



POLITECNICO
MILANO 1863

Tesi di Laurea Magistrale

Numerical Simulation of Condensing CO₂ Supersonic Flows at High Pressure

Candidato:
Stefano ZANZI
Matricola 874776

Relatore:
Prof. Giacomo PERSICO

Corso di Laurea Magistrale in Ingegneria Meccanica
SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Anno Accademico 2017/2018

Abstract

Considering its flexibility and potential efficiency, the s-CO₂ cycle is a promising candidate for development and commercialisation in the field of power generation, and is extensively studied in literature. Nonetheless, components and especially compressor design, which may experience condensing high speed flows, remains problematic.

In this framework, the purpose of this work is the definition of a reliable setup for the simulation of high pressure, condensing, supersonic carbon dioxide flows; ANSYS CFX and ANSYS Fluent are both employed and compared. Practically, a test case - a series of supersonic nozzle blowdowns experimentally detailed by Lettieri [22] experiencing non-equilibrium condensation – is analysed.

For reference purposes, a quasi-1D solution implementing simple two-phase capacities is developed in MATLAB and compared to CFD results. Details concerning all aspects of the simulations (turbulence models, numerical schemes, mesh) are explored. Convergence could be achieved only with Eulerian-Eulerian condensation models, as already available in CFX and Fluent. The accurate Span-Wagner Equation of State is implemented via user-defined materials making use of Lookup Tables generated by MATLAB, in connection with REFPROP, for the definition of real gas properties (which included metastable states).

Thorough analysis of the results highlights the superiority of ANSYS Fluent, due to its stronger numerical and multiphase capacities. User interaction is also more flexible, at the cost of complexity. Due to simplified condensation models, accurate replication of experimental results could not be achieved, but general trends are consistent.

Simulations failed at near-critical inlet total pressure conditions, due to the extreme thermodynamic gradients and proximity of the spinodal line, as well as to the significant amount of liquid generated in the corresponding experiments. Further work in terms of thermodynamic and numerical options is required for the setup extension to these regions.

Therefore, the setup discussed in this work is ultimately thought to be applicable to the simulation of more complex geometries, as long as the two-phase regions of the flow are small compared to the overall domain.

Keywords: Carbon Dioxide, CFD, Lookup Tables, Metastable Properties, Multiphase Flows, Real Gas.

Sommario

Data la sua flessibilità e potenziale efficienza, il ciclo a CO₂ supercritica è considerato promettente per ulteriori sviluppi nel campo della generazione di potenza, ed è argomento di ampi studi in letteratura. Ciò nonostante, la progettazione delle sue componenti ed in primo luogo del compressore, in cui sussiste il rischio di condensazione di flussi ad alta velocità, è ancora problematica.

In questo contesto, lo scopo di questo lavoro è la definizione di un setup affidabile per la simulazione di flussi supersonici di anidride carbonica in condensazione. ANSYS CFX e ANSYS Fluent sono entrambi utilizzati e messi a confronto. Lo studio si estrinseca nell'analisi di un caso studio - una serie di espansioni condensanti in ugelli di De Laval studiata da Lettieri [22].

Per confronto, una soluzione monodimensionale con semplici capacità bifase è sviluppata in MATLAB e comparata ai risultati CFD. Tutti gli aspetti riguardanti la simulazione (modelli di turbolenza, schemi numerici, mesh) sono studiati in dettaglio. Per ottenere convergenza risulta necessario impiegare modelli di condensazione Euleriani-Euleriani già presenti in CFX e Fluent. L'Equazione di Stato Span-Wagner è implementata attraverso materiali *user-defined* facenti riferimento a tabelle di ricerca generate in MATLAB, in connessione con REFPROP, per la definizione delle proprietà termodinamiche del gas reale (comprendenti gli stati metastabili).

Un'analisi dettagliata dei risultati denota la netta superiorità di ANSYS Fluent, a causa delle migliori capacità numeriche e multifase, nonché della flessibilità in termini di implementazione da parte dell'utente, a discapito di una maggiore complessità.

Le simulazioni sono risultate divergenti in condizioni di pressione totale all'imbocco prossime al punto critico, a causa degli elevati gradienti termodinamici e della vicinanza della linea spinodale, nonché dell'alta quantità di liquido presente nei casi sperimentali corrispondenti. Ulteriori sviluppi sono necessari per l'estensione all'analisi di queste regioni.

In conclusione, si ritiene che il setup qui presentato sia in ultima analisi in grado di essere applicato alla simulazione di geometrie più complicate, purché le regioni bifase siano sufficientemente piccole.

Parole Chiave: Anidride Carbonica, CFD, Flussi Multifase, Gas Reale, Proprietà Metastabili, Tabelle di Ricerca.

Contents

Abstract	iii
List of Figures	xi
List of Tables	xv
1 Introduction and General Context	1
1.1 General premise	1
1.2 The s-CO ₂ cycle	3
1.3 Scope of the work	4
2 Condensation Phenomenon and Models	9
2.1 Non-equilibrium condensation of carbon dioxide	9
2.1.1 Experimental reference	12
2.2 Numerical models for condensation simulation	14
2.2.1 The Homogeneous Equilibrium Model (HEM)	15
Numerical implementation	17
2.2.2 The Lee Model	17
Numerical implementation	19
3 Thermodynamic Properties of CO₂ and CFD Input	21
3.1 Equations of State for carbon dioxide	21
3.1.1 The Peng-Robinson Model	21
3.1.2 The Lee-Kesler Model	22
3.1.3 The Span-Wagner Model	23
3.2 Implementation strategy	24
3.2.1 ANSYS CFX [®]	26

3.2.2	ANSYS Fluent®	29
	The User Defined Real Gas Model (UDRGM)	29
	Second phase definition	31
4	Quasi-1D Solution	33
4.1	Analytical framework	33
4.2	Numerical implementation	35
4.2.1	Condensation model	36
4.3	Algorithm validation	37
4.4	Algorithm limitations	39
5	CFD Setup	43
5.1	Geometry model	43
5.2	Boundary conditions	44
5.3	Meshing process	45
5.3.1	Mesh independence analysis	47
	Y+ analysis	50
5.4	Numerical methods	50
5.4.1	ANSYS CFX®	51
5.4.2	ANSYS Fluent®	52
5.4.3	Turbulence Model	53
5.5	LuT dimension analysis	54
5.6	Lee Model coefficient analysis	56
6	Results Analysis	59
6.1	Cross-section pressure variation	59
6.2	Single-phase solution comparison for Case 1-1	59
6.2.1	Single-phase behaviour for ANSYS CFX	64
6.3	Two-phase solutions comparison for Case 1-1	66
6.3.1	Thermodynamic behaviour	66
6.3.2	Condensation locus assessment	74

6.4	Solutions for Case 2	78
6.5	Solutions for higher total pressure cases	81
6.6	Solutions accuracy	84
7	Conclusions	87
A	Geometry	91
B	Wilson Line	93
C	Fluent Lookup Tables Generator	95
D	User Defined Functions	105
	Bibliography	115

List of Figures

1.1	Global electricity consumption (International Energy Agency data)	2
1.2	Closed Joule-Brayton cycle	3
1.3	Compressor realised for SANDIA National Laboratories, alongside the blank and one of the tools used for machining (from [35])	5
1.4	Two phase regions (in red) in the SANDIA compressor (from [30])	5
1.5	Condensation in high speed flows (from [21])	6
2.1	Non equilibrium condensation (from [21])	10
2.2	Enthalpy maps, with (below) and without (above) metastable vapour properties	11
2.3	T-s diagram showing transition lines	12
2.4	P-T diagram showing transition lines	13
3.1	Sample surface representing enthalpy as function of temperature and pressure	25
3.2	CFX simulation scheme	27
3.3	Structure of a .rgp file	28
3.4	Fluent solution scheme	32
4.1	Single-phase pressure comparison with CFD codes	37
4.2	Two-phase pressure comparison with CFX, which employs the HEM	38
4.3	Two-phase liquid mass fraction comparison with CFX, which employs the HEM	39
4.4	Residuals for Case 1-1 (normalised with mass flow value)	40
4.5	Sound speed for single-phase and two-phase solutions	41
4.6	Residuals (normalised with mass flow value) for single-phase solution for Case 3-1	42
4.7	Residuals (normalised with mass flow value) for two-phase solution for Case 3-1	42

5.1	Geometry model	43
5.2	Boundary conditions	44
5.3	Pressure in case 1-1 on the symmetry line for different inlet total pressure	46
5.4	Total temperature comparison with the reference	46
5.5	Pressure in Case 1-1 on the symmetry line for different 2D meshes (Fluent)	48
5.6	Difference between the pressures for Case 1-1 on different 2D meshes (Fluent)	48
5.7	Pressure in Case 1-1 on the symmetry line for different 3D meshes (CFX) .	49
5.8	Difference between the pressures for Case 1-1 on different 3D meshes (CFX)	49
5.9	Y+ value	51
5.10	Pressure in Case 1-1 on the symmetry line for different Lookup Table sizes	55
5.11	Difference between the pressures for Case 1-1 with different Lookup Table sizes	55
5.12	Pressure in case 1-1 on symmetry line for different Lee Model coefficient values	57
5.13	Difference between the pressures for Case 1-1 at different values of the Lee model coefficient (blue and red lines overlap)	57
5.14	Liquid Volume Fraction in case 1-1 on symmetry line for different Lee Model coefficient values	58
6.1	Cross-section pressure for Case 1-1 in the throat	60
6.2	Single-phase pressure for Case 1-1	61
6.3	Single-phase entropy for Case 1-1	61
6.4	Single-phase temperature for Case 1-1	62
6.5	CFX and Fluent comparison of different properties (denoted by different legend entries), normalised with Fluent values	62
6.6	Single-phase sound speed for Case 1-1	63
6.7	Single-phase Mach number for Case 1-1	64
6.8	Single-phase pressures for .rgp tables transitioning at spinodal and Wilson	65
6.9	Single-phase entropies for .rgp tables transitioning at spinodal and Wilson	65
6.10	P-T behaviour for single-phase .rgp tables transitioning at spinodal and Wilson	66
6.11	Two-phase pressures	67

6.12 Two-phase temperatures	67
6.13 Two-phase solutions behaviour on T-P plane	68
6.14 CFX and Fluent comparison of different two-phase properties (denoted by different legend entries), normalised with Fluent values	69
6.15 Two-phase Mach number	70
6.16 Two-phase velocity	70
6.17 Two-phase sound speed (blue and yellow lines nearly perfectly overlap) .	71
6.18 Two-phase entropy	71
6.19 Pressure contours for CFX (above) and Fluent (below) for Case 1-1	72
6.20 Comparison between Fluent (above) and Hosangadi (below) [16] for Case 1-2	73
6.21 Difference between simulations and experimental values, normalised with Fluent values	74
6.22 Pressure comparison for mono- and two-phase simulation	75
6.23 Temperature comparison for mono- and two-phase simulation	75
6.24 Liquid mass fractions and condensation onset (dashed) for two-phase sim- ulations	76
6.25 T-P diagram showing thermodynamic location of condensation onset for all assessments	77
6.26 Liquid Mass Fraction contour for Fluent for Case 1-1, compared with the experimental reference (from [22])	78
6.27 Pressure for Case 2-1	79
6.28 Pressure for Case 2-2 with Fluent (above) and as obtained by Hosangadi [16] (below)	80
6.29 Liquid Mass Fraction contour for Fluent for Case 2-1, compared with the experimental reference (from [22])	81
6.30 Single-phase sound speed for Case 2-2	82
6.31 Single-phase pressure for Case 2-2	82
6.32 Single-phase and two-phase difference for Case 2-2 with experimental val- ues, normalised with experimental values	83
6.33 Differences between simulations and experiments, normalised with exper- imental values	84

List of Tables

1.1	Carbon dioxide properties	3
2.1	Ideal expansions	14
2.2	Experimental tank conditions and condensation location	14
5.1	Mesh data	47
5.2	Inflation feature data	47
5.3	Fluent numerical schemes	52
5.4	LuT details	54
6.1	Axial location of condensation	77
6.2	Axial location of condensation for all simulations	85
7.1	Recap of Fluent options	89

Chapter 1

Introduction and General Context

1.1 General premise

Due to the large population growth and overall economy development of the last century, which are forecast to continue well into the next decades, global energy consumption is destined to increase dramatically.

As a reference, it has been estimated [17] that, depending on different scenarios, energy consumption in 2040 may be as much as doubled compared to 1990.

This is a significant issue for our society, which, coupled with the always more stringent requirements for a sustainable use of our resources, will need to be tackled from multiple points of view.

A possible contribution to the solution is the development of new and more efficient energy conversion devices, capable of extracting as much energy as possible from a primary source (a fossil fuel, the sun, the wind, a nuclear reaction) and turning it into electricity or heat as required by a end-user. Historically, this is among the oldest problems of engineering relevance.

Two solutions - Joule-Brayton and Rankine cycles - have been traditionally employed. Both are conceptually composed of four thermodynamic transformations; compression, heating, expansion and cooling, bringing back the fluid to its original state. However, the latter also involves phase transition (generally, water-steam) during both heating and cooling, which is especially advantageous because of the isothermal heat addition and rejection - the most efficient thermodynamic solution.

Not only; compression of a liquid substance, in a pump, requires much less work than that of a highly compressible fluid, and is order of magnitudes lower than the one generated by an expansion. Components size is extremely large, owing to the surfaces needed for heat exchange to take place. Conversely, high efficiency in Rankine cycles requires an extremely complex layout (regenerations and reheatings), since economical feasibility limits the temperature range.

The traditional Joule-Brayton cycle, instead, can take advantage of another feature. If air is chosen as working fluid, an open layout can be sought. This means that no cooling takes physically place inside the devices making up the cycle, but the fluid is instead continuously discarded at the turbine outlet and absorbed at the compressor inlet. Moreover, air can be burned in order to increase its temperature, which greatly reduces size compared to the use of a heat exchanger; therefore, power density becomes extremely

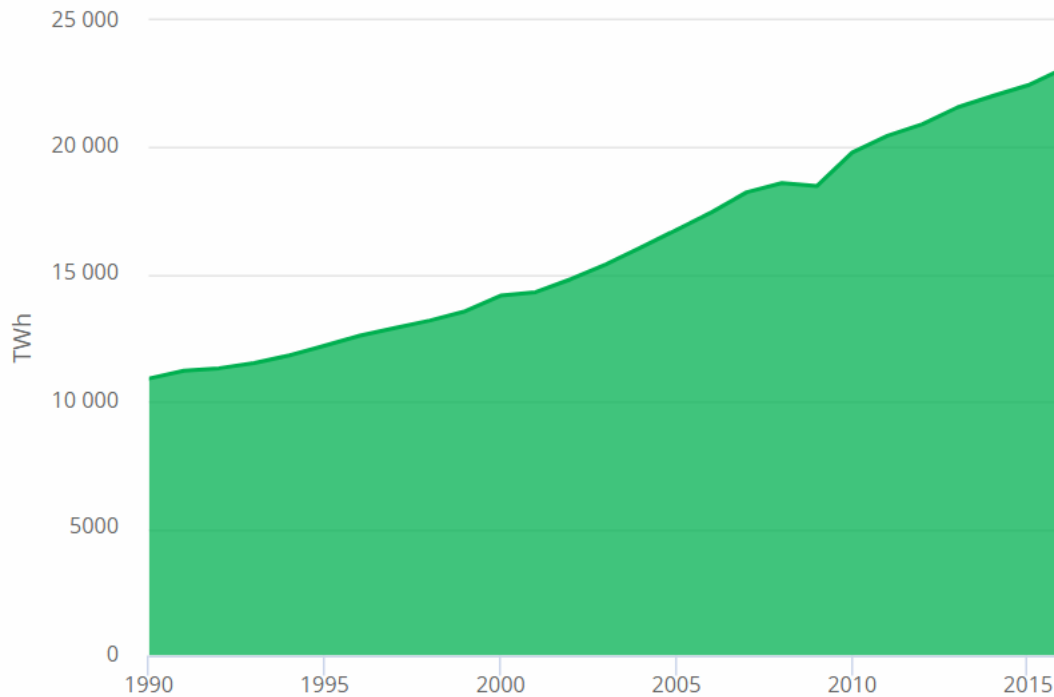


FIGURE 1.1: Global electricity consumption (International Energy Agency data)

high, which is why it is the reference solution for jet aero-engines. Contrarily, components design is complex and compressor work is high, which can be disadvantageous and in general limits efficiency.

The standard solution for modern power generation involves the combination of these two cycles; that is, hot exhaust gases at the outlet of a Joule-Brayton turbine are used as primary heat source for a Rankine cycle. This type of solutions has proven to be extremely efficient (up to 60% in terms of useful effect - heat input ratio). That notwithstanding, combined cycles are complex and cumbersome, which makes them unsuitable for a series of situations.

In the gaps left by these technologies, several new solutions have emerged in the past few decades.

Among them, a promising candidate for further development and potential commercialisation is the so-called supercritical carbon dioxide cycle. This is, in short, a Joule-Brayton cycle that makes use of carbon dioxide in (mostly) supercritical state as the working fluid. Obviously, it makes use of a closed layout (shown, in its simplest form, in Figure 1.2), which, albeit conceptually analogous, brings about several practical modifications compared to the open version.

Conceptually, this cycle could virtually make use of any fluid; the specific choice of carbon dioxide is linked mainly to its critical point properties, reported in Table 1.1. Namely, temperature is quite near to atmospheric conditions (at least in mild climates), whereas pressure is acceptably low (though still high in terms of absolute value) and can be attained and maintained with current technology in industrial equipment. As a reference, it is around one third that of water. Therefore, reaching this thermodynamic condition is feasible. Other important features come in form of non-toxicity, chemical inertness (which lessens security issues) and availability at a low price.

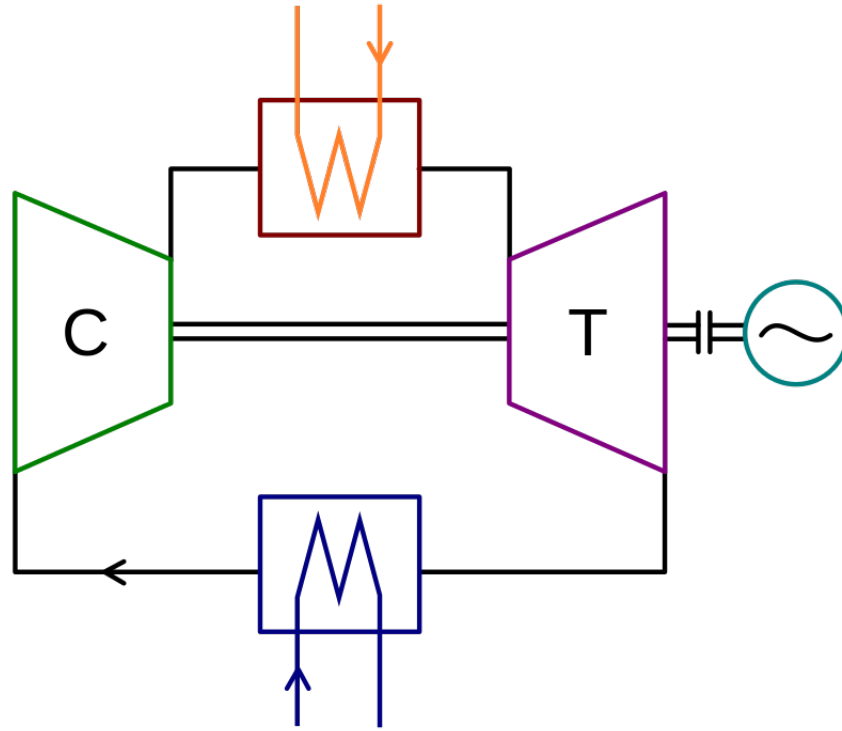


FIGURE 1.2: Closed Joule-Brayton cycle

Critical Temperature	304.13 <i>K</i>
Critical Pressure	73.773 <i>bar</i>
Critical Density	467.6 <i>kg/m</i> ³
Triple Point Temperature	216.59 <i>K</i>
Triple Point Pressure	5.1796 <i>bar</i>
Molar Mass	44.01 <i>kg/kmol</i>

TABLE 1.1: Carbon dioxide properties

1.2 The s-CO₂ cycle

The cycle concept dates back to the late '60s [3] [13], when its main characteristics were first proposed and highlighted. However, technological issues hindered its development; interest in the field was revived in 2004 [12]. Since then, it has been debated and studied quite extensively in literature, mainly in terms of components design (machines and heat exchangers) and new layouts (a very detailed review has been carried out in [11]).

Its advantages come from the very nature of the critical point and its thermodynamic properties. Liquid and gas properties asymptotically coincide; the fluid is in an intermediate state. Therefore, density is rather high. This implies that a compression starting from this thermodynamic area makes use of a near-liquid fluid, and requires less work with respect to the standard Joule-Brayton cycle. Size is consequently small, owing both to the low fluid specific volume and the low pressure ratio (due to the high absolute value for the compressor inlet pressure). Conversely, after compression and heating the fluid lies far from the transition curve, into a near-ideal state; this brings about increased expansion efficiency.

Overall, provided the layout is suitably arranged - a regenerator at the turbine outlet is required, and extremely complex solutions have been devised - this cycle has the thermodynamic premise for high thermodynamic efficiencies at a moderate size and in a wide variety of temperature ranges. Both features make it extremely flexible, which led to proposals for its application to a wide spectrum of situations (ranging from Concentrated Solar Power to nuclear reactors [1]). Power density is quite high, making it also suitable for modular and compact applications; it has even been proposed as a potential solution for space systems [9].

However, technological challenges are still significant [10]. Specifically, as CO_2 cannot be heated through combustion (differently from air), complex, high-efficiency and compact (if the size advantage is to be exploited) heat exchangers are required; the solution comes only with the modern and technologically advanced Printed Circuit Heat Exchangers (PCHEs). The closed cycle configuration and unsafe thermodynamic conditions (high pressure) imply tight sealings. Design for both machines, turbine and compressor, is complex, as they make use of a fluid in highly specific conditions, at low volumetric flow rates. Due to this, rotation speeds can be extremely high; as a reference, the shaft hosting compressor and turbine for the experimental loop realised at the SANDIA National Laboratories, based in Albuquerque, New Mexico (US) [35] is designed to rotate at 75.000 *RPM*. This causes high mechanical stresses and requires, for instance, specific gas-foil bearings.

The compressor design, due to its small size and complex thermodynamic framework, has proven to be especially critical for the technology development of the technology. No full scale machine exists yet, only prototypes - most notably, the one realised for the SANDIA loop, shown in Figure 1.3, which has already been extensively studied and simulated in literature [2] [30]. No general design strategy or framework exist yet. More work is therefore needed on this matter, if the cycle is to fully realise its potential.

As an aside, compression of high pressure carbon dioxide is of interest in a variety of fields (for instance, Carbon Capture and Storage), which further extends the research interest highlighted above.

1.3 Scope of the work

A less intuitive, yet remarkable, issue regarding the s-CO_2 compressor is linked to condensation and its assessment. In order to achieve high density at the compressor inlet, the thermodynamic condition is chosen to be as near as possible to the critical point and consequently the saturation line. However, local acceleration at the leading edge of the compressor blades may bring the flow below saturation conditions, hence potentially causing condensation inside the machine itself (a dual phenomenon with respect to cavitation in pumps). In fact, this has already been verified operatively at the SANDIA test facility in off-design conditions [35], and confirmed by simulations of its compressor (as highlighted in Figure 1.4). It is intuitive that large quantities of condensate are either not desirable for machines designed to work with a single phase fluid, or at least require very specific design techniques. Due to the high flow speed, condensation will not happen exactly as the flow crosses the saturation line; it is therefore a non-equilibrium phenomenon (it will be discussed in more depth in Chapter 2). This further complicates the assessment procedure. Hence, accurate predictions of the flow thermodynamics and its condensation conditions are necessary.

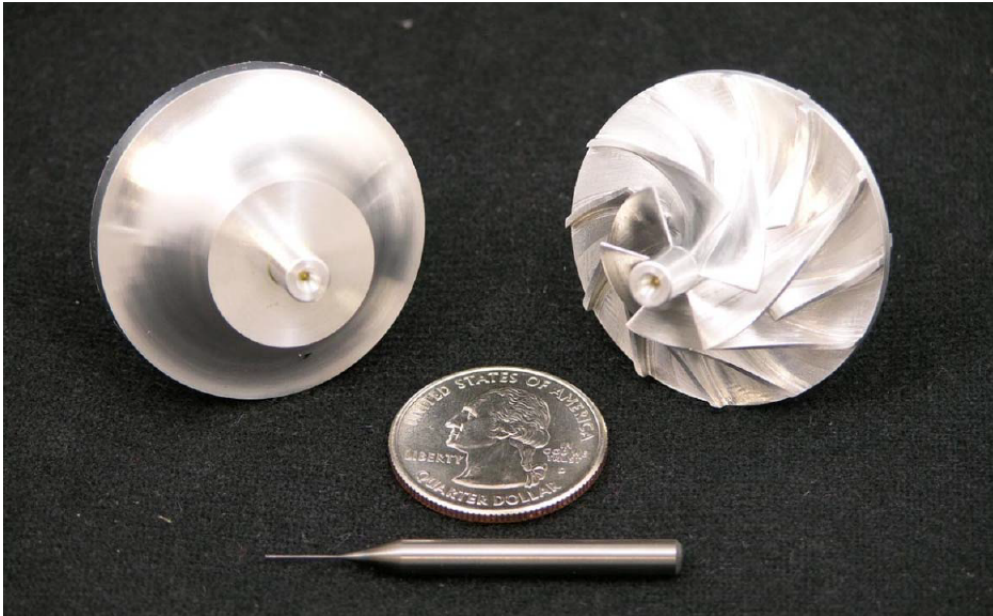


FIGURE 1.3: Compressor realised for SANDIA National Laboratories, alongside the blank and one of the tools used for machining (from [35])

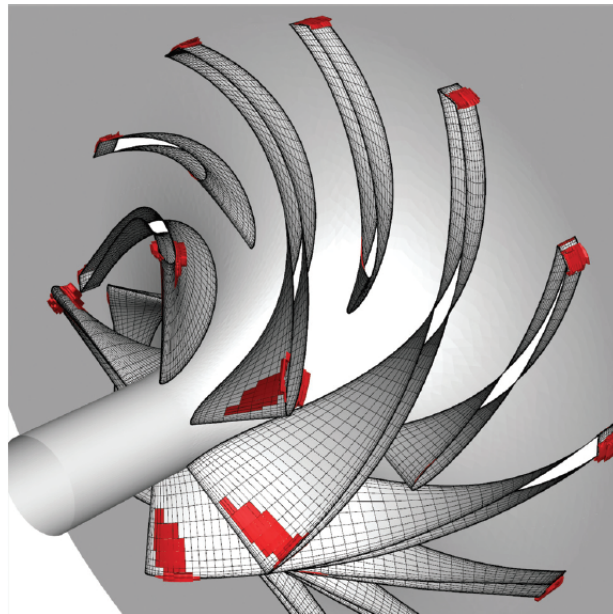


FIGURE 1.4: Two phase regions (in red) in the SANDIA compressor (from [30])

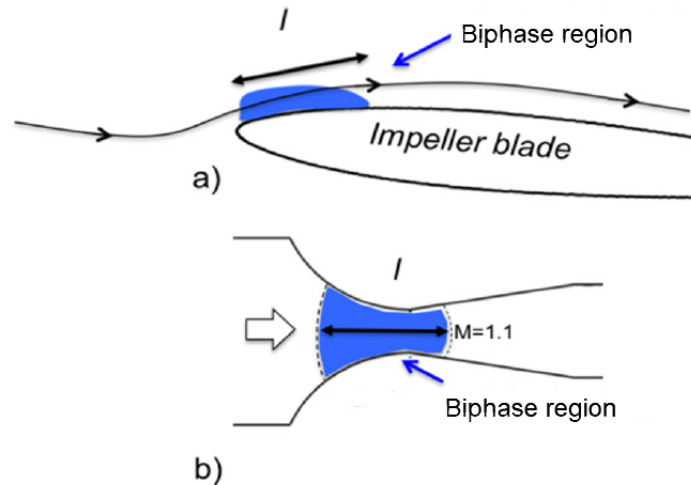


FIGURE 1.5: Condensation in high speed flows (from [21])

The thermodynamic limit defining the condensation onset (which is usually called Wilson line, and is a flow, rather than a fluid, property) has been experimentally determined for CO_2 by Bier [7] and, more recently, Lettieri [22], via supersonic nozzle blow-downs. As highlighted in Figure 1.5, this is analogous to the flow condensation taking place at a compressor leading edge.

The replication of this phenomenon in Computational Fluid Dynamics (CFD), and which setup (in terms of software choice, condensation model and fluid properties) can yield the most consistent and accurate predictions for it, is still an open issue and it is of great interest for further developments in the field.

As far as the present work is concerned, software choice was limited to well-known commercial options, for availability, ease of use and lack of specific results. Specifically, ANSYS Fluent[®] 19.0 and ANSYS CFX[®] 19.0 were employed, and a comparison of their performances is to be carried out. Also, a simplified quasi-1D solution developed by means of a MATLAB[®] code will be also analysed in order to provide further insight into the simulations behaviour and accuracy.

Historically, the foremost interest in non-equilibrium condensation of high-speed flows originated from low pressure steam turbines, where the phenomenon has been extensively studied. This led to the development and implementation of most numerical models available in commercial CFD codes; however, the validity of such models for CO_2 is still to be determined.

Therefore, the aim of this work lies in the definition of a reliable thermodynamic setup for high pressure, condensing carbon dioxide. This will be analysed through a comparison with an experimental test case - a nozzle blowdown - available in literature (the aforementioned work by Lettieri [22]).

Two practical aspects will be analysed. In general, accurate simulation of two phase states can be of interest if condensation is expected to involve large areas of the flow domain. This may not be the case for s- CO_2 compressors; in this case, a more limited assessment of condensation - i.e. a quantitative determination of whether condensation may happen, and a qualitative information regarding its eventual size and influence - may be enough. This does not necessarily require condensation models (or at least can take advantage of simplified versions), and it can be determined even in post-processing

(for instance, by determining whether local temperature and pressure are below the Wilson line).

Chapter 2

Condensation Phenomenon and Models

The purpose of this chapter is the illustration of non-equilibrium condensation, the physical phenomenon that is of interest for the present work. Consequently, a general overview of the models applied for its simulation in the CFD software with details referring to their technical implementations will be discussed.

Both CFX and Fluent offer several possibilities regarding the general topic of multiphase simulations that will not be mentioned here, as it was deemed that they would not be applicable.

2.1 Non-equilibrium condensation of carbon dioxide

Condensing fluids at high speeds, such as those attainable in a compressor or a nozzle, exhibit interesting properties. In fact, there is no immediate phase transition at the saturation line, owing to the fact that the flow is not locally and instantaneously in equilibrium conditions (because of its velocity). Hence, an excursion of some degree into the two-phase region (defined as *subcooling*, because condensation would cause a temperature increase) will happen. A fluid at those conditions is defined as *metastable*, and the determination of its thermodynamic properties is usually rather challenging.

Physically, this excursion into the two-phase region is limited in both time and space; eventually, when the fluid slows down, or after enough time has passed, it will spontaneously start to condensate, and revert to equilibrium conditions. Physically, temperature and pressure will rise because of droplets formation and growth. At the end, it will thermodynamically lie on the saturation line. The process is visualised in Figure 2.1, and necessarily brings about wetness losses, which are expressed as entropy increases [21]. It is a continuous transition and requires time, and therefore the flow (or at least the vapour fraction) is still in metastable conditions, even after condensation has begun.

There exist a lower thermodynamic limit, however, below which the fluid in vapour state simply cannot exist thermodynamically and which serves as a barrier for the vapour excursion into the two-phase dome. This is called the *spinodal line*, and it can be defined mathematically from the fluid equation of state [4]. Therefore, this line is a property of the fluid itself.

It is worth mentioning that the same exact principle can be applied to evaporating liquid flows (thus, a liquid spinodal which, in the $T - s$ plane, lies *above* the saturation line, rather than below, exists as well).

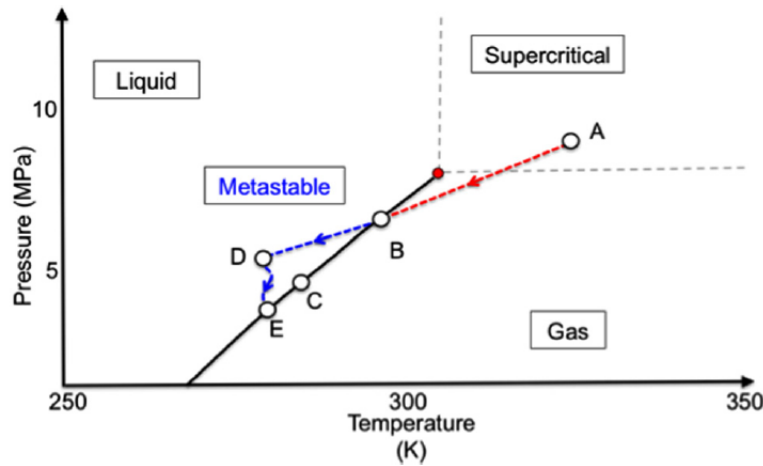


FIGURE 2.1: Non equilibrium condensation (from [21])

As far as the present work is concerned, the metastable thermodynamic properties and the spinodal line were directly obtained by REFFPROP, which operates an extrapolation of the Span-Wagner (discussed in more depth in Chapter 3) formulation into the liquid area. An example of this solution is shown in Figure 2.2, where the second map also includes metastable vapour properties (therefore showing a discontinuity, which represents the phase transition, at lower temperatures and higher pressures).

In practical applications the vapour spinodal line should never be reached due to the general instability and thermal fluctuations that the flow is experiencing during the expansion [4]. The actual condensation thermodynamic onset will therefore lie in an intermediate state between saturation and spinodal line; it must be found experimentally, and is called *Wilson line*. Intuitively, the excursion in the two-phase dome will be deeper the faster the flow; hence, the Wilson line depends on the flow conditions and namely its expansion rate as well, rather than being a simple fluid property.

The Wilson line adopted in the present work was determined by Lettieri [22] by means of five series of experiments conducted at increasing total pressure (approximately from 58 to 83 bar) and repeated twice, in a De Laval nozzle. More details are provided below.

This provided us with 8 experimental points - the two corresponding to the highest pressure test case were excluded (as in the reference), because they were lying below the spinodal line itself (as determined by REFFPROP; this sheds some doubts on its accuracy in the proximity of the critical point).

For purposes linked to its CFD implementation, there was the need to create a continuous Wilson line from triple to critical point. Therefore, the eight points (defined on the $T - s$ plane) were first joined by means of a linear interpolation (in the least squares sense), then this experimental line was extrapolated, by keeping the Wilson line's relative distance from saturation and spinodal lines constant compared to the highest pressure experimental point (for the extrapolation up to the critical point) or to the lowest pressure one (for the extrapolation down to the triple point). All these operations were carried out via a MATLAB script; the final result is reported in Appendix B. Although this process is completely arbitrary and its accuracy cannot be assessed (especially far from the experimental points), the replication of the original test cases should never bring the flow inside

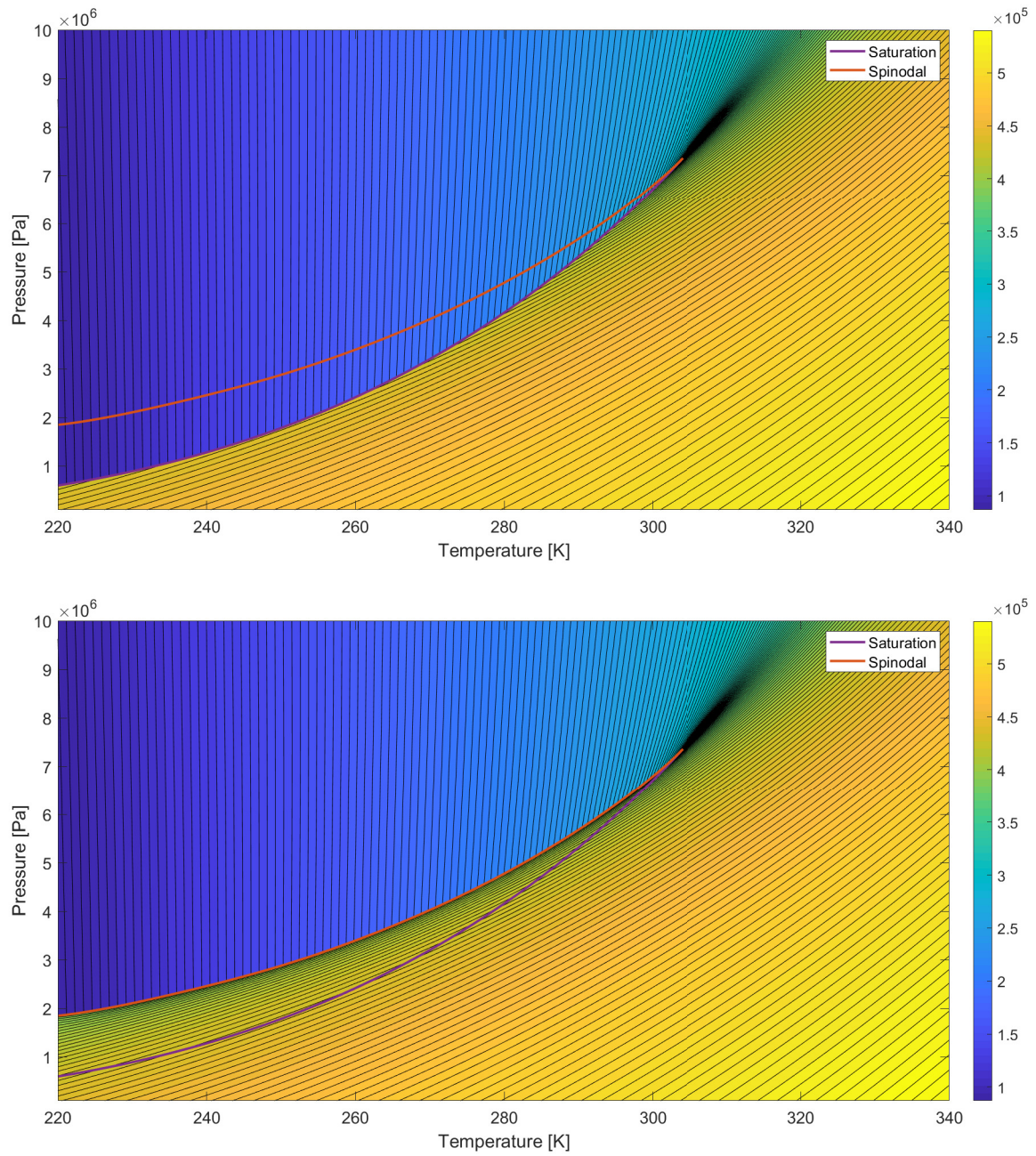


FIGURE 2.2: Enthalpy maps, with (below) and without (above) metastable vapour properties

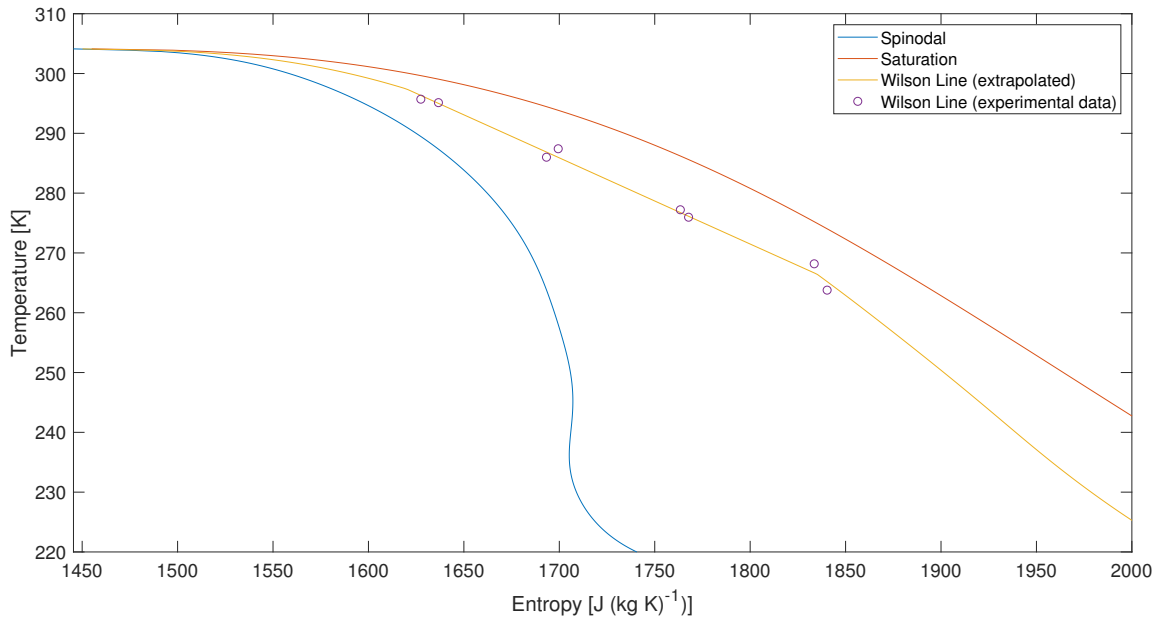


FIGURE 2.3: T-s diagram showing transition lines

the extrapolated Wilson region (in general, not too far from the experimental points), except for the highest pressure case. Anyhow, in regions near the critical point, a precise assessment is going to be challenging nonetheless, and the distance between saturation and spinodal is limited no matter the chosen approach, hence this process should not impact significantly the accuracy of our simulations.

The final result, showing also saturation and spinodal vapour line, is visible in Figures 2.3 and 2.4, in the $T - s$ (where the area lying below the half dome represents two-phase conditions) and $P - T$ (where only monophasic states are represented, i.e. the saturation line denotes transition from superheated vapour to subcooled liquid) thermodynamic planes.

It is worth mentioning that at low temperatures the REFPROP formulation for the spinodal line exhibits an unusual behaviour and is probably not completely accurate. However, the flow expansions of interest for the present work, which can as a first approximation be conceptualised as vertical lines passing through the purple points in Figure 2.3, are not going to cross those regions. In fact, the two isentropic expansions at lowest pressure (higher entropy) are never going to cross the spinodal at all. Theoretically, they should eventually cross the triple point (which lies quite near the lower limit of the graph, at around 216 K), but it is expected that the flow will start to condensate before then. The modelling of ice formation would be rather complex and out of scope for our interests, and was therefore excluded.

2.1.1 Experimental reference

More details on the experiments carried out by Lettieri [22], which will be used as reference, are provided.

Five different thermodynamic conditions were tested; each one was replicated to ensure repeatability, for a total of 10 different experiments.

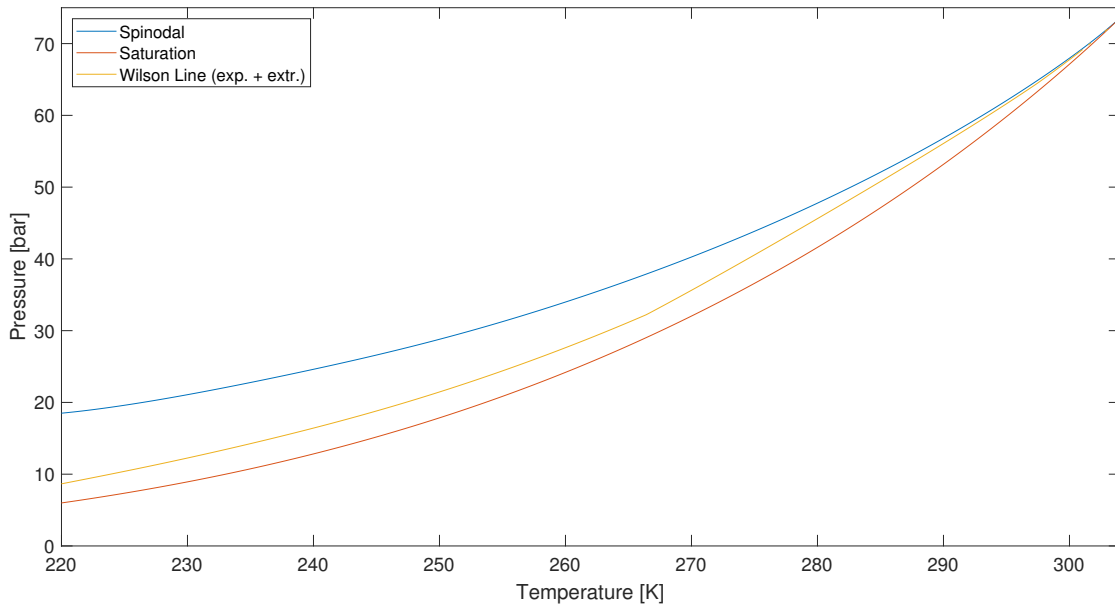


FIGURE 2.4: P-T diagram showing transition lines

The desired or conceptual thermodynamic conditions of the expansions are reported in Table 2.1, while Table 2.2 details temperature and pressure that were actually achieved during the experiments in the charging tank, alongside the condensation location on the nozzle axis. There are some slight differences that are imputable to the practical calibration of parameters. The entropy reported by Lettieri in his equivalent of Table 2.1 does not correspond precisely to the values at those pressures and temperatures as calculated by the Span-Wagner Equation of State; we will report our values. That notwithstanding, the difference is quite low - around $10 \text{ J}/(\text{kg K})$.

More importantly, the difference between actual and ideal conditions introduces some uncertainty on the proper boundary condition values to introduce in our numerical simulations, which will be discussed in Chapter 5.

The reference provides as well the numerical values of static pressure at 13 probes (with an uncertainty of 0.1 bar) equally spaced along the nozzle walls, where the fifth measures the throat pressure. The experimental reference does not provide precise values for the probes locations. From the original paper of Lettieri, further confirmed by analysing the results provided by Hosangadi [16], who simulated this nozzle, it was determined that the first probe lies at 28 mm upstream the throat, in the convergent, while the last is 56 mm downstream of the throat, in the divergent. The spacing, consequently, is constant between each probe and it is equal to 7 mm .

The nozzle has a rectangular cross shape with a throat area of 20 mm^2 ; its complete length is 155 mm (though only a section of it, i.e. the one that actually shows the convergent-divergent behaviour, was deemed of interest for the work and therefore simulated). The outlet to throat area ratio is 1.3; the inlet to throat area ratio is 4.1. The nozzle is charged from a tank of volume amounting to 400 l . Its expansion rate is as follows:

$$\dot{p} = -\frac{u}{p} \frac{dp}{dx} = 7.5 * 10^3 \frac{1}{s}$$

Case	Total Temperature [K]	Total Pressure [bar]	Entropy [J/kg K]
1	310	58	1843
2	310	65	1784
3	310	73	1703
4	311	80	1617
5	312	84	1560

TABLE 2.1: Ideal expansions

Case	Total Temperature [K]	Total Pressure [bar]	Condensation location [mm]
1-1	314.78	57.24	5.416
1-2	314.67	58.96	3.748
2-1	311.87	63.9	2.022
2-2	311.99	65.35	1.660
3-1	313.37	72.53	0.388
3-2	313.60	63.53	-0.505
4-1	313.68	79.18	-3.150
4-2	313.94	79.99	-3.462
5-1	313.83	84.02	-4.624
5-2	313.88	84.74	-4.675

TABLE 2.2: Experimental tank conditions and condensation location

2.2 Numerical models for condensation simulation

In general, multi-phase flows present a significant challenge in terms of numerical implementation in CFD codes. Two approaches are possible: Eulerian-Eulerian and Eulerian-Lagrangian. In the former, both phases are represented in Eulerian terms - i.e., properties are represented by flow fields. Contrarily, in Eulerian-Lagrangian models a phase (generally the nucleating one, liquid for condensation or vapour for evaporation) is represented by single particles (droplets or bubbles) that have their own specificity and are separated from the surrounding flow.

Eulerian-Lagrangian models available in commercial CFD codes have been developed and extensively validated for steam condensation; they refer to the classical non-equilibrium condensation theory for steam turbines which dates back to the '60s [24] and is based on the free energy analysis of a two-phase system for a single, pure substance. In a realistic situation, this phenomenon has to be coupled with the so-called heterogeneous condensation, which is instead due to wall contact, impurities, etc., and also plays an important role. Most solutions couple these two condensation models, which express the nucleation rate (how many particles will appear in a set volume); a third set of equations, regarding the particle growth and its interaction with the surrounding flow, has to be solved as well.

There is a wide literature on the applications of these models to steam expansions; for instance, Starzmann [34] made use of them in CFX for the analysis of a multi-stage LP steam turbine, and was able to track quantitatively the amount and radius of liquid droplets that were generated in each stage.

As far as carbon dioxide is concerned, literature references are less common. Furu-sawa [14] was able to achieve convergence on his in-house code with a model of this kind for the lowest pressure case of Lettieri's experiments. He coupled the condensation model with two thermodynamic options: ideal and real gas. The former proved to be inadequate for the case at hand.

Lettieri himself [22], in his paper, proposes his own simulations for the case at hand, employing a condensation model written in CEL (CFX own programming language) inside ANSYS CFX. His results, which are limited to the convergent section of the nozzle, could not be replicated in this work. No convergence could be obtained without including the divergent in the geometry model.

The overall accuracy and precision of these models would make them the most suitable choice for the phenomenon at hand. However, they have been developed for a different substance (water) and different thermodynamic conditions - pressure in the final stages of a steam turbine is one order of magnitude lower than for critical or near-critical carbon dioxide. This implies that their adaptation to the present case would require significant fine tuning of semi-experimental parameters and the employment of different sub-models. Some of them, in commercial CFD codes, may not even be directly accessible by the user.

As-is, convergence with these models proved to be unattainable in the course of this work in all situations.

Therefore, they will not be discussed in further detail; the main focus of this work will instead lie on Eulerian-Eulerian two-phase models.

While these models cannot predict as accurately two-phase quantities, they can still be useful if the purpose of the analysis is to assess the condensation risk and its influence (in terms of volume or mass fractions) on the overall flow quantities (pressure, temperature). This can be applied to single-phase CO₂ compressors, where the determination of condensation-free conditions may be of interest, or even to two-phase machines if condensation is thought to have a small enough role.

These models may be categorised on the amount of flow fields (velocity, pressure, temperature) that are shared between the phases. This allows for a remarkable simplification of the numerical solution, as equations can be summed up and solved for the mixture rather than for individual phases, with subsequently improved convergence properties. Specifically, if the velocity field is to be the same, the model will be defined as *homogeneous*.

Another important feature comes from the relative thermodynamic equilibrium between the phases, which will be the main difference between the two models that will be employed in the course of this work and are discussed below.

2.2.1 The Homogeneous Equilibrium Model (HEM)

The HEM model is the simplest mathematical form to handle a situation of multiphase flow. It assumes complete equilibrium between the two phases [28]. Therefore, the flow fields for pressure, temperature and velocity are shared between vapour and liquid; all flow equations are not solved for a single phase, but for the overall mixture, and they take the same analytical (and numerical) form as in a monophasic situation. Naturally,

the difference comes from the fluid properties - which are the mixture ones (rather than those of a single phase).

Most importantly, the model assumes the complete thermodynamic equilibrium between the two phases. Once the transition line is crossed, vapour and liquid phase, being at equilibrium, reach and maintain saturated conditions; this binds pressure and temperature to lie on the $P - T$ transition line. The various mixture quantities (enthalpy, entropy, density) are simply as follows:

$$\phi(T, P) = \phi(T_{trans}, P_{trans}) = \alpha_v \phi_v(T_{trans}, P_{trans}) + \alpha_l \phi_l(T_{trans}, P_{trans}) \quad (2.1)$$

Where α represents the mass fraction for mass-based quantities (enthalpy, entropy) but the volume fraction for density, while ϕ represents the saturated quantity for the relevant phase (v, l subscripts stand for vapour and liquid), defined as its value on the chosen transition line.

From a purely physical point of view, this should be the saturation line; the model strictly does not allow for any excursion into the two-phase dome and negates the very definition of non-equilibrium condensation. However, when applied to a CFD software, the transition line can be defined as any thermodynamic line where liquid and vapour phase can coexist (i.e. any thermodynamic line between the vapour and liquid spinodal), by defining the saturated properties as the relative liquid and vapour values on the chosen thermodynamic loci. If the Wilson line is to be chosen, this allows us to account for metastable properties as well, and provide results that are theoretically accurate as far as the condensation onset and single-phase region are concerned. However, once non-equilibrium condensation begins, the physical flow will slowly start to revert to saturation conditions, while the HEM will force a numerical equilibrium and make it lie on the chosen transition line.

Considering that the system is composed only of two phases, this obviously holds:

$$\alpha_v + \alpha_l = 1$$

Therefore, the knowledge of any of the two fractions, coupled with either pressure or temperature, implies the knowledge of the complete thermodynamic state of the flow.

This model has been employed by Rinaldi [30] in his in-house solver for the analysis of a supercritical CO₂ compressor, with the ultimate aim of detecting regions where condensation may take place. It was determined that these regions were small enough to have negligible effects on the overall machine performance.

De Lorenzo [23] used it to simulate fast transient water flows, with good results. In general, the HEM model is a sound choice whenever the deviation from equilibrium is limited and can be either neglected or otherwise modelled with simple additional specific terms in the equations. Its simplicity implies it is also numerically robust.

This model may be suitable only for small penetrations into the two-phase flow region, where the assumption of thermodynamic equilibrium, although incorrect, may not impact significantly the overall accuracy. Larger penetrations, aside from being incorrect, may also end up being numerically unstable, as the solver tries to enforce an unphysical situation - $P - T$ equilibrium - in large areas of the flow. It will be shown that

this model, however, does not achieve realistic results even for the lowest total pressure expansion of Lettieri's test case.

Numerical implementation

This model was tested on ANSYS CFX, where it is called *Equilibrium Phase Change Model*; it is directly implemented if the chosen material type inside the software GUI is a Homogeneous Binary Mixture (and therefore turned off if, instead, the chosen type is a Continuous Fluid, i.e. a single-phase gas or liquid). In that case, the software has knowledge of the liquid and gas saturation properties; once it detects that the solution is falling below the transition line, it autonomously starts transition, applying the lever rule, by inverting Equation 2.1. Namely, the calculation is carried out from static enthalpy [18]:

$$x_v = \frac{h_{mix} - h_{trans,l}}{h_{trans,v} - h_{trans,l}}$$

The transition enthalpies are determined from the local flow pressure, while the mixture value comes from the solution of the energy equation. Depending on the value of x_v , three situations are possible:

- If $x_v > 1$, the flow is superheated vapour, and therefore only vapour properties are needed;
- If $x_v < 0$, the flow is subcooled liquid, and thus it takes liquid properties;
- If $0 < x_v < 1$, the flow is under these assumptions a mixture of saturated vapour and liquid, and its properties are in general calculated from Equation 2.1.

In the last situation, density, a volumetric quantity, is calculated through the use of harmonic averaging (this avoids the explicit derivation of volume fractions):

$$\frac{1}{\rho_{mix}} = \frac{1 - x_v}{\rho_{trans,L}} + \frac{x_v}{\rho_{trans,v}}$$

Its derivatives behave coherently. The Homogeneous Binary Mixture in CFX can be expressed both as a user-defined real gas / liquid material (through the use of .rgp files, as described in Chapter 3), or from already implemented real gas equations of state.

Other multiphase options and models are possible if, instead of the single mixture, two materials (representing liquid and gas) are specified in the domain definition. However, while this allows for more complex modelling of interphase heat and mass transfer, no simple solution that could allow for satisfying and converging results was found.

2.2.2 The Lee Model

The Lee Model is among the simplest solutions that allow for the simulation of non-equilibrium condensation. It requires that the phases share a single pressure flow field

(which is quite natural for the case at hand); it does not enforce thermodynamic or thermal equilibrium, and can be applied to dishomogeneous flows as well.

In the present case, it was determined that the slip effect would be negligible, considering that the liquid volume fraction will be numerically limited due to the extremely fast nature of the expansion. Similarly, the potential temperature differences were considered to be limited. They are also impossible to assess from the experimental reference (which provides only values from pressure probes). Not only; even if the heat phase change between the two phases were to be modelled, the intrinsic limitations due to the Eulerian-Eulerian framework would already cause significant errors.

Therefore, only the thermodynamic non-equilibrium will be considered, and it will be assumed that temperature and velocity will be shared.

As a consequence, the fluid properties are not strictly bound by the relevant saturated values and excursion in the two-phase dome is allowed; the lever rule as described by Eq. 2.1 is still valid for the mixture properties; however, since pressure and temperature are not forced to lie on any given thermodynamic line, here ϕ_v and ϕ_l are not saturated quantities but simply the fluid properties calculated at the given conditions (therefore, they can be metastable states as well).

Mathematically, the transition between the two phases is governed by a source term that takes the form [19]:

$$m_{cond} = K_{cond} \alpha_v \rho_v \frac{T_{sat} - T_v}{T_{sat}} \quad T < T_{sat} \quad (2.2)$$

$$m_{evap} = K_{evap} \alpha_l \rho_l \frac{T_l - T_{sat}}{T_{sat}} \quad T > T_{sat} \quad (2.3)$$

Where T_{sat} represents the saturation temperature at the flow pressure. Although, once again, the saturation line could be switched for any desired thermodynamic line where liquid and vapour can coexist, in this framework any choice of this kind (except the actual saturation line) would simply be incorrect, as the terms described by Equations 2.2 and 2.3 are already supposed to represent non-equilibrium effects. In general, the condensation term is activated if a vapour flow crosses the saturation line; the opposite for the evaporation one. The net balance of the two terms appears as a source in the vapour fraction transport equation; namely:

$$\frac{\partial}{\partial t}(\alpha_v \rho_v) + \nabla \cdot (\alpha_v \rho_v \vec{V}_v) = m_{evap} - m_{cond}$$

Once condensation is activated, eventually the source terms are supposed to bring the flow back to equilibrium conditions; they will drive the numerical process to achieve $T = T_{sat}$. The speed by which this phenomenon will happen depends on a coefficient - K - which, in general, can be interpreted as a *relaxation time* for the condensation onset. It can be determined analytically for simple cases from thermodynamic considerations and should rigorously be different for the two terms (evaporation and condensation); however, in practical cases it usually requires some experimental fine-tuning [19].

In general, the higher the value, the greater the source term (at the same degree of subcooling, i.e. distance from the saturation line); consequently, at high coefficient values the non-equilibrium effects will be less important, because less subcooling will be allowed before condensation becomes prevalent and brings the flow back to equilibrium conditions.

There are some results from this model for high pressure CO₂; Mazzelli [15] successfully used it to predict carbon dioxide flashing (i.e. abrupt evaporation) in a test case by Nakagawa [26], using ANSYS Fluent. He included a numerical analysis of the K coefficient (for evaporation) to determine the value that could achieve better results; ultimately, the value of 10^5 was chosen (it is worth mentioning that it is higher than the admittedly loose upper bound value of 10^3 provided by Fluent user guide [19]).

While the mathematical nature of the relations discussed above - namely, the form of the source terms and their position in the system of equations - is specific to Fluent, several variations that make use of the same concept - phase change driven by source terms whose magnitude depends on the thermodynamic distance from the saturation line - exist. The Homogeneous Relaxation Model, for instance, can be assimilated to the present case, though it is usually applied in situations of evaporating flows and it considers the vapour phase to lie at saturation conditions; it has been extensively studied and analysed (for instance, in [23] [28]).

Hosangadi [16] developed for his commercially available CRUNCH CFD software an original formulation of a condensation model that belongs to this framework. As already mentioned, he simulated Lettieri's experimental test case, and achieved very good adherence with the experimental reference. His source terms take the form:

$$m = \frac{1}{\tau \frac{L_{scale}}{V_{scale}}} * \frac{P - P_{sat}}{\frac{1}{2} \rho_{scale} V_{scale}^2}$$

Where τ , the time scale for the droplet (or bubble) formation, can be different for evaporation and condensation. The terms are non-dimensionalised through the use of a scale length, velocity and density.

This type of formulation is slightly different from Fluent own implementation, namely in the non-dimensionalisation process. While in Fluent phase-relevant quantities are employed, Hosangadi's model makes use of general scale terms. A quantitative and detailed comparison of the differences between the two models is hard and difficult to separate from the fact that they are implemented in different programs.

Numerical implementation

ANSYS Fluent can easily implement this model in the framework of the Mixture Multiphase Model, under the Evaporation-Condensation Mass Transfer mechanism. All such features require the specification of a sign convention in the GUI, under the form of 'from' and 'to' phases, to determine which mass flux will be defined as positive. The software convention for the Evaporation-Condensation model requires the liquid phase to be the 'from' phase, while vapour will be the 'to' phase.

This mechanism could also be implemented with the Volume Of Fluid (VOF) Multiphase Model; however, this was thought mainly for non-interpenetrating phases (such as sludges). According to the software guidelines [19], as well as the previously mentioned paper by Giacomelli [15], the Mixture Model was deemed to be the correct choice. The other possibility allowed for by the software would be the Eulerian Multiphase Model, that allows for more degrees of freedom and can be applied to a very wide range

of multiphase phenomena. However, these options were not of interest for the present case; although it may provide increased accuracy, it was noted that the Mixture Model provided satisfying results, while the most notable issues lied in obtaining convergence. Therefore, the added complexity and numerical instability of the Eulerian model ruled out its use.

In general, the Mixture Model is the simplest option for multiphase flows implemented in Fluent; the name is due to the fact the code solves the momentum, continuity and equations for the mixture, rather than for single phases. It can allow for velocity slip (calculated through algebraic expressions) and thus can be applied to model dishomogeneous flows as well (though it assumes local equilibrium at short spatial length scales); as previously mentioned, this was not of interest for the present case and therefore the flow was assumed to share a single velocity field between the phases.

The Mass Transfer mechanism required the specification of the saturation temperature as either constant or function of the saturation pressure; this was implemented through a `DEFINE_PROPERTY` User Defined Function (UDF) which interpolates between a dense array (300 points) of temperature and pressure values that was compiled beforehand, from `REFPROP`, with a MATLAB script (details on how thermodynamic calls were handled are provided in Chapter 3, while the script is provided in Appendix C); it was stored into memory by Fluent via a `DEFINE_ON_DEMAND` routine. The source code is provided in Appendix D. The two arrays were bound by triple and critical point.

In general, at runtime, the software will need the definition of properties for both phases at each and every iteration, even if the phase itself is not active (i.e. its volume fraction is null). Then, it activates the source terms as described by Equations 2.2 and 2.3, obtains the relevant volume fractions and proceeds to calculate the overall mixture properties following the standard lever rule (Equation 2.1).

This implies that property values for both phases in all conditions need to exist, even if they are physically meaningless (e.g. liquid properties in superheated vapour conditions). They should not be referred to in the final converged solution, which is supposed to have physical meaning.

Chapter 3

Thermodynamic Properties of CO₂ and CFD Input

In this chapter the thermodynamic properties of carbon dioxide and the algorithms, procedures and utilities needed for their implementation into the CFD software will be described.

3.1 Equations of State for carbon dioxide

Generally, Equations of State represents the relation between state variables (P, ρ, T) that lets us ascertain the fluid properties at given conditions. It is the analytical determination of the fluid nature.

The Equation of State for ideal gases ($P/\rho = RT$) represents its most simple form and is one of the basic relations of thermodynamic. It assumes that there is no interaction between individual particles; it is valid for light (low molar mass) gases at conditions that are far from phase transition. It becomes inaccurate in proximity of the saturation line; therefore, other equations - for real gases, accounting for the more complex interactions taking place inside the fluid - are necessary. Historically, they were developed as semi-empirical 'correction' terms that represent the fluid deviation with respect to the ideal behaviour.

In proximity of the critical point, the determination of thermodynamic properties becomes even more challenging; thermodynamic gradients are extreme and any type of approximation that may link the fluid behaviour to an ideal gas is incorrect. Even non-specific real gas models may have troubles determining properties with appropriate accuracy, owing to their general form simply being unable to correctly model such complex conditions. This issue requires the use of a suitably complex state equation. A few common alternatives are reported below.

3.1.1 The Peng-Robinson Model

One of the most successful and common Equations of State for real gases has been developed by Peng and Robinson [31]. It is expressed as follows:

$$P = \frac{RT}{v - b} - \frac{a(T)}{v(v + b) + b(v - b)} \quad (3.1)$$

Derivation of other thermodynamic quantities is mostly straightforward. Equation 3.1 equates to expressing compressibility ($Z = Pv/RT$) as a cubic equation. Coefficients a and b are fluid specific; the latter is also constant.

$$b = 0.778 \frac{RT_c}{P_c}$$

$$a(T) = 0.45724 \frac{R^2 T_c^2}{P_c} \left(1 + n \left(1 - \sqrt{\frac{T}{T_c}} \right) \right)^2$$

$$n = 0.37464 + 1.54266 \omega - 0.26993 \omega^2$$

Where subscript c represents critical quantities. Coefficients are expressed starting from critical conditions, which in general are known *a priori* for the fluid and can be determined experimentally quite simply. The acentric factor ω is the only other property of interest, and has been tabulated for a great variety of fluids. It represents in some form the molecular shape difference compared to a sphere and is calculated from saturated vapour data.

The simple yet reliably accurate form of Equation 3.1 makes it suitable for a wide variety of purposes. In fact, the Peng-Robinson model is readily available and already implemented in most commercial CFD codes. REFPROP (Reference Fluid Thermodynamic and Transport Properties Database, a software developed by the US-based National Institute of Standards and Technologies), which is generally considered the reference concerning the determination of thermodynamic properties for fluids, can apply it to all substances available in its database.

3.1.2 The Lee-Kesler Model

The Lee-Kesler model [20] has been employed quite extensively by Baltadijev [6] for carbon dioxide, despite the fact that it was originally conceived for complex fluids (hydrocarbons). It defines as starting point compressibility Z . This is because it assumes that non-dimensional compressibility of any fluid (even complex ones, such as mixtures) can empirically be expressed as follows:

$$Z = \frac{P_r V_r}{T_r} = Z_0 + \frac{\omega}{\omega_r} (Z_r - Z_0) \quad (3.2)$$

In Equation 3.2 compressibility of the generic fluid, Z , is expressed in terms of a sum of the compressibility of a simple fluid (i.e. one that obeys only to fluid dynamics laws), Z_0 , and its deviation with respect to a reference fluid (expressed by Z_r). This equates to the assumption that the behaviour of reduced temperature and pressure is linear with respect to the acentric factor.

The compressibility factors can be expressed as follows:

$$Z_{0,r} = \frac{P_r V_r}{T_r} = 1 + \frac{B}{V_r} + \frac{C}{V_r^2} + \frac{D}{V_r^5} + \frac{c_4}{V_r^2 T_r^3} \left(\beta + \frac{\gamma}{V_r^2} \right) \exp \left(\frac{\gamma}{V_r^2} \right) \quad (3.3)$$

All coefficients in Equation 3.3 can be calculated empirically, and are tabulated. Subscript r denotes reduced quantities for temperature and pressure, but not for volume:

$$T_r = \frac{T}{T_c} \quad P_r = \frac{P}{P_c} \quad V_r = \frac{P_c V}{RT_c}$$

Subscript c represents critical quantities. The procedure to obtain compressibility and consequently all other thermodynamic quantities is detailed in [20], starting from a single pair of temperature and pressure values. The full determination of all thermodynamic quantities depends only on knowledge of the critical point and the acentric factor, similarly to Peng-Robinson.

This model is supposed to improve property accuracy in the near critical region and at low temperatures for liquids; both conditions (especially the first) are of interest for the present case. That notwithstanding, it is not the reference solution for carbon dioxide, it is not available in CFX or Fluent, or even REFPROP, and therefore its use would require the full manual implementation of all thermodynamic relations. This, eventually, ruled its use out for the present case.

3.1.3 The Span-Wagner Model

The models detailed above have been developed for general purposes, and can therefore be applied to a variety of fluids. A specific and precise multi-parameter equation was developed for CO₂ in 1996 by Span and Wagner [32], which employs a thorough fit of the vast experimental data available for the substance. A precise analysis of its behaviour and formulation is out of scope for the present work; a few traits regarding its formulation are nonetheless introduced below.

Fundamentally, it is based on a non-dimensional formulation of the Helmholtz free energy, which can subsequently be used for analytical derivation of all thermodynamic properties.

The Helmholtz free energy is written as follows:

$$\phi(\rho_r, T_r) = \phi^0(\rho_r, T_r) + \phi^1(\rho_r, T_r) \quad (3.4)$$

All quantities reported above are non-dimensional; namely:

$$\phi = \frac{A}{RT} \quad \rho_r = \frac{\rho}{\rho_c} \quad T_r = \frac{T}{T_c}$$

Here A represents the dimensional Helmholtz free energy. Equation 3.4 represents its division into two terms; the first represents the contribution due to ideal quantities and is defined analytically, while the second represents the real gas contribution and is determined by the fit of experimental data.

Ideal Helmholtz energy is defined as follows:

$$A^0 = h^0(T) - RT - Ts^0(\rho, T) \quad (3.5)$$

Enthalpy and entropy of the ideal gas can be expressed from an equation for the ideal isobaric specific heat, c_p^0 .

The end result represents the state of the art for carbon dioxide; its accuracy is comparable to experimental uncertainty. It is also employed by all recent versions of REFPROP as the default option for carbon dioxide [33], therefore representing the most common alternative for the fluid simulation, if real gas effects are thought to be of interest.

Considering that our aim lies in the determination of an accurate setup for carbon dioxide, thermodynamic precision was deemed to be crucial. Therefore, the Span-Wagner equation was implemented in all simulations discussed in this work. Details concerning this process are reported below.

3.2 Implementation strategy

The complexity and novelty of the Span-Wagner formulation implies that it lacks direct implementation in most commercial CFD codes, which, nonetheless, allow for full specification of thermodynamic properties via the so-called user-defined materials. Their use was therefore needed for the implementation of this model.

The direct solution of the equation at runtime would be technically possible, but complex in implementation; also, it would unnecessarily slow down the solution process. Hence, a different approach - in the form of lookup tables - was employed. Basically, a list of tables that represent all relevant properties for the material (e.g. enthalpy, entropy, specific heat...) as function of two independent thermodynamic variables is compiled beforehand and stored in memory. At runtime, the software can read from the tables and interpolate between the nearest values so to return the desired property.

There are obvious limitations to this approach; most importantly, only a finite area of the thermodynamic plane can be mapped. Consequently, care must be put into choosing appropriate table bounds so to include all possible flow conditions inside the domain. Another issue lies in the table number of points and the implemented interpolation method; dense enough tables must be created in order to reduce interpolation errors between the single points, which will be, in any case, always present.

The number and type of properties required, as well as the two independent variables required for the table compilation, is generally determined by the CFD software requirements and its architecture. Both ANSYS Fluent and ANSYS CFX use pressure and temperature as independent variables, though Fluent allows also for density. The supplied properties vary and depend on software architecture.

In general, defining Lookup Tables from intensive properties (such as $P - T$) implies that two-phase states cannot be represented (differently from what happens, for instance, in the $T - s$ plane) and a third variable, in form of volume or mass fractions, is required for the determination of the complete flow state. This would seemingly make $T - d$ tables a better choice, at least as far as Fluent is concerned. However, two issues arise. Firstly, both Fluent and CFX use a single set of tables to represent a single phase. Therefore, there is no strict need to model two-phase states. But, most importantly, even if a set of tables was used to represent a full two-phase mixture, an issue arises in the compilation of regions lying between saturation and spinodal lines (as a reference, the region between the red and the blue line in Figure 2.3). As already discussed in Chapter 2, if metastable states are to be accounted for, these regions are intrinsically ambiguous: in terms of $T - s$ (analogously for $T - d$), a single point could correspond to both metastable vapour and

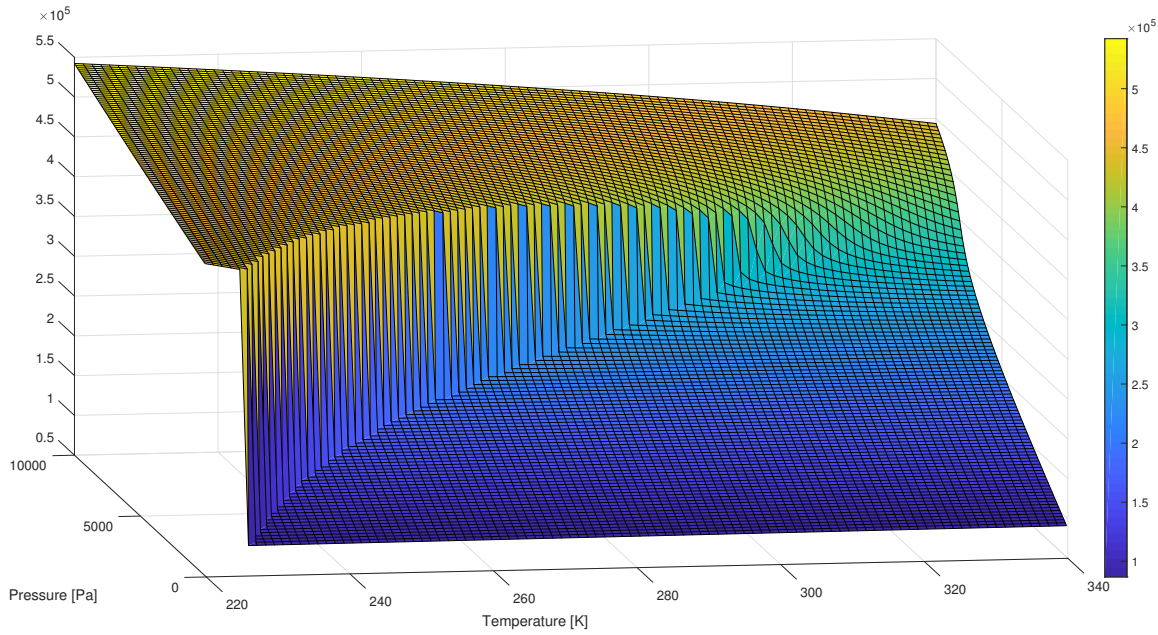


FIGURE 3.1: Sample surface representing enthalpy as function of temperature and pressure

two-phase mixture. As a consequence, similarly to what is necessary for $P - T$ tables, two sets of tables need to be compiled, and appropriate programming to resolve between them is ultimately complex. Conversely, in $P - T$ tables ambiguity arises between metastable vapour and subcooled liquid states; this is much easier to resolve, owing to the fact that only knowledge of mass, or volume, fractions is needed.

However, the transition line represents an issue concerning table compilation. If the full tables are to be written, a sharp discontinuity would arise at the transition line (be it saturation or spinodal) between vapour and liquid properties. This can be easily seen from Figure 3.1 (and it was already indirectly shown in Figure 2.2), which shows the 3D representation of a sample enthalpy table where the discontinuity is represented by the saturation line. This is problematic even if a single set of tables is used to represent a single phase, because in general the software may not be aware of that transition line, and therefore may ask during the solution process for an input lying beyond it, which is clearly harmful for numerical convergence.

This specific issue arises because of the use of the Span-Wagner Equation of State, which shows non-convergent behaviour beyond the spinodal line as implemented by REFPROP. If the Peng-Robinson or the Lee-Kesler model were employed, instead, their functional form would allow for properties to be smoothly extrapolated all across the $P - T$ plane. This would be physically meaningless, though potentially interesting for numerical purposes.

All lookup tables were generated and written as ASCII files with MATLAB scripts; the thermodynamic calls were carried out with a function (`refpropm`, freely available on GitHub) that links REFPROP and its libraries with MATLAB and allows for simple and compact syntax. The function was slightly modified from its original form so to include the possibility to call for metastable properties.

Differently from equilibrium states, as already discussed, metastable properties are intrinsically ambiguous; a single input could return two different thermodynamic states

(e.g. metastable vapour or subcooled liquid). REFPROP resolves this issue with a secondary input, a specific integer value which forces its search into either of the phases [33]. This value was therefore added among the required input values for the `refpropm` function. Forcing the calculation of unfeasible thermodynamic inputs - for instance vapour states below the spinodal - will cause the software to return an error.

Finally, it must be mentioned that the choice of $P - T$ as independent variables greatly simplified the table writing procedure, as metastable properties in REFPROP can be called only as functions of temperature and pressure or temperature and density. If different independent variables were required, a more complicated approach involving bidimensional interpolation would need to be devised (as briefly illustrated by Paxson [29], though for other purposes).

3.2.1 ANSYS CFX[®]

ANSYS CFX offers a single choice - lookup tables - in terms of user-defined materials; the complete implementation of an arbitrary equation of state via the specification of all thermodynamic relations is not possible.

The tables are written in a very specific format (named `.rgp`, which stands for *real gas properties*, though it can be used to model real liquid properties as well) [18]. Once the file has been created, it can be easily loaded into the software GUI; table storage into memory and the interpolation at runtime are carried out autonomously by the software, and so are out-of-bounds situations; the user cannot influence its behaviour (though they are warned of the situation at runtime).

The file is composed of two types of tables; one supplies the superheated (or subcooled, for a liquid material) fluid properties, the other both liquid and vapour properties at an arbitrarily defined transition line - the saturation as well as the spinodal or, in general, any thermodynamic line where liquid and vapour properties can be determined (even if one phase is technically metastable, the software cannot discern that as long as it is fed coherent values).

A single set of tables, composed of the two aforementioned types, represents a single material in a single phase. Consequently, in two-phase problems two different sets must be generated: one for the liquid phase, one for the vapour phase. Then, two different materials each representing one phase and employing the appropriate set of tables must be defined inside CFX. The information regarding which phase is of interest is directly required by the GUI, so that the software may know which side of the tables it must work with (above or below transition).

Finally, the last table of the set, containing saturation properties, is directly called and read only by a third material defined inside the GUI - as *Homogeneous Binary Mixture*, or HBM. The last table should be the same for both liquid and vapour sets; therefore, the HBM can be associated to either of the two indifferently. The choice of this material, which requires to specify the two phases that compose it, automatically activates the HEM model, as described in Chapter 2.

This simulation scheme is represented in Figure 3.2.

The `.rgp` file structure is as follows. Firstly, the file requires some initialisation parameters; these are, in order:

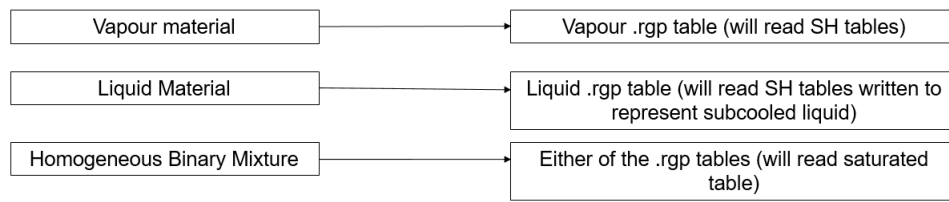


FIGURE 3.2: CFX simulation scheme

- Fluid name;
- Type of unit system applied (SI, or variations);
- Pressure and temperature bounds for the superheated tables;
- Temperature bounds for the saturation table;
- Triple and critical point properties (needed to define the area where liquid and vapour phase may coexist);
- Gas constant: R/M , where R is the universal gas constant and M is the fluid molar mass.

The superheated tables are comprised of five separate elements:

- Temperature query points;
- Pressure query points;
- Property values (expressed by first varying temperature at a single pressure level, for all pressure levels);
- Saturation line temperatures as function of the pressure query point;
- Saturated material properties.

In general, this creates a $P - T$ grid in which an arbitrary line representing the phase transition between liquid and vapour phase is determined by the last section of the table; this, as mentioned before, generates a discontinuity. The property values linked to the phase which is not of interest for the material are, according to the guide, to be clipped in temperature; this means they will be represented as $\phi = \phi(T_{sat}, P)$, where T_{sat} represents the saturation temperature at pressure P . In general, they will never be used by the code.

The saturation table requires a simpler syntax:

- Saturation pressure query points (as functions of temperature, which defines the line bounds);
- Saturation temperature query points;
- Saturation properties (first saturated liquid, then saturated vapour properties in the same order).

CFX requires 9 properties for both single-phase and saturation tables (here expressed in SI measure units):

- Enthalpy [J/kg];

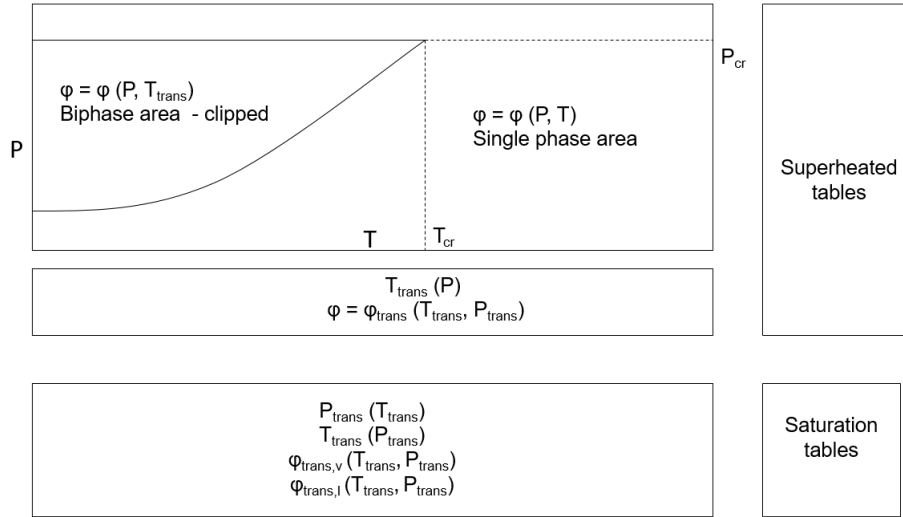


FIGURE 3.3: Structure of a .rgp file

- Sound speed [m/s];
- Specific volume [m^3/kg];
- Specific heat at constant volume [$J/(kg K)$];
- Specific heat at constant pressure [$J/(kg K)$];
- Partial derivative of pressure with respect to specific volume at constant temperature [$Pa kg/m^3$];
- Entropy [$J/(kg K)$];
- Thermal conductivity [$W/(m K)$];
- Dynamic viscosity [$Pa s$].

The last two properties are optional, and can be otherwise specified in the software GUI by means of some other internal correlations. Nonetheless, their generation was straightforward and simple to include in the MATLAB routine that was developed for the table generation. In general, it included metastable properties up to the spinodal line as calculated by REFPROP and the ability to handle any phase transition line specified by the user, in order to create tables that could be clipped at the Wilson line (extrapolated as mentioned in Chapter 2). The structure of the .rgp file is visualised in Figure 3.3.

If an input beyond the chosen transition line is required during the solution process, and no condensation model is active (i.e. the material inside the flow domain is not defined as the mixture, but rather as single-phase gas or liquid), the software behaviour is not clear and is not discussed, to the author's knowledge, in the documentation. From some tests that were conducted to this purpose (which will be discussed in more depth in Chapter 6), it is thought that, since a transition line is still specified (in the last section of the superheated tables and via the clipped area), the software extrapolates autonomously the properties following a quadratic line. This may be unstable, and in general is arbitrary.

Therefore, one last possibility regarding the table generations involves setting all saturation-related properties to a constant arbitrary value (i.e. 1). If this is done, CFX effectively does not recognise that any sort of phase transition is happening, and therefore never extrapolates or clips values. The extrapolation scheme can therefore be defined by the user in the table writing process to fully cover the $P - T$ thermodynamic plane as desired.

This comes at a downside; without awareness of the transition line, CFX cannot initiate condensation. Therefore, simulations performed with these tables are bound to be single-phase. This may still be relevant if condensation is considered not to be relevant or to be assessed only in post-processing, and in general to obtain a converging solution even in unfeasible physical areas.

3.2.2 ANSYS Fluent®

Differently from CFX, Fluent offers several possibilities regarding user-defined materials. Technically, Fluent can be linked to REFPROP [19] in order to either directly supply the properties or automatically generate lookup tables. However, this option is limited to a single material and is directly linked to REFPROP standard formulation - the inclusion of metastable properties is not straightforward and, most importantly, a property request in a physically unfeasible area (beyond the spinodal; this may happen during the first iterations) would return an error from REFPROP and therefore crash the software. As a consequence, this feature could not be employed, and a User Defined Real Gas Model was instead developed for the present case; this consists of a C file that was written, then compiled inside Fluent to create the required libraries, and eventually loaded.

The foremost issue coming from this procedure lies in the fact that only one UDRGM can be employed at a given time and one UDRGM can model only a single material, i.e. a single phase. Considering our problem regards condensation and namely its onset, liquid properties are somewhat less relevant than vapour properties. Hence, the choice was relatively straightforward and this model was employed for simulating the vapour phase. A workaround had to be found to circumvent the issue and include the liquid phase, which will be discussed below.

In this framework, the condensation model is not linked to the choice of material types in the software GUI, and therefore can be defined separately from the materials. Consequently, there is no need to define any type saturation properties here; the saturation line is fed separately to the software (the technical approach was mentioned in Chapter 2 and involves a `DEFINE_PROPERTY` routine).

The User Defined Real Gas Model (UDRGM)

The UDRGM is a series of C functions [19] that is directly called at runtime and returns the desired property values. This undoubtedly allows for much larger flexibility in terms of approaches and solutions. This is its general structure:

- A function for handling error messages;
- A function for setup, which is executed when the Fluent simulation is loaded;
- 12 functions that each return a specific property value; their input is defined by Fluent to be limited to pressure, temperature, density and eventually mass fractions (for mixtures).

Considering the input limitations and the complexity that would arise from trying to directly implement the S-W model, for simplicity and analogy with CFX, lookup tables

were externally written (in a MATLAB script), while the UDRGM was used to read and interpolate from these tables.

In the setup function, tables are read and stored in memory, in a single three-dimensional array (its dimensions are thus the number of temperature points, the number of pressure points, and the total number of properties loaded in the function; all of these are defined as constants). Subsequently, a general bilinear interpolation function is written, which is meant to take as inputs pressure, temperature and a numerical index that determines which table will be used for interpolation. Possible out-of-bounds thermodynamic inputs are dealt with by clipping and setting them to the minimum (or maximum) value where the table exists. This is done separately for temperature and pressure; i.e. if temperature is out of bounds the function sets it as the minimum value and proceeds to linearly interpolate between two values corresponding at minimum temperature and nearest pressures.

Subsequently, each function linked to a property simply initialises an integer, passed to the interpolation function which uses it to discriminate which 2-D array (matrix) it should use for the bilinear interpolation.

Only 11 tables are supplied; the twelfth function returns the molar mass (which is initialised as a constant). These are the thermodynamic properties:

- Density;
- Specific heat at constant pressure;
- Enthalpy;
- Entropy;
- Speed of sound;
- Viscosity;
- Thermal conductivity;
- Density derivative with respect to temperature at constant pressure;
- Density derivative with respect to pressure at constant temperature;
- Enthalpy derivative with respect to temperature at constant pressure;
- Enthalpy derivative with respect to pressure at constant temperature;

Optionally, a function for the calculation of the enthalpy mixture in case of chemical reactions can be included. This was not of interest for the present case, and was thus ignored.

The problem regarding the phase transition in the P-T plane was dealt with during the table-writing phase. Two possible solutions were implemented in the table generator; either a clipping (similarly to the one that was discussed for ANSYS CFX) or an extrapolation of the properties to cover the entire P-T domain. First, a table in the physically sensible domain was written; then those values were used for the algorithm input and the resulting function was evaluated at the previously void domain points.

The same approach can also be used in case the table bounds needed to be extended near the triple-point area; at around 220 K, REFPROP and its formulation become rather unstable (due to the fact that the solid phase may begin to be present) and table compilation may fail. However, should the desired process bring the fluid around this area (which, although unphysical, may be touched either in the converging process or in case condensation models were to be deactivated), this solution may be required.

The source script is available in Appendix D.

Second phase definition

As mentioned before, the approach discussed above can deal with only one phase (namely, vapour). The question arises for the definition of the second phase. In general, another material should be defined inside the software itself to include the liquid properties.

One possibility that was applied by Mazzelli [15] employs a slightly different model - a *mixture* UDRGM. It is essentially a modification of the standard UDRGM that allows for the inclusion of multiple species. The properties - the same that were listed above - are defined also in terms of mass fractions, alongside the standard thermodynamic properties. It would normally be used to define a mixture of species in chemical equilibrium (the mixture UDRGM template coming from the software guide, for instance, defines air as a mixture of perfect gases); thus, it must be tweaked for our purposes (which involve *multiphase*, rather than *multispecies*, flows).

In fact, a two-species model should be written, where each species represents a phase. Once the model is loaded inside the GUI, the user-defined mixture can be duplicated; at this point, we have two mixtures. One of them is going to represent the liquid phase, the other the vapour. However, we still have four species; two must be numerically suppressed. If, at inlet and outlet, each mixture is bound to be composed entirely of one single specie - the relevant phase - and diffusion inside each mixture is prohibited (by setting the relevant coefficients to near-zero values), each mixture works effectively as a single material, and thus several user-defined materials (coded as species inside the mixture UDRGM) can be employed in a single simulation.

This approach, however, is rather complex and attempts to employ it proved to be unfruitful. A simpler solution was thus sought.

Alongside the more complete models that were just discussed, Fluent allows for the definitions of some material properties (namely: density, viscosity, thermal conductivity, speed of sound, specific heat and sensible enthalpy; these last two only as function of temperature) through its `DEFINE_PROPERTY` and `DEFINE_SPECIFIC_HEAT` routines.

Technically, density, speed of sound, viscosity and thermal conductivity all require a different `DEFINE_PROPERTY` routine for their definition; it was decided to employ once again lookup tables. The tables themselves are read and stored in memory inside a different routine, called `DEFINE_ON_DEMAND`, that can be executed at any time from the Fluent GUI; a separate function was written for the bilinear interpolation, which, similarly to what was mentioned above, is called by each routine for the respective property calculation. The independent thermodynamic properties representing flow state were accessed in each `DEFINE_PROPERTY` routine with the standard macros provided by Fluent (`C_P` for pressure and `C_T` for temperature). Specific heat was defined through the specific `DEFINE_SPECIFIC_HEAT` routine. It can take only temperature as input, and is used to calculate both the specific heat and the sensible enthalpy (i.e. the difference from the arbitrary reference value, which must be defined inside the software GUI alongside the reference temperature). Entropy, instead, is calculated internally by Fluent as:

$$s = C_p(T_{mean}) * \log \frac{T}{T_{ref}}$$

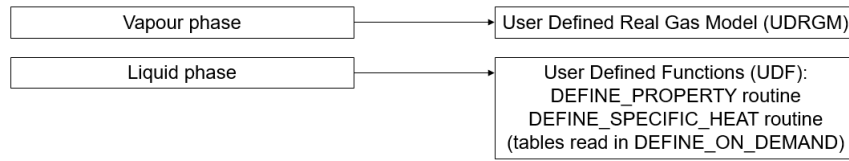


FIGURE 3.4: Fluent solution scheme

Where T_{mean} represents the mean logarithmic temperature between the actual value and the reference temperature.

This solution is illustrated in Figure 3.4.

This approach lacks the depth linked to using a full UDRGM; most importantly, the specific heat is bound to be only a function of temperature, rather than pressure and temperature as in a full UDRGM. However, these limitations were deemed not to impact significantly the solution, considering that any error linked to the liquid phase heat transfer is probably going to have a major impact only if condensation was modeled with absolute accuracy and large quantities of liquid appeared in the domain. The limitation should nonetheless be kept in mind. Should future software releases allow for multiple UDRGMs, that would probably be the most efficient and accurate route.

The source function for the liquid properties is available in Appendix D.

As one last note, this problem could be completely avoided if the User Defined Wet Steam Model was employed, as in that specific case all the required liquid and vapour properties would be user-defined. However, this model incorporates an Eulerian-Lagrangian condensation model (as discussed in Chapter 2), is rather complex and, as already mentioned, no converging simulations employing it could be obtained during the present work. For this reason, such models and related results are not further discussed.

Chapter 4

Quasi-1D Solution

The purpose of this chapter is the illustration of the monodimensional solution that was developed in MATLAB during the course of this work, in order to provide a numerical reference for the nozzle solution that could be fully controlled and governed. The problem to be schematised consists of a blow-down from a tank at a given pressure and temperature into a converging-diverging duct, also known as De Laval nozzle.

4.1 Analytical framework

The purpose of this section is only to briefly remind the analytical schematisation of the problem at hand; for a more complete analysis, there are several resources available - for instance [27].

In general, any Newtonian fluid, including carbon dioxide, obeys to the well-known Navier-Stokes equations. They consist of a system of four equations - continuity, momentum, energy, coupled with the fluid Equation of State. A direct implementation of their full numerical solution, as carried out by a CFD code, goes far beyond the scopes of the present work.

In fact, compressible flows in a nozzle can be subjected to some important simplifying hypothesis.

First of all, we can limit ourselves to the analysis of a steady-state problem, if we neglect any initial transition. Subsequently, the problem can be considered one-dimensional, if we assume that the cross-section has a regular shape and all boundary layers have negligible effects on the flow. This, depending on the flow viscosity, can be fairly limiting - especially as far as the boundary layers on the nozzle converging-diverging wall are concerned.

That notwithstanding, under these assumptions we can say that the only variation to the flow initial conditions is due to the section change and expansion ratio, and this condition is defined as *quasi-monodimensional*. Mathematically, this means that all flow related quantities are a function of a single parameter - axial coordinate x .

Now, considering the absence of sources of any kind, mass flow is obviously a constant all across the nozzle (this is but a reduction of the continuity equation):

$$\dot{m} = \rho V A = \text{const}$$

At this point, considering that we have removed any source for dissipation and thermodynamic loss, if we can assume the walls to be adiabatic (which is generally the case),

the flow can be considered to be *isentropic*. Entropy is constant and is a function only of the flow total conditions ($s = s(T_t, P_t)$), which are defined to be the conditions where the flow velocity is null (such as in the storage tank). Under these assumptions the energy equation is reduced to the following:

$$h_t = h(P, T) + \frac{V^2}{2} = \text{const}$$

Which states that total enthalpy is constant. From this definition it is intuitive that $h_t = h(T_t, P_t)$ and is thus defined by initial conditions.

Concerning the momentum equation, under the current hypothesis it can be reduced to the following:

$$\rho V dV = -dP$$

It can be combined with the continuity equation (in differential form) and the isentropic expansion law to yield this famous result:

$$(1 - Ma^2) \frac{dV}{V} = \frac{dA}{A}$$

Where $Ma = v/a$ represents the Mach number, the ratio between flow velocity and sound speed in the substance. This equation explains the basic flow behaviour in De Laval nozzles; namely, it depends on whether it is supersonic ($Ma > 1$) or subsonic ($Ma < 1$). In the first case, the flow will accelerate in diverging sections (which is naturally counter-intuitive). In general, any sonic transition ($Ma = 1$) implies the existence of a throat (i.e. a region where $dA/A = 0$); which equates to saying that any supersonic flow needs a converging-diverging duct of some shape to be established.

In general, depending on the pressure outlet value, the flow can show three different possible behaviours.

If the outlet pressure is high enough, the flow never reaches sonic conditions at the throat. In this case, the flow slows down in the divergent and shows incompressible behaviour (an incompressible medium theoretically has infinite sound speed).

If pressure is low enough, the flow can reach sonic conditions and expand freely in the divergent, accelerating all along the axis, behaving fully supersonically. This was deemed to be the condition of interest for the present case, as all expansions in the experimental reference showed a continuous pressure decrease trend.

In intermediate conditions, the flow tends to accelerate, but at the outlet pressure is simply too high - this brings about shock waves, which disrupt the continuity of all flow fields. In this case, however, the hypothesis of isentropic flow fails - and, aside from being rather complex, is not of interest for the algorithm development, which is focused on the present problem (it is, nonetheless, a relevant future development).

Under all of these assumptions, a simplified MATLAB code for the solution of this problem was developed. Its most important features are the generalised thermodynamic treatment, by virtue of using a thermodynamic library, and the inclusion of a HEM condensation model, to allow for its direct comparison to the CFD solution.

The Span-Wagner Equation of State was used for both liquid and gas properties, as all

thermodynamic related calls were directly handled by REFPROP (via `refpropm`, as in Chapter 3).

4.2 Numerical implementation

Considering that in isentropic flows the pressure and temperature field are strictly connected (as $T = T(P, s)$ and $P = P(T, s)$), while velocity can be easily found from the total enthalpy equation, the problem can be numerically formulated as the search for a single temperature (or pressure) value. Furthermore, we want the flow to behave subsonically in the convergent, and supersonically in the divergent. This further simplifies the problem at hand.

The code input is the flow total conditions (and consequently its entropy and total enthalpy), alongside the nozzle shape.

Theoretically, infinite solutions are possible starting from a single set of total conditions; another boundary condition is required. This could (and rigorously should) be imposed in the form of outlet pressure. However, in our conditions we are assuming the flow to be isentropic and therefore naturally expand in the divergent without any type of discontinuity (i.e. shock). Thus, the numerical pressure outlet value is meaningless, and the simplest choice for boundary conditions therefore takes the form of the imposition of sonic conditions in the throat.

Practically, this was obtained with a function which takes as input the total conditions and the nozzle throat area, and returns the only thermodynamic state that allows for $Ma = 1$ in that section with a simple brute force approach (calculating every possible thermodynamic solution from triple point to total conditions). Subsequently, mass flow is calculated (from throat conditions); it will be kept as constant all along the solution.

At this point, the algorithm proceeds to divide the nozzle into arbitrarily small points, and then find the adequate temperature at each section; this is done separately for the convergent (proceeding backwards, from the throat) and the divergent (moving forward). In each section two solutions - subsonic and supersonic - are still possible, but we force *a priori* the search in one direction thus finding only the solution of interest (at higher temperature in the convergent, and lower in the divergent). The search proceeds iteratively; at each step (denoted by subscript i), these three operations are performed:

$$\begin{aligned} V_i &= 2\sqrt{(h_t - h_i(T))} \\ \rho_i(T) V_i A &= \dot{m}_i \\ \Delta &= \left| \dot{m} - \dot{m}_i \right| \end{aligned}$$

The momentum equation is not needed in any form simply because we are forcing, as input, the search direction for temperature and therefore the condition (subsonic, or supersonic) of the flow.

Density and enthalpy are directly called from temperature and constant entropy; subsequently, velocity is calculated, and with it, mass flow. If at any given iteration the mass flow residual (Δ) is found to be decreasing with respect to the previous iteration,

the search goes on, changing temperature of a fixed amount (which is in general small enough), until the minimising solution is found.

Consequently, the process starts anew in the following section; the starting search value always comes from the adjacent section (for the first section in both the convergent and divergent, it comes from the throat).

At the end of the iterative process, the solution is properly composed, and from temperature and (constant) entropy all flow quantities (pressure, velocity, sound speed, Mach number) can be calculated.

Obviously, this approach to the problem solution is limited; it can fail to accurately represent large changes (i.e. around the throat) and it strictly depends on REFPROP and the availability of its properties. In fact, around the triple point, or if the flow crosses the spinodal line, the properties simply can no longer be retrieved and the calculation usually fails. This notwithstanding, the results for the low total pressure expansions were found to be fairly accurate.

4.2.1 Condensation model

A version of the HEM model (as described in Chapter 2) was developed and implemented in the MATLAB code that was discussed above.

In general, at constant entropy, under the model assumptions, the condensation onset will be determined by a single thermodynamic quantity falling below a given threshold, representing its intersection with the chosen transition line (in general, any line between saturation and spinodal).

The problem can once again be formulated in terms of a single independent variable. In fact, at a given temperature, the assumption of constant entropy can be used to find the mass fractions of each phase and with them any desired property. Inverting Equation 2.1:

$$\alpha_v = \frac{s - s_l}{s_v - s_l}$$

Where the subscripts v, l represent the transition vapour and liquid quantities. Once mass or volume fractions are known, the transition quantities are obtained by forcing REFPROP search in either of the two phases at a single $P - T$ input. Once this is done, the thermodynamic state is known, and the process can go on as specified earlier.

A problem arises in the definition of the sound speed for a two-phase mixture. This property is not supplied by REFPROP; this formula was found in literature [8]:

$$a = \sqrt{\frac{1}{(d_l \alpha_l + d_v \alpha_v) * (\frac{\alpha_l}{d_l a_l^2} + \frac{\alpha_v}{d_v a_v^2})}} \quad (4.1)$$

It gives continuous results for the mixture sound speed with respect to that of the single phase, similarly to what happens for the other thermodynamic quantities when they cross the transition line.

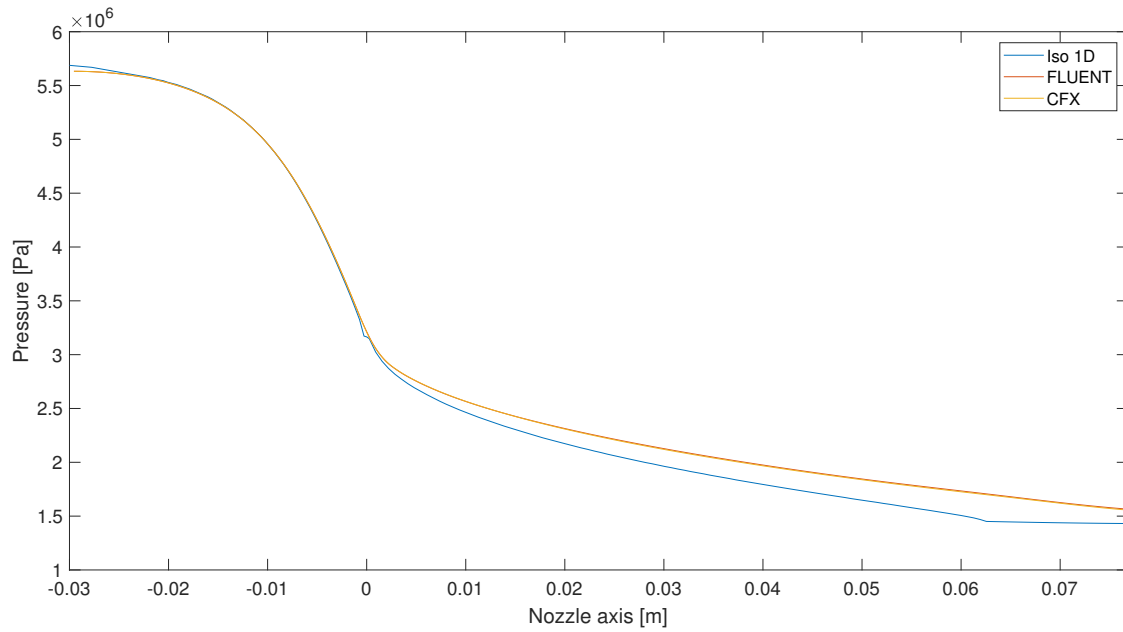


FIGURE 4.1: Single-phase pressure comparison with CFD codes

This formula was clearly applied both for post-processing calculation of sound speed and Mach number and for the throat initialisation process.

As a last remark, to ensure that the algorithm always reaches a solution and remove a potential error cause, the HEM model was always employed in case the spinodal line was crossed; as already discussed, below it single-phase properties would not be available from REFPROP.

In general, results were found to be quite accurate far from the critical point and the condensation onset prediction was determined to be coherent with the CFD forecasts. More detail is provided below and in Chapter 6.

4.3 Algorithm validation

The expansion at lowest pressure (case 1-1 from Table 2.2; $P_t = 57.24 \text{ bar}$ and $T_t = 310 \text{ K}$) as tested by Lettieri was evaluated separately by the algorithm above described (referred to as *Iso 1D*), CFX and Fluent (full detail on the simulations is provided in Chapter 5). At first, all condensation models were turned off (the 1D code employed the spinodal line as transition; but it is never crossed). All results were evaluated on the nozzle axis, and are substantially coherent between themselves.

Three main differences can be highlighted from Figure 4.1.

Firstly, near the nozzle the quasi-1D solution is not influenced by the imposition of boundary conditions, and thus reaches, unimpeded, higher values, while the CFD solution slightly underpredicts the expansion rate (and is more adherent to experimental values, as it will be shown in Chapter 6).

In the immediate proximity of the nozzle throat, the chosen approach for the boundary condition causes some local instability for the numerical solution, due to the imprecision of the (rough) minimisation process that was described above.

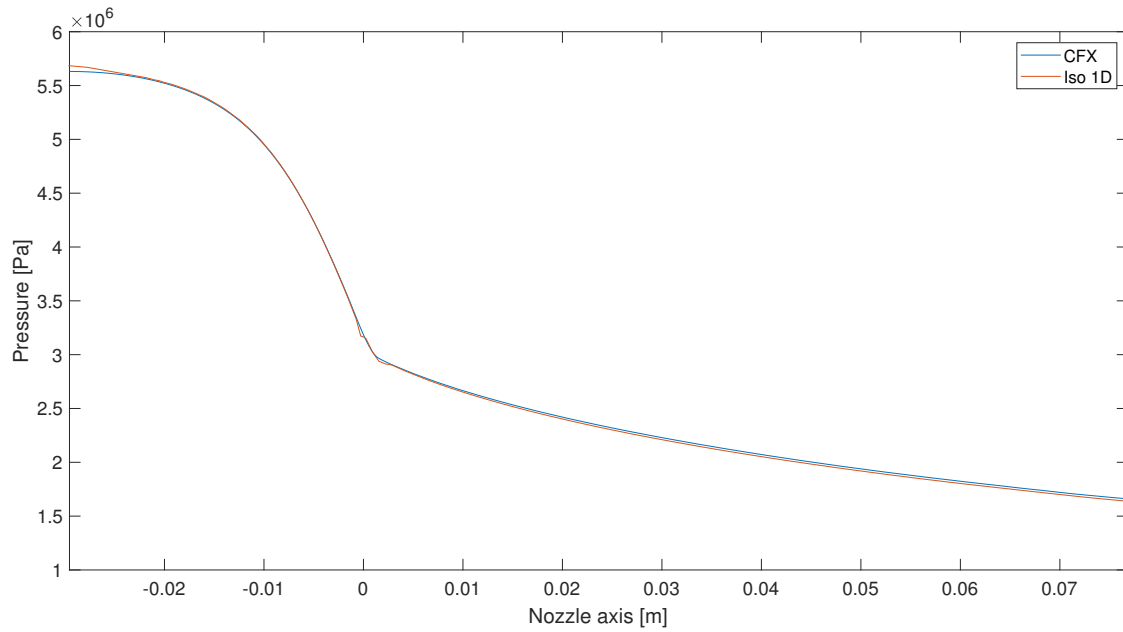


FIGURE 4.2: Two-phase pressure comparison with CFX, which employs the HEM

In the divergent, the largest difference between the solutions is due to the absence of any type of viscous effect in the quasi-1D solution. In fact, these phenomenons tend to block a part of the cross-section, due to the displacement thickness of the boundary layer. The consequence is a reduction of the actual flow section that can proceed at full speed. Therefore, the expansion rate corresponding to the isentropic supersonic solution is reduced.

An obvious way to assess the difference between 1D and CFD models is to perform a CFD simulation without viscous effects, namely specifying slip condition on the nozzle wall. However, CFD simulations with this type of boundary condition diverged and thus a direct comparison could not be obtained. In fact, such inviscid solution brings the fluid to low pressures and temperatures, up to reaching the triple point, since this specific isentropic expansion is bound to never cross the spinodal (as shown in Figure 2.3). In those thermodynamic regions, the REFPROP formulation becomes unstable (and eventually fails), and the algorithm returns semi-constant values.

Subsequently, a comparison between the condensation models is carried out, to see whether our algorithm yields consistent results compared to the reference HEM (as implemented in CFX). The transition line was chosen to be the Wilson line. Figure 4.2 and 4.3 detail the results in terms of pressure and liquid mass fractions. In this case, the CFD simulation converged even with a slip boundary condition for the wall, as the condensation causes an increase in pressure and temperature and brings the flow far enough from the triple point area.

The results are surprisingly accurate; firstly, the consistence in pressure results in the divergent does confirm our earlier hypothesis that the main differences in the single-phase simulations highlighted in Figure 4.1 were due to the boundary layer presence. That notwithstanding, small differences persist in that area, probably due to the bidimensional flow effects which, although small, are still present. Maybe most importantly,

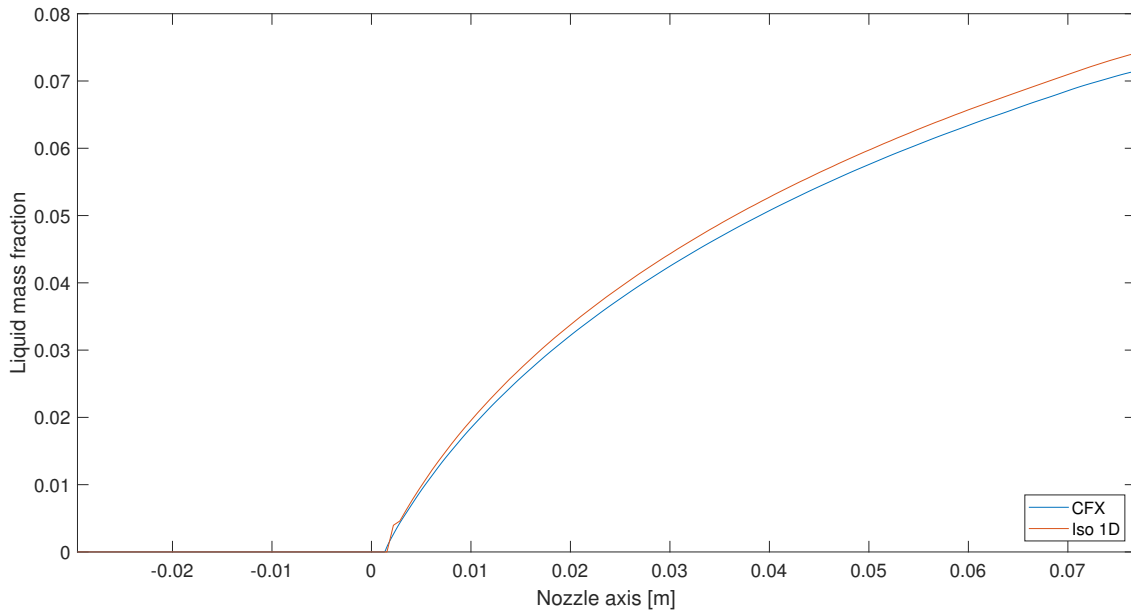


FIGURE 4.3: Two-phase liquid mass fraction comparison with CFX, which employs the HEM

CFX employs a different formula for the calculation of the two-phase sound speed, as detailed in Chapter 6.

The same considerations concerning the throat and inlet regions are still valid.

The liquid mass fraction trend has a similar behaviour; in particular, it is worth mentioning that the prediction of the condensation onset is remarkably consistent between the two solutions, owing to the fact that the thermodynamic flow conditions are similar and therefore the Wilson line intersection is bound to happen in the same region. Quantitatively, the predicted fractions are consequently similar, owing to the analogous condensation model implemented.

Temperature and other thermodynamic quantities show a similar behaviour, owing to the flow constant entropy.

4.4 Algorithm limitations

The underlying numerical mechanism used to find the solution relies on the search of a minimisation temperature that allows for the flow to reach the goal mass flow, which is in turn calculated by the imposition of sonic conditions in the throat. The relation between mass flow residuals and temperature in a given section shows two minimum values, corresponding to subsonic and supersonic solution. The desired solution is found by brute force (lower temperature for supersonic, higher temperature for subsonic).

There are at least two types of bounds for the temperature span that can prevent it from reaching the desired minimising value. Firstly, at constant entropy, temperature is limited by the spinodal line - there exists a lower limit below which it cannot go. If entropy is too high to intersect the spinodal, the triple point (and its proximity) serves the same purpose. This phenomenon was highlighted in the analysis of Figure 4.1.

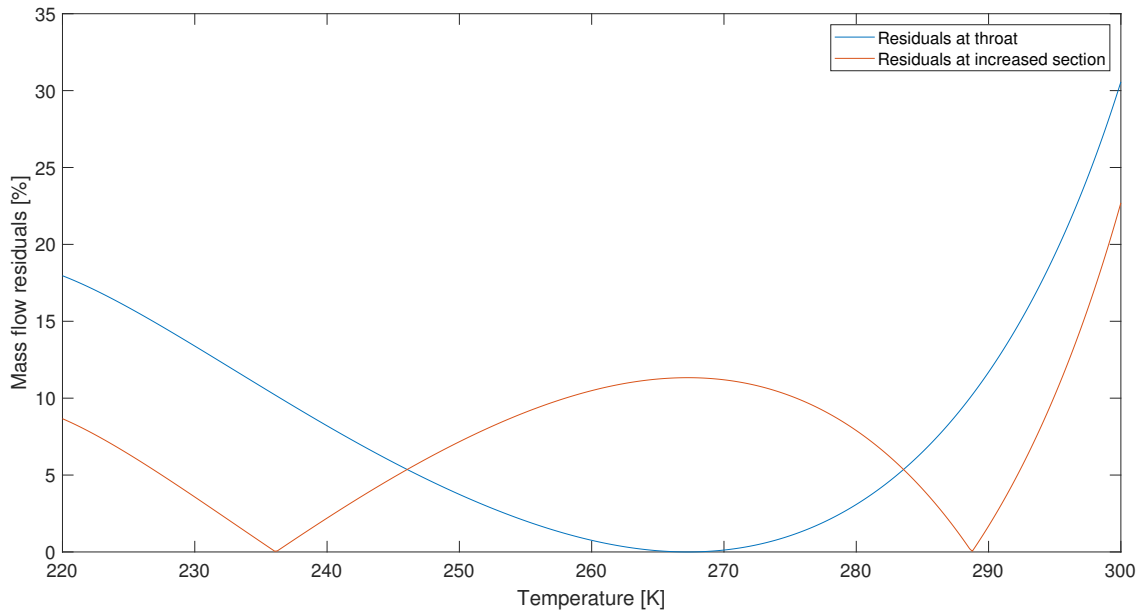


FIGURE 4.4: Residuals for Case 1-1 (normalised with mass flow value)

If the flow is in two-phase states, similarly, an upper bound is constituted by the intersection with the transition line.

At the throat, however, there exists a single minimisation point, owing to the fact that we are imposing sonic conditions. This phenomenon is highlighted in Figure 4.4 for the lowest pressure expansion (Case 1-1 from Table 2.2).

Now, when the inlet total pressure is increased, the throat conditions may slip into the two-phase area. For instance, in case 3-1 ($P_t = 73.24 \text{ bar}$), the throat will be in two-phase conditions if transition is initiated at the Wilson line, but single-phase metastable vapour if transition is started at the spinodal. The two solutions will be numerically different; but there is a subtler difference, which regards the chosen definition for the two-phase sound speed (Equation 4.1). The formula, in fact, allows for the mixture sound speed to have a radically different behaviour from that of the pure substance, as shown in Figure 4.5; for the mixture, the sound speed actually increases rather than decreasing at decreasing temperature. This is necessarily reflected onto the Mach number.

An important consequence that invalidates our chosen approach arises. If the throat is in two-phase conditions, a solution at increasing temperature must be searched in the convergent. However, at increasing temperature, the sound speed is no longer increasing as in single-phase conditions; therefore, the Mach number may not necessarily decrease and no stable subsonic solution may be found. In fact, the two-phase residuals (Figure 4.7) show exactly this behaviour - if the section is increased, the only stable solution is found at a lower temperature - therefore the flow can only behave supersonically, and no transition to single-phase vapour is possible. This is compared to the behaviour of residuals for the single-phase solution (Figure 4.6), which shows the standard behaviour that was already highlighted for case 1-1 (Figure 4.4).

It is still worth mentioning that the higher gradients and generally more complicated thermodynamic conditions - due to the proximity of the critical point and higher thermodynamic gradients - already start to invalidate our solution process (which relies on

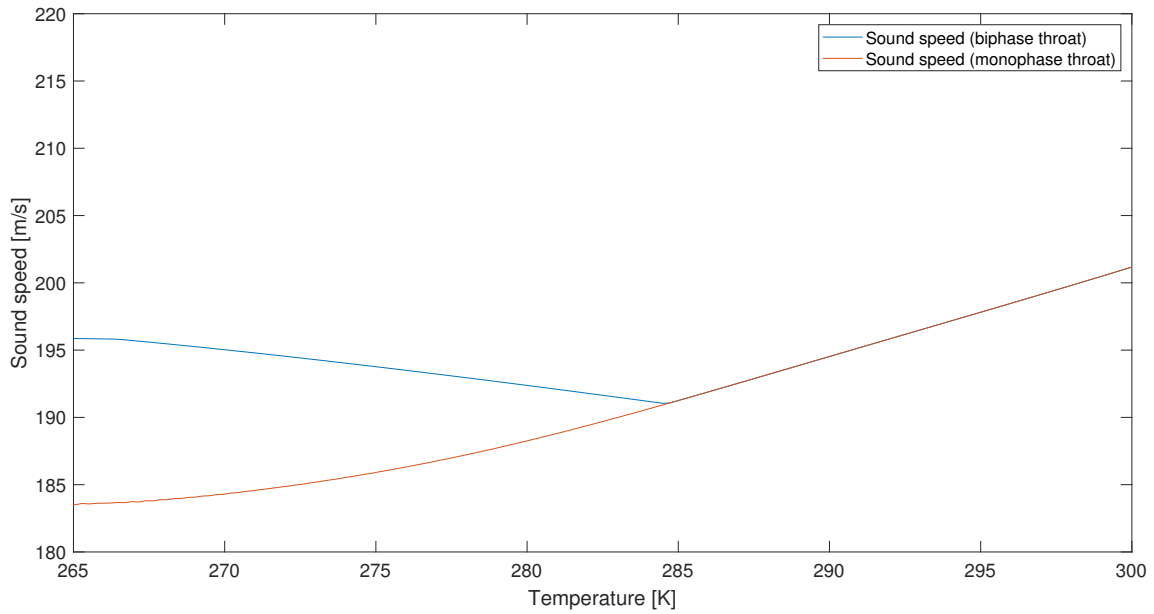


FIGURE 4.5: Sound speed for single-phase and two-phase solutions

properties showing a clear trend as temperature increases or decreases), as shown by the irregular behaviour of the residuals.

A possible solution to this issue would require finding a new expression for two phase sound speed. However, even if the topic has been researched in literature [5], with much more depth and with the actual solution of 1D Navier-Stokes equations, there is little or no material involving the solution with our simplified approach. It was also determined that the two-phase model and its actual behaviour was of little interest for the higher pressure cases, aside from detecting an eventual condensation onset; this can also be determined in post-processing from a single-phase solution (in this framework, a solution that transitions at the spinodal line). Moreover, the solution process itself starts to fail and eventually becomes unstable. From Case 4-1, in fact, the flow in the throat lies below the spinodal, therefore for the flow properties to exist a two-phase state should be initialised. However, no solution could be obtained.

Clearly, the solution process detailed in this chapter does not replicate with satisfying accuracy the physical phenomenon; convergence failure is thought to be due to the HEM (as detailed in more depth in Chapter 6) and the strict assumption of isentropic and inviscid flow. That notwithstanding, this code was useful for providing a reference benchmark in Case 1-1 - the one that will be most extensively analysed, by setting a minimum accuracy that CFD would need to reach (therefore verifying the correct implementation of the S-W Equation and the condensation options), and assessing some flow characteristics, based on the 1D behaviour (most importantly, the prevalence of boundary layers, and what underlying issues for convergence may sussist).

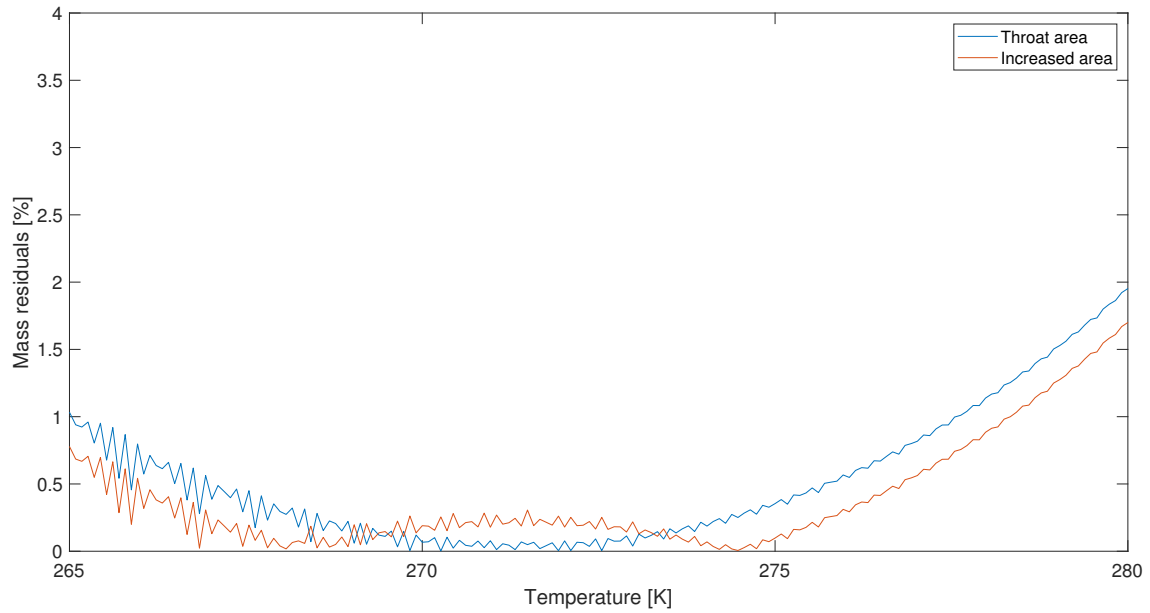


FIGURE 4.6: Residuals (normalised with mass flow value) for single-phase solution for Case 3-1

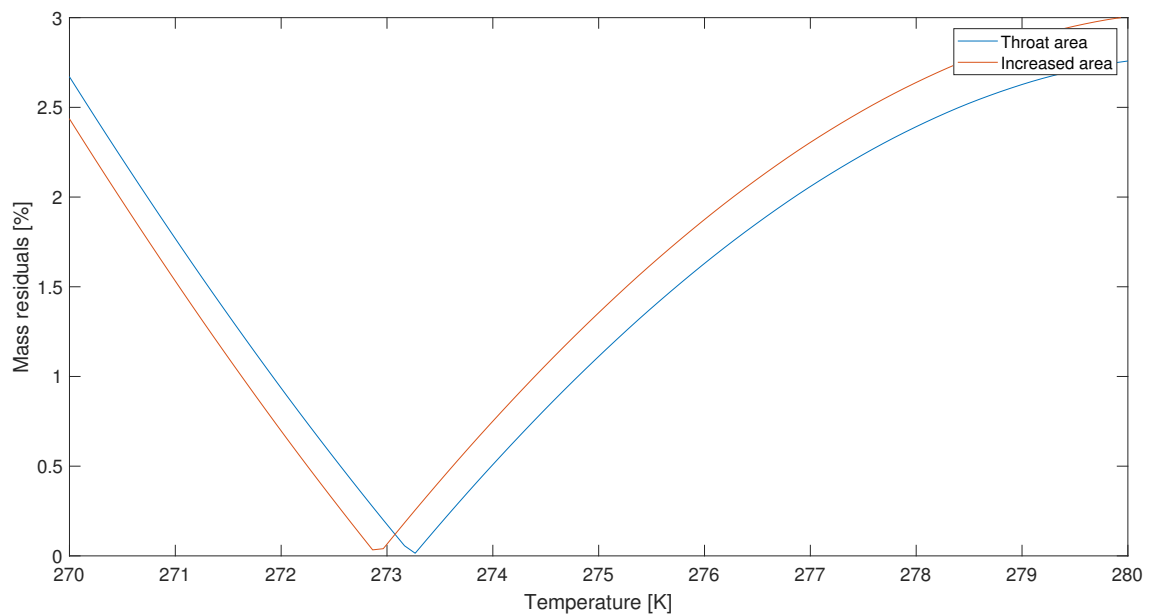


FIGURE 4.7: Residuals (normalised with mass flow value) for two-phase solution for Case 3-1

Chapter 5

CFD Setup

In this chapter all other aspects regarding the CFD simulations will be discussed; namely, geometry, mesh, numerical methods, turbulence model. A mesh independence and a lookup table size analysis will be carried out to determine their optimal value. Finally, the Lee Model coefficient influence will be discussed in ANSYS Fluent.

5.1 Geometry model

The domain of interest is fully bidimensional; the experimental reference determined that the cross-section boundary layers are completely negligible. Not only; the domain can be halved, taking advantage of its symmetry, and only half the nozzle section can therefore be simulated.

A known limitation of ANSYS CFX is its inability to handle bidimensional simulations; as a consequence, a 3D model (shown in Figure 5.1) had to be created, with a rectangular cross-section (arbitrary thickness of 1 mm). The nozzle section was generated in Autodesk Inventor by importing, in .xls format, the $X - Y$ coordinates of an array of points¹ representing the wall profile, subsequently joined by means of linear interpolation. Then, the model was saved in the standard .stp format, and finally imported in ANSYS Workbench for meshing operations, via the Workbench own CAD software, SpaceClaim. There, the 2D section simulated in Fluent was extracted.

All simulations included both convergent and divergent sections of the nozzle. No result could be obtained otherwise.

The geometry defines the throat as reference for the coordinate system ($x = 0$), with the convergent represented by negative values and the divergent by positive (following the flow direction). The symmetry line is assumed to be the horizontal axis ($y = 0$). The same convention will be reported in all graphs discussing the simulation results.

¹Informations complemented by Dr. Ir. M. Pini, T.U. Delft.

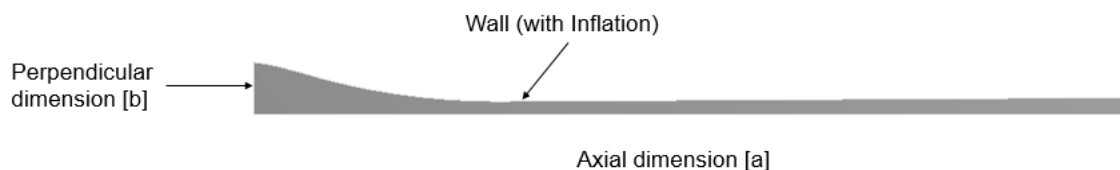


FIGURE 5.1: Geometry model

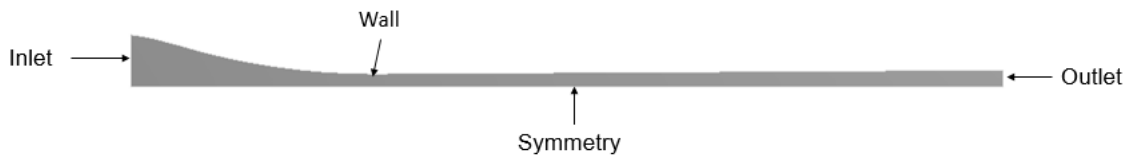


FIGURE 5.2: Boundary conditions

5.2 Boundary conditions

Four boundary conditions are necessary (plus another two for the 3D model). These are shown in Figure 5.2. Clearly, symmetry is needed on the lower side of the model to replicate the results for the other half section; the nozzle is then bound by no-slip adiabatic smooth (no roughness was considered) wall, inlet (of the *pressure inlet* type in ANSYS Fluent) and outlet (*pressure outlet* in Fluent) conditions. Total pressure and total temperature coming from the experimental reference and flow direction normal to boundary were assigned here.

The two faces that emerge in the 3D model were assigned a Free Slip adiabatic boundary condition type in CFX, so that they would not cause any dissipation, but would also prevent any type of 3D flow effect from arising.

More specifically, concerning the numerical values to be assigned to the inlet condition, it was determined that the total pressure should come from the experimental tank values (Table 2.2), whereas total temperature was to be assigned from conceptual conditions (Table 2.1).

Let us start from the first choice. The ideal total pressure values were simply slightly incorrect compared to the values coming from the pressure probes, as shown in Figure 5.3 (which details two simulations carried out at same total temperature, 310 K, but varying total pressure). Even if the conceptual total pressure value seems to obtain a slightly better adherence with experimental results, the value at the first probe - the one closest to the actual boundary condition - clearly shows that the tank total pressure values are to be chosen.

The choice of total temperature is slightly subtler, as no experimental data for that thermodynamic variable are provided. However, considering that the expansions are (mostly) isentropic, the experimental Wilson points provided by the reference should be (approximately) at the same entropy as the total conditions. The greatest thermodynamic loss, in fact, comes from condensation itself, which brings the fluid back to equilibrium conditions, and should therefore not influence the thermodynamic location of the Wilson points, which represent the phase change onset.

Consequently, a comparison of the Wilson points locations with all possible total conditions of interest (i.e. the ones at total pressures shown in Table 2.2 and total temperatures from both Table 2.1 and Table 2.2) was analysed. Results are shown in Figure 5.4, where the Wilson line was generated as explained in Chapter 2.

It is clear that no boundary condition perfectly matches the desired entropy. However, the conceptual total temperature values are closer. No explanation is immediately available for this behaviour; it is supposed that between the tank and the actual nozzle inlet a limited cooling takes place, bringing temperature down to values close to the ones referred to in Table 2.1. This also explains why total temperature in the tank was set at higher values to begin with.

At the outlet, a reference pressure was imposed as well, as required by the CFD software - though it is not supposed to affect the supersonic flow behaviour, as briefly explained in Chapter 4. The value was set to be low enough not to generate shockwaves. Actually, the two programs treat the Outlet boundary condition differently; namely, Fluent can recognise if the flow is locally supersonic and, if so, ignores the set value. CFX, instead, 'hard' imposes the value. As a consequence, if the flow itself is at lower pressures, this generates a local discontinuity, namely a shock wave in the outflow boundary that ultimately does not have a major impact on the flow. In general, outlet pressure values will not be reported unless of specific interest.

Finally, in case of multiphase simulations, the inlet fraction of vapour was defined to be 1 (which means that the liquid fraction is 0, i.e. the flow is single-phase vapour as expected at the nozzle inlet).

5.3 Meshing process

Though there are some slight differences between the meshing process for the 2D and the 3D model, the largest part of the operations was similar, even if the end results did not match completely.

All meshing operations were carried out in ANSYS Meshing (in the framework of ANSYS Workbench).

The chosen meshing method was *Multizone* for its ability to produce a regular mesh in a simple geometry. At this point, there are three choices of interest for the meshing process; axial sizing (dimension a), perpendicular sizing (dimension b) and the boundary layer placed at the nozzle wall, which was generated by means of an *Inflation* feature. Among all these features, the most important element was determined to be the axial sizing, considering that it meshed directly the symmetry line and therefore captured the pressure profile development on the axis - which is the most important characteristic of a nozzle blowdown and the one which is to be compared directly with the experimental reference.

The thickness for the 3D model was meshed with 2 elements (therefore each 0.5 mm long).

As far as the inflation process was concerned, it was determined that the thickness of the first element would be chosen; this was done in order to control the y^+ value, which accounts for the mesh quality in the boundary layer area, and namely how well the turbulence model can capture its development. Ideally, this should be independent from the axial or perpendicular sizing; however, if there is too much disparity between these values, the elements skewness (i.e. the geometrical ratio between the two sides of the element) becomes too high and meshing fails. Therefore, a balance had to be sought. The number of layers was determined with this formula (and euristically adjusted if needed):

$$N_l = \log_{\text{growth rate}} \left(\frac{\text{Normal Element thickness}}{\text{First element thickness}} \right)$$

The growth rate - the ratio between the thickness of each subsequent layer of elements - was kept at the standard value of 1.2. Normal elements were those assumed to be uninfluenced by the Inflation feature and therefore far from the wall.

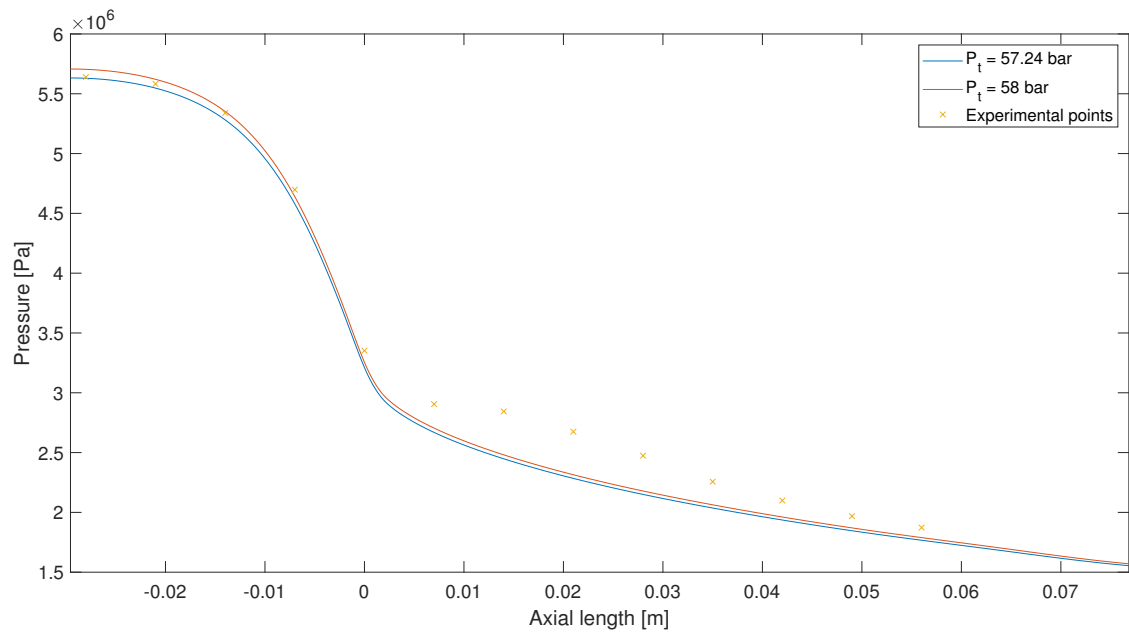


FIGURE 5.3: Pressure in case 1-1 on the symmetry line for different inlet total pressure

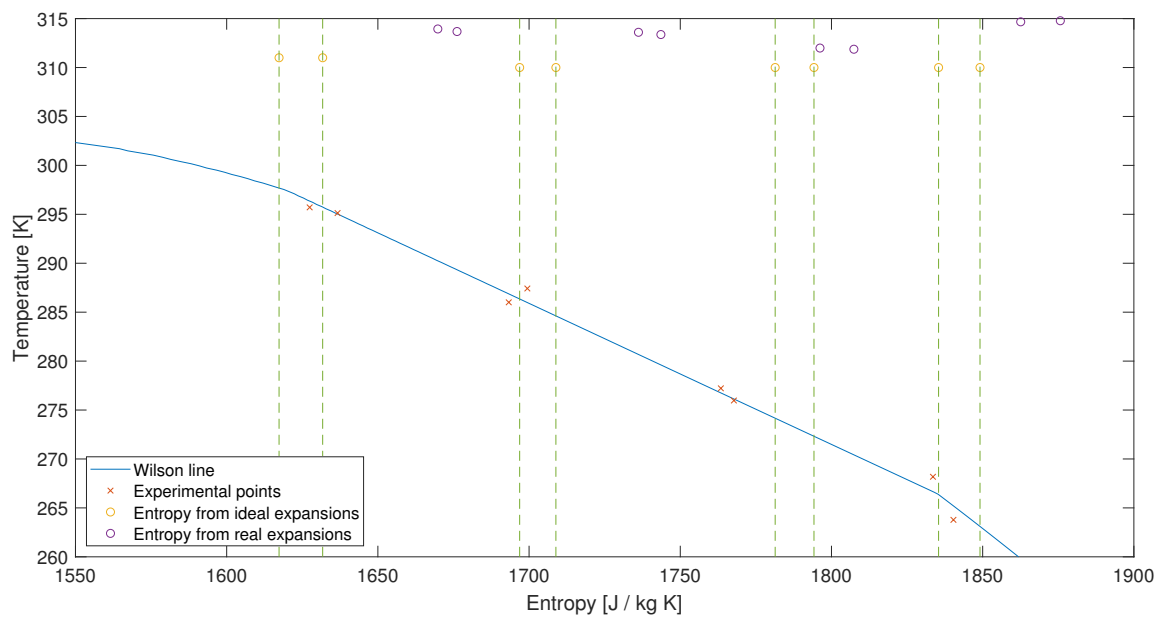


FIGURE 5.4: Total temperature comparison with the reference

In the 2D model, the *Face Meshing* option was chosen, alongside the *Multizone Quad / Tri* method, in order to replicate the obtained mesh for the 3D case. The mesh data are reported in Table 5.1; similar, but not exactly equal, results (with doubled elements, accounting for the thickness) were achieved for the two models, due to the different meshing methods and options available in the software for the different situations.

Three meshes were generated for each model, in order to determine which size could obtain the most efficient, computationally speaking, results, reported in Table 5.1.

Mesh	a size [m]	b size [m]	2D Elements	3D Elements
1	$5e^{-4}$	$5e^{-4}$	4.708	8.132
2	$1e^{-4}$	$1e^{-4}$	86.589	162.488
3 (2D)	$2.5e^{-5}$	$1e^{-4}$	342.000	-
3 (3D)	$5e^{-5}$	$1e^{-4}$	-	323.456

TABLE 5.1: Mesh data

Mesh	First inflation layer thickness	Number of inflation layers
1	$1e^{-5}$	10
2 (2D)	$1e^{-6}$	20
2 (3D)	$1e^{-6}$	13
3 (2D)	$1e^{-6}$	15
3 (3D)	$1e^{-6}$	13

TABLE 5.2: Inflation feature data

As highlighted by Table 5.1, at first the finest mesh was supposed to achieve an axial spacing of $2.5e^{-5}$ m. However, the 3D model showed convergence issues with that mesh. Also, the number of elements was deemed to be quite high, and in order to limit computational requirements a slightly less fine mesh was generated. There were some slight issues with the Inflation feature as well, that caused a difference in the number of layers between the two models.

Therefore, it was deemed necessary to perform an independence analysis for both the 3D model (in CFX) and the 2D one (in Fluent).

5.3.1 Mesh independence analysis

All tests were carried out with single-phase simulations, for Case 1-1; no condensation models were involved, but the fluid was set to be a real gas. Results for Fluent are shown in Figure 5.5 and 5.6.

Firstly, it is evident that the differences between the results are not remarkable and do not account for major discrepancies compared to the experimental results. Despite the computational cost increasing sensibly (though it is still overall moderate), results did not improve remarkably for Mesh 3, suggesting that, as long as convergence can be achieved, the mesh choice and its refinement are not fundamental. That notwithstanding, though Mesh 1 achieved good results, it gave higher errors than the probe accuracy of 0.1 bar, and was therefore excluded.

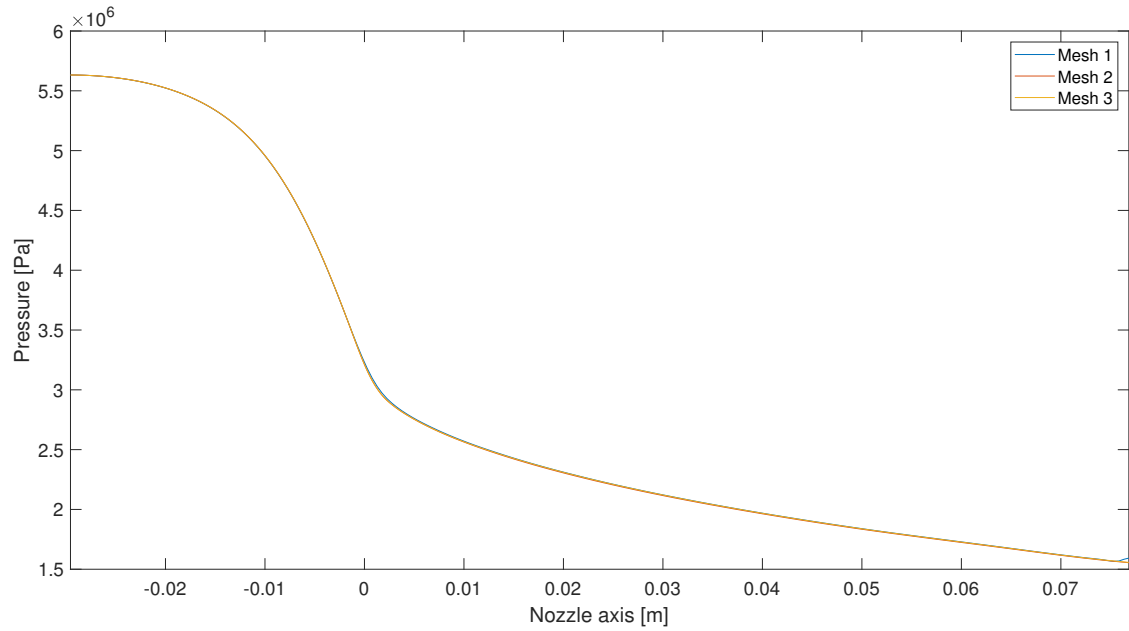


FIGURE 5.5: Pressure in Case 1-1 on the symmetry line for different 2D meshes (Fluent)

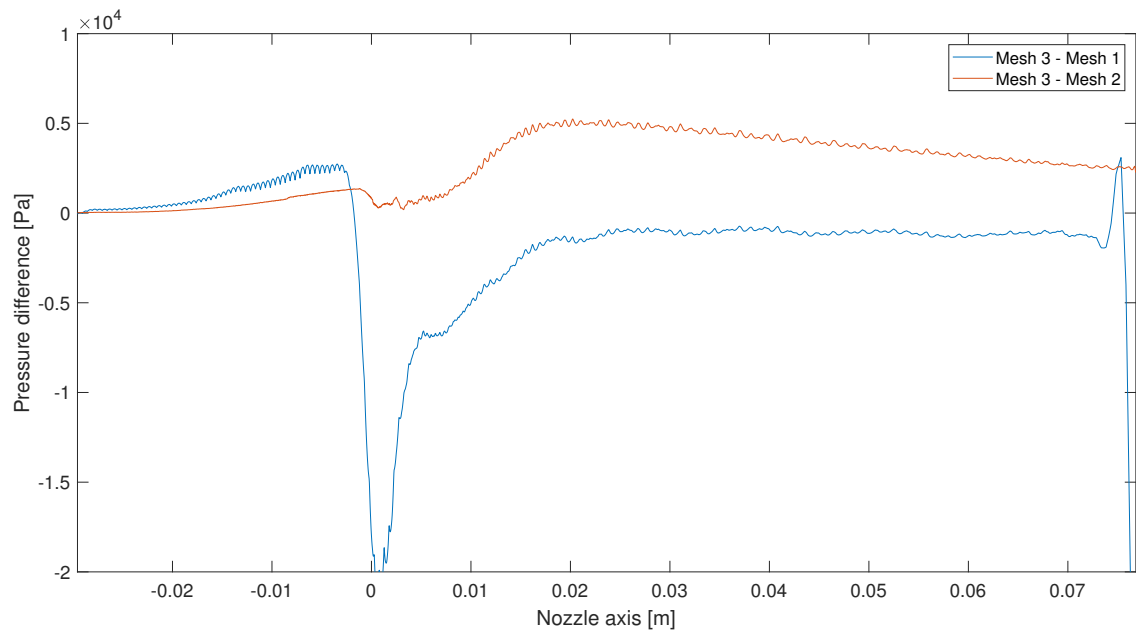


FIGURE 5.6: Difference between the pressures for Case 1-1 on different 2D meshes (Fluent)

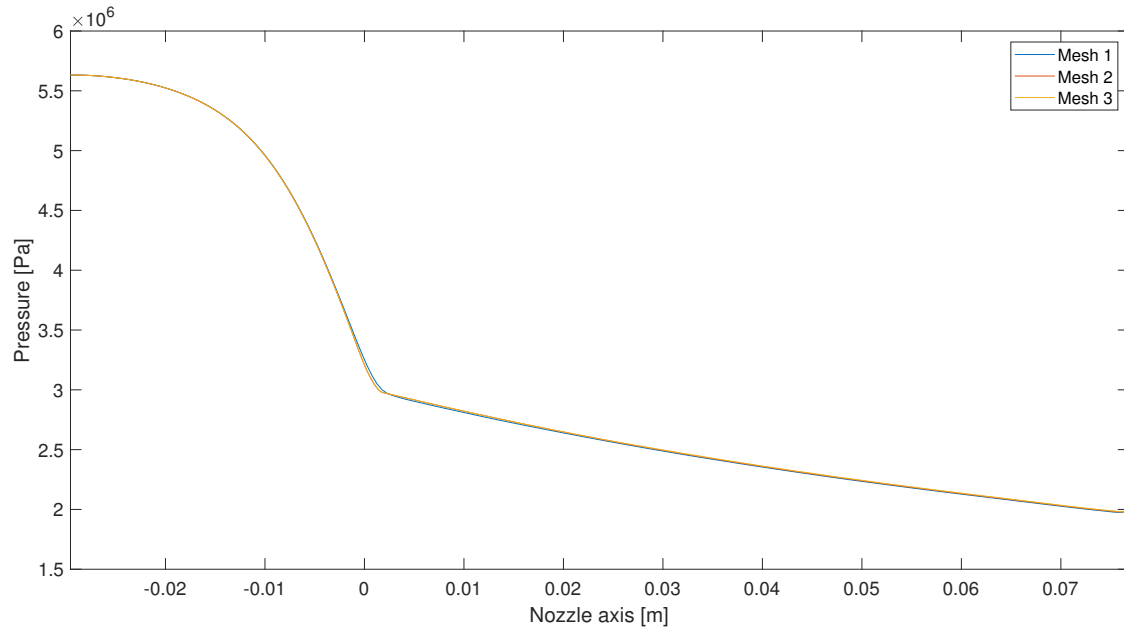


FIGURE 5.7: Pressure in Case 1-1 on the symmetry line for different 3D meshes (CFX)

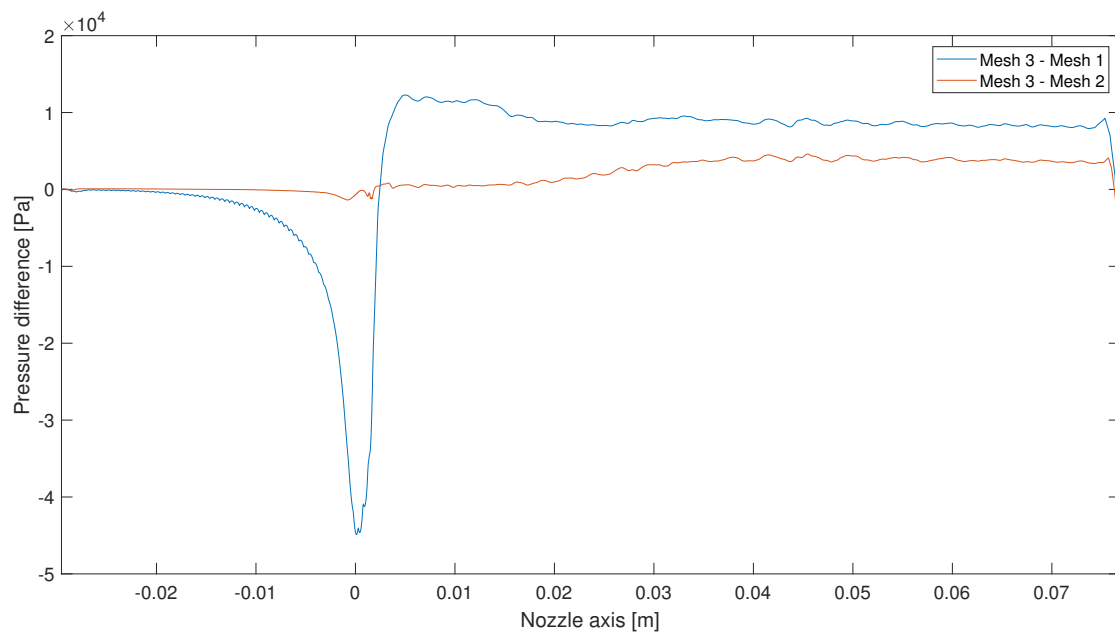


FIGURE 5.8: Difference between the pressures for Case 1-1 on different 3D meshes (CFX)

Consequently, Mesh 2 was chosen for all simulations representing Case 1-1 and 1-2, which was most extensively analysed to provide insight into the software behaviour and the simulation accuracy, and tried for all other situations.

Furthermore, Mesh 2 is consistent with the mesh reported by Hosangadi [16], who obtained results on a similar mesh for Case 1. He had to reduce spacing up to $2.5e^{-6} m$ (40 times compared to Mesh 2) in order to achieve convergence for higher total pressure cases, however.

Results for CFX are shown in Figure 5.7 and 5.8. They were obtained with single-phase simulations employing lookup tables clipped at the Wilson line, because convergence with spinodal LuTs proved harder to achieve. The reason for this behaviour is not immediately clear, but it is believed to be caused by real gas effects becoming more prevalent in deep metastable conditions ultimately influencing the solver stability on finer meshes.

Results, once again, show that there is no dramatic improvement if mesh is refined excessively; similarly as above, Mesh 2 is chosen. Mesh 1 does exhibit a somewhat irregular behaviour; the (relatively) large difference around the throat is probably due to the Wilson line crossing happening in a slightly different location, which has a heavy influence on the solution behaviour. This is ultimately due to the Lookup Table type, and will be discussed in more depth in Chapter 6.

Y+ analysis

The values for the Y+ are reported in Figure 5.9 for simulations in both CFX and Fluent, carried out with Mesh 2 as detailed above. They are rather high, and they do not allow to properly simulate the near-wall region of the boundary layer. Nonetheless, it was determined that the flow behaviour in that region was not of primary interest for the simulation at hand. Further mesh refinements that brought the value close to 1 did not cause remarkable variations for pressure on the symmetry line, despite being computationally much heavier. Therefore, no further mesh variations were applied.

5.4 Numerical methods

This section details the choices relevant to turbulence and numerical methods employed for the solution of the Reynolds Averaged Navier Stokes (RANS) Equations.

Both software employ the finite volume method for the discretisation and solution of the equations; there are differences in their formulation and their structure. However, it was determined that these were not of primary interest, as converging solutions with the two programs employing the same thermodynamic and multiphase framework showed remarkably consistent behaviour, as will be discussed in more depth in Chapter 6. The main limitation, instead, was due to the availability and type of condensation models, as already discussed in Chapter 2, and the degree of flexibility allowed for by the solver. In both circumstances, Fluent proved to be superior, as will be once again shown in more depth in Chapter 6.

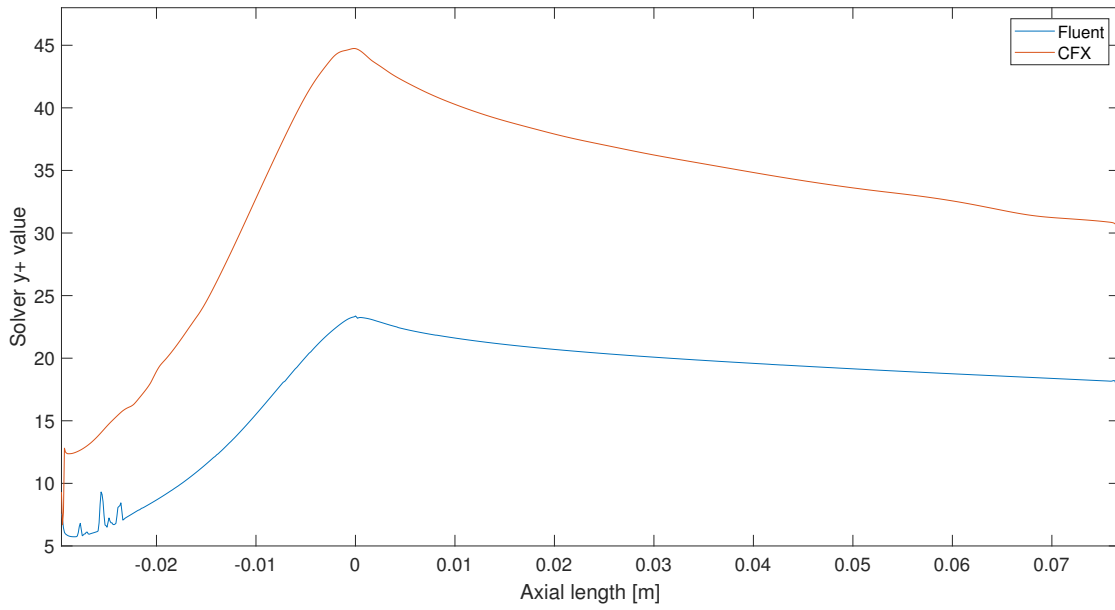


FIGURE 5.9: Y+ value

5.4.1 ANSYS CFX®

CFX [18] uses a pre-defined solver that applies a Coupled approach, i.e. solving all flow equations at the same time. Even for steady-state solutions, a time-marching solution strategy that guides the numerical process to convergence is always present.

Little choice is available in the software in terms of solver customisation; all results here reported, unless otherwise specified, were obtained employing the so-called *High Resolution* scheme for advection and both flow and turbulence terms in equations. If necessary, results were obtained first with the *Upwind* scheme, more robust, then used as basis for convergence with the more precise scheme.

The advection term, in general, represents the transport of a scalar quantity in a vector field. Several terms in the flow equations requires the knowledge of the flow field on the boundary of the element, rather than at its centroid (which is where they are available and the values that are actually stored and available in post-processing). This is obtained through the solution of this expression:

$$\phi_{ip} = \phi_{up} + \beta \nabla \phi \cdot \Delta \vec{r}$$

Where ϕ represents the variable of interest, r is the distance and β depends on the choice between numerical methods; it is 0 in the Upwind scheme and has a more complicated formulation in the High Resolution case, in order to represent more accurately the variable gradient inside the element.

Basically, in the Upwind scheme there is no difference between the flow variable value inside the cell or on its face.

The Total Energy model, with the inclusion of the Viscous Work Term, was chosen for all simulations. Viscous terms are discretised with second order differences.

Convergence was determined by the behaviour of the RMS residuals; the threshold was generally set when possible at $1e^{-6}$, though simulations at times exhibited converging behaviour at slightly higher values.

5.4.2 ANSYS Fluent®

Fluent [19] offers much more choice in terms of numerical methods and their control. The pressure-based solver was employed, as the density-based solver is not compatible with multiphase models (except for Wet Steam, but it was not deemed of interest for the present work). At first, a pressure-velocity coupling (SIMPLE) was employed for the simulations. However, during the course of the work, it was determined that it was not the best choice for achieving convergence. Instead, a Coupled scheme, involving the simultaneous solution of all flow equations, was employed.

All results reported here, unless differently specified, involve Second Order spatial discretisation schemes for all equations, except for the Volume Fraction, for which it is unavailable. Full details are reported in Table 5.3. Quite small underrelaxation factors were employed, especially for the flow Courant number, depending on the cases. As with CFX, when convergence could not be achieved immediately, a First Order solution was calculated first and subsequently used as basis for refinement. First Order and Second Order schemes loosely correspond to the definitions for Upwind and High Resolution in CFX.

Actually, when Multiphase models are enabled, Fluent suggests PRESTO! as Pressure Discretisation scheme (and disables First / Second Order choices), which is more peculiar and directly calculates the actual pressure values on the face cell from the discretisation of the differential equations, rather than from an expression of some sort starting from the cell center. Converging satisfying results could be obtained with both it and Second Order schemes depending on the simulation type.

No fixed criterion for convergence was imposed on the simulations; basically, simulations were left running until it was clear, by the residuals behaviour (i.e. flat line or fixed oscillation at low numerical values, certainly lower than $1e^{-3}$ for all equations and namely continuity), that convergence had been achieved.

Equation	Discretisation Scheme
Gradient	Least Squares Cell Based
Pressure	PRESTO! - Second Order
Density	Second Order Upwind
Momentum	Second Order Upwind
Volume Fraction	First Order Upwind
Turbulent Kinetic Energy	Second Order Upwind
Specific Dissipation Rate	Second Order Upwind
Energy	Second Order Upwind

TABLE 5.3: Fluent numerical schemes

5.4.3 Turbulence Model

In all simulations the viscous Shear Stress Transport (SST) $k - \omega$ turbulence model developed by Menter [25] was employed as implemented in ANSYS CFX and Fluent [18] [19] for the solution of the RANS equations.

In general, the SST $K - \omega$ model is analogous to the Baseline (BSL) $K - \omega$ (aside from the variation discussed in Equation 5.5), which combines the robust and accurate formulation of the $K - \omega$ model in the near-wall region with the freestream independence of the $K - \epsilon$ model in the far field. To achieve this, the two formulations are blended with a specific multiplying function that activates the correct model in the correct region. The corrections introduced by the SST variation improve its accuracy and reliability for a series of flows (for example, adverse pressure gradient flows, airfoils, transonic shock waves).

Generally, the $K - \omega$ two-equations model for the closure of the RANS equations is based on two variables - K , the turbulence kinetic energy, and ω , the specific dissipation rate. Both are obtained from their respective transport equations:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_j} \left(\Gamma_k \frac{\partial k}{\partial x_j} \right) + G_k - Y_k \quad (5.1)$$

$$\frac{\partial}{\partial t}(\rho \omega) + \frac{\partial}{\partial x_i}(\rho \omega u_i) = \frac{\partial}{\partial x_j} \left(\Gamma_\omega \frac{\partial \omega}{\partial x_j} \right) + G_\omega - Y_\omega \quad (5.2)$$

Where G_k and G_ω represent the generation terms and Y_k and Y_ω the dissipation ones. In the SST $K - \omega$ model, their definition is as in the Baseline $k - \omega$ model (and different from the standard $k - \omega$). Other source terms could be included as user-defined. Γ_k and Γ_ω represent the diffusivity terms, defined as follows:

$$\Gamma_k = \mu + \frac{\mu_t}{\sigma_k} \quad (5.3)$$

$$\Gamma_\omega = \mu + \frac{\mu_t}{\sigma_\omega} \quad (5.4)$$

Where σ_k, σ_ω are the turbulent Prandtl numbers for k and ω , respectively. The turbulent viscosity, μ_t , is specific to the SST model and calculated as follows:

$$\mu_t = \frac{\rho k}{\omega} \frac{1}{\max \left[\frac{1}{\alpha^*}, \frac{S F_2}{\alpha_1 \omega} \right]} \quad (5.5)$$

Here S is the modulus of the mean rate of stress tensor, α_1 is a constant and F_2 is calculated from a combination of parameters. Conceptually, the idea is that turbulent viscosity in the Baseline version of the model is overpredicted, due to the fact that the shear stress transport term is not included in the equations. This impacts upon the accuracy of the solution. Therefore, a limiter - in the form of the second part of equation 5.5 - is introduced.

5.5 LuT dimension analysis

The choice of employing Lookup Tables, though mandatory for the case at hand, does come at some disadvantages. Aside from being able to map only a section of the thermodynamic plane, outside of which the retrieval of properties is not straightforward or directly fails, it introduces interpolation errors that are clearly larger the larger the spacing between individual points is. Naturally, increasing excessively the table size is not beneficial either, if only because of the excessive computational cost required for their generation and reading. Therefore, a balance for their size is to be sought.

For Case 1-1, four different sets of tables were tested, progressively denser. Boundaries were kept the same; namely 220 – 340 K for temperature and 10 – 65 bar for pressure. Table points were calculated from the real gas Span-Wagner equation of state. Even though non-square tables could be implemented, for simplicity the number of points for temperature and pressure was kept the same. Details are reported in Table 5.4.

LuT Points	T spacing [K]	P spacing [bar]
100	1.2	0.55
200	0.6	0.275
300	0.4	0.1833
400	0.3	0.1375

TABLE 5.4: LuT details

Results for ANSYS Fluent are shown in Figures 5.10 and 5.11; no analysis was carried out on ANSYS CFX because it was deemed that the overall interpolation errors would show a similar behaviour in the two codes. In fact, even if the interpolation scheme between single points in the .rgp table is not discussed in depth in the documentation, it is most certainly going to show at least the same degree of accuracy as the user defined bi-interpolation function implemented via UDF in Fluent.

Most importantly, as immediately visible, the Lookup Table size is not a major variable, and has even less impact on the solution accuracy than the mesh. All refinements have an effect on pressure that is far below the uncertainty of the pressure probe itself. However, the refining behaviour is consistent, highlighting the substantial accuracy of the chosen approach.

Denser tables do not bring about a major computational cost increase; the instability reported by Ameli [2] when employing extremely dense tables was not evidenced here - perhaps because the flow conditions lie quite far from the critical point. Considering these factors, the 300 points table was chosen for all simulations of Case 1-1. Simulation of other cases obviously require a boundary change for tables; the difference between minimum and maximum mapped values was not kept constant, especially for pressure. Therefore, spacing was kept the same and the number of points varied accordingly.

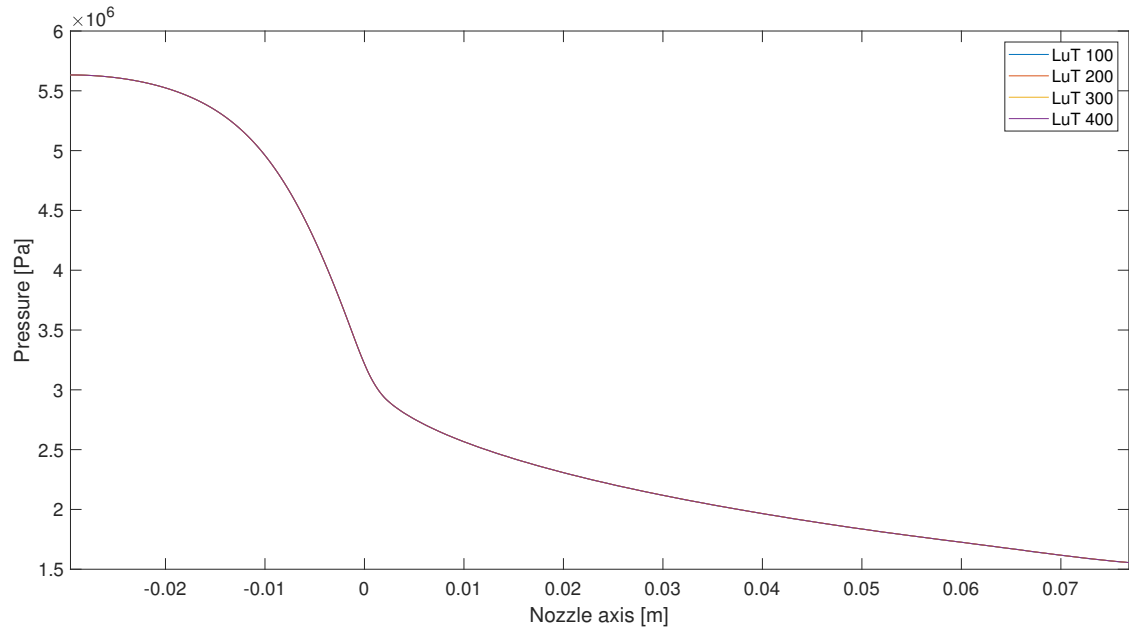


FIGURE 5.10: Pressure in Case 1-1 on the symmetry line for different Lookup Table sizes

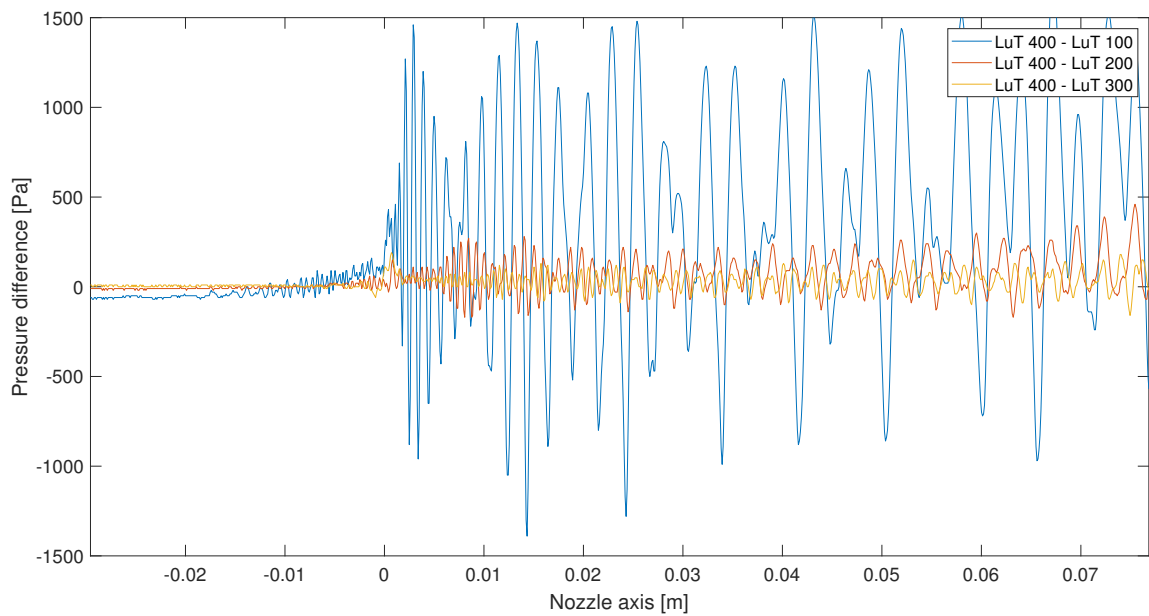


FIGURE 5.11: Difference between the pressures for Case 1-1 with different Lookup Table sizes

5.6 Lee Model coefficient analysis

One last parameter needs to be analysed; namely, the Lee model coefficients represented as K in Equations 2.2 and 2.3, discussed in Chapter 2. Considering that our analysis regards condensation, only the term of Equation 2.2 is actually of interest and influences the solution.

Clearly, this analysis is restricted to ANSYS Fluent, where the model was applied. Implementing the same model in ANSYS CFX was not possible, due to its complexity.

Aside from simulations varying the parameter value, a simulation where the transition line is defined as the spinodal is also presented, and all are compared with a simulation without any condensation model turned on. Results are reported in Figure 5.12 and 5.13 for pressure, and Figure 5.14 for the liquid volume fraction.

Several considerations are in order. First of all, condensation is initiated once the saturation line is crossed - marked as a black dashed line in all figures. The location is around 0.1 mm before the throat. Therefore, no direct comparison with the experimental condensation locations is possible, because experimental condensation is initiated by definition when the Wilson line is encountered. The model includes non-equilibrium effects in terms of condensate generation and its effect on the flow thermodynamics, but this is visible far from the condensation onset, where complex effects regarding droplet formation and interaction with the flow, which are not considered in this framework, are most important. As already discussed in Chapter 2, non-equilibrium effects depend on the coefficient value; the higher it is, the larger (at the same distance from the saturation line) source terms will be, bringing the flow back to equilibrium conditions more quickly.

This is quite evident in Figure 5.14; when the coefficient has a value of $1e^3$ or $1e^6$, although condensation begins at (around, due to numerical imprecisions) the same location, it has a much larger impact and affects a larger portion of the flow. This in turn causes a remarkable pressure increase, shown in Figure 5.12.

In fact, if the value is too high, the solution ends up being driven to equilibrium too fast, and behaviour recalls the HEM one - which will be discussed in more depth in Chapter 6.

Contrarily, if the coefficient has a low value (0.1, or even 10) its effects on the flow field are almost negligible, to the point that the pressure field is basically unaffected. The solution for coefficient 0.1 shows almost the exact same pressure trend as the solution where, instead of the saturation line, the software was fed the spinodal line - and therefore condensation was never initiated, as this expansion is bound to never cross it (as shown in Figure 2.3). For this reason no liquid fraction is generated and this expansion is not reported in Figure 5.14. The differences with the single-phase case where no condensation model is activated are therefore simply due, in both cases, to the different solution process which activates different equations inside the solver.

In the end, the value of $1e^3$ was chosen for the coefficient. Although the highest value seems to achieve better results, in reality the trend for pressure values in the first region of the divergent is due to the non-equilibrium condensation onset and droplet formation. Therefore, our model should instead achieve adherence with pressure values at the end of the divergent, far from the condensation onset and in a situation where both phases can be considered in Eulerian terms. This type of behaviour was also obtained by Hosangadi [16].

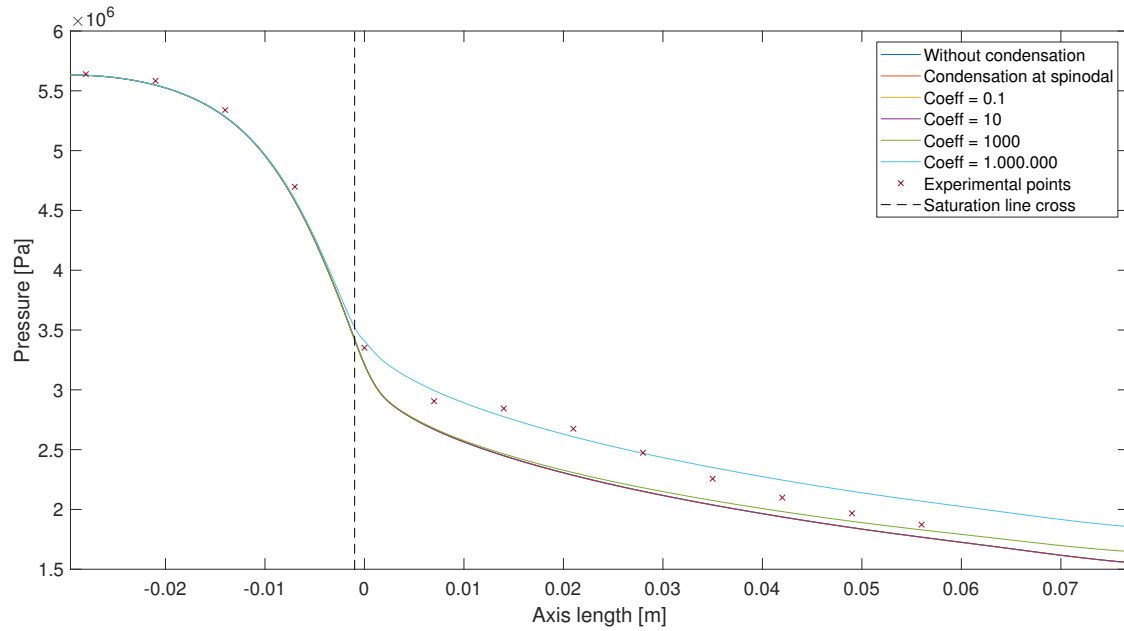


FIGURE 5.12: Pressure in case 1-1 on symmetry line for different Lee Model coefficient values

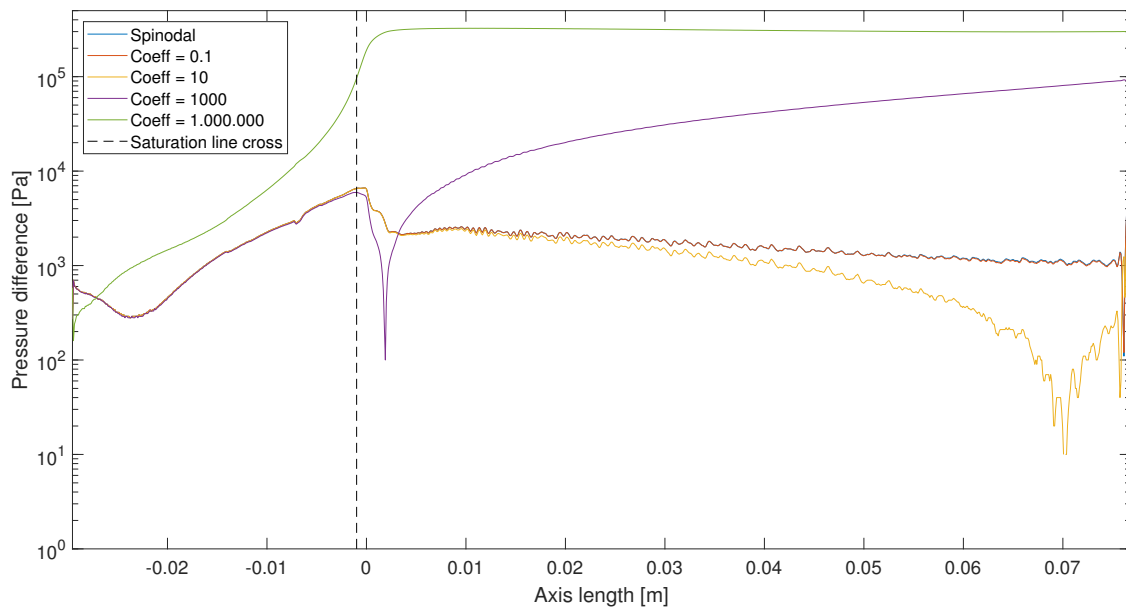


FIGURE 5.13: Difference between the pressures for Case 1-1 at different values of the Lee model coefficient (blue and red lines overlap)

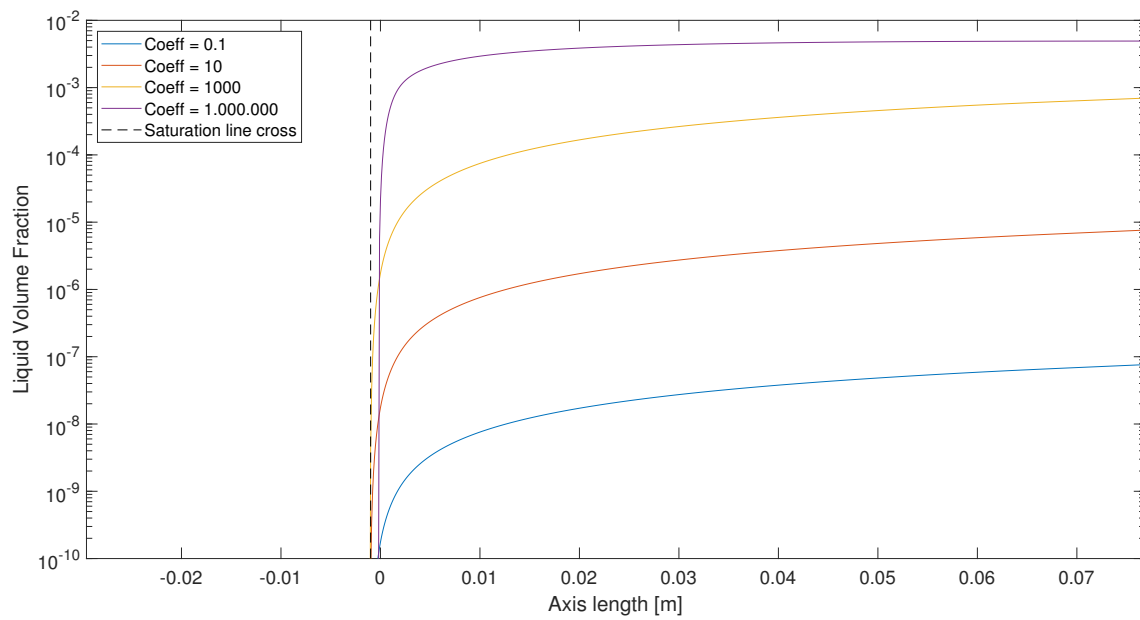


FIGURE 5.14: Liquid Volume Fraction in case 1-1 on symmetry line for different Lee Model coefficient values

Chapter 6

Results Analysis

In this chapter all results concerning simulations will be discussed.

6.1 Cross-section pressure variation

It was implicitly assumed in simulations reporting experimental data (see for instance Figure 5.12) that the experimental pressure values, which are reportedly obtained from sensors located on the nozzle walls, are comparable with the pressure calculated by the software on the symmetry line of the nozzle. This needs to be verified. Therefore, we compared the cross-sectional pressure variation in the throat for single-phase simulations of Case 1-1 in CFX and Fluent in Figure 6.1.

The variation is modest, in the order of magnitude of around 0.1 *bar* (less than 1% with respect to the throat value); at other sections pressure was nearly constant. This is still comparable with the sensor inaccuracy, therefore it was deemed that comparing experimental values with numerical values on the symmetry line was correct.

6.2 Single-phase solution comparison for Case 1-1

In this section, a comparison of the main thermodynamic and numerical quantities of single-phase solutions obtained in Fluent and CFX employing spinodal lookup tables for Case 1-1 will be carried out. Both simulations employ the same Lookup Table (300 points, bounds 10 – 65 *bar*, 220 – 340 *K*) and Mesh 2 as described in Chapter 5.

Pressure on the symmetry line is reported in Figure 6.2 for CFX, Fluent and the monodimensional code (set to transition at the spinodal, therefore showing noncondensing behaviour) discussed in Chapter 4.

Pressure is remarkably coherent in the convergent section for all three solutions. However, it still fails somewhat short of the experimental data, even if the flow physically condenses only in the divergent. Though no quantitative comparison could be carried out, a similar solution with similar discrepancies was also obtained by Hosangadi [16]. No immediate explanation is available for this phenomenon; it is thought that the experimental flow does not exhibit perfect isentropic behaviour even in the single-phase region (such as the one shown here, except for small rounding errors, as highlighted in Figure 6.3), and therefore deviates from the somewhat idealised numerical solution.

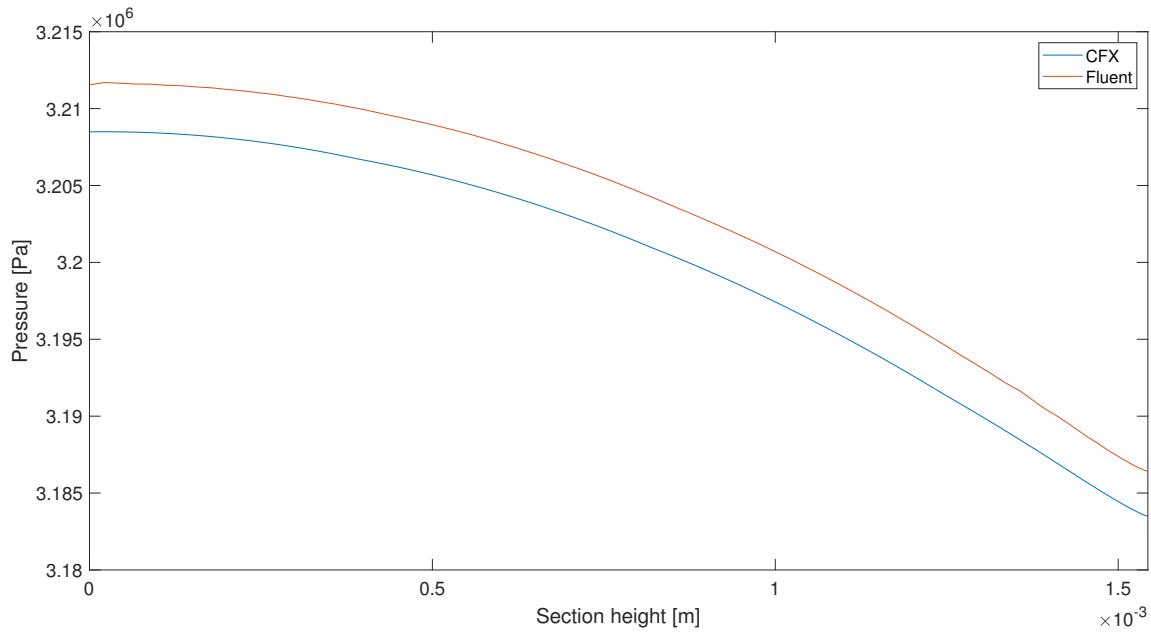


FIGURE 6.1: Cross-section pressure for Case 1-1 in the throat

Concerning the divergent, there is a larger pressure difference that is caused by the condensation onset and development. This causes a pressure increase that our simulations at this stage are not supposed to model.

At the outlet and its immediate proximity, CFX shows irregular behaviour due to its treatment of the boundary condition, as briefly discussed in Chapter 5. Setting an exact pressure value to remove this discontinuity is not easy *a priori*, as condensation modifies the pressure outlet value. Moreover, even when it was done, no significant differences in simulations arose.

Temperature is reported in Figure 6.4, and shows more or less the same trend already described for pressure, although there are no experimental data for direct comparison. It almost reaches the lower boundary of the lookup tables; if that was crossed, simulations would behave erratically.

Further extensions of the lower LuT temperature bound would be hard due to the proximity of the triple point; once that is crossed, solid phase would appear as well. In general, all thermodynamic calls would be meaningless, or unfeasible (REFPROP fails in this area), and the overall simulation framework would need to be revisited.

The difference between the two CFD solutions (CFX and Fluent) is rather small, as shown in Figure 6.5. Deviations are calculated by normalising the absolute difference between the two solutions with the property value determined by Fluent, and are largest in the throat and its proximity, where gradients are higher and flow evolves from subsonic to supersonic. This is naturally more complicated.

The sound speed difference is much lower than that of other quantities; considering that it is calculated by pure Lookup Table interpolations in both codes, it implies that the user-defined interpolation scheme implemented in Fluent is consistent with CFX .rgp option.

Conversely, velocity differences are higher. Considering that velocity is one of the direct

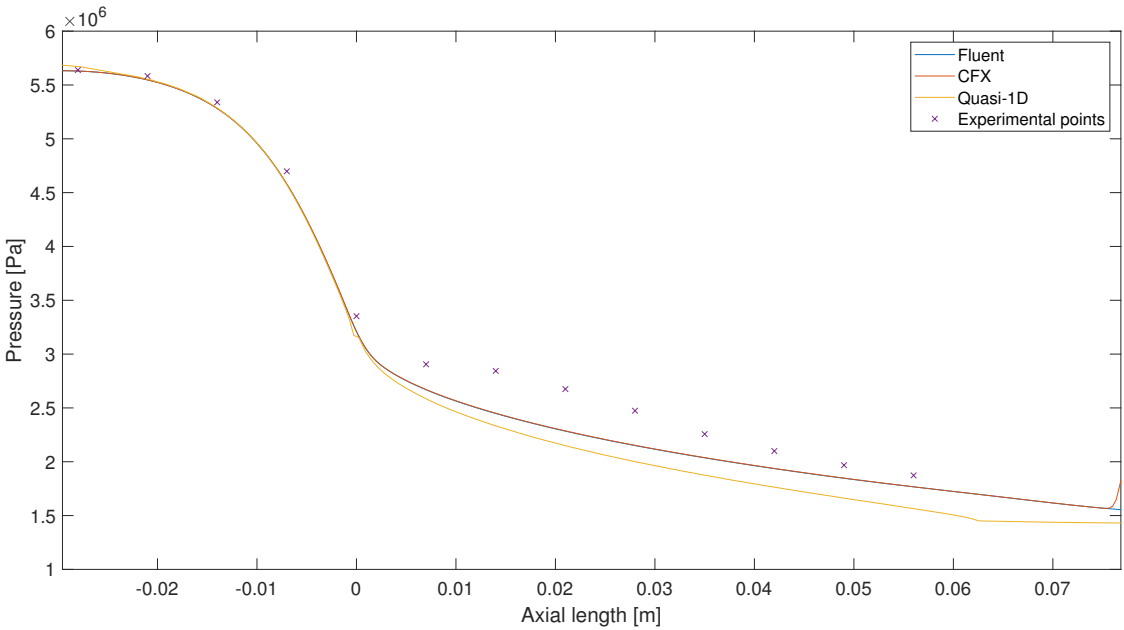


FIGURE 6.2: Single-phase pressure for Case 1-1

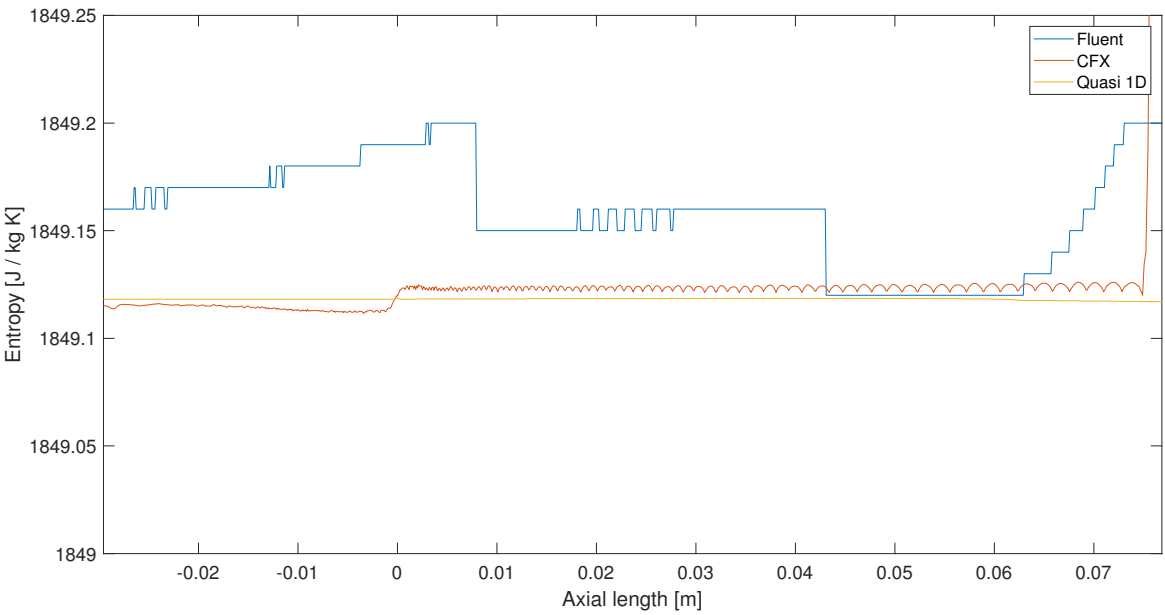


FIGURE 6.3: Single-phase entropy for Case 1-1

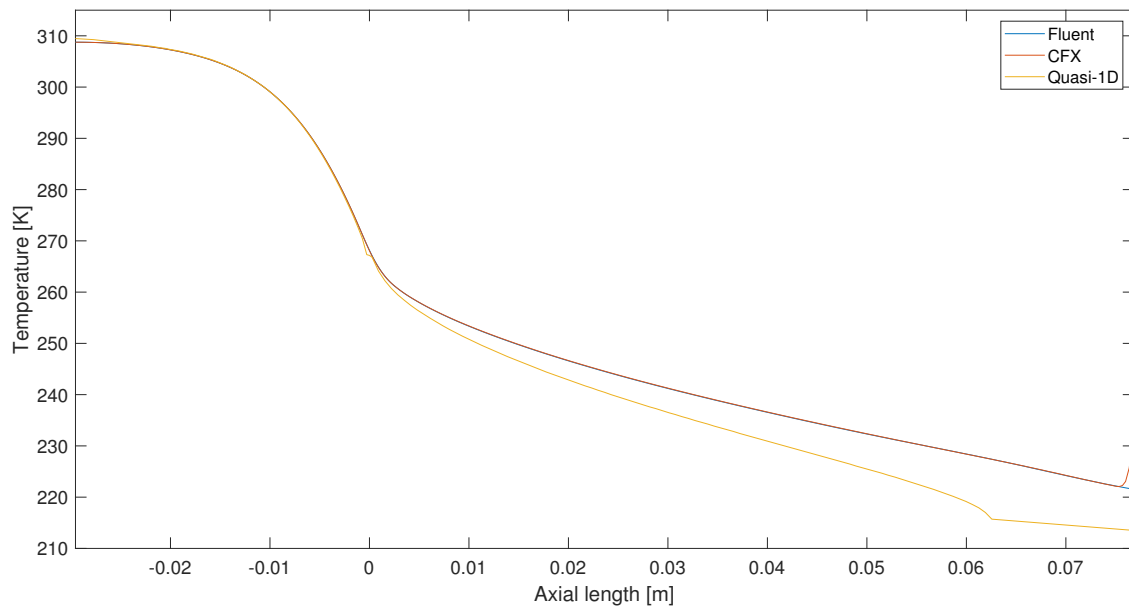


FIGURE 6.4: Single-phase temperature for Case 1-1

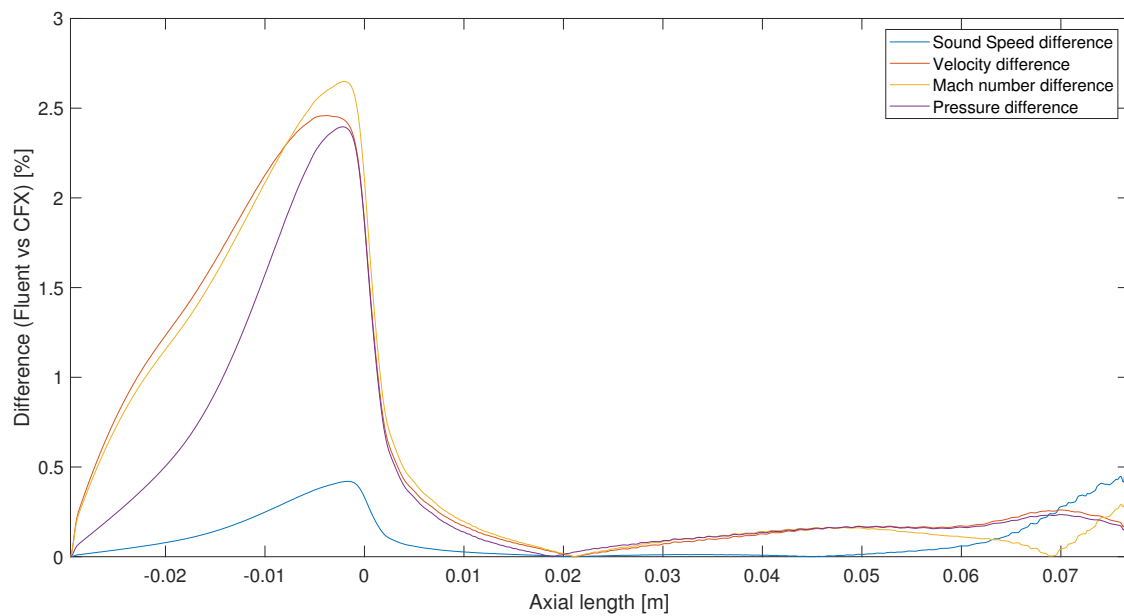


FIGURE 6.5: CFX and Fluent comparison of different properties (denoted by different legend entries), normalised with Fluent values

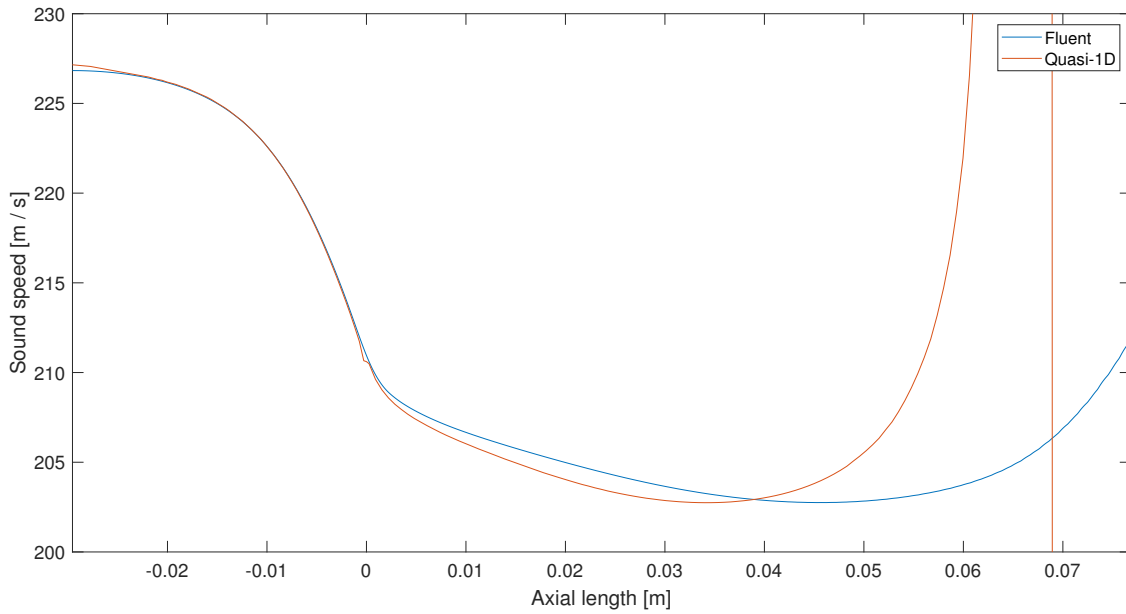


FIGURE 6.6: Single-phase sound speed for Case 1-1

unknowns of the differential equations system, that error is believed to be due to the difference in numerical solving schemes implemented into the codes. User choice is limited in that sense, and therefore it is deemed that it cannot be reduced. The overall difference is still lower than 3% in the throat, which is thought to be an acceptable tolerance. That order of magnitude is reflected onto the Mach number, whose difference is mainly due to velocity differences.

Therefore, pressure differences are most likely due to different software behaviour, and cannot be controlled by the user at this stage.

Figures 6.6 and 6.7 represent a peculiar real gas effect that takes place in the far metastable region, near the spinodal line, especially at low temperatures; the sound speed has an irregular behaviour and eventually, despite temperature and pressure decreasing, increases to the point of showing divergent behaviour near the spinodal line; calculations do not return meaningful values. This has obvious opposite effects on the Mach number. The single-phase solution, which expands to lower pressures (since, as discussed in Chapter 4, it cannot model boundary layers and viscous effects), better shows this trend and eventually the Mach number calculation, carried out in post-processing, fails. The same trend is shown by Fluent (and CFX, unreported here, as it would basically overlap with Fluent), though it is less marked because the penetration into the metastable area is smaller.

It is thought that if an attempt to replicate the thermodynamic conditions shown by the quasi-1D solution was tried in CFD, simulations would fail for the unstable behaviour of the sound speed (and consequently Mach number), even if the spinodal line would ultimately not be crossed. In fact, as already mentioned in Chapter 4, CFD for Case 1-1 failed when the wall boundary condition was modified to allow for slip. However, convergence failure could be due as well to the proximity of the lower boundary of the Lookup Tables.

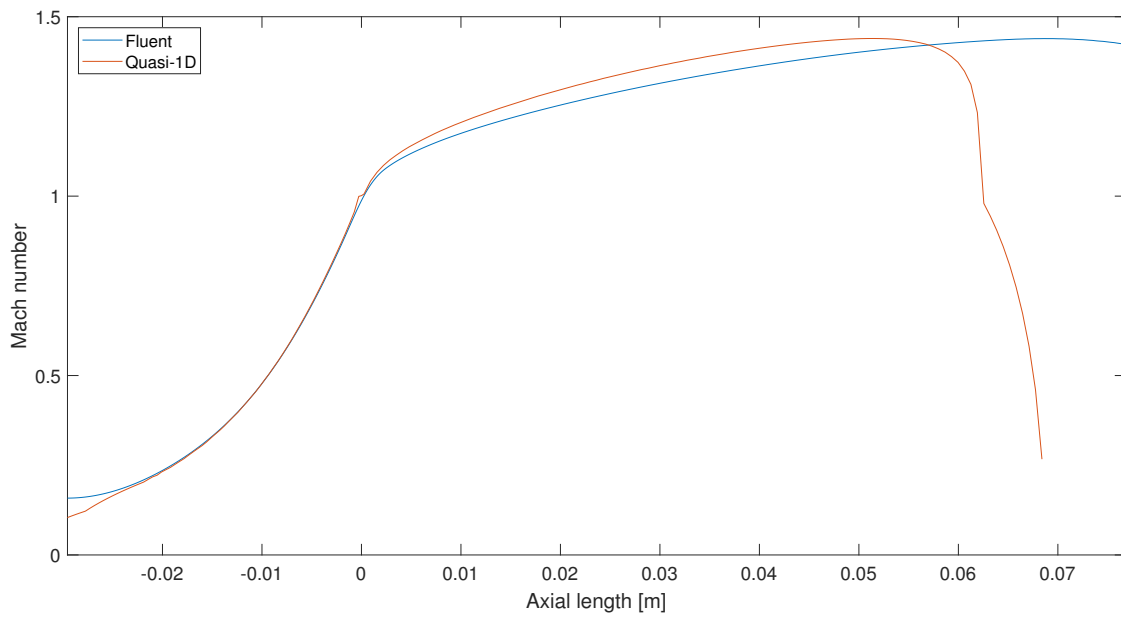


FIGURE 6.7: Single-phase Mach number for Case 1-1

6.2.1 Single-phase behaviour for ANSYS CFX

In this section the behaviour shown by ANSYS CFX when single-phase simulations happen to cross the transition line defined by the `.rgp` file will be discussed. The solution that was already presented above for CFX is compared with a simulation that employs a single-phase table which defines the Wilson line for transition purposes.

The two solutions show the same behaviour until the Wilson line is reached and crossed. This, in two-phase simulations, would mark the onset for condensation. Instead, here properties for the simulation transitioning at the Wilson line are extrapolated along a quadratic line. This was determined through the analysis of entropy; though it is supposed to be constant all along the nozzle (as visible in Figure 6.3), once the transition line is crossed the CFD solution shows an irregular growth, as highlighted in Figure 6.9. MATLAB functions `polyfit` and `polyval` were used to create an interpolating quadratic line that almost perfectly overlaps with the simulation trend.

This behaviour naturally extends to all flow properties. Furthermore, from the analysis of the expansion in the $P - T$ plane, it is visible that the expansion ends up lying on the chosen transition line - analogously to two-phase HEM simulations. However, entropy is generated.

Though it may not generate large errors for small and limited penetrations in the two-phase dome, it is concluded that `.rgp` tables should include metastable states and specify the spinodal line for transition, if penetration of the saturation line is thought to be possible. The extrapolated behaviour (which is never notified to the user by CFX) would be harder to detect in more complicated simulations, and in general it generates unreliable results.

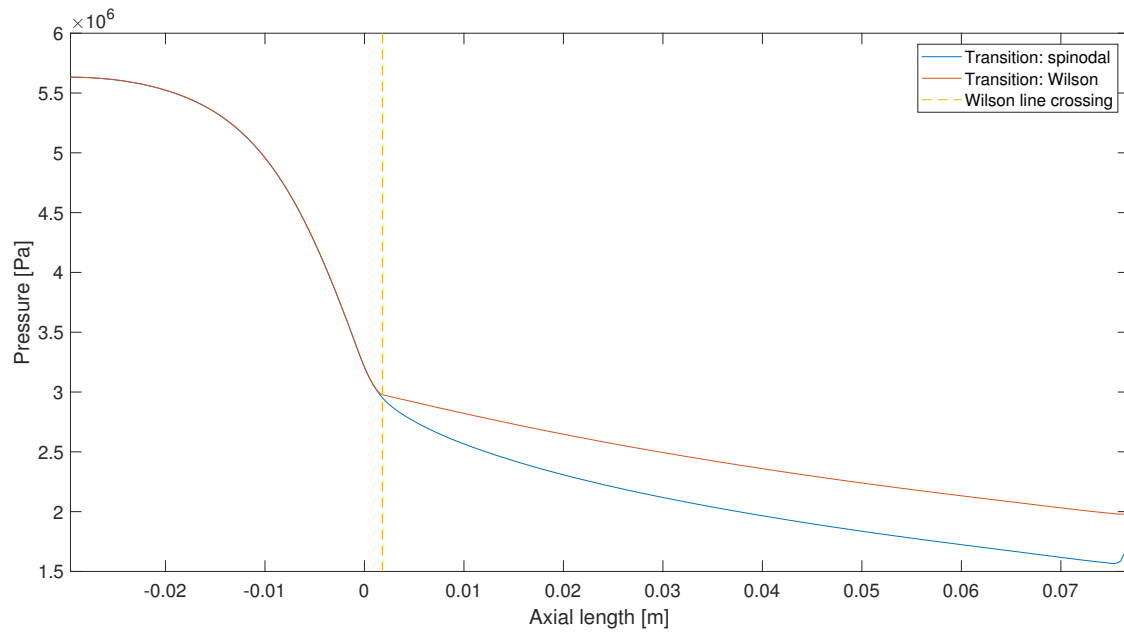


FIGURE 6.8: Single-phase pressures for .rgp tables transitioning at spinodal and Wilson

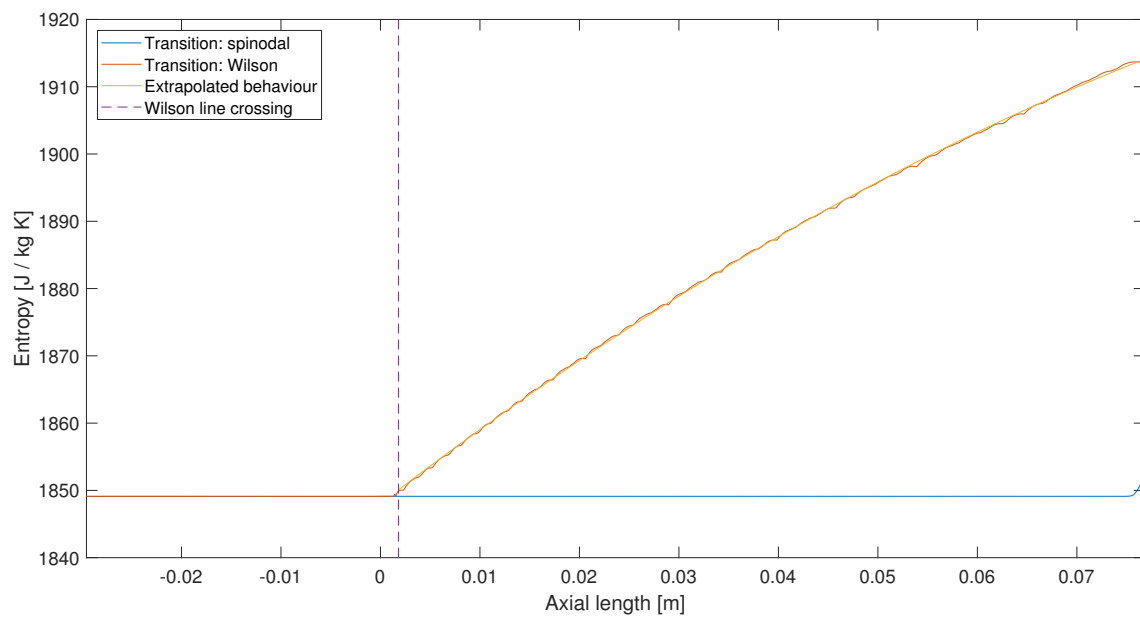


FIGURE 6.9: Single-phase entropies for .rgp tables transitioning at spinodal and Wilson

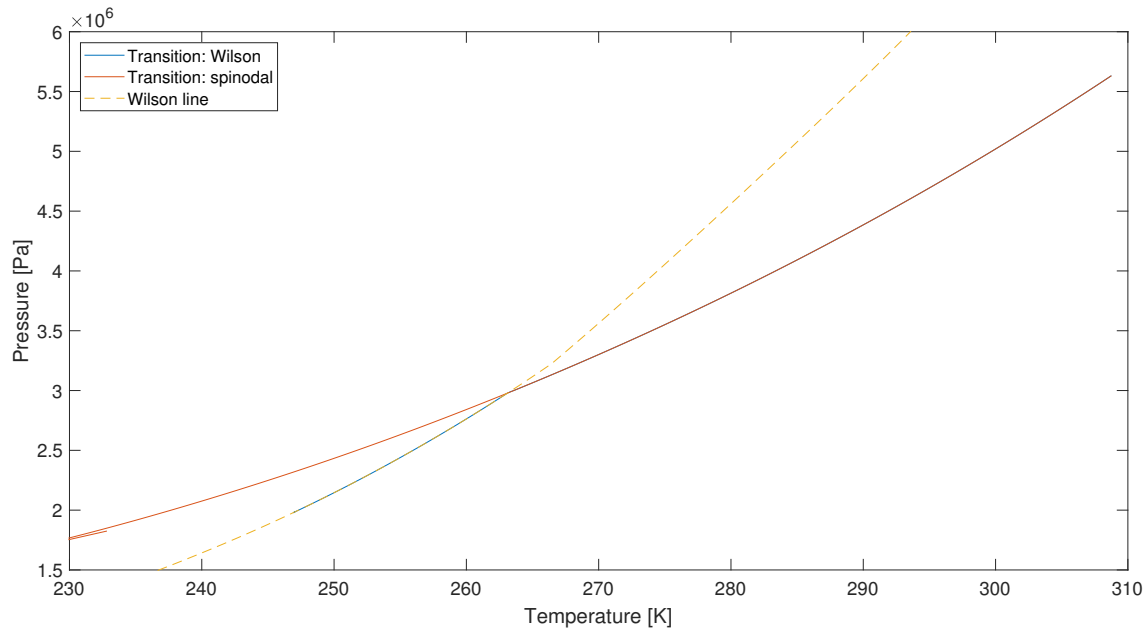


FIGURE 6.10: P-T behaviour for single-phase .rgp tables transitioning at spinodal and Wilson

6.3 Two-phase solutions comparison for Case 1-1

Two-phase solutions are compared below. The HEM employed in CFX and in the quasi-1D code defined the Wilson line for transition purposes to include as much as possible non-equilibrium effects, while Fluent employed the Lee Model as discussed in Chapter 2.

Both pressure values and condensation onset loci are analysed.

6.3.1 Thermodynamic behaviour

Results in Figure 6.11 and 6.12 show pressure and temperature profile for aforementioned simulations and models. Large differences arise in the divergent section, after condensation is initiated (before then, solutions show single-phase behaviour, discussed above). In fact, after the Wilson line is crossed, the HEM forces the solution to strictly lie on it in terms of $P - T$ coupling (flow is assumed to be in saturated conditions), while Fluent does not maintain this constraint and shows a two-phase trend which looks to be a natural continuation of its behaviour in the single-phase area. The HEM constraint is highlighted in Figure 6.13, and causes all flow quantities to show an irregular behaviour and clear slope change which is not justified otherwise. Therefore, even if it seems that the experimental two-phase pressure behaviour can be somewhat accurately replicated by CFX, these results are unreliable; namely, the fact that the solution over-estimates pressure at the last probes, far from the onset, implies that the complete trend is inaccurate and its slope merely happens to cross the experimental evolution at some point. This is more evident from the temperature profile, which exhibits an unnatural bending when the Wilson line is crossed and flow enters into two-phase state. In general, the $P - T$ constraint and the fact that the flow is assumed to be at equilibrium is not justified in any mean by the nature of the experiment.

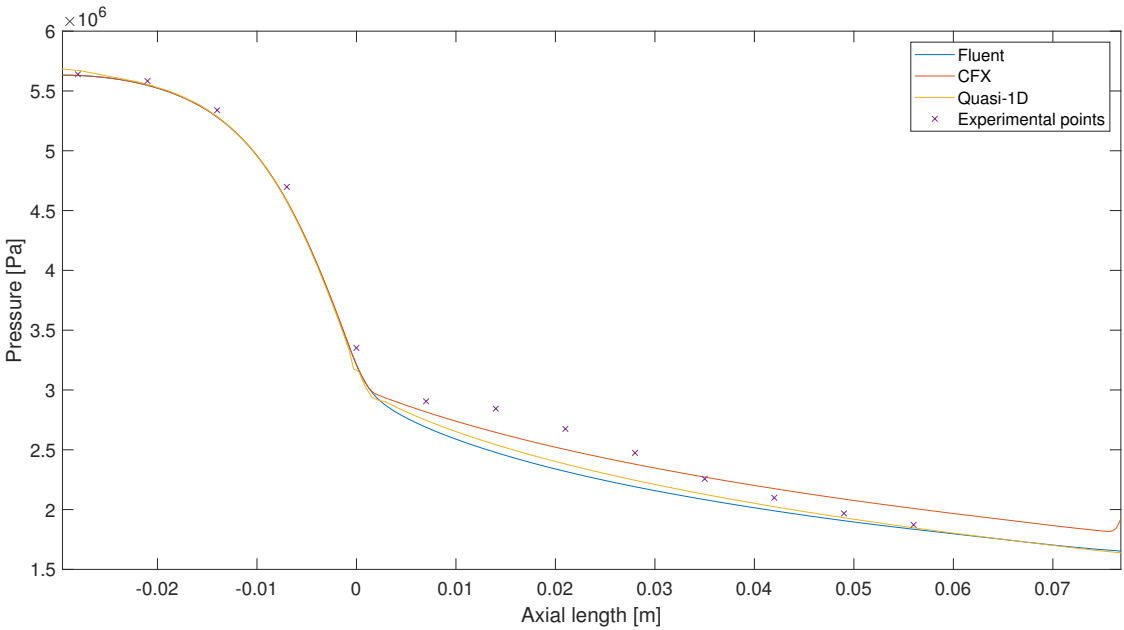


FIGURE 6.11: Two-phase pressures

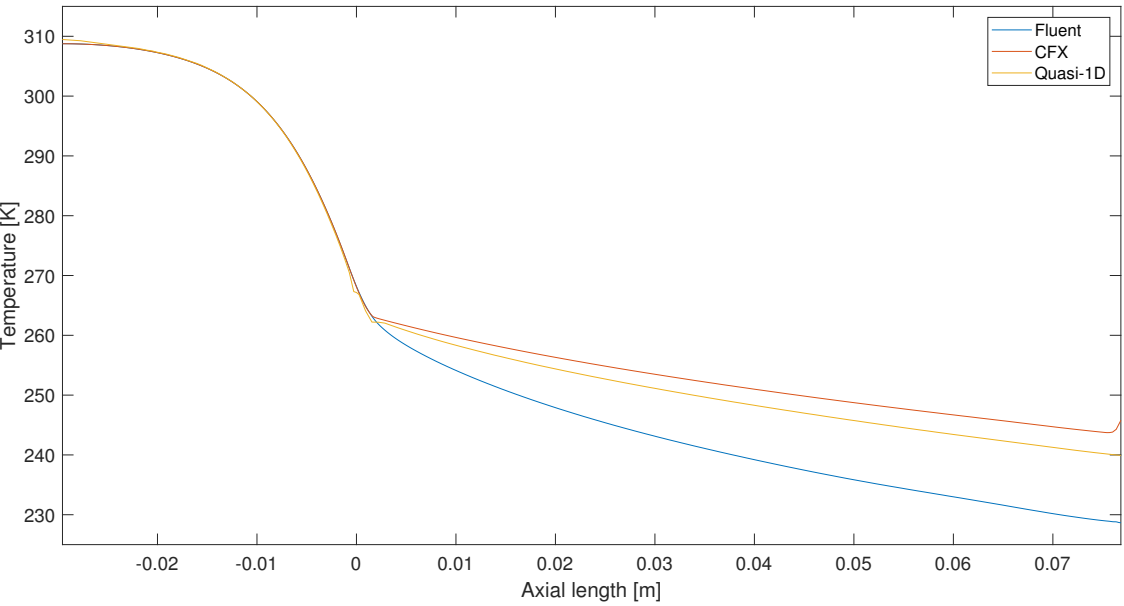


FIGURE 6.12: Two-phase temperatures

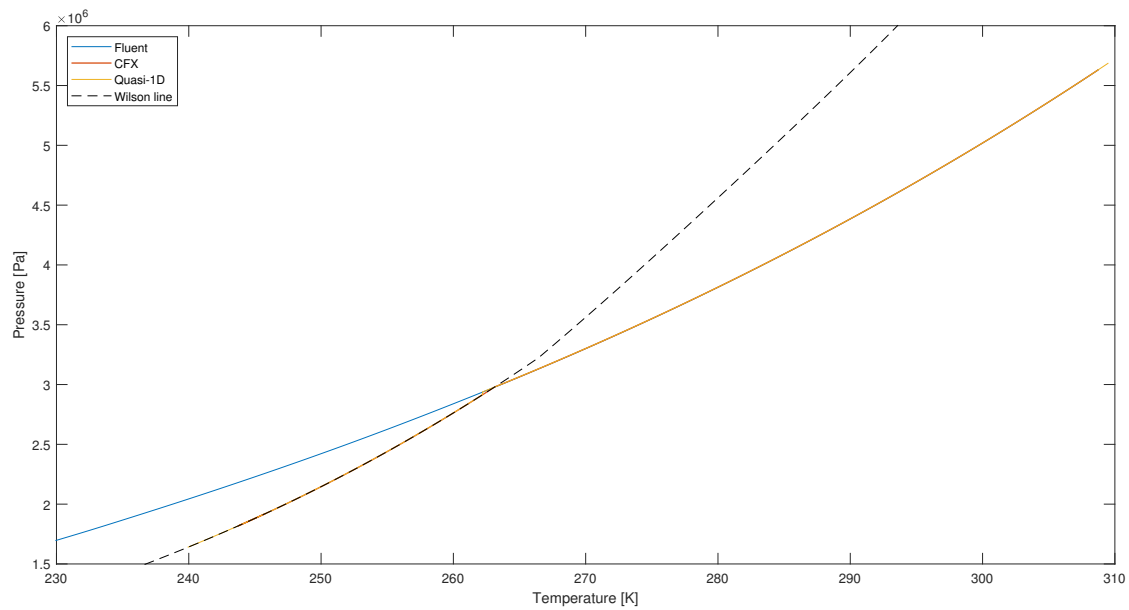


FIGURE 6.13: Two-phase solutions behaviour on T-P plane

Finally, the physical flow is eventually due to return to equilibrium states; this cannot be simulated in CFX, where the equilibrium lies on the Wilson line, but is not experienced in Fluent either - as clearly shown by the slope of the $P - T$ line representing the Fluent simulation in Figure 6.13. Numerically, this implies that the source terms have not had enough influence yet to bring the flow to saturation. However, pressure data are coherent; this implies that, experimentally, this evolution is most likely to happen only once the flow has slowed down, or after enough time has passed; as such, out of our domain of interest.

The unnatural behaviour is confirmed by the behaviour of Mach number (Figure 6.15); Fluent shows a more natural trend, whilst the HEM transition causes a significant reduction of the flow expansion - it becomes significantly slower, as highlighted in Figure 6.16, due to the same constraint that was highlighted before and which affects unphysically the solution (here, through the difference of total and static enthalpy, as the first one is constant).

The difference between Fluent and CFX in terms of Mach number, sound speed and velocity is shown in Figure 6.14, a similar representation with respect to Figure 6.5. The differences in the two-phase area are visibly larger, while those in the single-phase region are consistent. It is worth mentioning that the error in terms of pressure and velocity between the two solutions is similar; this trend, which also appeared in the single-phase simulation, suggests that the difference due to numerical schemes adds almost linearly to the one for the condensation model in the case at hand. The error for Mach number is significantly larger than for single-phase simulations due to the presence of thermodynamic differences impacting upon the sound speed.

Though it may appear from Figure 6.11 that the quasi-1D solution shows a different behaviour and that actually improves its adherence with the experimental results, these were obtained in absence of boundary layers. Therefore, the fluid expanded to lower pressures, penetrating deeper in the two-phase dome. This effect somewhat balances the

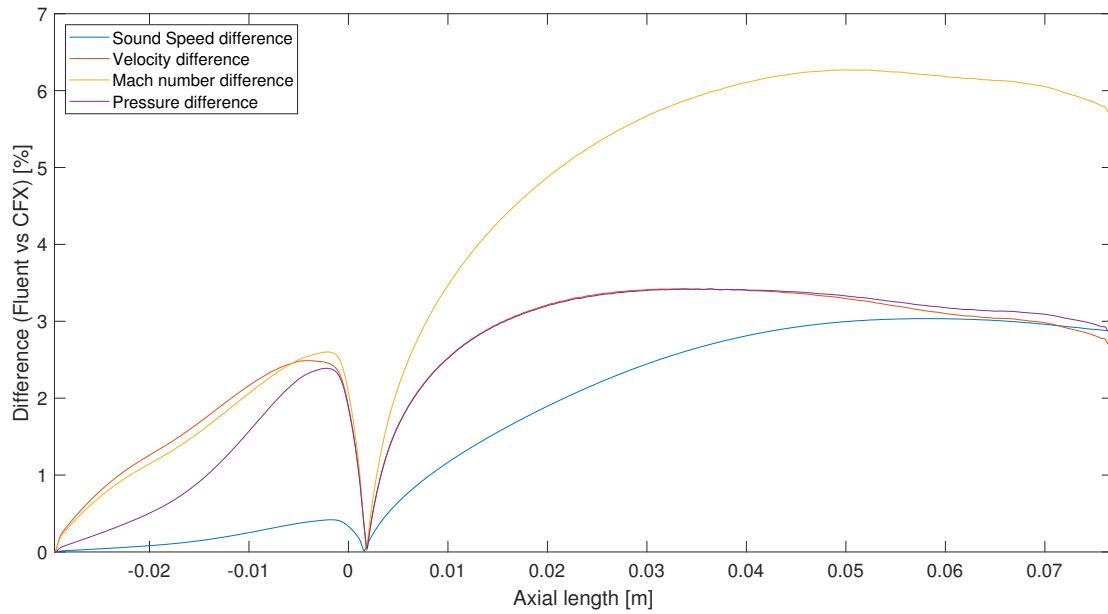


FIGURE 6.14: CFX and Fluent comparison of different two-phase properties (denoted by different legend entries), normalised with Fluent values

error due to the condensation model; therefore, accuracy is not higher, but this solution merely happens to be closer to experimental values. In fact, it is still bound by the Wilson line in terms of $P - T$ behaviour, and tends to overlap with CFX (as discussed in Chapter 4) if the CFD wall boundary condition is changed to Free Slip (rather than No Slip).

A noteworthy portion of the differences between the quasi-1D code and CFX is also caused by the different evaluation of the sound speed in two-phase states. This is highlighted in Figure 6.17, where sound speed was re-evaluated in post-processing from pressure, temperature and liquid volume fraction of the solutions according to Equation 4.1. The post-processed values overlap perfectly for Fluent (which therefore employs Equation 4.1, as also mentioned by Giacomelli [15]), but differ significantly in CFX. This also explains the small differences that were highlighted in Chapter 4 concerning inviscid two-phase simulations.

In general, this inviscid solution is unreliable, and for the sake of clarity is not always reported.

Entropy production (shown in Figure 6.18) is modest for all models; strictly speaking, CFX should behave isentropically, similarly to the quasi-1D code (which shows a little bump after the condensation onset due to numerical rounding errors). The discrepancy and the entropy production, which is present even in the absence of boundary layers, is probably due to numerical errors in the solution. Fluent is not supposed to behave isentropically, because thermodynamic equilibrium is not directly enforced by the model, and modest entropy production is thought to be coherent with the experimental phenomenon; also, entropy values directly displayed by the software were evidently incorrect, and the variable shown here was actually calculated in post-processing from pressure, temperature and liquid mass fraction in MATLAB.

Pressure contours are reported for both CFX and Fluent in Figure 6.19. In general, they attest to the monodimensional evolution of the simulation; though the comparison is merely qualitative, it reinforces our earlier conclusions regarding the overall coherence

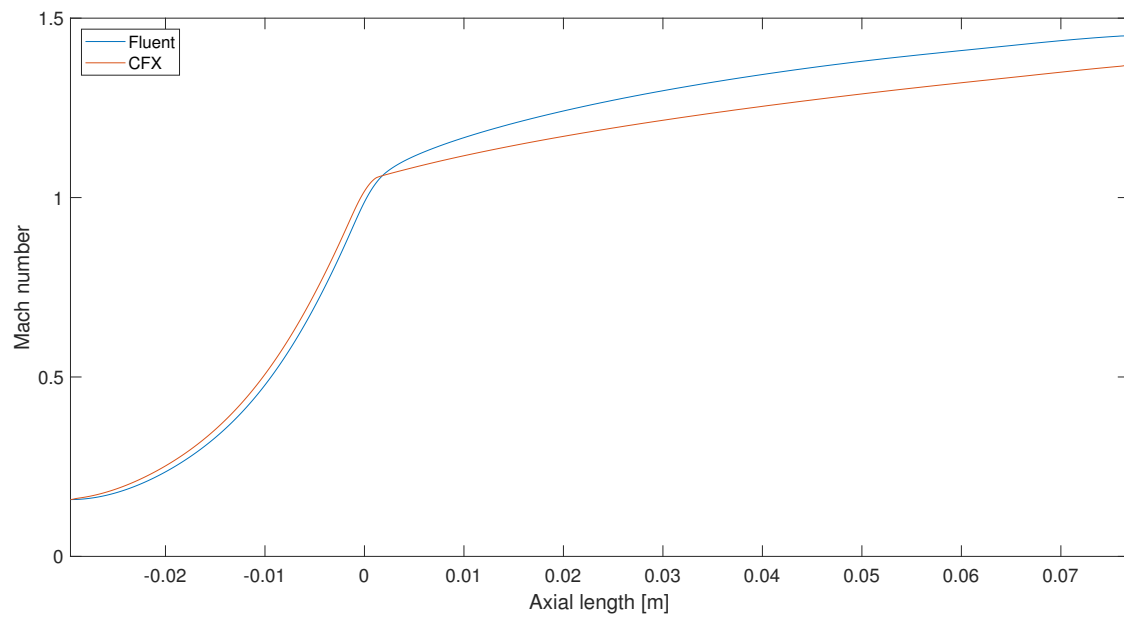


FIGURE 6.15: Two-phase Mach number

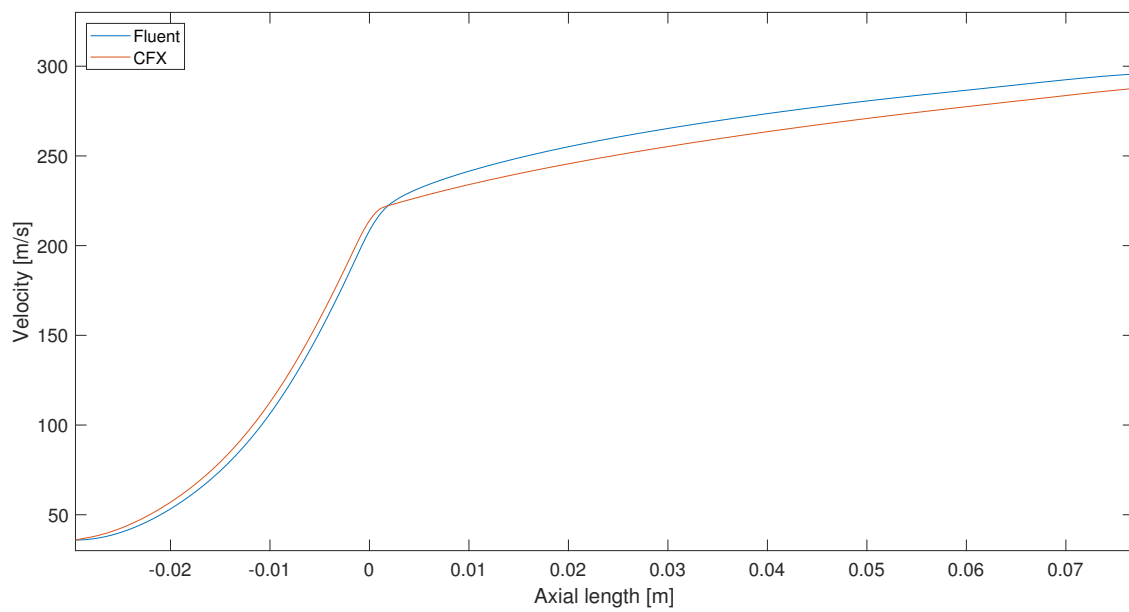


FIGURE 6.16: Two-phase velocity

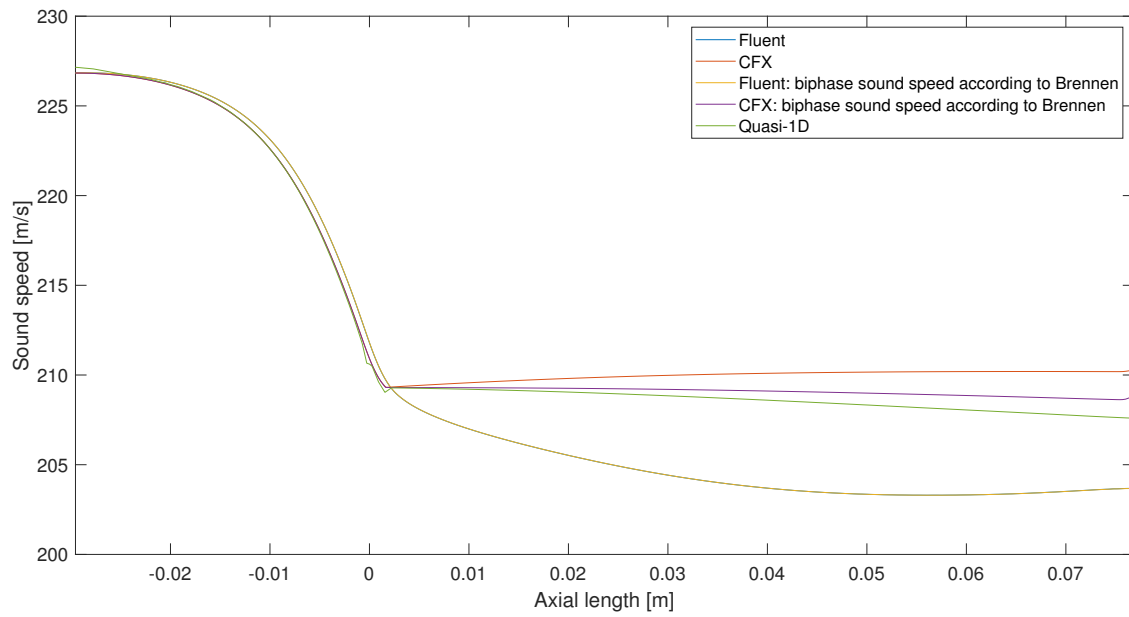


FIGURE 6.17: Two-phase sound speed (blue and yellow lines nearly perfectly overlap)

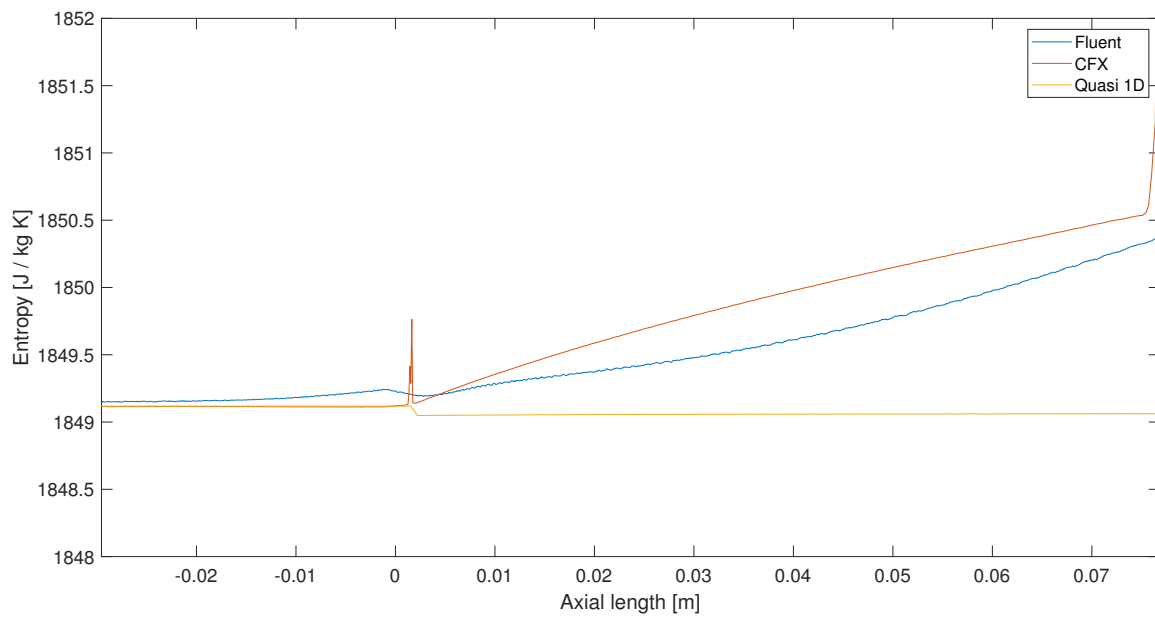


FIGURE 6.18: Two-phase entropy

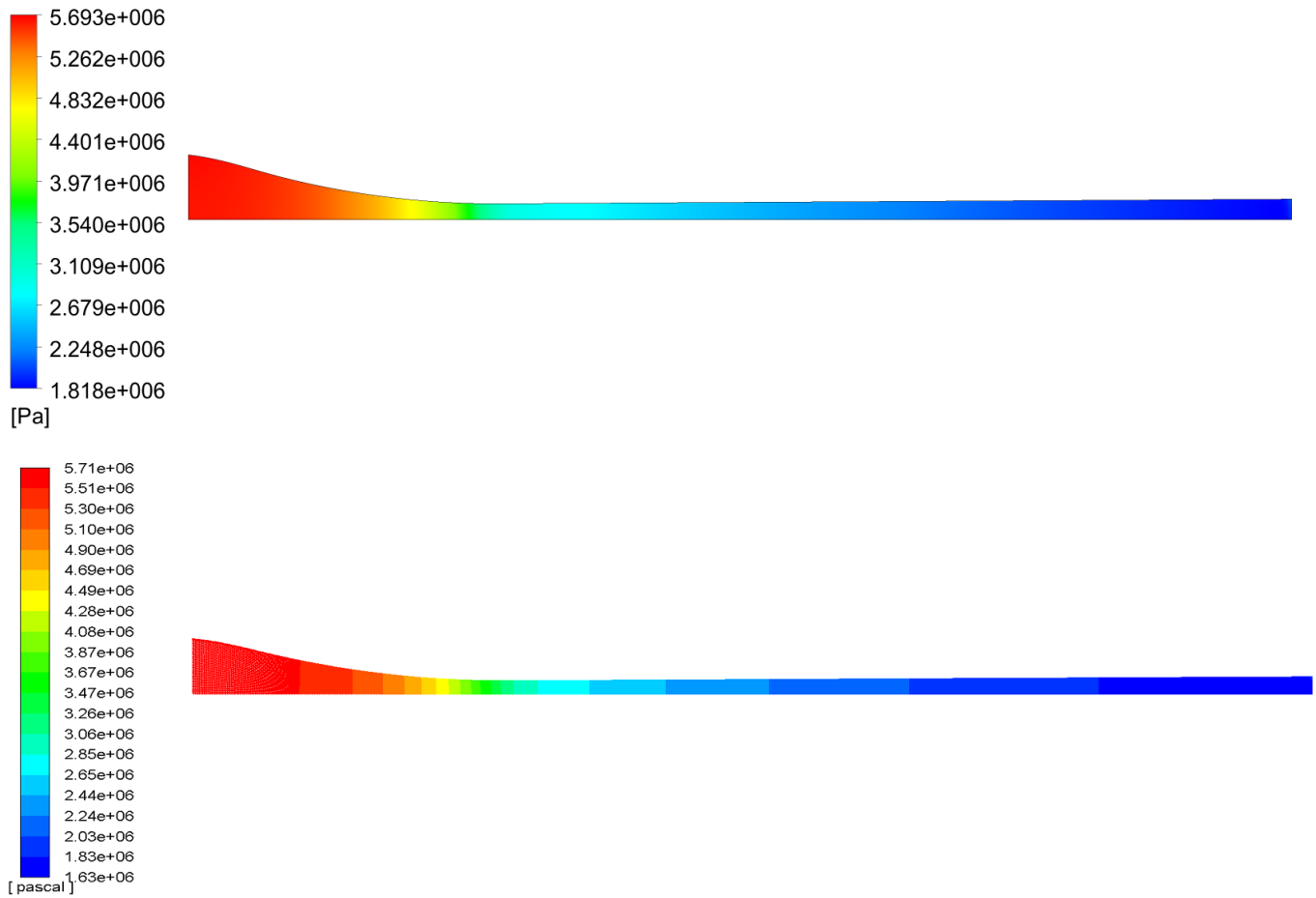


FIGURE 6.19: Pressure contours for CFX (above) and Fluent (below) for Case 1-1

in the single-phase region, compared to larger differences in the two-phase one.

The results clearly highlight that the most reliable option to assess pressure behaviour is the Lee Model as implemented in Fluent. The difference between the simulations and the experimental trend is quantified in Figure 6.21. Firstly, all solutions are extremely coherent in the single-phase region. Concerning the two-phase areas, Fluent has consistent behaviour, showing higher values in the near onset of condensation (in the throat region), and decreasing in the near outlet, where they are comparable to the single-phase region. Instead, CFX error trend is irregular, even overpredicting, rather than underpredicting, pressure. Fluent clearly fails to account for the effects due to droplet formation and interaction with the surrounding flow which are most important in the near-onset region, as they are simply not included in the condensation model.

Despite these limitations, the solution is consistent with literature findings; as a reference, results obtained by Hosangadi [16], who was ultimately able to extend his setup on a full compressor geometry with good results, for Case 1-2 are compared to our solution and reported in Figure 6.20. It is not certain whether the experimental trend can be replicated with state-of-the-art condensation models implemented in CFD codes.

As a reference, simulations for Case 1-2 ($P_t = 58.96 \text{ bar}$, $T_t = 310 \text{ K}$) were absolutely analogous in terms of mesh, numerical methods and LuTs employed with respect to Case 1-1, and showed similar behaviour.

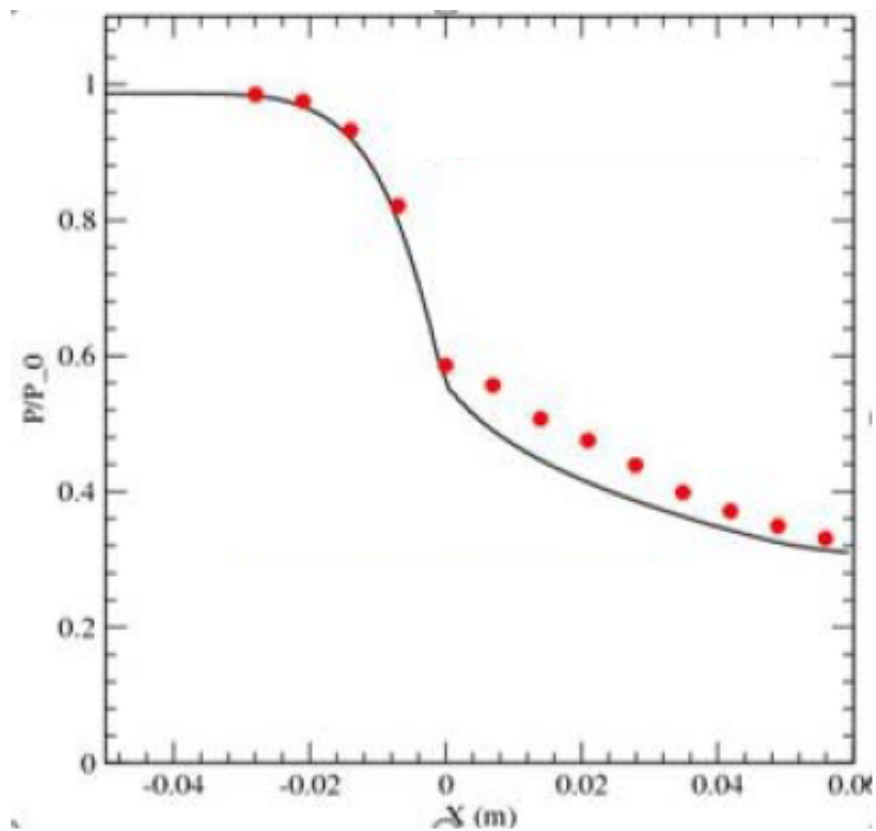
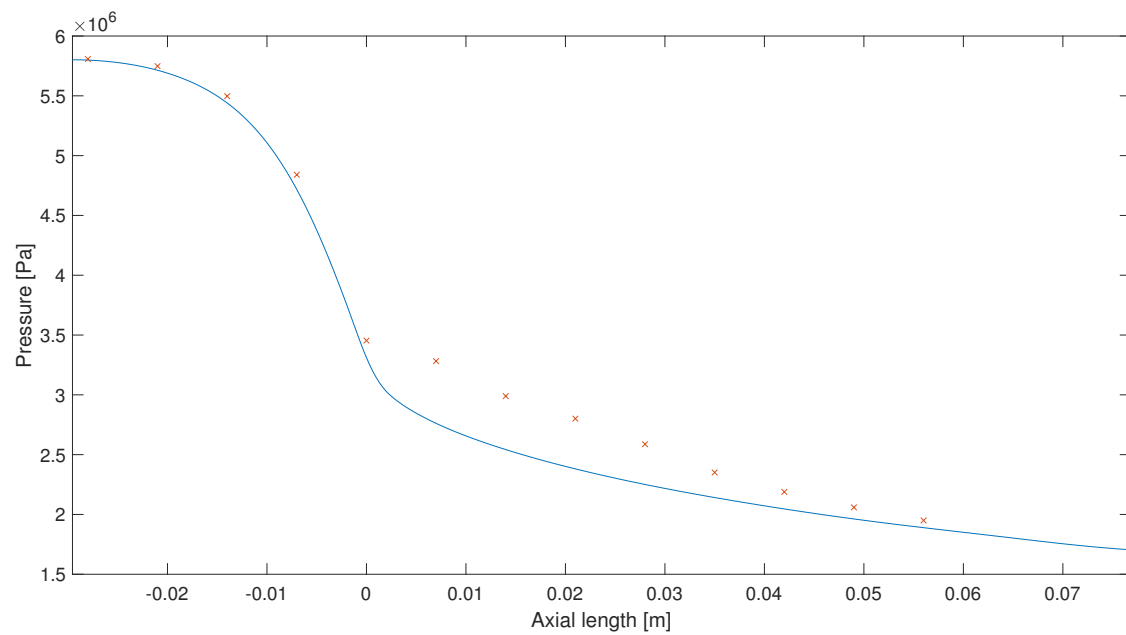


FIGURE 6.20: Comparison between Fluent (above) and Hosangadi (below) [16] for Case 1-2

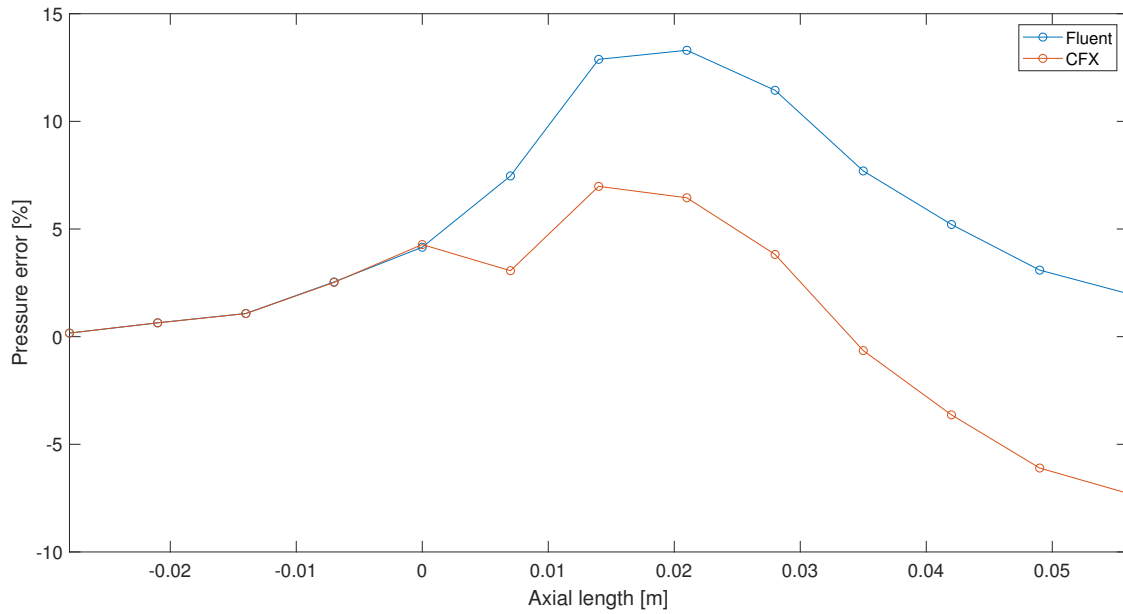


FIGURE 6.21: Difference between simulations and experimental values, normalised with Fluent values

Therefore, this thermodynamic setup is quite accurate as far as the thermodynamic effects of nonequilibrium condensation on flow behaviour far from the onset are concerned. Due to the fact that we lack direct experimental data, we cannot guarantee that the simulation is able to replicate accurately other thermodynamic conditions (namely, temperature), nor that it is going to be able to drive the flow to equilibrium (as it is going to happen once the flow slows down) comparably with the physical situation.

The effect of the condensation model on the overall Fluent simulation is shown in Figure 6.22 and 6.23; it clearly highlights the temperature and pressure rise (especially the former) due to condensation, which releases latent heat. This improves the adherence with the experimental reference, as expected.

6.3.2 Condensation locus assessment

The analysis of the flow pressure in the two-phase region is not the only parameter of interest. Assessment of condensation and namely its onset was analysed from the behaviour of liquid mass fractions (directly available in CFX and calculated in post-processing from Fluent). This behaviour is studied in Figure 6.24. Naturally, the quantitative information is not of primary interest here (even though it ultimately influences pressure, and is therefore thought to be quite accurate for Fluent at the near outlet); it is still worth mentioning that the large penetration in the two-phase area, which ultimately constrains CFX to lie on the Wilson line for a large portion of the expansion, generates an unphysically large quantity of condensate.

In order to account for numerical rounding errors, the tolerance for assessing liquid presence and therefore condensation onset was set at $1e^{-6}$ [kg/kg]. The thermodynamic condensation onset is analysed in Figure 6.25 in terms of pressure and temperature.

Experimental assessment was based on optical visualisations; no information regarding its uncertainty or what amount of liquid fraction it may be able to detect is available.

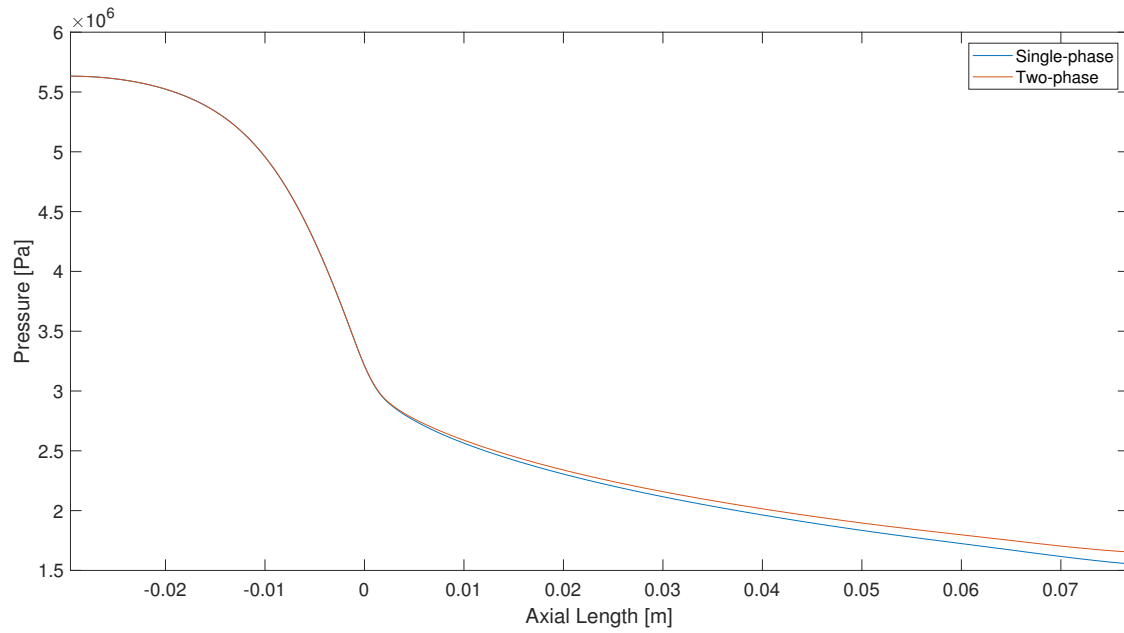


FIGURE 6.22: Pressure comparison for mono- and two-phase simulation

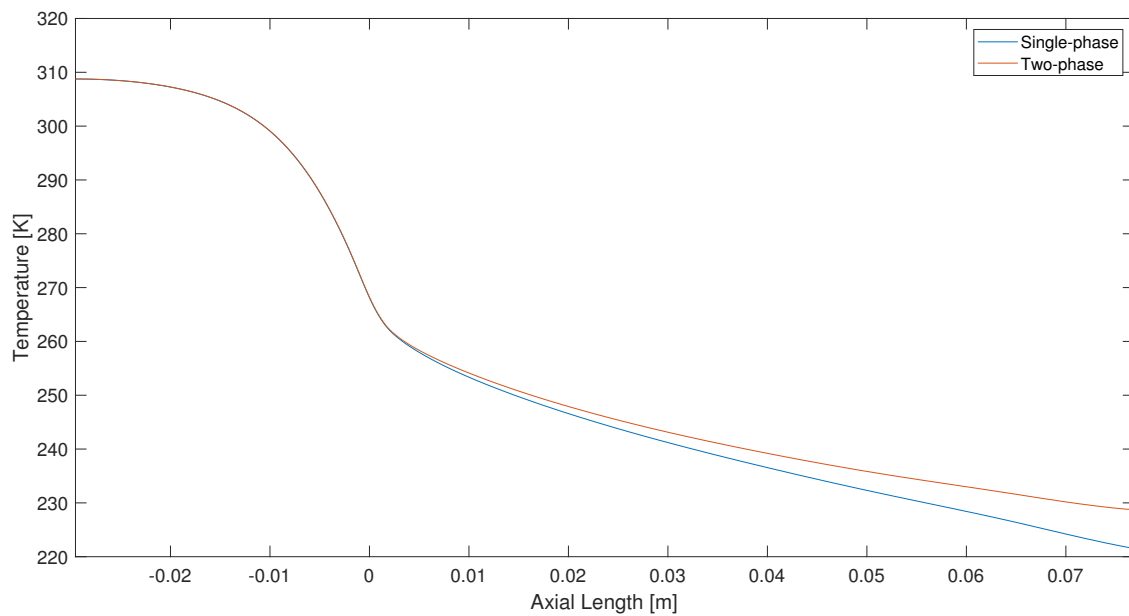


FIGURE 6.23: Temperature comparison for mono- and two-phase simulation

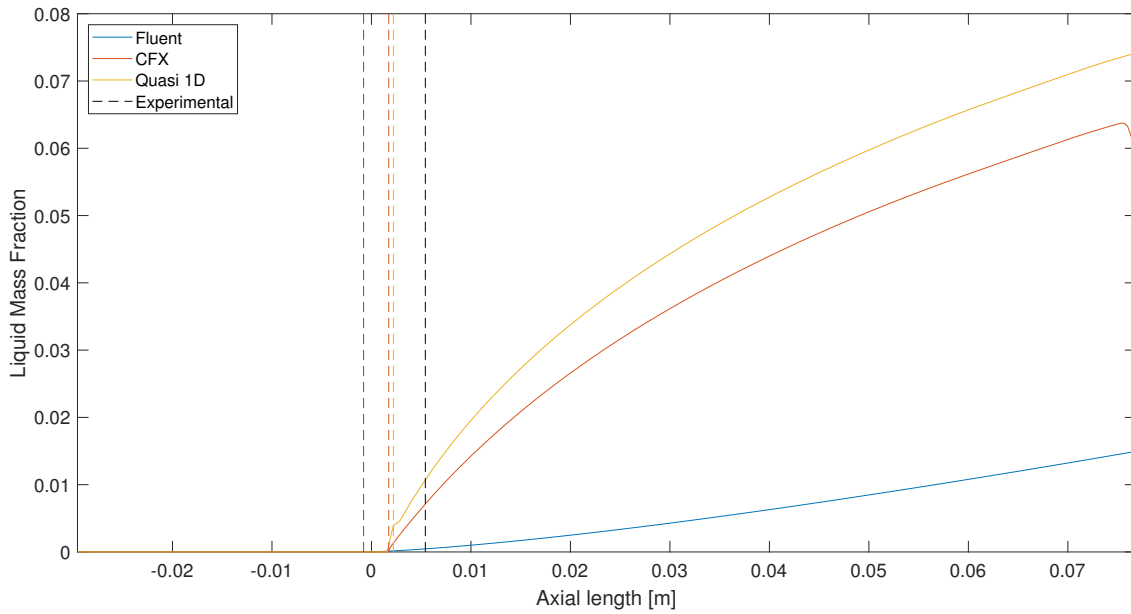


FIGURE 6.24: Liquid mass fractions and condensation onset (dashed) for two-phase simulations

Naturally, the Lee Model shows the worst results on this type of assessment, as it activates the source terms and therefore transfers mass to the liquid once the saturation line is crossed.

The reason for the discrepancy between experiments and numerical solution for the HEM as implemented in CFX and the MATLAB code needs instead a deeper analysis; theoretically, both physical and numerical condensation should start at the same thermodynamic locus, defined by the crossing of the Wilson line.

Therefore, the difference is due to other factors regarding the overall simulation. In fact, as analysed in Figure 5.4 of Chapter 5, our inlet entropy (which stays approximately constant) does not match perfectly the experimental equivalent; secondly, the experimental flow does not necessarily behave fully isentropically in the single-phase region (as shown by pressure differences in that area). Finally, we must remember that the Wilson line as implemented in the HEM was obtained by means of interpolation and extrapolation and therefore does not overlap perfectly with experimental points (as clearly visible in Figure 6.25, where the experimental locus does not precisely lie on the Wilson line as defined in the present work). Most likely, the difference is due to a mix of these factors, though it is not easy to determine which is prevalent.

It is worth mentioning that CFX and the quasi-1D solution are once again quite consistent, probably due to the fact that boundary layers have larger effects in supersonic flows; therefore they influence the solution in the divergent, after the start of condensation. This hypothesis is also suggested by the behaviour of pressure in the converging section of the nozzle.

This type of assessment can also be carried out on a single-phase simulation by the analysis of $P - T$ behaviour to detect the crossing of the Wilson line. This was carried out on both the CFX and Fluent simulations set to show single-phase behaviour (transition at spinodal in the .rgp), with similar results.

In order to carry out this analysis, several approaches are possible; the easiest possible,

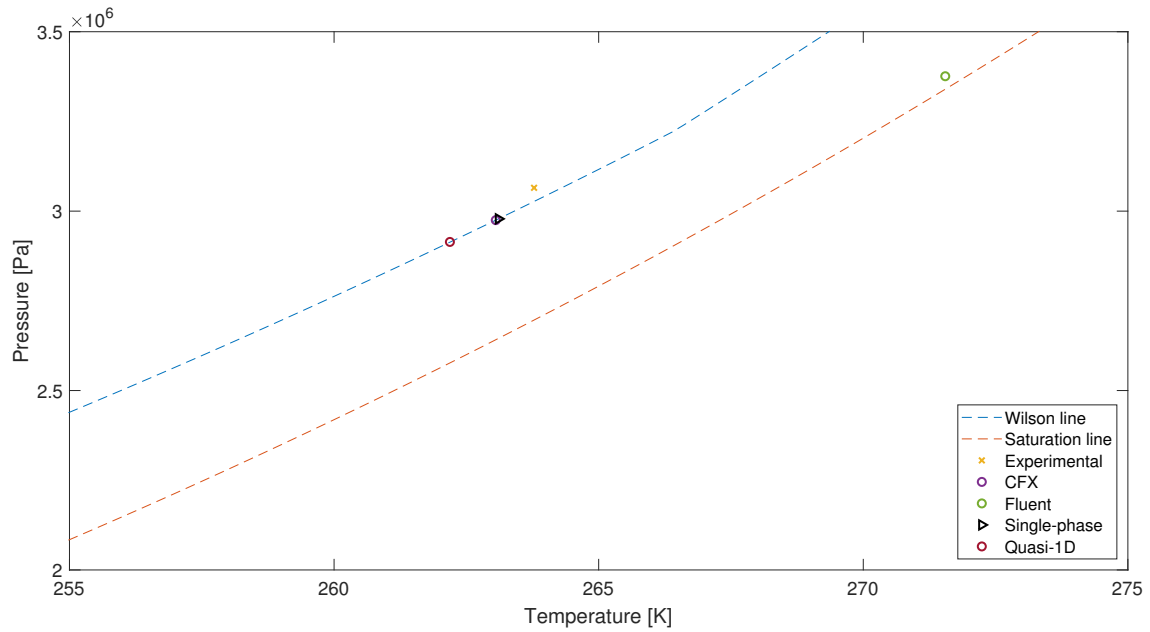


FIGURE 6.25: T-P diagram showing thermodynamic location of condensation onset for all assessments

considering the overall isentropic behaviour of the simulation, is the calculation of the temperature or pressure on the Wilson line corresponding to inlet entropy assigned to the present simulation. At that point the transition is defined to happen when pressure or temperature (both yield the same result, as the simulation behaves isentropically) fall below the threshold. This threshold, having been defined only from the inlet conditions, is the same for both CFX and Fluent (therefore this assessment is reported only once in Figure 6.25) and, in terms of pressure and temperature, overlaps almost perfectly with the one coming from the two-phase HEM simulation. The axial location varies slightly due to the fact that the solution themselves have a slightly different evolution, having been obtained with the solution of different equations and models. Results for all assessment are reported in Table 6.1.

Simulation type	x condensation [mm]
Quasi - 1D	2.206
Fluent (single-phase)	1.544
Fluent (two-phase)	-0.7889
CFX (single-phase)	1.007
CFX (two-phase)	1.739
Experimental	5.416

TABLE 6.1: Axial location of condensation

Even if an accurate replication of the experimental condensation locus was unattainable for all simulations in this work, the simulation HEM setup, as well as the threshold analysis, is going to yield perfect results in those terms as long as all other experimental conditions can be replicated accurately, and is not supposed to introduce any type of error *per se*.

The real challenge in this framework is the determination of a suitable Wilson line, which

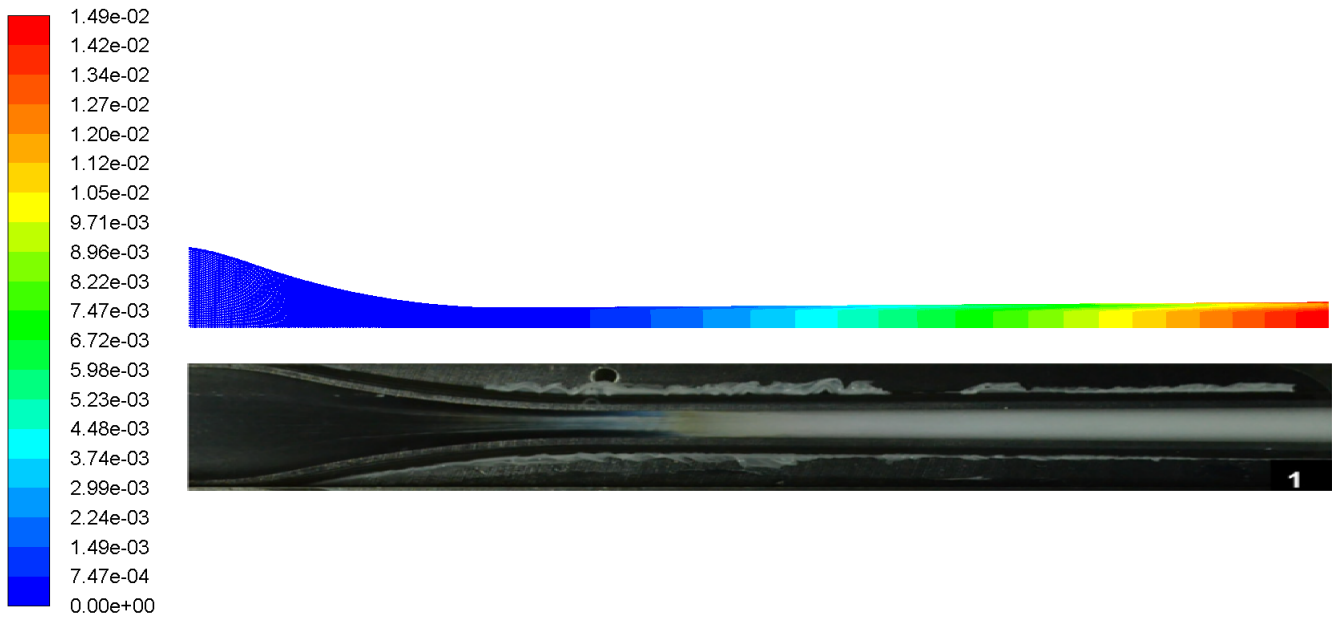


FIGURE 6.26: Liquid Mass Fraction contour for Fluent for Case 1-1, compared with the experimental reference (from [22])

is *flow dependent* and should rigorously be determined experimentally for each and every situation. The potential error coming from the use of a general Wilson line will need to be considered on a case-by-case scale.

Also, in the framework of a more complicated simulation or geometry, the determination of two-phase behaviour in a single area, if it is not small enough, may lead to inaccuracies extending to the overall flow which may consequently invalidate the detection of other areas. From this perspective, a HEM simulation is allegedly going to provide slightly better results if condensation affects only a small area, but it is not certain if that is going to be remarkably beneficial, considering the added computational complexity and the overall model inaccuracy, which causes the simulation to show unrealistic behaviour as discussed above.

Fluent liquid mass fraction contour, compared with a visualisation of the experiment, is reported in Figure 6.26. Though the comparison is clearly only qualitative, there is some degree of similitude between the two solutions. Generally, the solution behaves almost monodimensionally during the flow evolution. However, the presence of boundary layers is quite visible in the divergent. Since the flow is slower, enthalpy and consequently temperature are higher, ultimately causing a reduction of the liquid fraction compared to the free stream regions.

6.4 Solutions for Case 2

Solutions for Cases 2-1 ($P_t = 63.9 \text{ bar}$, $T_t = 310 \text{ K}$) and 2-2 ($P_t = 65.35 \text{ bar}$, $T_t = 310 \text{ K}$) are reported below. No convergence could be achieved for higher total pressure values; reasons for this will be discussed below.

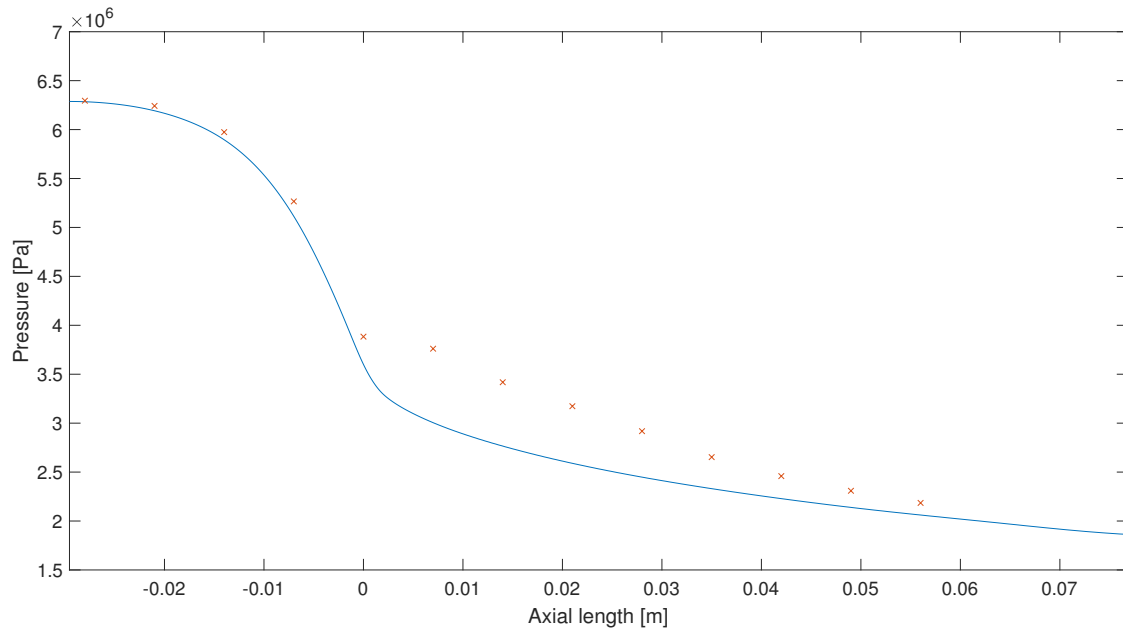


FIGURE 6.27: Pressure for Case 2-1

Considering that the condensation locus assessment failed to achieve accurate results despite the fact that the model behaves as expected and shows transition where it is numerically supposed to, it is determined that no further insight could come from CFX and the HEM in this framework. Therefore, only Fluent solutions will be discussed thereafter. Namely, two-phase pressure trends will be compared with the experimental reference, while condensation locus will be assessed from single-phase spinodal simulations.

Convergence for Case 2 was harder to achieve, and this also led to either unreliable or unstable simulations for CFX. That notwithstanding, second order accurate simulations were ultimately obtained with Fluent, and results are reported below for two-phase simulations (Figures 6.27 and 6.28). Results obtained by Hosangadi [16] are also reported for Case 2-2, and they are once again quite consistent.

The mesh was the same as for Case 1, but LuTs had slightly different boundaries - 15 – 75 *bar* in pressure and 220 – 340 K in temperature, to ensure the complete mapping of the entire expansion.

Single-phase spinodal simulations were calculated as well. However, an issue arised. Due to the lower entropy of Case 2 compared to Case 1, the thermodynamic flow condition is ultimately nearer the spinodal. Single-phase simulations, showing a higher degree of penetration in the two-phase dome, ultimately reached its proximity. Simulations did not completely fail by going in unmapped, or clipped, areas of the LuT; but sound speed ultimately exhibited the same behaviour that was already discussed in the quasi-1D solution in Figure 6.6. This is reported in Figure 6.30 for Case 2-2. This had obvious consequences on both pressure and temperature, as visible in Figure 6.31. The last section of the nozzle, where the solution is meaningless, is directly cut and not reported here (which explains the different values on the x -axis). Single-phase simulations were therefore ultimately unstable.

However, this instability was most apparent in the last portion of the divergent part of the nozzle, and is assumed to ultimately be a local phenomenon. Therefore, condensation

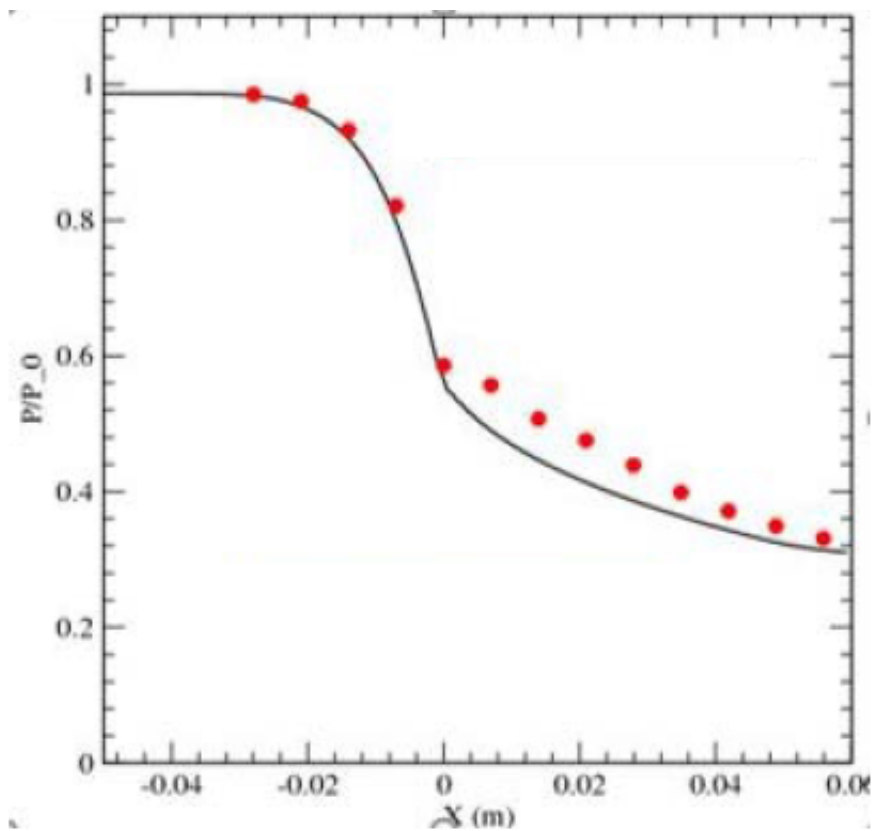
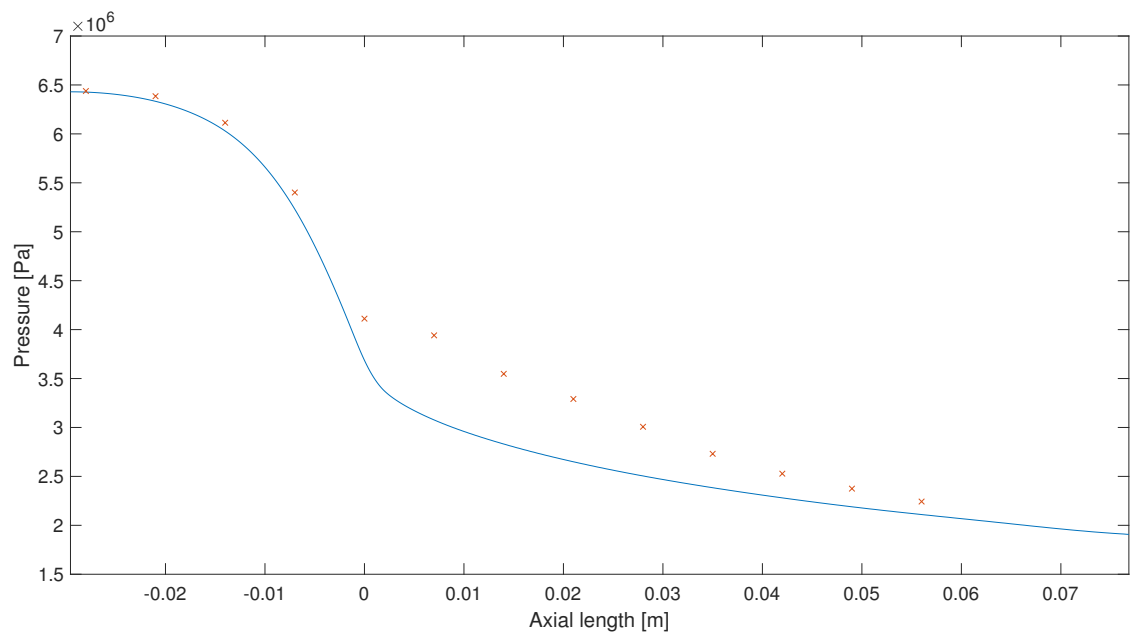


FIGURE 6.28: Pressure for Case 2-2 with Fluent (above) and as obtained by Hosangadi [16] (below)

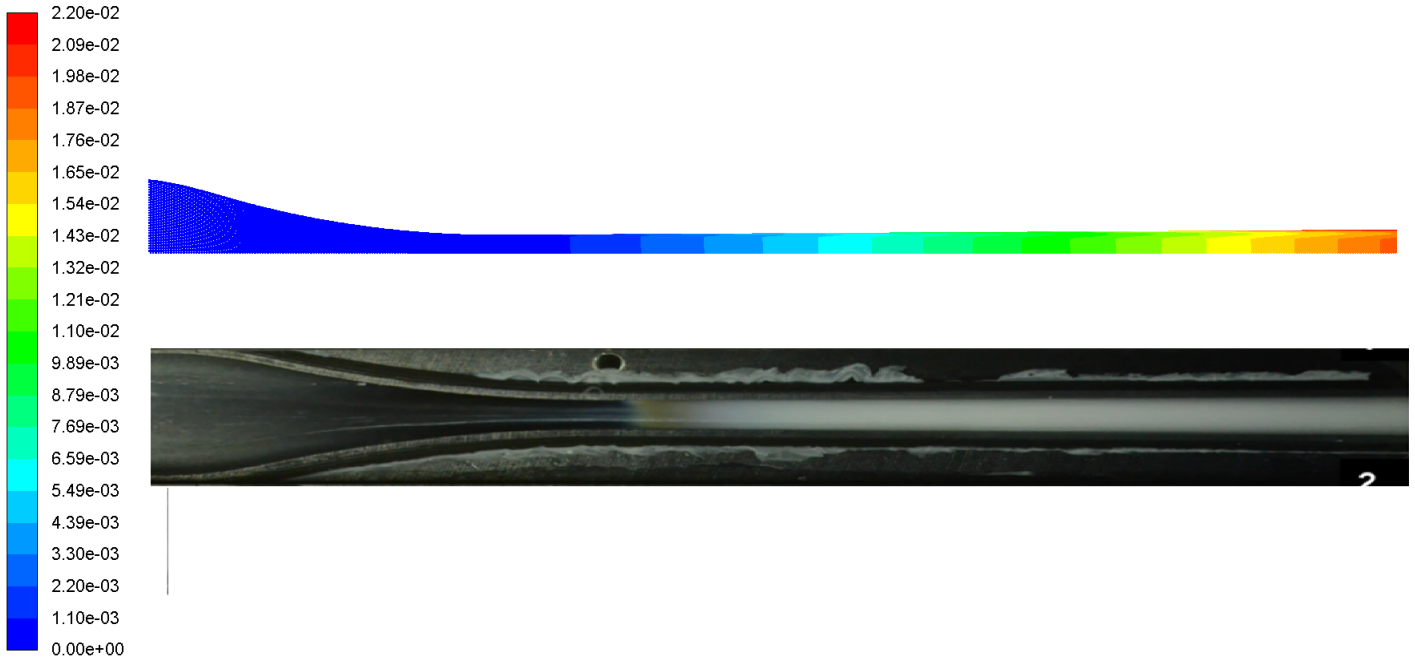


FIGURE 6.29: Liquid Mass Fraction contour for Fluent for Case 2-1, compared with the experimental reference (from [22])

assessment - which takes place in the throat or near-throat region - is thought to be mostly unaffected. In fact, the solution in the convergent, which is single-phase, matches with the two-phase solution that does not show any sign of this instability, as highlighted in Figure 6.32. As a final consequence, condensation assessment was still carried out on the unstable solutions, and results are ultimately coherent with those for Case 1-1 and Case 1-2, over-predicting condensation with respect to the experiments.

6.5 Solutions for higher total pressure cases

The same issue reported for single-phase simulations in Case 2 was obviously amplified at higher total pressures. Due to the fact that the thermodynamic region between saturation and spinodal lines is proportionally narrower, this led to local instability appearing upstream, in the near-throat region, with larger and invalidating effects on the overall simulation. Not only; expansions locally travelled below the spinodal line as well. This caused LuT queries in unfeasible areas where table compilation cannot be carried out thermodynamically. This behaviour is clearly unphysical, but it is unknown whether it is simply a temporary effect of the time-marching solution (and if the eventual final solution would have physical meaning), or if it is due to the condensation model being unable of generating physically coherent results. Higher values of the Lee Model coefficient were also tried to increase the generated quantity of condensate, without success.

Ultimately, solutions to these issues could not be determined in the present work, and may be of interest for further developments. The behaviour for the sound speed can be artificially smoothed for the near spinodal region, in order to generate a continuous and regular surface for the complete metastable vapour area. However, this is only a part of the problem, and it was deemed that it would not ultimately affect the solution *per se*.

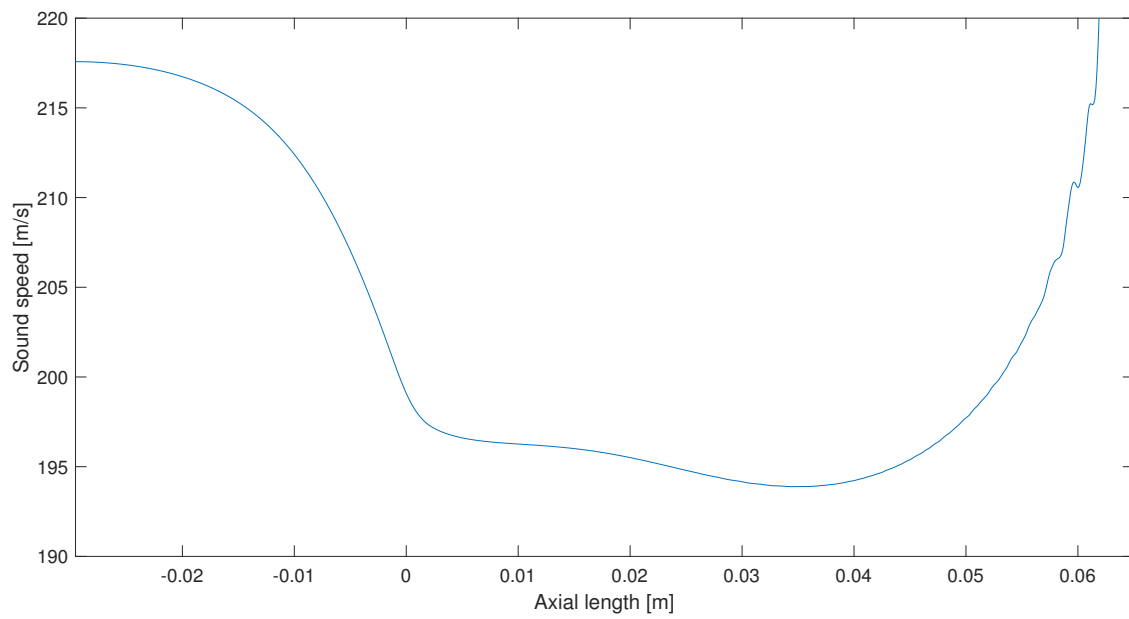


FIGURE 6.30: Single-phase sound speed for Case 2-2

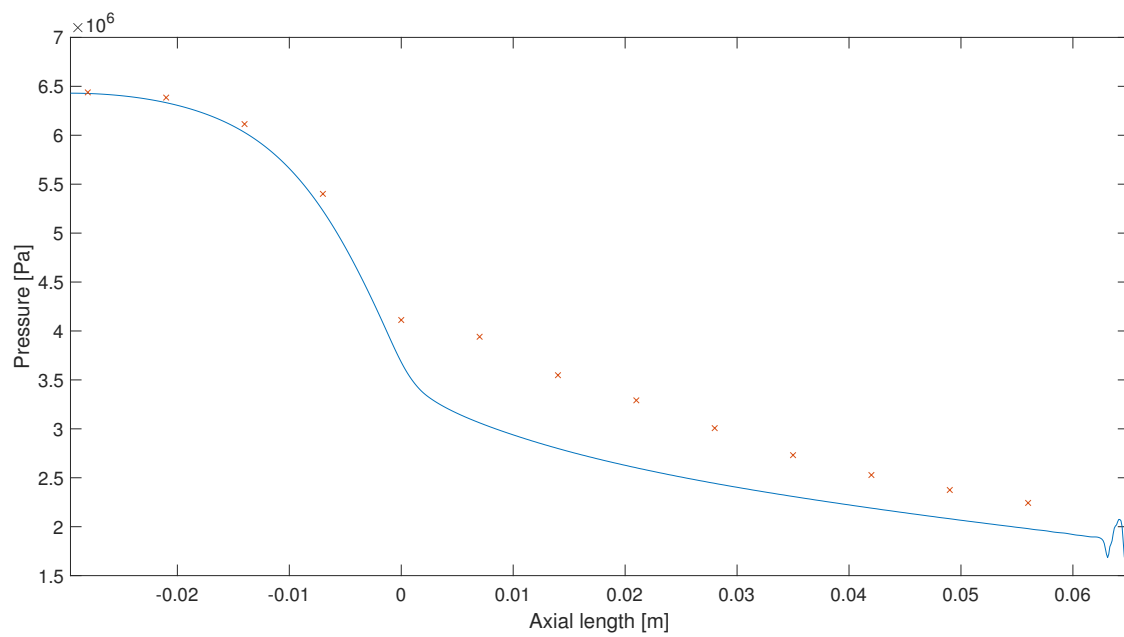


FIGURE 6.31: Single-phase pressure for Case 2-2

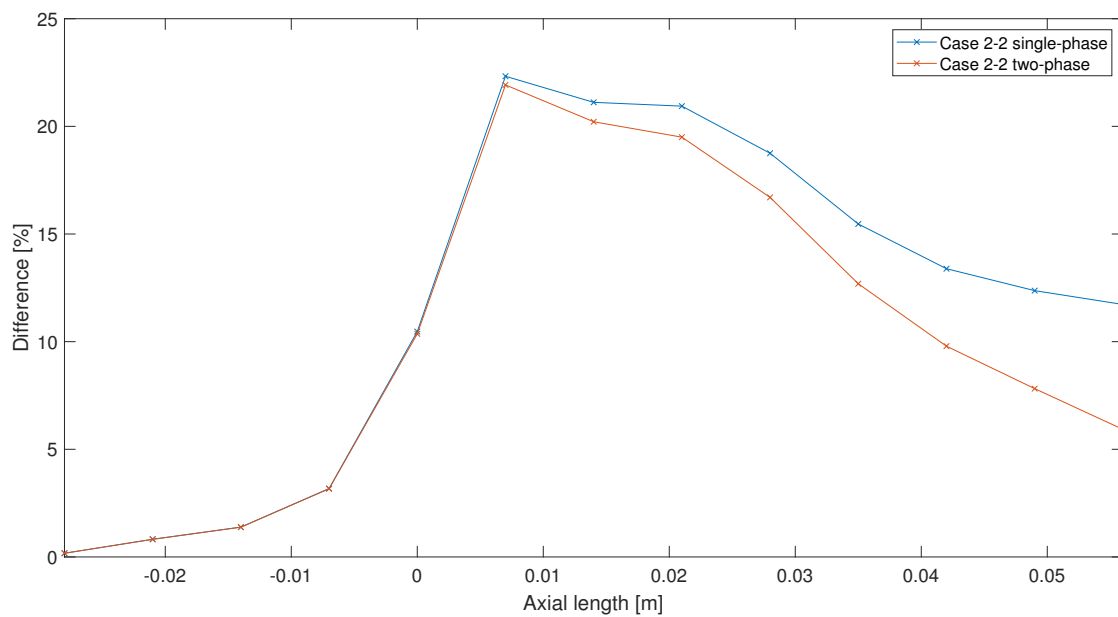


FIGURE 6.32: Single-phase and two-phase difference for Case 2-2 with experimental values, normalised with experimental values

As far as the thermodynamic calls beyond the spinodal lines are concerned, these can be dealt with if a complete and continuous 2-D surface for all vapour properties as functions of $P - T$ can be created, without the discontinuity determined by the transition line. Two approaches are possible. First, all thermodynamic properties can be mathematically extrapolated on the entire mapped region of the plane, without any concern for their physical meaning. This can be implemented quite simply in MATLAB, but requires the definition of a mathematical function to cover the unphysical areas. Attempts to generate this from fits of the physical regions, following either spline or polynomial functions, did not ultimately prove useful. Both monodimensional (line; both iso-pressure and iso-temperature approaches were attempted, corresponding to interpolation by rows and columns) and bidimensional (surface) extrapolations were attempted.

It is thought that this may not be the best solution. As briefly mentioned in Chapter 3 simple Equations of State can be defined all along the $P - T$ plane without any discontinuity (though without any physical meaning); this happens, for instance, for the ideal gas model. More accurately for our situation, the Peng-Robinson or the Lee-Kesler equations show the same behaviour. Consequently, they can be extended beyond the spinodal line. However, for thermodynamic accuracy the single-phase and metastable vapour regions require to be defined with Span-Wagner. Therefore, there would need to be a somewhat smoothed area between the metastable vapour region, defined with Span-Wagner, and the unphysical area beyond the spinodal.

In fact, Baltadijev [6] reported implementation of the Lee-Kesler equation and its use for supercritical carbon dioxide with good results, but replicating the full thermodynamic implementation of the L-K equation was deemed to be quite demanding.

It is thought, however, that even if thermodynamic effects are remarkable, these are not the only reason for convergence issues. Numerical effects due to the applied scheme are thought to be important as well. Hosangadi [16] reports that he had to apply a finer mesh for Case 3 (and one even finer for Cases 4 and 5) compared to Case 1 and 2. Its

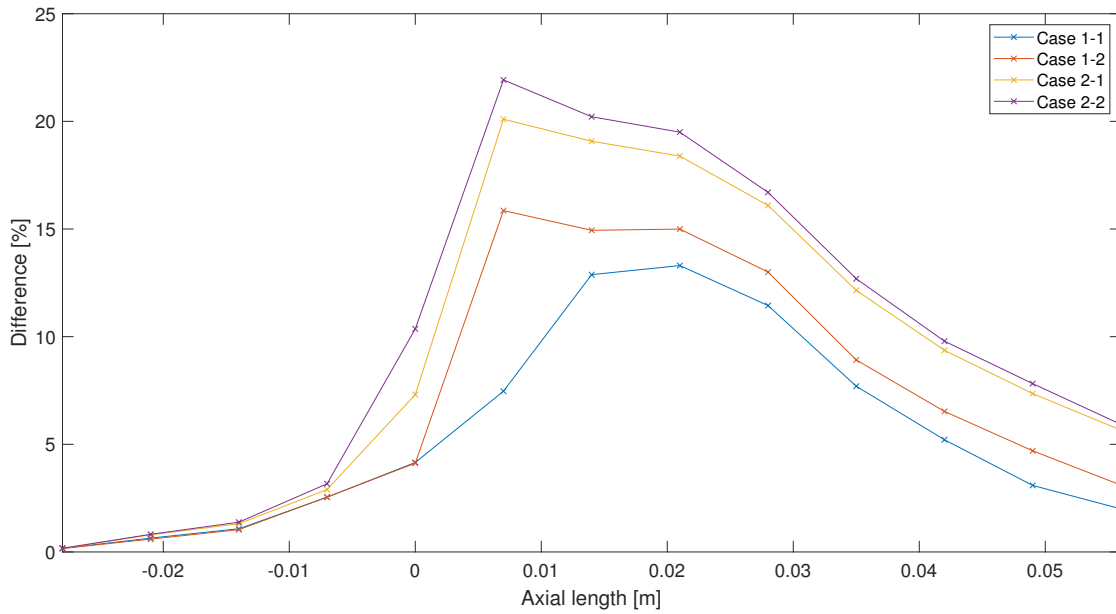


FIGURE 6.33: Differences between simulations and experiments, normalised with experimental values

spacing is comparable to Mesh 3 as reported in Chapter 5. However, this did not solve our issues, and in general his results cannot be directly translated in Fluent, since they were obtained on a different software. In fact, the author does not report issues linked to thermodynamic queries beyond the spinodal line and extrapolation as discussed just above, and his results, in terms of pressure and temperature, are physically coherent (above the spinodal line).

Ultimately, they could not be replicated in the present framework.

6.6 Solutions accuracy

The overall accuracy of all solutions in terms of pressure compared to the experimental values is reported in Figure 6.33. The errors are gradually bigger the higher the total pressure, implying that the complex thermodynamic effects that ultimately prevent solution for higher total pressure cases are already affecting lower pressure cases. Trend is similar to what was discussed for Case 1-1; once again, errors are largest in the near onset of condensation, where droplets interaction with the flow are more prevalent. Errors in the single-phase region are smallest, implying - obviously - that the thermodynamic setup yields accurate results there.

Condensation assessment for all simulated cases is reported in Table 6.2 via the procedure described for single-phase spinodal simulations in Fluent.

There is a systematic over-prediction of the condensation onset by our assessment procedure. Reasons for this behaviour were ultimately analysed above, and are thought to be mostly analogous for different simulations. Difference between experimental and numerical results is mostly coherent between cases at similar total pressure, further suggesting that the procedure itself is ultimately correct, but the simulation cannot accurately

Case	Experimental location [mm]	Numerical location [mm]	Difference [mm]
1-1	5.416	1.544	3.872
1-2	3.748	0.4924	3.256
2-1	2.022	-0.6574	2.679
2-2	1.660	-1.201	2.861

TABLE 6.2: Axial location of condensation for all simulations

model the real expansion (as also shown by the pressure differences in the convergent single-phase section, which, albeit small, are not negligible). The slight improvement in accuracy for Case 2 is thought to be due to the fact that condensation onset is more rapid at higher total pressures, because the metastable region is narrower. Not only; this shifts the onset slightly upstream in the throat, which is the region where gradients are highest. Therefore, even if there is less difference in terms of axial distance, the one in terms of thermodynamic quantities is thought to be more or less constant.

Chapter 7

Conclusions

In the course of this work, parts of an experimental campaign run by Lettieri [22] conducted to explore the phenomenon of non-equilibrium condensation for carbon dioxide and the determination of the Wilson line for the substance were simulated on three different numerical codes; ANSYS CFX, ANSYS Fluent and a quasi-1D in-house MATLAB code. The ultimate aim was the comparison of these solutions and ultimately the creation of a reliable setup to be applied to more complex geometries (i.e. turbo-compressors) making use of high pressure, or supercritical, carbon dioxide, which may experience condensation in the context of high-speed flows.

Thermodynamically speaking, all codes employed the Span-Wagner Equation of State, as implemented in REFPROP and available in MATLAB through an appropriate wrapper freely available. This was applied directly in the monodimensional code, and through $P - T$ lookup tables implemented via user-defined materials for CFX and Fluent. All lookup tables also included metastable vapour states as defined by the thermodynamic model.

Concerning condensation, CFX and the quasi-1D solution implemented the Homogeneous Equilibrium Model employing the Wilson line for transition, while Fluent simulations were performed with the Evaporation-Condensation mass transfer mechanism (Lee Model). All codes made use of the model most readily implemented and available, on the ground that simple options would need to be explored first.

Both models make use of the Eulerian-Eulerian concept; the HEM assumes thermodynamic equilibrium, while the Lee Model initiates phase transition based on source terms in the vapour transport equation (which are activated once the flow crosses the saturation line). It is worth mentioning that the HEM as implemented here is not entirely predictive, since it needs a pre-specified Wilson line which has to be found experimentally. The Lee Model, instead, has no such requirement. Both models were employed with the assumption of $P - T$ equilibrium between the phases; this is thought to be quite realistic and adherent to the physical reality, owing to the overall small dimension of nucleating droplets.

Clearly, better accuracy would come from Eulerian-Lagrangian models, which include droplet formation and growth effects; however, simulations employing them could not reach convergence, due to their complexity and instability. Further work for the development of a new model in this framework or tuning of existing ones can be considered, but this would require a high level of user interaction with the software, which may not be feasible in CFX or Fluent.

The three solutions, with and without condensation models, were analysed in detail, mainly in terms of pressure values compared to experimental data and condensation

onset location.

The quasi-1D solution showed satisfying behaviour for the single-phase subsonic flow, but due to the lack of viscous effects and boundary layers modelling its accuracy was ultimately low for the supersonic region, even without considering two-phase effects. Instead, the CFD results for single-phase regions were nearly identical between the two codes; due to the employed lookup tables, they both simulated coherently metastable states, up to the condensation onset which is physically determined by the Wilson line crossing (as represented by the B-D segment in Figure 2.1). For the sake of precision, the HEM simulates this single-phase area in a rigorously correct fashion, while the Lee Model, starting condensation at the saturation line, causes the flow to include a fraction of liquid already before crossing the Wilson line. Its overall effects on the solution are, however, negligible.

In two-phase regions, instead, Fluent and the Lee Model achieved much better results than CFX and the HEM (due to its strong assumption of thermodynamic equilibrium which constrains the flow on the defined transition line), to the point of almost replicating the experimental results in regions far from the condensation onset, where droplet effects are most important. Fluent can, therefore, quite accurately predict the actual quantity of condensate and the rate of its creation in the flow, thereby confirming that the assumption of $P - T$ equilibrium between the phases is not limiting.

The physical flow will eventually evolve back to equilibrium states, but this is expected to happen outside the domain of interest for the present work. Consequently, no assessment regarding the capacity of our models to simulate this aspect could be carried out.

If there is less interest for the accurate determination of two-phase states, a simpler condensation assessment can be carried out quite simply on single-phase simulations (i.e. without condensation models, but including metastable vapour properties). Non-equilibrium condensation is detected if the flow thermodynamic conditions happen to locally lie below the Wilson line. CFX and Fluent obtained almost identical results for this type of analysis - as also confirmed by the overall similitude of their calculations for single-phase regions highlighted above. Consequently, the only advantage to CFX lies in the intrinsic implementation simplicity of the HEM, which proved to be quite robust, albeit inaccurate. If two-phase areas are thought to have only a small size (and thus their non-equilibrium effects can be neglected), this may be of interest. However, condensation involves more than half of our test case domain, which is why the HEM is ultimately unsuitable here.

Convergence could be obtained only for Case 1 and Case 2 of the experimental reference (inlet total pressure at around 58 and 65 bar). Simulations failure for higher total pressure cases is believed to be caused by complex thermodynamic effects ultimately due to the proximity of the critical point and the spinodal line, experienced during the solution process. Higher total pressure expansion, being nearer to the critical point, tend to reduce the thermodynamic gap between saturation and spinodal lines; liquid mass fraction also tend to be higher, and two-phase regions occupy a larger portion of the domain. Non-optimal numerical settings in the CFD are also thought to play a role, considering the high specificity of the situation. Both aspects are to be further analysed and discussed; a strong extrapolation scheme capable of generating a full, smooth and continuous map of all thermodynamic properties in the $P - T$ plane, even beyond the spinodal line, is

thought to be especially significant for further analysis, as it would ensure that simulations can converge in all thermodynamic situations, therefore exonerating that side from causing problems. Use of a different Equation of State may be considered to this purpose.

Fluent is highly recommended for future studies, because of its flexibility in terms of both thermodynamic and numeric options, which proved to be crucial for the simulation of complex thermodynamic phenomena such as those described here. This led to higher accuracy, as mentioned above. Different models (for instance, different mass transfer mechanisms or Equations of State) can be implemented quite easily via User Defined Functions (written in C), whereas CFX own native language, CEL, and its user influence on schemes and models are much more limited.

The Fluent setup that was developed during the course of this work and applied in all simulations is summarised in Table 7.1.

Solver Type	Pressure-Based
Pressure-Velocity coupling	Coupled
Turbulence Model	SST $k - \omega$
Material for Vapour Phase	User Defined Real Gas Model
Material for Liquid Phase	DEFINE_PROPERTY routines
Multiphase Model	Mixture (no phase slip)
Mass Transfer Mechanism	Evaporation-Condensation

TABLE 7.1: Recap of Fluent options

Two thermodynamic aspects resulted problematic for this setup; near-critical inlet conditions and large portion of condensing domain. Their combination in the present test case causes the flow to cross complicated thermodynamic regions and involve severe non-equilibrium effects. This situation may be experienced in two-phase compressors, specifically designed to cause condensation inside the flow. In fact, this is demanded by some s-CO₂ cycle layouts.

However, most s-CO₂ compressors (such as the machine for the SANDIA loop) are designed to be single-phase machines, where condensation is required to either not happen at all or influence only small flow regions, and therefore metastable penetration is small. In particular, if the spinodal line or its proximity is not involved thermodynamically, this setup may already yield satisfying results.

Appendix A

Geometry

The geometry used for all simulations is reported here. The reference system has its origin in the throat ($x = 0$); the convergent is represented by negative axial coordinate values, while the divergent is represented by positive x values.

X [mm]	Y [mm]
-29.5495	6.260231892
-28.6486	6.141914908
-27.7477	5.986616334
-26.8468	5.799913733
-25.9459	5.587384668
-25.0450	5.354606702
-24.1441	5.107157633
-23.2432	4.851022317
-22.3423	4.593427979
-21.4414	4.341838025
-20.5405	4.103361223
-19.6396	3.879821292
-18.7387	3.668984418
-17.8378	3.468535712
-16.9369	3.277293462
-16.0360	3.095712811
-15.1351	2.924370595
-14.2342	2.763533512
-13.3333	2.613014569
-12.4324	2.47259684
-11.5315	2.342164159
-10.6306	2.221683056
-9.7297	2.111127266
-8.8288	2.010489157
-7.9279	1.919762442
-7.0270	1.8388991
-6.1261	1.76782321
-5.2252	1.706494688
-4.3243	1.654915459
-3.4234	1.613110038
-2.5225	1.58111552
-1.6216	1.558789014
-0.7207	1.545995739
0.2596	1.543447375
1.5575	1.549141701
2.8554	1.556712293
4.1532	1.564198517
5.4511	1.571717041
6.7490	1.579245961
8.0469	1.586759615
9.3448	1.594260351
10.6427	1.601752905
11.9406	1.609242216
13.2385	1.616734744
14.5364	1.624240382
15.8342	1.631766518
17.1321	1.639298942

18.4300	1.646813969
19.7279	1.65430129
21.0258	1.661781513
22.3237	1.669279352
23.6216	1.676803612
24.9195	1.684337339
26.2173	1.691861248
27.5152	1.69936448
28.8131	1.706856754
30.1110	1.714350872
31.4089	1.721858406
32.7068	1.729378799
34.0047	1.7369046
35.3026	1.744428275
36.6005	1.751943622
37.8983	1.759451119
39.1962	1.766953344
40.4941	1.774452872
41.7920	1.781952126
43.0899	1.789452266
44.3878	1.796953829
45.6857	1.804457348
46.9836	1.811963359
48.2815	1.819472593
49.5793	1.82698621
50.8772	1.83450543
52.1751	1.842031471
53.4730	1.849565532
54.7709	1.857105556
56.0688	1.86464236
57.3667	1.872165826
58.6646	1.879665832
59.9624	1.887132258
61.2603	1.894557901
62.5582	1.901963439
63.8561	1.90938505
65.1540	1.916859075
66.4519	1.924421859
67.7498	1.932109742
69.0477	1.939949835
70.3456	1.947875861
71.6434	1.955767316
72.9413	1.963503031
74.2392	1.970961836
75.5371	1.97802256
76.8350	1.984564033

Appendix B

Wilson Line

The Wilson Line obtained by Lettieri [22] and extrapolated as described in Chapter 2 is reported here.

T [K]	P [kPa]
216.592	517.964343
217.260017	770.349316
218.137372	801.241646
219.014727	831.659883
219.892083	861.828072
220.769438	891.964503
221.646793	922.152028
222.524149	952.494662
223.401504	983.074038
224.27886	1013.974487
225.156215	1045.2258
226.03357	1076.857988
226.910926	1108.899706
227.788281	1141.371635
228.665637	1174.287882
229.542992	1207.657298
230.420347	1241.484825
231.297703	1275.772916
232.175058	1310.523076
233.052414	1345.737469
233.929769	1381.420513
234.807124	1417.580314
235.68448	1454.229778
236.561835	1491.38725
237.43919	1529.076589
238.316546	1567.327567
239.193901	1606.175819
240.071257	1645.651863
240.948612	1685.792476
241.825967	1726.635035
242.703323	1768.212344
243.580678	1810.555611
244.458034	1853.692414
245.335389	1897.647181
246.212744	1942.440888
247.0901	1988.085944
247.967455	2034.600857
248.844811	2081.994365
249.722166	2130.265468
250.599521	2179.426076
251.476877	2229.4787
252.354232	2280.424643
253.231587	2332.264764
254.108943	2385.000214
254.986298	2438.633141
255.863654	2493.167336
256.741009	2548.598746
257.618364	2604.931611
258.49572	2662.185868

259.373075	2720.365552
260.250431	2779.466669
261.127786	2839.533235
262.005141	2900.539475
262.882497	2962.540217
263.759852	3025.536094
264.637208	3089.537293
265.514563	3154.58322
266.391918	3220.689939
267.269274	3301.96729
268.146629	3384.940922
269.023984	3468.645208
269.90134	3553.037766
270.778695	3638.076735
271.656051	3723.721227
272.533406	3809.931798
273.410761	3896.670926
274.288117	3983.903531
275.165472	4071.597528
276.042828	4159.724432
276.920183	4248.241451
277.797538	4337.004656
278.674894	4426.121443
279.552249	4515.587585
280.429605	4605.400905
281.30696	4695.563802
282.184315	4785.922618
283.061671	4876.637979
283.939026	4967.746357
284.816381	5059.148724
285.693737	5150.86262
286.571092	5243.11153
287.448448	5335.706415
288.325803	5428.854959
289.203158	5522.65337
290.080514	5616.875259
290.957869	5712.004415
291.835225	5807.729423
292.71258	5904.363527
293.589935	6002.310611
294.467291	6100.852274
295.344646	6200.703716
296.222002	6301.983193
297.099357	6404.760935
297.976712	6512.998899
298.854068	6625.229977
299.731423	6740.783109
300.608778	6858.214561
301.705473	7009.345589

Appendix C

Fluent Lookup Tables Generator

The table generator used for generating ASCII files subsequently fed to the C UDFs (detailed in Appendix D) is reported. Written in MATLAB, it generates a folder with 24 ASCII files; 22 tables (11 for each phase), plus arrays for temperature and pressure. The extrapolation scheme beyond the transition line is defined in the function `matrix_extr`, also reported below. It can be modified to express the desired behaviour, though its use did not lessen the issues encountered in this work. A simple scheme operating a spline interpolation of each column (therefore operating an extrapolation of lines representing thermodynamic properties at same pressure, rather than the extrapolation of the full surface) is here reported.

As already discussed, all thermodynamic calls are carried out with the `refpropm` function.

Two slight modifications - the first of which is fundamental - are made with respect to the reference version:

- Added line `phaseFlag = varargin{7}`; at line 334 to allow for the call of metastable properties directly from any script making use of `refpropm`;
- Commented lines 537-539 (in the original version) to force solution in some special situations.

```
clc
close all
clear all

% Generation of ASCII LUTs to be read by FLUENT C UDF as thermodynamic
% input. LUTs can be clipped to the desired line.

n = 400;           % Discretisation for T
m = 400;           % Discretisation for P
pmin_sh = 10*100;  % [KPa]
pmax_sh = 65*100;
Tmin_sh = 220;     % [K]
Tmax_sh = 340;
fluid = 'CO2';
mod_flag = 2;      % 0: saturation; 1: Wilson; 2: spinodal; 3: liquid spinodal (upper)
interp_flag = 0;   % 0: clip at the line; 1: Interpolant beyond the line
liq_gen_flag = 0;  % 0: No LIQ tab; 1: LIQ tab;
vap_gen_flag = 1;  % 0: No VAP tab; 1: VAP tab;

Tminclip = 210;    % Extrapolating near the triple point of CO2

% Loading Wilson line

Wilson = load('wilsonco2');
Twil = Wilson(:, 1);
```

```

Pwil = Wilson (:, 2);

% Generation of T / P grid

T = linspace (Tmin_sh, Tmax_sh, n);
P = linspace (pmin_sh, pmax_sh, m);

[Pgrid, Tgrid] = meshgrid(P,T);

% Evaluation of critical and triple point properties

Tcr = refpropm ('T', 'C', 0, ' ', 0, fluid, 0);
Pcr = refpropm ('P', 'C', 0, ' ', 0, fluid, 0);
Ttr = refpropm ('T', 'R', 0, ' ', 0, fluid, 0);
Ptr = refpropm ('P', 'R', 0, ' ', 0, fluid, 0);

switch mod_flag
    case 0
        fprintf ('Writing saturation tables. \n');
    case 1
        fprintf ('Writing Wilson tables (for CO2) \n');
    case 2
        fprintf ('Writing lower (vapour) spinodal tables \n');
    case 3
        fprintf ('Writing upper (liquid) spinodal tables \n');
end

% Initial grid allocation for efficient computation
% starting zero grids are necessary for subsequent extrapolation, NB!

enth_v = zeros(n, m);
speed_v = zeros(n, m);
dens_v = zeros(n, m);
cp_v = zeros(n, m);
entr_v = zeros (n, m);
visco_v = zeros (n, m);
cond_v = zeros (n, m);
rhot_v = zeros (n, m);
rhop_v = zeros (n, m);
enth_t_v = zeros (n, m);
enth_p_v = zeros (n, m);

if vap_gen_flag == 1

    offset = 0;

    flag = -2;

    fprintf ('Generation of vapour tables \n');
    fprintf ('Number of T / P discretisations: %d %d \n', n, m);

    fprintf ('In progress: \n');

    dt = 0.01;
    dp = 1;

    for i=1:n

        for j = 1:m

            % Evaluation of spinodal properties to determine where condition
            % i, j lies. Obviously this is necessary only if i, j is not a
            % supercritical condition

            if (Pgrid (i, j) < Pcr && Tgrid (i, j) < Tcr && Tgrid (i, j) > Tminclip)
                flag = -2;
                switch mod_flag
                    case 0
                        flag = 0;
                        Pclip = refpropm ('P', 'T', Tgrid (i, j), 'Q', 1, fluid, 0);

```

```

case 1
    flag = -2;
    Pclip = interp1 (Twil, Pwil, Tgrid (i, j));

case 2
    % Adding a dp to ensure we fall in the correct phase
    % when evaluating
    flag = -2;
    d_sp = refpropm ('J', 'T', Tgrid (i, j), 'Q', 1, fluid, 0);
    P_sp = refpropm ('P', 'T', Tgrid (i, j), 'D', d_sp, fluid, -2);
    Pclip = P_sp - dp;
case 3
    flag = -1;
    d_sp = refpropm ('J', 'T', Tgrid (i, j), 'Q', 1, fluid, 0);
    P_sp = refpropm ('P', 'T', Tgrid (i, j), 'D', d_sp, fluid, -2);
    Pclip = P_sp;
end
else
    Pclip = Inf; % Inf -> eq liquid prop inclusion; Pcr no
    flag = 0;
end

if Tgrid (i, j) > Tminclip
    if Pgrid (i, j) > Pclip && Tgrid (i, j) < Tcr
        if interp_flag == 0 % Clipping at desired values (else leaving blank)
            dens_v (i, j) = refpropm ('D', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            enth_v (i, j) = refpropm ('H', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            entr_v (i, j) = refpropm ('S', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            cp_v (i, j) = refpropm ('C', 'T', Tgrid (i, j), 'P', Pclip, fluid,
flag);
            speed_v (i, j) = refpropm ('A', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            visco_v (i, j) = refpropm ('V', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            cond_v (i, j) = refpropm ('L', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            rhot_v (i, j) = refpropm ('W', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            rhop_v (i, j) = refpropm ('R', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag)/1000;
            entht_v (i, j) = refpropm ('(', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
            enthp_v (i, j) = refpropm ('*', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag)/1000;
        end
    else
        % Evaluation of properties according to "Flag" as defined above.
        dens_v (i, j) = refpropm ('D', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        enth_v (i, j) = refpropm ('H', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        entr_v (i, j) = refpropm ('S', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        cp_v (i, j) = refpropm ('C', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        speed_v (i, j) = refpropm ('A', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        visco_v (i, j) = refpropm ('V', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        cond_v (i, j) = refpropm ('L', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        rhot_v (i, j) = refpropm ('W', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
        rhop_v (i, j) = refpropm ('R', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag)/1000;
        entht_v (i, j) = refpropm ('(', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
    end
end

```

```

        enthp_v (i, j) = refpropm ('*', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag)/1000;
    end
end

end
perc = round (i*100/n);
if perc > offset
    fprintf ('%d%% ', perc-1);
    offset = offset + 10;
end

end
fprintf ('\n');
fprintf ('Vapour grids generated. \n');

end

% Generation of liquid tables – mostly analogous
% Starting zero grids are necessary for subsequent extrapolation, NB!

enth_l = zeros(n, m);
speed_l = zeros(n, m);
dens_l = zeros(n, m);
cp_l = zeros(n, m);
entr_l = zeros (n, m);
visco_l = zeros (n, m);
cond_l = zeros (n, m);
rhot_l = zeros (n, m);
rhop_l = zeros (n, m);
enth_l = zeros (n, m);
enthp_l = zeros (n, m);

flag = -1;

offset = 0;

fprintf ('\n');
fprintf ('Generation of liquid tables \n');
fprintf ('Number of T / P discretisations: %d %d \n', n, m);
fprintf ('In progress: \n');

if liq_gen_flag == 1

for i=1:n

    for j = 1:m

        if (Tgrid (i, j) < Tcr && Tgrid (i, j) > Tminclip && Pgrid (i, j) < Pcr)
            flag = -1;
            switch mod_flag
                case 0
                    Pclip = refpropm ('P', 'T', Tgrid (i, j), 'Q', 1, fluid, 0);

                case 1
                    Pclip = interp1 (Twil, Pwil, Tgrid (i, j));

                case 2
                    d_sp = refpropm (']', 'T', Tgrid (i, j), 'Q', 1, fluid, 0);
                    P_sp = refpropm ('P', 'T', Tgrid (i, j), 'D', d_sp, fluid, -2);
                    Pclip = P_sp;

                case 3
                    % Adding DP to ensure we're in liquid phase
                    d_sp = refpropm ('[', 'T', Tgrid (i, j), 'Q', 1, fluid, 0);
                    P_sp = refpropm ('P', 'T', Tgrid (i, j), 'D', d_sp, fluid, -2);
                    Pclip = P_sp + dp;

            end
        else
            flag = 0;

```

```

        Pclip = 0;
    end

    if Tgrid (i, j) > Tminclip
        if Pgrid (i, j) < Pclip && Tgrid (i, j) < Tcr
            if interp_flag == 0 % Clipping (else leaving blank)
                dens_l (i, j) = refpropm ('D', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                enth_l (i, j) = refpropm ('H', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                entr_l (i, j) = refpropm ('S', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                cp_l (i, j) = refpropm ('C', 'T', Tgrid (i, j), 'P', Pclip, fluid,
flag);
                speed_l (i, j) = refpropm ('A', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                visco_l (i, j) = refpropm ('V', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                cond_l (i, j) = refpropm ('L', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                rhot_l (i, j) = refpropm ('W', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                rhop_l (i, j) = refpropm ('R', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag)/1000;
                entht_l (i, j) = refpropm ('(', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag);
                enthp_l (i, j) = refpropm ('*', 'T', Tgrid (i, j), 'P', Pclip,
fluid, flag)/1000;
            end
        else
            dens_l (i, j) = refpropm ('D', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            enth_l (i, j) = refpropm ('H', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            entr_l (i, j) = refpropm ('S', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            cp_l (i, j) = refpropm ('C', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            speed_l (i, j) = refpropm ('A', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            visco_l (i, j) = refpropm ('V', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            cond_l (i, j) = refpropm ('L', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            rhot_l (i, j) = refpropm ('W', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            rhop_l (i, j) = refpropm ('R', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag)/1000;
            entht_l (i, j) = refpropm ('(', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag);
            enthp_l (i, j) = refpropm ('*', 'T', Tgrid (i, j), 'P', Pgrid (i, j),
fluid, flag)/1000;
        end
    end
end
end
perc = round (i * 100 / n);
if perc > offset
    fprintf ('%d%% ', perc-1);
    offset = offset + 10;
end
end
fprintf ('\n');
fprintf ('Liquid grids generated. ');

end

grid = {dens_v, dens_l, enth_v, enth_l, entr_v, entr_l, cp_v, cp_l, speed_v, ...
speed_l, visco_v, visco_l, cond_v, cond_l, rhot_v, rhot_l, rhop_v, rhop_l, ...
entht_v, entht_l, enthp_v, enthp_l};

prop = [ 'D', 'D', 'H', 'H', 'S', 'S', 'C', 'C', 'A', 'A', 'V', 'V', 'L', 'L', 'W', 'W',
'R', 'R', '(', '(', ...

```

```

    '*' , '*' ];
flagvec = [-2 -1];
flagvec = repmat (flagvec , 1, 11);

% Extrapolating if required

if interp_flag == 1
fprintf ('Extrapolating grids. \n');
for i = 1:length (grid)
    j = i;
    fprintf ('Extrapolating grid: %d \n', j);
    matr = cell2mat (grid (j));
    matr_ex = matrix_extr (matr, Tgrid, Pgrid, fluid, prop(i), flagvec(i));
    grid (j) = mat2cell (matr_ex, n, m);
    if j == 1 || j == 3 || j == 5 || j == 7 || j == 9
        figure
        surf (Pgrid, Tgrid, matr)
        figure
        surf (Pgrid, Tgrid, matr_ex);
    end
end
end

% ASCII files writing section

fprintf ('Now writing grids. \n');

switch mod_flag
case 0
    appendix_mod = ('sat');
case 1
    appendix_mod = ('wil');
case 2
    appendix_mod = ('spin');
case 3
    appendix_mod = ('spinliq');
end

switch interp_flag
case 0
    appendix_int = ('_clip');
case 1
    appendix_int = ('_int');
end

appendix = strcat (appendix_mod, appendix_int);

name_file = ["dens_v", "dens_l", "enth_v", "enth_l", "entr_v", "entr_l", "cp_v",
    "cp_l", "speed_v", ...
    "speed_l", "visco_v", "visco_l", "cond_v", "cond_l", "rhot_v", "rhot_l", "rhop_v",
    "rhop_l", ...
    "enth_v", "enth_l", "enthp_v", "enthp_l"];

% Creating folder

folder_name = strcat ('lut', appendix);
mkdir (folder_name);

% Writing T - P arrays

filenamet = fullfile (folder_name, 'temparr');
fid = fopen (filenamet, 'w');
fprintf (fid, '%14.10e ', T);
fclose (fid);

filenamep = fullfile (folder_name, 'pressarr');
fid = fopen (filenamep, 'w');
fprintf (fid, '%14.10e ', P*1000);
fclose (fid);

```

```
% Writing tables

for i = 1:length(name_file)
    name = strcat (name_file (i), '_', appendix);
    filename = fullfile (folder_name, name);
    fid = fopen (filename, 'w');
    mat = cell2mat (grid (i));
    fprintf (fid, '%14.10e ', mat);
    fclose(fid);
end

fprintf ('Finished with success! \n');

function [F] = matrix_extr (F, Tgrid, Pgrid, prop)

F (F == 0) = NaN;
F = fillmissing (F, 'spline', 2);

end
```

The MATLAB scripts for the generation of the saturation and spinodal line to be fed to the DEFINE_PROPERTY UDFs are provided, alongside a script for the generation of the arrays of specific heat and enthalpy for the DEFINE_SPECIFIC_HEAT routine. These follow very much the structure of the Lookup Table generator that was provided earlier, and make use of `refpropm` for the thermodynamic calls.

*% Writing ASCII for Saturation table as P-T arrays where P will be used as
% independent variable (hence must be built with linspace).*

```
clc
clear all
close all

fluid = 'CO2';

flag = 0;
Tcr = refpropm ('T', 'C', 0, ' ', 0, fluid, 0);
Ttr = refpropm ('T', 'R', 0, ' ', 0, fluid, 0);
Pcr = refpropm ('P', 'C', 0, ' ', 0, fluid, 0);
Ptr = refpropm ('P', 'R', 0, ' ', 0, fluid, 0);

dim = 300;
Plut = linspace (Ptr, Pcr - 0.01, dim);

for i = 1:dim
    Tlut (i) = refpropm ('T', 'P', Plut (i), 'Q', 1, fluid, 0);
end

name = 'sat_lut';
fid = fopen (name, 'w');
fprintf (fid, '%14.10f ', Tlut);
fprintf (fid, '%14.10f ', Plut*10^3);
fclose (fid);
```

*% Writing ASCII for Vapour Spinodal table as P-T arrays where P will be used as
% independent variable (hence must be built with linspace).*

*% Calling T spinodal as function of P is not directly possible -> first
% creating T array, then P array, then using those to interpolate and
% create T array on equally spaced P*

```
clc
clear all
close all

fluid = 'CO2';

flag = 0;
Tcr = refpropm ('T', 'C', 0, ' ', 0, fluid, 0);
Ttr = refpropm ('T', 'R', 0, ' ', 0, fluid, 0);
Pcr = refpropm ('P', 'C', 0, ' ', 0, fluid, 0);
Ptr = refpropm ('P', 'R', 0, ' ', 0, fluid, 0);

dim = 300;
T = linspace (Ttr, Tcr - 0.01, dim);

for i = 1:dim
    d_sp (i) = refpropm (' ', 'T', T (i), 'P', 0, fluid, 0);
    P (i) = refpropm ('P', 'T', T (i), 'D', d_sp (i), fluid, -2);
end

Plut = linspace (P (1), P (end) - 0.01, dim);

Tlut = interp1 (P, T, Plut, 'linear', 'extrap');

% figure
% plot (T, P)
% hold on
% plot (Tlut, Plut)

name = 'spin_lut';
fid = fopen (name, 'w');
```

```

fprintf (fid , '%14.10f ' , Tlut);
fprintf (fid , '%14.10f ' , Plut*10^3);
fclose (fid);

% Writing CP - H lut for DEFINE_SPECIFIC_HEAT UDF on FLUENT. Only T
% accepted as input.
% P for definition is taken - arbitrarily - as Psat.
% This LuT is technically independent from the one generated in ASCII_GEN.
% However for coherence T bounds and density are kept as the same. Same for
% choice of interpolating.

clc
clear all
close all

dim = 300;          % Discretisation for T

Tmin_sh = 218;      % [K]
Tmax_sh = 340;
fluid = 'CO2';
interp_flag = 0;    % 0: no interpolation; 1: interpolation

Teval = linspace (Tmin_sh, Tmax_sh, dim);
Tminclip = 210;

cp = zeros (1, dim);
h = zeros (1, dim);

% For extrapolation stability: Tcr is set at Tcr - 8 for extrapolation.

switch interp_flag
case 0
    Tcr = refpropm ('T', 'C', 0, ' ', 0, fluid, 0) - 8;
    Pcr = refpropm ('P', 'C', 0, ' ', 0, fluid, 0);
case 1
    Tcr = refpropm ('T', 'C', 0, ' ', 0, fluid, 0) - 8;
    Pcr = refpropm ('P', 'T', Tcr, 'Q', 1, fluid, 0);
end

for i = 1:dim
    if Teval (i) < Tcr && Teval (i) > Tminclip
        Peval = refpropm ('P', 'T', Teval (i), 'Q', 1, fluid, 0);
        h (i) = refpropm ('H', 'T', Teval (i), 'P', Peval, fluid, -1);
        cp (i) = refpropm ('C', 'T', Teval (i), 'P', Peval, fluid, -1);
    elseif Teval (i) >= Tcr
        h (i) = 1;
    elseif Teval (i) <= Tminclip
        h (i) = 2;
    end
end

figure
plot (Teval, h);
hold on

index_low = find (h == 2);
index_high = find (h == 1);

switch interp_flag
case 0 % Clipping at Tcr
    if any (index_low) ~= 0
        h (index_low) = ones (1, length(index_low)) * h(index_low(end) + 1);
        cp (index_low) = ones (1, length(index_low)) * cp(index_low(end) + 1);
    end
    if any (index_high) ~= 0
        h (index_high) = ones (1, length(index_high)) * h(index_high(1) - 1);
        cp (index_high) = ones (1, length(index_high)) * cp(index_high(1) - 1);
    end
case 1 % Extrapolation between Tcr and Tmax
    if any (index_low) ~= 0
        n_low = index_low (end) + 1;

```

```

        else
            n_low = 1;
        end
        if any (index_high) ~= 0
            n_high = index_high (1) - 1;
        else
            n_high = n;
        end
        if any (index_low) ~= 0
            h (index_low) = interp1 (Teval (n_low:n_high), h (n_low:n_high), Teval
(1:n_low - 1), 'spline');
            cp (index_low) = interp1 (Teval (n_low:n_high), cp (n_low:n_high), Teval
(1:n_low - 1), 'spline');
        end
        if any (index_high) ~= 0
            h (index_high) = interp1 (Teval (n_low:n_high), h (n_low:n_high), Teval
(n_high+1:end), 'spline');
            cp (index_high) = interp1 (Teval (n_low:n_high), cp (n_low:n_high), Teval
(n_high+1:end), 'spline');
        end
    end
end

plot (Teval, h)

% Writing
name = 'cp_h_liq_lut';
fid = fopen (name, 'w');
fprintf (fid, '%14.10f ', Teval);
fprintf (fid, '%14.10f ', cp);
fprintf (fid, '%14.10f ', h);
fclose (fid);

```

Appendix D

User Defined Functions

Here the UDFs described in the course of the work as implemented in Fluent are reported. First the User Defined Real Gas Model is provided, then the C file containing all the necessary UDFs for the other properties is provided. In order to work, they need the table generator and script provided in Appendix C.

```
// User Defined Real Gas Model for Real Gas Properties of Vapour Phase of CO2

/*
 * Windows Warning!!! Including udf.h is for getting definitions for
 * ANSYS FLUENT constructs such as Domain. You must
 * NOT reference any ANSYS FLUENT globals directly from
 * within this module nor link this against any ANSYS
 * FLUENT libs, doing so will cause dependencies on a
 * specific ANSYS FLUENT binary such as fl551.exe and
 * thus won't be version independent.
 */

#include "udf.h"
#include "stdio.h"
#include "ctype.h"
#include "stdarg.h"

#if RP_DOUBLE
#define SMALL 1.e-20
#else
#define SMALL 1.e-10
#endif

// Here the LuT size and number of properties is defined (11 in total)

#define DISCR_T 300
#define DISCR_P 300
#define NUMPROP 11

static int (*usersMessage)(char *,...);
static void (*usersError)(char *,...);

// Declaration of static variables, including pointer to array char for file paths -
// sent as names to FOPEN

static double DT, DP, Tmin, Pmin, Tmax, Pmax;
static double properties [NUMPROP] [DISCR_T] [DISCR_P];
static double Tarr [DISCR_T];
static double Parr [DISCR_P];
static char *namefile [NUMPROP];
static char *Tname, *Pname;

double Interpolant(double Temp, double P, int index);

DEFINE_ON_DEMAND(I_do_nothing)
{
}
```

```

void CO2MONOLUT_error(int err, char *f, char *msg)
{
    if (err)
        usersError("CO2MONOLUT_error (%d) from function: %s\n%s\n",err,f,msg);
}

// Operate a substitution of spin_clip with the desired string (wil_clip or sat_int for
// instance) to switch to the desired set
// of LuTs as generated by the ASCII_gen script

void CO2MONOLUT_Setup(Domain *domain, cxboolean vapor_phase, char *filename, int
    (*messagefunc)(char *format, ...),
    void (*errorfunc)(char *format, ...))
{ /* Use this function for any initialization or model setups*/
    usersMessage = messagefunc;
    usersError = errorfunc;
    usersMessage("\nLoading spin_clip Library: %s\n", filename);

    namefile [0] = ".\\lutspin_clip\\dens_v_spin_clip";
    namefile [1] = ".\\lutspin_clip\\cp_v_spin_clip";
    namefile [2] = ".\\lutspin_clip\\enth_v_spin_clip";
    namefile [3] = ".\\lutspin_clip\\entr_v_spin_clip";
    namefile [4] = ".\\lutspin_clip\\speed_v_spin_clip";
    namefile [5] = ".\\lutspin_clip\\visco_v_spin_clip";
    namefile [6] = ".\\lutspin_clip\\cond_v_spin_clip";
    namefile [7] = ".\\lutspin_clip\\rhot_v_spin_clip";
    namefile [8] = ".\\lutspin_clip\\rhov_v_spin_clip";
    namefile [9] = ".\\lutspin_clip\\enth_t_v_spin_clip";
    namefile [10] = ".\\lutspin_clip\\enth_p_v_spin_clip";
    Tname = ".\\lutspin_clip\\temparr";
    Pname = ".\\lutspin_clip\\pressarr";

    FILE *Tpoint, *Ppoint, *Fpoint;
    Tpoint = fopen (Tname, "r+");
    Ppoint = fopen (Pname, "r+");

    int i, j, q;

    // Reading T, P

    for (i=0; i<DISCR_T; i++)
        {fscanf (Tpoint, "%lf", &Tarr [i]);}
    fclose (Tpoint);
    for (i=0; i<DISCR_P; i++)
        {fscanf (Ppoint, "%lf", &Parr [i]);}
    fclose (Ppoint);

    Message ("T and P arrays loaded \n");

    DT = Tarr [1] - Tarr [0];
    DP = Parr [1] - Parr [0];
    Tmin = Tarr [0];
    Pmin = Parr [0];
    Tmax = Tarr [DISCR_T - 1];
    Pmax = Parr [DISCR_P - 1];

    // Reading properties from ASCII tables

    for (i=0; i<NUMPROP; i++)
    {
        Fpoint = fopen (namefile[i], "r+");
        for (j=0; j<DISCR_T; j++)
        {
            for (q=0; q<DISCR_P; q++)
            {
                fscanf (Fpoint, "%lf", &properties [i][q][j]);
            }
        }
        Message ("LuT %d loaded \n", i);
        fclose (Fpoint);
    }
}

```

```

}

// NB position of function declaration must be BEFORE the function itself is called

double Interpolant(double Temp, double P, int index)
{
    int indTlow, indThigh, indPlow, indPhigh;

    // Handling out of bounds T, P and effectively clipping them and the relative index

    if (Temp < Tmin)
    {
        indTlow = 0;
        indThigh = 0;
    }
    else if (Temp > Tmax)
    {
        indTlow = DISCR_T - 1;
        indThigh = DISCR_T - 1;
    }
    else
    {
        indTlow = floor((Temp - Tmin) / DT);
        indThigh = ceil((Temp - Tmin) / DT);
    }

    if (P < Pmin)
    {
        indPlow = 0;
        indPhigh = 0;
    }
    else if (P > Pmax)
    {
        indPlow = DISCR_P - 1;
        indPhigh = DISCR_P - 1;
    }
    else
    {
        indPlow = floor((P - Pmin) / DP);
        indPhigh = ceil((P - Pmin) / DP);
    }

    double A_V, B_V, C_V, D_V;
    double Tlow, Thigh, Plow, Phigh;
    double interp_tlow, interp_thigh, Interpolated;

    Tlow = Tarr[indTlow];
    Thigh = Tarr[indThigh];
    Plow = Parr[indPlow];
    Phigh = Parr[indPhigh];

    A_V = properties[index][indTlow][indPlow];
    B_V = properties[index][indThigh][indPlow];
    C_V = properties[index][indTlow][indPhigh];
    D_V = properties[index][indThigh][indPhigh];

    // Handling case where T, P might exactly be on the array -> linear interpolation or no
    // interpolation at all

    if (indTlow == indThigh && indPlow != indPhigh)
    {
        Interpolated = (Phigh - P) / DP * B_V + (P - Plow) / DP * D_V;
    }
    else if (indPlow == indPhigh && indTlow != indThigh)
    {
        Interpolated = (Thigh - Temp) / DT * A_V + (Temp - Tlow) / DT * B_V;
    }
    else if (indPlow == indPhigh && indTlow == indThigh)
    {
        Interpolated = A_V;
    }
    else
    {
        // Standard case - bilinear interpolation: first on T then on P
        interp_tlow = (Thigh - Temp) / DT * A_V + (Temp - Tlow) / DT * B_V;
        interp_thigh = (Thigh - Temp) / DT * C_V + (Temp - Tlow) / DT * D_V;

        Interpolated = (Phigh - P) / DP * interp_tlow + (P - Plow) / DP * interp_thigh;
    }

    return Interpolated;
}

double CO2MONOLUT_density(cxboolean vapor_phase, double Temp, double P, double yi[])
{

```

```

    int ind = 0;
    double r = Interpolant (Temp, P, ind);
    return r;      /* (Kg/m^3) */
}

double CO2MONOLUT_specific_heat(double Temp, double density, double P, double yi[])
{
    int ind = 1;
    double cp = Interpolant (Temp, P, ind);
    return cp;      /* (J/Kg/K) */
}

double CO2MONOLUT_enthalpy(double Temp, double density, double P, double yi[])
{
    int ind = 2;
    double h = Interpolant (Temp, P, ind);
    return h;      /* (J/Kg) */
}

double CO2MONOLUT_entropy(double Temp, double density, double P, double yi[])
{
    int ind = 3;
    double s = Interpolant (Temp, P, ind);
    return s;      /* (J/Kg/K) */
}

double CO2MONOLUT_mw(double yi[])
{
    double MW = 44.01;
    return MW;      /* (Kg/Kmol) */
}

double CO2MONOLUT_speed_of_sound(double Temp, double density, double P,
    double yi[])
{
    int ind = 4;
    double a = Interpolant (Temp, P, ind);
    return a; /* m/s */
}

double CO2MONOLUT_viscosity(double Temp, double density, double P, double yi[])
{
    int ind = 5;
    double mu = Interpolant (Temp, P, ind);
    return mu;      /* (Kg/m/s) */
}

double CO2MONOLUT_thermal_conductivity(double Temp, double density, double P, double
    yi[])
{
    int ind = 6;
    double kt = Interpolant (Temp, P, ind);
    return kt;      /* W/m/K */
}

double CO2MONOLUT_rho_t(double Temp, double density, double P, double yi[])
{
    int ind = 7;
    double drdT = Interpolant (Temp, P, ind);
    return drdT;      /* (Kg/m^3/K) */
}

double CO2MONOLUT_rho_p(double Temp, double density, double P, double yi[])
{
    int ind = 8;
    double drdp = Interpolant (Temp, P, ind);
    return drdp;      /* (Kg/m^3/Pa) */
}

double CO2MONOLUT_enthalpy_t(double Temp, double density, double P, double yi[])
{
    int ind = 9;

```

```

    double dhdT = Interpolant (Temp, P, ind);
    return dhdT; /* J/(Kg.K) */
}

double CO2MONOLUT_enthalpy_p (double Temp, double density, double P, double yi[])
{
    int ind = 10;
    double dhdp = Interpolant(Temp, P, ind);
    return dhdp; /* J/ (Kg.Pascal) */
}

/*****
/* Auxiliary Mixture Functions */
*****/

UDF_EXPORT RealGasFunctionList =
{
    CO2MONOLUT_Setup,          /* initialize */
    CO2MONOLUT_density,        /* density */
    CO2MONOLUT_enthalpy,       /* enthalpy */
    CO2MONOLUT_entropy,        /* entropy */
    CO2MONOLUT_specific_heat,  /* specific_heat */
    CO2MONOLUT_mw,             /* molecular_weight */
    CO2MONOLUT_speed_of_sound, /* speed_of_sound */
    CO2MONOLUT_viscosity,      /* viscosity */
    CO2MONOLUT_thermal_conductivity, /* thermal_conductivity */
    CO2MONOLUT_rho_t,          /* drho/dT |const p */
    CO2MONOLUT_rho_p,          /* drho/dp |const T */
    CO2MONOLUT_enthalpy_t,     /* dh/dT |const p */
    CO2MONOLUT_enthalpy_p      /* dh/dp |const T */
};

/*****

```

```

// User Defined Functions for Liquid Properties , Cp / H of Liquid and saturation /
// spinodal temperatures

// Parameters defined here include the dimension of the LuT for saturation / spinodal
// line , the size of the CP / H LuT
// The size of the liquid LuTs and the number of properties that need to be read from
// those LuTs (density , sound speed , viscosity
// and thermal conductivity

#include "udf.h"
#define DIM_SAT_LUT 300
#define DIM_SPIN_LUT 300
#define NUM_SAT 2
#define DIM_CP_H_LUT 300
#define DISCR_PROPR_T 300
#define DISCR_PROPR_P 300
#define REF_ENTH_CO2 2e5
#define NUMPROP_LIQ 4

static real Tarr_liq [DISCR_PROPR_T];
static real Parr_liq [DISCR_PROPR_P];
static real prop_liq [NUMPROP_LIQ][DISCR_PROPR_T][DISCR_PROPR_P];
static real DT_liq, DP_liq, Pmin_liq, Pmax_liq, Tmin_liq, Tmax_liq;

// Assuming Sat & Spin LuT have the same dimension , storing the relevant properties in
// a matrix

static real Parr_sat [NUM_SAT][DIM_SAT_LUT], Tarr_sat [NUM_SAT][DIM_SAT_LUT];
static real Ptr[NUM_SAT], Pcr[NUM_SAT], Ttr[NUM_SAT], Tcr[NUM_SAT], DP_sat[NUM_SAT];

static real Tarr_cp [DIM_CP_H_LUT];
static real propcp_liq [2][DIM_CP_H_LUT];
static real DT_cp, Tmin_cp, Tmax_cp;

char *namefile[NUMPROP_LIQ], *satname[2], *cpname, *Tname, *Pname;

real Interpolation(real P, real T, int index);

// If variables are to be initialised and accessible by other macros , the 'on demand'
// macro must be used.
// Execute on loading was shown to be invalid and lead to errors – the define property
// macro could not handle the
// data initialisation.

DEFINE_ON_DEMAND(lutload)
{
    int i, j, k;
    FILE *Tpoint, *Ppoint, *Fpoint;

    namefile [0] = ".\\lutspinliq_clip\\dens_l_spinliq_clip";
    namefile [1] = ".\\lutspinliq_clip\\speed_l_spinliq_clip";
    namefile [2] = ".\\lutspinliq_clip\\cond_l_spinliq_clip";
    namefile [3] = ".\\lutspinliq_clip\\visco_l_spinliq_clip";
    Tname = ".\\lutspinliq_clip\\temparr";
    Pname = ".\\lutspinliq_clip\\pressarr";
    satname [0] = "sat_lut";
    satname [1] = "spin_lut";
    cpname = "cp_h_liq_lut";

    // Reading T, P, LuT and assigning constants

    Tpoint = fopen (Tname, "r+");
    Ppoint = fopen (Pname, "r+");

    if (Tpoint == NULL || Ppoint == NULL)
    { Error ("Could not reach T – P liquid arrays!"); }

    for (i=0; i<DISCR_PROPR_T; i++)
    { fscanf (Tpoint, "%lf", &Tarr_liq [i]); }
    fclose (Tpoint);

```

```

for (i=0; i<DISCR_PROPR_P; i++)
{ fscanf (Ppoint, "%lf", &Parr_liq [i]); }
fclose (Ppoint);

Message ("T - P arrays loaded! \n");

DT_liq = Tarr_liq [1] - Tarr_liq [0];
DP_liq = Parr_liq [1] - Parr_liq [0];
Tmin_liq = Tarr_liq [0];
Pmin_liq = Parr_liq [0];
Tmax_liq = Tarr_liq [DISCR_PROPR_T - 1];
Pmax_liq = Parr_liq [DISCR_PROPR_P - 1];

for (k = 0; k < NUMPROP_LIQ; k++)
{
    Fpoint = fopen (namefile [k], "r+");
    if (Fpoint == NULL)
    { Error ("Could not reach %d LUT! \n", k); }

    for (i=0; i<DISCR_PROPR_T; i++)
    {
        for (j=0; j<DISCR_PROPR_P; j++)
        { fscanf (Fpoint, "%lf", &prop_liq [k][j][i]); }
    }
    fclose (Fpoint);
    Message ("LuT %d loaded! \n", k);
}

Message ("Property LuT loading ended! \n");

// SAT_LuT - SPIN_LuT load

for (i = 0; i < NUM_SAT; i++)
{
    Fpoint = fopen (satname [i], "r+");

    if (Fpoint == NULL)
    { Error ("Couldn't find saturation LuT %d!", i); }

    for (j=0; j<DIM_SAT_LUT; j++)
    { fscanf (Fpoint, "%lf", &Tarr_sat [i][j]); }

    for (j=0; j<DIM_SAT_LUT; j++)
    { fscanf (Fpoint, "%lf", &Parr_sat [i][j]); }

    fclose (Fpoint);

    Ptr [i] = Parr_sat [i][0];
    Pcr [i] = Parr_sat [i][DIM_SAT_LUT - 1];
    Ttr [i] = Tarr_sat [i][0];
    Tcr [i] = Tarr_sat [i][DIM_SAT_LUT - 1];
    DP_sat [i] = Parr_sat [i][1] - Parr_sat [i][0];

    Message ("Saturation LUT %d loaded! \n", i);
}

// CP - H LuT load
// Reading T, P, LuT and assigning constants

Fpoint = fopen (cpname, "r+");
if (Fpoint == NULL)
{ Error ("Pointer could not reach CP-H LuT! \n"); }

for (i = 0; i < DIM_CP_H_LUT; i++)
{ fscanf (Fpoint, "%lf", &Tarr_cp [i]); }
Message ("T array for CP LuT loaded! \n");

for (i = 0; i < 2; i++)
{
    for (j = 0; j < DIM_CP_H_LUT; j++)
    { fscanf (Fpoint, "%lf", &propcp_liq [i][j]); }
}

```

```

}
fclose (Fpoint);
Message ("CP - H lut loaded! \n");

DT_cp = Tarr_cp [1] - Tarr_cp [0];
Tmin_cp = Tarr_cp [0];
Tmax_cp = Tarr_cp [DIM_CP_H_LUT - 1];
}

// Interpolation function
real Interpolation(real P, real T, int index)
{
    int indTlow, indThigh, indPlow, indPhigh;

    // Handling out of bounds T, P and effectively clipping them and the relative index

    if (T < Tmin_liq)
    {
        indTlow = 0;
        indThigh = 0; }
    else if (T > Tmax_liq)
    {
        indTlow = DISCR_PROPR_T - 1;
        indThigh = DISCR_PROPR_T - 1;}
    else
    {
        indTlow = floor((T - Tmin_liq) / DT_liq);
        indThigh = ceil((T - Tmin_liq) / DT_liq); }

    if (P < Pmin_liq)
    {
        indPlow = 0;
        indPhigh = 0; }
    else if (P > Pmax_liq)
    {
        indPlow = DISCR_PROPR_P - 1;
        indPhigh = DISCR_PROPR_P - 1; }
    else
    {
        indPlow = floor((P - Pmin_liq) / DP_liq);
        indPhigh = ceil((P - Pmin_liq) / DP_liq); }

    real A_L, B_L, C_L, D_L;
    real Tlow, Thigh, Plow, Phigh;
    real interp_tlow, interp_thigh, Interpolated;

    Tlow = Tarr_liq [indTlow];
    Thigh = Tarr_liq [indThigh];
    Plow = Parr_liq [indPlow];
    Phigh = Parr_liq [indPhigh];

    A_L = prop_liq [index] [indTlow][indPlow];
    B_L = prop_liq [index] [indThigh][indPlow];
    C_L = prop_liq [index] [indTlow][indPhigh];
    D_L = prop_liq [index] [indThigh][indPhigh];

    // Handling case where T, P might exactly be on the array -> linear interpolation or no
    interpolation at all

    if (indTlow == indThigh && indPlow != indPhigh)
    { Interpolated = (Phigh - P) / DP_liq * B_L + (P - Plow) / DP_liq * D_L; }
    else if (indPlow == indPhigh && indTlow != indThigh)
    { Interpolated = (Thigh - T) / DT_liq * A_L + (T - Tlow) / DT_liq * B_L; }
    else if (indPlow == indPhigh && indTlow == indThigh)
    { Interpolated = A_L; }
    else
    {
        // Standard case - bilinear interpolation: first on T then on P
        interp_tlow = (Thigh - T) / DT_liq * A_L + (T - Tlow) / DT_liq * B_L;
        interp_thigh = (Thigh - T) / DT_liq * C_L + (T - Tlow) / DT_liq * D_L;

        Interpolated = (Phigh - P) / DP_liq * interp_tlow + (P - Plow) / DP_liq * interp_thigh;
    }

    return Interpolated;
}

real Interpolation_sat (real press, int i)

```

```

{
  real sattemp;

  if (press < Ptr [i])
  { sattemp = Ttr [i]; }
  else if (press > Pcr [i])
  { sattemp = Tcr [i]; }
  else
  {
    int indPlow = floor ((press - Ptr [i]) / (DP_sat [i]));
    int indPhigh = ceil ((press - Ptr [i]) / (DP_sat [i]));

    if (indPlow == indPhigh)
    { sattemp = Tarr_sat [i][indPlow]; }
    else
    {
      real Plow = Parr_sat [i][indPlow];
      real Phigh = Parr_sat [i][indPhigh];
      real Tlow = Tarr_sat [i][indPlow];
      real Thigh = Tarr_sat [i][indPhigh];

      sattemp = (press - Plow) / DP_sat[i] * Thigh + (Phigh - press) / DP_sat[i] * Tlow;
    }
  }
}
return sattemp;
}

DEFINE_PROPERTY(LIQ_DENSITY, c, t)
{
  int index = 0;
  real Press_read = C_P(c,t);
  real Temp_read = C_T(c,t);

  real prop = Interpolation (Press_read, Temp_read, index);
  return prop;
}

DEFINE_PROPERTY(LIQ_SSPEED, c, t)
{
  int index = 1;
  real Press_read = C_P(c,t);
  real Temp_read = C_T(c,t);

  real prop = Interpolation (Press_read, Temp_read, index);
  return prop;
}

DEFINE_PROPERTY(LIQ_K, c, t)
{
  int index = 2;
  real Press_read = C_P(c,t);
  real Temp_read = C_T(c,t);

  real prop = Interpolation (Press_read, Temp_read, index);
  return prop;
}

DEFINE_PROPERTY(LIQ_VISCO, c, t)
{
  int index = 3;
  real Press_read = C_P(c,t);
  real Temp_read = C_T(c,t);

  real prop = Interpolation (Press_read, Temp_read, index);
  return prop;
}

DEFINE_SPECIFIC_HEAT(LIQ_CP, T, Tref, h, yi)
{
  // Interpolation section: must be carried out twice
  int i, indTlow, indThigh;

```

```

if (T < Tmin_cp)
{
    indTlow = 0;
    indThigh = 0; }
else if (T > Tmax_cp)
{
    indTlow = DIM_CP_H_LUT - 1;
    indThigh = DIM_CP_H_LUT - 1; }
else
{
    indTlow = floor((T - Tmin_cp) / DT_cp);
    indThigh = ceil((T - Tmin_cp) / DT_cp); }

real A, B;
real Tlow, Thigh;
real Interpolated [2];

Tlow = Tarr_cp [indTlow];
Thigh = Tarr_cp [indThigh];

// Interpolation must be carried out twice: first gives CP, second gives absolute
enthalpy
// Calculation of sensible enthalpy will be obtained through subtraction of a constant

for (i = 0; i < 2; i++)
{
    A = propcp_liq [i][indTlow];
    B = propcp_liq [i][indThigh];

    // Handling case where T might exactly be on the array -> linear interpolation or no
interpolation at all
    if (indTlow == indThigh)
    {Interpolated [i] = A;}
    else
    { // Standard: linear interpolation
      Interpolated [i] = (Thigh - T) / DT_cp * A + (T - Tlow) / DT_cp * B; }
    }

    *h = Interpolated [1] - REF_ENTH_CO2;

return Interpolated [0];
}

DEFINE_PROPERTY(SAT_TEMP, c, t)
{
    real press = C_P(c,t);
    int index = 0;
    real sattemp = Interpolation_sat(press, index);
    return sattemp;
}

DEFINE_PROPERTY(SPIN_TEMP, c, t)
{
    real press = C_P(c,t);
    int index = 1;
    real spintemp = Interpolation_sat(press, index);
    return spintemp;
}

```

Bibliography

- [1] Y. Ahn et al. *Review of supercritical CO₂ power cycle technology and current status of research and development*. ID: 312207. 2015. DOI: <https://doi.org/10.1016/j.net.2015.06.009>.
- [2] A. Ameli et al. "Effects of Real Gas Model Accuracy and Operating Conditions on Supercritical CO₂ Compressor Performance and Flow Field". English. In: *Journal of Engineering for Gas Turbines and Power* 140.6 (2018). DOI: 10.1115/1.4038552.
- [3] G. Angelino. "Carbon Dioxide Condensation Cycles For Power Production". In: *Journal of Engineering for Power* 90.3 (1968), p. 287.
- [4] Peder Aursand et al. *The spinodal of single- and multi-component fluids and its role in the development of modern equations of state*. ID: 271415. 2017. DOI: <https://doi.org/10.1016/j.fluid.2016.12.018>.
- [5] L. Azzini, M. Pini, and P. Colonna. *Semi-analytical model for the prediction of the Wilson point for homogeneously condensing steam flows*. ID: 271482. 2018. DOI: <https://doi.org/10.1016/j.ijheatfluidflow.2018.01.004>.
- [6] Nikola D. Baltadjiev. "An Investigation of Real Gas Effects in Supercritical CO₂ Compressors". MA thesis. 2012.
- [7] K. Bier, F. Ehrler, and G. Theis. "Spontaneous Condensation in Stationary Nozzle Flow of Carbon Dioxide in a Wide Range of Density". In: *Adiabatic Waves in Liquid-Vapor Systems*. Ed. by Gerd E. A. Meier and Philip A. Thompson. ID: 10.1007/978-3-642-83587-212. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 129–141. ISBN: 978-3-642-83587-2.
- [8] Christopher E. Brennen. *Fundamentals of Multiphase Flows*. Cambridge University Press, 2008.
- [9] T. Conboy. "An approach to turbomachinery for supercritical brayton space power cycles". English. In: *Nuclear and Emerging Technologies for Space, NETS 2013*. 2013, pp. 124–131. ISBN: 9781627485586.
- [10] Thomas Conboy et al. "Performance Characteristics of an Operating Supercritical CO₂ Brayton Cycle". In: *Journal of Engineering for Gas Turbines and Power* 134.11 (2012). 10.1115/1.4007199, pp. 111703–111703–12.
- [11] F. Crespi et al. *Supercritical carbon dioxide cycles for power generation: A review*. ID: 271429. 2017. DOI: <https://doi.org/10.1016/j.apenergy.2017.02.048>.
- [12] V. Dostal, M. J. Driscoll, and P. Hejzlar. *A Supercritical Carbon Dioxide Cycle for Next Generation Nuclear Reactors*. 2004.
- [13] E. G. Feher. "The Supercritical Thermodynamic Power Cycle". In: *Energy Conversion* 8 (1967), p. 85.
- [14] Takashi Furusawa et al. *Numerical Method for Simulating High Pressure CO₂ Flows With Nonequilibrium Condensation; Turbo Expo: Power for Land, Sea, and Air*. TY: PROC; 10.1115/GT2018-75592. 2018.

- [15] Francesco Giacomelli, Federico Mazzelli, and Adriano Milazzo. *A novel CFD approach for the computation of R744 flashing nozzles in compressible and metastable conditions*. ID: 271090. 2018. DOI: <https://doi.org/10.1016/j.energy.2018.08.050>.
- [16] Ashvin Hosangadi et al. "Numerical Simulations of CO₂ Compressors: Subcritical Inlet Conditions". In: *The 6th International Supercritical CO₂ Power Cycles Symposium*. Pittsburgh, Pennsylvania, 2018.
- [17] IEA. *World Energy Outlook*. 2014.
- [18] ANSYS Inc. *CFX 19.0 Documentation*.
- [19] ANSYS Inc. *Fluent 19.0 Documentation*.
- [20] Byung Ik Lee and Michael G. Kesler. "A generalized thermodynamic correlation based on three-parameter corresponding states". In: *AIChE Journal* 21.3 (1975). doi: 10.1002/aic.690210313; 25, pp. 510–527. DOI: 10.1002/aic.690210313. URL: <https://doi.org/10.1002/aic.690210313>.
- [21] C. Lettieri, D. Yang, and Z. Spakovszky. "An Investigation of Condensation Effects in Supercritical Carbon Dioxide Compressors". In: *Journal of Engineering for Gas Turbines and Power* 137.8 (2015). 10.1115/1.4029577, pp. 082602–082602–8. URL: <http://dx.doi.org/10.1115/1.4029577>.
- [22] C. Lettieri et al. "Characterization of Nonequilibrium Condensation of Supercritical Carbon Dioxide in a de Laval Nozzle". English. In: *Journal of Engineering for Gas Turbines and Power* 140.4 (2018). DOI: 10.1115/1.4038082.
- [23] M. De Lorenzo et al. *Homogeneous two-phase flow models and accurate steam-water table look-up method for fast transient simulations*. ID: 271484. 2017. DOI: <https://doi.org/10.1016/j.ijmultiphaseflow.2017.06.001>.
- [24] James E. McDonald. "Homogeneous Nucleation of Vapor Condensation. I. Thermodynamic Aspects". In: *American Journal of Physics* 30.12 (1962). doi: 10.1119/1.1941841; 21, pp. 870–877. DOI: 10.1119/1.1941841. URL: <https://doi.org/10.1119/1.1941841>.
- [25] F. R. Menter. "Two-equation eddy-viscosity turbulence models for engineering applications". In: *AIAA Journal* 32.8 (1994). doi: 10.2514/3.12149; 06, pp. 1598–1605. DOI: 10.2514/3.12149. URL: <https://doi.org/10.2514/3.12149>.
- [26] Masafumi Nakagawa, Menandro Serrano Berana, and Akinori Kishine. *Supersonic two-phase flow of CO₂ through converging-diverging nozzles for the ejector refrigeration cycle*. ID: 271448. 2009. DOI: <https://doi.org/10.1016/j.ijrefrig.2009.01.015>.
- [27] Carlo Osnaghi. *Teoria delle Turbomacchine*. II Ed. Esculapio, 2013.
- [28] Michal Palacz et al. *HEM and HRM accuracy comparison for the simulation of CO₂ expansion in two-phase ejectors for supermarket refrigeration systems*. ID: 271641. 2017. DOI: <https://doi.org/10.1016/j.applthermaleng.2016.12.122>.
- [29] Derek E. Paxson. "Experimental Characterisation of Condensation Behavior for Metastable Carbon Dioxide". MA thesis. 2016.
- [30] R. Pecnik, E. Rinaldi, and P. Colonna. "Computational fluid dynamics of a radial compressor operating with supercritical CO₂". English. In: *Journal of Engineering for Gas Turbines and Power* 134.12 (2012). Cited By :29. DOI: 10.1115/1.4007196.
- [31] Ding-Yu Peng and Donald B. Robinson. "A New Two-Constant Equation of State". In: *Industrial and Engineering Chemistry Fundamentals* 15.1 (1976). doi: 10.1021/i160057a011, pp. 59–64. DOI: 10.1021/i160057a011. URL: <https://doi.org/10.1021/i160057a011>.

- [32] Roland Span and Wolfgang Wagner. "A New Equation of State for Carbon Dioxide Covering the Fluid Region from the Triple-Point Temperature to 1100 K at Pressures up to 800 MP". In: *Journal of Physical and Chemical Reference Data* 25 (1996).
- [33] National Institute of Standards and Technology (NIST). *REFPROP 10.0 Documentation*. 2018.
- [34] Joerg Starzmann et al. *Modelling and Validation of Wet Steam Flow in a Low Pressure Steam Turbine; Turbo Expo: Power for Land, Sea, and Air*. TY: PROC; 10.1115/GT2011-45672. 2011.
- [35] Steven A. Wright et al. *Operation and Analysis of a Supercritical CO₂ Brayton Cycle*. 2010.