0	[Pa·s]
ν	Kinematic viscosity	[m^{2}s^{-1}]
ν_t	Kinematic eddy viscosity	[m^{2}s^{-1}]
ρ	Density	[kgm^{-3}]
ρ_0	Density upstream of orifice	[kgm^{-3}]
σ_k	Constant	
σ_ω	Constant	
$\sigma_{\omega 2}$	Constant	
τ_{ij}	Reynold stress tensor	[Nm^{-2}]

τ_w	Wall shear stress	[Nm^{-2}]
$\tau_{xx}, \tau_{yx}, \tau_{zx},$ $\tau_{xy}, \tau_{yy}, \tau_{zy},$ $\tau_{xz}, \tau_{yz}, \tau_{zz}$	Viscous stress components	[Nm^{-2}]
$\varphi'(t)$	Fluctuating value Reynolds decomposition	
Φ	Steady mean value Reynolds decomposition	
Φ_d	Dissipation function	
Φ_s	Scalar variable	
ω	Specific rate of dissipation	[s^{-1}]
ω_a	Acentric factor	

Acknowledgements

This thesis is a part of the master's program in Mechanical and Structural Engineering and Materials Science at the University of Stavanger, with specialization Machine Constructions.

Working with this thesis has been both challenging and interesting. From countless hours of troubleshooting and testing, I have gained valuable experience with the CFD code OpenFOAM, which can be useful in my future career. I have also obtained a better understanding of fluid dynamics in general and supersonic flow phenomenon in particular.

I would like to thank my supervisor at the University of Stavanger, Professor Bjørn H. Hjertager, for helpful discussions and input during my work with this thesis. I am also very grateful for the help and input from my external supervisor Mr. Jørgen Osenbroch. In addition, I would like to thank Mr. Knut Erik Giljarhus for advice regarding solvers and simulation set-up in OpenFOAM.

Finally, I would like to thank my wife, for her support and patience during my work with this thesis.

Stavanger, June 2015

Jakob Trydal

1. Introduction

This chapter contains a description of the background for the thesis. Previous work, which is relevant for the topic, is presented and the objective of the thesis is explained.

1.1. Background

Consider a gas flowing from a high- to a low-pressure pipe, through a restriction orifice. If the pressure difference between the pipes is larger than a critical pressure, the flow becomes choked, and the gas reaches sonic velocity through the orifice. This is also known as critical flow [1]. The rapid increase in flow speed and decrease in pressure, just after the orifice, results in a significant temperature drop in the downstream gas [2], [12].

The cold gas also affects the temperature in the pipe material, causing it to decrease. In the oil and gas industry, this problem typically occurs during pressurization of gas pipelines or during depressurization of a pipe segment, due to for example a platform blowdown. The decreased temperature in the pipe material can cause material properties to change, significantly reducing for example strength and toughness. It is therefore important to investigate this effect and make sure that the temperature in the pipe material is within specified limits.

The specific problem described in this thesis is the pressurization of a pipe segment connected to the gas injection compressor at the Skarv field. Skarv is a Floating Production, Storage and Offloading (FPSO) platform, operated by BP. It is located at Haltenbanken approximately 210 km west of Sandnessjøen.

The thesis is written in collaboration with Aker Solutions, which is one of the leading oil service companies within the oil and gas industry.

1.2. Problem Description

The pipe segment in question is shown in Figure 1-1. It consists of a 12” pipe, with a 2” bypass pipe. The 2” pipe is used to equalize the pressure during pressurization, before opening the 12” valve. A restriction orifice is installed in the 2” pipe, to limit the flow rate through the pipe. It has an inner diameter of 9,08 mm and a length of 45 mm.

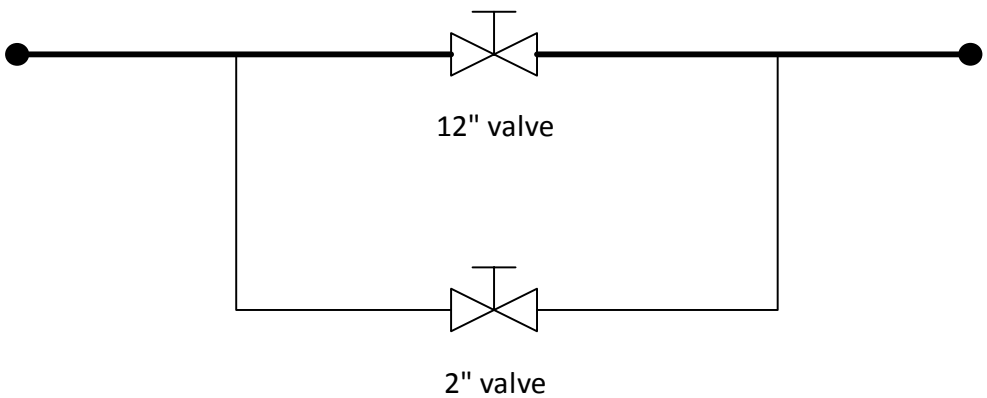

Figure 1-1 12" Pipe segment with 2" bypass pipe

Since the flow during pressurization goes through the 2” bypass pipe, the situation can be modelled as shown in Figure 1-2.

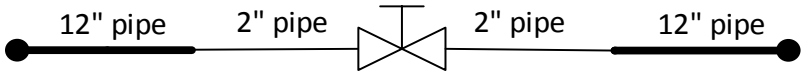

Figure 1-2 Flow at pressurization

Prior to pressurization, the gas in the 12” pipe upstream of the valve is stagnant. It is assumed that the pipe segment containing stagnant gas has a length of 25,1 m, according to Figure 1-3. This region is heat traced, and the operating conditions states that the pipe should be heated to 50°C to avoid too low temperatures in the downstream pipe.

Aker Solutions have analyzed the temperature in the downstream pipe, using a simulation tool called Olga. An illustration of the simulation setup is shown in Figure 1-3 [3].

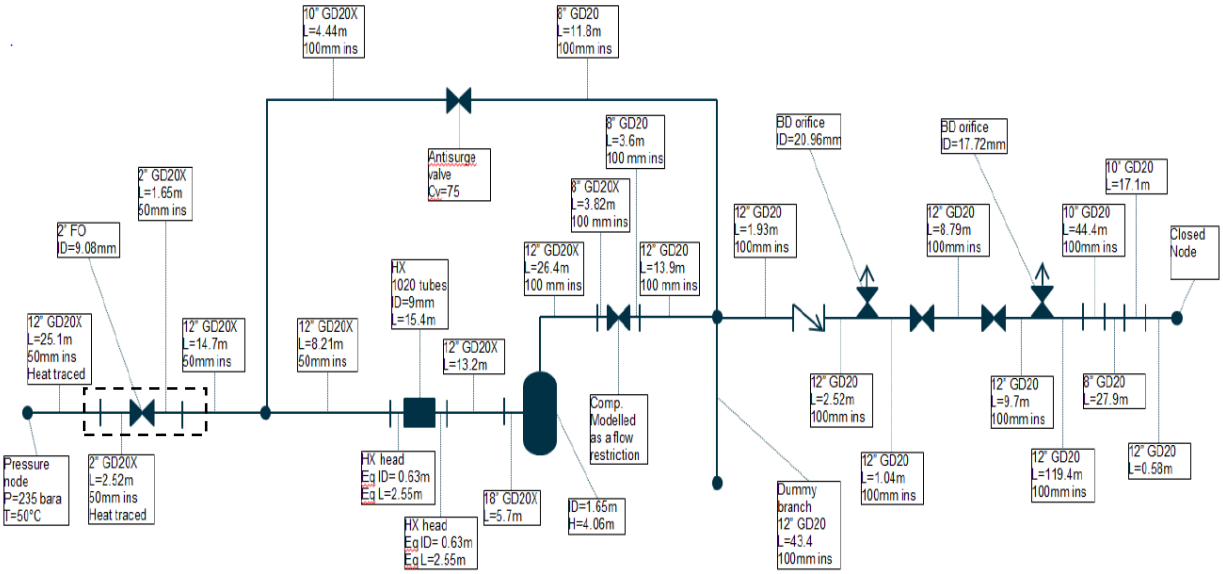

Figure 1-3 Olga simulation model [3]

We will not go into details about this simulation model, but rather summarize the results. The objective of Aker’s work was to check the consequences of insufficient heat tracing, resulting in upstream gas temperatures below 50°C.

The lowest ambient temperature considered in their report is -13°C. When the heat tracing is switched completely off, the temperature in the pipe material, as well as the temperature of the upstream gas, is assumed to be equal to the ambient temperature. Aker performed seven different simulations, with upstream gas temperatures ranging from -13°C to 50°C, listed in Table 1-1. These are referred to as case 1a to 1g.

If the system is pressurized immediately after a blowdown, the temperature in the pipes can be lower than the ambient temperature. According to Aker’s report, the minimum temperature in the downstream pipe after a blowdown is -19°C. The initial temperature in

the downstream pipe was changed to -19°C and all seven cases listed in Table 1-1 was rerun. These cases are known as case 2a to 2g.

Table 1-1 Flow conditions case 1a to 1g

Upstream conditions		Downstream conditions		Graph color
T [°C]	P [bar]	T [°C]	P [bar]	
-13°C	235	-13°C	1	Blue
0°C	235	-13°C	1	Black
10°C	235	-13°C	1	Red
20°C	235	-13°C	1	Green
30°C	235	-13°C	1	Brown
40°C	235	-13°C	1	Pink
50°C	235	-13°C	1	Orange

Figure 1-4 shows the inner wall surface temperature, measured in the 2" downstream pipe, as a function of time for case 1a to 1g. The equivalent plot for case 2a to 2g is shown in Figure 1-5.

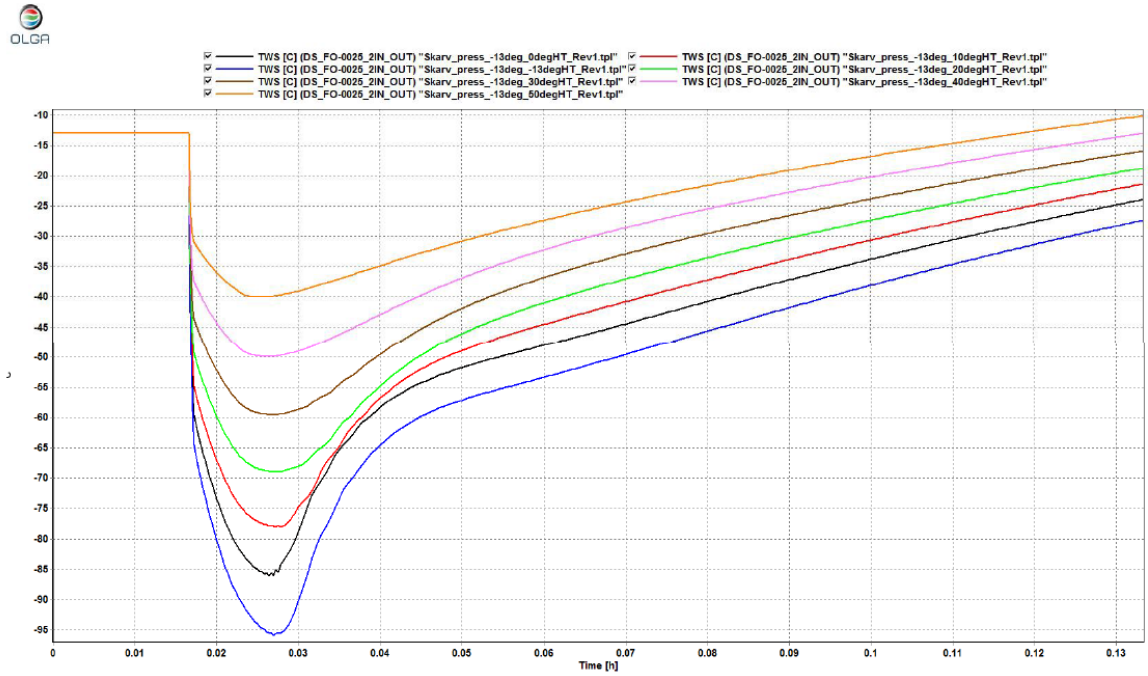

Figure 1-4 Inner wall surface temperature in 2" downstream pipe case 1a to 1g [3]

The gas located upstream of the 25,1 m stagnant segment is assumed to have a constant temperature of 50°C. When this gas reaches the orifice, and the backpressure in the system rises, we see that the temperature increases. The lowest temperature occurs when the downstream pressure is approximately 5 bar.

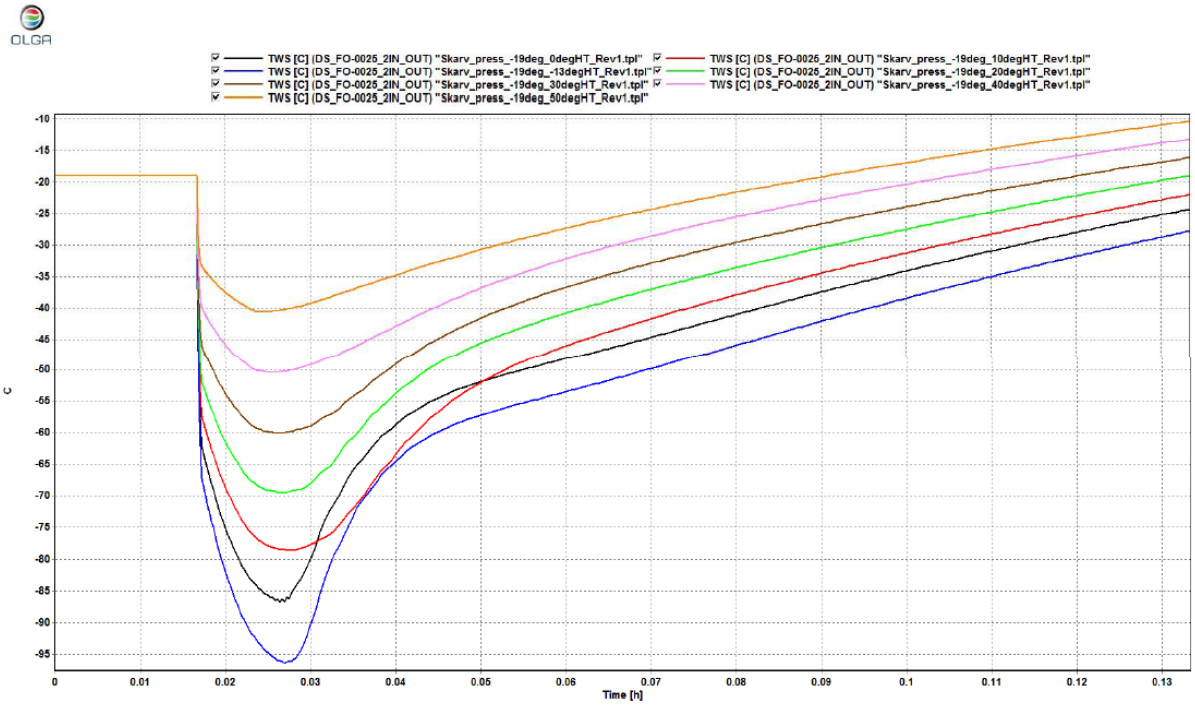

Figure 1-5 Inner wall surface temperature in 2" downstream pipe case 2a to 2g [3].

In addition to the inner wall surface temperature, the minimum gas temperature in the 2" downstream pipe was recorded for both case 1 and 2. The results are shown in Figure 1-6 and Figure 1-7.

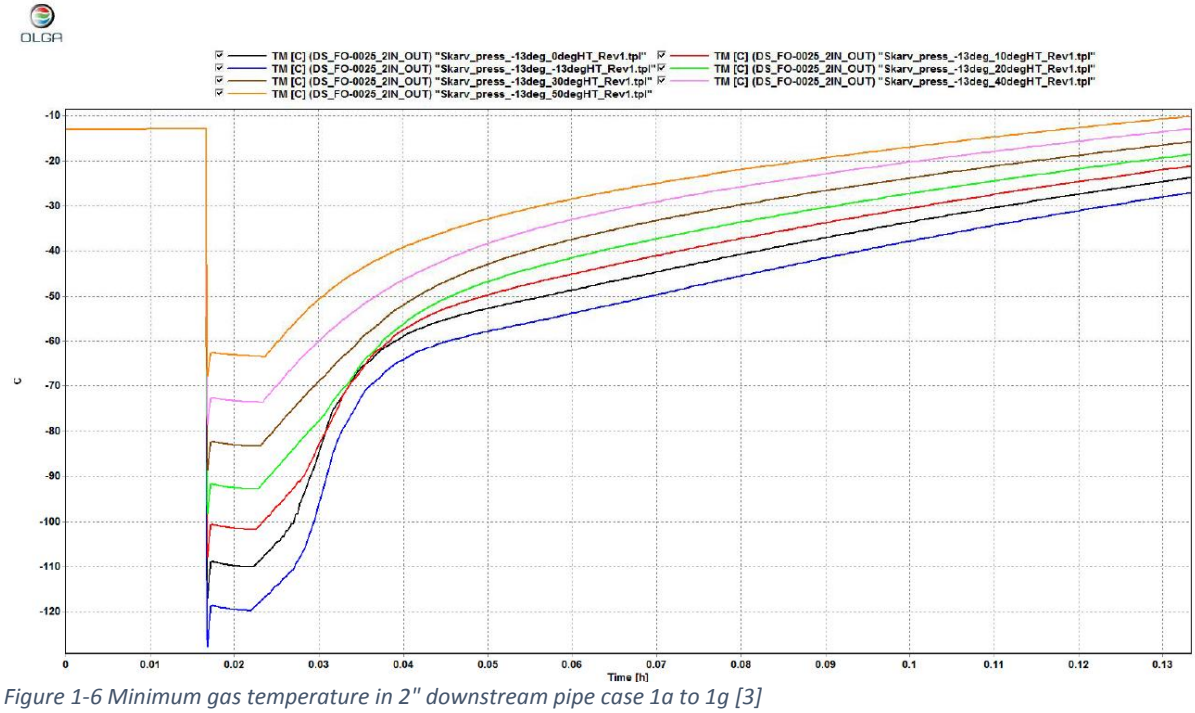

Figure 1-6 Minimum gas temperature in 2" downstream pipe case 1a to 1g [3]

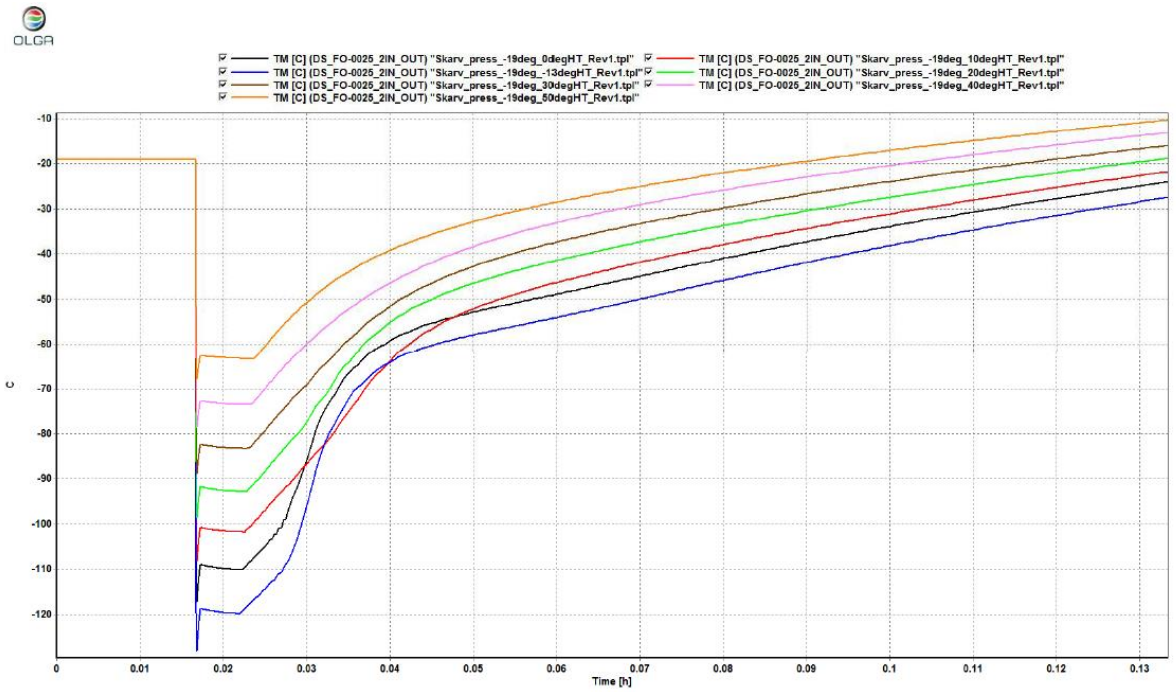

Figure 1-7 Minimum gas temperature 2" downstream pipe case 2a to 2g [3]

Table 1-2 gives an overview of Aker's results in tabulated form. It is evident from the results that the 2" pipe is much more affected by the cold gas, than the 12" pipe further downstream. According to Aker, this is due to the higher velocity and slightly lower gas temperature in this region, and the fact that the 2" pipe has much less steel mass per unit length than the 12" pipe.

Table 1-2 Tabulated results for all cases [3]

Case	Initial temp ds orifice [°C]	Initial temp us orifice [°C]	Minimum gas temp 2" segment ds orifice [°C]	Minimum inner wall surface temp 2" segment ds orifice [°C]	Minimum mean wall temp 2" segment ds orifice [°C]	Pressure at min inner surface temp 2" section ds orifice [bara]	Minimum gas temp 12" segment ds orifice [°C]	Minimum inner wall surface temp 12" segment ds orifice [°C]	Minimum mean wall temp 12" segment ds orifice [°C]
1a	-13	-13	-127.7	-95.9	-91.4	5.9	-120.1	-31.3	-25.9
1b	-13	0	-116.9	-86.1	-81.6	5.7	-106.6	-29.6	-24.1
1c	-13	10	-107.8	-78.1	-75.1	6.2	-95.5	-27.9	-22.9
1d	-13	20	-98.3	-69.0	-67.1	6.0	-83.8	-25.9	-21.6
1e	-13	30	-88.5	-59.6	-58.0	5.6	-71.2	-23.9	-20.1
1f	-13	40	-78.4	-49.9	-48.6	5.5	-57.4	-21.4	-18.5
1g	-13	50	-67.8	-40.0	-38.9	5.1	-43.2	-18.7	-16.8
2a	-19	-13	-128.1	-96.5	-92.1	5.9	-120.3	-36.1	-29.8
2b	-19	0	-117.3	-86.7	-82.7	5.7	-106.9	-34.4	-28.1
2c	-19	10	-108.2	-78.5	-76.7	6.2	-95.7	-32.7	-27.4
3d	-19	20	-98.6	-69.4	-67.5	6.0	-84.0	-30.9	-25.7
2e	-19	30	-88.8	-60.0	-58.5	5.7	-71.5	-28.8	-24.4
2f	-19	40	-78.6	-50.4	-49.1	5.2	-57.7	-26.3	-22.9
2g	-19	50	-67.9	-40.6	-39.4	4.9	-43.5	-23.6	-21.4

The minimum design temperature for the pipe material is -46°C. The design pressure at this temperature is 425,5 bar. Aker's report points out that even though the temperature in the 2" pipe drops to -96,5°C in the worst case (2a), this happens at a very low pressure (5,9 bar) compared to the design pressure. However, the report does not conclude whether this low temperature is acceptable or not.

Olga is a 1D-simulation tool, and does not capture all the details of the flow. By using Computational Fluid Dynamics (CFD) to analyze the problem, more details about the temperature distribution in the pipe and the gas, as well as the pressure, velocity and other flow variables, can be obtained. It is especially interesting to further investigate how much of the 2" pipe that experiences temperatures below the design temperature as well as how far the low temperature propagates up- and downstream of the orifice.

Since the lowest temperatures occur in the 2" pipe, this is the most critical part of the system. In this thesis, we will therefore limit ourselves to conducting a CFD analysis of the flow in the 2" bypass pipe (inside the dotted lines in Figure 1-3).

As already mentioned, the temperature in the pipe increases when warm gas reaches the orifice and the backpressure rises. It has been calculated that it takes approximately 100 seconds before warm gas reaches the orifice, see calculation in Appendix C.1. Note that in the figures from Aker's report, the simulations starts after 60 seconds of elapsed time. This means that the warm gas reaches the orifice after 160 seconds or 0,044 hours of elapsed time, see for example Figure 1-4.

Based on these observations, and since we are only interested in the lowest temperatures, the work in this thesis is limited to analyzing the first 100 seconds of the flow.

1.3. Objectives

The objective of this thesis is to further investigate the temperature distribution in the 2" pipe, upstream and downstream of the orifice, using CFD. The problem will be modelled as a 2" pipe with a restriction orifice, shown in Figure 1-8. The valve is assumed to be located at the orifice. It is also assumed that it does not influence the flow.

The goal is to determine how far the low temperature propagates up- and downstream of the orifice as well as how the temperature develops as the back pressure rises. Results obtained from CFD simulations will be compared to Aker's results. Case 1a will be used for comparison, since there is very little difference in minimum temperature at the inner pipe wall in Aker's case 1a and 2a.

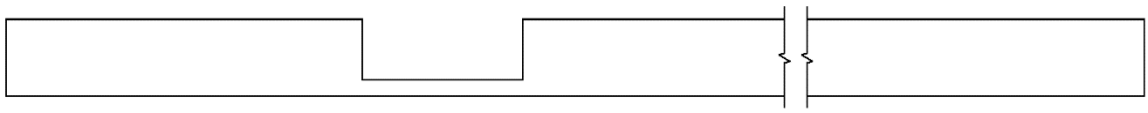

Figure 1-8 2" pipe with orifice

Due to the large pressure difference between the upstream and downstream pipe, the flow in the orifice is expected to become choked and hence the gas velocity will become sonic. The highly compressed upstream gas will expand when it exits the orifice. This is expected to create an under-expanded jet. It is also expected that the downstream flow conditions will be quite complex involving Mach disks and shock waves.

In addition, a case with 235 bar upstream pressure and 230 bar downstream pressure will be simulated, to check how the temperature develops at lower pressure differences.

A summary of the objectives:

- Create a CFD model of the problem in OpenFOAM 2.3.x
  - Select appropriate solver
  - Create and evaluate mesh
  - Select appropriate boundary conditions
- Simulate the fluid flow and the temperature development in the pipe from 0 to 100 seconds
- Determine how far the low temperatures propagates up- and downstream of the orifice
- Validate results
- Compare results to Aker's report (Case 1a)
- Run a 5 bar difference case

1.4. Previous work

An extensive literature search have been conducted. It has been found that the flow structure of under-expanded jets have been studied extensively. Temperature distribution in pipelines, due to large pressure differences and flow restrictions, does not seem to have caught the same amount of interest.

The complex and periodic shock cell structure of an under-expanded jet was studied by Pack [4]. Under-expanded jets are typically classified as moderate or highly under-expanded, based on the nozzle pressure ratio (NPR). The NPR can be found by dividing the jet exit total pressure by the ambient static pressure. For NPRs above approximately 4, the jet becomes highly under-expanded and normal shocks, also known as Mach disks, occur [5].

Crist et al. [6] studied the highly under-expanded free sonic jet using a modified wind tunnel. They found that the location of the Mach disk was insensitive to ratio of specific heats, condensation, nozzle slip geometry and absolute pressure level. The distance from the exit of the jet to the Mach disk was found to vary as the square root of the overall pressure ratio, for ratios up to 100 000. Hence, the location of the Mach disk, x_M, can be taken as:

$$x_M = 0{,}645 d_n \sqrt{\frac{P_0}{P_\infty}} \tag{1-1}$$

Where d_n is the nozzle diameter, P_0 is the pressure upstream of the nozzle and P_∞ is the pressure in the nozzle exit region, downstream of the orifice.

Addy [7] studied the effect of nozzle geometry on Mach disk characteristics for pressure ratios ranging from 1 to 10. He found that the diameter of the Mach disk, D_M, for an orifice type nozzle, obeyed the following relation:

$$D_M = 0{,}31 d_n \left(\frac{P_0}{P_\infty} - 5\right)^{1/2} \tag{1-2}$$

Under-expanded jets have also been studied numerically, using CFD. Birkby et al. [8] studied under-expanded free, sonic jets, exiting into a quiescent domain with NPRs between 3,5 and 30. Using the standard k-ε turbulence model, with an optional compressibility correction, they found that the wavelength of the shock cell was correctly predicted, compared to experimental data. However, it decayed too rapidly, as shown in Figure 1-9. They also found that for NPR larger than six, the jet became unsteady due to interaction between the Mach disk and the shear layer. Mach disk locations for NPRs between 5 and 30 were found to be in excellent agreement with the experimental data provided by Love et al. [9].

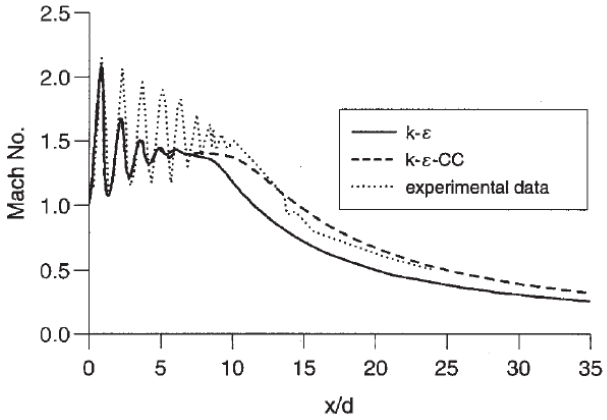

Figure 1-9 Shock cell wavelength for NPR =3,5 [8]

Jaramillo et al. [10] studied different Reynolds-Averaged Navier-Stokes (RANS) turbulence models and their performance in predicting turbulent, internal, forced-convection flow. They found that turbulence models based on the ω length-scale quantity, gave better results than the k-ε model near solid walls. In addition, the k-ω models showed better convergence and stability properties.

Liu et al. [11] studied dispersion of CO_2 released into the atmosphere, from a hole in a high-pressure pipeline, using CFD. They divided the problem into two parts, a jet model and a dispersion model. The jet model was simulated with pressures ranging from 1 to 15 MPa. The k-ω SST turbulence model was found to perform better than the k-ε model, in predicting the Mach disk location and the overall velocity field.

Xu et al. [12] studied a highly under-expanded hydrogen jet with a NPR of 105. They found that only one large Mach disk occurred at this high NPR. In addition, a significant temperature drop was recorded in the nozzle exit region, due to the rapid decrease in pressure and increase in velocity.

Wilkes et al. [13] used fluorescence imaging to study under-expanded nitrogen jets with NPRs ranging from 2 to 35 and compared the results to CFD simulations. They found that CFD simulations, for the highest NPRs, had a tendency to over-predict the size of the Mach disk and the distance from the nozzle exit to the first Mach disk.

Fu et al. [14] studied under-expanded supersonic jets, with and without combustion. Some of their results, without combustion, might be suitable for validation of the flow structure obtained in our simulations.

Rogers et al. [15] found that the Mach disk location and diameter was dependent on gas type. Both the distance from nozzle exit to first Mach disk, and the diameter of the Mach disk were found to increase with increasing density. Mach disk diameters of CNG jets were also found to have a reasonable fit with the experimental data of [7]

Most of the supersonic jets studied in literature are either free or impinging jets. The under-expanded jets in our case are confined by the pipe walls, which might influence the flow structure. Kandakure et al. [16] studied confined turbulent jets, but only for subsonic velocities. The author has not been able to find studies of supersonic confined jets.

1.5. Thesis layout

The thesis is divided into six chapters. Relevant theory on heat transfer and under-expanded jets, as well as governing equations of fluid flow and heat transfer can be found in chapter 2.

Chapter 3 contains a general description of CFD and an overview of OpenFOAM. The chapter also contains a description of the solvers that have been used, numerical discretization schemes, convergence criteria and turbulence modelling theory.

The modelling approach is described in chapter 4. This includes mesh parameters, boundary conditions, modelling of transport properties and thermodynamics etc. Chapter 5 contains the results as well as validation and discussion. The conclusion is given in chapter 6.

The thesis assignment can be found in Appendix A. Detailed results and calculations can be found in Appendix B and Appendix C respectively. OpenFOAM dictionaries are located in Appendix D and Appendix E. A detailed tutorial for creating multi-region STL files and meshes are found in Appendix F. Appendix G contains instructions on how to run cases etc. Case files and other enclosed digital material are located in Appendix H.

2. Theory

Governing equations of fluid flow and heat transfer are presented in this chapter. Relevant theory regarding choked flow, the under-expanded jet and basic heat transfer are also described.

2.1. Governing equations of fluid flow and heat transfer

The governing equations of fluid flow and heat transfer can be derived from the conservation laws of physics. Here they are presented in Cartesian coordinates. This section is based on [17].

2.1.1. Conservation of mass

The equation of mass conservation, also known as the continuity equation, is derived by considering the mass flow in and out of a control volume. The rate of change of mass in the control volume must equal the rate of mass flowing in, minus the rate of mass flowing out.

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} + \frac{\partial (\rho w)}{\partial z} = 0 \tag{2-1}$$

This can also be written in more compact vector notation:

$$\frac{\partial \rho}{\partial t} + \mathrm{div}(\rho \boldsymbol{u}) = 0 \tag{2-2}$$

2.1.2. Momentum and Navier-Stokes equations

From Newton's second law we get that the rate of change of momentum of a fluid particle, equals the sum of forces acting on the particle. This gives us the following momentum equations in x-, y- and z- direction respectively:

$$\rho \frac{Du}{Dt} = \frac{\partial (-p + \tau_{xx})}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} + S_{Mx} \tag{2-3}$$

$$\rho \frac{Dv}{Dt} = \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial (-p + \tau_{yy})}{\partial y} + \frac{\partial \tau_{zy}}{\partial z} + S_{My} \tag{2-4}$$

$$\rho \frac{Dw}{Dt} = \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial (-p + \tau_{zz})}{\partial z} + S_{Mz} \tag{2-5}$$

If we assume that the rate of deformation in the fluid is linearly proportional to the shear stress, we have a Newtonian fluid. The momentum equations then reduces to the compressible Navier-Stokes equations, here given for x-, y- and z-direction:

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \text{div}(\mu \text{ grad u}) \quad (2\text{-}6)$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \text{div}(\mu \text{ grad v}) \quad (2\text{-}7)$$

$$\rho \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \text{div}(\mu \text{ grad w}) \quad (2\text{-}8)$$

Body forces are neglected, but can easily be added if necessary.

2.1.3. Energy equation

The energy equation is derived from the first law of thermodynamics. It states that the rate of change of energy of a fluid particle equals the net rate of heat added to the particle, plus the net rate of work done on the particle. If we extract the changes due to kinetic energy, and introduce the assumption of a Newtonian fluid, we get the equation for internal energy:

$$\rho \frac{Di}{Dt} = -p\, div(\boldsymbol{u}) + div(k\, grad\, T) + \Phi_d + S_i \quad (2\text{-}9)$$

Effect of viscous stresses are described by Φ_d, the dissipation function.

2.1.4. Equation of state

The equation of state gives a relationship between the different state variables of the fluid, such as temperature, pressure and volume. The ideal gas law states:

$$pV = nRT \quad (2\text{-}10)$$

Where p is the pressure, V is the volume, n is the number of moles, R is the universal gas constant and T is the temperature.

Real gas effects, which is not modelled by the ideal gas law, can be taken into account by using for example the Peng-Robinson equation of state, proposed by Peng et al. [18].

$$p = \frac{RT}{V_m - b} - \frac{a\alpha}{V_m^2 - 2bV_m - b^2} \quad (2\text{-}11)$$

$$a = \frac{0{,}457235 R^2 T_c^2}{p_c}$$

$$b = \frac{0{,}077796 R T_c}{p_c}$$

$$\alpha = (1 + \kappa(1 - \sqrt{T_r}))^2$$

$$\kappa = 0{,}37464 + 1{,}54226\omega_a - 0{,}26992\omega_a^2$$

$$T_r = \frac{T}{T_c}$$

V_m is the volume of 1 mole of gas, also known as molar volume, ω_a is an acentric factor, p_c and T_c are the critical pressure and temperature and R is the universal gas constant [19].

2.2. Choked flow and the under-expanded jet

If the pressure difference upstream and downstream of an orifice is above a critical value, the flow through the orifice becomes choked. The critical pressure is found from equation (2-12) [20].

$$\frac{P_0}{P_\infty} = (\frac{\gamma + 1}{2})^{\frac{\gamma}{\gamma - 1}} \tag{2-12}$$

Where P_0 is the upstream pressure, $P_\infty$is the downstream pressure, and γ is the ratio of specific heat capacities. For an ideal gas with γ=1,4 the critical pressure ratio becomes:

$$\frac{P_0}{P_\infty} = (\frac{1{,}4 + 1}{2})^{\frac{1{,}4}{1{,}4-1}} = 1{,}89 \tag{2-13}$$

When the flow becomes choked, the mass flow can be estimated using equation (2-14) [20].

$$\dot{m}_{gas} = C_d \cdot A_n \cdot \sqrt{\gamma \rho_0 P_0 (\frac{2}{\gamma + 1})^{\frac{\gamma+1}{\gamma-1}}} \tag{2-14}$$

C_d is the coefficient of discharge, A_n is the cross-sectional area of the orifice and ρ_0 is the density of the gas upstream of the orifice. The orifice can be regarded as a nozzle and the flow will become a jet. When the pressure at the exit of the nozzle is greater than the surrounding pressure, the jet is said to be under-expanded.

Under-expanded jets can be classified as moderately or highly under-expanded, depending on the pressure ratio, $\frac{P_0}{P_\infty}$. According to Donaldson et al. [5], the jet becomes moderately under-expanded when the pressure ratio is between 2,08 and 3,85. The jet is expanding to the surrounding pressure through a series of oblique shocks, displayed in the upper part of Figure 2-1. When the pressure ratio is larger than 3,85, the jet becomes highly under-expanded. Such jets are characterized by the presence of normal shock disks, also known as Mach disks, which can be seen in the lower part of Figure 2-1. Due to the large pressure difference, expansion along the centerline of the jet becomes rapid. This results in a very low axial pressure that alters the structure of the first shock cell and results in the Mach disk.

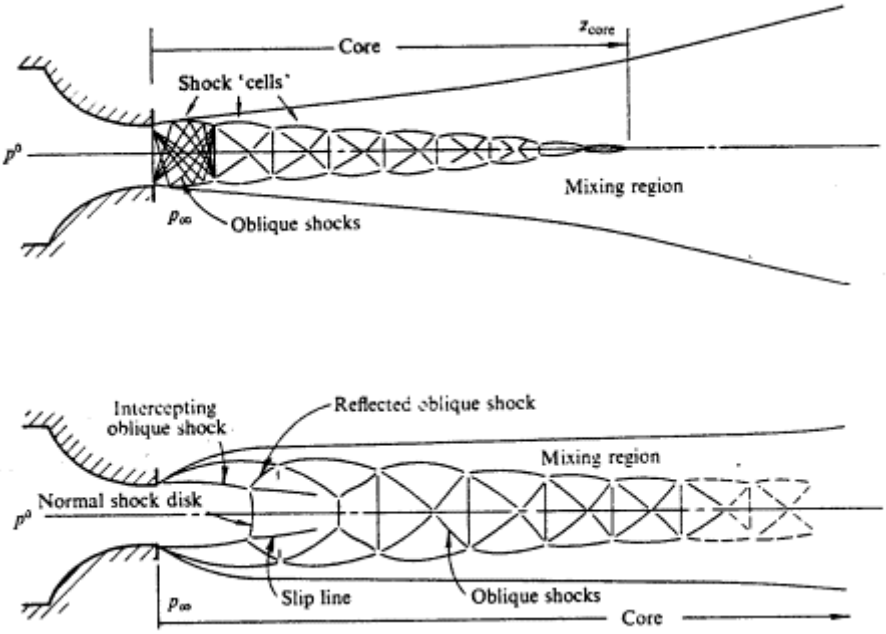

Figure 2-1 Moderately and highly under-expanded jet [5]

Crist et al. [6] studied the structure of the initial shock cell in detail. When the fluid exits the throat of the nozzle it accelerates and expands to the atmospheric pressure through an expansion fan. As Figure 2-2 shows, the expansion waves are reflected off the free jet boundary as compression waves. The compression waves join to form the intercepting shock. The flow between the intercepting shock and the jet core, as well as the core itself, is supersonic. However, the jet core has a higher Mach number. Immediately after the Mach disk, the flow speed is subsonic along the centerline, while the flow outside the slip line still is supersonic. Where the Mach disk, the intercepting shock and the reflected shock meet, a triple point is formed. The distance from the exit of the nozzle to the Mach disk is proportional to the square root of the nozzle pressure ratio, according to equation (1-1).

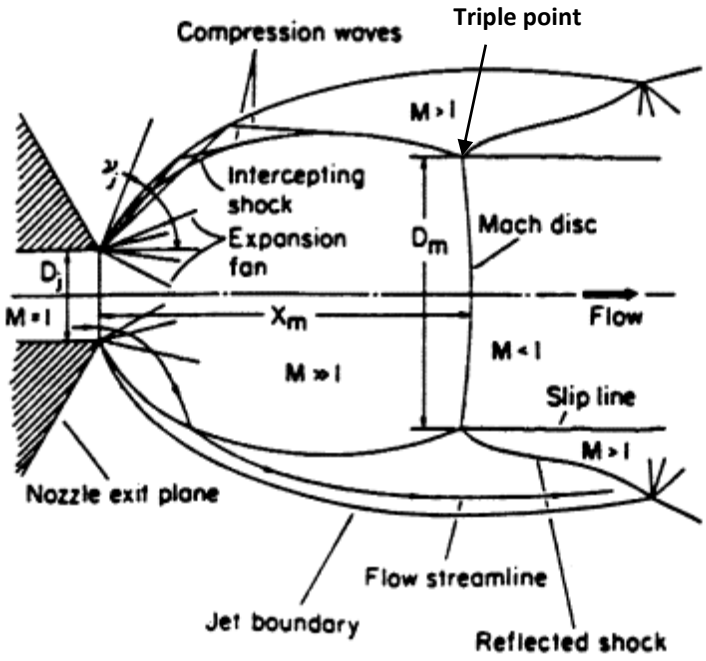

Figure 2-2 First shock cell of a highly under-expanded jet [6]

2.3. Heat Transfer

In general, three different modes of heat transfer exist: conduction, convection and radiation. Radiation is neglected in this case.

Heat transfer problems can be either steady state or transient. Solutions to steady state problems only varies with location, while transient problems also varies with time. Steady state problems are therefore easier to solve, since all derivatives with respect to time is equal to zero.

2.3.1. Conduction

This section is mainly based on references [21],[22] and [24]. Heat transfer due to conduction takes place in solids and quiescent fluids. The heat is transferred by diffusion and collisions between particles, without any mass flow. Heat flows from a high- to a low-temperature region due to the temperature gradient between those regions.

The heat transfer rate varies, depending on the material, geometry, and the temperature gradient. Fourier's law states the relationship between the heat flow and the temperature gradient, here shown for a one-dimensional problem.

$$\dot{Q}_{cond} = -k_c A \frac{dT}{dx} \tag{2-15}$$

k_c is the thermal conductivity, which is a measure of a materials ability to transfer heat by conduction. A is the cross-sectional area, and $\frac{dT}{dx}$ is the temperature gradient. $\dot{Q}_{cond}$ is the heat flux. The negative sign indicates that the heat flows in the opposite direction of the temperature gradient.

Materials with the atoms closely spaced, such as solids, generally have the highest thermal conductivity. Gases and vapors have the lowest conductivity due to greater distance between the atoms [23].

The thermal conductivity is also temperature dependent. In many pure metals, it tends to decrease with increasing temperature. In gases however, the opposite is true. Higher temperatures results in greater thermal conductivity. For anisotropic materials, k_c also varies with orientation.

Another important material property is the thermal diffusivity, α. It is defined as the thermal conductivity divided by density, ρ, times specific heat capacity, c_p [25].

$$\alpha = \frac{k_c}{\rho c_p} \tag{2-16}$$

In transient problems, the thermal diffusivity is a measure of how quickly the heat is conducted through the material. The quantity ρc_p is often referred to as the volumetric heat capacity. Thus, the thermal conductivity is a measure of a materials ability to conduct heat relative to its volumetric heat capacity.

The distribution of heat due to conduction is described by a parabolic partial differential equation, also known as the heat equation. For an isotropic material without internal heat generation, the one-dimensional heat equation becomes:

$$\frac{\partial T}{\partial t} = \frac{k_c}{\rho c_p}\left(\frac{\partial^2 T}{\partial t^2}\right) = \alpha\left(\frac{\partial^2 T}{\partial x^2}\right) \tag{2-17}$$

For a general hollow cylinder, such as a pipe, the equation becomes [26]:

$$\frac{1}{r}\frac{d}{dr}\left(k_c r \frac{dT}{dr}\right) = 0 \qquad r_1 < r < r_2 \tag{2-18}$$

r is the radial coordinate in the pipe, while r_1 and r_2 is the inner and outer radius respectively.

2.3.2. Convection

This section is mainly based on reference [27]. In the presence of bulk fluid motion, heat is transferred trough a fluid by convection. We usually distinguish between forced and natural convection. When the flow is initiated by the buoyancy effect, we have natural convection. If the fluid motion is caused by external means, such as a pump, we have forced convection. It is also usual to classify convection as either internal or external, depending on whether the flow occurs over a plate or inside a pipe.

The rate of heat transfer in a fluid due to convection is much larger than the rate due to conduction. This is because the fluid motion brings warmer and cooler portions of fluid into contact, which increases the heat transfer rate.

Convection is the most complicated heat transfer mode, and the rate of heat transfer depends on several fluid properties such as: dynamic viscosity μ, thermal conductivity k_c, density ρ, specific heat capacity c_p and fluid velocity. Other important variables are: geometry, surface roughness, and whether the flow is turbulent or laminar. However, despite the complexity, the rate of heat transfer due to convection is proportional to the temperature difference. This is expressed in Newton's law of cooling:

$$\dot{Q}_{conv} = h_c A_s (T_s - T_\infty) \tag{2-19}$$

Where h_c is the convective heat transfer coefficient, A_s is the surface area with heat transfer, T_s is the surface temperature and T_∞ is the temperature in the fluid sufficiently far from the surface. Even though this expression looks relatively simple, the convective heat transfer coefficient is difficult to determine, since it depends on many of the above-mentioned fluid properties. A more detailed description of convection can be found in reference [27].

3. Computational Fluid Dynamics

This chapter contains a general description of CFD and a brief overview of the open-source CFD code OpenFOAM. Solvers, numerical discretization schemes and solution controls are described in detail, and turbulence modelling theory is explained.

3.1. Introduction to CFD

This section is based on reference [17]. CFD is used to analyze fluid flow, heat transfer, chemical reactions and other phenomena associated with fluid dynamics. CFD codes are based on numerical algorithms that can solve problems related to fluid flow. All CFD codes consists of three basic components: pre-processor, solver and post-processor.

The pre-processor is used to specify the computational domain, generate the mesh, apply boundary conditions and define the physical properties of the problem. User input data are subsequently transformed into a form that the solver can use.

When the problem is properly defined in the pre-processor, the solver can compute a solution. Most commercial CFD codes, as well as OpenFOAM, use the finite volume method (FVM). The FVM consists of three main steps:

- Integrating governing equations over all control volumes in the computational domain
- Discretizing the integral equations into a set of algebraic equations
- Solving the algebraic equations by iterative methods

The integration of the governing equations provides an exact expression for the conservation of relevant properties, e.g. velocity, for each cell in the computational domain. Hence, there is a clear connection between the underlying conservation laws of physics, and the numerical solution method.

Numerous discretization schemes are available and they must be carefully selected, depending on the characteristics of the problem at hand.

The results can be visualized in the post-processor. There are several data visualization techniques available, for example:

- Contour plot
- Vector plot
- Tracking of particles
- Plot of variables over time or space

Complex physics are involved in solving fluid flow problems. It is important that the user have an understanding of the underlying physics. A nice looking result might not necessarily be physically correct. It is therefore paramount that the results are compared with results

from experiments or data from similar problems reported in journals or literature. Simulation results should also be tested for grid independence.

3.2. OpenFOAM

Open Source Field Operation And Manipulation (OpenFOAM) is an open source C++ library. It is designed to create executables, called applications. The applications can be divided into two categories: utilities and solvers. Solvers are made to solve a specific type of problems in continuum mechanics, while utilities are designed to perform data manipulation [28] .

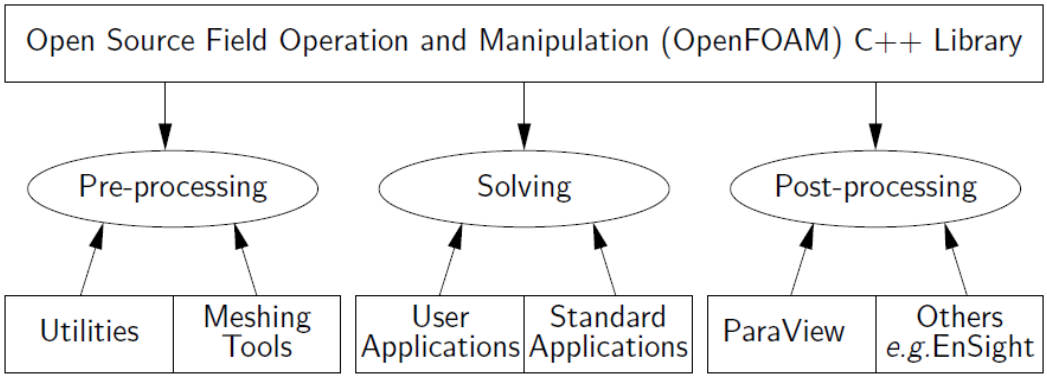

Figure 3-1 Overview of OpenFOAM structure [28]

Users can create their own solvers and utilities, if they have the necessary knowledge of the physics involved as well as the needed programming skills. OpenFOAM comes with both pre- and post-processing environments. It has no graphical user interface (GUI) and input data are given in text files called dictionaries. Results can be visualized in ParaView.

3.2.1. Case Structure

All OpenFOAM cases contain three basic folders, namely: system, constant and 0, see Figure 3-2.

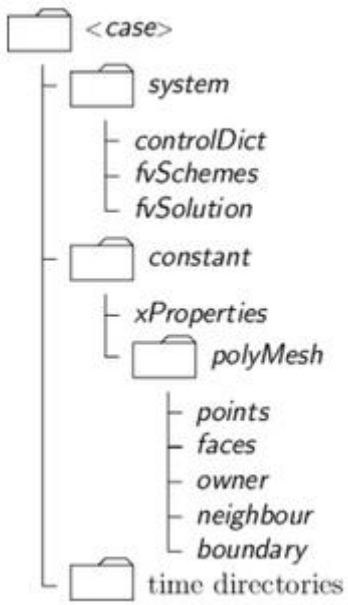

Figure 3-2 General OpenFOAM case structure [29]

Parameters associated with the solution procedure are specified in dictionaries inside the system folder. The most important dictionaries are: controlDict, fvSchemes and fvSolution. Length of time steps, write intervals and write precision etc. are specified in the controlDict. Discretization schemes for the different variables are selected in fvSchemes, while fvSolution controls solvers, tolerances and other algorithm control functions.

The constant folder contains the mesh, in a subfolder called polyMesh, as well as dictionaries that specify physical properties of the problem such as transportProperties and thermophysicalProperties.

Dictionaries containing boundary conditions for the different fields are located in the 0 folder. For each new time step the solver computes, a new time folder is created containing the fields for this particular time step.

3.3. Solvers

3.3.1. chtMultiRegionFoam

chtMultiRegionFoam is a transient, compressible solver for conjugate heat transfer problems. It supports multiple fluid and solid regions. The solver is a combination of heatConductionFoam for the solid region, and boyantFoam for the fluid region.

In the early stages of this thesis, it was assumed that this solver could be used. STL files were created in Inventor and a 3D mesh was created using snappyHexMesh. However, even though chtMultiRegionFoam is called a compressible solver, it assumes that the density of the gas is not a function of pressure. Hence, only the temperature influences the density. This assumption is acceptable for flow speeds up to Mach 0,3. Due to the choked flow conditions in our case, the flow speed will become supersonic in the region after the orifice. chtMultiRegionFoam is unable to handle such high flow speeds, and the calculation crashes.

One possible solution could be to create a new solver by replacing the fluid solver in chtMultiRegionFoam with a real compressible solver that handles sonic velocities. However, the complex flow that occurs at sonic velocity requires extremely short time steps, as explained in chapter 4.1. This approach was therefore abandoned in favor of the quasi-transient solution, also explained in detail in chapter 4.1.

To reduce the number of cells and hence the computational time, the 3D mesh was replaced with a 2D axisymmetric mesh, see section 4.2. Appendix F contains a description of how to create multi-region STL files of high quality, how to create multi-region 3D meshes with snappyHexMesh and how to run a 3D case in chtMultiRegionFoam. I hope that the knowledge gathered there could be useful for others.

In this case, chtMultiRegionFoam is used to solve the heat conduction in the pipe. LaplacianFoam is also capable of solving for the heat transfer in the solid region. However, chtMultiRegionFoam supports a special boundary condition, called externalWallHeatFluxTemperature, which is not supported by laplacianFoam. It enables the user to specify insulation at the pipe wall and thereby take into account the effect of the

ambient temperature. In addition, chtMultiRegionFoam has been used to solve the 5 bar difference case, where the flow speed is subsonic.

3.3.2. rhoCentralFoam

rhoCentralFoam is a compressible solver for high-speed viscous flows, based on semi-discrete, non-staggered central schemes, created by Greenshields et al. [30]. A brief description of the solver's solution algorithm is presented in this section. For further details on how the solver works, the reader is referred to [30].

rhoCentralFoam solves each of the governing equations separately. First, the density is calculated from the continuity equation, using velocity values from the previous time step.

$$\frac{\partial \rho}{\partial t} + \mathrm{div}(\rho \boldsymbol{u}) = 0 \tag{3-1}$$

Next, the momentum equation is solved. To avoid a completely explicit solution procedure, which produces severe time step limitations, this equation is solved in two steps. First, the inviscid momentum density, $\widehat{\boldsymbol{u}}$, is calculated. $(\frac{\partial \widehat{\boldsymbol{u}}}{\partial t})_I$ is the time derivative due to inviscid fluxes only.

$$(\frac{\partial \widehat{\boldsymbol{u}}}{\partial t})_I + \mathrm{div}(\boldsymbol{u}\widehat{\boldsymbol{u}}) + \mathrm{div}\, p = 0 \tag{3-2}$$

Since $\widehat{\boldsymbol{u}} = \rho \boldsymbol{u}$, a new velocity value can be found using the calculated values for $\widehat{\boldsymbol{u}}$ and ρ. The viscous forces are then taken into account by solving a diffusion correction equation for $\boldsymbol{u}$. $(\frac{\partial(\rho \boldsymbol{u})}{\partial t})_V$ is the time derivate related to diffusion only and $\mathbf{T}_{visc}$ is the viscous stress tensor.

$$(\frac{\partial(\rho \boldsymbol{u})}{\partial t})_V - \mathrm{div}(\mu\, grad\, \boldsymbol{u}) - \mathrm{div}\, \mathbf{T}_{visc} = 0 \tag{3-3}$$

The energy equation is solved using a similar approach. First, it is solved for total energy density, $\hat{E}$, neglecting the diffusive heat flux. Note that $\hat{E} = \rho E$.

$$(\frac{\partial \hat{E}}{\partial t})_I + \mathrm{div}\big(\boldsymbol{u}(\hat{E} + p)\big) + \mathrm{div}\,(\mathbf{T} \cdot \mathbf{u}) = 0 \tag{3-4}$$

The temperature is then calculated from equation (3-5), using the already calculated values for $\hat{E}$, ρ and $\boldsymbol{u}$. c_v is the specific heat capacity at constant volume.

$$T = \frac{1}{c_v}(\frac{\hat{E}}{\rho} - \frac{|\boldsymbol{u}|^2}{2}) \tag{3-5}$$

Finally, a diffusion correction equation is solved for T, to include the diffusive heat flux.

$$(\frac{\partial(\rho c_v T)}{\partial t})_V - \mathrm{div}(k_c\, grad\, T) = 0 \tag{3-6}$$

The variables k and μ are functions of temperature and they are updated within each iteration. They then remain constant until the next iteration. In addition, the pressure is updated at the end of each iteration, using $p = \rho RT$. Figure 3-3 shows an illustration of the solution algorithm used by rhoCentralFoam.

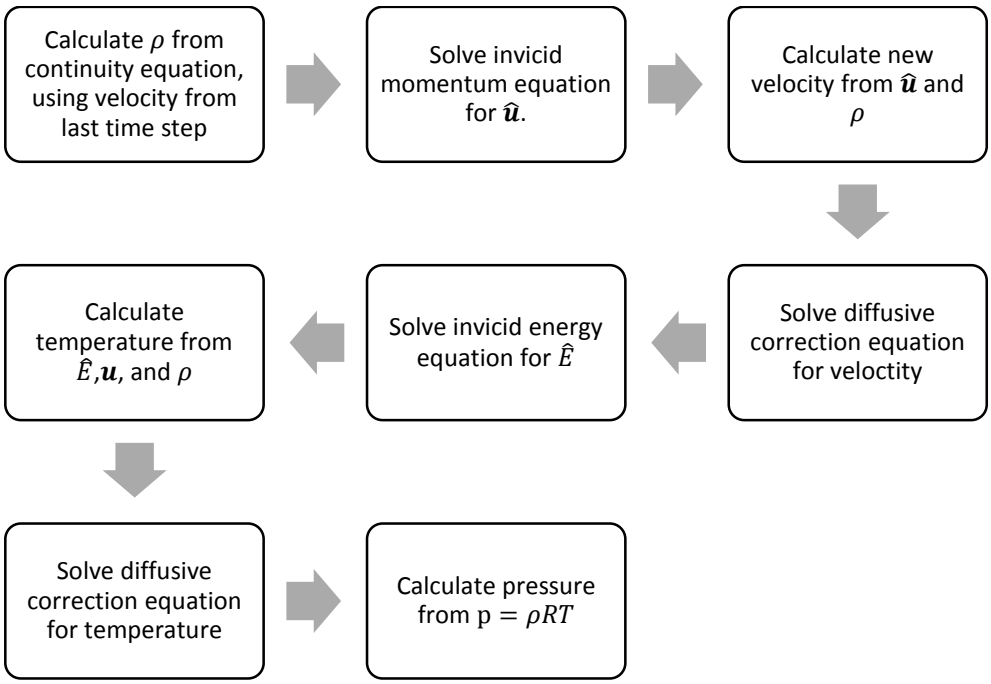

Figure 3-3 Solution algorithm for rhoCentralFoam

In their paper, Greenshields et al. [30] validated the rhoCentralFoam solver against experimental data and analytical solutions using four different test cases. Generally, the accuracy of the solver was found to be good and it gave satisfactory flow predictions.

One of the test cases, the supersonic Ladenburg jet, is especially interesting, since it is quite similar to the problem that is considered in this thesis. Ladenburg et al. [32] used an interferometer to measure the density of a supersonic jet. Dry air was discharged into the open atmosphere from a pressurized tank, through a circular, converging nozzle. The nozzle exit diameter, was 10 mm, see Figure 3-4. Greenshields et al. [30] created an equivalent case using rhoCentralFoam. Here is how they describe their case setup:

"The case is simulated as axisymmetric with a domain of height 10mm, i.e. 2× the orifice radius, and length 30mm. A mesh of 240 cells along the length and 80 cells in the radial direction was used, which was sufficiently fine to produce a solution in which the location of the Mach disk did not change appreciably under further mesh refinement. The solver was run to a steady state at a CFL number of 0.5; typically, it took approximately 20 characteristic flow times to reach steady state, where the characteristic flow time is the time that a particle would take to travel the length of the geometry moving at the jet discharge velocity, i.e. approximately 2ms in this case." [30]

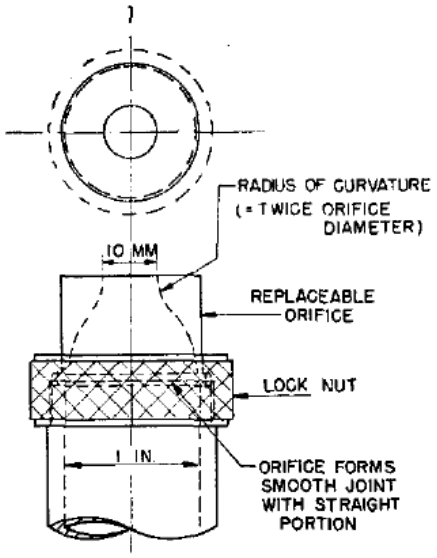

Figure 3-4 Ladenburg nozzle [31]

To test if the solver was capable of reproducing the Mach disk feature, the pressure in the tank was set to 4,14 bar. Boundary conditions for the case are listed in Table 3-1.

Table 3-1 Boundary conditions Ladenburg jet validation case

	Nozzle throat	**Freestream**
Pressure	2,72 bar	1,01 bar
Velocity	315,6 m/s	0
Temperature	247,1 K	297 K

Figure 3-5 shows a comparison of the density contours produced by rhoCentralFoam and the experimental data.

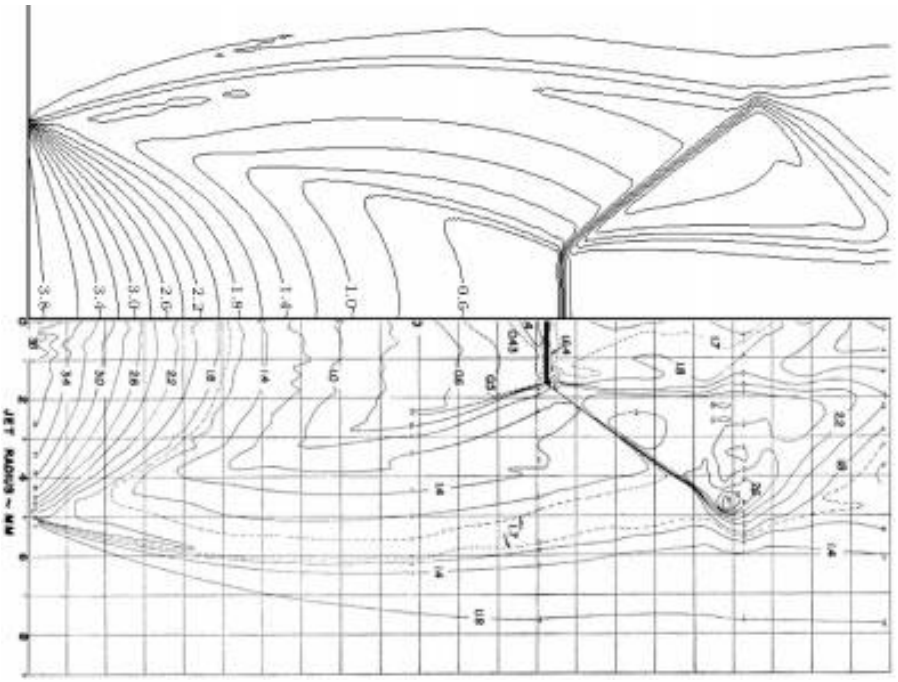

Figure 3-5 rhoCentralFoam simulation (top) and experimental data (bottom) [30]

The Mach disk forms a little further downstream in the simulation than in the experiment. However, the simulation error is within 3 % of the experimental data. According to [30], the differences in inlet boundary conditions between the experiment and the simulation could be the cause of this error. This comparison shows that rhoCentralFoam is able to reproduce the Mach disk feature and that shocks are predicted in a satisfactory way.

OpenFOAM comes with several tutorial cases. The Ladenburg jet is one of those cases. To check the validity of the current version of rhoCentralFoam, a comparison between the tutorial case and the results obtained by Greenshields et al. [30] was made, see the density contour plot in Figure 3-6.

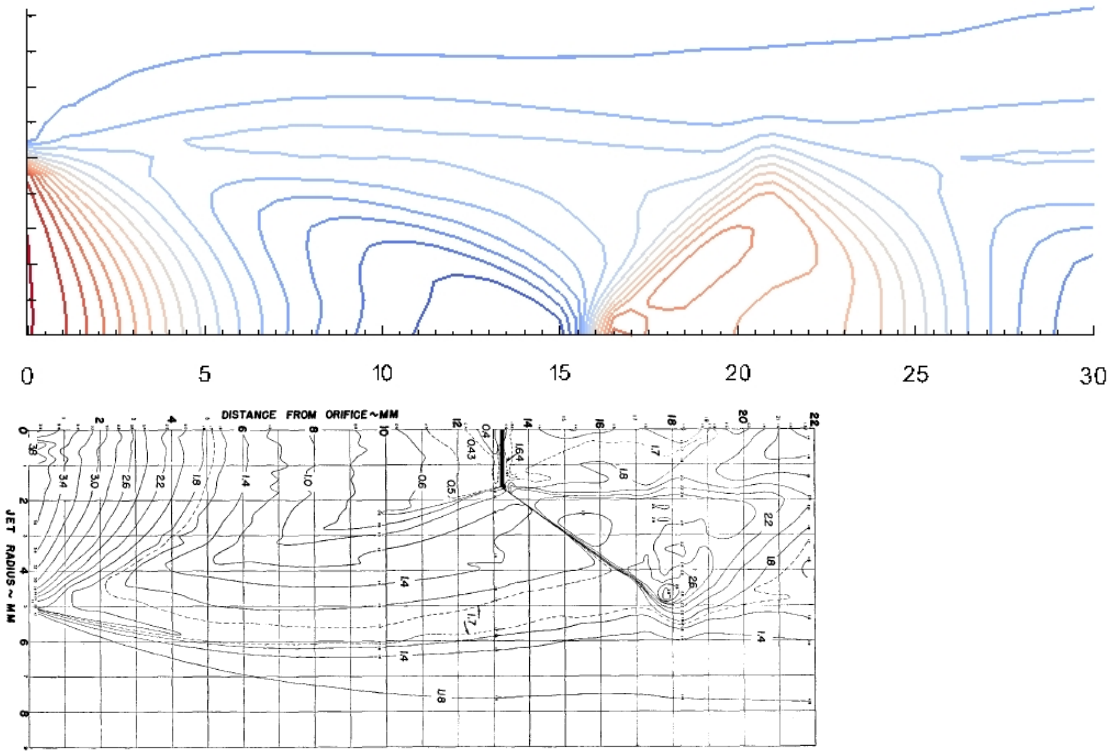

Figure 3-6 Comparison of Ladenburg tutorial case and data from experiment

It can be seen that the Mach disk in the tutorial case forms too far downstream, compared to Figure 3-5. It is assumed that this is due to different mesh refinement levels. Greenshields' mesh had typical cell length of 0,125 mm, while the typical cell length in the tutorial is 0,5 mm.

3.4. Numerical schemes

According to the FVM the governing equations are integrated over the control volume. The integral equations are then discretized into algebraic equations. Several different numerical discretization schemes are available in OpenFOAM. The terms in the integral equations are divided into different groups, shown in Table 3-2 [33].

Numerical discretization schemes must be specified for each group of terms. In addition, it is possible to select specific discretization schemes for each of the variables within a group.

Table 3-2 Groups of terms

Group	Description
interpolationSchemes	Point-to-point interpolations of values
snGradSchemes	Component of gradient normal to a cell face
gradSchemes	Gradient ∇ terms
divSchemes	Divergence $\nabla \cdot$ terms
laplacianSchemes	Laplacian ∇^2 terms
timeScheme	First and second time derivatives
fluxRequired	Fields that require generation of a flux

Table 3-3 shows the schemes applied in this thesis. For case 11, described in section 4.1, some additional schemes have been used. These can be seen in Appendix E.3.3. The selection of numerical schemes is based on the fvSchemes files from the rhoCentralFoam and chtMultiRegionFoam tutorials that come with OpenFOAM.

Table 3-3 Selection of numerical discretization schemes in fvSchemes dictionary

Group	rhoCentralFoam	chtMultiRegionFoam
fluxScheme	Kurganov	N/A
ddtSchemes	Euler	Euler
gradSchemes	Gauss linear	Gauss linear
divSchemes	div(tauMC) Gauss linear div(phi,U) Gauss limitedLinearV 1 div(phi,h) Gauss limitedLinear 1 div(phi,k) Gauss limitedLinear 1 div(phi,omega) Gauss linear	N/A
laplacianSchemes	Gauss linear corrected	laplacian(alpha,h) Gauss linear corrected;
interpolationSchemes	default linear reconstruct(rho) vanLeer reconstruct(U) vanLeerV reconstruct(T) vanLeer	linear
snGradSchemes	default corrected snGrad(U) corrected	corrected
fluxRequired	default no	default no

Some of the schemes have a “V” added to their name. They are improved versions for vector fields where the direction of the field is taken into account by the limiter [33].

We will not go into details about the different discretization schemes as this is far beyond the scope of this thesis. However, it is interesting to note how the selection of schemes can affect both the efficiency and accuracy of the solution.

For example, applying discretization schemes such as the central difference scheme, the upwind scheme, the hybrid difference scheme or the power law scheme in this thesis would require an extremely fine mesh to produce accurate shock predictions. The discontinuous

changes in velocity, density and pressure, caused by the shocks, could easily introduce unphysical oscillations in the solution. By applying Total Variation Diminishing (TVD) schemes such as the vanLeer or limitedLinear, we are able to capture the sharp shocks on a coarser mesh, without introducing any spurious oscillations in the solution. This saves computational time and ensures a correct solution. Harten introduced TVD. For further details, the reader is referred to his paper, [34], and [35]. For further tips on how to select appropriate numerical schemes, reference [36] might be useful.

3.5. Solution and algorithm control

It is always important to verify the convergence of a CFD solution, to ensure that the solution is physically correct. The convergence of a simulation can be evaluated by monitoring the residuals and the continuity error. In addition, important flow parameters can be plotted as a function of time, to see if they converge. It is also important to monitor the Courant number and make sure that it is below one. However, a converged solution does not need to be physically correct. It is therefore important to verify its validity by, for example, comparing it to experimental data.

3.5.1. Residuals and continuity error

Residuals is a measure of the error in the solution. The residuals for each time step are found by substituting the current solution into the equations and calculate the absolute value of the difference between the left and the right hand side. In addition, the residuals are normalized to make sure that they are independent of the scale of the problem [37].

The continuity error is a measure of the error in the solutions mass balance. The sum of the magnitude of the flux imbalance for all cells is a good measure of the continuity error [38]. This can be found in the solver output from OpenFOAM, called “sum local” continuity error.

3.5.2. Courant number

The Courant number can be calculated for each cell. It is defined as the distance the fluid travels during one time step, divided by the length of the cell. It is an important indication of whether the time steps are small enough for a good time discretization [38].

$$Co = \frac{\Delta t \cdot |U|}{\Delta x} \tag{3-7}$$

Δt is the length of the timestep, Δx is the cell length and U is the flow velocity. The Courant number should be kept below one in the simulations, to make sure that the flow propagates adequately and does not skip cells. Lower Courant number gives more stable simulations [38].

It has been found that an initial Courant number of 0,5 gives a stable solution. Increasing the Courant number, increases the length of Δt, and hence the computational time. Experiments with different Courant numbers has shown that it can be increased to 0,9, after the solution time has passed 1 ms, without distorting the stability of the solution.

3.5.3. fvSolution

A solver must be specified for each of the discretized equations. This is done in the fvSolution dictionary. Important solution parameters such as tolerance and relative tolerance can be specified for each solver. A preconditioner can also be added.

The equations are solved by reducing the residuals through a series of iterations. Before an equation is solved, a value for the initial residual is obtained, using the current values of the field. The residual is re-calculated after each iteration. If the residuals falls below the specified tolerance or relative tolerance value, the solver stops. Relative tolerance is the ratio of current to initial residuals.

Table 3-4 shows the solvers that have been used in this thesis, including tolerances and relative tolerances. The choice of solvers is mostly based on the tutorial cases that come with OpenFOAM. Solvers for case 11, described in section 4.1, can be found in Appendix E.3.5. They are not listed here.

Table 3-4 Solvers selected in fvSolution

Equation	**rhoCentralFoam**	**chtMultiRegionFoam**
U, k, ω	solver smoothSolver; smoother GaussSeidel; nSweeps 2; tolerance 1e-09; relTol 0.01;	N/A
ρ, ρU, ρE	solver diagonal	N/A
e	solver smoothSolver; smoother GaussSeidel; nSweeps 2; tolerance 1e-10; relTol 0.0;	N/A
h	solver smoothSolver; smoother GaussSeidel; nSweeps 2; tolerance 1e-10; relTol 0.0;	solver PCG; preconditioner DIC; tolerance 1e-06; relTol 0.1;
hFinal	N/A	solver PCG; preconditioner DIC; tolerance 1e-06; relTol 0.0;

rhoCentralFoam uses smooth solvers with the Gauss Seidel smoother, which is the most reliable option. The nSweeps keyword indicates how many sweeps the smoother performs, before the residuals are re-calculated. The diagonal solver is used for explicit systems of equations [37]. chtMultiRegionFoam uses a Preconditioned Conjugate Gradient (PCG) solver with the Diagonal Incomplete-Cholesky (DIC) preconditioner. The preconditioner multiplies both sides of the equation with a new matrix, which might make the solution process easier [39].

3.6. Turbulence modelling

This section is based on reference [17], [40] and [46].
The flow in our case is clearly turbulent, due to the high velocity through and after the orifice. In a turbulent flow, all flow variables vary in a random manner. Figure 3-7 illustrates how the velocity at a point in a turbulent flow, typically varies with time.

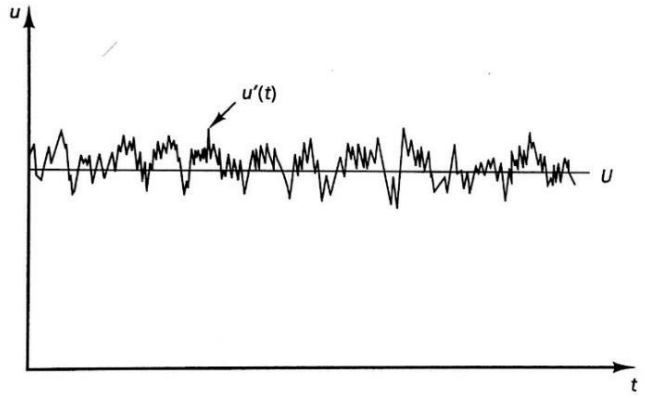

Figure 3-7 Turbulent flow [17]

Turbulent flow contains rotational flow structures, known as turbulent eddies, with a wide range of length scales. The largest eddies are dominated by inertia forces and extract energy from the mean flow by a process called vortex stretching. This process leads to eddies with smaller length scales as well as smaller time scales. We will not go into details about this process here. More information can be found here [40].

There are three basic approaches to turbulence modelling. Reynolds Averaged Navier-Stokes models (RANS), Large Eddie Simulation (LES) and Direct Numerical Simulation (DNS). Figure 3-8 illustrates the accuracy of the different approaches.

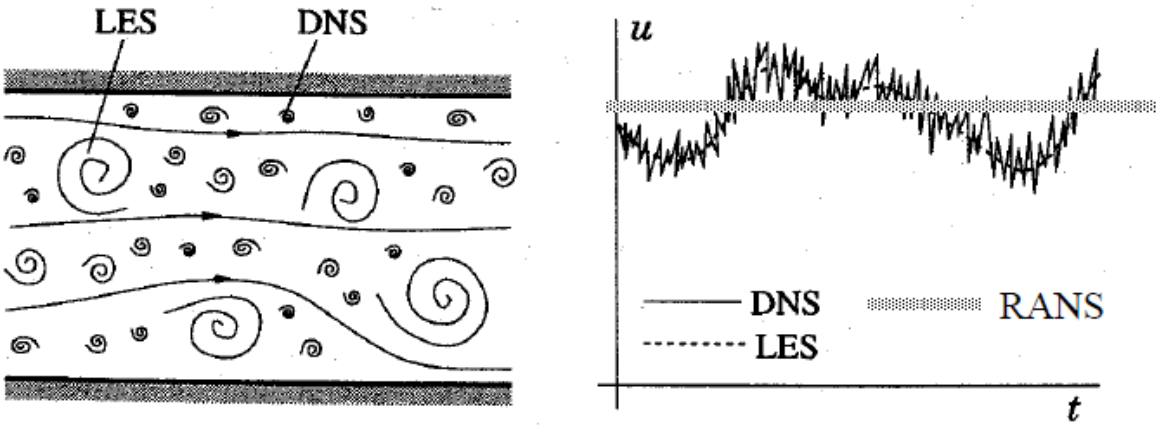

Figure 3-8 RANS, LES and DNS [40]

DNS resolves all the details of the flow, and is the method that requires most computational power. LES resolves only the largest eddies, while the smaller ones are approximated. The RANS-models requires least computational power. The flow variables are decomposed into a steady mean value, Φ, and a fluctuating component $\varphi'(t)$. This is known as Reynolds decomposition. RANS-modelling is assumed to be sufficient in our case.

Introducing the Reynolds decomposition into the governing equations results in the appearance of new terms. The new terms in the momentum equations are often referred to as Reynolds stresses. In the scalar transport equations, the new terms for example represents the heat or mass flux due to turbulence. Governing equations for the new terms can be derived, but these also include new unknowns. It is therefore necessary to find a relation between known terms and these new unknowns. This is done by introducing a turbulence model.

A turbulence model is a set of equations that express a relation between the unknown Reynolds stresses and known flow quantities. Most RANS turbulence models are based on the concept of turbulent viscosity, also known as eddy viscosity, proposed by Boussinesq in 1877. The Reynold stress tensor is assumed to be proportional to the mean rate of deformation [41].

$$\tau_{ij} = 2\mu_T S_{ij} - \frac{2}{3}\rho k \delta_{ij} \tag{3-8}$$

Where τ_{ij} is the Reynold stress tensor, μ_T is the eddy viscosity, $S_{ij}$is the mean strain rate tensor, k is the turbulent kinetic energy and δ_{ij} is the Kroenecker-delta. The eddy viscosity is not a flow property and must be estimated from the turbulence model.

It can be shown from dimensional considerations that the eddy viscosity is proportional to the characteristic velocity of the turbulence, u_T, and the characteristic length scale, l_T.

$$\mu_T = \rho u_T l_T \tag{3-9}$$

As equation (3-9) shows, the eddy viscosity is dependent on two variables, apart from the density. This has led to the development of so-called “two-equation” turbulence models. These models use two extra transport equations to calculate u_T and l_T, which in turn is used to calculate μ_T. The k-ω SST model, which is applied in this thesis, is an example of a two-equation model.

Turbulent transport of heat, mass and other scalar properties can be modelled in a similar way. The turbulent transport of a scalar is assumed to be proportional to the gradient of the mean value of the transported quantity [17].

$$-\rho\overline{u_i'\varphi'} = \Gamma_T \frac{\partial \Phi_s}{\partial x_i} \tag{3-10}$$

Where $\overline{u_i'\varphi'}$ is the term that appear in the scalar transport equation due to Reynolds decomposition, Γ_T is the eddy diffusivity and Φ_s is the scalar variable.

3.6.1. k-ω SST

Menter proposed the k-ω SST turbulence model [42]. It is a combination of the widely used k-ε model and the k-ω model. A common problem with the k-ω model is that it is too sensitive to the inlet boundary condition for turbulence in the free-stream region. This problem is avoided in the k-ω SST model, since the Shear Stress Transport (SST) formulation

switches from the k-ω model in the boundary layer, to the k-ε model in the free-stream region [43].

As already mentioned, the k-ω SST model uses two partial differential equations, with two variables, to model the effect of turbulence. The variables are turbulent kinetic energy, k, and the specific rate of dissipation, ω. ω is implicitly defined using the turbulent kinetic energy and the turbulent dissipation rate ε.

$$\omega = \frac{\varepsilon}{k} \tag{3-11}$$

ω can be regarded as the rate at which the turbulent kinetic energy is converted into thermal internal energy per unit volume and time. Equation (3-12) shows the relationship between k and u_T and k, ω and l_T.

$$u_T = \sqrt{k} \qquad l_T = \frac{\sqrt{k}}{\omega} \tag{3-12}$$

If we insert equation (3-12) into (3-9), we obtain an expression for the eddy viscosity in the k-ω SST model.

$$\mu_T = \rho \frac{k}{\omega} \tag{3-13}$$

The transport equations for k and ω are given in equation (3-14) and (3-15)

$$\frac{\partial \mathrm{k}}{\partial t} + \mathrm{U}_j \frac{\partial \mathrm{k}}{\partial x_j} = \mathrm{P}_k - \beta^* k\omega + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_k \nu_t)\frac{\partial k}{\partial x_j}\right] \tag{3-14}$$

$$\frac{\partial \omega}{\partial t} + \mathrm{U}_j \frac{\partial \omega}{\partial x_j} = \alpha_1 S^2 - \beta\omega^2 + \frac{\partial}{\partial x_j}\left[(\nu + \sigma_\omega \nu_t)\frac{\partial \omega}{\partial x_j}\right] + 2(1 - F_1)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i} \tag{3-15}$$

P_k is the rate of production of turbulent kinetic energy, ν is the kinematic viscosity, ν_t is the kinematic eddy viscosity, S is the mean rate of strain, and β, β^*, σ_k, σ_ω, $\sigma_{\omega 2}$ and α_1 are constants. F_1 is known as the blending function. It has a value of one in the free stream and is zero in the boundary layer. This is how the k-ω model is activated in the boundary layer and turned off in the free stream. (Note that tensor notation is used in the above equations. $x_1 = x, x_2 = y,$ and $x_3 = z, u_1 = u, u_2 = v,$ and $u_3 = w$). More detailed explanations regarding model constants and derivation of the transport equations can be found in [17].

The k-ω SST is reported to perform well for both under-expanded jets and heat transfer applications, see [10], [11] and [12].

3.6.2. Wall functions and y^+

This section is based on [44],[45],[46] and [47]. A dimensionless wall coordinate, y^+, can be defined:

$$y^+ = \frac{u_\tau \cdot y_p}{\nu} \tag{3-16}$$

Where ν is the kinematic viscosity and y_p is the distance from the wall to the center of the first cell in the mesh. u_τ is the shear velocity, defined as $\sqrt{\frac{\tau_w}{\rho}}$, where τ_w is the wall shear stress and ρ is the fluid density. In addition, a dimensionless velocity, u^+, can be defined where u_l is the local velocity.

$$u^+ = \frac{u_l}{u_\tau} \tag{3-17}$$

According to the no-slip condition, the fluid is stationary at a solid wall. The fluid close to the wall is retarded and a turbulent boundary layer forms. This boundary layer can be divided into three regions, shown in Figure 3-9 [44]:

- The viscous sublayer $y^+ < 5$
- The buffer layer $5 < y^+ < 30$
- The logarithmic or log-law region $30 < y^+ < 300$

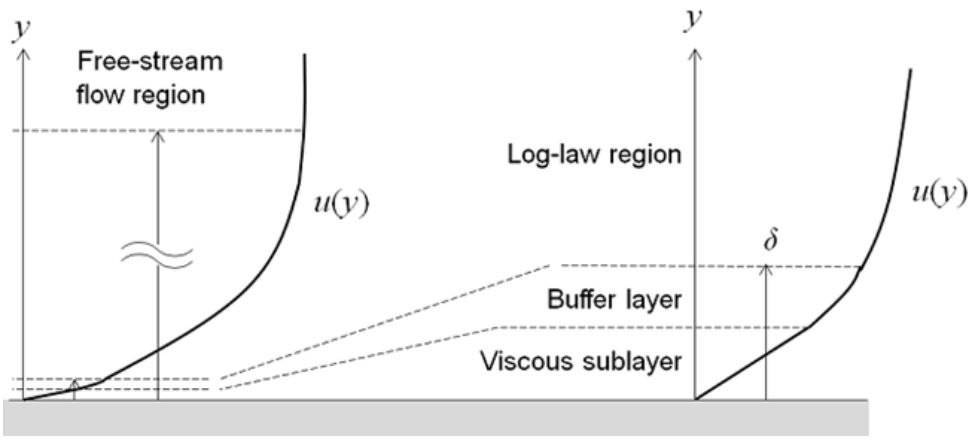

Figure 3-9 Turbulent boundary layer regions [45]

Viscous forces dominate the viscous sublayer. The mean flow velocity in this region is a linear function of the distance from the wall, and $u^+ = y^+$ [48]. Outside the viscous sublayer, the buffer layer occurs. In this region, the transition to turbulent flow begins. There is no analytical solution for the velocity in this region. The flow in the logarithmic region is fully turbulent. The average flow velocity in this region is related to the logarithm of the distance to the wall:

$$u^+ = \frac{1}{\kappa}\ln(y^+) + B \tag{3-18}$$

Where κ is von Karman's constant (≈0,41) and B is a constant (≈5,1).
Figure 3-10 shows the actual velocity profile through the boundary layer (red), and the approximations for the viscous layer and the log-law region.

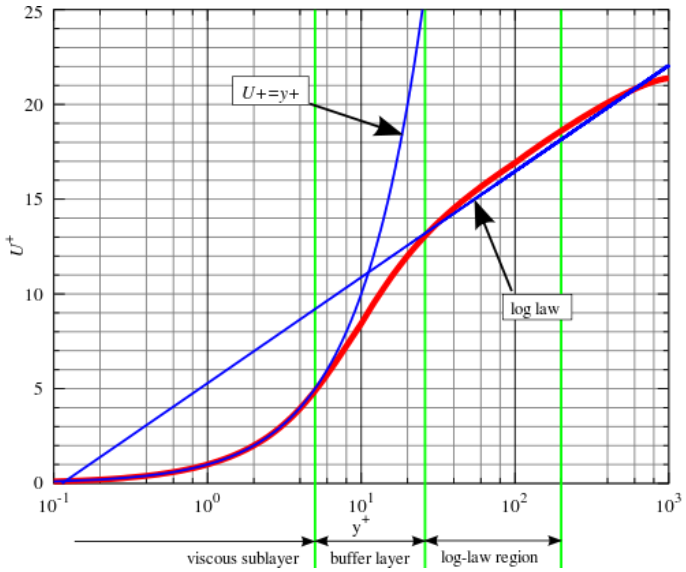

Figure 3-10 Boundary layer velocity profile [49]

Fully resolving the viscous sublayer is computationally expensive. Since the thickness of the layer is so small, a very fine mesh is needed. This can be avoided by the use of so-called wall functions. The center of the first cell is then placed in the log-law region and an analytical solution is assumed for the velocity in the viscous layer.

It is important to check the y^+ values after running a simulation. To make sure that the center of the first cell actually is in the log-law region, the y^+ should be above 30 and below approximately 300. The y^+ can be calculated by running the yPlusRAS utility in OpenFOAM. Note that yPlusRAS computes a quantity, called y^*, which is comparable to y^+. k is the turbulent kinetic energy and C_μ is a constant (≈0,09).

$$y^* = \frac{C_\mu^{0,25} \cdot \sqrt{k} \cdot y_p}{\nu} \tag{3-19}$$

4. Modelling approach

This chapter contains a description of how the simulation was performed, including geometry, meshing and boundary conditions.

4.1. Quasi-transient analysis

A transient analysis of the under-expanded jet in our case requires significant amounts of computational power. Time steps in the order of 10^{-7} to 10^{-9} seconds must be used to resolve the complex shock structures that occurs in the flow downstream of the orifice. Calculating just one millisecond of the flow, can take multiple hours on a standard laptop, depending on the grid size.

On the other hand, the heat transfer in the pipe is a much slower process. At least 20-30 seconds needs be simulated to see how the low temperature at the inner wall affects the temperature distribution in the entire pipe.

Ideally, one would simulate the problem as a conjugate heat transfer problem and create a CFD model that includes both the fluid and the solid region. Such a model could have been simulated using a modified version of the chtMultiRegionFoam solver, as explained in section 3.3.1. However, due to the time step limitations explained above, the computational time for such a model would be unacceptably long.

The problem was therefore divided into two parts. The first part considers only the fluid flow inside the pipe, while the other part deals with the heat transfer in the pipe. This approach is illustrated in Figure 4-1. Liu et al. [11] used a comparable approach when they encountered a similar time step problem related to dispersion of CO_2 from a jet.

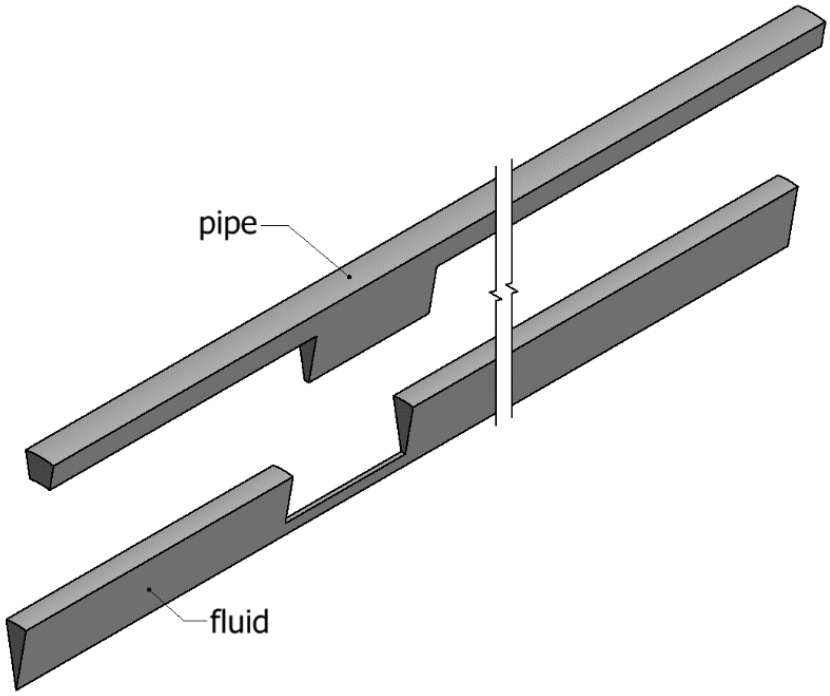

Figure 4-1 Fluid and pipe region are simulated separately

The temperature field calculated on the fluid surface was used as boundary condition on the inner pipe wall. However, since the system backpressure rises at a rate of 10 bar per minute [3], the temperature field in the fluid is expected to change over time. To approximate the effect of the pressurization rate on the temperature field, a quasi-transient approach was chosen.

A set of cases with different outlet pressures were defined, shown in Table 4-1. Each case was run with a constant outlet pressure until a converged wall temperature was reached. The temperature on the inner pipe wall was then updated with the new temperature field. The outlet pressure was increased in steps of 1 bar for the first 30 seconds, as shown in Figure 4-2 and Table 4-1. Hence, the temperature field in the pipe was updated every sixth second.

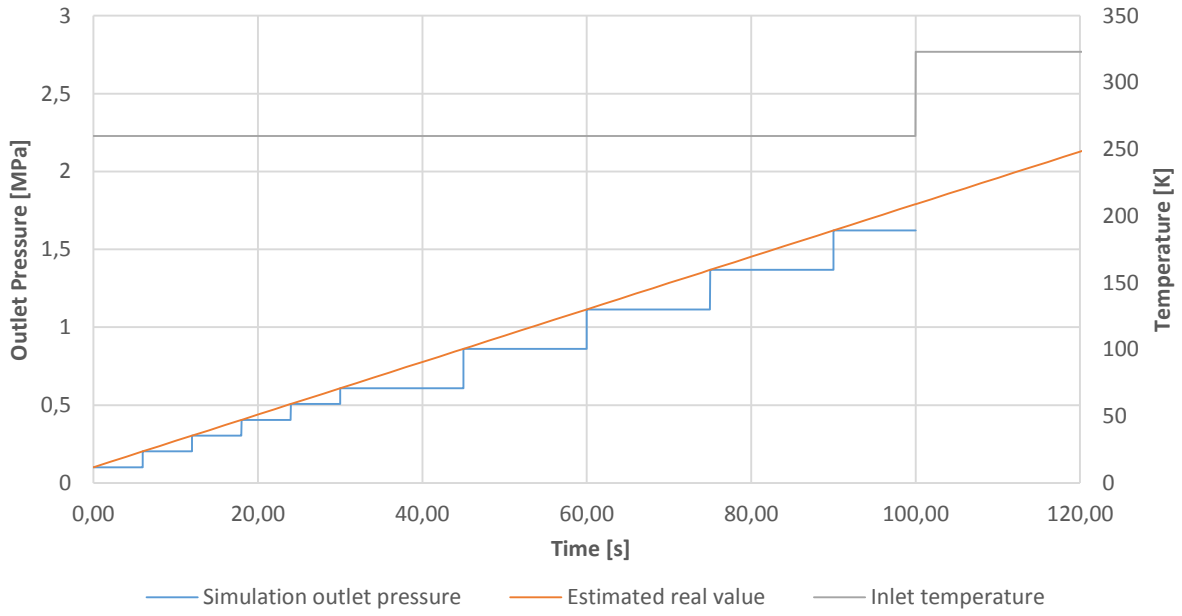

Figure 4-2 Quasi-transient approach

The outlet pressure was then increased in steps of 2,5 bar for the next 70 seconds, which leads to an updated temperature field in the pipe every 15th second.

Table 4-1 Cases

Case number	Outlet pressure [bar]	Time [s]	NPR
1	1	0	235
2	2	6	117,5
3	3	12	78,3
4	4	18	58,8
5	5	24	47
6	6	30	39,2
7	8,5	45	27,6
8	11	60	21,4
9	13,5	75	17,4
10	16	90	14,7

In addition to the 10 cases defined in Table 4-1, a case with 235 bar upstream pressure and 230 bar downstream pressure was simulated. It is referred to as case 11. Due to the low NPR of this case, the flow through the orifice does not reach sonic velocity. The case was therefore simulated using the chtMultiRegionFoam solver.

4.2. Mesh

The mesh was created using the blockMesh utility that comes with OpenFOAM. BlockMesh creates the mesh from input data given in the blockMeshDict, located in the constant/polyMesh folder. The geometry domain is decomposed into a set of three-dimensional hexahedral blocks. Eight vertices define each block. Inside each block, the number of cells and refinement in each spatial direction must be defined [50].

Creating a mesh with blockMesh can be done in the following way:

- Define vertices with coordinates
- Define blocks based on the vertices
- Define number of cells in each direction and grading inside each block
- Define boundary patches

Our problem is an example of a two dimensional axisymmetric problem. Instead of considering the entire circumference, we only look at a wedge. This is an advantage because it significantly reduces the number of cells in the mesh, and hence the computational time. OpenFOAM supports wedges that are one cell thick and less than 5°, as shown in Figure 4-3.

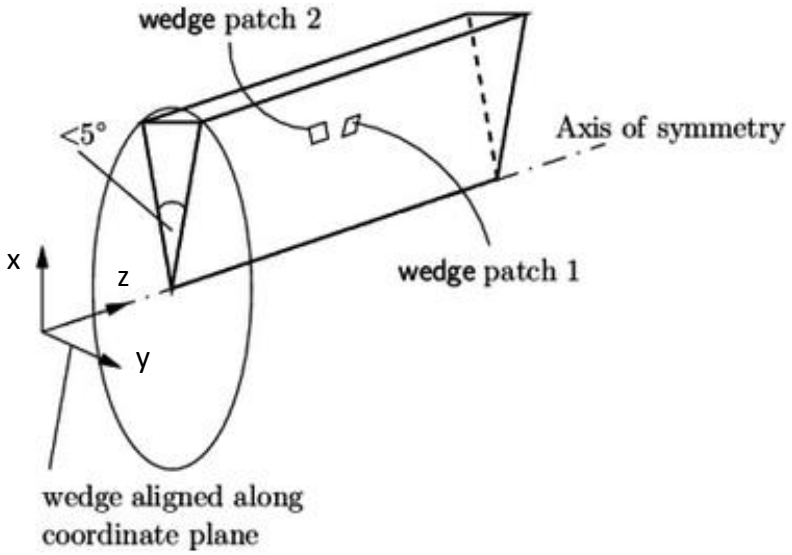

Figure 4-3 Wedge mesh [51]

Two separate meshes were created, one for the fluid part, and one for the pipe. Both were created as wedges. The dimensions are given in Figure 4-4.

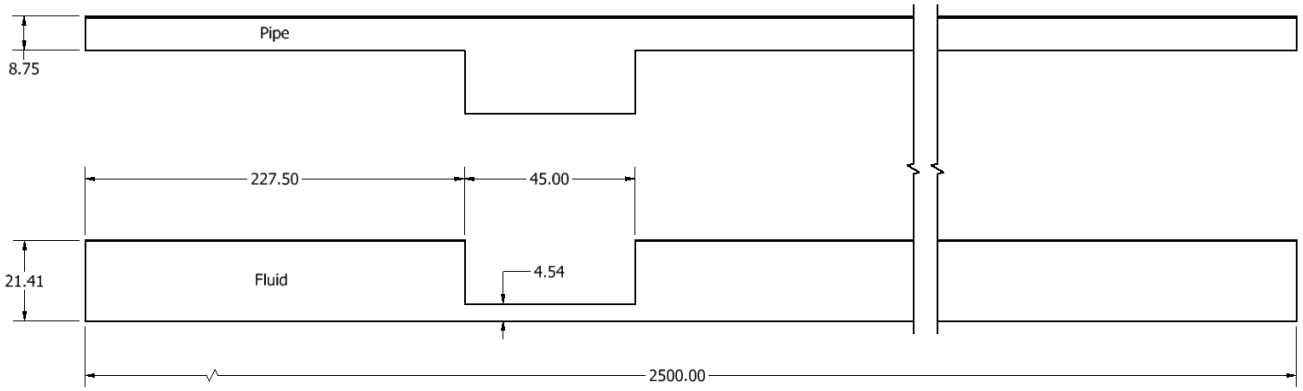

Figure 4-4 Mesh dimensions

4.2.1. Fluid mesh

To enable different refinement levels throughout the mesh, the computational domain was divided into several blocks see Figure 4-5. Vertices that lie on wedge patch 1 (Figure 4-3) are colored red, while the equivalent vertices on wedge patch 2 are written in blue. Refinement blocks along the wall was added to obtain reasonable y^+ values. All the blocks along the wall have equal height, but different refinement levels, due to the large variations in velocity throughout the computational domain.

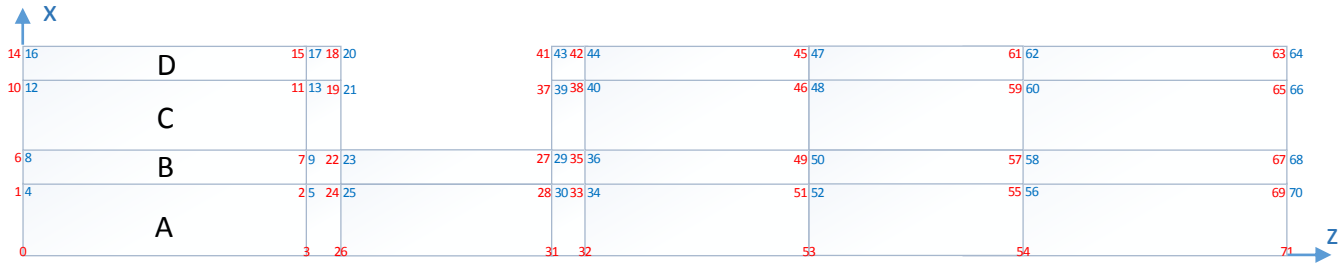

Figure 4-5 Vertices and blocks with numbering for fluid mesh

The wedge cross section is symmetric about the xz-plane, as shown in Figure 4-3. The distance between vertices 14 and 16 in Figure 4-5 was set to 1,8 mm (upper width of the wedge), which leads to an angle between the wedge patches of 4,814°. All other y-coordinates were calculated based on this length.

Multiple different levels of refinement were used. The smallest cells are found along the walls and in the orifice outlet region. Further downstream, the cell size is increased gradually. All mesh details can be found in the blockMeshDict in Appendix D.2.1.

4.2.2. Pipe mesh

The cell dimensions at the inner wall of the pipe and the surface of the fluid must be equal to enable mapping of the temperature field from the fluid to the pipe. Thus, the number of cells in z-direction of the pipe mesh is directly given from the fluid mesh.

Since there is no need for mesh refinement near the walls, the number of cells in the x-direction was set to eight, which leads to a cell height of 1,2 mm. See Appendix E.2.1 for more details.

4.2.3. Combined mesh for case 11

For case 11, the mesh regions were coupled and the entire mesh was created from one blockMeshDict. The mesh was split into two regions using the splitMeshRegions utility. See Appendix H for details.

4.3. Grid independence test

It is important that the results are independent of the mesh size. A grid independence test was therefore conducted. The test was carried out on the fluid mesh only, since the number of cells in the pipe mesh is dependent on the size of the fluid mesh.

The mesh is divided into five regions in the z-direction, if we disregard the refinement layers on the walls. These regions are referred to by the names shown in Figure 4-6.

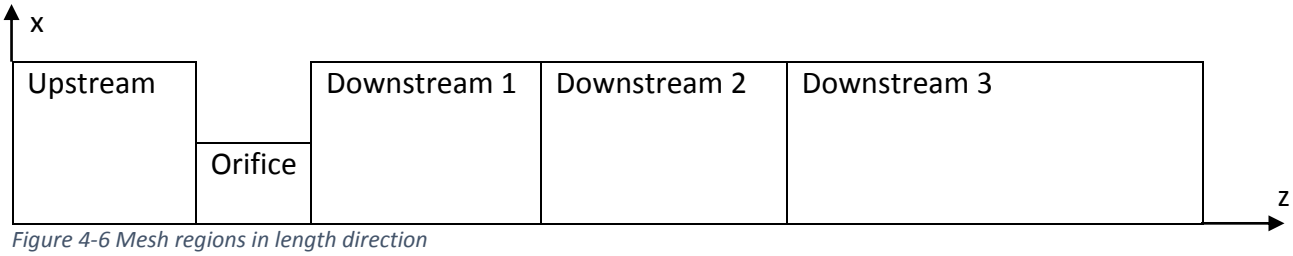

Figure 4-6 Mesh regions in length direction

Four different meshes with number of cells varying from 6300 to 15200 were tested. The k-ω SST turbulence model was applied and equal boundary conditions were used in all cases, see Table 4-4. Table 4-2 shows the number of cells, in the z-direction, for each region, as well as the total number of cells for each of the four meshes.

Table 4-2 Number of cells in z- direction

	Number of cells					Total number of cells	Increase in cell number
Mesh	Upstream	Orifice	Down-stream 1	Down-stream 2	Down-stream 3		
Coarse	20	20	100	75	75	6300	
Coarse2	20	20	100	75	75	8000	27%
Medium	20	30	150	100	100	10880	36%
Fine	20	45	200	150	150	15200	40%

Each of the regions shown in Figure 4-6, are further divided into four blocks in the x-direction, referred to as A,B,C and D, see Figure 4-5. Due to limitations in the blockMesh utility, the number of cells in the x-direction of each block must be equal throughout the entire mesh. The number of cells in x-direction for each of the four blocks are listed in Table 4-3.

Table 4-3 Number of cells in x-direction

Block	Coarse		Coarse2, Medium, Fine	
	Number of cells	Cell height [mm]	Number of cells	Cell height [mm]
A	2	1,52	3	1,01
B	5	0,30	5	0,30
C	10	1,54	15	1,02
D	5	0,30	5	0,30

To avoid instability in the simulations, the cell length is gradually increased downstream of the orifice. Figure 4-7 shows how the length of the cells varies with z-coordinate.

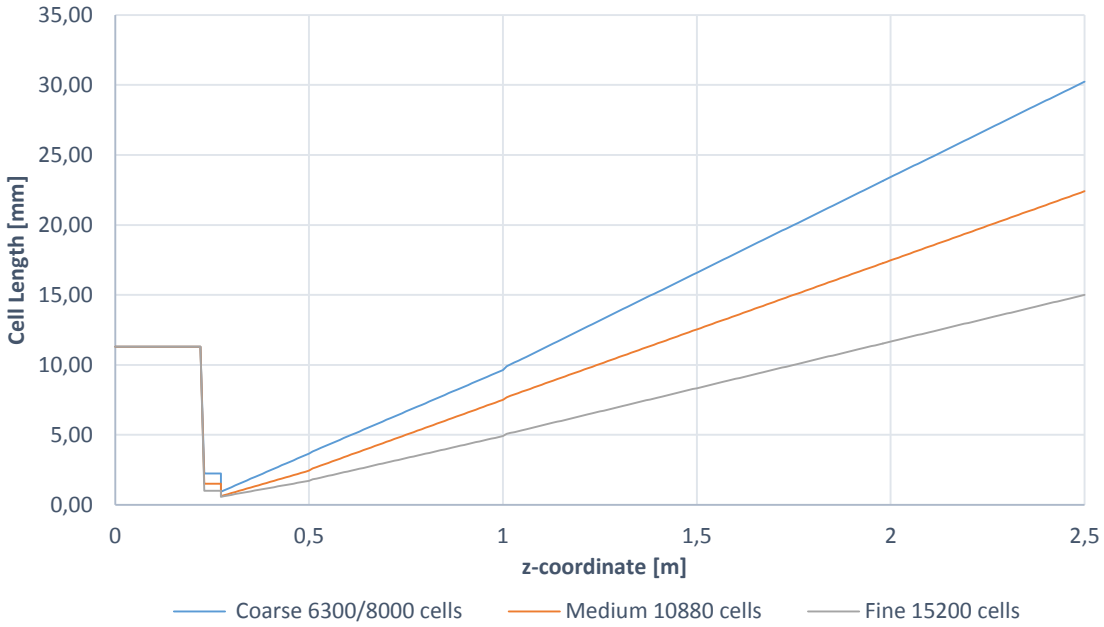

Figure 4-7 Cell length as function of z-coordinate

Results were recorded after 5 ms of flow time. Comparisons of the temperature and the axial velocity, measured along the centerline, are shown in Figure 4-8 and Figure 4-9 respectively. Results from all the four meshes follows the same trend and the accuracy increases with number of cells, as expected. The fine mesh captures fluctuations in the flow further downstream than the medium and the coarse meshes. However, the medium and the fine mesh seem to perform almost identically.

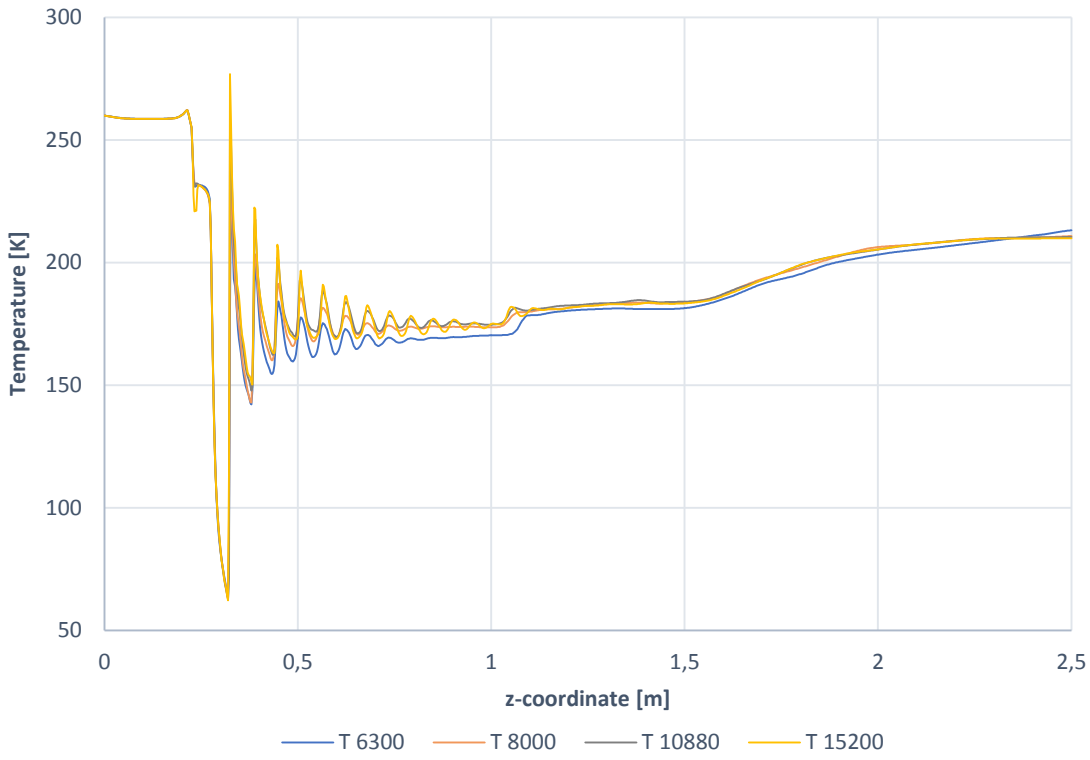

Figure 4-8 Centerline temperature

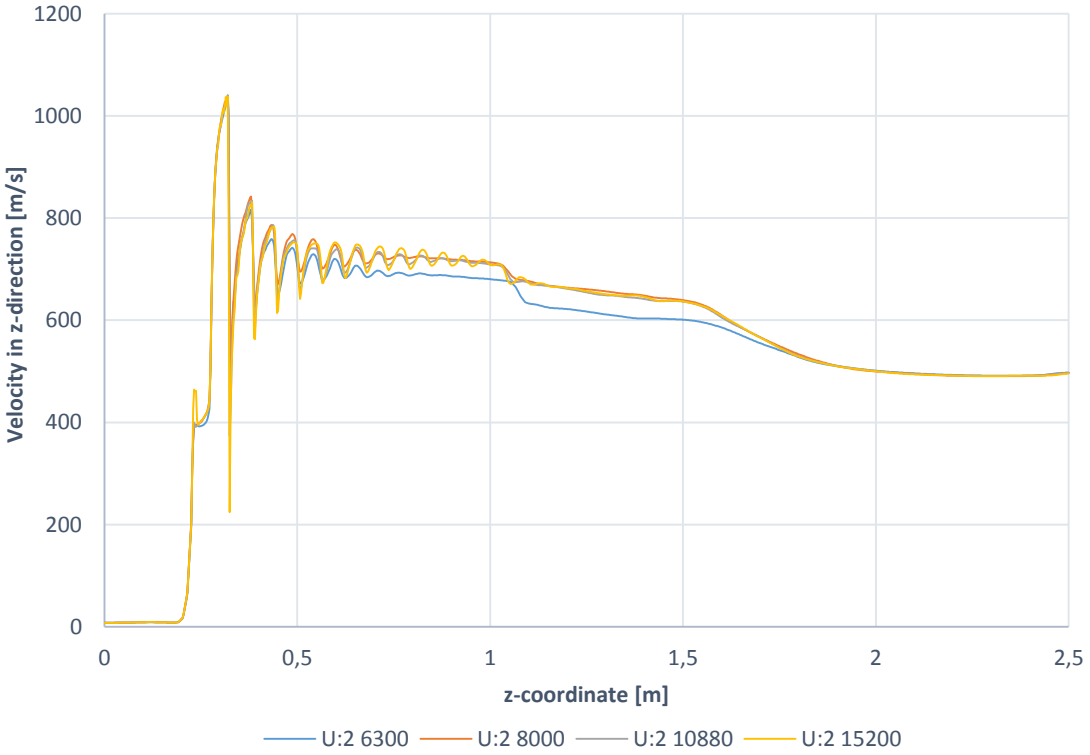

Figure 4-9 Velocity in z-direction measured along centerline

Figure 4-10 shows the temperature measured at the fluid surface. The coarsest mesh predicts the highest temperature, while the medium and the fine mesh predicts the lowest temperatures. More and more fluctuations are captured as the number of cells increases.

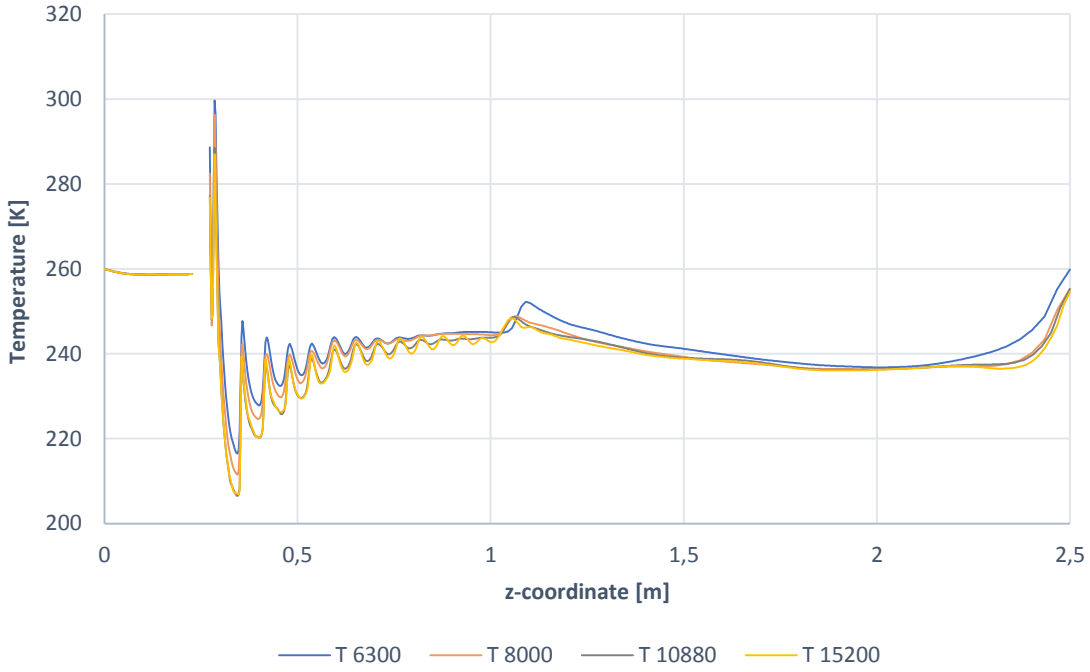

Figure 4-10 Fluid surface temperature

The difference in surface temperature going from the medium to the fine mesh, as shown in Figure 4-11, is well below ±1%.

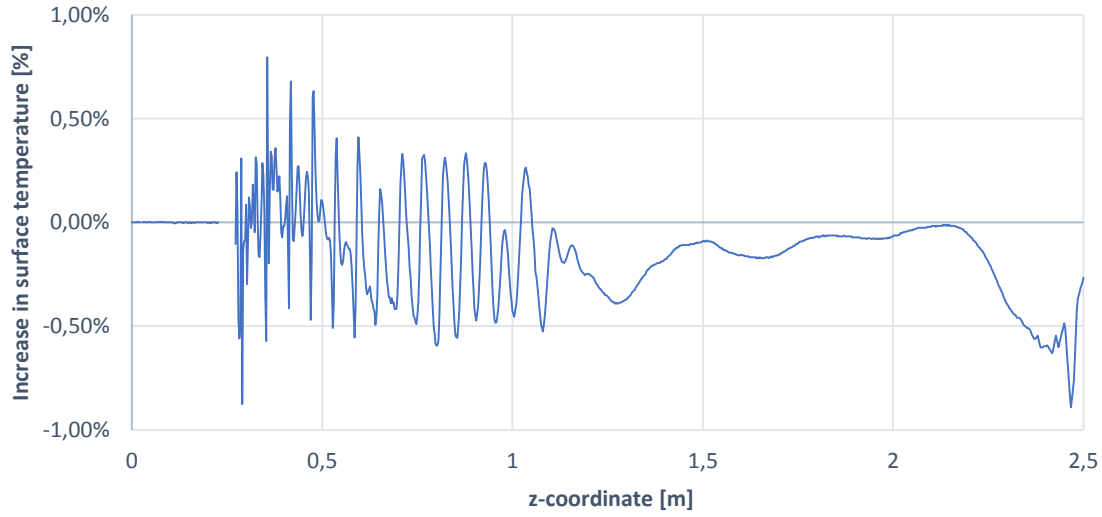

Figure 4-11 Difference between medium and fine mesh

The medium mesh is therefore selected for further simulations. Output from the checkMesh utility, for the medium mesh, is available in Appendix B.3.

4.4. Boundary conditions

The mesh surface must be divided into different regions, also known as patches, to enable specification of boundary conditions. Figure 4-12 shows the patches of both the fluid and the pipe mesh.

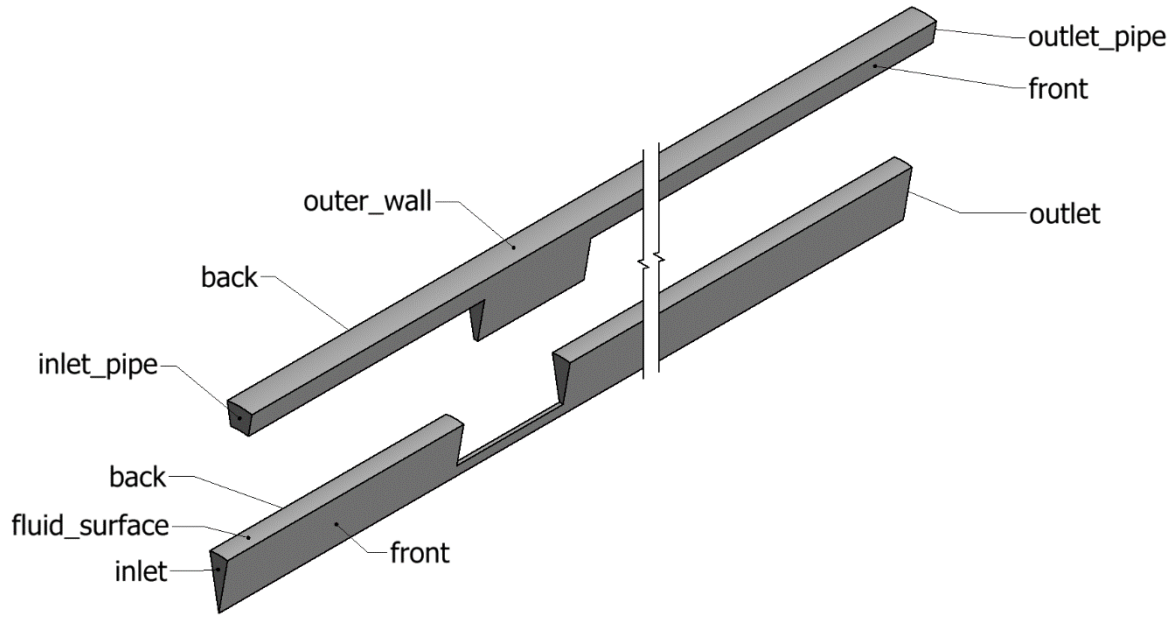

Figure 4-12 Patches

4.4.1. Fluid

The boundary condition for each patch is listed in Table 4-4. At the inlet, the pressure is known to be 235 bar ($2{,}38114 \cdot 10^7$ Pa). The velocity is calculated, based on this pressure, using the pressureInletOutletVelocity boundary condition. Inlet gas temperature is set to 260 K (-13°C). The pressure at the outlet is initially set to 1 bar (101325 Pa). It is increased as described in section 4.1. The waveTransmissive outlet boundary condition is used to avoid reflection of pressure waves that occur downstream of the orifice.

At the surface of the fluid, the no-slip condition is specified for the velocity and zeroGradient for the pressure. The wall is assumed adiabatic, which is reasonable since the simulation only lasts approximately 15-20 ms, see section 4.6.

Table 4-4 Boundary conditions fluid

Patch	**Boundary Condition**		
	U	p	T
inlet	pressureInletOutletVelocity uniform (0 0 0);	fixedValue uniform $2{,}38114 \cdot 10^7$	fixedValue uniform 260
outlet	inletOutlet inletValue uniform (0 0 0) value uniform (0 0 0)	waveTransmissive gamma 1,4 fieldInf uniform 101325 lInf 2 value uniform 101325	zeroGradient
fluid_surface	fixedValue uniform (0 0 0);	zeroGradient	zeroGradient
front	wedge	wedge	wedge
back	wedge	wedge	wedge
internalField	uniform (0 0 0);	uniform 101325	uniform 260

In reality, the valve/orifice assembly separates the high- and the low-pressure region of the pipe. The internal pressure field should therefore be equal to the inlet value of 235 bar all the way up to the orifice, while the initial pressure in the downstream region should be 1 bar. This was obtained by using the setFields utility, which is used to change the internal pressure field. In general, the setFields utility enables the user to select any region of the mesh and set new values for selected fields in this region. The resulting initial pressure field is shown in Figure 4-13. See Appendix D.3.5 for details.

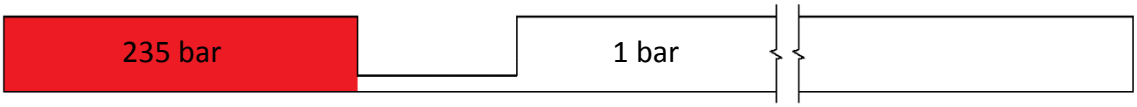

Figure 4-13 Pressure field upstream and downstream of the orifice

The turbulence model, k-ω SST, requires two input parameters, namely k and ω. To estimate these parameters we begin by estimating the turbulent intensity. According to [52], internal pipe flow is a medium turbulent case, with turbulent intensity of 1-5%. A turbulent intensity of 2 % at the inlet was assumed. Furthermore, the turbulent length scale can be estimated to 3,8% of the hydraulic diameter of the pipe [53].[1]

$$l_{turb} = 0{,}038 \cdot d_{hyd} = 0{,}038 \cdot 42{,}82 = 1{,}63\ mm \qquad (4\text{-}1)$$

The turbulent kinetic energy, k, can be estimated from the turbulence intensity and the mean inlet flow velocity. A plot of the velocity at the inlet is shown in Figure 4-14. The average value was calculated to 17,16 m/s using the patchAverage post-processing utility.

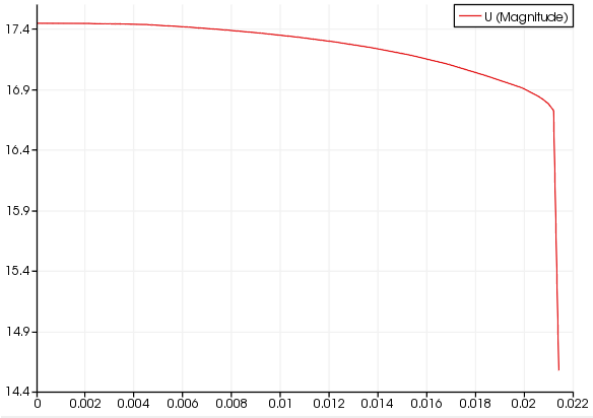

Figure 4-14 Mean flow velocity at inlet, case 1

Hence, the turbulent kinetic energy becomes:

$$k = \frac{3}{2}(u_{mean} \cdot I)^{2} = \frac{3}{2}(17{,}16 \cdot 0{,}02)^{2} = 0{,}177 \qquad (4\text{-}2)$$

The specific turbulent dissipation rate, ω, was calculated from k and l_{turb}, see equation (4-3).

[1] Hydraulic diameter is equal to the diameter for circular pipes [54]

$$\omega = \frac{\sqrt{k}}{l_{turb}} = \frac{\sqrt{0{,}177}}{1{,}627 \cdot 10^{-3}} = 258{,}6 \quad (4\text{-}3)$$

All boundary conditions for turbulence parameters k and ω are listed in Table 4-5. The values specified for the wall functions are only used as initial guesses.

Table 4-5 Turbulence boundary conditions

Patch	**Boundary Condition**	
	k	ω
inlet	inletOutlet inletValue $internalField value $internalField	inletOutlet inletValue $internalField value $internalField
outlet	inletOutlet inletValue $internalField value $internalField	inletOutlet inletValue $internalField value $internalField
fluid_surface	compressible::kqRWallFunction value $internalField	compressible::omegaWallFunction Value $internalField
front/back	wedge	wedge
internalField	uniform 0,177	uniform 258,6

The sample utility was used to extract the temperature field from the fluid_surface patch of each finished simulation. The temperature values for each cell were stored in raw format and the output file was used as boundary condition at the inner pipe wall. See Appendix D.3.4 and G.3 for further details.

4.4.2. Pipe

Temperature boundary conditions for the pipe are listed in Table 4-6. Patches are named according to Figure 4-12.

Table 4-6 Boundary conditions pipe

Patch	**Temperature boundary condition**
inlet_pipe	zeroGradient
outlet_pipe	zeroGradient
inner_wall	fixedValue nonuniform List<scalar>

outer_wall	type externalWallHeatFluxTemperature;
	kappa solidThermo;
	Ta uniform 260.0; // ambient temperature [K]
	h uniform 7.9; // heat transfer coeff [W/Km^2]
	value uniform 260; // initial temperature [K]
	kappaName none;
	thicknessLayers (0.050); // insulation thickness [m]
	kappaLayers (0.046); // thermal cond. [W/mK]

front/back	wedge
internalField	uniform 260

The extracted temperature field is specified as a non-uniform list of scalar values. The pipe ends are assumed adiabatic and insulation is specified at the outer wall, according to data from Aker's report [3]. The heat transfer coefficient is taken from [55].

4.4.3. Case 11

Boundary conditions for case 11 are similar to the ones described above. The only difference is that the fluid_surface patch is renamed fluid_to_pipe and the inner_wall patch is called pipe_to_fluid. In addition, the temperature boundary conditions for these two patches are coupled, see Table 4-7.

Table 4-7 Boundary conditions for case 11

Patch	Temperature boundary condition
fluid_to_pipe	type compressible::turbulentTemperatureCoupledBaffleMixed; Tnbr T; kappa fluidThermo; kappaName none; value uniform 260;
pipe_to_fluid	type compressible::turbulentTemperatureCoupledBaffleMixed; Tnbr T; kappa solidThermo; kappaName none; value uniform 260;

4.5. Thermophysical modelling

To ensure that the simulation results are as close to reality as possible, it is important to select appropriate thermophysical models. In OpenFOAM, this can be done in the thermophysicalProperties dictionary, located in the constant folder.

4.5.1. Fluid

The models selected for the fluid region are listed in Table 4-8.

Table 4-8 Thermophysical properties fluid

Keyword	Selected model	Description
type	hePsiThermo	General thermophysical model based on compressibility
mixture	pureMixture	General thermophysical model calculation for passive gas mixtures
transport	const	Constant μ and Prandtl number
equationOfState	perfectGas	Ideal gas equation of state
thermo	hConst	Constant specific heat capacity and enthalpy of fusion
energy	sensibleEnthalpy	All heat leads to temperature change, not including heat of formation

The natural gas consists of 86,75 % methane and several other species, listed in Table 4-9. The combined molar weight is 19,19 g/mole.

Table 4-9 Actual gas composition [3]

Component	**Mole %**	**Molar weight [g/mole]**
Methane (C1) CH_4	86,75 %	16,04
Ethane (C2) C_2H_5	5,71 %	30,07
CO_2	2,78 %	44,01
Propane (C3)	2,59 %	44,1
n-C4	0,7 %	58,12
Nitrogen	0,59 %	14,0067
i-C4	0,32 %	58,12
C6+	0,22 %	72,0642
n-C5	0,18 %	72,15
i-C5	0,16 %	72,15

In the simulations, the gas was modelled as pure methane. This greatly simplifies the estimation of material properties such as viscosity, heat capacity, thermal diffusivity etc. Since the actual gas consists of 86,75% methane, this is assumed to be a reasonable approximation.

The transport properties of the gas were modelled as constant. For Newtonian fluids, the viscosity is assumed to be independent of the stress, thus independent of the pressure [57]. The value of the dynamic viscosity, μ, was set to 10^{-5} Pa·s. Figure 4-15 shows how the value of μ actually varies with temperature.

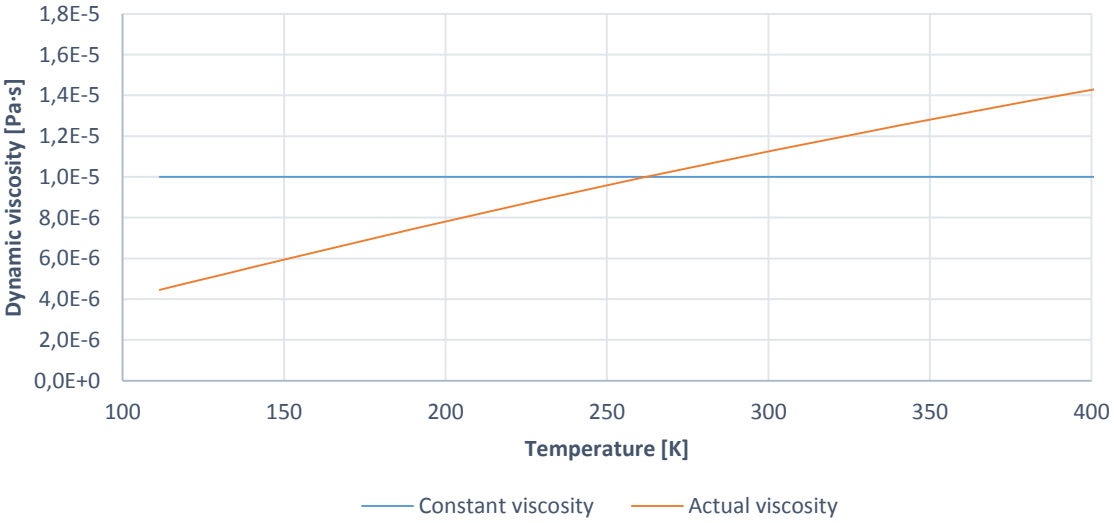

Figure 4-15 Dynamic viscosity as function of temperature [58]

The thermal diffusivity is implicitly specified using the Prandtl number. The Prandtl number is the ratio between the momentum diffusivity and the thermal diffusivity [59].

$$Pr = \frac{\nu}{\alpha} = \frac{c_p \mu}{k_c} \quad (4\text{-}4)$$

ν is the kinematic viscosity and α is the thermal diffusivity given as $\alpha = k_c/\rho c_p$.

According to [60], the Prandtl number for methane varies from 1,04 at 110 K to 0,74 at 300K. A constant value of 0,76 was selected.

The specific heat capacity c_p, and the enthalpy of fusion, H_f, was set to constant values, using the thermodynamic model hConst. Figure 4-16 shows how c_p actually varies with temperature compared to the selected constant value of 2180 J/kgK.

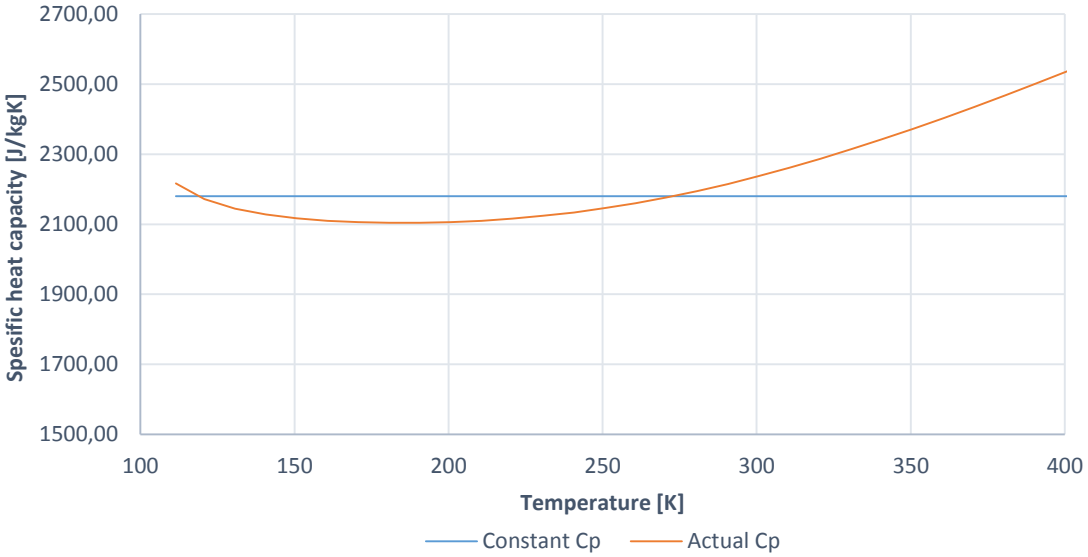

Figure 4-16 Specific heat capacity as function of temperature at constant pressure [58]

H_f was set to 58,99·10^3 J/kg according to [61].

Constant transport properties, c_p, H_f, and Prandtl number are assumed to give sufficient accuracy in our case. However, it is possible to choose more accurate thermophysical models, such as polynomials, JANAF thermodynamic tables, Sutherland's transport equation etc. See reference [62] for details.

For the fluid region of case 11, the only difference is that hePsiThermo is replaced with heRhoThermo.

4.5.2. Pipe

The 2" pipe is made of GD20X, which is a duplex steel containing 22 % chromium, see Figure 1-3. The thermophysical models selected for the pipe are shown in Table 4-10. A molar mass of 55 g/mole was assumed. The thermal conductivity and specific heat capacity was set to 15 W/mK and 485 J/kgK respectively, based on values from Aker's report [3].

Table 4-10 Thermophysical properties pipe

Keyword	Selected model	Description
type	heSolidThermo	General thermophysical model for solids
mixture	pureMixture	General thermophysical model calculation for passive gas mixtures
transport	constIso	Constant thermal conductivity
equationOfState	rhoConst	Constant density
thermo	hConst	Constant specific heat capacity
energy	sensibleEnthalpy	All heat leads to temperature change, not including heat of formation

The thermal diffusivity is specified in the transportProperties dictionary, see Appendix E.2.3. The value is calculated from equation (4-5).

$$\alpha = \frac{k_c}{\rho c_p} = \frac{15}{7800 \cdot 485} = 3{,}965 \cdot 10^{-6} \tag{4-5}$$

4.6. Convergence criterion and probes

Probes were used to log temperature, velocity and other variables as function of time, at different locations, shown in Figure 4-17 and Figure 4-18.

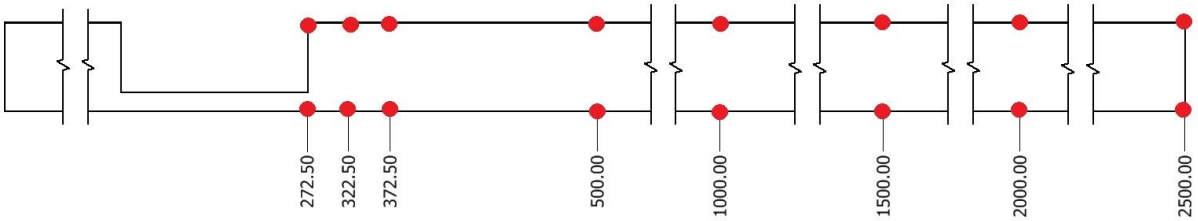

Figure 4-17 Probe locations in fluid

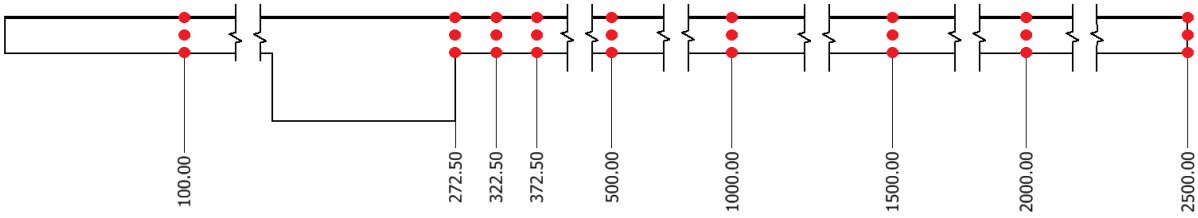

Figure 4-18 Probe locations in pipe

Probes are specified in the controlDict dictionary. For more details, see Appendix D.3.1 and E.3.1.

The fluid surface temperature was monitored and used as a convergence criterion in the fluid simulations. Each case was run until the temperature reached a steady value. Due to

the transient nature of the flow, the temperature never becomes 100% steady. Figure 4-19 shows the temperature development for case 1, measured at different z-coordinates along the fluid surface. It is clear that the fluid surface temperature reaches an approximate steady state, with only minor fluctuations.

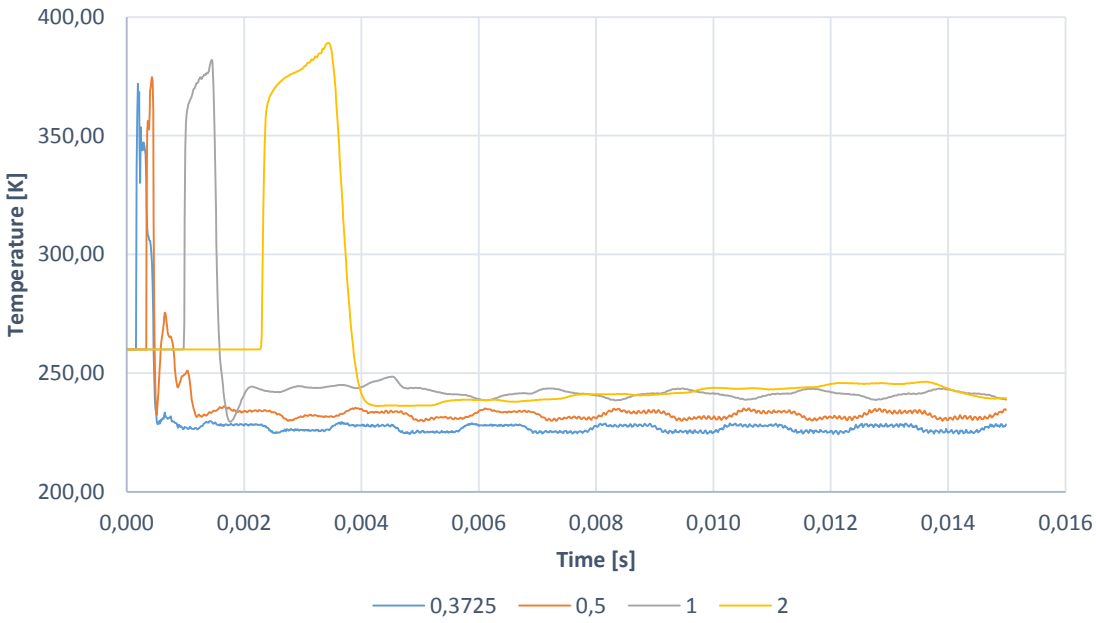

Figure 4-19 Temperature development at fluid surface, case 1

Similar plots for case 7,8 and 10 can be found in Appendix B.2.1.

The time until convergence increases with the backpressure, since the flow develops more slowly at lower NPRs. The convergence time for each case is listed in Table 4-11.

Table 4-11 Convergence time

Case	**Time to convergence of temperature**
1-6	15 ms
7	17 ms
8-9	20 ms
10	22 ms

5. Results and Discussion

This chapter contains the most important results from the simulations. The validity of the results are discussed and sources of uncertainty are pointed out. Detailed results for all cases can be found in Appendix B.

5.1. Fluid surface temperature

Figure 5-1 shows the temperature, measured at the fluid surface, for case 1 to 4, recorded after 15 ms.

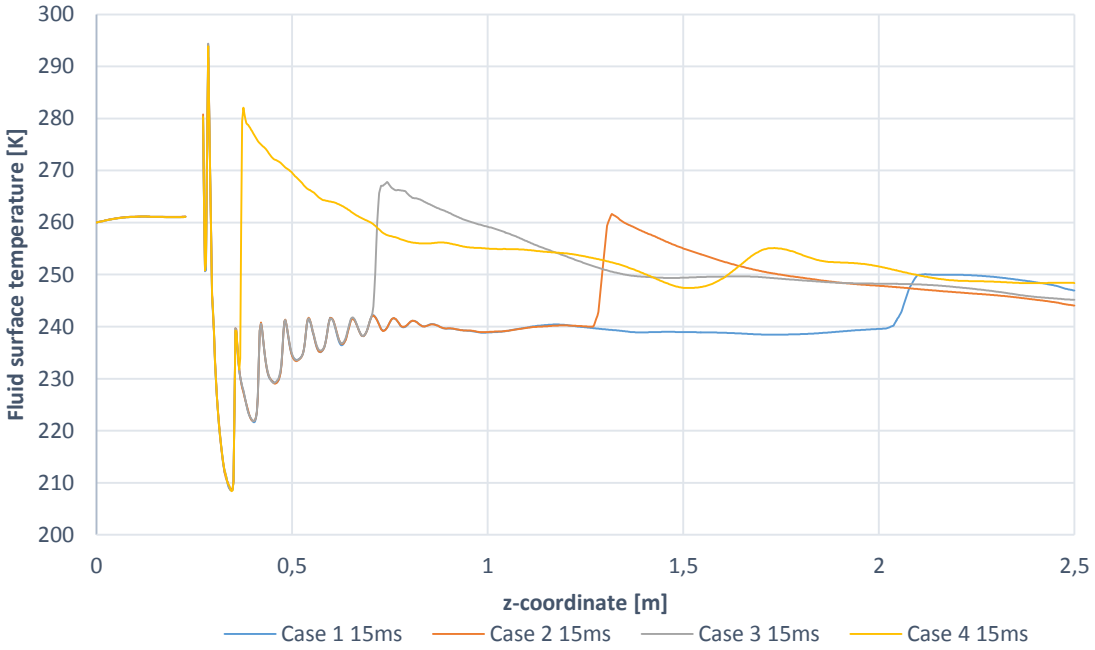

Figure 5-1 Temperature at the fluid surface, case 1 to 4

The temperature profiles for the first four cases seem to follow the same trend. As the backpressure in the system increases, the velocity decreases and hence the temperature profile develops more slowly. It is assumed that the temperature profile for case 2,3 and 4 will become equal if the simulation time is extended. To investigate this, case 3 was run for 30 ms. Figure 5-2 shows the temperature profile at the fluid surface for case 3 at 15 and 30 ms.

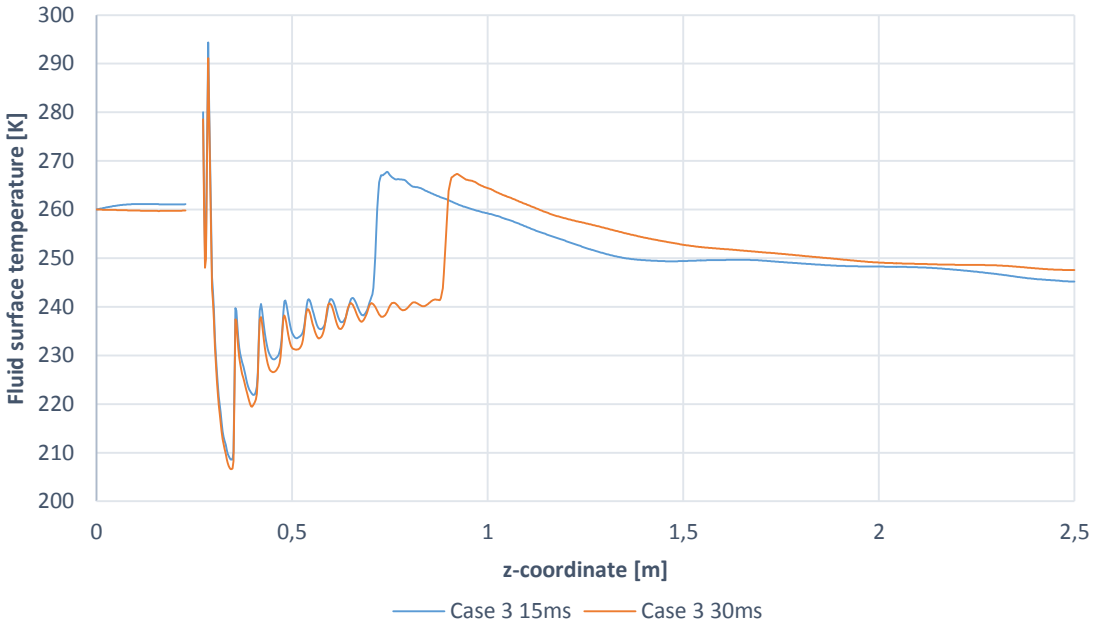

Figure 5-2 Temperature at the fluid surface, case 3

The temperature profile develops as expected. Based on this we assume that it will converge towards the profile of case 1. To save computational time, the temperature profile from fluid case 1 was used for all the first four cases.

This simplification could result in too low temperatures being predicted in the downstream pipe. However, it will not influence the minimum wall temperature, which is almost equal for case 1, 2, 3 and 4.

5.1.1. Time-averaged temperature

Figure 5-3 shows how the unsteady nature of the jet results in fluctuations in the fluid surface temperature downstream of the orifice for case 6 to 10.

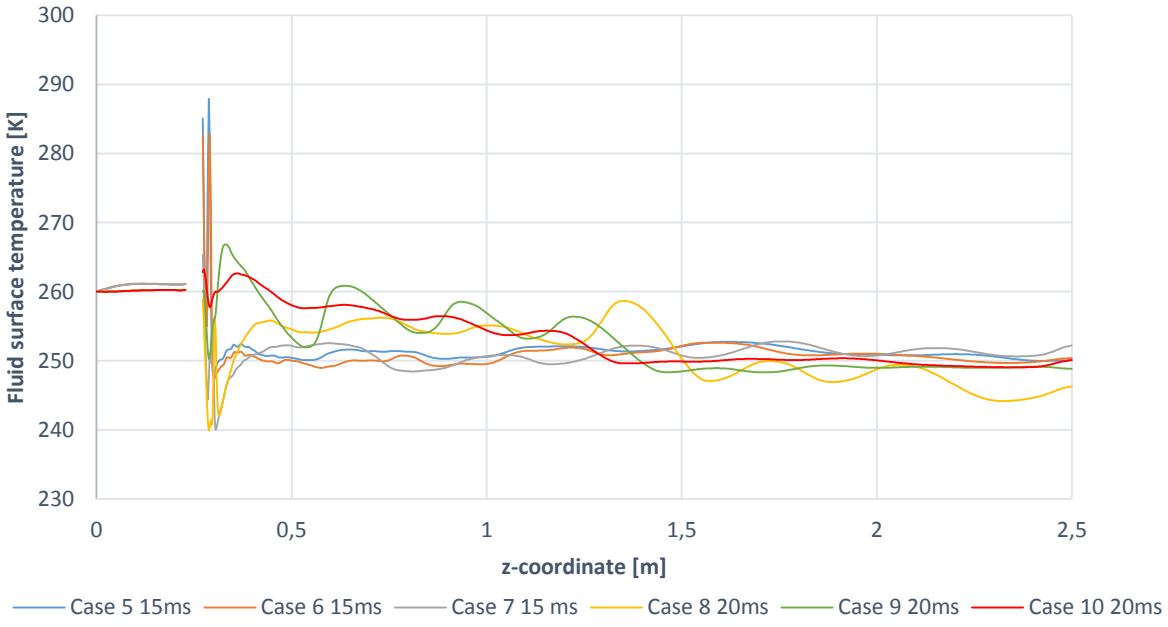

Figure 5-3 Fluctuating temperature at fluid surface, case 6 to 10

To obtain smoother temperature profiles, the temperatures were time-averaged over the time ranges shown in Table 5-1. See Appendix G.2 for details.

Table 5-1 Time averaging range

Case	Time range
1,5,6	13-15 ms
7	15-17 ms
8,9	18-20 ms
10	20-22 ms

The time-averaged temperature profiles are shown in Figure 5-4.

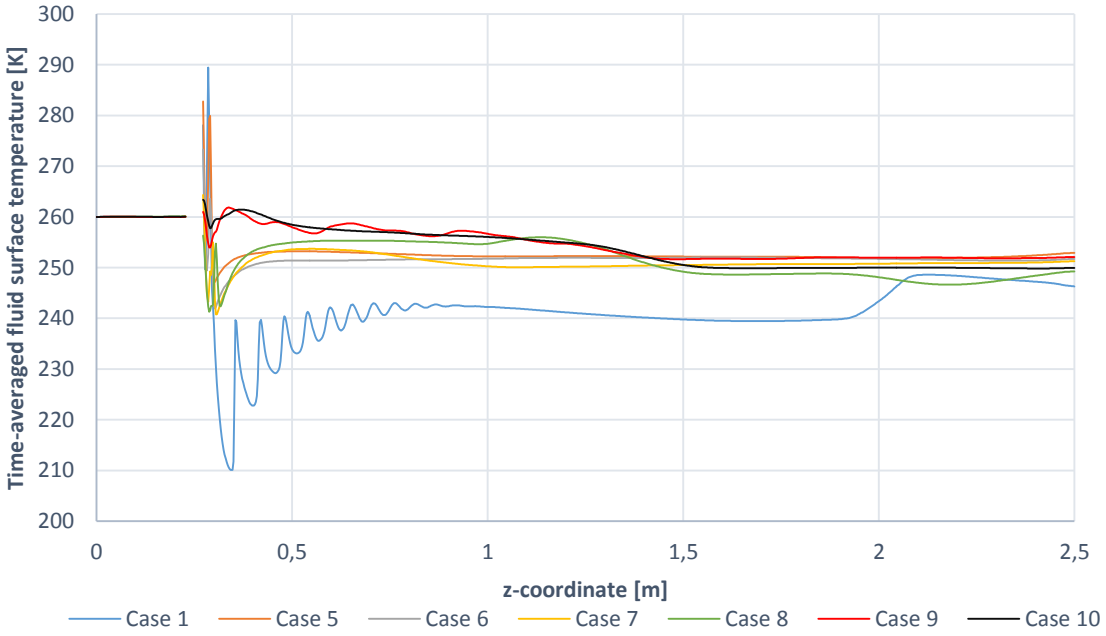

Figure 5-4 Time-averaged temperature profiles

5.1.2. Peng-Robinson equation of state

The behavior of a real gas is often sufficiently modelled using the ideal gas law, which has been applied in this thesis. However, close to the condensation point, the critical point and at high pressures, the deviation between the ideal gas prediction and the real gas behavior becomes significant. To check how this influences our results, case 1 was re-run using Peng-Robinson equation of state. This case is referred to as case 1P.

When using Peng-Robinson, OpenFOAM does not allow the viscosity to be modelled as constant. For this reason, the Sutherland equation was used. In addition, the critical pressure, volume and temperature of the gas, as well as an acentric factor, must be specified. See Appendix C.2 for details. All other parameters are equal to the ones listed in section 4.5.1.

Figure 5-5 shows the temperature at the fluid surface for case 1 and 1P, after 15ms of flow time. The temperature predicted by Peng-Robinson is clearly lower than the one predicted by the ideal gas law.

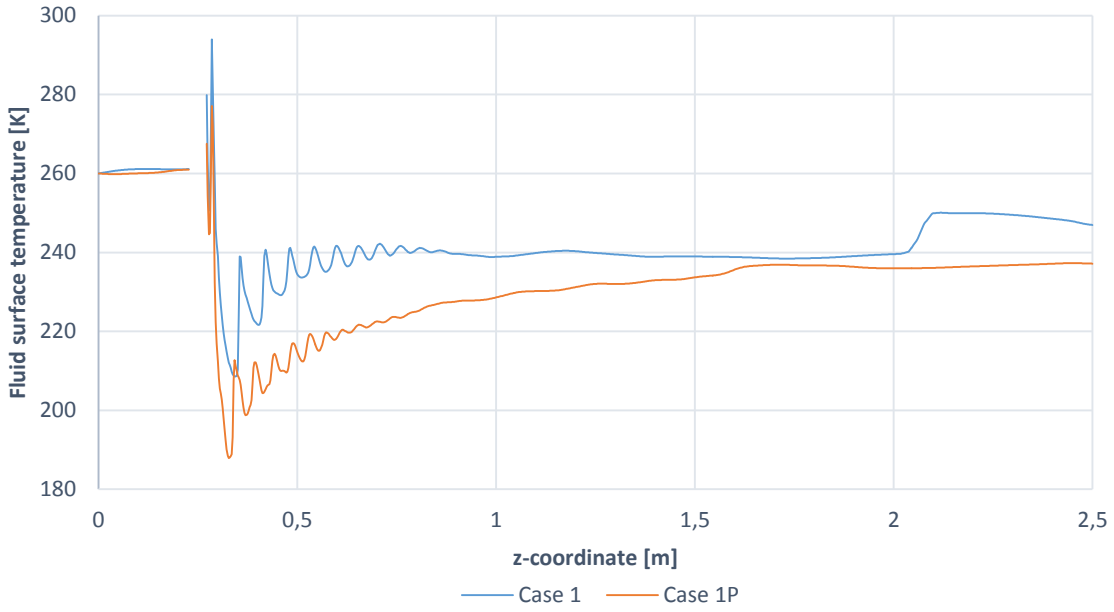

Figure 5-5 Fluid surface temperature comparison case 1 and case 1P, 15 ms

Ideally, all cases should be re-run using the Peng-Robinson equation of state. However, due to limited time, this was not possible. Case 5,7 and 9 were selected and re-run using Peng-Robinson. They are referred to as case 5P,7P and 9P respectively. A comparison between case 5 and 5P is shown in Figure 5-6. Results from case 7 and 7P are compared in Figure 5-7, while Figure 5-8 shows the results from case 9 and 9P. It is evident that the Peng-Robinson equation of state predicts lower temperatures than the ideal gas law for all cases.

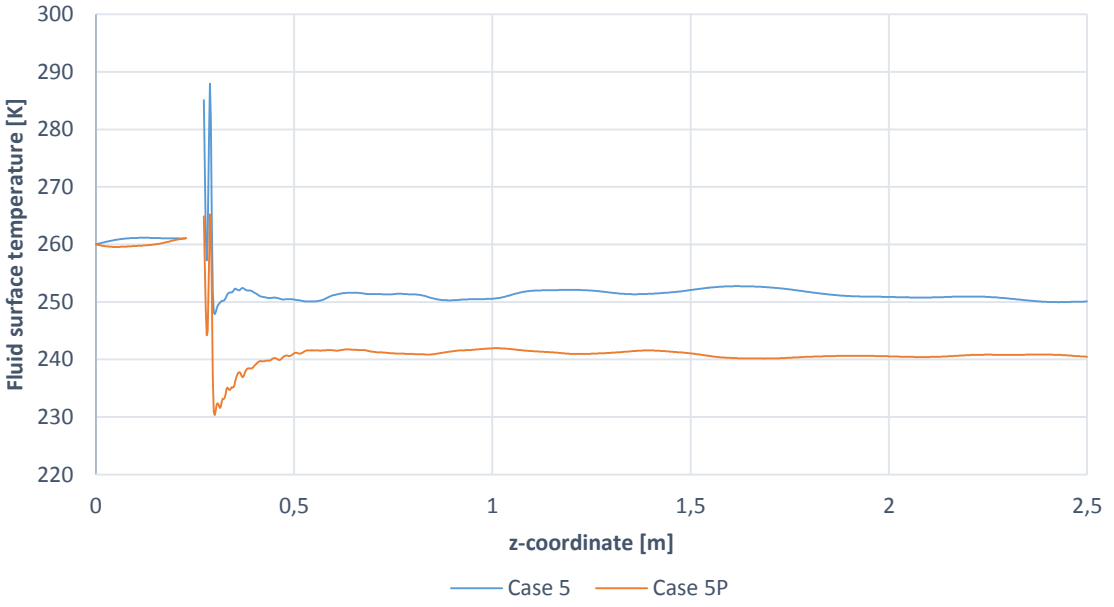

Figure 5-6 Fluid surface temperature comparison case 5 and 5P, 15 ms

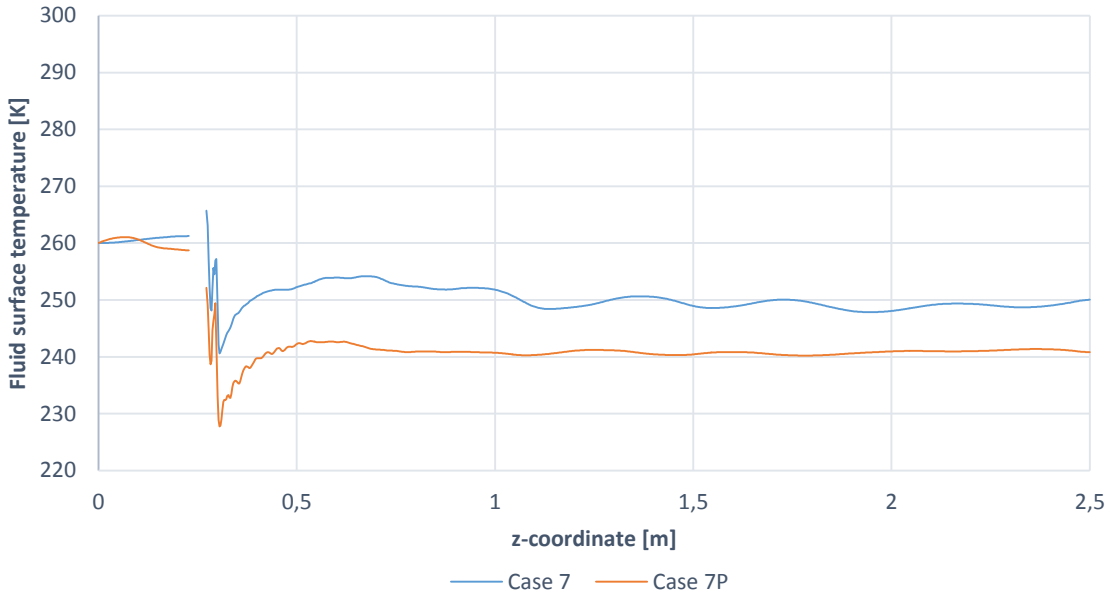

Figure 5-7 Fluid surface temperature comparison case 7 and 7P, 17 ms

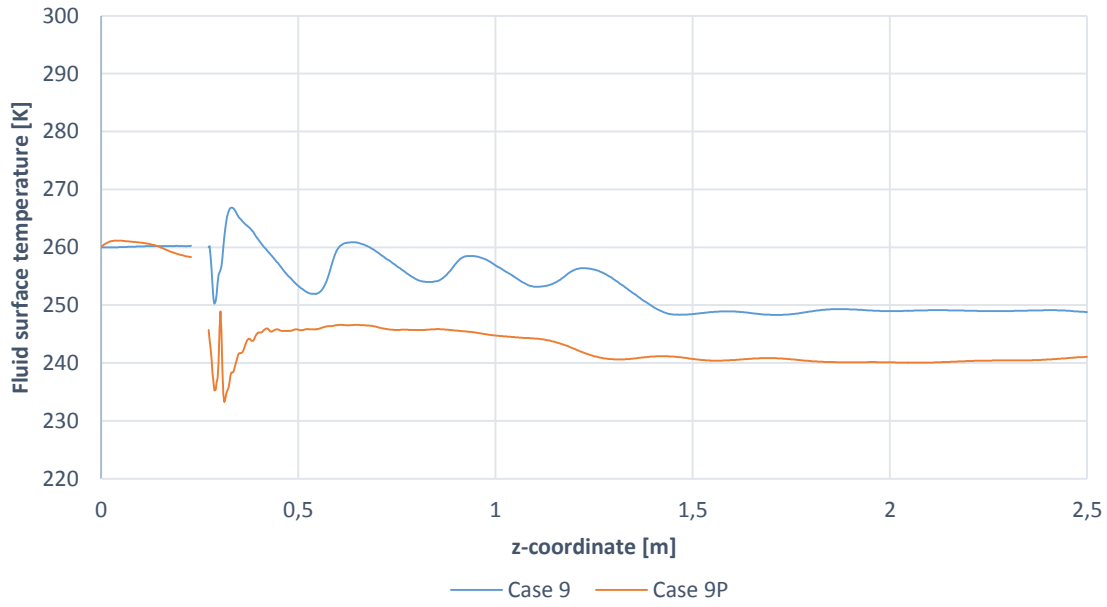

Figure 5-8 Fluid surface temperature comparison case 9 and 9P, 20 ms

The time-averaged temperature profiles for all Peng-Robinson cases are shown in Figure 5-9. See Table 5-1 for time averaging ranges.

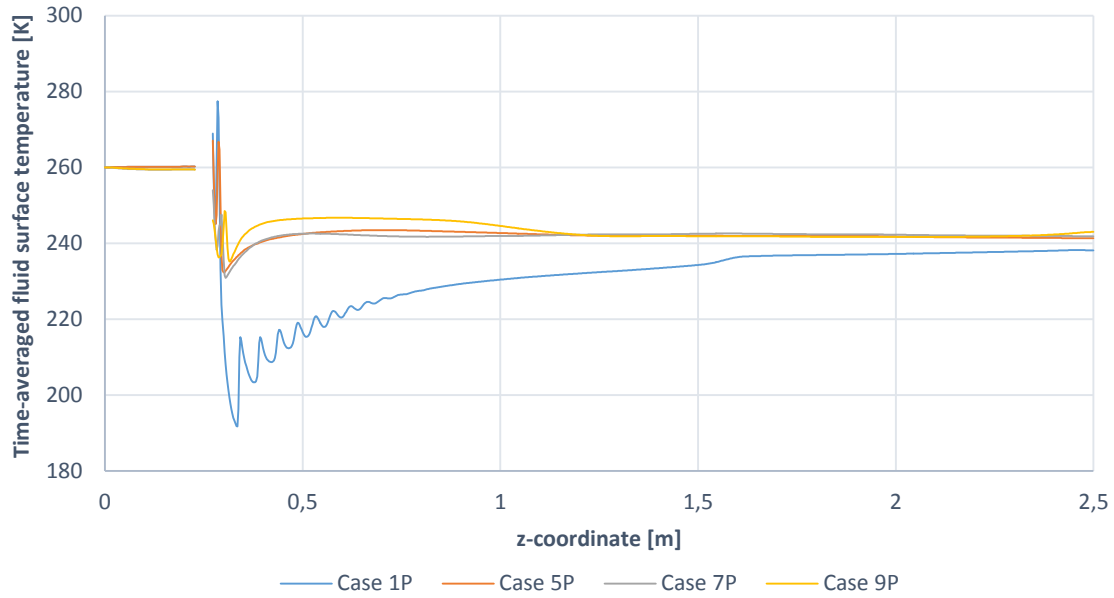

Figure 5-9 Time-averaged temperature profiles for Peng-Robinson cases

5.2. Pipe temperature

The pipe temperature simulations are divided into two cases, see Table 5-2. In Case A, the temperature fields obtained from the ideal gas law are used, while the temperature fields in case B come from the simulations with Peng-Robinson. The inner wall temperature of case B is not updated as frequently as the one in case A, since there was not enough time to re-run all the cases. However, since case B takes into account the real gas effects, and predicts the lowest pipe temperature, it is assumed that it provides the most accurate results.

Table 5-2 Pipe temperature boundary condition

Time [s]	Temperature fields Case A (Ideal Gas)	Temperature fields Case B (Peng-Robinson)
0	1	1P
6	1	1P
12	1	1P
18	1	1P
24	5	5P
30	6	5P
45	7	7P
60	8	7P
75	9	9P
90	10	9P

5.2.1. Case A

Figure 5-10 shows how the temperature in the pipe develops for case A. The pipe upstream of the orifice is almost not affected by the cooling, while the downstream pipe experiences a significant temperature drop.

Just after the exit of the orifice, a small portion of gas is compressed by the expansion waves of the jet, creating a hot region. Refer to the fluid results in Appendix B for details. This hot region, which is clearly seen in Figure 5-10, becomes a barrier between the hot upstream and the cold downstream region.

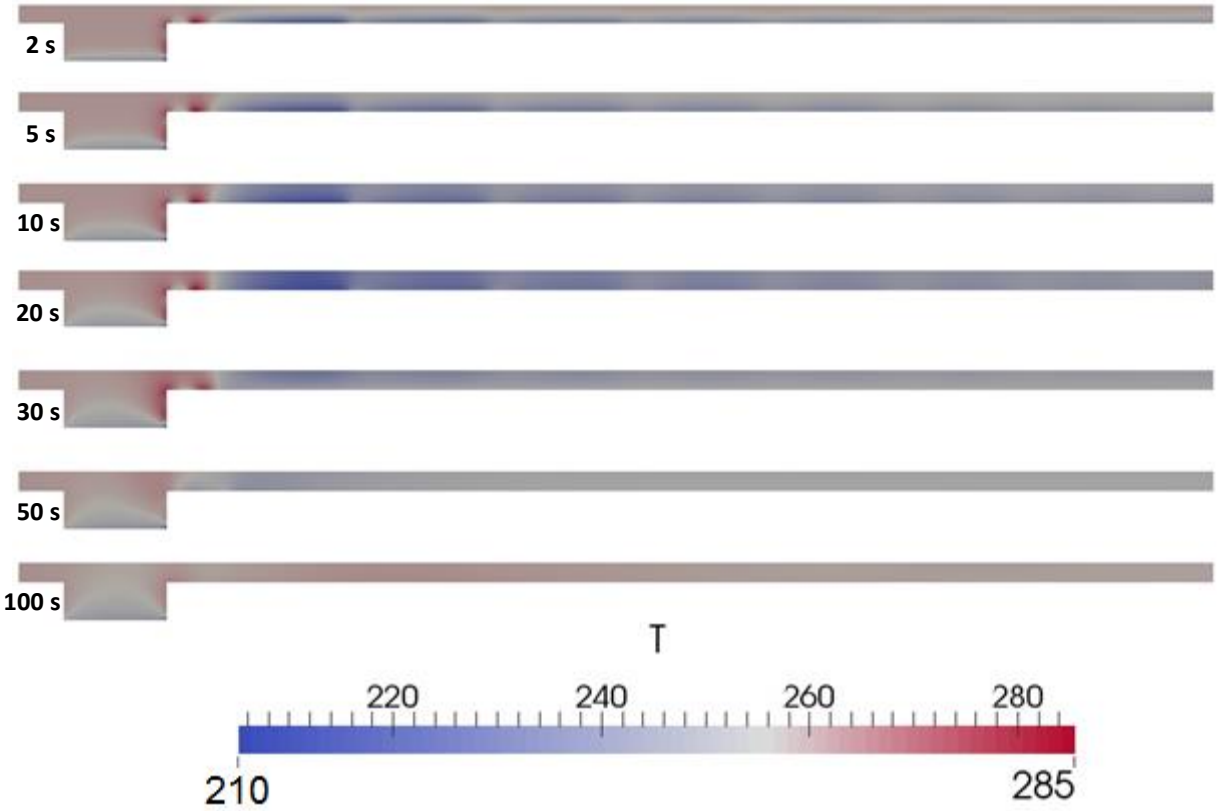

Figure 5-10 Temperature development in pipe, case A

The temperature, as a function of time measured at different probes in the middle of the pipe wall, is shown in Figure 5-11. See Figure 4-18 for probe locations.

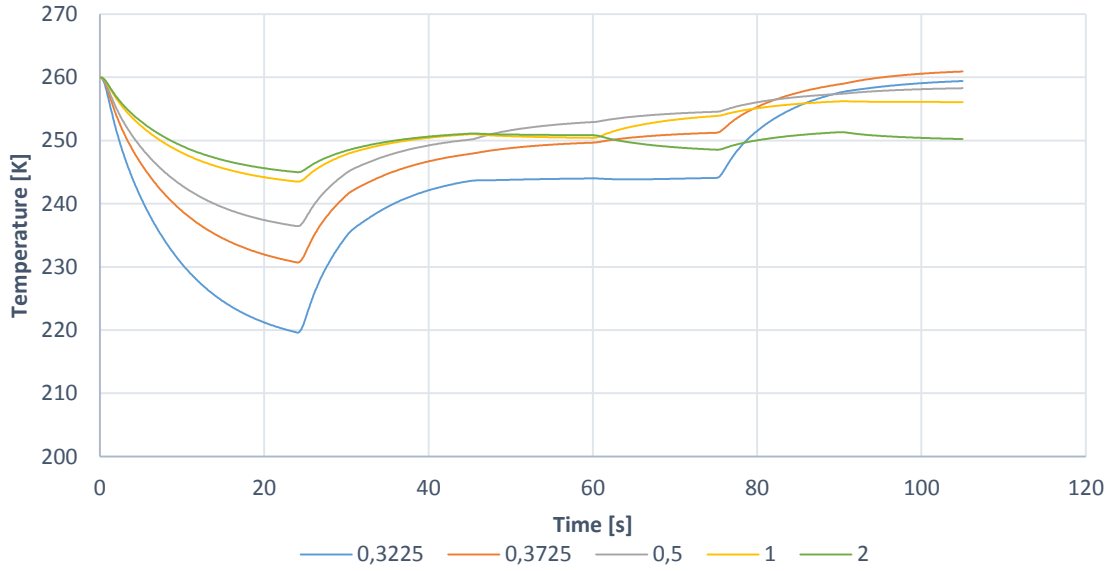

Figure 5-11 Temperature measured at probes in the middle of pipe wall, case A

The lowest temperature that occurs in the pipe is 210,1 K (-62,9°C), see Figure 5-4.

At the fluid surface, before time averaging the temperature profiles, the lowest temperature is 208,4 K (-64,6°C), see Figure 5-1.

5.2.2. Case B

Figure 5-12 shows the temperature development in the pipe for case B. We see that the temperature distribution is similar to the one showed in Figure 5-10, even though the temperatures are lower. The hot region at the orifice exit prevents low temperatures from spreading to the upstream pipe. Figure 5-13 displays how the temperature varies with time, at different probe locations in the middle of the pipe wall.

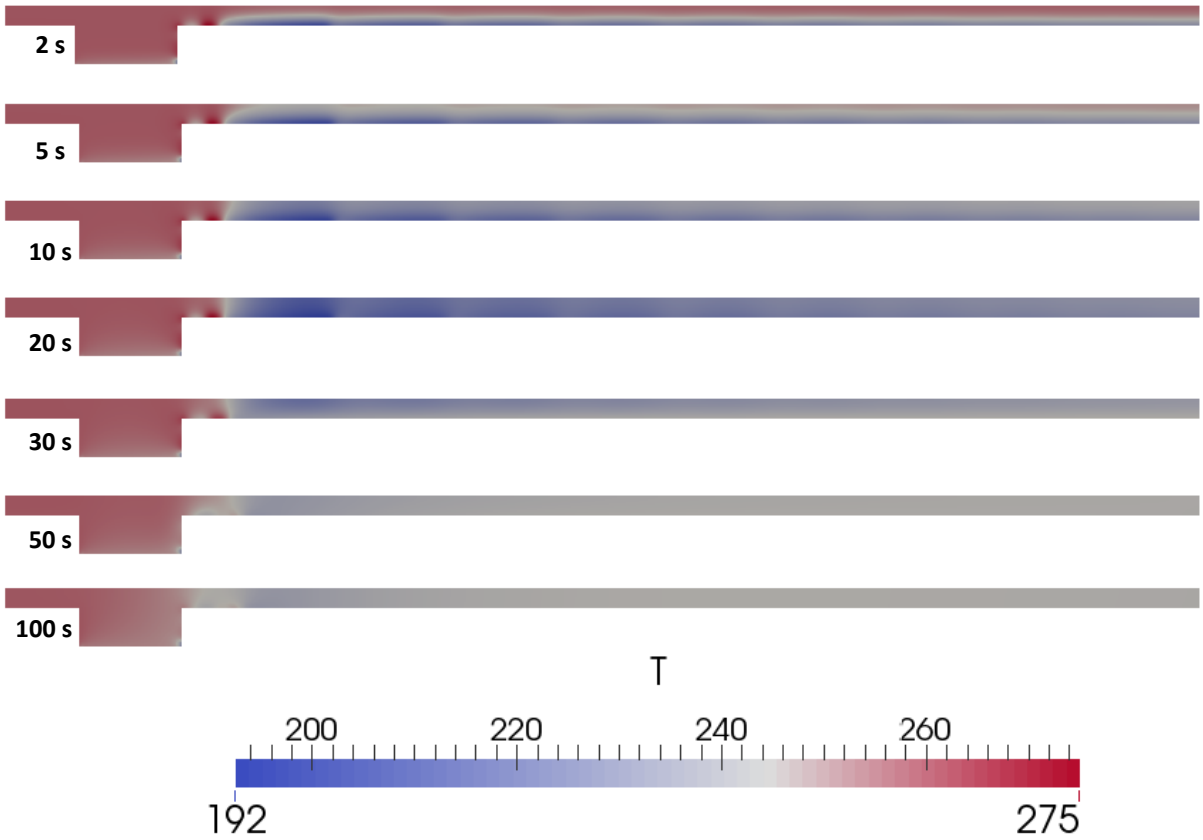

Figure 5-12 Temperature development in pipe, case B

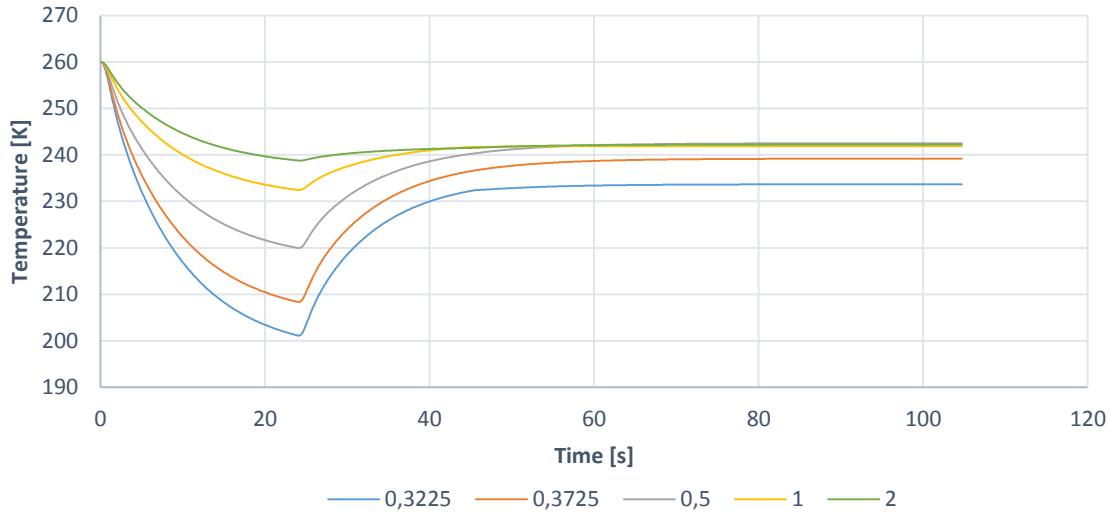

Figure 5-13 Probed temperature in the middle of pipe wall, case B

The lowest temperature that occurs in the pipe is 191,8 K (-81,2°C), see Figure 5-9. At the fluid surface, before time averaging the temperature profiles, the lowest temperature is 187,9 K (-85,1°C), see Figure 5-5. Plots equivalent to Figure 5-11 and Figure 5-13, for temperatures measured at the outer wall of the pipe, can be found in Appendix B.2.2.

5.2.3. Minimum design temperature

The minimum design temperature for the pipe is -46°C, as described in section 1.2. The temperature at the fluid surface is above this limit for all cases, except case 1 and 1P, see Figure 5-4 and Figure 5-9.

In our simulations, the lowest temperatures in the pipe occur at approximately 23 seconds, before the inner wall temperature is updated. Figure 5-14 shows the pipe temperature, measured in the middle of the pipe wall, for case A and B, compared to the minimum design temperature.

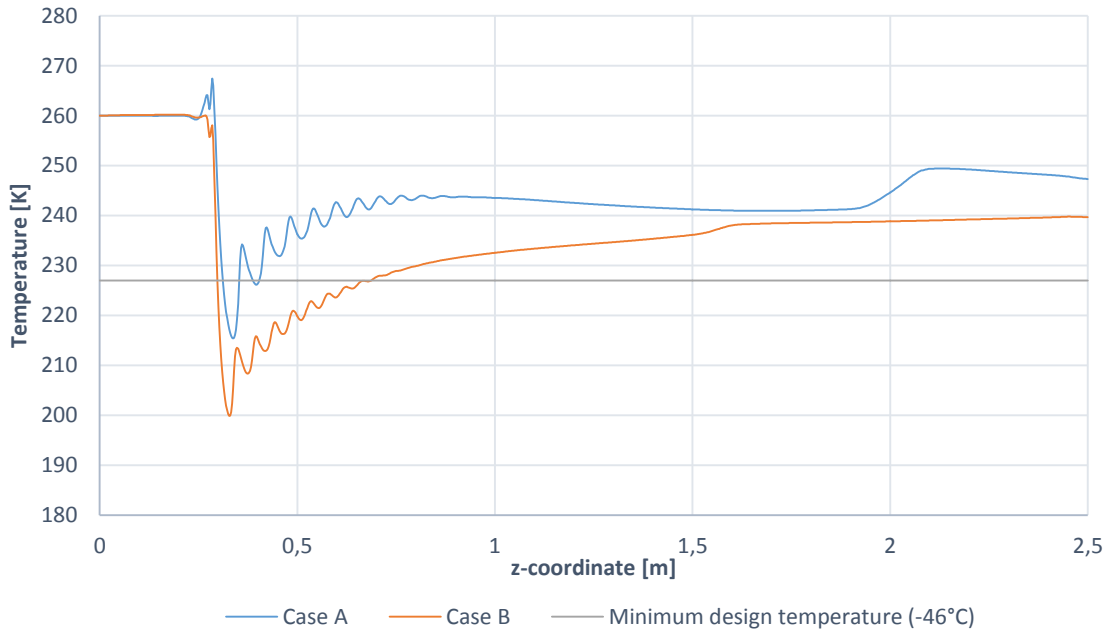

Figure 5-14 Temperature in the middle of pipe wall after 23 seconds

As expected, case B predicts the lowest temperature. Approximately 0,5 m of the pipe downstream of the orifice is cooled below the minimum design temperature. The regions with temperatures below the limit are shown in blue in Figure 5-15 and Figure 5-16, for case A and B respectively.

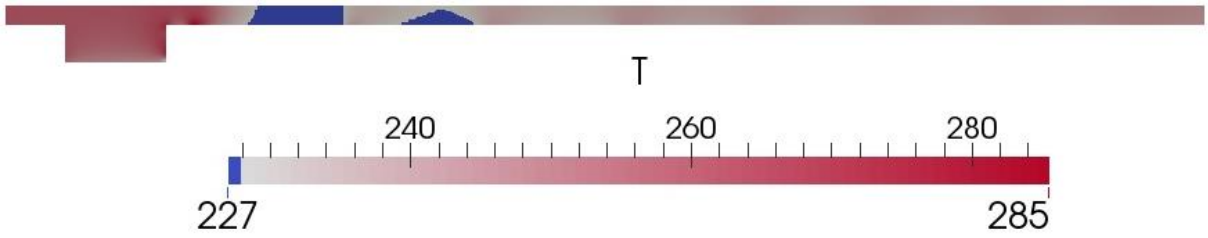

Figure 5-15 Regions with temperatures below minimum design temperature, case A

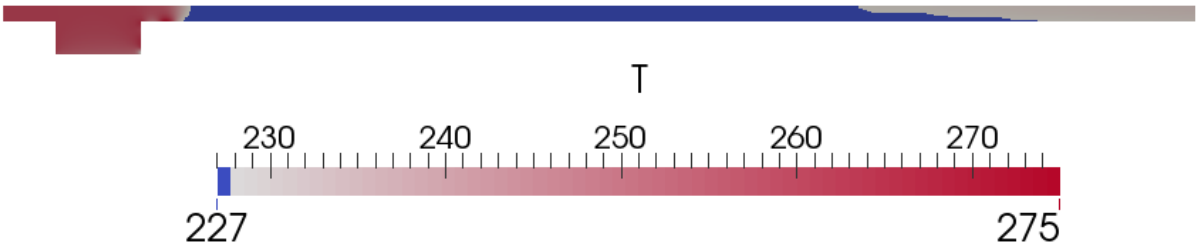

Figure 5-16 Temperature below minimum design temperature, case B

When the temperature boundary condition is changed to case 5 and 5P, after 24 seconds, the temperature in the pipe rises. Figure 5-17 shows the minimum temperature in the middle of the pipe wall, as a function of time. An equivalent plot for the lowest temperature, measured at the outer wall, is shown in Figure 5-18.

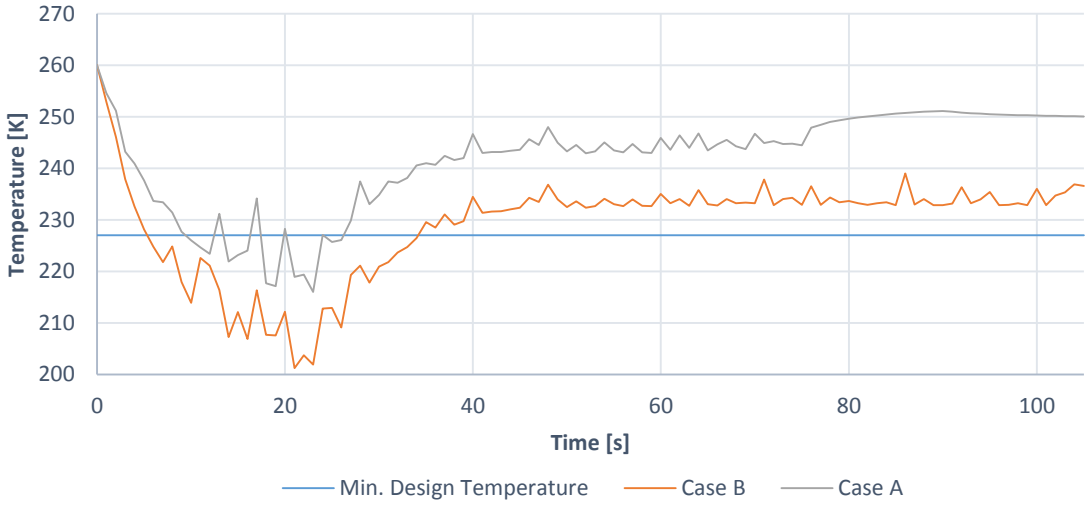

Figure 5-17 Minimum temperature in the middle of pipe wall

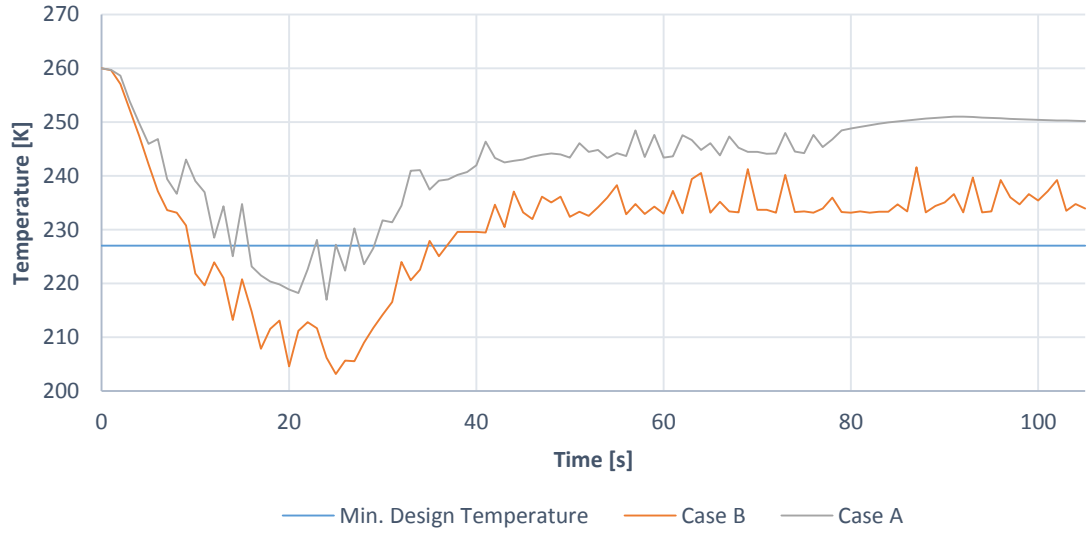

Figure 5-18 Minimum temperature at outer pipe wall

The temperature in the entire pipe is above the minimum design temperature after approximately 30 and 40 seconds, for case A and B respectively.

In Aker's report, this time is estimated to 228 seconds; refer to Appendix B.2.3 for details.

5.3. Case 11

Figure 5-19 shows the velocity in z-direction for case 11.

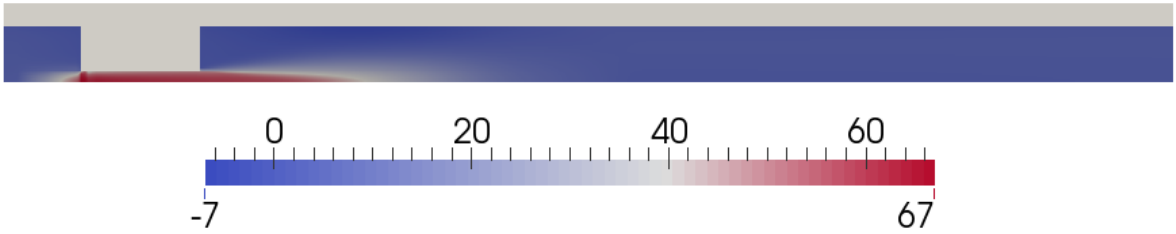

Figure 5-19 Velocity in z-direction case 11, 15 ms

The temperature distribution in both pipe and fluid is shown in Figure 5-20.

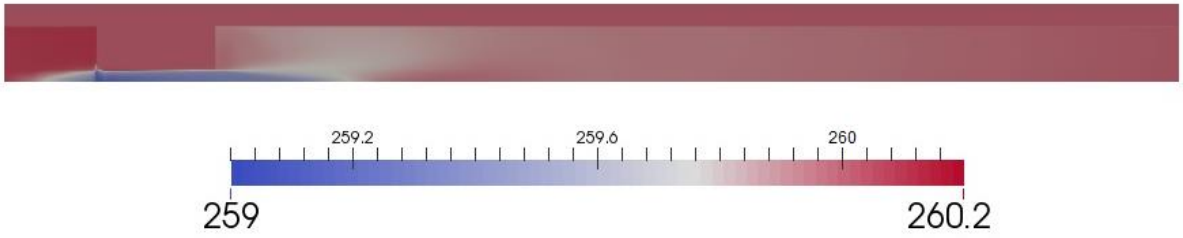

Figure 5-20 Temperature case 11, 15 ms

Due to the low NPR of this case, the temperature drop in the fluid is negligible and the velocity is clearly subsonic. The pipe temperature is almost unaffected by the flow. See case files in Appendix H for more detailed results.

5.4. Validation and discussion

5.4.1. Rapid increase in fluid surface temperature

As shown in section 5.1, the fluid surface temperature increases significantly, when the backpressure is increased to 5 bar or more. If we look at the minimum temperature in the core of the jet, shown in Appendix B.1, we see that it increases from approximately 80 K for case 1P to 104 K for case 9P. One could therefore expect that the fluid surface temperature also should stay low for backpressures larger than 5 bar.

According to equation (1-1) and (1-2), the size of the under-expanded jet is a function of the NPR. The jet's size decreases as the backpressure increases, due to the decreasing NPR. As the jets become smaller, they interact less with the pipe wall, and therefore the temperature at the fluid surface is less influenced by the cold jet.

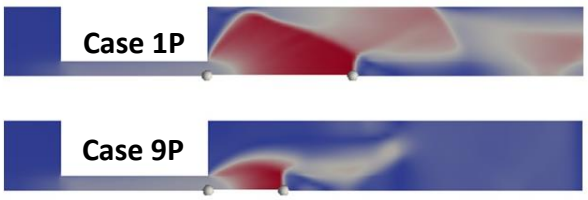

Figure 5-21 Smaller jet and less wall interaction for case 9P compared to 1P

This is clearly seen in Figure 5-21, which shows a comparison of the jet size and wall interaction in case 1P and 9P. Flow visualizations shown in Appendix B.1 show the same trend. Based on these observations it seems reasonable that the temperature at the fluid surface increases more rapidly than the temperature in the core of the jet.

5.4.2. Condensation in orifice

The temperature in the exit region of the orifice becomes very low, especially in case 1 and 1P, as shown in Figure 5-22. This could result in condensation of the gas.

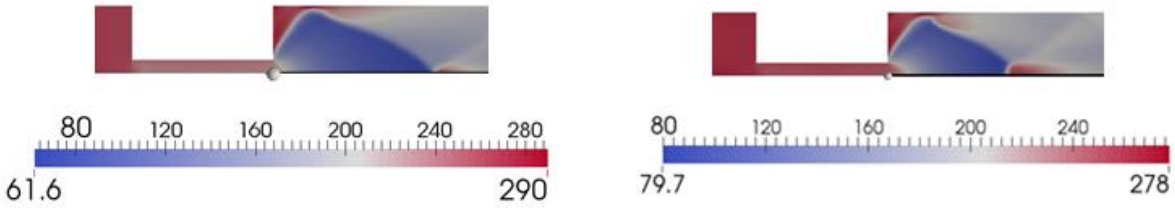

Figure 5-22 Temperature after orifice, case 1 (left) and 1P (right)

Figure 5-23 shows the pressure and temperature in the jet, measured along the centerline, for case 1. The pressure drops to almost 0,1 bar (10^4 Pa), at the point where the lowest temperature occurs. According to the phase diagram for methane, shown in Figure 5-25, the gas will start to undergo a phase change at approximately 85 K at this pressure.

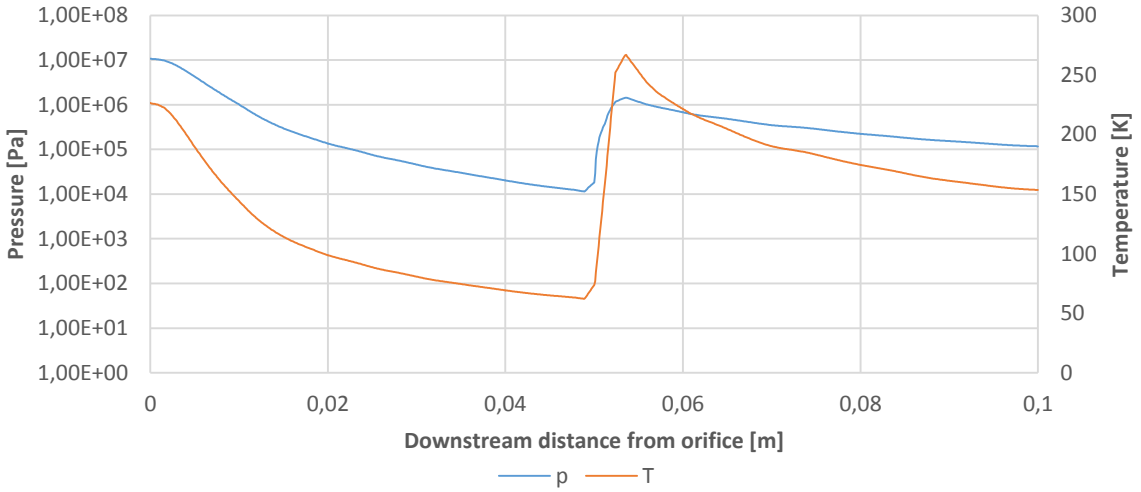

Figure 5-23 Pressure and temperature along centerline of jet, case 1

The pressure and temperature, measured along the centerline, for case 1P is shown in Figure 5-24. The pressure drops to approximately 0,3 bar ($3 \cdot 10^4$ Pa). At this pressure, the phase change is expected to occur at approximately 96 K, see Figure 5-25.

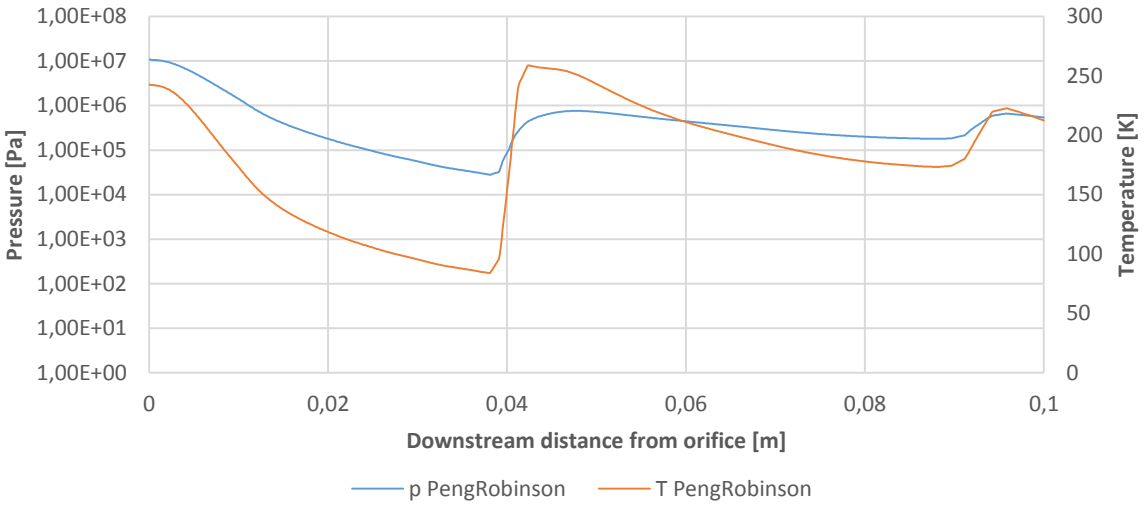

Figure 5-24 Pressure and temperature along centerline of jet, case 1P

Since the temperature in the exit zone drops to 61,6 K in the ideal gas simulations and 79,7 K in the Peng-Robinson simulations, this could result in a phase change. However, our solver does not support multiple phases. It assumes that all energy is used to change the temperature of the gas, see Table 4-8.

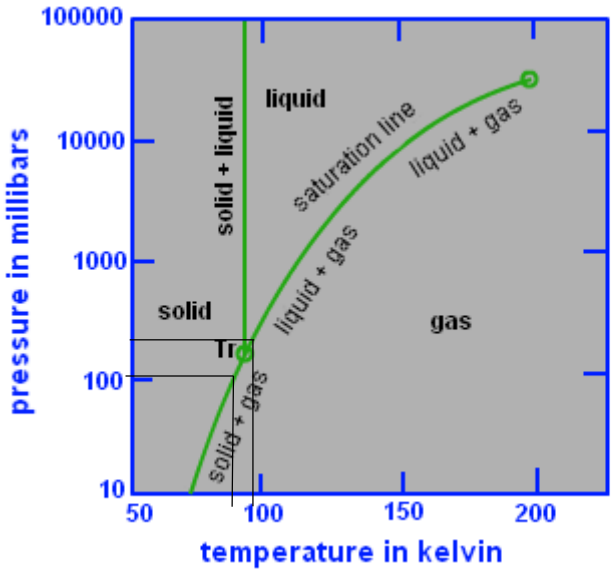

Figure 5-25 Phase diagram methane [63]

Because of this, too low simulation temperatures could be predicted in the exit zone. However, since the phase change problem only applies to a small region at the exit of the orifice, it is assumed to have little influence on the temperature at the fluid surface. Considering phase change is also beyond the scope of this thesis.

5.4.3. Comparison with results from Aker's report

Table 5-3 contains a comparison between results from Aker's report, see Table 1-2, and our results, see Appendix B.

Table 5-3 Comparison with Aker's results

	Aker's report (Case 1a)	**Case A (IdealGas)**	**Case B (Peng-Robinson)**
Minimum gas temperature	145,3 K (-127,7°C)	62 K (-211°C)	79,7 K (-193,3°C)
Minimum inner wall surface temperature	177,1 K (-95,9°C)	210,1 K (-62,9°C)	191,8 K (-81,2°C)
Time of lowest temperature	29,4 sec	23 sec	23 sec
Pressure at minimum temperature	5,9 bar	5 bar	5 bar
Time until minimum wall temperature above minimum design temperature (approximately)	228 sec (see Appendix B.2.3)	30 sec	40 sec

Our results predict a much lower minimum gas temperature than Aker. The CFD simulations are able to capture the extreme gas expansion and cooling, which occurs in the under-expanded jets, in a more detailed manner than Olga does. It should be noted that there is some uncertainty linked to these values and that they might be too low, as explained in section 5.4.2.

The minimum temperature at the inner wall of the pipe is a bit higher in our results, -81,2 °C, compared to Aker's -95,9°C (Case 1a). Similar to what Aker found, the pressure in the fluid is around 5 bar when the lowest pipe temperature occurs. Finally, our results predict that the temperature in the pipe will rise above the minimum design temperature after just 40 seconds, compared to Aker's 228 seconds. The maximum pressure that occurs while the pipe temperature is below the design temperature, is approximately 7,7 bar in our simulations. The equivalent pressure in Aker's simulations is estimated to 39 bar; see Appendix C.3 for details.

5.4.4. Mach disc location and diameter

The distance from the exit of the orifice to the first Mach disk for a free jet, can be estimated using equation (1-1). A comparison between this expression and our results is shown in Figure 5-26. The results were recorded after 0,3 ms. For the cases with NPR below 80 we find a good agreement between the equation and our results. The jet structure in case 1 and 2 is greatly affected by the confining pipe wall, and the results therefore deviate from equation (1-1). The influence of the pipe wall, for case 1 and 2, can also be seen in Figure 5-28, which shows the location of the Mach disk at 0,3 ms for case 1-10. A similar visualization for the Peng-Robinson cases is found in Appendix B.1.15.

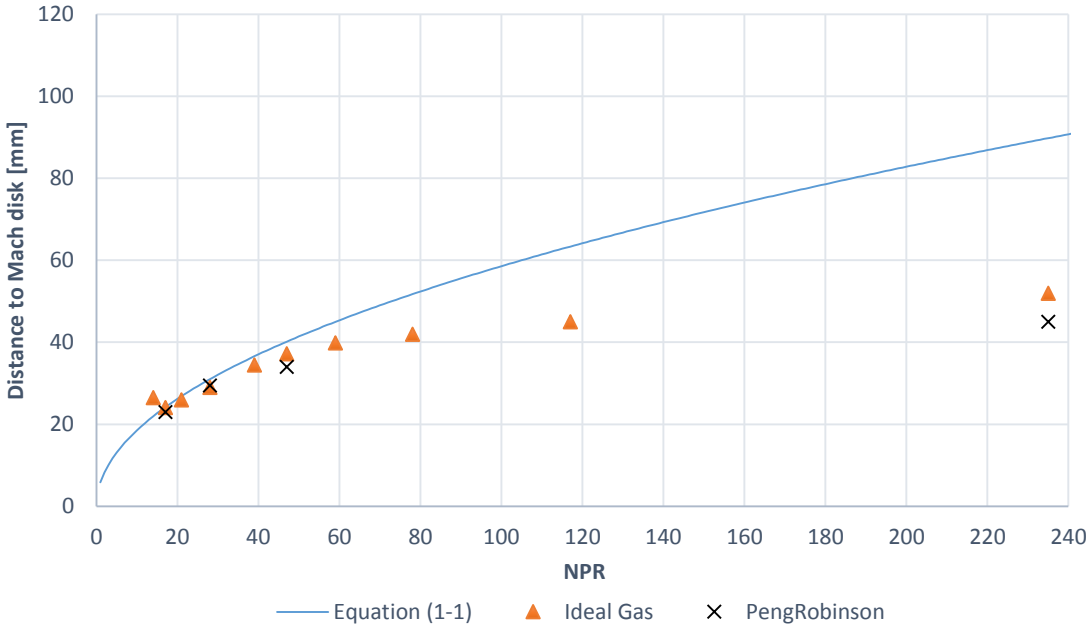

Figure 5-26 Distance from orifice exit to first Mach disk

Figure 5-27 shows the diameter of the Mach disks from our simulations, compared to the expression given in equation (1-2).

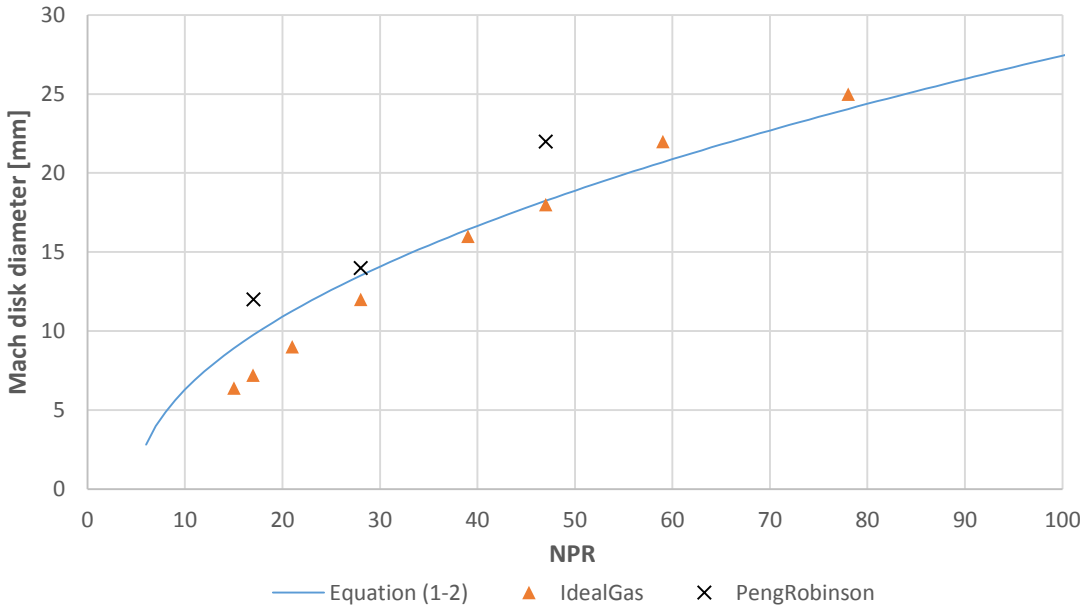

Figure 5-27 Mach disc diameter

We see that our results follow the same trend as the equation predicts, with some deviation. Case 1 and 2 has been omitted from this comparison, since the Mach disk almost disappears due to the effect of the wall, see Figure 5-28. It should be noted that equation (1-2) was derived based on experiments with NPR up to 10 only. Its validity above this NPR is therefore uncertain.

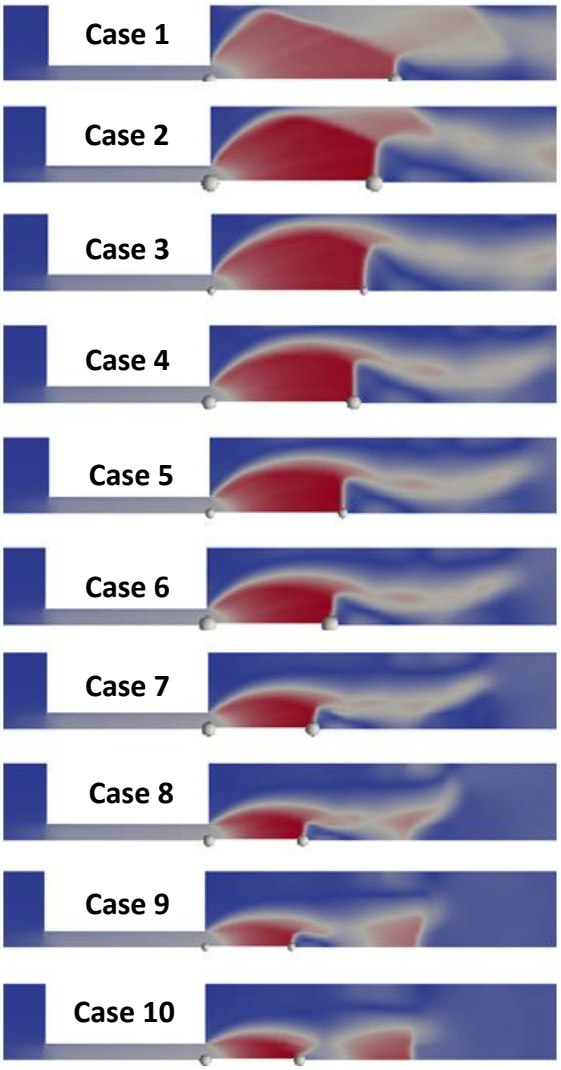

Figure 5-28 Mach disk location plotted for velocity magnitude after 0,3 ms, case 1-10

5.4.5. Comparison with papers

Experimental data for jets with NPR above 30-35 has proved to be difficult to find. In addition, no studies of supersonic confined jets have been found, see section 1.4. The validation of our results, beyond Mach disk location and diameter, therefore mainly rely on the solver's ability to predict the flow.

As described in section 3.3.2, the solver showed good agreement with experimental data for the Ladenburg validation case. Based on this we assume that the solver is capable of predicting jets with high NPR as well as how the jet flow structure is influenced by the wall of the pipe.

According to Table 4-1, case 7P and 9P, have an NPR of 27,6 and 17,4 respectively. The flow structure of these cases has been compared to similar free jets found in literature.

Wilkes et al. [13] studied free, axisymmetric nitrogen jets. Figure 5-29 shows the structure of their jets for NPRs of 18 and 31.

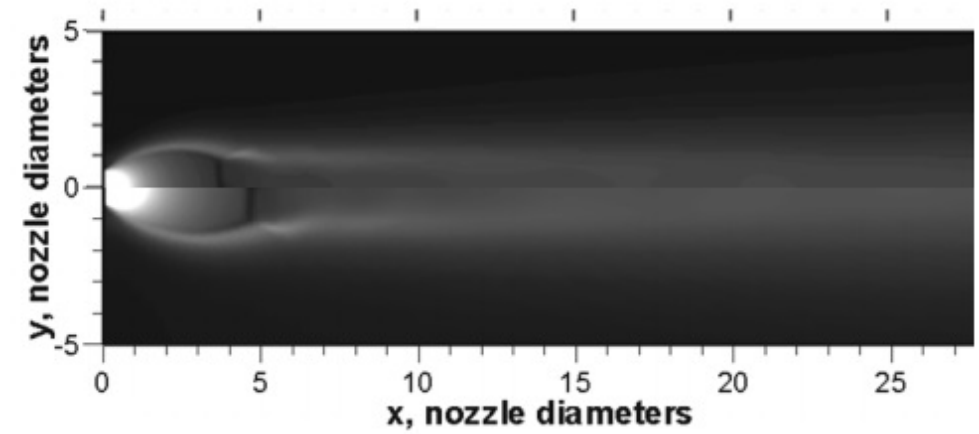

Figure 5-29 Nitrogen jet, NPR 18 (top) and 31 (bottom) [13]

Our results, for case 7P and 9P, are shown in Figure 5-30. The results are recorded after 0,5 ms.

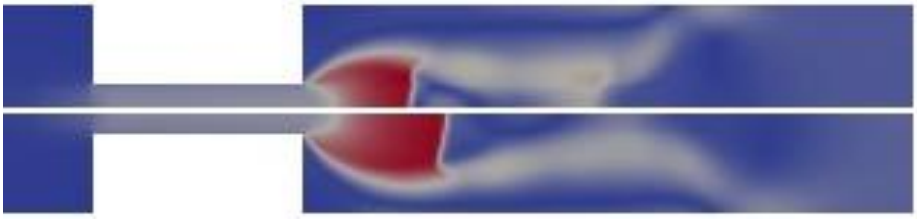

Figure 5-30 Case 9P (top) and 7P (bottom)

The flow structure is similar, and the jet with highest NPR extends furthest downstream. This is also expected according to equation (1-1). The Mach disks in our cases form at $2{,}5d_n$ and $3{,}25d_n$, which is close to the values predicted by equation (1-1), namely $2{,}7d_n$ and $3{,}4d_n$. d_n is the orifice diameter. In Figure 5-29, the Mach disks form at approximately $4d_n$ and $5d_n$.

Fu et al. [14] studied a high temperature jet (1415,1 K) jet with an NPR of 17,2, shown in Figure 5-31.

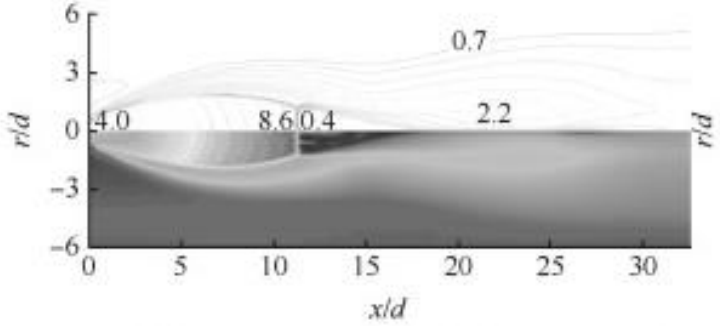

Figure 5-31 Mach number distribution NPR=17,2 [14]

It can be seen that the structure of the jet is quite similar to our case 9P, with one clearly defined Mach disk. However, the distance to the Mach disc is much smaller in our case, approximately $2{,}8d_n$, compared to approximately 11-12d_n in Figure 5-31.

From these comparisons, it is clear that the overall structure of the jets, predicted by rhoCentralFoam, seems to be reasonable.

5.4.6. Turbulence model comparison

The choice of turbulence model might influence the fluid surface temperature. To investigate this, case 1P was run using the RNG k-ε, the realizable k-ε and the standard k-ε, in addition to the k-ω SST turbulence model. The k-ω SST model was chosen due to its reported performance for both heat transfer and under-expanded jets, see section 3.6.1.

The medium mesh with 10880 cells was used, and the boundary conditions were equal to the ones listed in Table 4-4. Results were recorded after 6 ms of flow time. The initial value for ε was calculated based on k and l_{turb} from equation (4-1) and (4-2) [56].

$$\varepsilon = C_\mu \frac{k^{\frac{3}{2}}}{l_{turb}} = 0{,}09 \cdot \frac{0{,}177^{\frac{3}{2}}}{1{,}627 \cdot 10^{-3}} = 4{,}12 \tag{5-1}$$

The results are shown in Figure 5-32. We see that the results from RNG k-ε, standard k-ε and k-ω SST are almost equal, while the realizable k-ε model predicts much lower temperatures.

The RNG k-ε model predicts up to 3% lower temperatures than the k-ω SST in the jet region, downstream of the orifice. Refer to Appendix B.2.4 for details. The standard k-ε model predicts a bit higher temperatures in this region, while the k-ω SST model predicts slightly higher temperatures than the two k-ε models further downstream. Eventually, all three models give almost equal temperatures.

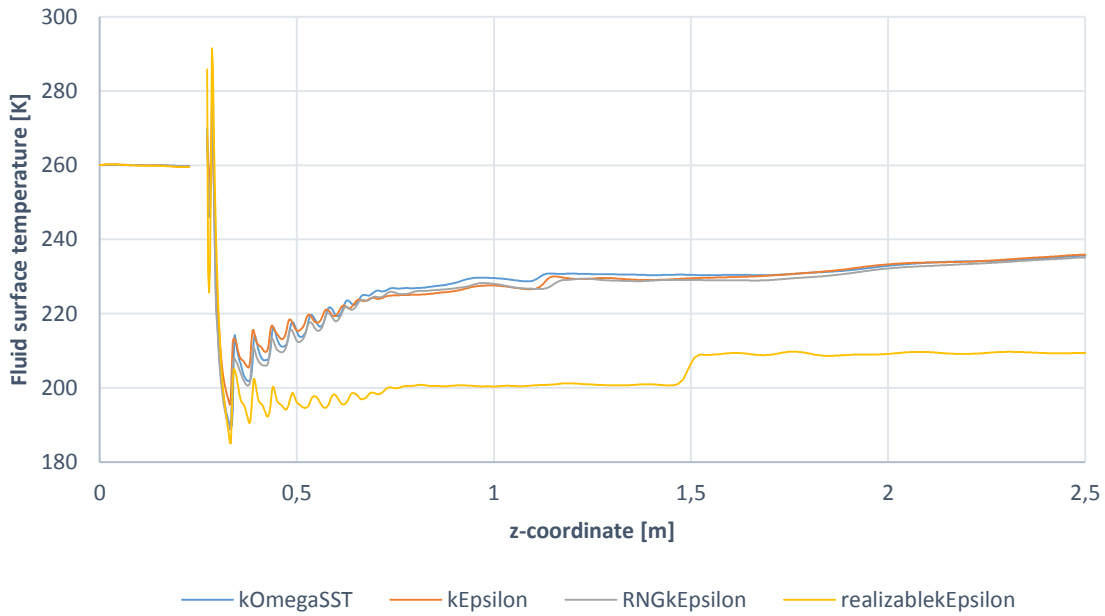

Figure 5-32 Turbulence model comparison, 6ms flow time

To investigate the difference between the realizable k-ε and the other turbulence models, the case was also run without turbulence modeling. A comparison between the laminar solution and the realizable k-ε is shown in Figure 5-33.

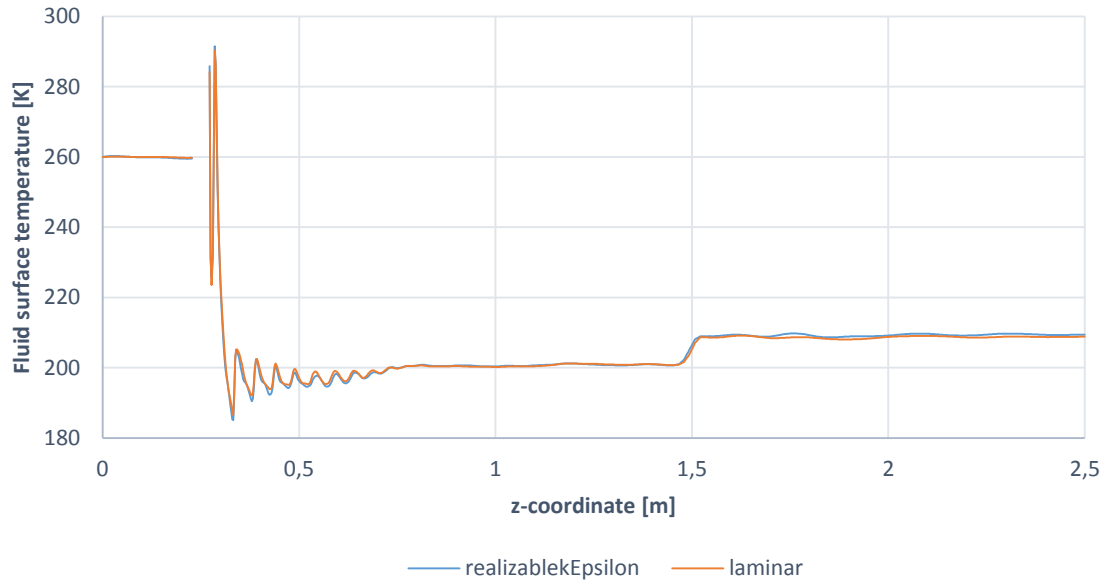

Figure 5-33 realizablekEpsilon compared to laminar solution

We see that the temperature profiles are almost identical. Since all the other turbulence models we have tested show that the turbulence has an effect on the fluid surface temperature, this result indicates that there might be something wrong with the implementation of the realizable k-ε turbulence model in OpenFOAM.

The turbulent kinetic energy, measured along the centerline, was compared for all the k-ε models, see Figure 5-34. We see that the turbulent kinetic energy predicted by the realizable k-ε model is very low, compared to the two other models. The flow velocity in the jet is around 1000 m/s, see Appendix B.1.1. If a turbulent intensity of only 1% is assumed, the turbulent kinetic energy should be at least 150 in this region, according to equation (4-2).

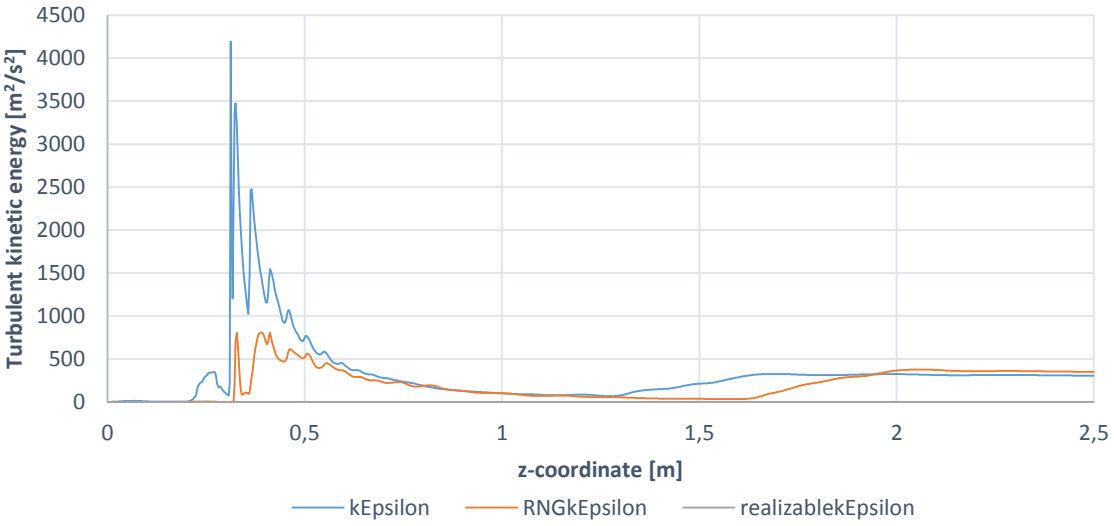

Figure 5-34 Turbulent kinetic energy at centerline

It is evident that the realizable k-ε predicts too low turbulent kinetic energy. Based on these observations, we assume that the temperature predicted by the realizable k-ε model is erroneous and can be neglected.

5.4.7. y^+

The y^+ values should lie between 30 and 300 to ensure a correct solution when using wall function, see section 3.6.2.

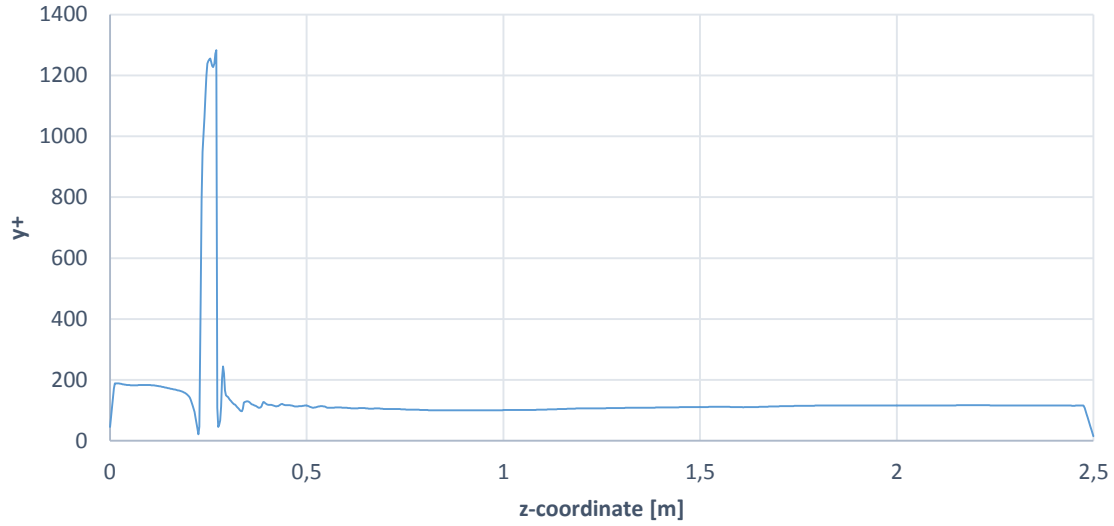

Figure 5-35 y^+ at wall for case 1P at 15 ms

Figure 5-35 shows the y^+ values for case 1P. We see that they are well within the limits, except inside the orifice, where the velocity is very high. It is assumed that this does not influence our solution and that these y^+ values are satisfactory.

5.4.8. Residuals

The residuals from all fluid cases are similar. Residuals from case 4 are plotted in Figure 5-36. It is obvious that the residuals are sufficiently small as the final residual for both enthalpy, h, and axial velocity are 10^{-8} or less. This also expected according to the specified limits, described in section 3.5.3.

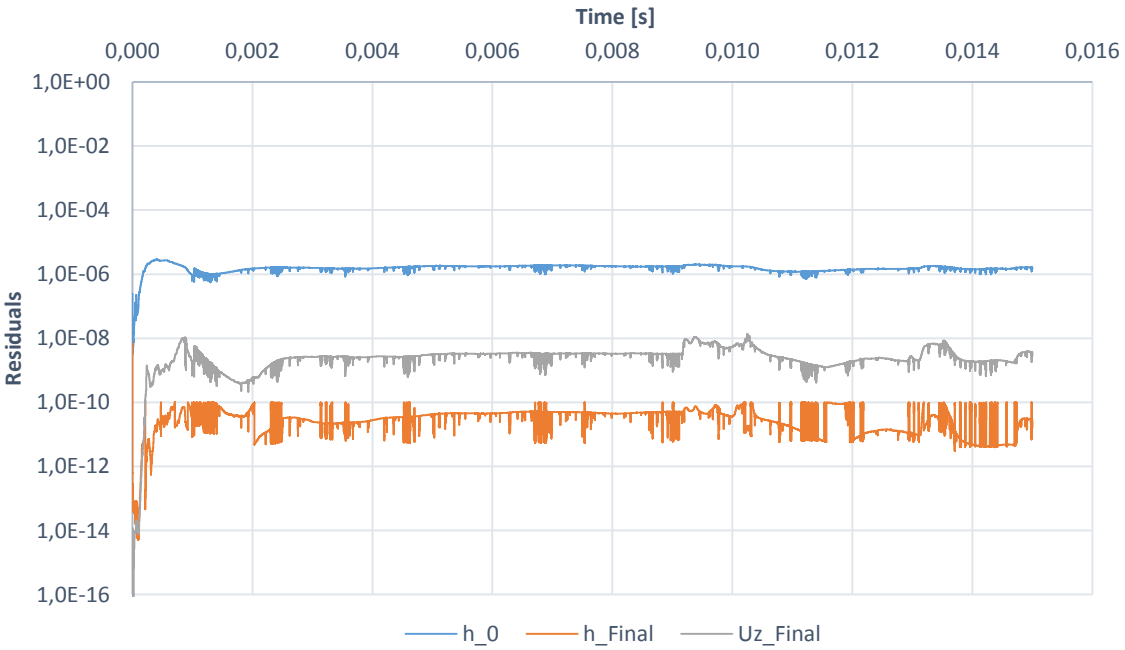

Figure 5-36 Residuals case 4

The residuals for pipe case A are shown in Figure 5-37.

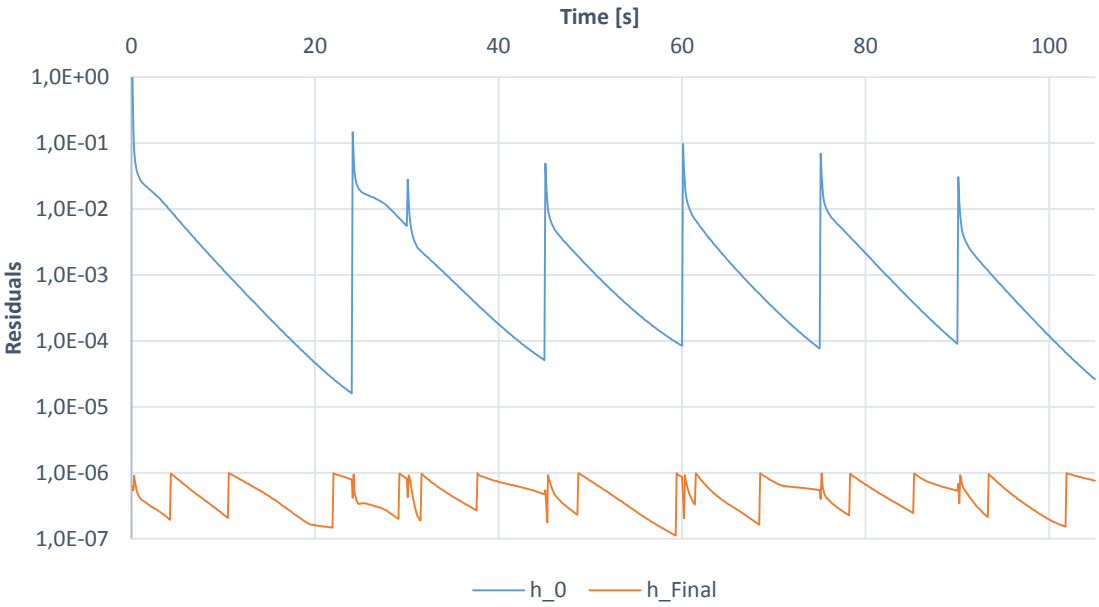

Figure 5-37 Residuals pipe case A

The initial h residual increases each time the temperature boundary condition is updated. However, the final h residual for each iteration is 10^{-6} or less, which is satisfactory.

5.4.9. General comments

Validation of our results against experimental data suggests that the predicted flow structure is physically correct. Based on Greenshields' et al. [30] validation, it is assumed that the solver is capable of predicting both the influence of the wall and the resulting fluid surface temperature.

However, the quasi-transient approach might influence our results. Since the jet is assumed to quickly reach steady state, and the resulting temperature field at the inner pipe wall is constant, possible transient effects in the jet during pressurization are not captured. If the jet structure is affected by the continuously rising backpressure, it is possible that the low temperatures could extend further upstream than indicated in our results.

6. Conclusion

The temperature distribution in a 2” pipeline, due to a flow restriction, has been investigated, using the open-source CFD code OpenFOAM. The fluid and the pipe were simulated separately, using the solvers rhoCentralFoam and chtMultiRegionFoam respectively. The natural gas was modelled as pure methane.

First, the fluid region was simulated until a reasonably steady state surface temperature was reached. The output pressure was increased stepwise to approximate the effect of the rising backpressure in the system. For each fluid case, the surface temperature was extracted and used as boundary condition for the inner pipe wall in the heat transfer simulations.

The results show a minimum pipe temperature of 191,8 K (-81,2°C) compared to Aker’s 177,1 K (-95,9°C). The pipe, upstream of the orifice, is virtually unaffected by the low temperatures, while approximately 0,5 m of the downstream pipe experiences temperatures below the minimum design temperature (-46°C). However, the low temperatures occur only for a very limited time. After approximately 40 seconds, the temperature in the entire pipe is above the limit. In Aker’s report, this time was estimated to 228 seconds.

According to Aker’s report, the design pressure at -46°C is 425,5 bar. The lowest temperature in our simulations occurs when the pressure is around 5 bar. The maximum pressure that occurs, while the temperature is below the limit, is 7,7 bar.

To validate the results, a grid independence test was performed. The number of cells was increased by 40% compared to the mesh used in the simulations. The resulting change in fluid surface temperature was below $\pm 1\%$, and thus the solution in assumed to be independent of the grid.

The choice of turbulence model might also influence the accuracy of the results. The k-ω SST model, which has been applied in this thesis, was compared to the standard k-ε and the RNG k-ε model. The RNG k-ε model predicted the lowest fluid surface temperature, up to 3% lower than the k-ω SST.

Due to the large pressure difference up- and downstream of the orifice a highly under-expanded jet occurs. Such jets are characterized by the presence of normal shocks, called Mach disks. The Mach disk location, for each fluid case, was compared to an experimental expression, valid for free jets. A similar comparison was made for the Mach disk diameter. For NPRs up to 80, the location and diameter of the Mach disks were found to fit the experimental expressions reasonably well. However, above this NPR, the pipe wall influences the structure of the jets and the expressions become invalid, since the jets are no longer free. It should also be noted that the validity of the expression for Mach disk diameter is uncertain for NPRs above 10.

The structure of the jets were found to be similar to results reported in literature for NPRs around 17 and 30.

Based on the validation against experimental data, it is assumed that the obtained results are physically correct and that the predicted temperature distribution in the pipe is a satisfactory estimate of the real temperature. Since no experimental data was found to validate the cases with NPR between 80 and 235, the validity of these results rely on the ability of the rhoCentralFoam solver to correctly predict the flow.

For the case with 5 bar pressure difference, the temperature drop in the pipe was found to be negligible.

6.1. Further work

Suggestions for further work are listed below.

- Re-run all cases using Peng-Robinson, including case 2-4
- Run simulations with correct gas composition
- Extend simulations to a full 3D mesh
- Verify the influence of the wall on the jet structure by experiments (NPR 80-235)

As mentioned in the introduction, there seems to be little work done on supersonic confined jets. It would therefore be interesting to further study how a confining wall, such as a pipe, influences the flow structure of the under-expanded jet.

Bibliography

[1] B. Holmes, 'CRITICAL FLOW', A-to-Z Guide to Thermodynamics, Heat and Mass Transfer, and Fluids Engineering, 2006.

[2] Wikipedia, 'De Laval nozzle', 2015. [Online]. Available: http://en.wikipedia.org/wiki/De_Laval_nozzle [Accessed: 09- Apr- 2015].

[3] Aker Solutions, 'Dynamic simulation report. Skarv gas injection compressor pressurization', Aker Solutions.

[4] D. PACK, 'ON THE FORMATION OF SHOCK-WAVES IN SUPERSONIC GAS JETS: TWO-DIMENSIONAL FLOW', The Quarterly Journal of Mechanics and Applied Mathematics, vol. 1, no. 1, pp. 1-17, 1948.

[5] C. Donaldson and R. Snedeker, 'A study of free jet impingement. Part 1. Mean properties of free and impinging jets', Journal of Fluid Mechanics, vol. 45, no. 02, p. 281, 1971.

[6] S. CRIST, D. GLASS and P. SHERMAN, 'Study of the highly underexpanded sonic jet.', AIAA Journal, vol. 4, no. 1, pp. 68-71, 1966.

[7] A. ADDY, 'Effects of axisymmetric sonic nozzle geometry on Mach disk characteristics', AIAA Journal, vol. 19, no. 1, pp. 121-122, 1981.

[8] P. Birkby and G. Page, 'Numerical predictions of turbulent underexpanded sonic jets using a pressure-based methodology', Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, vol. 215, no. 3, pp. 165-173, 2001.

[9] E. S. Love, C. E. Grigsby, L. P. Lee and J. M. Woodling, 'Experimental and theoretical studies of axisymmetric free jets'. *NASA Technical Report R-6,* 1959.

[10] J. Jaramillo, C. Pérez-Segarra, A. Oliva and K. Claramunt, 'Analysis of different RANS models applied to turbulent forced convection', International Journal of Heat and Mass Transfer, vol. 50, no. 19-20, pp. 3749-3766, 2007.

[11] X. Liu, A. Godbole, C. Lu, G. Michal and P. Venton, 'Source strength and dispersion of CO2 releases from high-pressure pipelines: CFD model using real gas equation of state', Applied Energy, vol. 126, pp. 56-68, 2014.

[12] B. Xu, J. Zhang, J. Wen, S. Dembele and J. Karwatzki, Numerical Study of a Highly Under-expanded Hydrogen Jet, 1st ed. London: School of Engineering, Kingston University, 2005. [Online]. Available: http://conference.ing.unipi.it/ichs2005/Papers/110122.pdf [Accessed: 14- Apr- 2015].

[13] J. Wilkes, C. Glass, P. Danehy and R. Nowak, 'Fluorescence Imaging of Underexpanded Jets and Comparison with CFD', 44th AIAA Aerospace Sciences Meeting and Exhibit, pp. 1-14, 2006. [Online]. Available: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20060004779.pdf [Accessed: 22- Apr- 2015].

[14] D. Fu, Y. Yu and Q. Niu, 'Simulation of underexpanded supersonic jet flows with chemical reactions', Chinese Journal of Aeronautics, vol. 27, no. 3, pp. 505-513, 2014.

[15] T. Rogers, P. Petersen, L. Koopmans, P. Lappas and A. Boretti, 'Structural characteristics of hydrogen and compressed natural gas fuel jets', International Journal of Hydrogen Energy, vol. 40, no. 3, pp. 1584-1597, 2015.

[16] M. Kandakure, V. Patkar and A. Patwardhan, 'Characteristics of turbulent confined jets', Chemical Engineering and Processing: Process Intensification, vol. 47, no. 8, pp. 1234-1245, 2008.

[17] H. Versteeg and W. Malalasekera, An introduction to computational fluid dynamics. Harlow, England: Pearson Education Ltd., 2007.

[18] D. Peng and D. Robinson, 'A New Two-Constant Equation of State', Ind. Eng. Chem. Fund., vol. 15, no. 1, pp. 59-64, 1976.

[19] Wikipedia, 'Equation of state', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Equation_of_state#Peng.E2.80.93Robinson_equation_of_state [Accessed: 30- Apr- 2015].

[20] Wikipedia, 'Choked flow', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Choked_flow [Accessed: 08- Apr- 2015].

[21] Thermalfluidscentral.org, 'Thermal-FluidsPedia | Heat and mass transfer | Thermal-Fluids Central', 2015. [Online]. Available: https://www.thermalfluidscentral.org/encyclopedia/index.php/Heat_and_Mass_Transfer [Accessed: 16- Jan- 2015].

[22] D. Hahn and M. Özişik, Heat conduction. Hoboken, N.J.: John Wiley & Sons, 2012.

[23] Thermalfluidscentral.org, 'Thermal-FluidsPedia | Basics of heat conduction | Thermal-Fluids Central', 2015. [Online]. Available: https://www.thermalfluidscentral.org/encyclopedia/index.php/Basics_of_heat_conduction [Accessed: 09- Apr- 2015].

[24] Wikipedia, 'Thermal conduction', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Thermal_conduction [Accessed: 09- Apr- 2015].

[25] Wikipedia, 'Thermal diffusivity', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Thermal_diffusivity [Accessed: 09- Apr- 2015].

[26] Thermalfluidscentral.org, 'Thermal-FluidsPedia | One-dimensional steady-state heat conduction | Thermal-Fluids Central', 2015. [Online]. Available: https://www.thermalfluidscentral.org/encyclopedia/index.php/One-dimensional_steady-state_heat_conduction [Accessed: 22- Apr- 2015].

[27] Y. Çengel and A. Ghajar, Heat and mass transfer. New York: McGraw-Hill, 2011.

[28] O. Foundation, 'User Guide', OpenFOAM.org, 2015. [Online]. Available: http://www.OpenFOAM.org/docs/user/ [Accessed: 22- Jan- 2015].

[29] CFD Direct, 'OpenFOAM User Guide: 4.1 OpenFOAM case directory', 2015. [Online]. Available: http://cfd.direct/OpenFOAM/user-guide/case-file-structure/#x17-930004.1 [Accessed: 22- Apr- 2015].

[30] C. Greenshields, H. Weller, L. Gasparini and J. Reese, 'Implementation of semi-discrete, non-staggered central schemes in a colocated, polyhedral, finite volume framework, for high-speed viscous flows', Int. J. Numer. Meth. Fluids, pp. 1-21, 2009.

[31] J. Winckler, 'The Mach Interferometer Applied to Studying an Axially Symmetric Supersonic Air Jet', Rev. Sci. Instrum., vol. 19, no. 5, p. 307, 1948.

[32] R. Ladenburg, C. Van Voorhis and J. Winckler, 'Interferometric Studies of Faster than Sound Phenomena. Part II. Analysis of Supersonic Air Jets', Physical Review, vol. 76, no. 5, pp. 662-677, 1949.

[33] CFD Direct, 'OpenFOAM User Guide: 4.4 Numerical schemes', 2015. [Online]. Available: http://www.OpenFOAM.org/docs/user/fvSchemes.php [Accessed: 15- Apr- 2015].

[34] A. Harten, 'High Resolution Schemes for Hyperbolic Conservation Laws', Journal of Computational Physics, vol. 135, no. 2, pp. 260-278, 1997.

[35] Wikipedia, 'Total variation diminishing', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Total_variation_diminishing [Accessed: 15- Apr- 2015].

[36] Introductory OpenFOAM Course, 1st ed. Genoa: University of Genoa, 2015. [Online]. Available: http://www.dicat.unige.it/guerrero/of2015a/9tipsandtricks.pdf [Accessed: 15- Apr- 2015].

[37] CFD Direct, 'OpenFOAM User Guide: 4.5 Solution and algorithm control', 2015. [Online]. Available: http://www.OpenFOAM.org/docs/user/fvSolution.php [Accessed: 15- Apr- 2015].

[38] B. Hjertager, Lecture notes on OpenFOAM, 1st ed. Stavanger: University of Stavanger, 2009, pp. 1-141.

[39] Wikipedia, 'Preconditioner', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Preconditioner [Accessed: 15- Apr- 2015].

[40] J. Perkins and V. Ranade, Computational flow for modeling for chemical reactor engineering. San Diego, Calif.: Academic, 2001.

[41] Cfd-online.com, 'Two equation turbulence models -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/Two_equation_turbulence_models [Accessed: 15- Apr- 2015].

[42] F. Menter, 'Two-equation eddy-viscosity turbulence models for engineering applications', AIAA Journal, vol. 32, no. 8, pp. 1598-1605, 1994.

[43] Cfd-online.com, 'SST k-omega model -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/SST_k-omega_model [Accessed: 15- Apr- 2015].

[44] J. Bredberg, 'On the Wall Boundary Condition for Turbulence Models', CHALMERS UNIVERSITY OF TECHNOLOGY, Gøteborg, 2000. [Online]. Available: http://www.tfd.chalmers.se/~lada/postscript_files/jonas_report_WF.pdf [Accessed: 13- Apr- 2015].

[45] COMSOL Blog, 'Which Turbulence Model Should I Choose for my CFD Application?', 2013. [Online]. Available: http://www.comsol.com/blogs/which-turbulence-model-should-choose-cfd-application/ [Accessed: 13- Apr- 2015].

[46] B. Hjertager, Computational Analysis of Fluid Flow Processes, 2nd ed. Stavanger, 2002, pp. 1-175.

[47] M. Dabic, 'Simulation of Fluid Suspended Particle Behaviour subject to Transverse Standing Acoustic Fields', University of Cape Town, Cape Town, 2012. [Online]. Available: http://www.cerecam.uct.ac.za/people/theses/Dabic[MSc][2012].pdf [Accessed: 15-Apr-2015].

[48] Cfd-online.com, 'Law of the wall -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/Law_of_the_wall [Accessed: 15- Apr- 2015].

[49] Wikipedia, Law of the wall. 2015. [Online] Available: http://en.wikipedia.org/wiki/Law_of_the_wall#/media/File:Law_of_the_wall_(English).svg [Accessed 15-Apr-2015].

[50] CFD Direct, 'OpenFOAM User Guide: 5.3 Mesh generation with blockMesh', 2015. [Online]. Available: http://www.OpenFOAM.org/docs/user/blockMesh.php [Accessed: 22- Apr- 2015].

[51] CFD Direct, 'OpenFOAM User Guide: 5.2 Boundaries', 2015. [Online]. Available: http://www.OpenFOAM.org/docs/user/boundaries.php [Accessed: 22- Apr- 2015].

[52] Cfd-online.com, 'Turbulence intensity -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/Turbulence_intensity [Accessed: 22- Apr- 2015].

[53] Cfd-online.com, 'Turbulence length scale -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/Turbulent_length_scale [Accessed: 22- Apr- 2015].

[54] Cfd-online.com, 'Hydraulic diameter -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/Hydraulic_diameter [Accessed: 22- Apr- 2015].

[55] Engineeringtoolbox.com, 'Overall Heat Transfer Coefficients for some common Fluids and Heat Exchanger Surfaces', 2015. [Online]. Available: http://www.engineeringtoolbox.com/overall-heat-transfer-coefficients-d_284.html [Accessed: 23- Apr- 2015].

[56] Cfd-online.com, 'Turbulence free-stream boundary conditions -- CFD-Wiki, the free CFD reference', 2015. [Online]. Available: http://www.cfd-online.com/Wiki/Turbulence_free-stream_boundary_conditions [Accessed: 22- Apr- 2015].

[57] Wikipedia, 'Viscosity', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Viscosity [Accessed: 17- Apr- 2015].

[58] Webbook.nist.gov, 'Thermophysical Properties of Fluid Systems', 2015. [Online]. Available: http://webbook.nist.gov/chemistry/fluid/ [Accessed: 22- Apr- 2015].

[59] Wikipedia, 'Prandtl number', 2015. [Online]. Available: http://en.wikipedia.org/wiki/Prandtl_number [Accessed: 17- Apr- 2015].

[60] Yutopian.com, 2015. [Online]. Available: http://www.yutopian.com/Yuan/prop/CH4.html [Accessed: 22- Apr- 2015].

[61] Wikipedia, 'Enthalpy of fusion', 2015. [Online] Available at: http://en.wikipedia.org/wiki/Enthalpy_of_fusion [Accessed 22 Apr. 2015].

[62] CFD Direct, 'OpenFOAM User Guide: 7.1 Thermophysical models', 2015. [Online]. Available: http://www.OpenFOAM.org/docs/user/thermophysical.php [Accessed: 22- Apr- 2015].

[63] TutorVista.com, CH4 Phase Diagram. 2015. [Online] Available: http://chemistry.tutorvista.com/physical-chemistry/phase-change-diagram.html [Accessed 27-Apr-2015].

[64] Pubs.usgs.gov, 'Methane Gas Volume Expansion Ratios and Ideal Gas Deviation Factors for the Deep-Water Bering Sea Basins: Peng-Robinson Equation of State', 2005. [Online]. Available: http://pubs.usgs.gov/of/2005/1451/equation.html [Accessed: 05- May- 2015].

[65] Engineeringtoolbox.com, 'Methane', 2015. [Online]. Available: http://www.engineeringtoolbox.com/methane-d_1420.html [Accessed: 05- May- 2015].

[66] J. Roberson and C. Crowe, Engineering fluid mechanics, 9th ed. New York: J. Wiley & Sons, 1997, p. Table A.2.

[67] A. Jackson, 'A comprehensive tour of snappyHexMesh', OpenFOAM Wiki, 2015. [Online]. Available: https://OpenFOAMwiki.net/images/f/f0/Final-AndrewJacksonSlidesOFW7.pdf [Accessed: 18- Feb- 2015].

[68] OpenFOAMwiki.net, 'Contrib/PyFoam - OpenFOAMWiki', 2015. [Online]. Available: https://OpenFOAMwiki.net/index.php/Contrib/PyFoam [Accessed: 24- Apr- 2015].

Appendix A The Assignment

A.1. The original assignment

AkerSolutions
Preferred partner

Title:
CFD analysis of temperature development due to flow restriction in pipeline

Background:
When gas is transferred from a high- to a low pressure system through a restriction orifice, a significant temperature reduction occurs. Typical scenarios could be pressurization of gas lift lines or depressurization of a segment during a platform blow down.

Objective:
The study shall model and simulate gas flowing from a high pressure system/pipe to a low pressure system/pipe via a restriction orifice. The objective is to investigate the temperature profile in the pipe steel/material and how far it propagates/creeps up- and downstream of the restriction orifice.

Normally the high and low pressure segment is separated with a valve upstream the orifice. At a given time the valve opens and gas flows through the orifice.

Option 1
If there is sufficient time, a simulation with a closed outlet boundary on the 10" pipe should be simulated.

Option 2
If there is sufficient time, a decaying pressure profile on the high pressure pipe boundary should be simulated to include the effect of depressurization. 10" outlet boundary shall be open.

Technical info:
Data source: Depressurisation of a gas export pipe over bridge landing. Layout as per sketch.

Orifice diameter = 26,87 mm

Pipeinfo upstream orifice (3")

- Pipe length : 10 m
- Outer diameter: 88,9 mm
- Wallthickness: 5,49 mm
- Inner diameter: -
- Pressure: 156 bara
- Initially gas filled

Pipeinfo downstream orifice (10")

- Length:10 m
- Outer diameter = 273,1 mm
- Inner diameter = 247,7 mm
- Pressure: Atmospheric
- Initial nitrogen filled

Gas

- Temperature: 60°C
- Pressure: 156 bara

Component	Mole Fractions
CO2	0,050
Methane	0,830
Ethane	0,090
Propane	0,030

Material info pipe

Cp [kJ/kg-C]	0,46
Density [kg/m3]	7800,00
Conductivity [W/m-K]	16,29

A.2. Updates to the assignment

In the early stages of this thesis, several changes were made to the original assignment, after discussions with Aker's representative, Jørgen Osenbroch. The objective of thesis has remained unchanged from the original assignment: to create a CFD simulation of the temperature development in a pipe, due to a flow restriction. However, the geometry, gas composition and temperature, orifice diameter etc. where changed, see section 1.2, 1.3 and 4.5.1 for details. The problem description in section 1.2 can be regarded as the final assignment.

Appendix B Results

This appendix contains the simulation results.

B.1. Flow visualization from ParaView

Visualizations of temperature, axial velocity and Mach number are taken at 15 ms for case 1-6, 17 ms for case 7, 20 ms for case 8-9 and 22ms for case 10.

B.1.1. Case 1

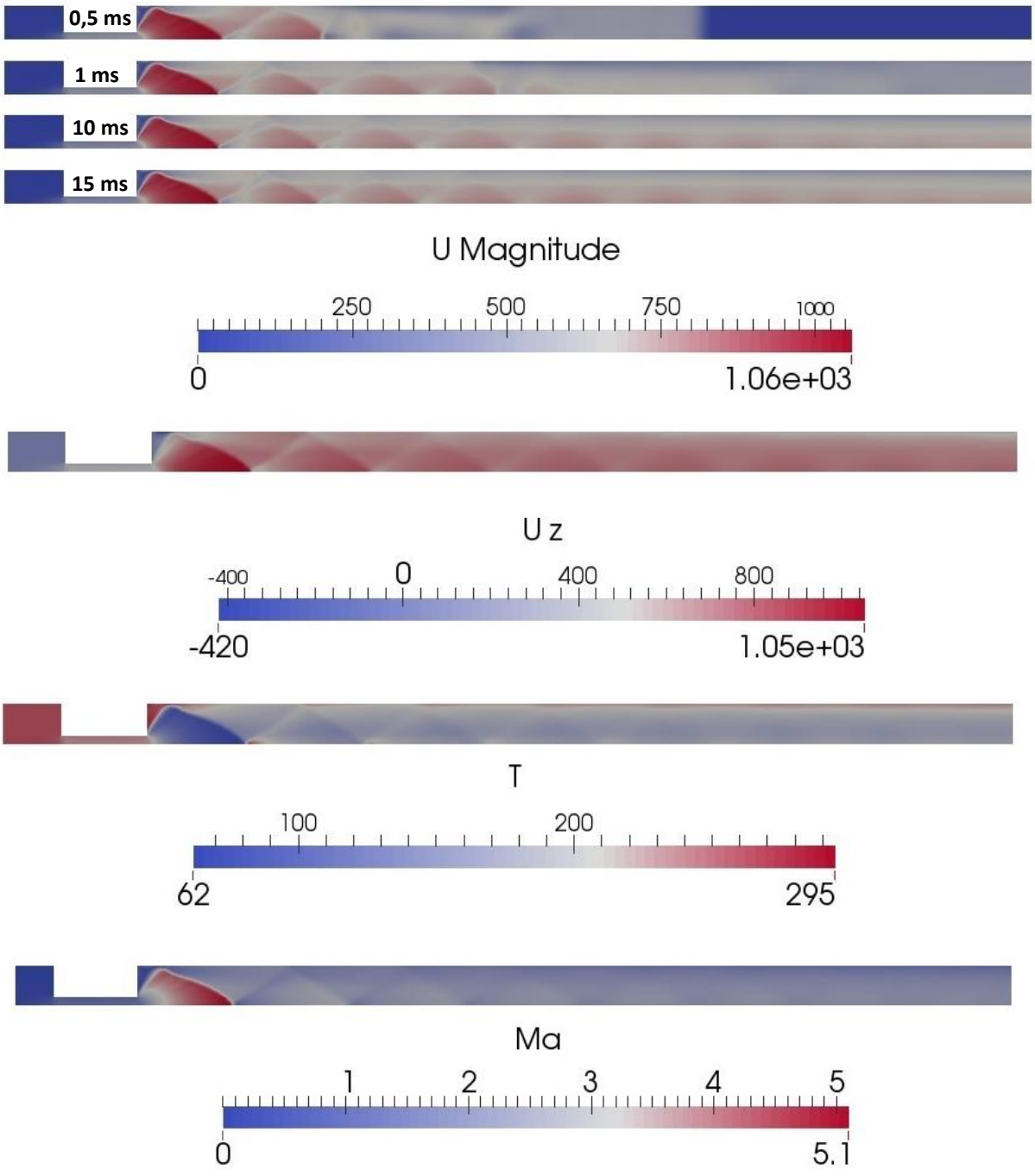

Figure 1 Results from case 1

B.1.2. Case 1P

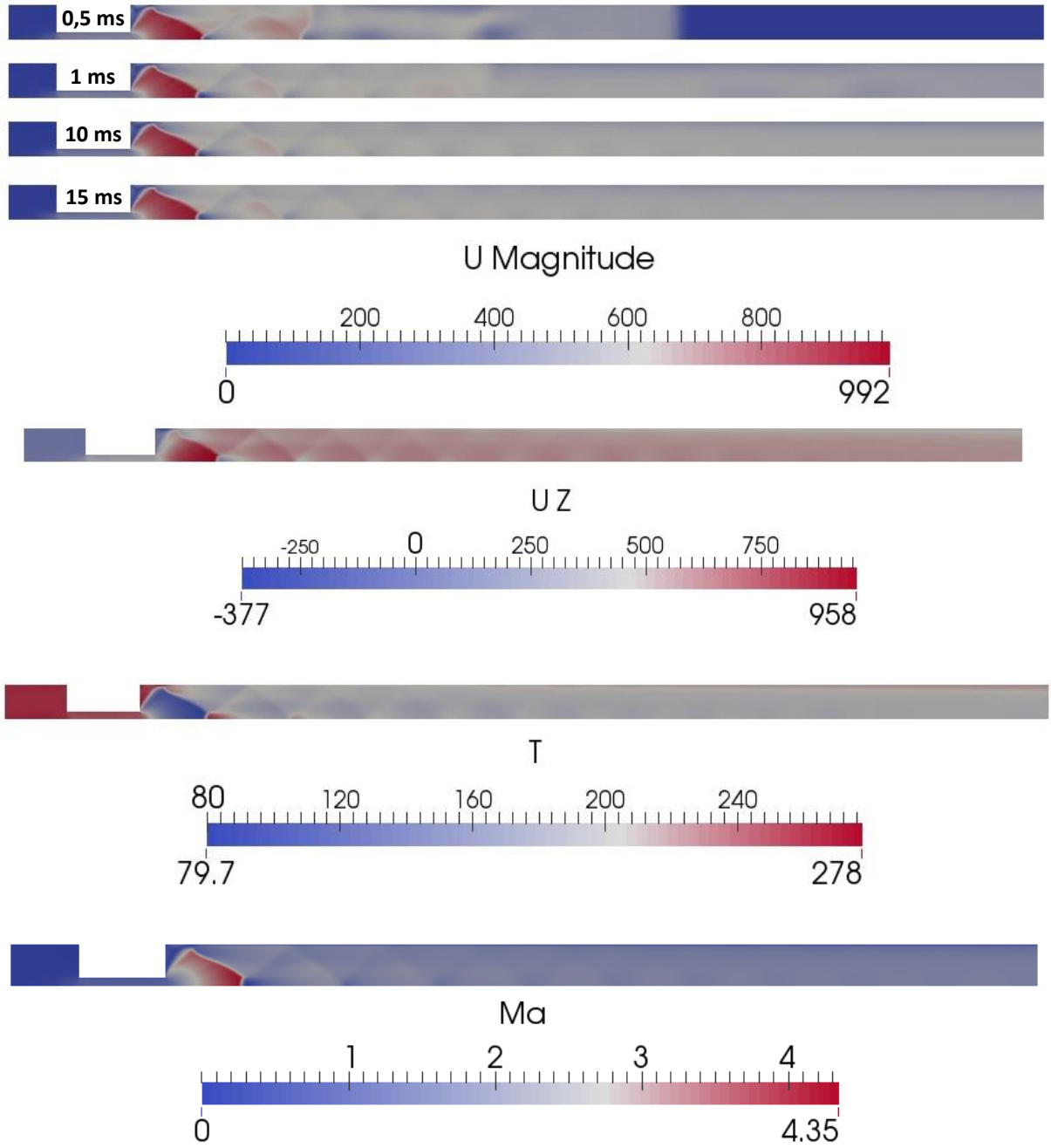

Figure 2 Results from case 1P

B.1.3. Case 2

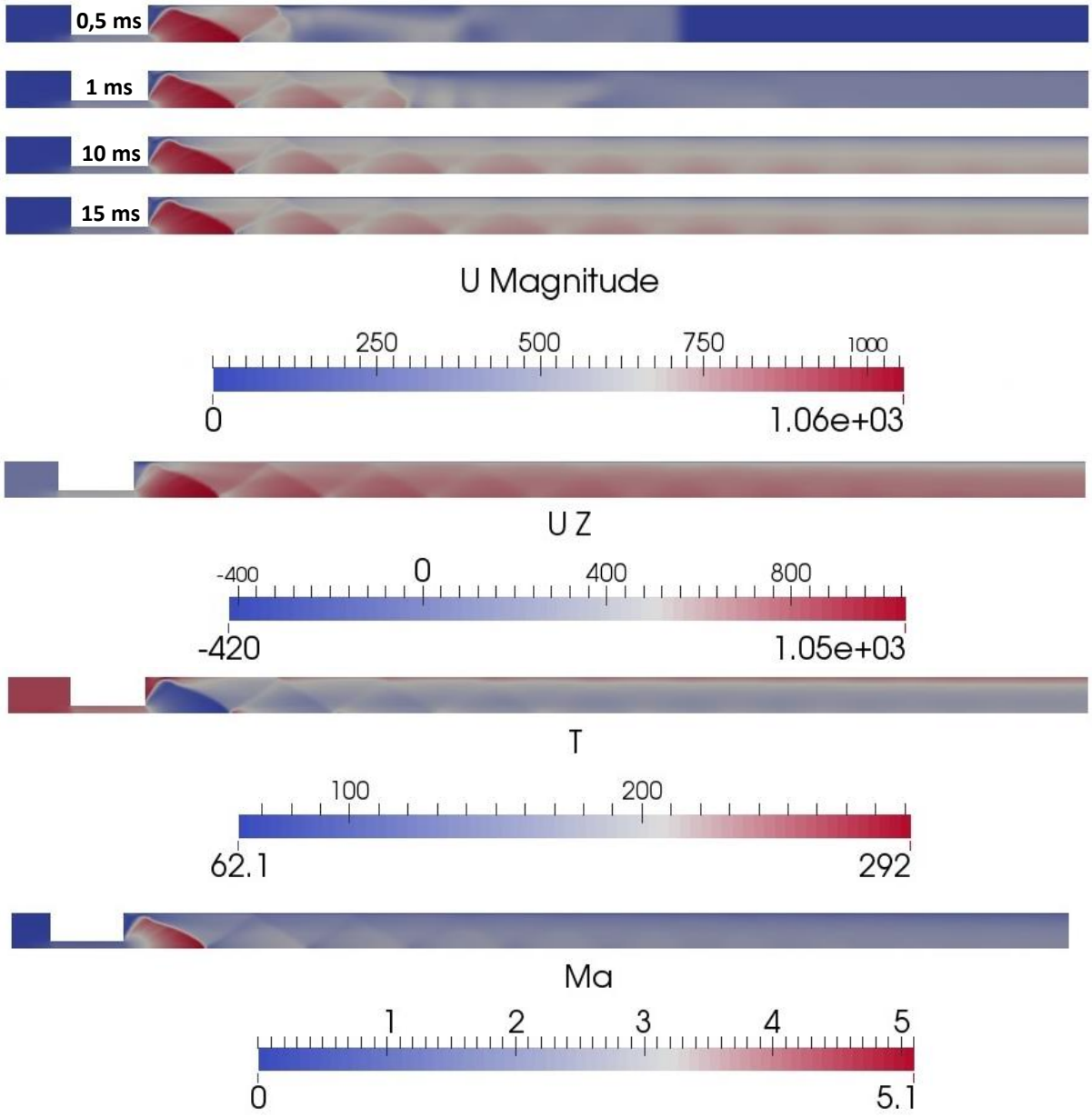

Figure 3 Results from case 2

B.1.4. Case 3

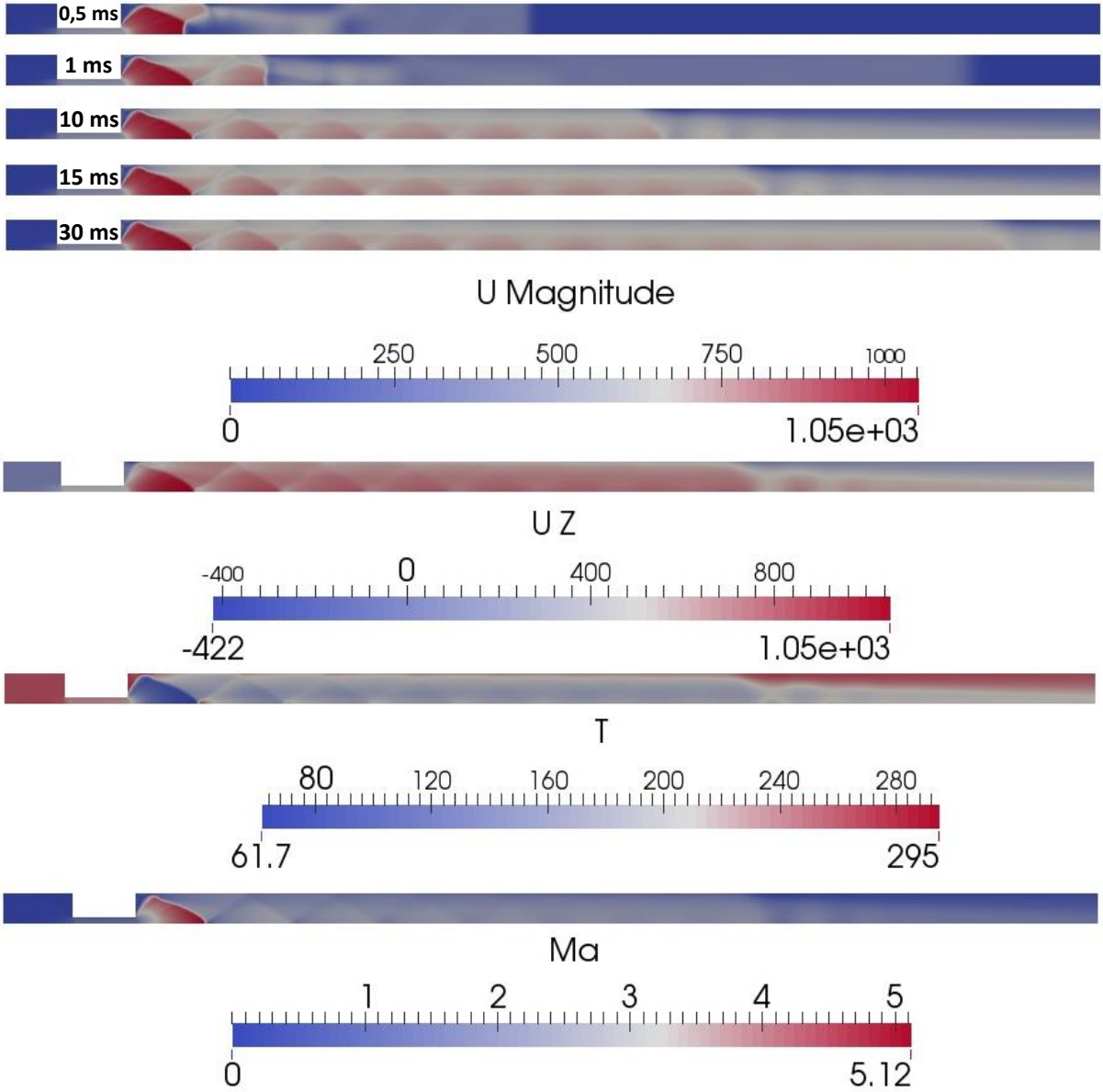

Figure 4 Results from case 3

B.1.5. Case 4

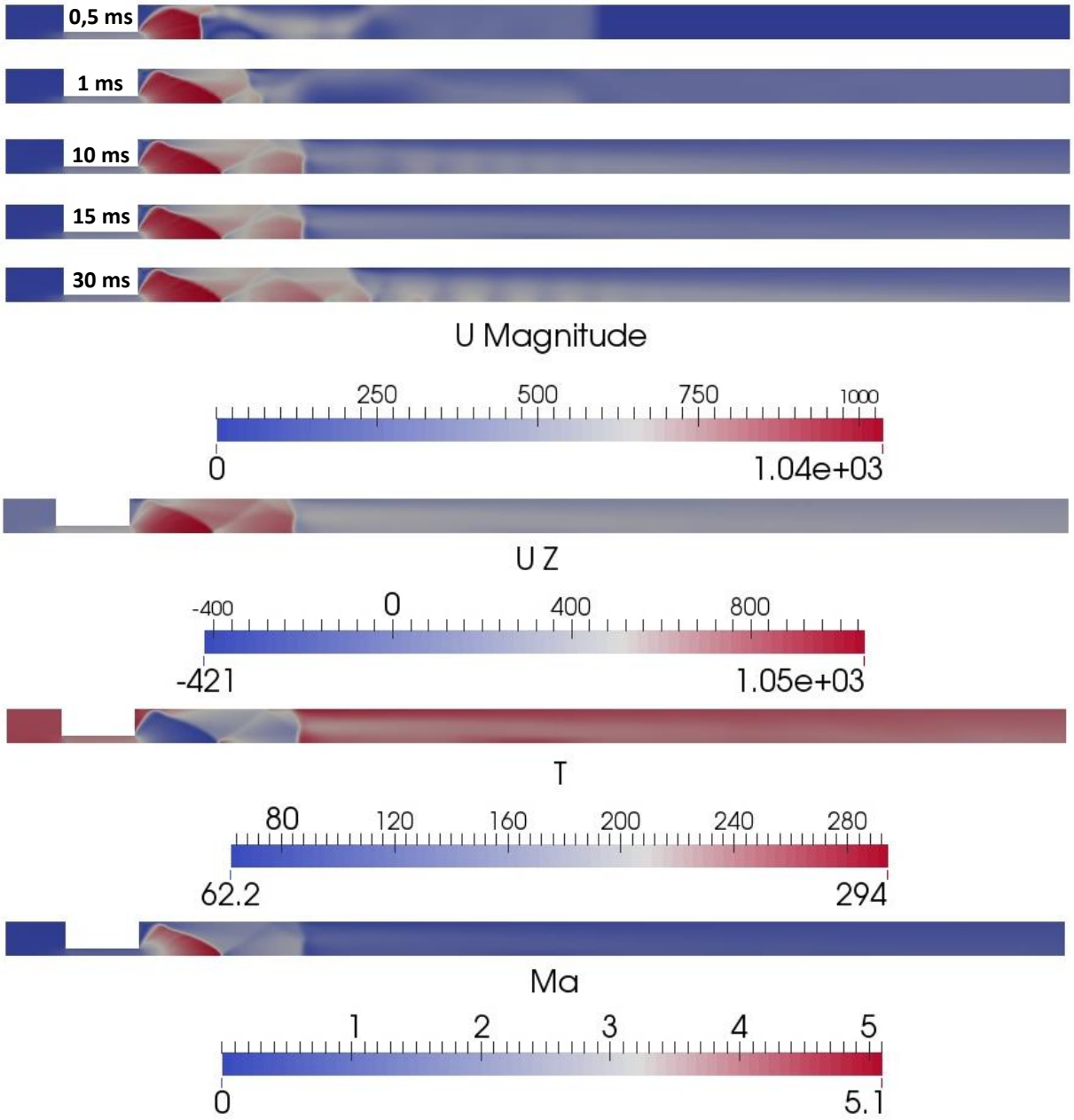

Figure 5 Results from case 4

B.1.6. Case 5

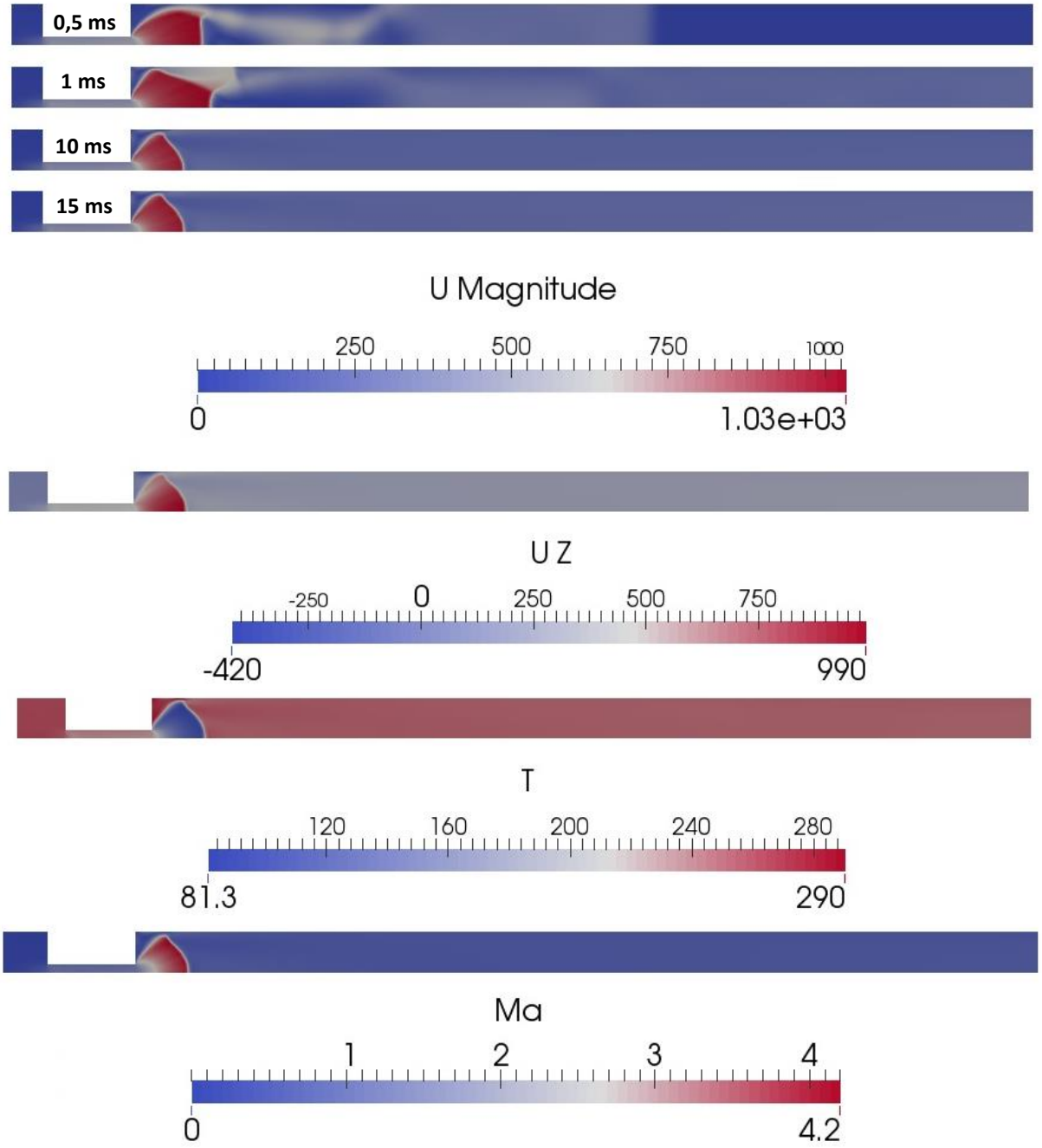

Figure 6 Results from case 5

B.1.7. Case 5P

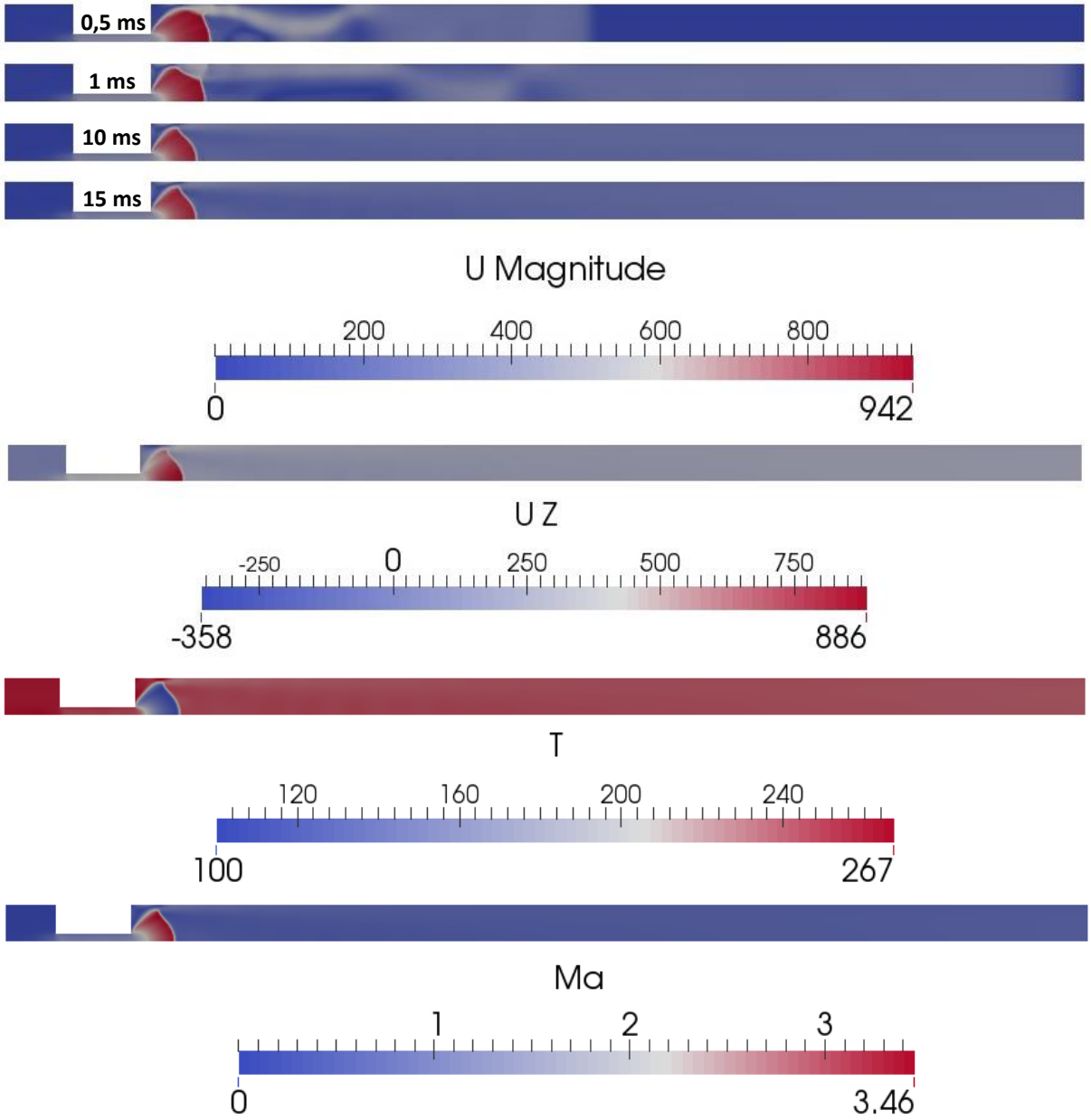

Figure 7 Results from case 5P

B.1.8. Case 6

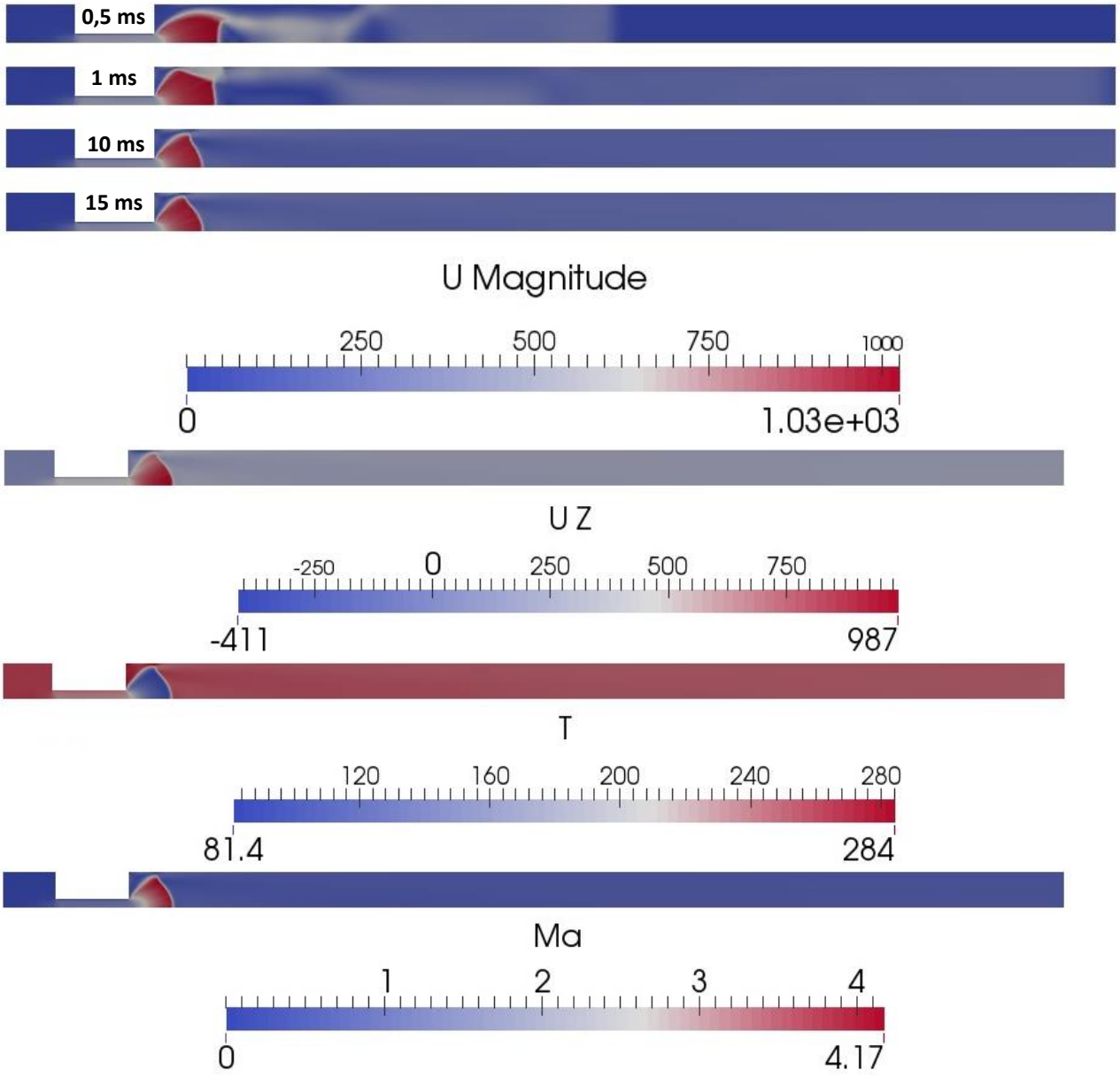

Figure 8 Results from case 6

B.1.9. Case 7

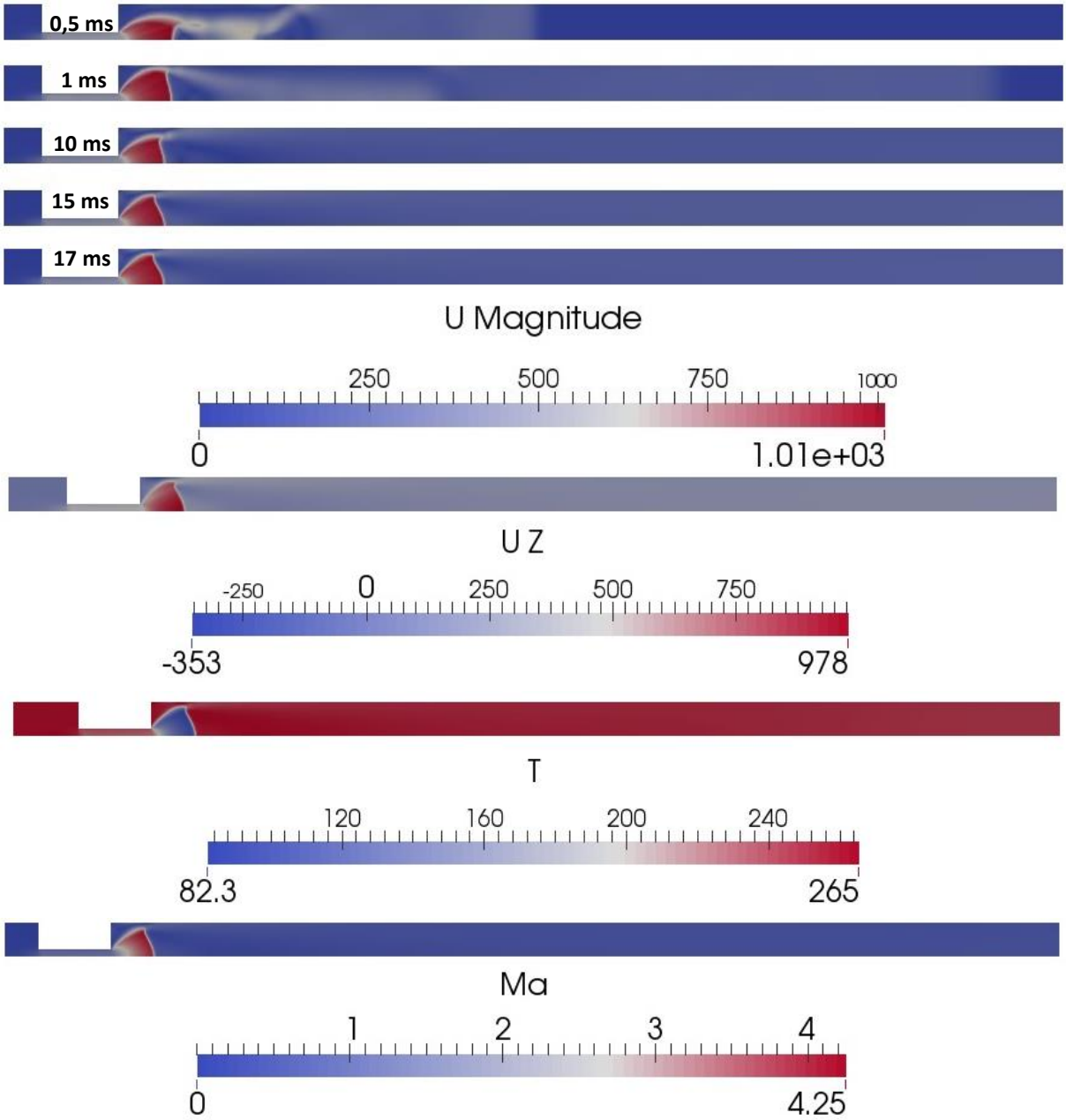

Figure 9 Results from case 7

B.1.10. Case 7P

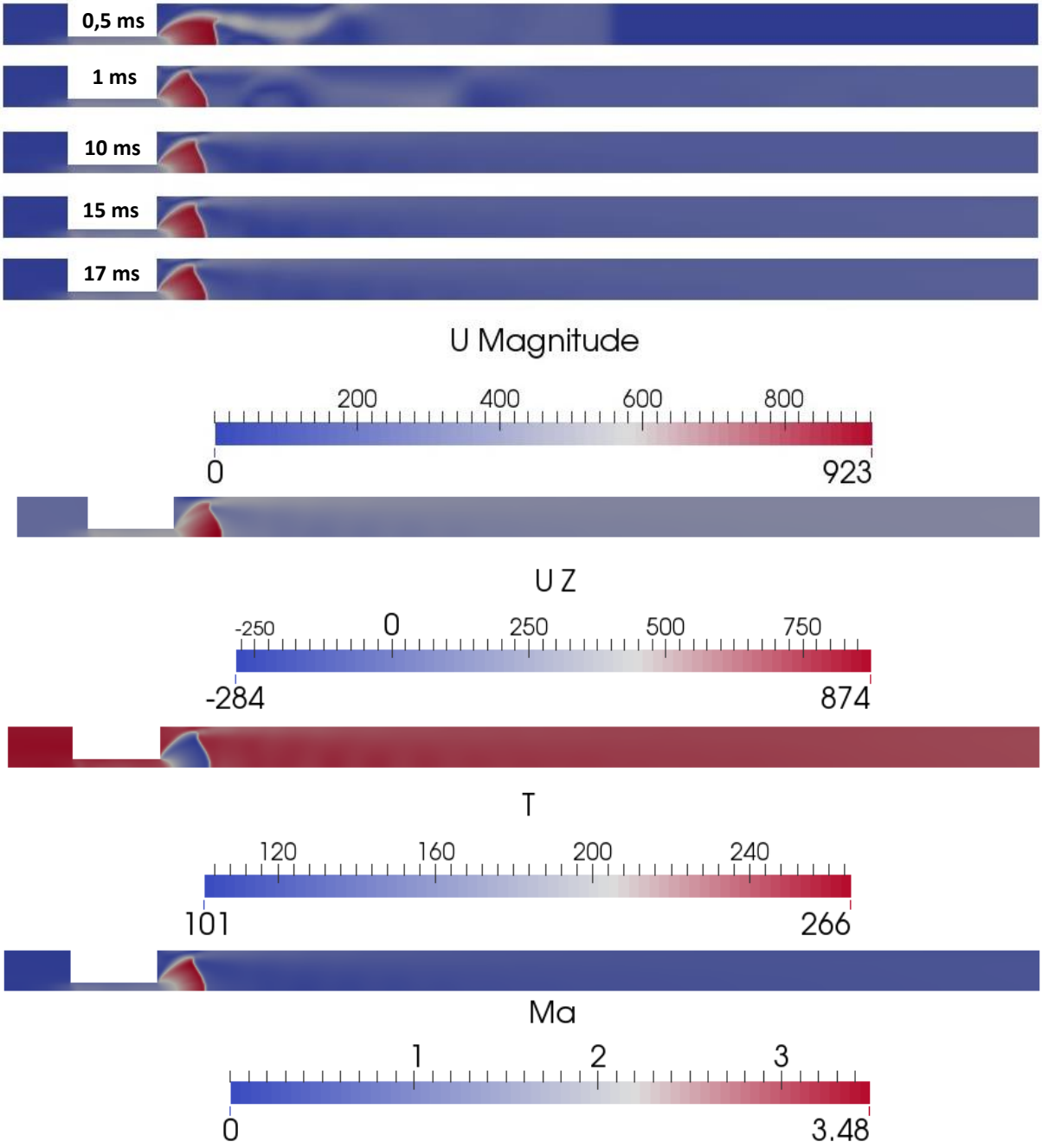

Figure 10 Results from case 7P

B.1.11. Case 8

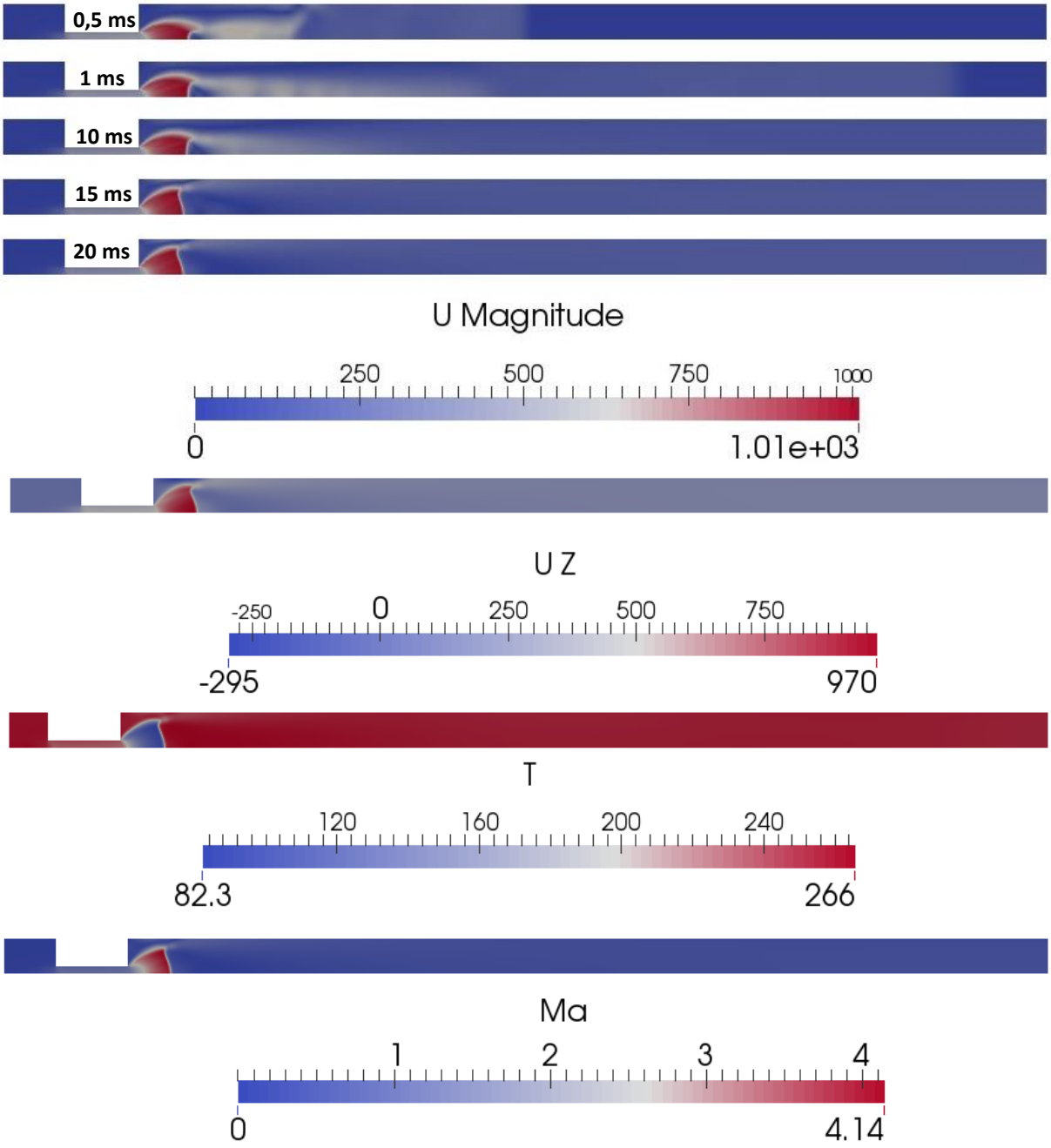

Figure 11 Results from case 8

B.1.12. Case 9

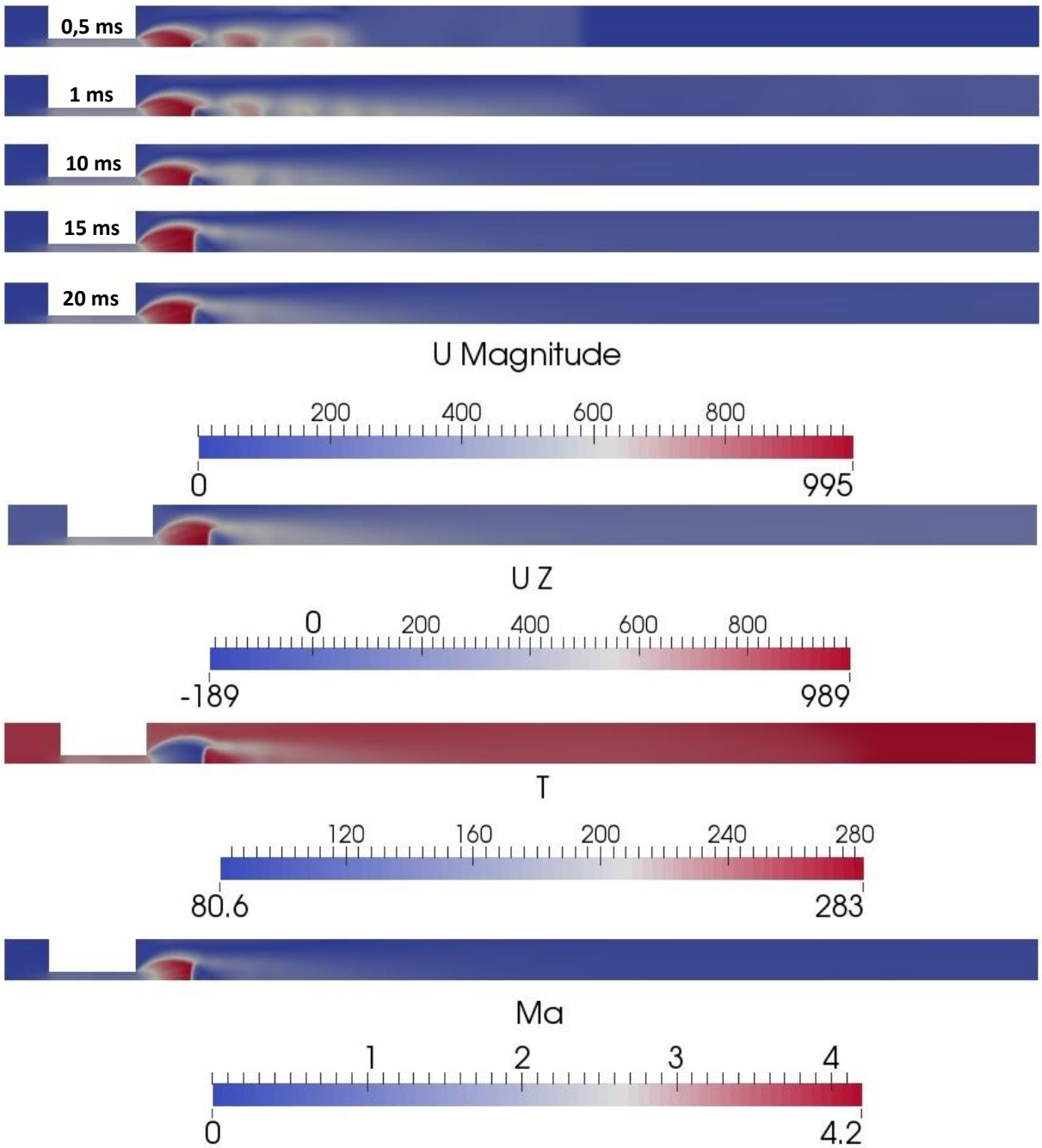

Figure 12 Results from case 9

B.1.13. Case 9P

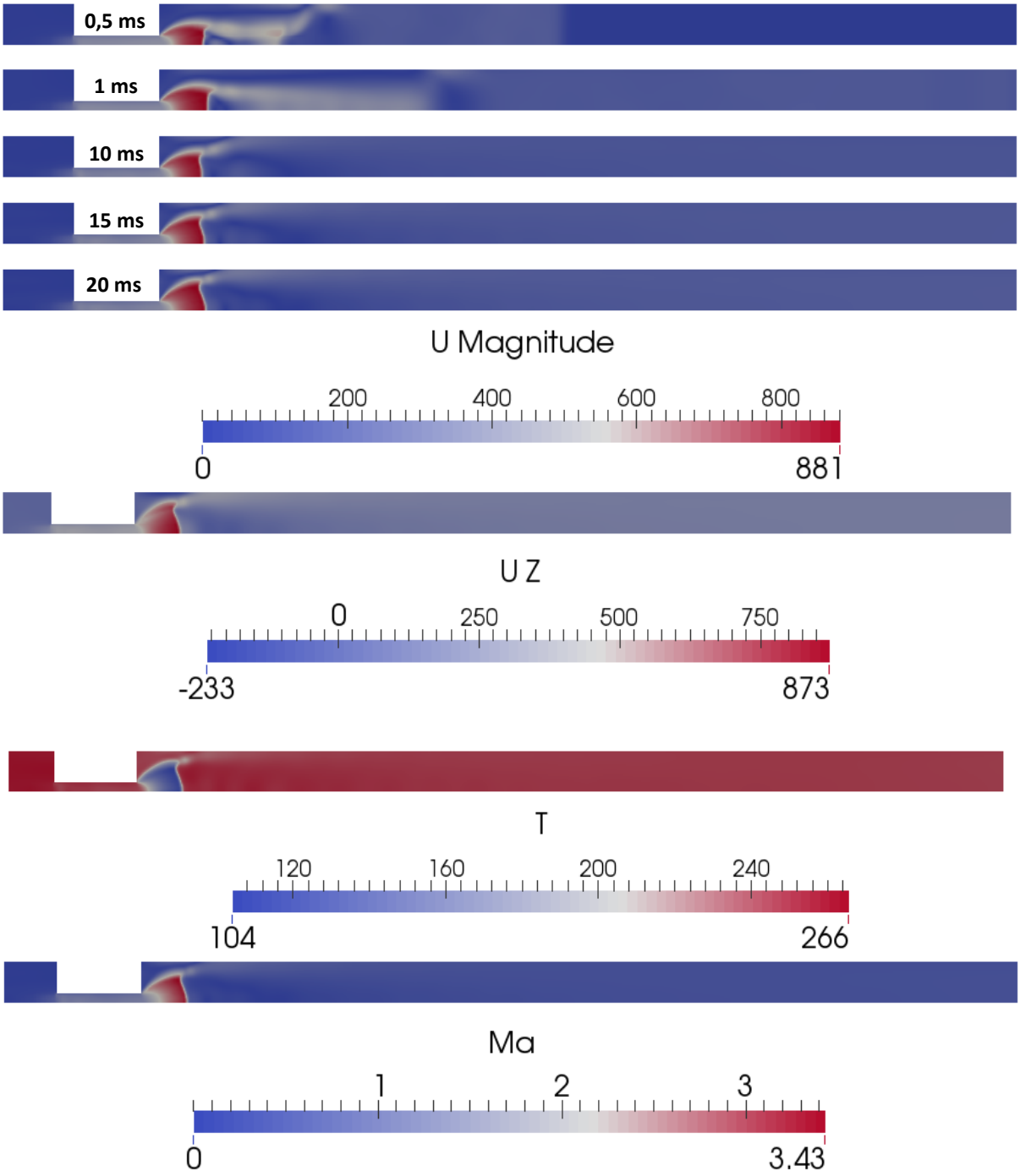

Figure 13 Results from case 9P

B.1.14. Case 10

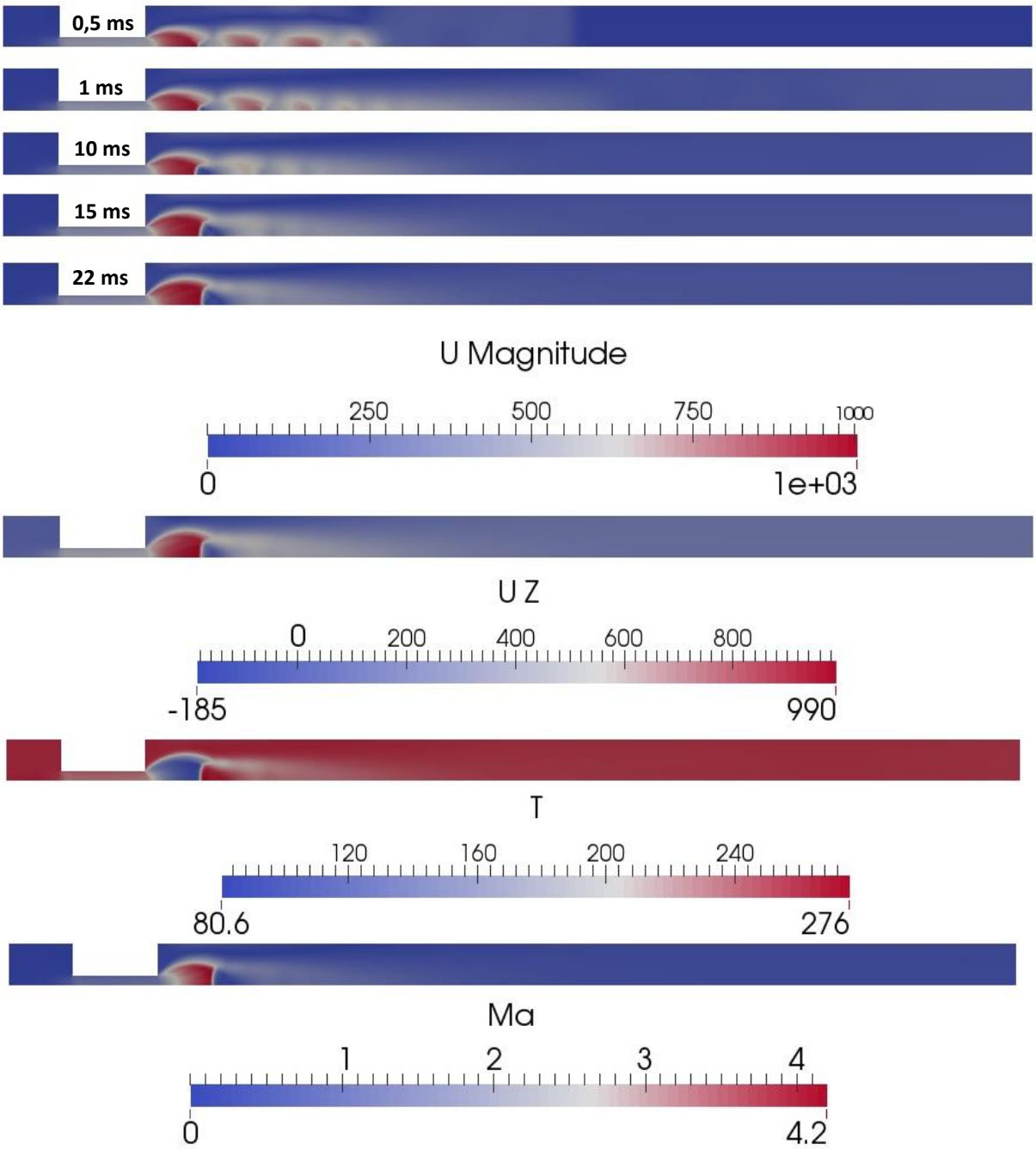

Figure 14 Results from case 10

B.1.15. Mach disk location and diameter for Peng-Robinson cases

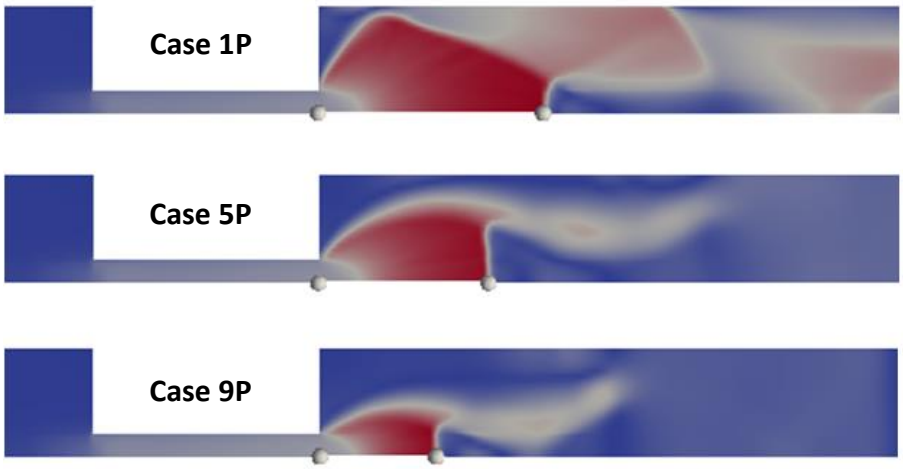

Figure 15 Mach disks at 0,3ms

Figure 15 shows the jet structure at 0,3 ms for case 1P, 5P and 9P. Case 1P is clearly influenced by the wall at this early stage, similar to case 1 in Figure 5-28.

B.2. Graphs and plots

B.2.1. Probed wall temperature as function of time

Similar plots for all cases, along both centerline and wall, can be found in the Excel-file "TempDevelopmentFluid" located in Appendix H.

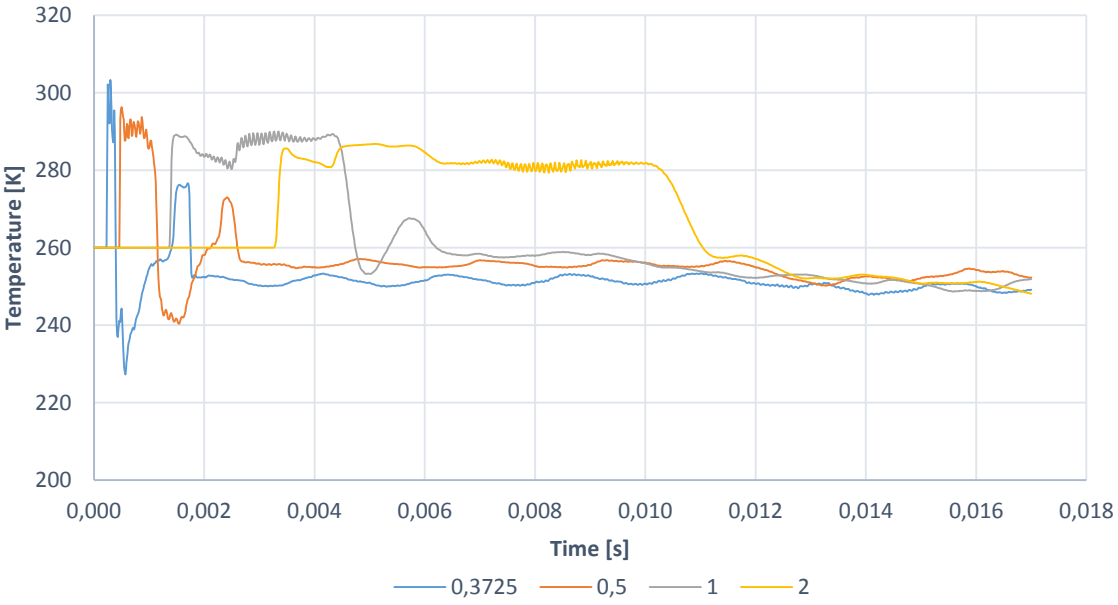

Figure 16 Temperature development along fluid wall, case 7

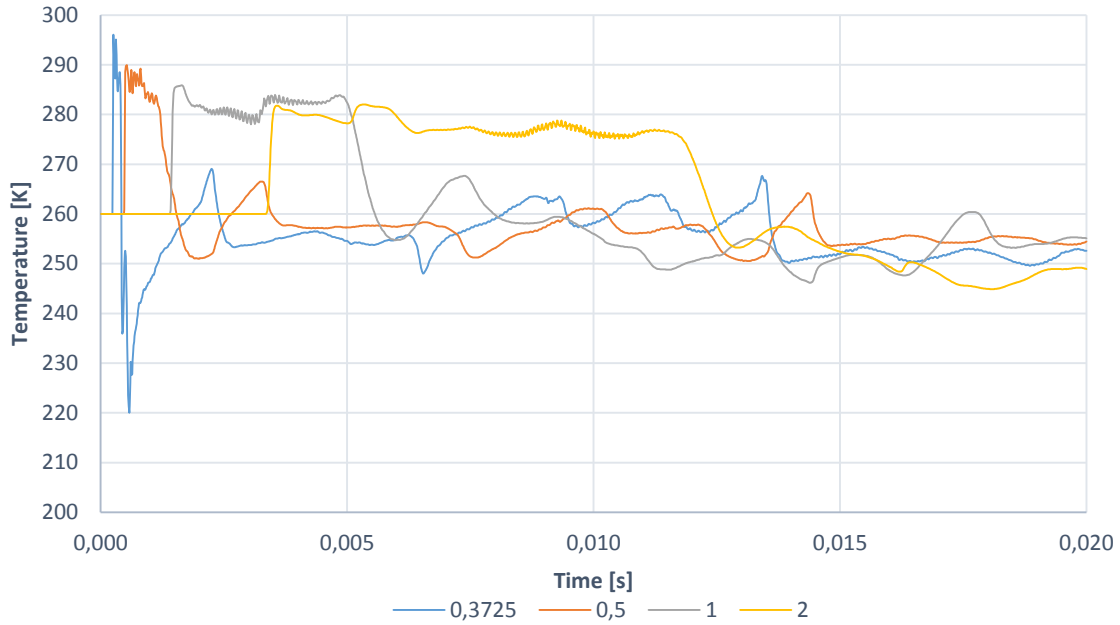

Figure 17 Temperature development along fluid wall, case 8

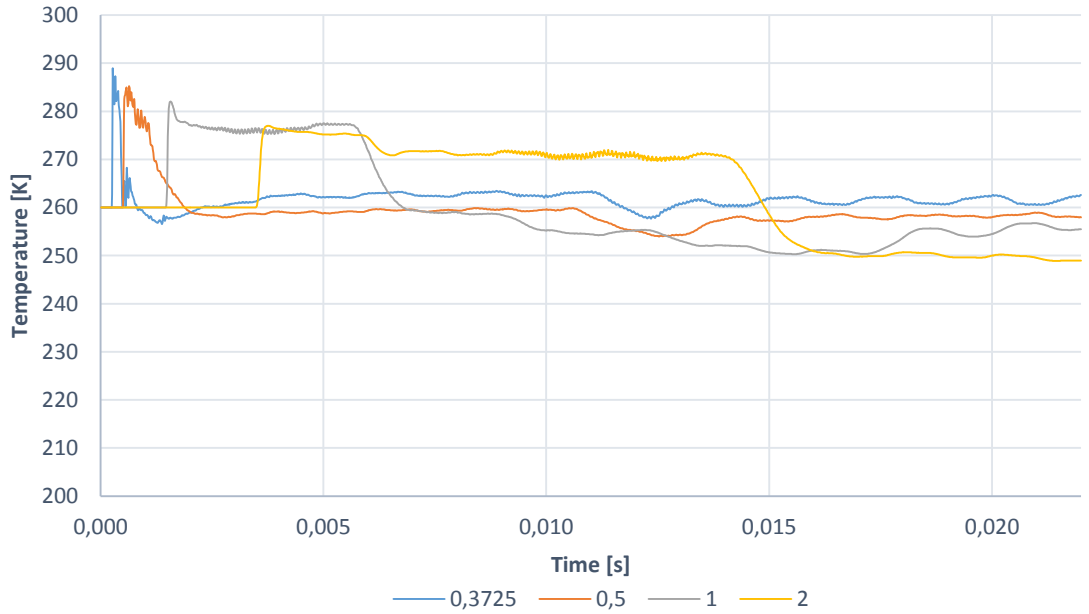

Figure 18 Temperature development along fluid wall, case 10

B.2.2. Temperature development at outer pipe wall

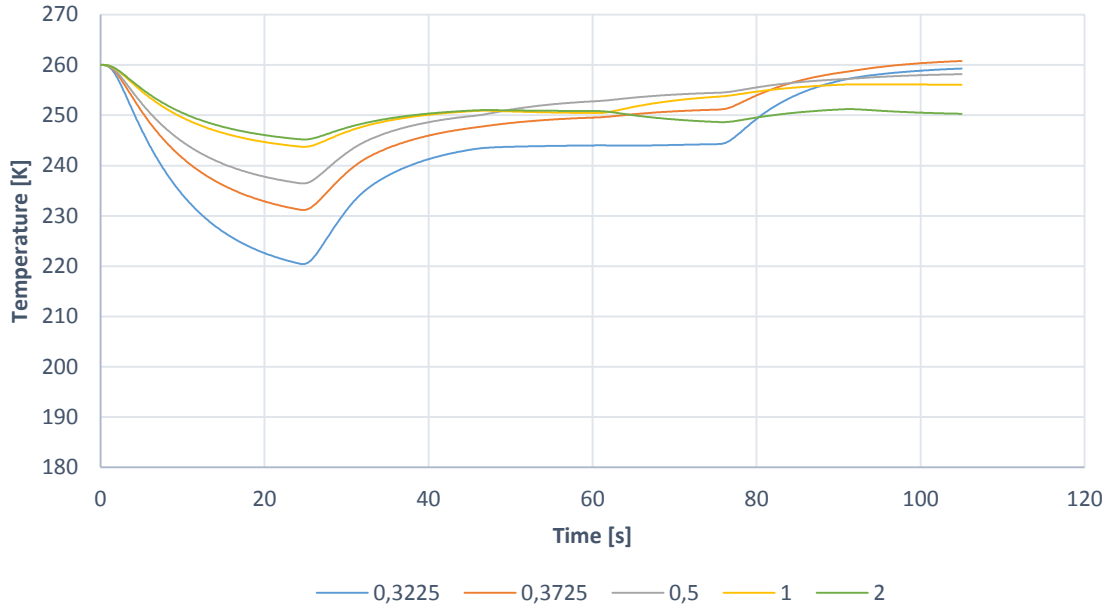

Figure 19 Temperature development at outer wall, case A

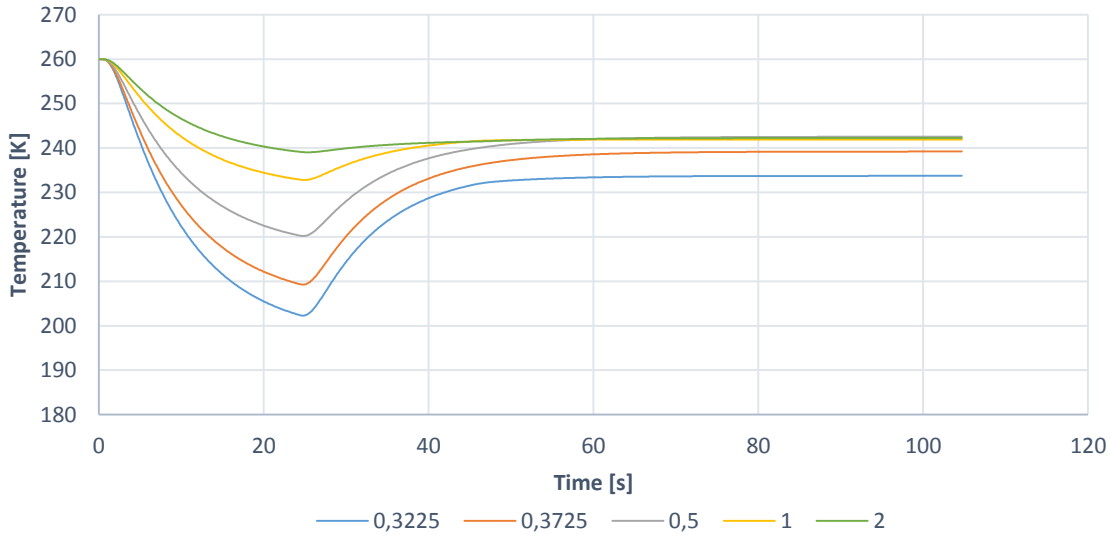

Figure 20 Temperature development at outer wall, case B

B.2.3. Time until min. pipe temperature is above min. design temperature

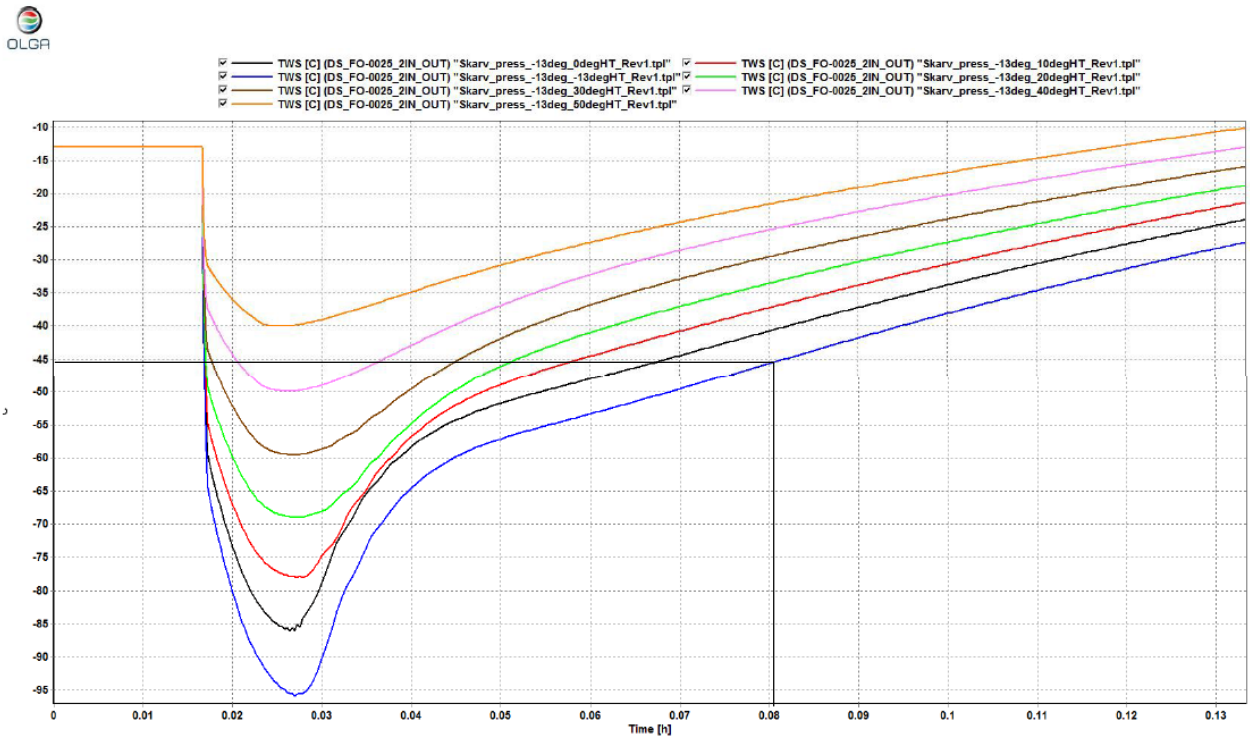

Figure 21 Time until min. pipe temperature reaches min. design temperature

Figure 21 shows the time until the minimum inner wall surface temperature in Aker's simulations reaches the minimum design temperature (the figure is the same as Figure 1-4). The simulation starts after 0,0167 h or 60 seconds. If we subtract this from 0,08 h, we find that the time until minimum design temperature is reached is 0,06333 h or 228 seconds.

B.2.4. Turbulence model comparison

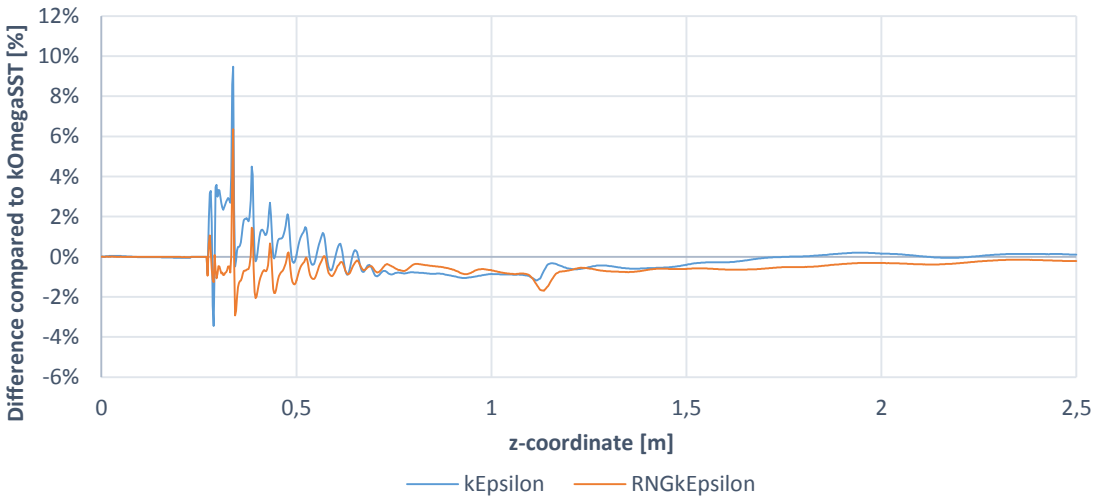

Figure 22 Difference in fluid surface temperature compared to kOmegaSST

Figure 22 shows how the difference in predicted fluid surface temperature is largest at the orifice outlet.

B.3. checkMesh output

```
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

Create time

Create polyMesh for time = 0

Time = 0

Mesh stats
    points:           22267
    internal points:  0
    faces:            43568
    internal faces:   21302
    cells:            10880
    faces per cell:   5.96232
    boundary patches: 6
    point zones:      0
    face zones:       0
    cell zones:       0

Overall number of cells of each type:
    hexahedra:     10470
    prisms:        410
    wedges:        0
    pyramids:      0
    tet wedges:    0
    tetrahedra:    0
    polyhedra:     0

Checking topology...
    Boundary definition OK.
    Cell to face addressing OK.
    Point usage OK.
    Upper triangular ordering OK.
    Face vertices OK.
    Number of regions: 1 (OK).

Checking patch topology for multiply connected surfaces...
    Patch               Faces    Points   Surface topology
    inlet               28       57       ok (non-closed singly connected)
    outlet              28       57       ok (non-closed singly connected)
    fluid_surface       450      902      ok (non-closed singly connected)
    front               10880    11339    ok (non-closed singly connected)
    back                10880    11339    ok (non-closed singly connected)
    defaultFaces        0        0        ok (empty)

Checking geometry...
    Overall domain bounding box (0 -0.0009 0) (0.02141 0.0009 2.5)
    Mesh (non-empty, non-wedge) directions (1 0 1)
```

```
    Mesh (non-empty) directions (1 1 1)
    Wedge front with angle 2.40709 degrees
    Wedge back with angle 2.40709 degrees
    All edges aligned with or perpendicular to non-empty directions.
    Boundary openness (-9.89697e-16 7.80458e-16 7.43822e-21) OK.
    Max cell openness = 2.20514e-16 OK.
    Max aspect ratio = 608.386 OK.
    Minimum face area = 2.81914e-09. Maximum face area = 4.41449e-05.  Face area
magnitudes OK.
    Min volume = 1.07029e-12. Max volume = 4.09823e-08.  Total volume = 4.73444e-
05.  Cell volumes OK.
    Mesh non-orthogonality Max: 0 average: 0
    Non-orthogonality check OK.
    Face pyramids OK.
    Max skewness = 0.330981 OK.
    Coupled point location match (average 0) OK.

Mesh OK.
```

Appendix C Calculations

C.1. Time until warm gas reaches orifice

When the 25,1 m 12” pipe upstream of the orifice is depleted of cold, stagnant gas, and warm gas reaches the orifice, the gas temperature is assumed to increase to 323 K. The time until the warm gas reaches the orifice can be calculated from the gas volume in the 12” pipe and an estimated mass flow rate. The volume of the cold, stagnant gas in the 12” pipe, upstream of the orifice:

$$V_{gas} = \frac{\pi \cdot d_{12"}^2}{4} \cdot L_{12"} = \frac{\pi \cdot 0{,}2519^2}{4} \cdot 25{,}1 = 1{,}25\ m^3 \quad \text{(I)}$$

The gas temperature is assumed to be 260K and the pressure 235 bar ($2{,}38114 \cdot 10^7$ Pa). Number of moles is calculated using the ideal gas law.

$$n = \frac{p \cdot V_{gas}}{R \cdot T_{12"}} = \frac{2{,}38114 \cdot 10^7 \cdot 1{,}25}{8{,}314 \cdot 260} = 13769{,}3 \quad \text{(II)}$$

Assuming a molar weight of 19,19 g/mol, see section 4.5.1 for details, the total mass of the stagnant gas becomes:

$$m_{gas} = \frac{19{,}19 \cdot 13769{,}3}{1000} = 264{,}23\ kg \quad \text{(III)}$$

Since the flow through the orifice is choked, the mass flow rate can be calculated from equation (2-14). According to Aker’s report, the orifice has a diameter of 9,08 mm and a coefficient of discharge is 0,84. The ratio of specific heats is assumed to be 1,4, and the density is calculated from the mass and the volume of the gas.

$$\dot{m}_{gas} = 0{,}84 \cdot \frac{\pi \cdot 0{,}00908^2}{4} \cdot \sqrt{1{,}4 \cdot \frac{264{,}23}{1{,}25} \cdot 2{,}38114 \cdot 10^7 \cdot \left(\frac{2}{1{,}4+1}\right)^{\frac{1{,}4+1}{1{,}4-1}}} = 2{,}64\ \frac{kg}{s} \quad \text{(IV)}$$

The time until the warm gas reaches the orifice can now be calculated from the total mass and the estimated mass flow rate.

$$t_{gas} = \frac{264{,}23}{2{,}642} = 100\ s \quad \text{(V)}$$

See Figure 1-4 to compare this result with Aker’s simulations. Note that the simulation starts after 60 seconds of elapsed time. According to our estimate, the warm gas should reach the orifice after 160 seconds or 0,044 h. At this point, the graph seems to transition into a linear increase. The cooling effect of the under-expanded jet is most severe the first 30 seconds, until the pressure reaches 5 bar, see section 5.2.2. When the warm gas reaches the orifice, the backpressure is approximately 16 bars and the cooling effect in the jet has decreased. The warm gas provides an approximately constant gas temperature inside the pipe, which

results in a linear increase of the pipe temperature. Based on these observations, our estimate of 100 s therefore seems to be reasonable.

C.2. Parameters for Peng-Robinson

The Peng-Robinson equation of state implemented in OpenFOAM requires the specification of four parameters. It is the critical pressure, volume and temperature of the gas, as well as an acentric factor. The critical temperature and the acentric factor for methane, is taken from reference [64], while the critical volume and pressure are taken from [65]. See Appendix D.2.4 for details.

C.2.1. Sutherlands equation

Sutherland's equation is used to calculate the dynamic viscosity as a function of temperature [57],[62]:

$$\mu = \frac{A_{su}\sqrt{T}}{1 + \frac{T_{su}}{T}} = \frac{A_{su}T^{\frac{3}{2}}}{T + T_{su}} \tag{VI}$$

T is the temperature, T_{su} is Sutherlands constant and A_{su} can be calculated from equation (VII)

$$A_s = \frac{\mu_0(T_0 + T_{su})}{T_{su}^{\frac{3}{2}}} \tag{VII}$$

μ_0 is the dynamic viscosity at reference temperature T_0. According to [66] T_{su} is 198 for methane at atmospheric pressure. According to Figure 4-15, μ_0 is set to 10^{-5} at T_0=260 K. A_{su} then becomes:

$$A_{su} = \frac{10^{-5}(260 + 198)}{198^{\frac{3}{2}}} = 1{,}0925 \cdot 10^{-6} \tag{VIII}$$

A_{su} and T_{su} are specified in the thermophysicalProperties dictionary.

Figure 23 shows how the viscosity predicted by Sutherland's formula is very close to the actual viscosity.

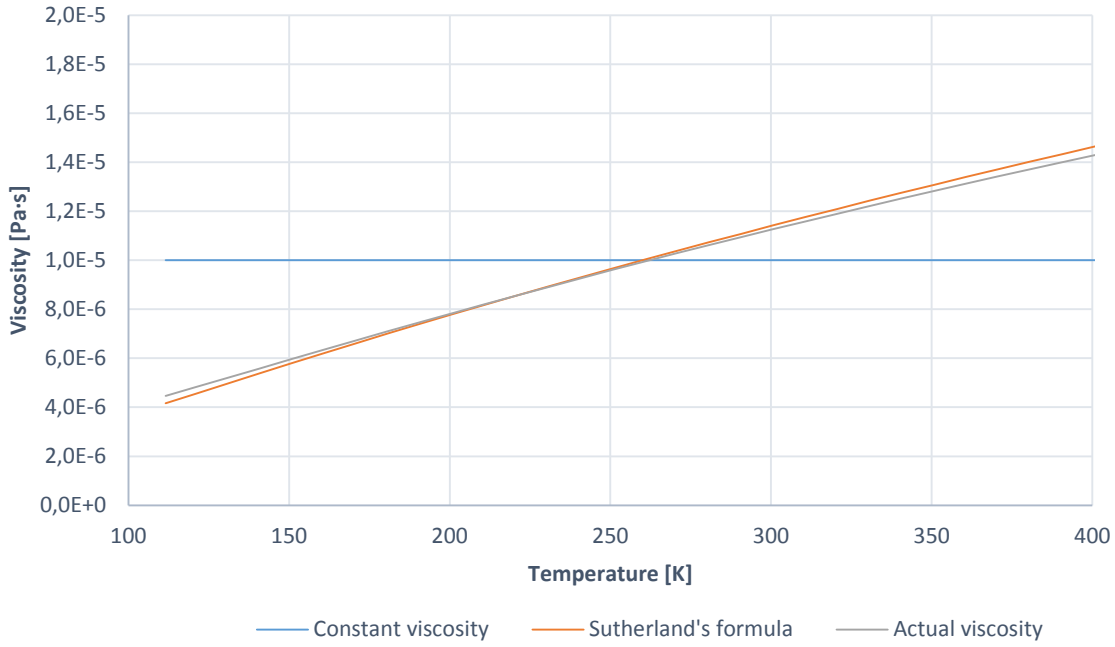

Figure 23 Viscosity predicted by Sutherland's formula

C.3. Pressure while pipe temperature is below min. design temperature

The minimum temperature in the pipe becomes larger than the minimum design temperature after approximately 40 seconds in our simulations, see section 5.2.3. The maximum pressure that occurs while the temperature is below the limit is:

$$1bar + \frac{40\ sec}{60\ sec} \cdot 10\ \frac{bar}{min} = 7{,}7\ bar \quad \text{(IX)}$$

The equivalent time in Aker's simulations is estimated to 228 seconds. The maximum pressure then becomes:

$$1bar + \frac{228\ sec}{60\ sec} \cdot 10\ \frac{bar}{min} = 39\ bar \quad \text{(X)}$$

Appendix D rhoCentralFoam dictionaries

This appendix contains a selection of rhoCentralFoam dictionaries. They are listed according to their parent folder. All dictionaries can be found among the case files in Appendix H.

D.1. 0 folder

D.1.1. p before setFields

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{   version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [1 -1 -2 0 0 0 0];
internalField   uniform 101325;
boundaryField
{
    inlet
    {
        type            fixedValue;
        value           uniform 23811375;
    }
    outlet
    {
        type            waveTransmissive;
        gamma           1.4;
        fieldInf        101325;
        lInf            2;
        value           uniform 101325;
    }
    fluid_surface
    {
        type            zeroGradient;
    }
    front
    {
        type            wedge;
    }
    back
    {
        type            wedge;
    }
}
// ************************************************************************* //
```

D.1.2. U

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

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

internalField   uniform (0 0 0);

boundaryField
{
    inlet
    {
        type            pressureInletOutletVelocity;
        value           uniform (0 0 0);
    }
    outlet
    {
        type            inletOutlet;     // if flow is inwards -> fixedValue
        inletValue      uniform (0 0 0); // if flow is outwards -> zeroGradient
        value           uniform (0 0 0);
    }
    fluid_surface
    {
        type            fixedValue;
        value           uniform (0 0 0);
    }
    front
    {
        type            wedge;
    }
    back
    {
        type            wedge;
    }
}

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

D.1.3. T

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      T;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 0 0 1 0 0 0];

internalField   uniform 260;

boundaryField
{
    inlet
    {
        type            fixedValue;
        value           uniform 260;
    }
    outlet
    {
        type            zeroGradient;
    }
    fluid_surface
    {
        type            zeroGradient;
    }
    front
    {
        type            wedge;
    }
    back
    {
        type            wedge;
    }
}

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

D.1.4. k

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      k;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

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

internalField   uniform 0.177;

boundaryField
{
    inlet
    {
        type            inletOutlet;
        inletValue      $internalField;
        value           $internalField;
    }
    outlet
    {
        type            inletOutlet;
        inletValue      $internalField;
        value           $internalField;
    }
    fluid_surface
    {
        type            compressible::kqRWallFunction;
        value           $internalField;
    }
    front
    {
        type            wedge;
    }
    back
    {
        type            wedge;
    }
}

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

D.1.5. omega

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.x                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      omega;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

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

internalField   uniform 258.6;

boundaryField
{
    inlet
    {
        type            inletOutlet;
        inletValue      $internalField;
        value           $internalField;
    }
    outlet
    {
        type            inletOutlet;
        inletValue      $internalField;
        value           $internalField;
    }
    fluid_surface
    {
        type            compressible::omegaWallFunction;
        value           $internalField;
        Cmu             0.09;
        kappa           0.41;
        E               9.8;
        beta1           0.075;
    }
    front
    {
        type            wedge;
    }
    back
    {
        type            wedge;
    }
}

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

D.2. constant folder

D.2.1. blockMeshDict

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
convertToMeters 1;   // 10880 cells
vertices
(   (0 0 0)
    (0.00304 0.000127790752 0)
    (0.00304 0.000127790752 0.226)
    (0 0 0.226)
    (0.00304 -0.000127790752 0)
    (0.00304 -0.000127790752 0.226)
    (0.00454 0.000190845399 0)
    (0.00454 0.000190845399 0.226)
    (0.00454 -0.000190845399 0)
    (0.00454 -0.000190845399 0.226)
    (0.01991 0.000836945353 0)
    (0.01991 0.000836945353 0.226)
    (0.01991 -0.000836945353 0)
    (0.01991 -0.000836945353 0.226)
    (0.02141 0.0009 0)
    (0.02141 0.0009 0.226)
    (0.02141 -0.0009 0)
    (0.02141 -0.0009 0.226)
    (0.02141 0.0009 0.2275)
    (0.01991 0.000836945353 0.2275)
    (0.02141 -0.0009 0.2275)
    (0.01991 -0.000836945353 0.2275)
    (0.00454 0.000190845399 0.2275)
    (0.00454 -0.000190845399 0.2275)
    (0.00304 0.000127790752 0.2275)
    (0.00304 -0.000127790752 0.2275)
    (0 0 0.2275)
    (0.00454 0.000190845399 0.2725)
    (0.00304 0.000127790752 0.2725)
    (0.00454 -0.000190845399 0.2725)
    (0.00304 -0.000127790752 0.2725)
    (0 0 0.2725)
    (0 0 0.274)
    (0.00304 0.000127790752 0.274)
    (0.00304 -0.000127790752 0.274)
    (0.00454 0.000190845399 0.274)
    (0.00454 -0.000190845399 0.274)
```

```
    (0.01991 0.000836945353 0.2725)
    (0.01991 0.000836945353 0.274)
    (0.01991 -0.000836945353 0.2725)
    (0.01991 -0.000836945353 0.274)
    (0.02141 0.0009 0.2725)
    (0.02141 0.0009 0.274)
    (0.02141 -0.0009 0.2725)
    (0.02141 -0.0009 0.274)
    (0.02141 0.0009 0.5)
    (0.01991 0.000836945353 0.5)
    (0.02141 -0.0009 0.5)
    (0.01991 -0.000836945353 0.5)
    (0.00454 0.000190845399 0.5)
    (0.00454 -0.000190845399 0.5)
    (0.00304 0.000127790752 0.5)
    (0.00304 -0.000127790752 0.5)
    (0 0 0.5)
    (0 0 1)
    (0.00304 0.000127790752 1)
    (0.00304 -0.000127790752 1)
    (0.00454 0.000190845399 1)
    (0.00454 -0.000190845399 1)
    (0.01991 0.000836945353 1)
    (0.01991 -0.000836945353 1)
    (0.02141 0.0009 1)
    (0.02141 -0.0009 1)
    (0.02141 0.0009 2.5)
    (0.02141 -0.0009 2.5)
    (0.01991 0.000836945353 2.5)
    (0.01991 -0.000836945353 2.5)
    (0.00454 0.000190845399 2.5)
    (0.00454 -0.000190845399 2.5)
    (0.00304 0.000127790752 2.5)
    (0.00304 -0.000127790752 2.5)
    (0 0 2.5)
);
blocks
(   hex (0 1 2 3 0 4 5 3) (3 20 1) simpleGrading (1 1 1)
    hex (1 6 7 2 4 8 9 5) (5 20 1) simpleGrading (0.05 1 1)
    hex (6 10 11 7 8 12 13 9) (15 20 1) simpleGrading (1 1 1)
    hex (10 14 15 11 12 16 17 13) (5 20 1) simpleGrading (0.075 1 1)
    hex (11 15 18 19 13 17 20 21) (5 5 1) simpleGrading (0.075 0.1 1)
    hex (7 11 19 22 9 13 21 23) (15 5 1) simpleGrading (1 0.1 1)
    hex (2 7 22 24 5 9 23 25) (5 5 1) simpleGrading (0.05 0.1 1)
    hex (3 2 24 26 3 5 25 26) (3 5 1) simpleGrading (1 0.1 1)
    hex (24 22 27 28 25 23 29 30) (5 30 1) simpleGrading (0.05 1 1)
    hex (26 24 28 31 26 25 30 31) (3 30 1) simpleGrading (1 1 1)
    hex (31 28 33 32 31 30 34 32) (3 5 1) simpleGrading (1 10 1)
    hex (28 27 35 33 30 29 36 34) (5 5 1) simpleGrading (0.05 10 1)
    hex (27 37 38 35 29 39 40 36) (15 5 1) simpleGrading (1 10 1)
    hex (37 41 42 38 39 43 44 40) (5 5 1) simpleGrading (0.1 10 1)
    hex (38 42 45 46 40 44 47 48) (5 150 1) simpleGrading (0.1 4 1)
    hex (35 38 46 49 36 40 48 50) (15 150 1) simpleGrading (1 4 1)
    hex (33 35 49 51 34 36 50 52) (5 150 1) simpleGrading (0.05 4 1)
    hex (32 33 51 53 32 34 52 53) (3 150 1) simpleGrading (1 4 1)
    hex (53 51 55 54 53 52 56 54) (3 100 1) simpleGrading (1 3 1)
    hex (51 49 57 55 52 50 58 56) (5 100 1) simpleGrading (0.05 3 1)
    hex (49 46 59 57 50 48 60 58) (15 100 1) simpleGrading (1 3 1)
    hex (46 45 61 59 48 47 62 60) (5 100 1) simpleGrading (0.1 3 1)
```

```
    hex (59 61 63 65 60 62 64 66) (5 100 1) simpleGrading (0.1 2.95 1)
    hex (57 59 65 67 58 60 66 68) (15 100 1) simpleGrading (1 2.95 1)
    hex (55 57 67 69 56 58 68 70) (5 100 1) simpleGrading (0.05 2.95 1)
    hex (54 55 69 71 54 56 70 71) (3 100 1) simpleGrading (1 2.95 1)
);
edges
();
boundary
(
    inlet
    {
        type patch;
        faces
        (
            (0 1 4 0)
            (1 6 8 4)
            (6 10 12 8)
            (10 14 16 12)
        );
    }
    outlet
    {
        type patch;
        faces
        (
            (71 71 70 69)
            (69 70 68 67)
            (67 68 66 65)
            (65 66 64 63)
        );
    }
    fluid_surface
    {
        type wall;
        faces
        (
            (14 15 17 16)
            (15 18 20 17)
            (18 19 21 20)
            (19 22 23 21)
            (22 27 29 23)
            (27 37 39 29)
            (37 41 43 39)
            (41 42 44 43)
            (42 45 47 44)
            (45 61 62 47)
            (61 63 64 62)
        );
     }
    front
    {
        type wedge;
        faces
        (
            (0 1 2 3)
            (1 6 7 2)
            (6 10 11 7)
            (10 14 15 11)
            (11 15 18 19)
```

```
            (7 11 19 22)
            (2 7 22 24)
            (3 2 24 26)
            (24 22 27 28)
            (26 24 28 31)
            (31 28 33 32)
            (28 27 35 33)
            (27 37 38 35)
            (37 41 42 38)
            (38 42 45 46)
            (35 38 46 49)
            (33 35 49 51)
            (32 33 51 53)
            (53 51 55 54)
            (51 49 57 55)
            (49 46 59 57)
            (46 45 61 59)
            (59 61 63 65)
            (57 59 65 67)
            (55 57 67 69)
            (54 55 69 71)
        );
    }
    back
    {
        type wedge;
        faces
        (

            (0 4 5 3)
            (4 8 9 5)
            (8 12 13 9)
            (12 16 17 13)
            (13 17 20 21)
            (9 13 21 23)
            (5 9 23 25)
            (3 5 25 26)
            (25 23 29 30)
            (26 25 30 31)
            (31 30 34 32)
            (30 29 36 34)
            (29 39 40 36)
            (39 43 44 40)
            (40 44 47 48)
            (36 40 48 50)
            (34 36 50 52)
            (32 34 52 53)
            (53 52 56 54)
            (52 50 58 56)
            (50 48 60 58)
            (48 47 62 60)
            (60 62 64 66)
            (58 60 66 68)
            (56 58 68 70)
            (54 56 70 71)
        );
    }
);
// ************************************************************************* //
```

D.2.2. turbulenceProperties and RASProperties

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

simulationType  RASModel;

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

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

RASModel kOmegaSST;

turbulence      on;

printCoeffs     on;

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

D.2.3. thermophysicalProperties (Ideal Gas)

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

thermoType
{
    type            hePsiThermo;
    mixture         pureMixture;
    transport       const;
    thermo          hConst;
    equationOfState perfectGas;
    specie          specie;
    energy          sensibleEnthalpy;
}
mixture
{
    specie
    {
        nMoles          1;
        molWeight       16.04;   // Pure CH4
    }
    thermodynamics
    {
        Cp              2180;         //  Units is J/kgK
        Hf              58.99e+3;     // Heat of fusion [J/kg]
    }
    transport
    {
       mu          1e-5;
       Pr          0.76;
    }
}

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

D.2.4. thermophysicalProperties (Peng-Robinson)

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

thermoType
{
    type            hePsiThermo;
    mixture         pureMixture;
    transport       sutherland;
    thermo          hConst
    equationOfState PengRobinsonGas;
    specie          specie;
    energy          sensibleEnthalpy;
}

mixture
{
    specie
    {
        nMoles          1;
        molWeight       16.04;         // Pure CH4
    }
    thermodynamics
    {
        Cp              2180;         //  Units is J/kgK
        Hf              58.99e+3;     // Heat of fusion [J/kg]
    }
    equationOfState
    {
        Tc      191.15;            // Critical temperature
        Vc      0.0062;            // Critical volume
        Pc      4.641e06;          // Critical pressure
        omega   0.015;             // Acentric factor
    }
    transport
    {

       As             1.0925e-06;
       Ts             198;


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

D.3. system folder

D.3.1. controlDict

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;

}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
libs
(
);
application       rhoCentralFoam;

startFrom         startTime;

startTime         0;

stopAt            endTime;

endTime           0.02;

deltaT            1e-9;

writeControl      adjustableRunTime;

writeInterval     0.0001;

cycleWrite        0;

writeFormat       ascii;

writePrecision    6;

writeCompression  off;

timeFormat        general;

timePrecision     6;

runTimeModifiable true;

adjustTimeStep    yes;

maxCo             0.5;   //0.9 after 0.001 s
```

```
maxDeltaT         1;

functions
{
    probes
    {
        // Where to load it from
        functionObjectLibs ( "libsampling.so" );
        type            probes;
        enabled         true;

        // Name of the directory for probe data
        name            probes;

        // Writecontrol
        outputControl   runTime;
        writeInterval   1e-5;

        // Fields to be probed
        fields
        (
            T
            U
            p
            rho
        );

        probeLocations
        (

// Along the centerline
       (0.0 0 0.250)        // inside orifice
       (0.0 0 0.2725)       // at orifice outlet
       (0.0 0 0.3225)       // 50 mm after orifice
       (0.0 0 0.3725)       // 100 mm after orifice
       (0.0 0 0.50)
       (0.0 0 1)
       (0.0 0 1.5)
       (0.0 0 2)
       (0.0 0 2.5)

// On the inner pipe wall
       (0.02141 0 0.2725)    // at orifice outlet
       (0.02141 0 0.3225)    // 50 mm after orifice
       (0.02141 0 0.3725)    // 100 mm after orifice
       (0.02141 0 0.50)
       (0.02141 0 1)
       (0.02141 0 1.5)
       (0.02141 0 2)
       (0.02141 0 2.5)

       );
}


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

D.3.2. fvSchemes

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
fluxScheme           Kurganov;

ddtSchemes
{
    default          Euler;
}
gradSchemes
{
    default          Gauss linear;
}
divSchemes
{
    default          none;
    div(tauMC)       Gauss linear;
    div(phi,U)       Gauss limitedLinearV 1;
    div(phi,h)       Gauss limitedLinear 1;
    div(phi,k)       Gauss limitedLinear 1;
    div(phi,omega)   Gauss linear;
}
laplacianSchemes
{
    default          Gauss linear corrected;
}
interpolationSchemes
{
    default          linear;
    reconstruct(rho) vanLeer;
    reconstruct(U)   vanLeerV;
    reconstruct(T)   vanLeer;
}
snGradSchemes
{
    default          corrected;
    snGrad(U)        corrected;
}
// ************************************************************************* //
```

D.3.3. fvSolution

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
solvers
{
    "(rho|rhoU|rhoE)"
    {
        solver          diagonal;
    }
    U
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        nSweeps         2;
        tolerance       1e-09;
        relTol          0.01;
    }
    h
    {
        $U;
        tolerance       1e-10;
        relTol          0;
    }
    e
    {
        $U;
        tolerance       1e-10;
        relTol          0;
    }
    "(k|epsilon|omega)"
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        nSweeps         2;
        tolerance       1e-09;
        relTol          0.01;
    }
}

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

D.3.4. sampleDict

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

interpolationScheme   cell;

setFormat             raw;

surfaceFormat         foamFile;

fields
(
                      T
);

surfaces
(
    fluid_surface
    {
        type          patch;
        patches       (fluid_surface);
        interpolate   false;
        triangulate    false;
    }
);

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

D.3.5. setFieldsDict

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

defaultFieldValues
(
     volVectorFieldValue U ( 0 0 0 )
     volScalarFieldValue T 260
     volScalarFieldValue p 101325
);

regions
(
     boxToCell
     {
           box ( 0 -0.001 0 ) ( 0.02141 0.001 0.2275 ) ;
           fieldValues ( volScalarFieldValue p 23811375 ) ;
     }
);

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

Appendix E chtMultiRegionFoam dictionaries

This appendix contains a selection of chtMultiRegionFoam dictionaries used to simulate the heat transfer in the pipe and case 11. Refer to case files in Appendix H for a complete set of dictionaries.

E.1. 0 folder

E.1.1. T (0/pipe) Case B

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{   version     2.0;
    format      ascii;
    class       volScalarField;
    object      T;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [0 0 0 1 0 0 0];
internalField   uniform 260;
boundaryField
{
    inlet_pipe
    {
        type            zeroGradient;
    }
    outlet_pipe
    {
        type            zeroGradient;
    }
    #include "include/T_avg1barPeng13to15ms" // read temperature field from fil
    outer_wall                               // heat flux from outside temp.
    {
        type            externalWallHeatFluxTemperature;
        kappa           solidThermo;
        Ta              uniform 260.0;      // ambient temperature /[K]
        h               uniform 7.9;        // heat transfer coeff /[W/Km2]
        value           uniform 260;        // initial temperature / [K]
        kappaName       none;
        thicknessLayers (0.050);            // insulation thickness [m]
        kappaLayers     (0.046);            // thermal cond. of ins. [W/mK]
    }
    front
    {       type            wedge;
    }
    back
    {       type            wedge;
    }
}
// ************************************************************************* //
```

E.2. constant folder

E.2.1. blockMeshDict (pipe)

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

convertToMeters 1;

// 3880 cells

vertices
(
    (0.02141 0.0009 0)
    (0.03016 0.001267818776 0)
    (0.03016 0.001267818776 0.226)
    (0.02141 0.0009 0.226)
    (0.02141 -0.0009 0)
    (0.03016 -0.001267818776 0)
    (0.03016 -0.001267818776 0.226)
    (0.02141 -0.0009 0.226)
    (0.03016 0.001267818776 0.2275)
    (0.02141 0.0009 0.2275)
    (0.03016 -0.001267818776 0.2275)
    (0.02141 -0.0009 0.2275)
    (0.03016 0.001267818776 0.2725)
    (0.02141 0.0009 0.2725)
    (0.03016 -0.001267818776 0.2725)
    (0.02141 -0.0009 0.2725)
    (0.01991 0.000836945353 0.2275)
    (0.01991 0.000836945353 0.2725)
    (0.01991 -0.000836945353 0.2275)
    (0.01991 -0.000836945353 0.2725)
    (0.00454 0.000190845399 0.2275)
    (0.00454 0.000190845399 0.2725)
    (0.00454 -0.000190845399 0.2275)
    (0.00454 -0.000190845399 0.2725)
    (0.03016 0.001267818776 0.274)
    (0.02141 0.0009 0.274)
    (0.03016 -0.001267818776 0.274)
    (0.02141 -0.0009 0.274)
    (0.03016 0.001267818776 0.5)
    (0.02141 0.0009 0.5)
    (0.03016 -0.001267818776 0.5)
    (0.02141 -0.0009 0.5)
```

```
    (0.03016 0.001267818776 1)
    (0.02141 0.0009 1)
    (0.03016 -0.001267818776 1)
    (0.02141 -0.0009 1)
    (0.03016 0.001267818776 2.5)
    (0.02141 0.0009 2.5)
    (0.03016 -0.001267818776 2.5)
    (0.02141 -0.0009 2.5)

);

blocks
(
    hex (0 1 2 3 4 5 6 7) pipe (8 20 1) simpleGrading (1 1 1)
    hex (3 2 8 9 7 6 10 11) pipe (8 5 1) simpleGrading (1 0.1 1)
    hex (9 8 12 13 11 10 14 15) pipe (8 30 1) simpleGrading (1 1 1)
    hex (16 9 13 17 18 11 15 19) pipe (5 30 1) simpleGrading (0.1 1 1)
    hex (20 16 17 21 22 18 19 23) pipe (15 30 1) simpleGrading (1 1 1)
    hex (13 12 24 25 15 14 26 27) pipe (8 5 1) simpleGrading (1 10 1)
    hex (25 24 28 29 27 26 30 31) pipe (8 150 1) simpleGrading (1 4 1)
    hex (29 28 32 33 31 30 34 35) pipe (8 100 1) simpleGrading (1 3 1)
    hex (33 32 36 37 35 34 38 39) pipe (8 100 1) simpleGrading (1 2.95 1)

);
edges
(
);
boundary
(
    inlet_pipe
    {
        type patch;
        faces
        (
            (0 4 5 1)
        );
    }
    outlet_pipe
    {
        type patch;
        faces
        (
            (37 39 38 36)
        );
    }
    inner_wall
    {
        type patch;
        faces
        (
            (0 3 7 4)
            (3 9 11 7)
            (9 16 18 11)
            (16 20 22 18)
            (20 21 23 22)
            (21 17 19 23)
            (17 13 15 19)
            (13 25 27 15)
            (25 29 31 27)
```

```
            (29 33 35 31)
            (33 37 39 35)
        );
     }
    outer_wall
    {
        type patch;
        faces
        (
            (1 2 6 5)
            (2 8 10 6)
            (8 12 14 10)
            (12 24 26 14)
            (24 28 30 26)
            (28 32 34 30)
            (32 36 38 34)
        );
     }
    front
    {
        type wedge;
        faces
        (
            (0 3 2 1)
            (3 9 8 2)
            (9 13 12 8)
            (16 17 13 9)
            (20 21 17 16)
            (13 25 24 12)
            (25 29 28 24)
            (29 33 32 28)
            (33 37 36 32)
        );
    }
    back
    {
        type wedge;
        faces
        (

            (4 7 6 5)
            (7 11 10 6)
            (11 15 14 10)
            (18 19 15 11)
            (22 23 19 18)
            (15 27 26 14)
            (27 31 30 26)
            (31 35 34 30)
            (35 39 38 34)
        );
    }
);

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

E.2.2. thermophysicalProperties (pipe) Case B

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{   version     2.0;
    format      ascii;
    class       dictionary;
    object      thermophysicalProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
thermoType
{
    type            heSolidThermo;
    mixture         pureMixture;
    transport       constIso;
    thermo          hConst;
    equationOfState rhoConst;
    specie          specie;
    energy          sensibleEnthalpy;
}
mixture
{
    specie
    {
        nMoles      1;
        molWeight   55;    //  22% Cr
    }

    transport
    {
        kappa   15;    // 22Cr
    }

    thermodynamics
    {
        Hf      0;
        Cp      485;   //According to Akers report 22Cr
    }

    equationOfState
    {
        rho     7800;  //According to Aker, 22Cr
    }
}
// ************************************************************************* //
```

E.2.3. transportProperties (pipe) Case B

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

DT              DT [ 0 2 -1 0 0 0 0 ] 3.965e-06;  //Calculated

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

E.3. system folder

E.3.1. controlDict

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

application     chtMultiRegionFoam;

startFrom       latestTime;

startTime       0;

stopAt          endTime;

endTime         24;

deltaT          0.1;

writeControl    adjustableRunTime;

writeInterval   1;

purgeWrite      0;

writeFormat     ascii;

writePrecision  8;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

maxCo           0.8;

maxDi           0.05;

adjustTimeStep  no;

functions
```

```
{
    probes
    {
        // Where to load it from
        functionObjectLibs ( "libsampling.so" );
        type            probes;
        enabled         true;
        region          pipe;

        // Name of the directory for probe data
        name            probes;

        // Writecontrols
        outputControl   timeStep;
        outputInterval  1;

        // Fields to be probed
        fields
        (
            T
        );

        probeLocations
        (
// At inner wall
       (0.02141 0 0.1)
       (0.02141 0 0.2725)
       (0.02141 0 0.3225)
       (0.02141 0 0.3725)
       (0.02141 0 0.50)
       (0.02141 0 1)
       (0.02141 0 1.5)
       (0.02141 0 2)
       (0.02141 0 2.5)
// Middle of wall
       (0.025785 0 0.1)
       (0.025785 0 0.2725)
       (0.025785 0 0.3225)
       (0.025785 0 0.3725)
       (0.025785 0 0.50)
       (0.025785 0 1)
       (0.025785 0 1.5)
       (0.025785 0 2)
       (0.025785 0 2.5)
// At outer wall
       (0.03016 0 0.1)
       (0.03016 0 0.2725)
       (0.03016 0 0.3225)
       (0.03016 0 0.3725)
       (0.03016 0 0.50)
       (0.03016 0 1)
       (0.03016 0 1.5)
       (0.03016 0 2)
       (0.03016 0 2.5)
        );
}

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

E.3.2. fvSchemes (pipe)

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

ddtSchemes
{
    default         Euler;
}

gradSchemes
{
    default         Gauss linear;
}

divSchemes
{
    default         none;
}

laplacianSchemes
{
    default             none;
    laplacian(alpha,h)  Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
}

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

E.3.3. fvSchemes (case 11, fluid)

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

ddtSchemes
{
    default Euler;
}
gradSchemes
{
    default         Gauss linear;
}
divSchemes
{
    default         none;

    div(phi,U)      Gauss upwind;
    div(phi,K)      Gauss linear;
    div(phi,h)      Gauss upwind;
    div(phi,k)      Gauss upwind;
    div((muEff*dev2(T(grad(U))))) Gauss linear;
    div(phi,omega) Gauss upwind;
}
laplacianSchemes
{
    default         Gauss linear corrected;
}
interpolationSchemes
{
    default         linear;
}
snGradSchemes
{
    default         corrected;
}
fluxRequired
{
    default         no;
    p_rgh;
}

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

E.3.4. fvSolution (pipe)

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

solvers
{
    h
    {
        solver           PCG;
        preconditioner   DIC;
        tolerance        1e-06;
        relTol           0.1;
    }

    hFinal
    {
        $h;
        tolerance        1e-06;
        relTol           0;
    }
}

PIMPLE
{
    nNonOrthogonalCorrectors 0;
}

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

E.3.5. fvSolution (case 11, fluid)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
solvers
{
    "(rho|rhoFinal)"
    {
        solver          PCG
        preconditioner  DIC;
        tolerance       1e-7;
        relTol          0;
    }

    p_rgh
    {
        solver           GAMG;
        tolerance        1e-7;
        relTol           0.01;

        smoother         GaussSeidel;

        cacheAgglomeration true;
        nCellsInCoarsestLevel 10;
        agglomerator     faceAreaPair;
        mergeLevels      1;
    }

    p_rghFinal
    {
        $p_rgh;
        tolerance        1e-7;
        relTol           0;
    }

    "(U|h|k|epsilon|R|omega)"
    {
        solver           PBiCG;
        preconditioner   DILU;
        tolerance        1e-7;
        relTol           0.1;
    }

    "(U|h|k|epsilon|R|omega)Final"
    {
        $U;
```

```
        tolerance        1e-07;
        relTol           0;
    }
}
PIMPLE
{
    momentumPredictor   on;
    nCorrectors         2;
    nNonOrthogonalCorrectors 10;
}
relaxationFactors
{
    fields
    {
    }
    equations
    {
        "h.*"           1;
        "U.*"           1;
    }
}
// ************************************************************************* //
```

E.3.6. setFieldsDict (case 11)

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.0                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      setFieldsDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
regions
(
   cylinderToCell
    {
        p1 ( 0 0 0 );
        p2 ( 0.0 0.0 0.2725 );
        radius 0.02141;

   fieldValues
        (
                 volScalarFieldValue p 23811375
                 volScalarFieldValue p_rgh 23811375
        ) ;
    }
);

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

Appendix F Multi-region STL

This appendix contains a summary of how to create multi-region STL files of high quality and how to create multi-region meshes with snappyHexMesh. How to set up a simulation with chtMultiRegionFoam is also presented.

F.1. Geometry

The pipe and the fluid were created as two separate files in Inventor, shown in Figure 24. The files were then exported as a high-quality STL files. The surface quality of the STL files is very important, since it greatly affects the quality of the mesh.

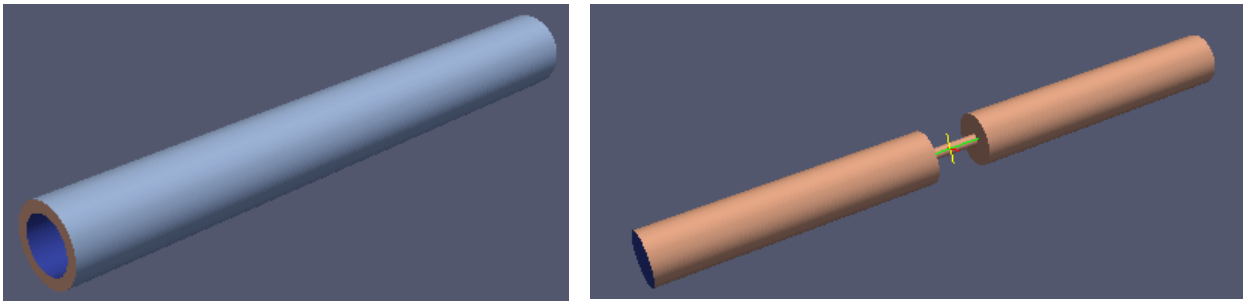

Figure 24 Pipe and fluid STL files

To enable specification of boundary conditions, patches must be added to the STL files. Inventor does not support this option, and this was therefore done in an open-source program called Salomé. A detailed description of this process is available in Appendix F.2. Figure 25 shows the STL mesh.

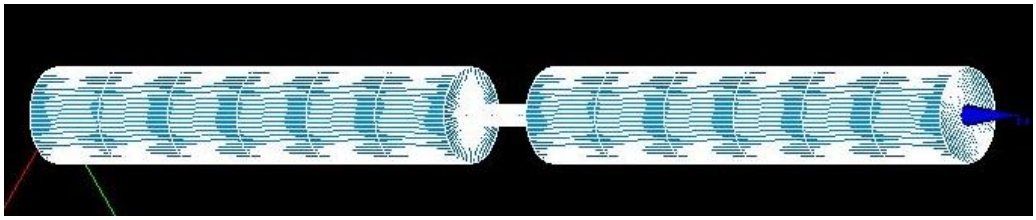

Figure 25 STL file in Salomé mesh module

F.1.1. snappyHexMesh

The finished STL files are used to generate the mesh. SnappyHexMesh starts with a base mesh, snaps it to the surface of the STL, and refines it around the surface according to the parameters given in snappyHexMeshDict. The base mesh is defined as “level 0”. Further refinement is carried out by splitting of the original cells, as shown in Figure 26.

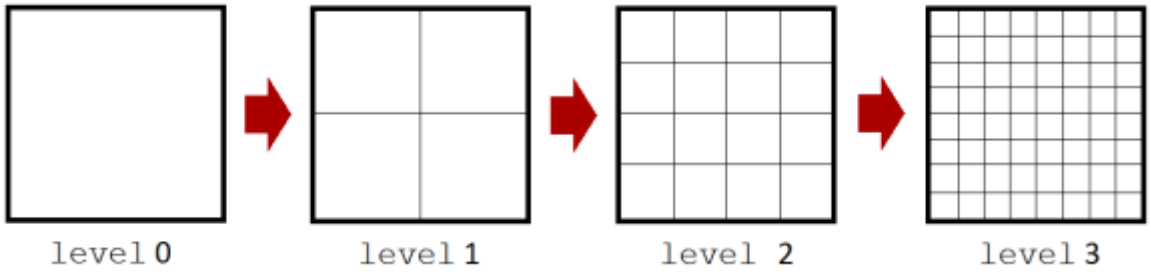

Figure 26 Refinement levels [67]

The symmetry of the pipe was exploited to reduce the number of cells. Only a quarter of the pipe was meshed, as shown in Figure 27.

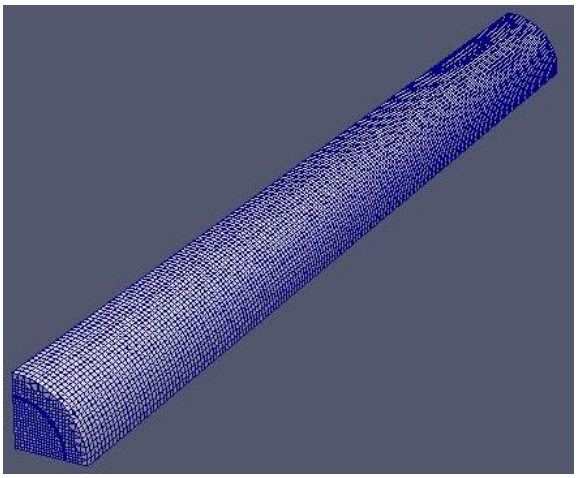
Figure 27 Only 25% of the pipe circumference is meshed

F.2. How to create multi-region STL files

1. Generate geometry in CAD program, e.g. Inventor
2. Export as STL file with high quality. Make sure scaling is in meters
3. Import STL file in Mesh module in Salome
4. Create groups by selecting, cutting and using union commands
5. Export each part as STL file.

Step 2-5 could be replaced with step 6-12, but this results in lower quality STL files, and larger file sizes. The advantage is that it simplifies the creation of groups in Salomé.

6. Export geometry as STEP (.stp) file.
7. Import .stp file into Salome
8. Create groups on faces, as shown e.g. in this Youtube video:
   https://www.youtube.com/watch?v=1zQbU-E4k1U
   The groups will become the boundary patches used in the simulation.

9. Mesh the geometry with triangular mesh. Use Netgen 1D-2D and very fine quality. Adjust mesh parameters max and min size to ensure a good mesh.
10. Create groups from geometry. This transfers the groups from the geometry to the mesh.
11. Export each group as a separate STL file. Mark the group in the mesh, right click and select Export->STL.
12. Check that each of the STL files is properly scaled, numbers should be in meters. If not, scale with: *surfaceTransformPoints –scale '(0.001 0.001 0.001)' inlet.stl inlet_m.stl*

13. Open each STL file in e.g. gedit and rename the patch in the first and last line of the file. First line should be*: solid patchname*, and the last line *endsolid patchname*. Example:

    solid inlet

    lots of numbers

    endsolid inlet

    These patch names will later be used to apply boundary conditions in the simulation
14. All groups that belong to the same part must be merged into one single STL file. This can be done with: *cat inlet.stl outlet.stl > merged.stl*

    Also see this video: https://www.youtube.com/watch?v=ObsFQUiVi1U
15. Check that each STL file is watertight. This can be done in for example Meshlab. Open Meshlab->Import Mesh. Select the merged STL file. Menu->Render->non-manifold edges. If this number is zero, the STL file is watertight. If it's not, consider using a finer mesh in Salome to get it watertight.
16. To check that you successfully have created a multiregional STL file, open ParaView and open the merged STL file. If it has multiple regions, each region is colored. If not, the entire model is gray.

F.3. Multi-region snappyHexMesh and chtMultiRegionFoam

1. Copy the tutorialcase snappyMultiRegionHeater
2. Copy your STL files into constant/triSurface
3. Define solid and fluid regions in constant/regionProperties
4. Edit blockMesh to make sure the box fits the geometry. Select appropriate number of cells
5. Edit snappyHexMeshDict and surfaceFeatureExtractDict
6. Rename 0 folder to 0.org. Remove 0 folder

7. Run blockMesh
8. Run surfaceFeatureExtract (not needed when using implicit edge snapping)
9. Run snappyHexMesh -overwrite
10. We want to remove the mesh outside the geometry. This can be done in the following way:
    a. Create a file named batch.setSet. The file should contain:

```
cellSet isolation new zoneToCell pipe
cellSet isolation add zoneToCell fluid
cellSet isolation subset
```

    b. Run the file by with: *setSet -batch batch.setSet -constant*
    c. We only want the parts of the mesh we have isolated, run: *subsetMesh -overwrite isolation*
11. Patches need to be redefined from STL to mesh. Run:
    surfaceToPatch constant/triSurface/fluid.stl
    surfaceToPatch constant/triSurface/pipe.stl
    This should be done for all STL files in triSurface folder
12. Copy the final mesh from the folder with highest time number for example 0.002, into constant/polyMesh folder:
    mv 0.002/polyMesh/ constant/polyMesh/*
13. Remove time folders including 0
14. Split the mesh into the different regions by running:
    splitMeshRegions –cellZones –overwrite
    Points 10-13 are described here:
    http://www.cfd-online.com/Forums/OpenFOAM-meshing-snappyhexmesh/96545-background-mesh-snappy-multi-domain-cht.html#post341623
15. We need to add viscous layers in the fluid region. This is done by running the following commands:

cp -f constant/polyMesh/blockMeshDict constant

*rm -r constant/polyMesh/**

mv constant/fluid/polyMesh/ constant/polyMesh/*

sed -i -e 18c"castellatedMesh   false;" system/snappyHexMeshDict

sed -i -e 19c"snap   false;" system/snappyHexMeshDict

sed -i -e 20c"addLayers   true;" system/snappyHexMeshDict

snappyHexMesh -overwrite >> logFile

sed -i -e 18c"castellatedMesh   true;" system/snappyHexMeshDict

sed -i -e 19c"snap   true;" system/snappyHexMeshDict

sed -i -e 20c"addLayers   false;" system/snappyHexMeshDict

mv constant/polyMesh/ constant/fluid/polyMesh*

These commands moves the fluid mesh into the polyMesh folder, edits the snappyHexMeshDict to only add layers, and runs it.

16. To view the mesh type paraFoam –touchAll
    Type ParaView in the terminal and open the fluid and pipe mesh from inside ParaView. Important to do it this way, as just typing paraFoam will result in errors.

    The followings steps must be done prior to running the simulation.

17. Remove fluid fields from solid regions by running:
    for i in pipe
    do
      rm -f 0/$i/{mut,alphat,epsilon,k,U,p_rgh}*
    done
18. Edit the changeDictionaryDict for both fluid and pipe and add proper boundary conditions. changeDictionaryDict files are located under system/fluid and system/pipe
19. Apply boundary conditions given in changeDictionaryDict by typing:
    for i in fluid pipe
    do
     changeDictionary -region $i > log.changeDictionary.$i 2>&1
    done
20. Optimize bandwidth by running renumberMesh -overwrite
21. Run chtMultiregionFoam

Appendix G Instructions

This appendix contains instructions on how to run cases, how to create time-averaged temperature fields and how to extract the fluid surface temperature and use it as boundary condition on the inner pipe wall.

G.1. How to run cases

The case files can be found in Appendix H. Each folder contains some of the computed time steps including the solution for the latest time step. Due to limited storage space, it is not possible to include all time steps.

G.1.1. rhoCentralFoam cases

The cases can be run by following this procedure:

1. Run *blockMesh*

2. Set the upstream pressure by running *setFields*

3. Open controlDict, make sure maxCo is at 0,5.

4. Run rhoCentralFoam. After 1 ms of flowtime maxCo can be changed to 0,9 for ideal gas cases and 0,7 for Peng-Robinson cases. Be aware that each case takes 15-20 hours to run on a standard laptop.

   If you have the pyFoam package installed, residuals can be automatically plotted during the simulation by running *pyFoamPlotRunner.py rhoCentralFoam.* See [68] for details.

5. The temperature field at the fluid surface can be extracted by running *sample –latestTime*, or *sample –time 0.015*.

G.1.2. chtMultiRegionFoam pipe case

1. Run blockMesh

2. Open controlDict and change endTime to 24.

3. Run chtMultiRegionFoam

4. Copy the include folder from 0/pipe to 24/pipe by typing:

   cp -r 0/pipe/include 24/pipe

Open the T file located in 24/pipe. Scroll down to approx. line 3920. Remove the boundary condition for inner_wall, including all 450 temperature values. Copy in the following line:

#include "include/T_avg5barPeng13to15ms"

The temperature field on the inner wall is now read from the include folder.

5. Open controlDict, change endTime to 30.

6. Run chtMultiRegionFoam

7. Repeat the process described in step 4 for the 30 time folder. The new temperature field is included with *#include "include/T_avg6bar13to15ms"*

8. Open controlDict, change endTime to 45. Run chtMultiRegionFoam. Then, include the new temperature field as described above. Repeat this process for all temperature fields described in Table 4-1.

G.1.3. chtMultiRegionFoam case 11

1. Run blockMesh

2. Split the fluid and the solid region by running *splitMeshRegions –cellZones –overwrite*

3. Set the upstream pressure in the fluid region by running *setFields –region fluid*

4. Open controlDict, make sure maxCo is at 0,5.

5. Run chtMultiRegionFoam. After 1 ms of flowtime maxCo can be changed to 0,9.

G.2. How to create time-averaged temperature fields

1. Create a case folder containing the time steps you want to average over.

2. Open ParaView, select the temperature field in the "Volume fields" section and click Apply. Select the "Temporal Statistics" filter and click Apply after selecting which values to calculate (average and standard deviation).

3. Save the data by selecting File->Save Data. Use .csv format. Enter a name and press Save. Select "Cells" for the Field Association option in the dialogue box that appears.

4. Now, we will create a temperature field with the averaged values. Import the saved .csv file into Excel. Copy the values from the T_average column into the T_template at line 23, between the brackets. The T_template file can be found in the

PipeCases/chtMultiRegionCaseFiles folder located in Appendix H. Make sure that the temperatures are given with . as decimal separator. Rename the template file “T”.

5. Create a new time folder called “1” and copy “T” into this folder. The time-averaged field can now be visualized and post-processed in ParaView, like any other field.

G.3. How to create temperature boundary condition for pipe

1. Run the sample utility in your fluid case folder, using “sample –time 0.015”. Modify time to fit your case.

2. Locate the output file called “T” in the “postProcessing/surface/0.015/fluid_surface/scalarField” folder.

3. Copy the file “T_BC_template” from the FluidCases/rhoCentralFoamCaseFiles folder in Appendix H.

4. Copy the content of the output file from step 2, into the template file at line 14. Rename the file “T_BC1” and save.

5. Move to the boundary condition directory of your pipe case, for example “0/pipe”. Create a folder called “include”. Copy the T_BC1 file into this folder.

6. Open the T file in your boundary condition directory. Remove the entry for inner_wall and replace with *#include “include/T_BC1”*. The boundary condition is now read from the file in the include folder you created. Study the PipeCases in Appendix H for further details.

Appendix H Enclosed files

The case files can be downloaded from Dropbox:

https://www.dropbox.com/sh/eihl9sl024sk8nx/AABYIkYfuKTkkRHy1z76Y_B2a?dl=0