Supercritical thermodynamic property evaluation via adaptive mesh tabulation

by

Sai Praneeth Mupparapu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2019

Abstract

Obtaining accurate computational simulations of fluid flows with a complex thermodynamic behaviour is difficult. This is partly due to the non-linear variation of thermophysical properties in the vicinity of critical point. Supercritical fluid flows are observed in a wide range of applications such as caffeine extraction, supercritical diesel fuel injection, nuclear reactors, and liquid rocket engines. In the near-critical regime, thermodynamic properties are accurately described by a multi-parametric equation of state, but most often using such complex state equations involve expensive computations, thereby consuming a significant portion of the available computational resources. To mitigate this issue, tabulation of state equations stands as an alternative yet they are inefficient due to the strong non-linear property variation in the proximity of critical point. In this thesis a novel tabulation method based on adaptive mesh refinement (AMR) approach is presented, it enabled accurate thermodynamic and physical property evaluation with minimal computational effort. Detailed analyses in the form of error validation, computational cost comparison were performed with reference to the commonly used cubic equations of state. In order to demonstrate the grid scaling effects on total computational cost of a real-fluid CFD simulation, a one dimensional harmonic acoustic wave case was chosen. We show a significant computational cost reduction with the adaptive tabulation approach relative to the cubic state equations (with an underlying iterative root-finding method). It also offered an accurate emulation of the backend equation of state with significantly less computation cost.

The developed adaptive tabular equation of state is also integrated into a Computational Fluid Dynamics (CFD) code which has numerical techniques enabling computation of flows with large density gradients. To validate this CFD solver and to observe accurate thermodynamics effects in gasdynamics simulations, the Sod-shock tube and Shu-Osher shock tube problems were solved computationally for both perfect, real fluid thermodynamics. Also the sod-shock tube problem was simulated with three different thermodynamic initial conditions (supercritical, nearcritical, subcritical regions) the nearcritical case has shown a relatively smaller change in temperature, pressure at the shock contact while having a large change in density compared to two other scenarios. It represented the underlying supercritical thermodynamic behaviour that the density increases drastically with a relatively small change in temperature.

Acknowledgements

I am extremely thankful to Prof. Jean-Pierre Hickey for giving me an exciting opportunity to purse Masters at University of Waterloo. His constant guidance for the last two and half years and support during challenging times is invaluable. His perseverance, work ethic is inspirational and something that I will remember lifetime.

I would also like to thank Prof. Fue-Sang Lien, Prof. Zhongchao Tan for being my readers.

I would be remiss, if I didn't mention Dr. Srihari Kurukuri's family, who had been my constant source of support and delicious homemade food. My life at Waterloo wouldn't have been more memorable without them.

Last but most importantly, I would like to give my greatest appreciation to my parents, friends, cousins for their unconditional support.

"Today is very difficult. Tomorrow is even more difficult. But the day after tomorrow is very beautiful. Most people die tomorrow evening. If you don't work hard, you don't see the sunshine.. *Never give up !* " - Jack Ma.

Dedication

To my beloved parents

అమ్మ , నాన్న

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CFD Computational Fluid Dynamics

RK Redlich-Kwong Equation of State

BWR Benedict-Webb-Rubin Equation of state

mBWR Modified Benedict-Webb-Rubin Equation of state

PR Peng-Robinson Equation of state

SRK Soave-Redlich Kwong Equation of state

GPU Graphical Processing Units

AMR Adaptive Mesh Refinement

HLLC Harten-Lax-van Leer-Contact Riemann solver

PDE Partial Differential Equation

ODE Ordinary Differential Equation

EOS Equation of state

CoolProp An open source thermodynamics and transport property database

NIST National Institute of Standard and Technology

REFPROP NIST Reference Fluid Thermodynamic and Transport Properties Database

WENO Weighted Essentially Non-Oscillatory

Chapter 1

Introduction

1.1 Overview

Matter exists in three different phases: solid, liquid and gas [53]. The boundaries between these phases are best illustrated with the help of a phase diagram, shown in figure 1.1. The equilibrium phase diagram for a single-species allows us to determine the matter's current phase based on, at most, two intensive thermodynamic properties such as: pressure, and temperature [35]. The dividing lines represent the phase boundaries and thereby indicate the physical processes of boiling, melting and sublimation. There are two special points in a phase diagram, namely: triple point and *critical point*. The triple point is a thermodynamic singularity in which all the three phases coexist in equilibrium [66]. Whereas, the critical point is an endpoint in the phase equilibrium curve between liquid and gas [12]. At a temperature and pressure above the critical point, one cannot differentiate the state of matter (at least not using classical metrics); in this state, we are in a supercritical regime [37]. Matter in this supercritical regime has distinct physical properties such as large compressibility and a high diffusivity (similar to gases). This unusual behaviour is exploited in various physical processes such as fluid extraction [48], separation and fuel injection. For example, supercritical diesel injection provides better air-fuel mixing due to the high molecular diffusivity (compared to liquid injection) and enables a better control over the engine operation [5]. Another property of the supercritical regime is low viscosity and non-existent surface tension [37] due to the fact there is no clear delineation between the liquid, and gas states. Several industrial applications such as Organic Rankine Cycle turbines, supercritical CO_2 power systems, transcritical heat exchangers, liquid propellant rocket engines[] are associated with near-critical, transcritical or supercritical flow regimes.

In order to improve these systems efficiencies and assure stable operation, accurate study of the flow physics is essential. Due to the cost and complexities involved in performing experiments with supercritical fluids, researchers and engineers have adopted Computational Fluid Dynamics (Computational Fluid Dynamics (CFD)) simulations as an analysis tool for design in many of these industrial applications.

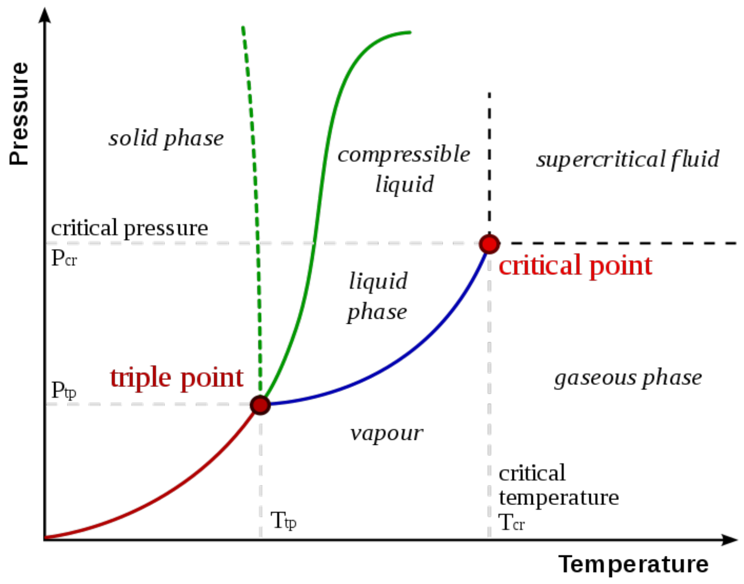


Figure 1.1: Phase diagram: green line, blue line, red line delineate the solid-liquid, liquid-vapour, solid-vapour transition lines, respectively. Source [1]

It is highly desirable to have an accurate, yet robust, CFD simulation for supercritical thermodynamic applications. But CFD simulations involving such pseudo-phase transitions are inherently unstable [44] not only due to the large density gradient near the critical point but also, due to the non-linearity of the thermophysical properties resulting in spurious oscillations[47] in the numerical flux computations. Figure 1.2 shows the sharp change in specific heat C_p at supercritical pressures for oxygen. Another aspect that inhibits the fidelity of such simulations is the accuracy of the equation of state (EOS) that is being used in the CFD solver. The ideal gas law is the most commonly used EOS but the inherent assumptions of this state equation break down in the supercritical thermodynamic regime. In such scenarios, the EOS must account for the intermolecular forces in the gas. For computational efficiency, CFD practitioners usually incorporate a cubic equation of state such as Redlich-Kwong Equation of State (RK) [52], Peng-Robinson PR

[49], Soave-Redlich Kwong SRK [58], all of which are derived from the Van der Waals equation, to estimate the thermodynamic properties of the fluid. But these cubic equations have nearly 10-15 % relative errors [62] as we approach critical point as shown in figure 1.3 when compared to a higher-order state equation such as the modified Benedict Webb Rubin Modified Benedict-Webb-Rubin Equation of state (mBWR) [59] equation of state. The mBWR has 32 parameters, 20 of which describe the saturation curve and its critical point. It matches suitably with experimentally obtained thermodynamic property data and is generally used to interpolate experimental data in thermodynamic databases such as CoolProp, REFPROP.

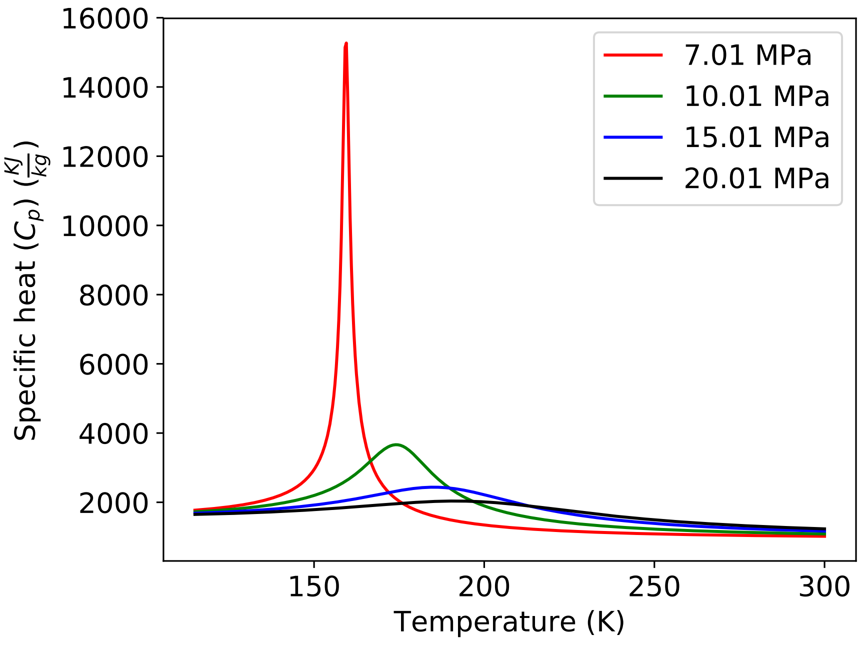
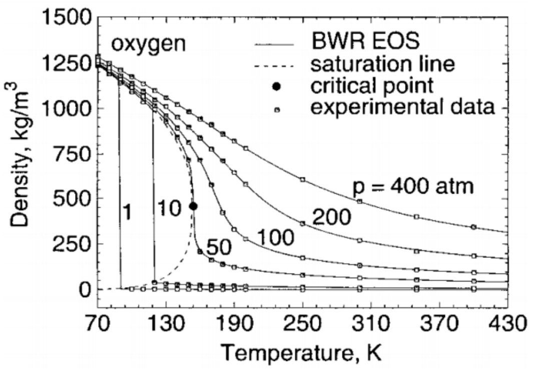
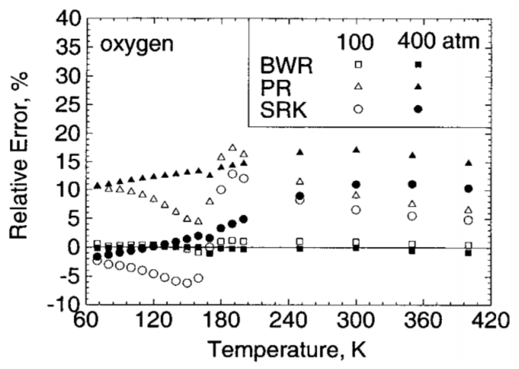


Figure 1.2: Isobaric specific heat (C_p) of oxygen.

In a typical CFD solution process, the compressible Navier-Stokes equations are closed using an equation of state EOS. In other words, given the conserved quantities such as $[\rho, \rho\vec{u}, \rho E]$, the EOS allows us to obtain the primitive variables pressure (P) and temperature (T), the gradients of which are needed to compute the Navier-Stokes equations. The key numerical difficulty in using a highly accurate non-linear EOS lies in computing the primitive thermodynamic variables, as the state equations generally take the form of $P = f(\rho, T)$ and are not a function of $f(\rho, e)$. In the supercritical regime, the internal energy of a fluid depends on two indepedent thermodynamic variables. In order to find approximate thermodynamic solution, iterative numerical methods such as Newton-Raphson solvers are used. One way to mitigate the cost of an iterative root-finding method is to build a pre-tabulated look-up table [50, 73] where the CFD solver looks-up the thermodynamic properties during run-time. However, the complexity lies in building such table in the transcritical critical regime, as we can see from the figure 1.2 the thermodynamic property gradient is highly non-linear near the critical point. Hence, any such look-up table would have to consider this abrupt change in gradient (either Pressure-Temperature or Density - Internal Energy).



(a) Oxygen density computed using Benedict-Webb-Rubin (BWR) Equation of State compared with experimental data



(b) Relative errors of density as determined by three different Equations of state with reference to Experimental data

Figure 1.3: Comparison of various polynomial equations of state with experimental data from Sychev et al. [62]

1.2 Motivation

The scale and complexity of computational fluid dynamics simulations are ever increasing, compared to 800,000 mesh points in 1970s to more recent direct numerical simulation of isotropic turbulence [32] with $4096^3 = 68$ billion grid points there is a huge advancement in terms of hardware and numerical methods that led to accelerated numerical convergence rates and stable computational algorithms [56]. The need for an accurate real fluid thermodynamic property evaluation in transcritical, supercritical CFD simulations had been suggested by several researchers [13, 30, 63, 44, 36]. A strong influence of real gas effects for high pressure supersonic methane jets, hydrogen jets was demonstrated by Hempert et al. [30] and Bonelli et al. [13], as shown in figure 1.4 there was a difference in shockfront width, a very small shockfront was recognizable in the case of real fluid while the absence of real fluid thermodynamics had wider shockfronts. However, the current simulations involving accurate real fluid thermodynamics are very expensive not only due to the number of chemical species involved, but also due to the inherent complexity in terms of their properties evaluation based iterative numerical methods such as computing compressibility factor (Z) using Newton-Raphson solver. Some of the widely used multi-parametric cubic equation of states, such as SRK and PR, were invented for the sole purpose of evaluating the hydrocarbon mixtures at low and high pressures. Even after 40 years of advancements in both hardware and computational methods, relying on the same old method of iterative root-finding to perform high fidelity simulations is a matter of concern. Hence, it is very important to research new approaches/algorithms towards developing computationally efficient yet, accurate method of thermodynamic properties evaluation, which can also be incorporated into the modern CFD codes with minimal effort.

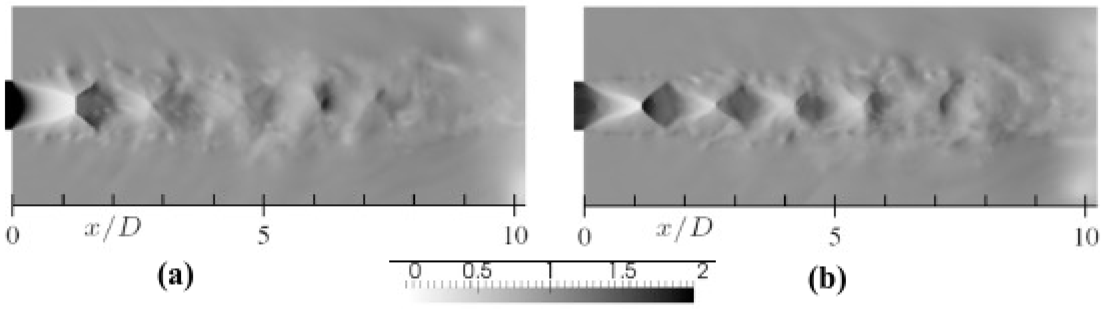


Figure 1.4: Pressure ratio $\frac{P_{in}}{P_{out}}$ for a supersonic jet, LES simulation of high pressure supersonic jet [30] with (a) Ideal gas thermodynamics (b) Real fluid thermodynamics

Thermodynamic and thermophysical property evaluations often consume the majority of computational resources in any CFD simulation involving real fluid thermodynamics. For example in a LES [25] of a supercritical fuel-jet in a cross flow, real fluid thermodynamic properties are considered and evaluated using PR EOS and computations are accelerated with a Graphical Processing Units (GPU). The profiling data is illustrated in the figure 1.5, 1.6. They show the fraction of time spent in various modules, subroutines to the fraction of total computation time. From figure 1.5 we can see that the thermodynamic-transport closures comprise 51.2% of total computation time yet they occupy 24.1 % of total floating operations. Thereby making single floating point operation of a thermodynamics-transport module significantly higher than other routines. Also, if we inspect thermodynamic-transport module more specifically (figure 1.6), the Z-factor sub-routine (compressibility factor calculation) has a big influence on the total floating point operations. Being able to lookup such computationally intensive operations during the run-time would significantly enhance the solver's capabilities.

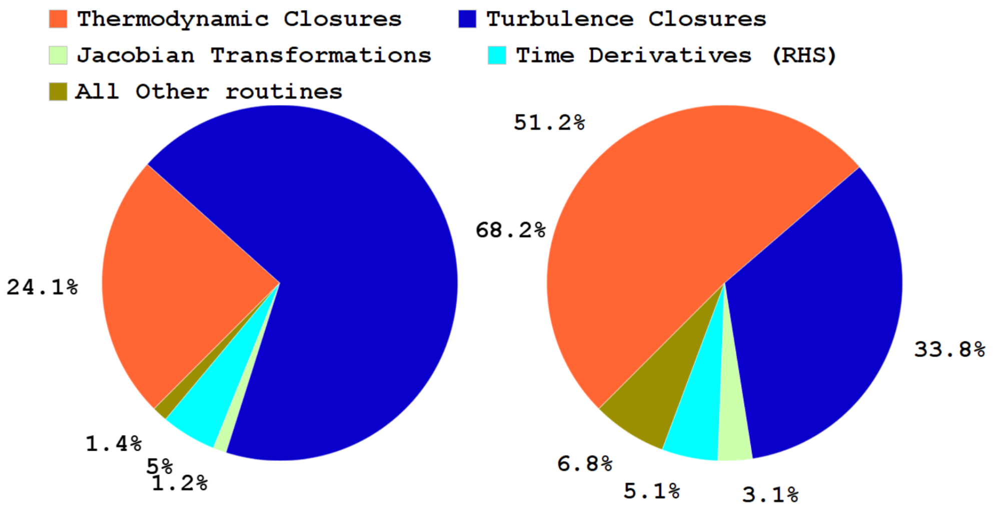


Figure 1.5: (a) Floating point operations as a percent of total flops for all routines (left) (b) Time for computation of different routines/modules as a percent of total computational time (right). Data obtained from an LES simulation of supercritical fuel jet [25].

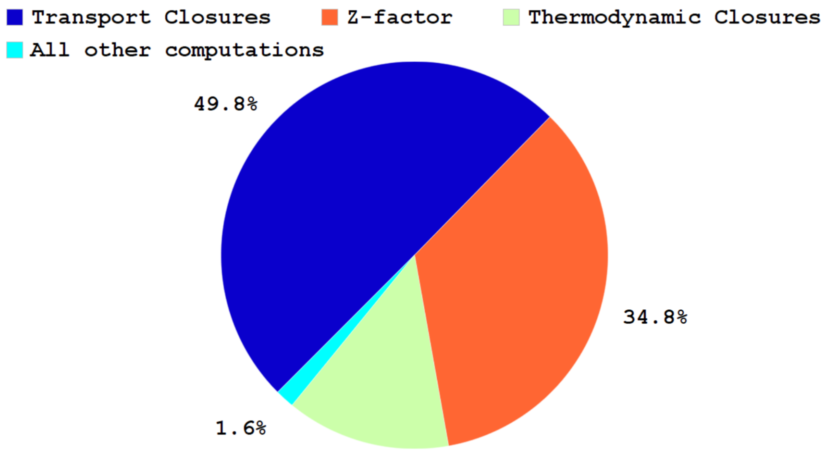


Figure 1.6: Floating point operations for different thermodynamic and transport routines[25].

1.3 Objectives

The objective of this work is to formulate a computationally efficient and accurate tabulated thermodynamic equation of state and thermophysical properties for CFD solvers and investigate its performance with reference to the current existing equation of states such as cubic, multiparametric in terms of speed and accuracy. Further, we implemented the developed EOS into a CFD code to simulate real fluid thermodynamic effects in gas dynamic test cases.

1.3.1 Contributions

The main contributions of this thesis are:

- A tabulated equation of state using block-structured adaptive mesh refinement (AMR) technique. Validation against the conventional methods of property evaluation such as cubic, multi-parametric EOS for speed and accuracy. Demonstration of computational bottlenecks in using a cubic EOS for real-fluid simulations, with the help of a one-dimensional harmonic acoustic wave test case.
- Extended an existing CFD code to enable real-fluid simulations, added features include: tabular EOS integration, adaptation of Harten-Lax-van Leer-Contact Riemann solver (HLLC) to real fluid inviscid flux evaluation, fourth-order accurate Runge-Kutta time advancement scheme with frozen specific heat ratio to prevent spurious oscillations due to large density gradients.
- The significance of using an accurate thermodynamic properties evaluation in CFD simulations is studied with the help of two gasdynamics test cases (Sod, Shu-Osher shock tube). Three different scenarios i.e., Supercritical, nearcritical, subcritical initial conditions were employed to observe non-classical gas dynamic behaviour.

1.4 Outline

This thesis is organized into six chapters, starting with chapter 2 on essential mathematical background with regard to underlying governing equations and thermodynamics. Chapter 3 details the tabulated thermodynamic EOS formulation using block-structured adaptive mesh refinement. Chapter 4 describes the CFD solver that is used to test this tabulated EOS and explains the intricacies of integrating a real fluid thermodynamic EOS into a Riemann solver. Using this combination of tabulated EOS and CFD solver, chapter 5 investigates gas dynamic test cases in three different thermodynamic regimes namely: supercritical, near-critical and subcritical. Conclusions and future work are presented in chapter 6.

Chapter 2 provides a detailed overview of compressible flow governing equations, their properties such as characteristic form, Eigen-values. The concept of equation of state is explained and various types of existing EOS such as cubic, multiparametric, thermodynamic databases are discussed in detail. The calculation of thermodynamic properties such as enthalpy, entropy using departure function is presented.

Chapter 3 presents a detailed framework on tabulated real fluid thermodynamics. Starting with the simplest form of tabulation i.e., homogeneous or logarithmic tabulation, various other forms of tabulation such as Block structured adaptive mesh refinement AMR and Cubic Bezier surface based adaptive mesh refinement AMR. The table generation process in each type of approach is discussed in detail. One of the important aspect in the tabulation of thermodynamics is its storage and retrieval (look-up) process, which will be discussed in this chapter. Also, tabulated EOS thermodynamic consistency evaluation is discussed. Further error estimates, speed-up in computational time when such tabular EOS is implemented in a CFD solver is discussed with the help of an 1D acoustic wave simulation.

Chapter 4 details the numerical schemes, specialties of the CFD code (TwoDThermoCode) that is used for this study, starting with the computational domain discretization of governing equations using finite volume method in an explicit form, we will discuss on obtaining solution to the Riemann problem, refraction & shocks. For this code, second order accurate reconstruction of cell interface states is achieved through piecewise parabolic method. The main objective of this chapter is to discuss the method of integrating real fluid thermodynamics into the HLLC Riemann solver to compute inviscid fluxes. Various numerical methods to overcome spurious oscillations are explained, and a fourth order accurate Runge-Kutta time advancement implementation is described.

Chapter 5 couples both the tabulated thermodynamic equation of state framework and the CFD solver together to study the real gas effects on compressible flow, two test cases are used for this study: the Sod shock tube and the Shu-Osher shock tube. A validation result is obtained by simulating a near critical test case taken from Terashima [64]. Three different thermodynamic initial conditions (subcritical, supercritical and near critical region) are simulated and the real gas effects on shocks/expansion fans are observed.

Chapter 2

Mathematical Background

2.1 Compressible flow governing equations

The governing equations of unsteady, inviscid, compressible flow describing the macroscopic motion of fluid is the following non-linear hyperbolic partial differential equation set, also known as the Euler equations

$$\frac{\partial \rho}{\partial t} + \nabla.(\rho \vec{\mathbf{u}}) = 0 \tag{2.1}$$

$$\frac{\partial \rho \vec{\mathbf{u}}}{\partial t} + \nabla.(\rho \mathbf{u}\mathbf{u}) + \nabla.(\vec{p}) = 0 \tag{2.2}$$

$$\frac{\partial \rho E}{\partial t} + \nabla.(\rho E \vec{\mathbf{u}} + \vec{p}.\vec{\mathbf{u}}) = 0 \tag{2.3}$$

Here, E is the total energy per unit mass which includes the specific internal energy, e, and kinetic energy of the flow:

$$E = e + \frac{1}{2}|\vec{\mathbf{u}}|^2 \tag{2.4}$$

Similarly for enthalpy, we have:

$$H = h + \frac{1}{2}|\vec{\mathbf{u}}|^2 \tag{2.5}$$

When viscous and heat transfer effects are included, the governing equation set takes on some features of a parabolic partial differential equation; formally, this equation set is referred to as the Navier-Stokes equations. In d space dimensions [1], we have up to

[1]In this thesis, we consider $d = 1, 2$

$d+3$ variables for a single species flow. For example, in 1D we have, $[\rho, u, p, E]$ and in 2D we have, $[\rho, u, v, p, E]$ and so on. We have $d+2$ equations in the form of continuity (2.1), conservation of momentum(2.2), conservation of energy(2.3) and an equation relating primitive variables such as temperature (T), pressure (p), density (ρ) to internal energy (e), often referred as the thermal equation of state serves as a constraint and thereby closes the problem.

Assuming a working fluid as an inviscid and non heat conducting, the relation where pressure (p) is a function of specific internal energy (e) and density (ρ), or mathematically

$$p = p(e, \rho) \tag{2.6}$$

We define the thermodynamic derivatives $\gamma(e, \rho), \kappa(e, \kappa)$ in order to simplify the flux vectors.

$$\gamma(e, \rho) \equiv \frac{\partial p(e, \rho)}{\partial \rho e}, \quad \kappa(e, \rho) \equiv \frac{\partial p(e, \rho)}{\partial \rho}, \quad dp(e, \rho) = \left(\frac{\partial p}{\partial \rho}\right)_e + \left(\frac{\partial p}{\partial e}\right)_\rho \frac{\partial e}{\partial \rho} \tag{2.7}$$

2.1.1 Flux formulations in Cartesian form

In two dimensions, the equations 2.1-2.3 can be written in Cartesian coordinates (x, y) as follows,

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} = 0 \tag{2.8}$$

$$\frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2 + p)}{\partial x} + \frac{\partial(\rho u v)}{\partial y} = 0 \tag{2.9}$$

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial \rho u v}{\partial x} + \frac{\partial(\rho v^2 + p)}{\partial y} = 0 \tag{2.10}$$

$$\frac{\partial \rho E}{\partial t} + \frac{\partial u(\rho E + p)}{\partial x} + \frac{\partial v(\rho E + p)}{\partial y} = 0 \tag{2.11}$$

The above conservative system of equations can be written as follows

$$U_t + \left(F^x(U)\right)_x + \left(F^y(U)\right)_y = 0 \tag{2.12}$$

where,

$$U = \underbrace{\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}}_{\text{Conservative Variables}} \quad F^x(U) = \underbrace{\begin{pmatrix} \rho u \\ \rho uu + p \\ \rho vu \\ u\rho E + pu \end{pmatrix}}_{\text{Flux vectors in x-direction}} \quad F^y(U) = \underbrace{\begin{pmatrix} \rho v \\ \rho uv \\ \rho vv + p \\ v\rho E + pv \end{pmatrix}}_{\text{Flux vectors in y-direction}} \tag{2.13}$$

This conservative form of the equations is symbolically expressed in the form [68]

$$U_t + \mathbf{J^x} U_x + \mathbf{J^y} U_y = \mathbf{0} \tag{2.14}$$

where, $\mathbf{J^x}$, $\mathbf{J^y}$ are the Jacobian matrices. Rewriting the flux vectors $F(U)$ in terms of conservative variables such as internal energy and density, with the help of equation of state, thermodynamic derivatives (2.7), in doing so we represent pressure (P) in terms of the internal energy (e), density (ρ), derivatives of pressure with density $\left(\frac{\partial p(e,\rho)}{\partial \rho} \equiv \kappa(e,\rho)\right)$ and internal energy $\left(\frac{\partial p(e,\rho)}{\partial e} \equiv \gamma(e,\rho)\right)$. and thereby, the flux vectors in 2.13 are written as

$$F^x(U) = \begin{pmatrix} \rho u \\ \rho u^2 + p(e,\rho) \\ \rho vu \\ \rho uE + p(e,\rho)u \end{pmatrix} \tag{2.15}$$

$$F^y(U) = \begin{pmatrix} \rho v \\ \rho v^2 + p(e,\rho) \\ \rho uv \\ \rho vE + p(e,\rho)v \end{pmatrix} \tag{2.16}$$

We can obtain a Jacobian matrix by taking the partial derivatives of all individual vectors in a given vector function, here in this case for the hyperbolic system of equations 2.12 with flux vectors $F^x(U), F^y(U)$, where $F = \left(f_1, f_2, f_3 ..., f_n\right)^T, U = \left(u_1, u_2, u_3 ..., u_n\right)^T$ for any orthonormal basis $\omega = (\omega_1, \omega_2, ..., \omega_n)^T \in \mathbb{R}^n, |\omega| = 1$, here $U = [\rho, \rho u, \rho v, \rho E]$, $F = [\rho u, \rho uu + p, \rho vu, \rho uE + pu], [\rho v, \rho uv, \rho vv + p, \rho vE + pv]$, as in equation 2.13, $\omega_1, \omega_2 .. \omega_n$ are unit vectors, normal to the flux vectors $f_1, f_2, .. f_n$ and Jacobian $\mathbf{J}$(F, ω) is given by

$$\mathbf{J}(\mathbf{F}, \omega) = \sum_{\mathbf{j=1}}^{\mathbf{n}} \omega_{\mathbf{j}} \mathbf{J_j}(\mathbf{F}) \tag{2.17}$$

and the matrix **J** has m real eigenvalues $(\lambda_1, \lambda_2, .., \lambda_m)$ and m linearly independent eigenvectors $(\vec{r_1}, \vec{r_2}, .., \vec{r_m})$

$$\mathbf{J}(F,\omega) = \mathbf{J^x}\omega_1 + \mathbf{J^y}\omega_2 \equiv \frac{\partial F}{\partial U} = \begin{pmatrix} \partial f_1/\partial u_1 & \partial f_1/\partial u_2 & \dots & \partial f_1/\partial u_n \\ \partial f_2/\partial u_1 & \partial f_2/\partial u_2 & \dots & \partial f_2/\partial u_n \\ \vdots & \vdots & \ddots & \vdots \\ \partial f_n/\partial u_1 & \partial f_n/\partial u_2 & \dots & \partial f_n/\partial u_n \end{pmatrix} \tag{2.18}$$

$$\mathbf{J^x} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \gamma\frac{u^2+v^2}{2} + \kappa - u^2 & (2-\gamma)u & -\gamma v & \gamma \\ -uv & v & u & 0 \\ u\left(\gamma\frac{u^2+v^2}{2} - H + \kappa\right) & H - \gamma u^2 & -\gamma uv & (1+\gamma)u \end{pmatrix} \tag{2.19}$$

$$\mathbf{J^y} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ \gamma\frac{u^2+v^2}{2} + \kappa - v^2 & -\gamma u & (2-\gamma)v & \gamma \\ v\left(\gamma\frac{u^2+v^2}{2} - H + \kappa\right) & -\gamma uv & H - \gamma u^2 & (1+\gamma)v \end{pmatrix} \tag{2.20}$$

2.1.2 Characteristics and Eigenvalues

The characteristics in the context of a partial differential equation are a set of curves along which the Partial Differential Equation (PDE) becomes an ordinary differential equation Ordinary Differential Equation (ODE) [68]. The Jacobian matrix (**J**) has 4 eigenvalues and they can be found by solving $|\mathbf{J} - \lambda\mathbf{I}| = 0$, where $|...|$ represents the determinant and λ are the eigenvalues of **J**.

$$\lambda^1 = \Gamma.\omega - c; \quad \lambda^2 = \Gamma.\omega; \quad \lambda^3 = \Gamma.\omega; \quad \lambda^4 = \Gamma.\omega + c; \quad c = \sqrt{\frac{\partial p}{\partial \rho} + \frac{\partial p}{\partial e}\frac{p}{\rho^2}} \tag{2.21}$$

where, $\Gamma = (u, v)^T$, $\omega^{\perp} = (-\omega_2, \omega_1)^T$, ω_1, ω_2 are unit vectors, normal to the flux vectors f_1, f_2, c is the speed of sound and the eigenvectors are,

$$r^1 = \begin{pmatrix} 1 \\ u - c\omega_1 \\ v - c\omega_2 \\ H - \Gamma.\omega c \end{pmatrix} \; r^2 = \begin{pmatrix} 1 \\ u \\ v \\ H - \frac{c^2}{\gamma} \end{pmatrix} \; r^3 = \begin{pmatrix} 0 \\ -\omega_2 \\ -\omega_1 \\ \Gamma.\omega^{\perp} \end{pmatrix} \; r^4 = \begin{pmatrix} 1 \\ u + c\omega_1 \\ v + c\omega_2 \\ H + \Gamma.\omega c \end{pmatrix} \tag{2.22}$$

In hyperbolic systems of partial differential equations, there exists one wave for each eigenvalue [68], here in this 2D case there will be a total of 4 waves associated, in which the acoustic waves resulting from the eigenvalues λ^1, λ^4 are non-linear in nature (meaning, travelling with sound speed ($\pm c$) relative to the flow). If the compressible flow is supersonic, we can observe such non-linear acoustic waves in the form of shocks & expansion fans. Whereas the other two waves are a contact discontinuity across which the pressure (p) and velocity (u) are constant.

2.2 Equation of state (Pressure-Explicit)

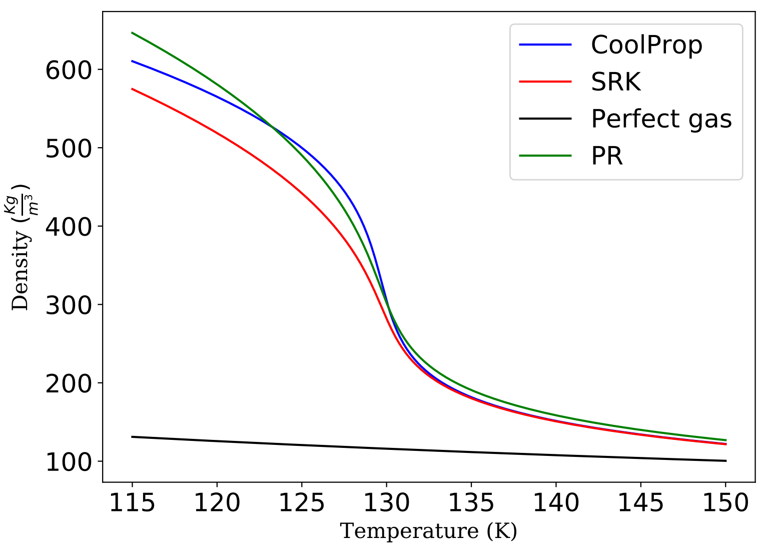
An equation of state is a mathematical expression that relates various thermodynamic state variables such as pressure, temperature, and volume/density. Some of the earliest approaches to establish such equation of states were Boyle's Law, Ideal gas Law, Gay-Lussac's Law, and Dalton's law of partial pressure. Van der Waals equation of state was the first state equation to account for the intermolecular forces present in a non-ideal gas. Some of the more modern state equations are multi-parametric written in pressure explicit form. Here we present some of the more common state equations.

2.2.1 Cubic Equation of state

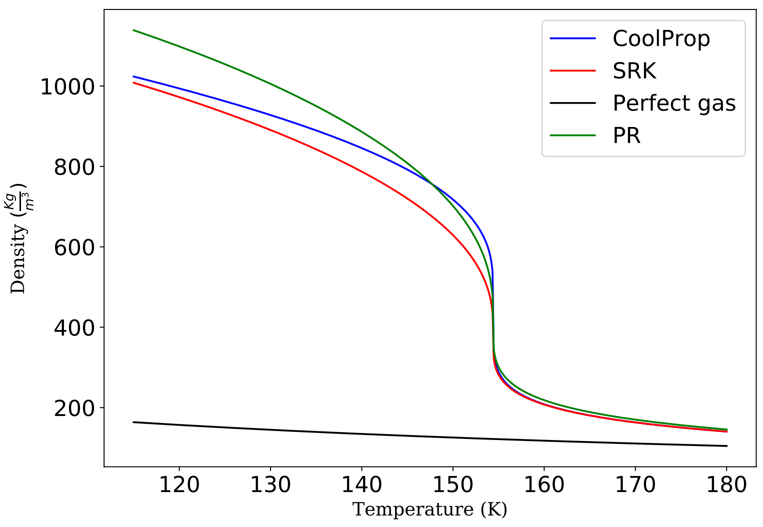
In an attempt towards accurate thermodynamic property evaluation over a wide range of temperatures and pressures, the first step was taken by Van der Waals [71] who proposed a cubic equation of state that can be applicable to both gaseous and liquid phases of a fluid. So far there are several hundred equations of states [24] that are built upon the Van der Waals EOS in order to extend the range of evaluation of real fluids.

Among those equations, the first major contributions is from Soave [58] in 1972 and then the Peng & Robinson[49] equation in 1976. Both were developed to accurately compute the thermodynamic properties for the oil and gas industries. To this day, they are the most widely used equation of states in the computational fluid dynamics software, this is largely due to their relative simplicity in terms of number of parameters involved and ability to quantify the underlying uncertainty when working with both pure components and mixtures, and also their ability to compute consistent mixture rules. Both the SRK and PR polynomial expressions have a similar form as in equation 2.23 and constants are tabulated in table 2.1

$$P = \frac{RT}{V-b} - \frac{a\alpha}{(V+\sigma b)(V+\epsilon b)} \tag{2.23}$$



(a) Density - Temperature plot of Nitrogen at 4 MPa



(b) Density - Temperature plot of Oxygen at 5 MPa

Figure 2.1: A density($\frac{kg}{m^3}$) vs Temperature (K) plot for nitrogen, oxygen where the density is evaluated with different equation of states namely, CoolProp ● , Soave Redlich-Kwong SRK, Ideal gas law, Peng-Robinson Equation of State PR

where, V is the specific molar volume, α, ϵ, are the constant parameters which differ with each equation of state, and the terms a, b are detailed in table 2.1. Even though we can compute pressure using 2.23 for a given temperature (T), density (ρ), to obtain density or temperature given the other two variables, one has to go through an iterative process of root-finding which is computationally inefficient, also the uncertainties associated with cubic equation of states are high [24] in comparison with experimental data or multiparametric equation of states. In terms of capabilities, when compared to SRK, RK equations of state, PR has better accuracy in calculating vapour liquid equilibrium [49].

Parameter	Redlich and Kwong	Soave-Redlich -Kwong	Peng-Robinson
σ	1	1	$1+\sqrt{2}$
ϵ	0	0	$1-\sqrt{2}$
a	$0.42747\ \frac{R^2{T_c}^2}{p_c}$	$0.42747\ \frac{R^2{T_c}^2}{p_c}$	$0.45724\ \frac{R^2{T_c}^2}{p_c}$
b	$0.8664\ \frac{RT_c}{p_c}$	$0.8664\ \frac{RT_c}{p_c}$	$0.07780\frac{RT_c}{p_c}$
k	$0.48508+$ 1.55171ω $-0.15613\omega^2$	$0.48508+$ 1.55171ω $-0.15613\omega^2$	0.37464 $+1.54226\omega$ $-0.26992\omega^2$
α	$\frac{1}{{T_c}^{0.5}}$	$\left[1+k\left(1-\sqrt{\frac{T}{T_c}}\right)\right]^2$	

Table 2.1: Various parameters of cubic equations of state [24], here ω is accentric factor

2.2.2 Multi-parametric Equation of state

Evaluating thermodynamic properties with very low uncertainties is crucial for many industrial and scientific applications. In this section, we discuss a few such equation of states that are derived based on the fundamental relations of thermodynamics such as, internal energy as a function of entropy and volume, Gibbs energy (or) Helmholtz energy as a function of temperature and pressure, enthalpy as a function of entropy and pressure.

Among the multi-parametric high accuracy models, the Benedict-Webb-Rubin equation of state [11] or Benedict-Webb-Rubin Equation of state (BWR) is the most common form that is still being used in a number of industrial and scientific applications, thermodynamics softwares. It was developed in 1940 by Benedict *et al.* to extend the Beattle-Bridgeman equation of state into high density regimes across various fluids. BWR is given as

$$p = \rho RT + \Big(B_0RT - A_0 - \frac{C_0}{T^2}\Big)\rho^2 + (bRT - a)\rho^3 + a\alpha\rho^6 + \Big(\frac{c\rho^3}{T^2}\Big)(1 + \gamma\rho^2)e^{-\gamma\rho^2} \quad (2.24)$$

The empirical constants $A_0, B_0, C_0, a, b, c, \alpha, \gamma$ are determined based on experimental data. This simple exponential form of representation evaluated properties in gas phase and low to medium density supercritical phases but when used for energy evaluation in the high density liquid phase and supercritical phases there was a $\pm 10\%$ relative error [24]. There were several attempts [10, 60] to improve and extend BWR equation of state's capabilities into wider thermodynamic ranges across various species. Its present day version, the mBWR (modified Benedit-Webb-Rubin) comes from Jacobsen-Stewart equation of state [33], which is an advanced version of its predecessor by incorporating 32 empirical parameters making the property calculations precise and accurate.

2.2.3 Thermodynamic property evaluation using Pressure-Explicit Equation of State

To calculate real fluid extensive properties such as enthalpy, entropy, internal energy, we need departure functions. A departure function for a given thermodynamic property is defined as the difference between the value computed for an ideal gas and its real value (computed in its existing form) [24]. Hence, the properties like enthalpy, entropy, internal energy are evaluated relative to a base state *i.e.*, as an amount of change between two thermodynamic state spaces [24]. For example in the case of enthalpy, the change between two thermodynamic states (states 0 and n), we first compute the departure function between ρ_1 and near zero density at $T = T_0$, then it is added to the ideal gas enthalpy change due to the temperature change form T_0 to T_n, then subtract the departure function value between ρ_n and near zero density. The enthalpy of any state is calculated as,

$$\begin{aligned} H(T,\rho_n) = H^0(T^0) + \int_0^{\rho_n}\left[\frac{R}{\rho_n} - \Big(\frac{1}{\rho_n^2}\Big)\Big(\frac{\partial p}{\partial T}\Big)_{\rho_n}\right]_T d\rho_n + \int_0^{\rho_n}\left[\Big(\frac{p}{\rho_n^2}\Big) - \Big(\frac{RT}{\rho_n}\Big)\right]_T d\rho_n \\ + \int_{T^0}^{T}\left[\frac{C_p^{pg}(T)}{T}\right]dT - Rln(RT\rho_n) \end{aligned} \quad (2.25)$$

The entropy, S, of any thermodynamic state is calculated using,

$$S(T,\rho_n) = S^0(T^0) + \int_{T^0}^{T} \left[\frac{C_p^{pg}(T)}{T}\right] dT - Rln(RT\rho_n) + \int_0^{\rho_n} \left[\frac{R}{\rho_n} - \left(\frac{1}{\rho_n^2}\right)\left(\frac{\partial p}{\partial T}\right)_{\rho_n}\right]_T d\rho \tag{2.26}$$

The speed of sound for any given fluid can be calculated as

$$c^2 = \left(\frac{\partial p}{\partial \rho}\right)_{s,X_i} = \frac{\gamma}{\rho \kappa_T} \tag{2.27}$$

Here the specific heat capacity ratio (γ) is equal to $\frac{c_p}{c_v}$ and the isothermal compressibility (κ) is estimated using the volume gradient with respect to pressure as, (ν is the volume)

$$\kappa_T = -\frac{1}{\nu}\left(\frac{\partial v}{\partial p}\right)_{T,X_i} \tag{2.28}$$

The heat capacity at constant volume $C_{V,m}$ is given by,

$$C_{V,m}(T,\rho_n) = \left[C_{p,m}^{pg}(T) - R\right] - \int_0^{\rho_n} \left[\frac{T}{{\rho_n}^2}\frac{\partial^2 p}{\partial T^2}\right] d\rho_n \tag{2.29}$$

The heat capacity at constant pressure $C_{p,m}$ calculated as,

$$C_{p,m}(T,\rho_n) = C_{V,m}(T,\rho_n) + \left[\frac{T}{{\rho_n}^2}\left(\frac{\partial p}{\partial T}\right)^2_{\rho_n}\left(\frac{\partial \rho_n}{\partial p}\right)_T\right] \tag{2.30}$$

2.2.4 Thermodynamics, transport properties database

The standard way of evaluating thermodynamic properties, physical properties (exceptions are surface tension, viscosity, thermal conductivity) to the highest possible accuracy is through an equation of state that is developed using Helmholtz energy formulations. It would be practically inconvenient to implement/use such formulations in a CFD software.

So, it is customary to use a database or wrapper module that evaluates the thermodynamic properties for a given physical state and composition. The most commonly used libraries are REFPROP[41], it is based on modified BWR EOS developed by NIST and its looka-like but open source version is CoolProp, developed by Bell et. al[9] both libraries offer thermodynamic, transport properties for pure, pseudo-pure phase mixtures. They evaluate properties based on highly accurate Helmholtz energy formulation, and most commonly viscosity, thermal conductivity are evaluated using Chung's method [16, 17]. With the help of wrappers in various interpretable languages such as python, Matlab CFD developers can import and implement into the solvers.

Chapter 3

Tabulated thermodynamics for Computational fluid Dynamics

3.1 Tabulated Thermodynamics

State equation tabulation involves storing the thermodynamic and transport properties of the species in a library (e.g. binary file, .xml file, .json file), by doing so we move the evaluation of the EOS to a preprocessing step. Thus decoupling the EOS computational complexity from runtime performance of the solver. This approach allows for the use of accurate high-quality multiparametric backend EOS such as BWR without major runtime penalty. Tabulated approaches also provide an effective control on the thermodynamic error of the state equation. For single species fluid, two independent thermodynamic variables are needed to fully define the local thermodynamic state in a single phase. Classically, the choice of the thermodynamic tuple has been pressure and temperature given the convenience of computing the isothermal and isobaric conditions. For CFD simulations, this approach is sub-optimal since the known transported quantities (in a conservative form) are ρ and e; thus making these the ideal basis for any thermodynamic table. Once the corresponding position in the table is found, all thermodynamic and transport properties may be interpolated or computed. Here, we consider three different tabulation approaches:

- Homogeneous of logarithmic tabulation
- Block structured Adaptive Mesh Refinement (AMR)
- Adaptive Mesh Refinement with Bezier patch fitting

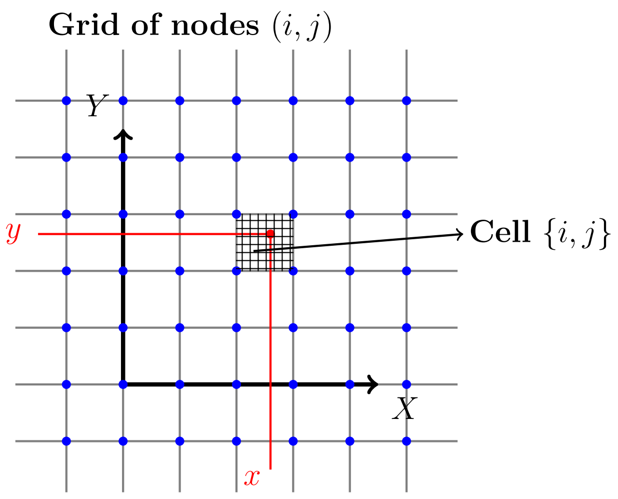


Figure 3.1: A typical cell in a 2D lookup table, the red dot indicating the unknown point

3.1.1 Previous works

The quest for calculating thermodynamic properties using interpolation approaches started in 1977, Theodore E. Fessler[22] worked on a "double-lagrangian" interpolation method that evaluates thermodynamic properties at intermediary points (x_i, y_i) to the tabulated nodes a_1, a_2, a_3, with the help of approximated functions $(F(a_1), F(a_2), F(a_3))$. This tabulation method has been explained and implemented in section 3.2.

Kunick [39] developed a "spline-based table look-up method (SBTL)" and also successfully tested it in a CFD solver with water and steam as working fluids, the SBTL method combines global bi-quadratic spline interpolation, a combination of linear transformation functions. In a project of International Association for the Properties of Water and Steam (IAPWS) the SBTL method has been applied to the industrial formulation of IAPWS-97. SBTL property functions for water and steam have been generated for the independent variables specific volume (ν) and internal energy (e).

However there are some shortcomings in this approach, the nodes are often clustered to consider the nonlinear behaviour of the fluid property function, which leads to computationally intensive cell search algorithms and also most frequently applied property functions are often calculated from inverse functions, rather than from an explicit forward function. [39]

As the requirement of an adaptive Cartesian mesh is obvious in the transcritical regions of the look-up table, many researchers started to develop such irregular grid spacing meshes. Even though there were many readily available adaptive meshing procedures, they are primarily used in the field of numerical analysis, computational mathematics, the first approach to apply an adaptive Cartesian mesh for complex fluid computations is done by Xia et al.[73]. The thermodynamic properties and their pertinent derivatives are evaluated by means of an adaptive Cartesian mesh in the thermodynamic plane that provides user-specified accuracy over any selected domain. After comparing with the REFPROP database, the properties are not only accurate but also their reconstructions are approximately two orders of magnitude faster than property evaluations from the original database(REFPROP).

Carpenter[15] worked on an adaptive tabulation scheme for multi-phase equations of state where in the computational efficiency is provided through the use of a quad-tree representation. Using both rectangular and triangular interpolation regions resulting in a accurate description of phase boundaries.

Zhiqi et. al[42] constructed an adaptive tabular thermodynamic representation with triangular elements, which is an efficient method to reconstruct the phase boundaries, thermodynamic properties of real fluids. It can provide user-defined accuracy over a selected thermodynamic plane with fewer nodes than a simple homogeneous tabulation. A generic function approximation algorithm for real valued functions in two independent variables is developed by Luke [19] where in a method for evaluating the approximate surface is generated using a piecewise C^1 - continuous cubic Bezier surface. The significance of this approach is observed in terms of the number of patches required to fit the supercritical region for a given accuracy of 0.1% (in reference to the REFPROP database) which is 1,968 whereas the isotropic tabular formulation of Xia et al[73] required 225,121 patches to fit. Though, Xia et al[73] reported similar speedup of performance improvements over REFPROP, the an-isotropic refinement used by Luke et al. [19]'s tabular fitting technique improved the efficiency of tabulation in terms of storage.

3.2 Homogeneous or Logarithmic tabulation

Homogeneous tabulation is the simplest form of storing data in a multidimentional array covering the entire thermodynamic domain with an equal spacing or alternatively under a known mapping function like a logarithmic distribution, so that the indexes map to the structured data points as shown in figure 3.1. The advantage of using such an approach is that the lookup time complexity is minimal as it is a matter of calculating the normalized x, y values and mapping them to the index for interpolation. But, this results in very large table sizes as the grid size is not adapted to the magnitude of the local error.

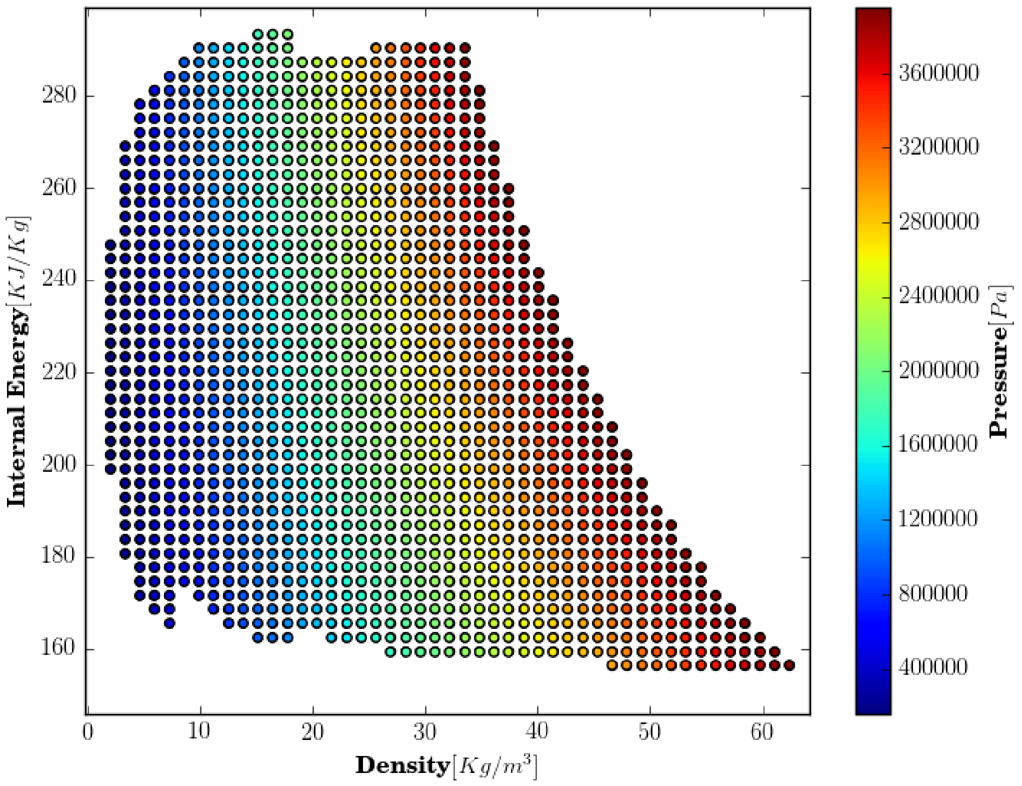


Figure 3.2: Uniform spaced EOS table for oxygen in internal energy (E), density (ρ) space.

The minimal accuracy is in the region in which the function varies most rapidly so that slowly varying regions must be over-refined to accommodate high-gradient regions. Tabulating in density/energy space, as shown in figure 3.2, allows for a simple lookup from the transport equations but simultaneously the corner regions of the table contain pressure and temperature values that are far outside the region of interest for most CFD problems. Therefore, typical boundaries of isobaric conditions of the simulation are mapped to the energy/density space.

3.3 Block structured Adaptive Mesh refinement (AMR)

In order to achieve an uniform accuracy of tabulated thermodynamic properties (i.e., when the error computed with reference to a NIST database), we have to balance or adjust the grid spacing on the basis of local property gradient throughout the thermodynamic domain space. Such adaptable grid spacing or "mesh refining" methods fall under the category of unstructured grids. Block structured adaptive mesh refinement is one such method, where the grid spacing/cell size is based on a hierarchical level of resolution that is defined for a particular block (a structured group of cells) within which the cell lies.

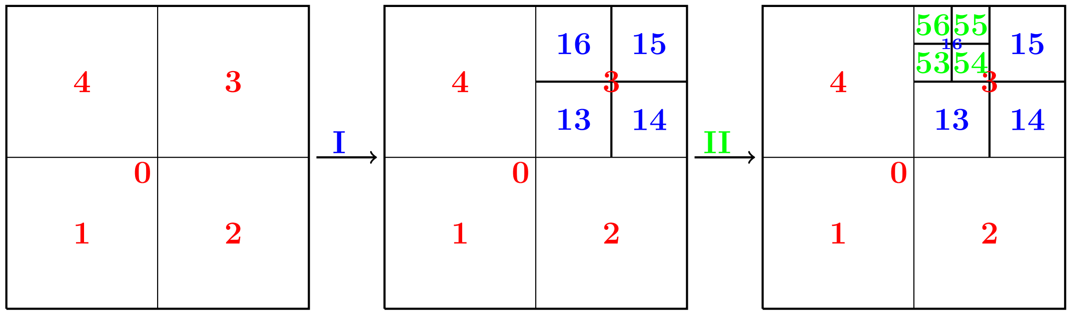


Figure 3.3: Block structured Adaptive Mesh refinement

The criteria on which a given block gets a new hierarchical level of resolution primarily depends on the accuracy with which we are able to compute the thermodynamic parameters. Blocks associated with a non-linear change in property values such as near-critical region of a phase diagram will have a finer grid spacing than the regions where the property gradient is much smoother or linear.

3.3.1 AMR Table Generation

The adaptive mesh refinement tabulation is based on Cartesian adaptive mesh structure. A rectangular x-y region defined by the table limits and these x, y variables can be Pressure, Temperature, density or internal energy. During initalization, the unknown thermodynamic properties are evaluated at each of the rectangle corners using NIST database and an approximate reconstruction method (interpolation) is used to evaluate these properties at pre-selected equidistant points. For simplicity, we chose bivariate interpolation to evaluate the properties at those selected 25 equidistant points for each cell created due to subdivision. We start by transforming the rectilinear domain of interest (i.e., χ) into a square coordinate system (κ) and the condition for subdivision is defined based on relative error(Φ) computed with reference to NIST database as,

$$\Phi = \frac{\chi_{Interpolated} - \chi_{NIST}}{\chi_{NIST}} \tag{3.1}$$

Algorithm 1: Block Structured Adaptive refinement based Tabular EOS Generation

```
$\epsilon \leftarrow$ User defined Error tolerance;
$\eta \leftarrow$ Table maximum resolution limit;
$[(min_1, max_1), (min_2, max_2)] \leftarrow$ State space($\chi$) range (P-T) or ($\rho$-e);
$f : \chi \rightarrow \kappa$ Projective transform($f$) of state space($\chi$) into square domain($\kappa$);
Evaluate thermodynamic properties using NIST, $\forall v \in [(min_1, max_1), (min_2, max_2)]$;
Initialize the Quadtree Root node with transformed state space;
Function Subdivide(Rootnode):
    Add new child nodes with properties evaluated, to the list;
    for child in childnodes : do
        for point ← 1 to 25 points do
            Interpolate thermodynamic properties using 4 neighbour nodes;
            Relative Error $\Phi \leftarrow$ Computed using 3.1 ;
            Check for thermodynamic consistency using 3.4;
            Append error to a list;
        end for
        if $max(Errors(\Phi)) \geq \epsilon$ & $Quadtree.depth < \eta$ then
            Subdivide(child);
        else
    end for

End Subdivide;
Prune(Rootnode);
Traverse(Rootnode);
Save Rootnode and its leaves in a list;
return Table;
```

The reconstructed (interpolated) properties are validated with the reference data obtained from NIST database or any backend EOS to quantify the interpolation error. If the error is more than the user-specified tolerance, the domain is further divided into four equal regions (squares), again they will be validated by comparing with the exact NIST database values. With such repetitive subdivisions, a majority of flatter(linear) regions of the mathematical expression will satisfy the desired accuracy within few subdivisions whereas the areas of sharper change such as near-critical region will be subdivided further. Storing this data in a dynamically spaced quadtree format can able quick lookup that can be comparable to the speed of equally of spaced tables.

3.3.2 Lookup Approach

The usual query/search in quadtrees follows either top-down or bottom-down approach like starting from the root node and isolate the child within which the point belongs to or vice versa. This has a time complexity of O(Log n) where 'n' is the depth of tree. As an alternative approach, we employ a fast lookup algorithm that requires additional memory to store the indices in tree data-structure, and the efficiency doesn't depend much on the depth of the tree. It is constructed by the following steps.

A uniform index array based on quadtree subdivision covering the entire computational domain is generated as per the pre-defined maximum refinement level. For example, at a given maximum refinement level of 3, the uniform index table is shown in the figure 3.5. If the maximum refinement level is n+1 (the refinement, level of the single rectangular cell that cover the entire thermodynamic region of interest is 0) then there will be 4^n number of individual patches (in this case rectangular cells), the integer number stored in the cell indicates the index of that patch.

```python
uni_old = np.zeros((1,1))
for n in range(QuadTree.maxdepth):
    uni_new = np.zeros((2**(n+1), 2**(n+1)))
    for i in range(2**n):
        for j in range(2**n):
            uni_new[2*i][2*j] = int(uni_old[i][j]*4 + 1)
            uni_new[(2*i)+1][2*j] = int(uni_old[i][j]*4 + 2)
            uni_new[(2*i)+1][(2*j)+1] =int(uni_old[i][j]*4+3)
            uni_new[2*i][(2*j)+1] = int(uni_old[i][j]*4 + 4)
    uni_old = uni_new
```

Listing 3.1: Generating an uniform array index

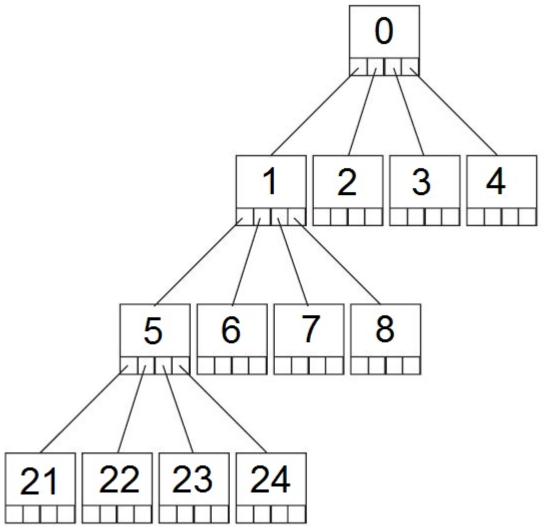


Figure 3.4: Sample quadtree subdivision

84	83	80	79	68	67	64	63
81	82	77	78	65	66	61	62
72	71	76	75	56	55	60	59
69	70	73	74	53	54	57	58
36	35	32	31	52	51	48	47
33	34	29	30	49	50	45	46
24	23	28	27	40	39	44	43
21	22	25	26	37	38	41	42

Figure 3.5: Uniform index for a maximum refinement level of 3

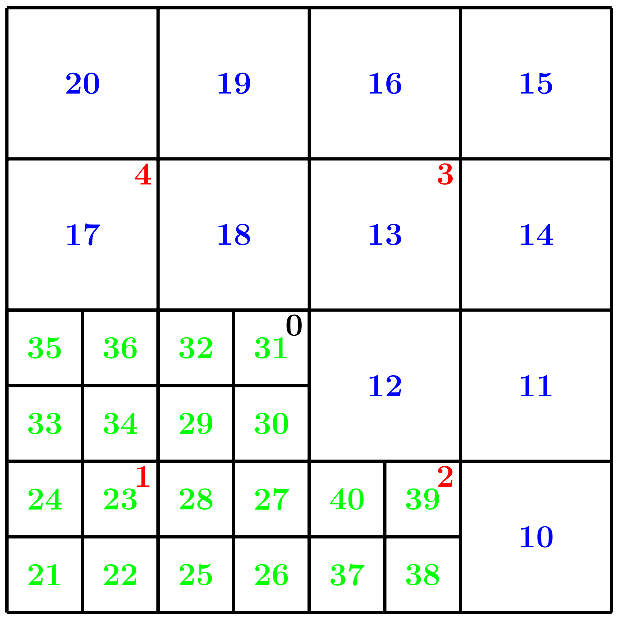


Figure 3.6: Overlapping index for uniform index at refinement levels $n = 0, 1, 2, 3$

An overlapping mapped index array is also generated which contains the child node index in the place of all further sub divisions pertaining to that child if there was a refinement as illustrated in figure 5. The cell within which a point (T_x, P_x) is located can be determined at an algorithmic complexity $O(1)$, directly without searching the tree (which is of algorithmic complexity $O(n)$ where n being the refinement level), by using the mapping index array. Furthermore, the storage required for the uniform index table is small for this generation computers, for example, if the maximum refinement level of the adaptive table is 10, and an integer requires 4 B of memory, the uniform index table requires $(2^9 x 2^9 x4)/(1024^2)MB = 1MB$.

Algorithm 2: Block Structured Adaptive refinement based Tabular EOS Lookup

```
Input: (Pressure(P), Temperature(T)) or (Density($\rho$), Internal-Energy)
Output: List of thermodynamic properties
$\partial\chi_1, \partial\chi_2 \leftarrow$ Calculate the grid resolution based on maximum depth of tree;
$f : \chi \rightarrow \kappa$ Projective transform($f$) of state space($\chi$) into square domain($\kappa$);
Compute the x-index, y-index based on the ratio of x-value, y-value to grid spacing;
$idx \leftarrow floor\left(\frac{\chi_1}{\partial\chi_1}\right)$;
$idy \leftarrow floor\left(\frac{\chi_2}{\partial\chi_2}\right)$;
$v \leftarrow$ Uniform Index[$idy$][$idx$]                    ▷ O(1);
$u \leftarrow$ Overlapping Index[v]                   ▷ O(1);
$w \leftarrow$ np.where(Quadtree list == u)                ▷ O(1);
$Data \leftarrow Quadtreedata[w]$                   ▷ O(1);
$P_n \leftarrow$ Data[0]              ▷ The square nodes in which the point lies;
$D_n \leftarrow$ Data[1:]      ▷ Access thermodynamic property data stored in those nodes;
$P \leftarrow$ Interpolation($P_n, D_n$);
return $P$;
```

3.3.3 AMR Storage

Once we subdivide the tree based on the underlying non-linear equation of state as described in algorithm 1, we traverse the tree top down and update the leaves (nodes) of the quadtree. Then for each element in leaves (Quadtree.nodes) we append the node's stored properties into a list file("Quadtree-leaves"), also as the data is unstructured we keep track of their indices into another file ("Quadtree index"). A uniform index array is constructed as described in 3.3.2 based on the table's observed maximum tree depth and written into a file ("Uniform Index"). The final task is to construct an overlapping index with the help of previously constructed quadtree index file and the uniform index file. The overlapping index superposes the uniform index in such a way that for any given unknown coordinate the overlapping index overrides its uniform index except, in case of child nodes where they both are equal. Finally, the projective transform module is also stored so that the same mapping will be used during the lookup to convert the thermodynamic state space variables (χ_1, χ_2) into a square coordinates (κ_1, κ_2).

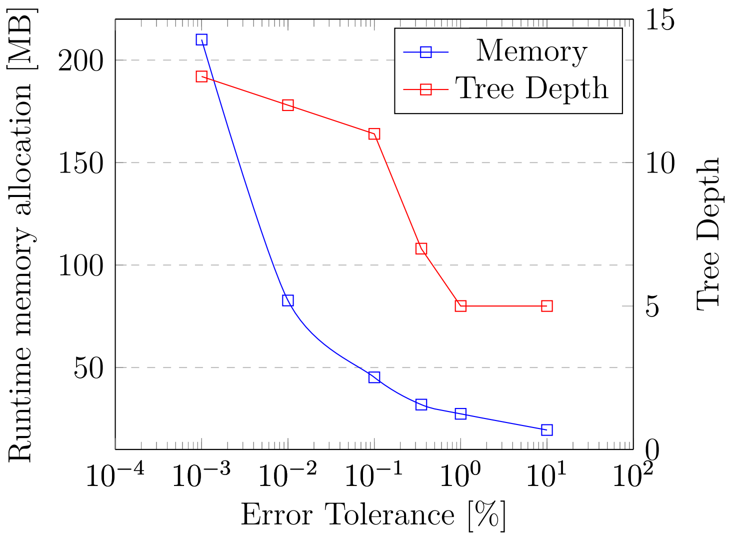


Figure 3.7: Memory allocation of Block structured AMR tabular EOS during runtime in Random Access Memory, for the physical conditions range; Pressure: 1.0 MPa - 10.0 MPa, Temperature: 100K - 900K, Oxygen. Here the error is calculated in reference to underlying backend EOS

In terms of size of files that need to loaded into the Random access memory (RAM) during the simulation runtime, it varies corresponding to the accuracy and range within which are are constructing the table. Figure 3.7 shows the runtime memory allocation with respect to error tolerance for a fixed range of physical conditions i.e., Oxygen in pressure (1.0 MPa - 10.0 MPa) and temperature (100K - 900K). Also the table is size is proportional to the maximum resolution to which the subdivision was allowed.

3.3.4 Visualization of the tabular thermodynamics

To visualize the generated tabular EOS, we recursively plot the cells, to provide a easier understanding readers can compare the below plots to the phase diagram (figure 1.1). The below (P, T) tables are for oxygen, with decreasing order of error tolerances (0.1%, 0.01%, 0.001%) and (ρ, e) table is also presented (figure 3.11). The evaluation ranges are $[P_{min}, T_{min}] = [0.02E06 \text{ Pa}, 85 \text{ K}]$, $[P_{max}, T_{max}] = [10E06 \text{ Pa}, 600 \text{ K}]$. Whereas in the case of (ρ, e) table, boundaries are $[\rho_{min}, e_{min}] = [0.14 \, \frac{Kg}{m^3}, 50000 \frac{J}{Kg}]$ and $[\rho_{max}, e_{max}] = [130.20 \, \frac{Kg}{m^3}, 300000 \frac{J}{Kg}]$

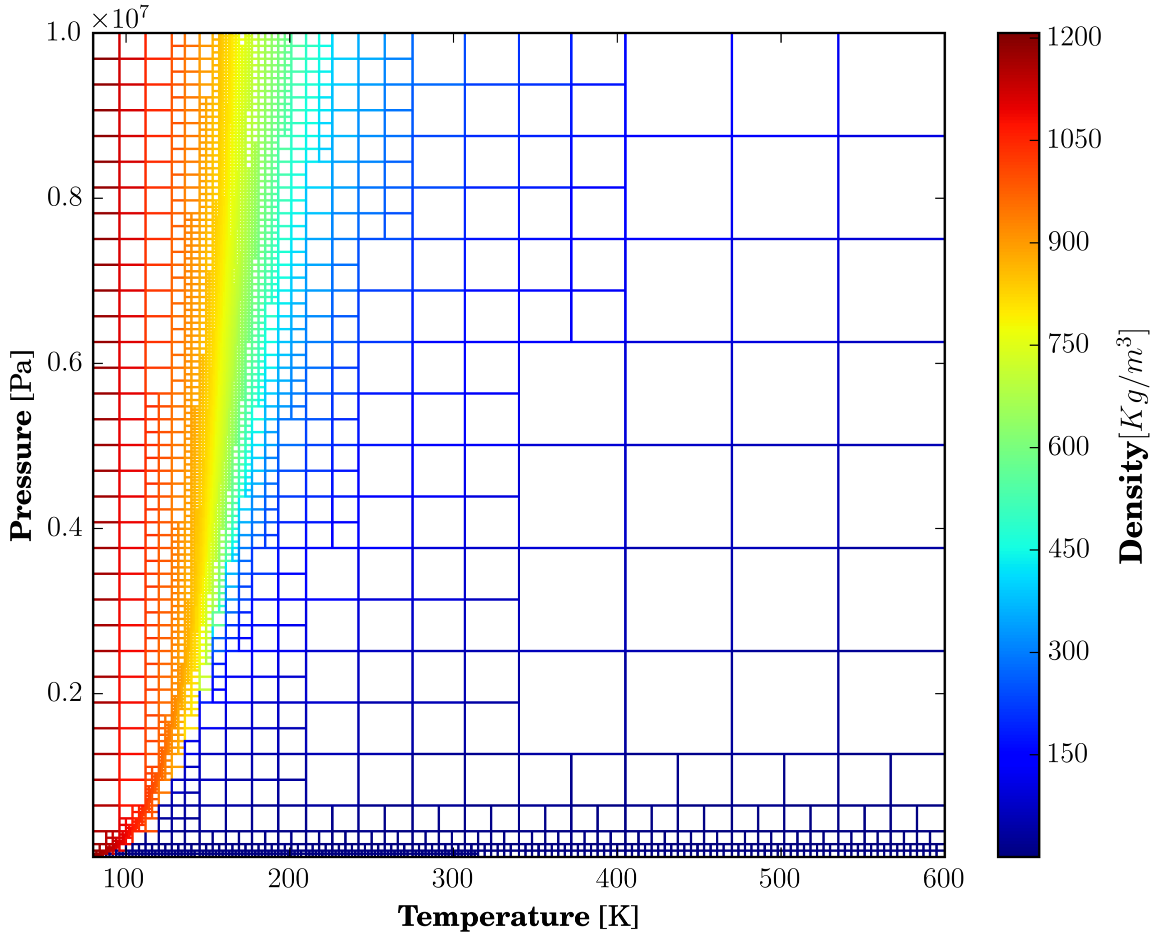


Figure 3.8: Block structured AMR Pressure(P) -Temperature(T) lookup table for oxygen with 0.1% error tolerance compared to NIST's REFPROP database.

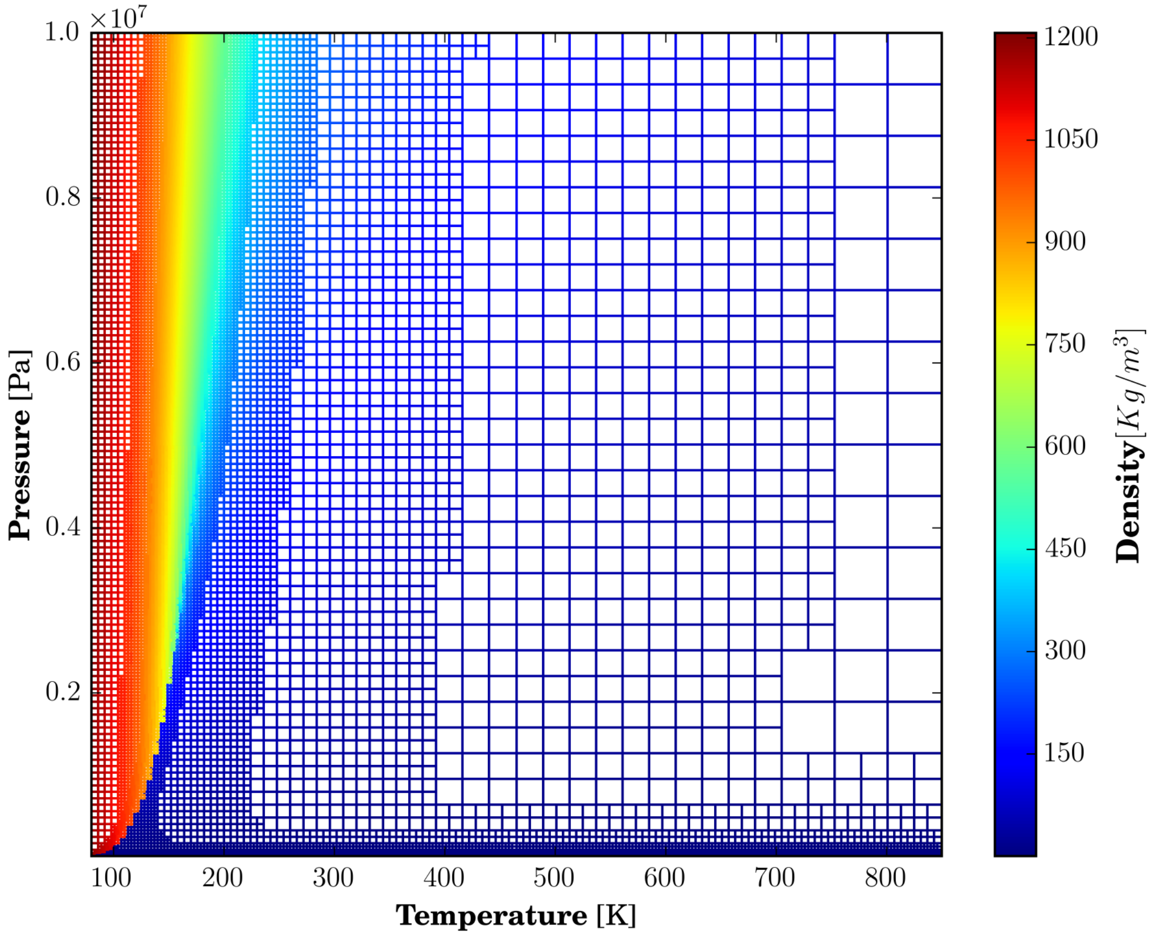


Figure 3.9: Block structured AMR Pressure(P) -Temperature(T) lookup table for oxygen with 0.01% error tolerance compared to NIST's REFPROP database.

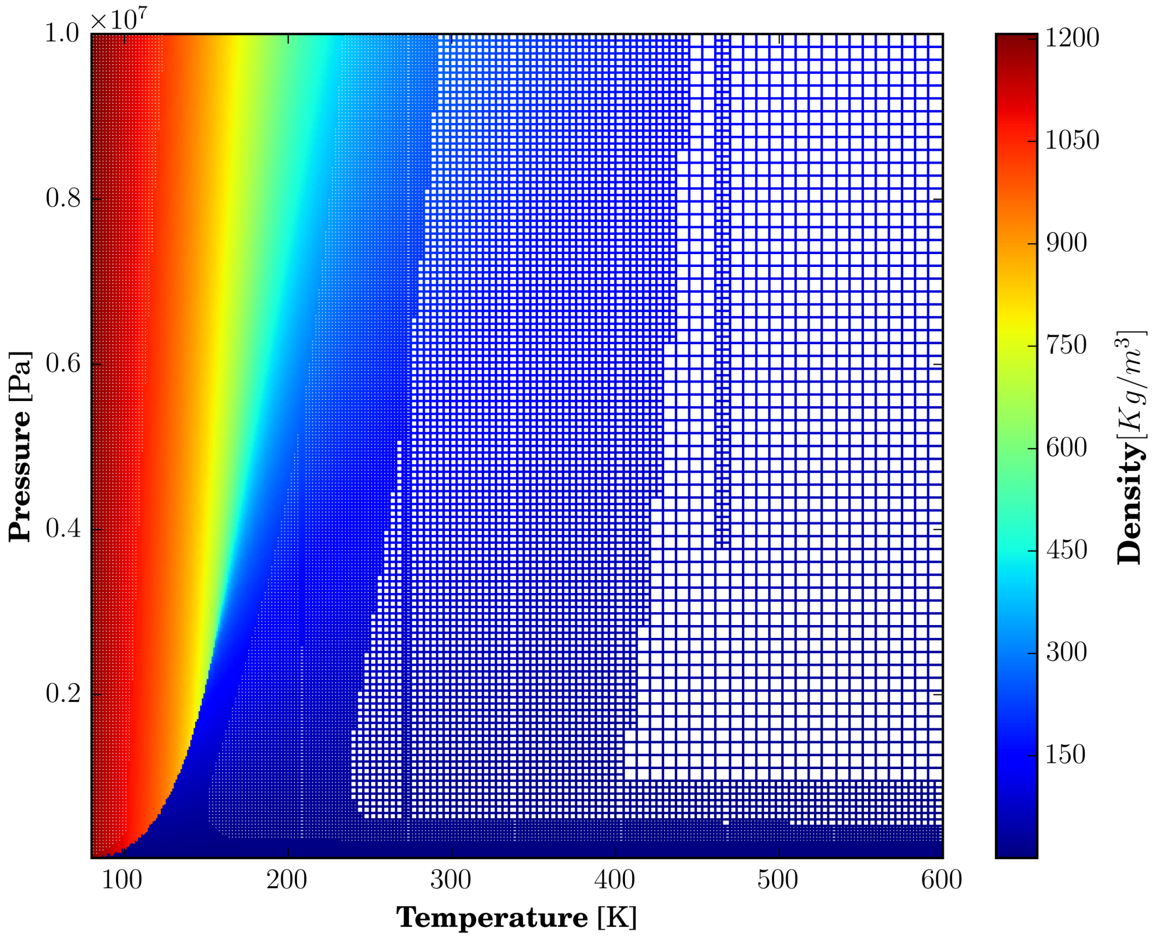


Figure 3.10: Block structured AMR Pressure(P) -Temperature(T) lookup table for oxygen with 0.001% error tolerance compared to NIST's REFPROP database.

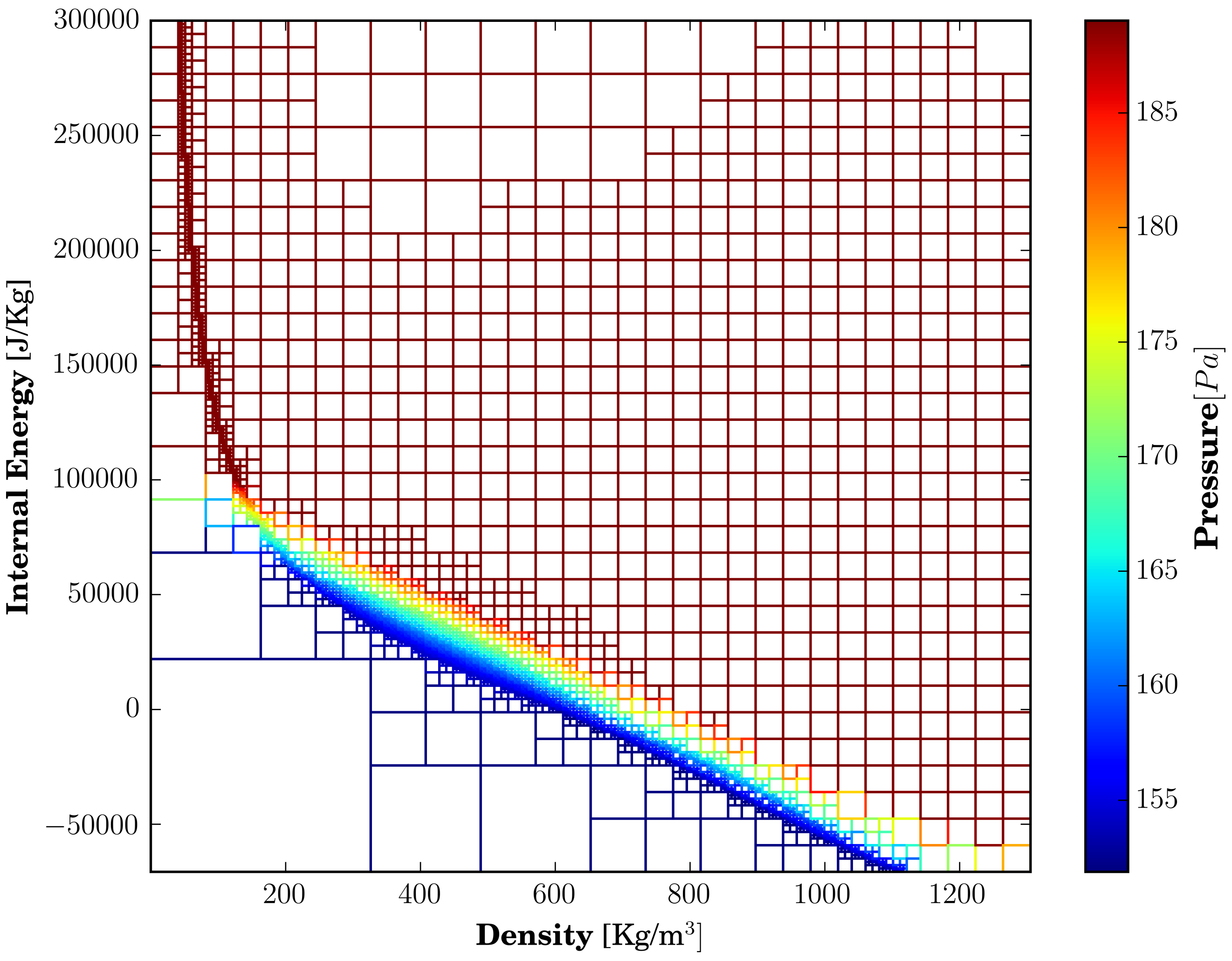


Figure 3.11: Block structured AMR lookup table for oxygen with 0.01% accuracy compared to NIST's REFPROP database., generated in (ρ, e) state space

3.4 AMR with Bezier patch fitting tabulation [31]

This method is a work done by Matthew Yao[31], for the block structured AMR table, the underlying criteria for refinement is based on the multivariate approximation. Rather than tabulating solely on the basis of bilinear interpolation error we can implement Bezier surfaces/patches that are approximated functions. This function of two independent variables is [19], here, $B_i^3(u)$ are the third-degree (fourth-order) Bernstein polynomials, b_{ij} are the control points.

$$F(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i^3(u)B_j^3(v)b_{ij} \tag{3.2}$$

$$B_i^3(u) = \frac{3!}{i!(3-i)!}(1-u)^{3-i}u^i \tag{3.3}$$

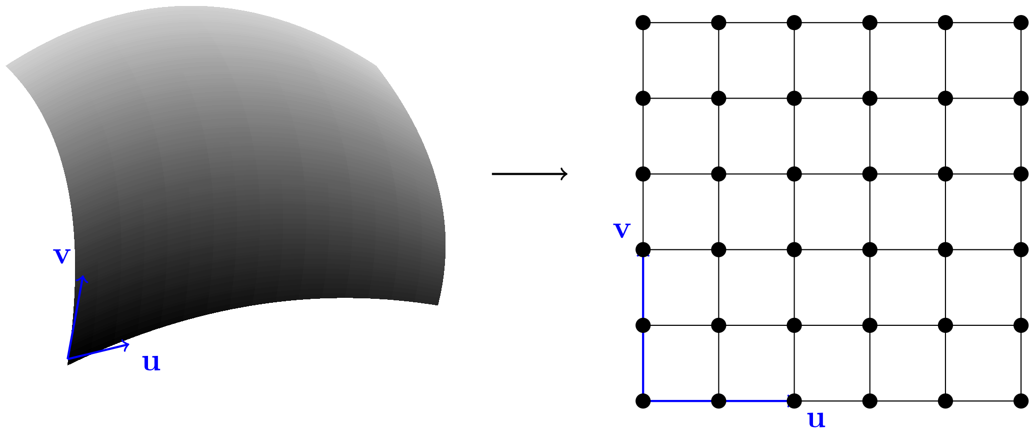


Figure 3.12: Sample Bezier patch being adapted into local grid

Using the linear combination of these b_{ij} control points, the shape of the surface is determined. The surface is generated by recursively subdividing the domain until the resulting Bezier patches are capable of reconstructing the original value to a specified error threshold or if the maximum refinement level is reached. The table generation process is similar to the previously described Block structured AMR, we use a quad tree to recursively refine the thermodynamic domain by dividing into four patches and evaluate the properties within each patch.

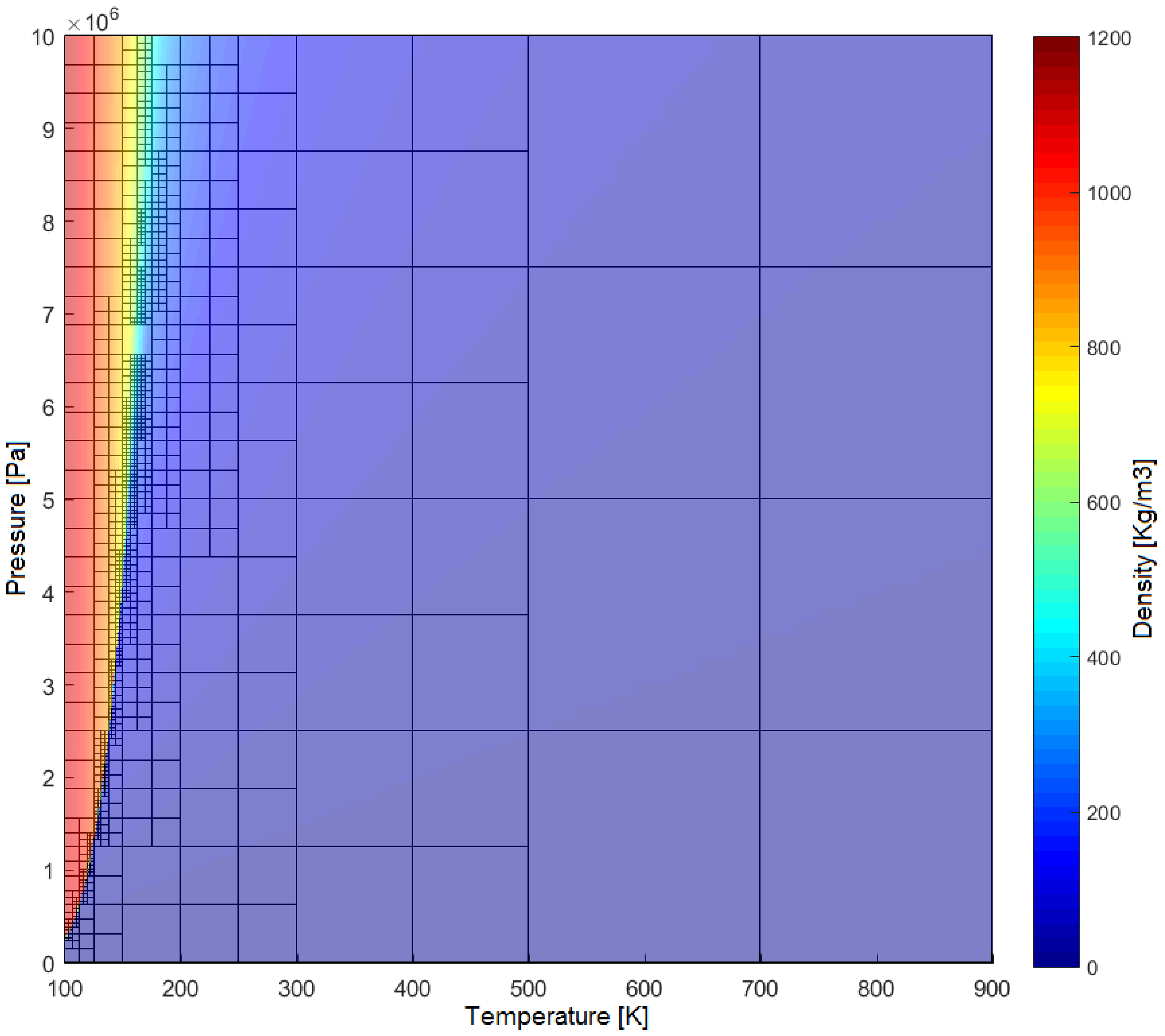


Figure 3.13: Bezier patch based Tabulated EoS for Density(ρ) adaptive refinement using quadtree, reproduced with permission from Matthew Yao[31]

3.5 Thermodynamic consistency

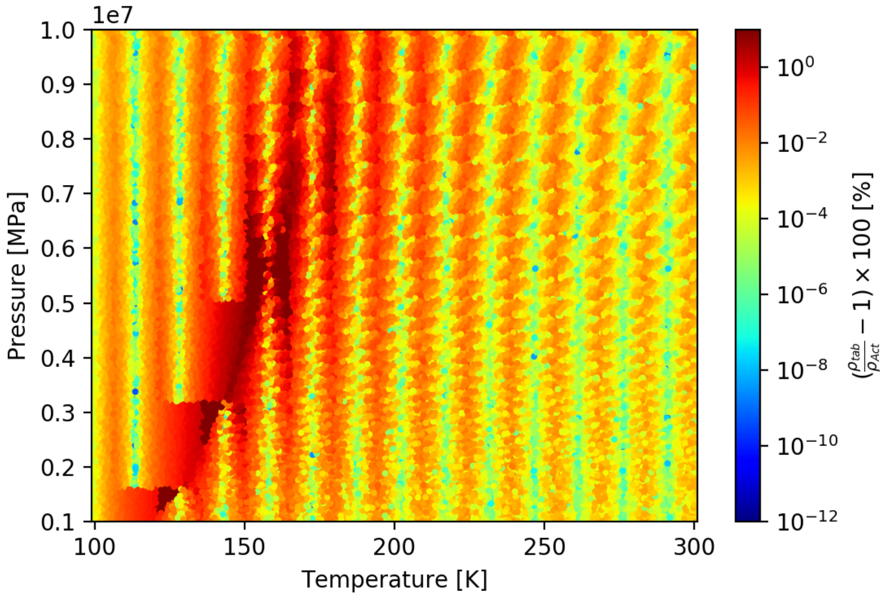
A valid EOS must satisfy the thermodynamic condition of consistency. Maintaining thermodynamic consistency involves satisfying the Maxwell relations [61] and also thermodynamic properties such as pressure(P), density(ρ), internal energy(e) must confine to their definitions in these derivatives. A tabular EOS may exhibit wide range of behaviour over the range of temperature and density if it is not consistent. Such behaviour manifests itself as sharp discontinuities in thermodynamic property evaluation near phase boundaries. In CFD codes, it is important to have a tabular interpolation scheme that can produce consistent thermodynamic properties. Even insignificant inconsistencies of the order of $\approx 10^{-5}$ can pose difficulties for the implicit solution of underlying governing equations. In order to be able to construct accurate fluxes, we must be able to know how to evaluate physically realistic values of several dimensionless quantities, thermodynamic derivatives. For example, if we express internal energy E as a function of entropy(S) and specific volume(V). The thermodynamic property definitions of temperature(T) and pressure(P) are

$$T = -\left.\frac{\partial E}{\partial S}\right|_V, P = -\left.\frac{\partial E}{\partial V}\right|_S \tag{3.4}$$

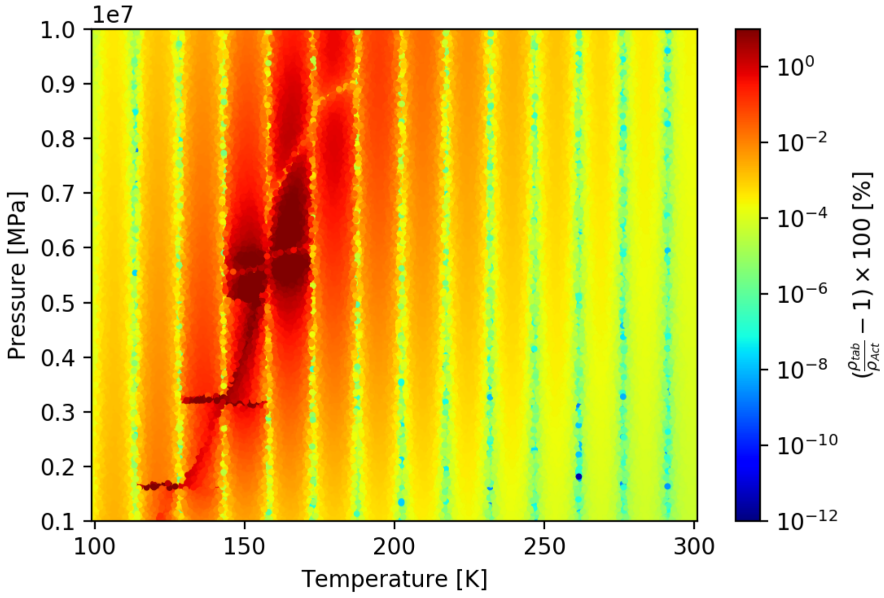
This is one of the thermodynamic consistency condition. With the bilinear interpolation, both the density and its first order derivatives vary linearly over each local rectangular cell. Nevertheless, as the thermodynamic properties stored on the mesh nodes are themselves obtained from a consistent thermodynamic database such as NIST, CoolProp), in a fine tabulation such intrinsic inconsistencies are going to be small. [73]

3.6 Error Estimates & Verification

To verify and understand the accuracy of the tabulation methods, we uniformly random sample the entire tabulation domain (T-P, ρ-E) and evaluate the error in density evaluation. Starting with the homogeneous tabulation and considering two interpolation schemes i.e., bilinear and bicubic the contour error plots are shown in figure 3.14a, 3.14b. when compared both, bicubic interpolation performed well in terms of maintaining accuracy. Except in the vapour-liquid curve and the supercritical state. This is attributed to the usage of derivatives across the interpolating nodes and thereby making bicubic interpolation C^1 continuous, which is the feature lacking in bilinear method as can be seen in the figure 3.14a the erroneous regions form patch like shape in between the grids compared to much smoother transition observed in the latter case.



(a) Tabulation with bilinear interpolation



(b) Tabulation with bicubic interpolation

Figure 3.14: Homogeneous Tabulation Relative Error[%] in Density estimates

The T-P tabular grid is sampled into 10000x10000 linear-linear grid (figure 3.15), log-linear (figure 3.17). In each patch 100 points are sampled and then the maximum relative error in density is computed. As seen from the relative error contour plot, the maximum error is along the phase change line or where the density changes rapidly. Except for that region the uniform accuracy is maintained as a result of adaptive subdivision. The highest error points lies close to the critical point. The number of patches required to capture the thermodynamics is different in three approaches (Homogeneous, Adaptive, Bezier patch) The highest being the homogeneous due to its uniform spacing, it will take enormous number of patches to get the error close to 0.01%, although Bezier patch has less number of patches it can only accommodate one property at a time, which means it requires individual table for each property. The Block based AMR has moderate number of patches that are not oversized but can tabulate all thermodynamic properties.

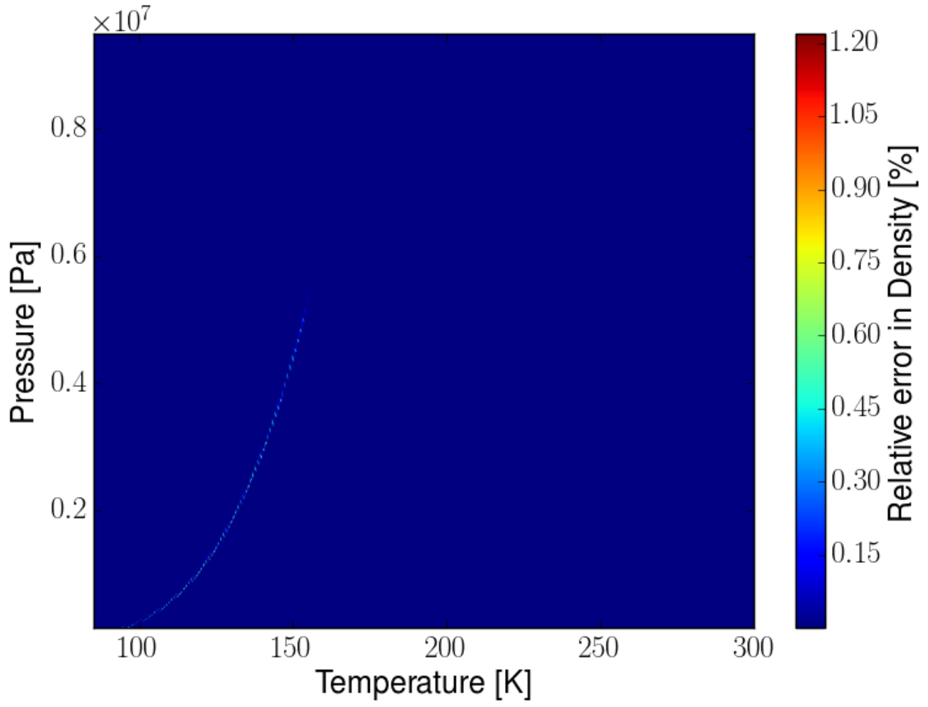


Figure 3.15: Relative error in density for the Block structured adaptive tabulation

The comparison of various tabulation approaches demands a quantitative thermodynamic error metric. As the exactness thermodynamic properties remain a subject of scientific relevance, the current study evaluates the error relative to the NIST database. These data points represent the highest fidelity data set for the computation of the thermodynamic and thermophysical properties and constructed from an ensemble of experimental data supplemented with Modified Benedict-Webb-Rubin curve fitting.

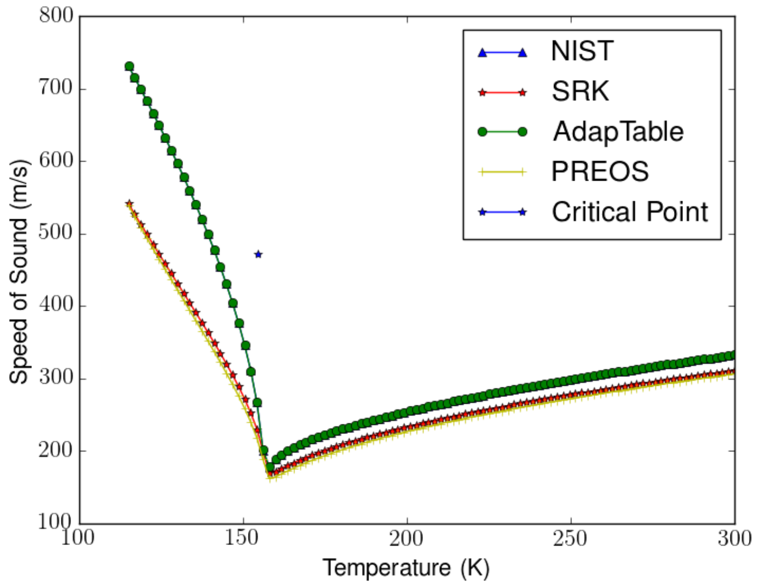


Figure 3.16: Speed of sound for Oxygen computed with REFPROP, SRK, PR, tabulated AMR EOS (AdapTable)

Table 3.1: Error estimates

Grid type	Block based AMR (0.0001 tol.)	Bezier Patch AMR (0.0001 tol.)
Total patches	12648	1120
% patches with rho error >0.001	0.101	-
Maximum rho error %	1.2	-

To evaluate different types of equation of states, we employ few statistical techniques such as Average Percent Relative Error (ARE%), calculated as a reference measure

which is defined below is a measure of the bias of the correlation; a value of zero indicates a random of the measured values around the correlation.

$$ARE\% = \frac{100}{N_d} \sum_{i=1}^{N_d} \frac{(\rho_i^{exp.} - \rho_i^{calc.})}{\rho_i^{exp.}} \tag{3.5}$$

The AARE%, is the arithmetic average of the absolute values of the relative errors; is an indication of the accuracy of the correlation.

$$AARE\% = \frac{100}{N_d} \sum_{i=1}^{N_d} \frac{|\rho_i^{exp.} - \rho_i^{calc.}|}{\rho_i^{exp.}} \tag{3.6}$$

The R^2, is the correlation coefficient; is a measure of the precision of fit of the data. If data are perfectly correlated, then $R^2 = 1$. A small value of AARE% and R^2 close value to one (simultaneously) denote a good correlation based on good data.

Another parameter, Sum of Absolute of Residual (SAR) shows the reliability of correlation for higher order data points.

$$SAR\% = \sum_{i=1}^{N_d} |\rho_i^{exp.} - \rho_i^{calc.}| \tag{3.7}$$

The statistical parameters of the well known EoSs are listed in table 3.2

Table 3.2: Error metrics comparison with NIST

Equation of State	AARE	ARE	SAR [Kg/m^3]	R^2
Peng-Robinson PR	3.015	0.993	70.600	0.991
Redlich-Kwong RK	4.409	-0.026	89.816	0.987
AMR Tabular EOS	1.022	0.0021	91.232	0.995
Soave-Redlich-Kwong SRK	6.388	6.381	131.009	0.990

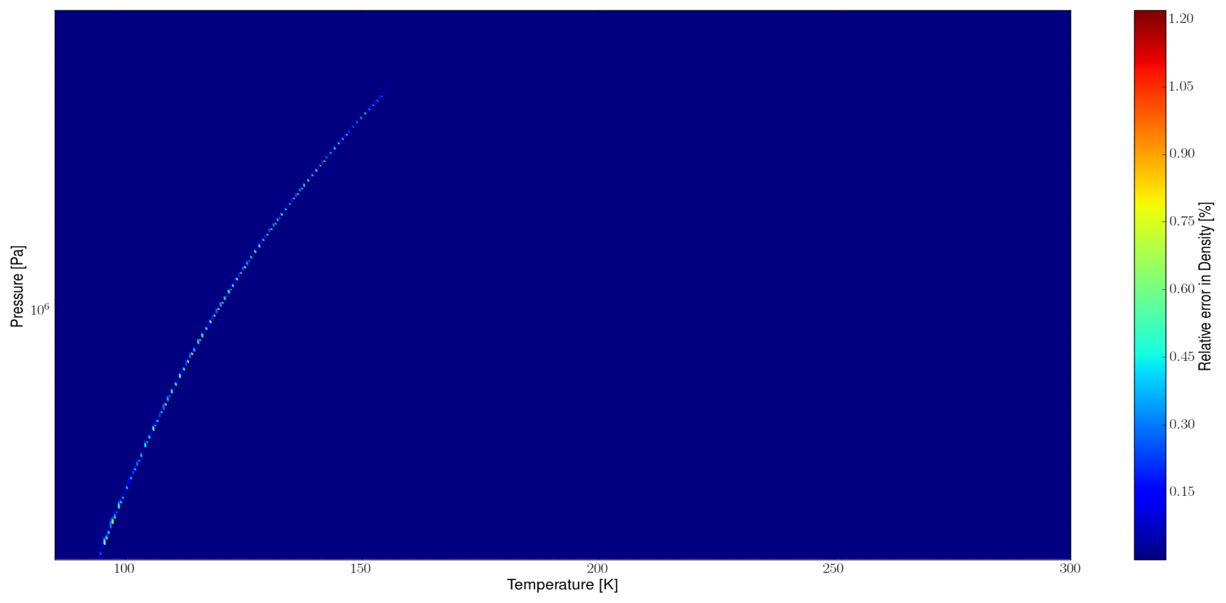
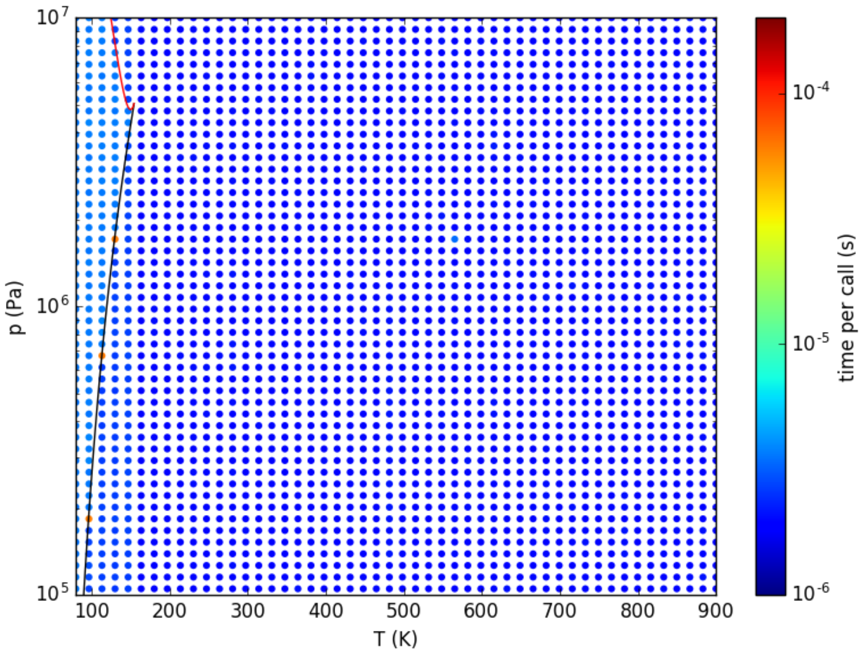


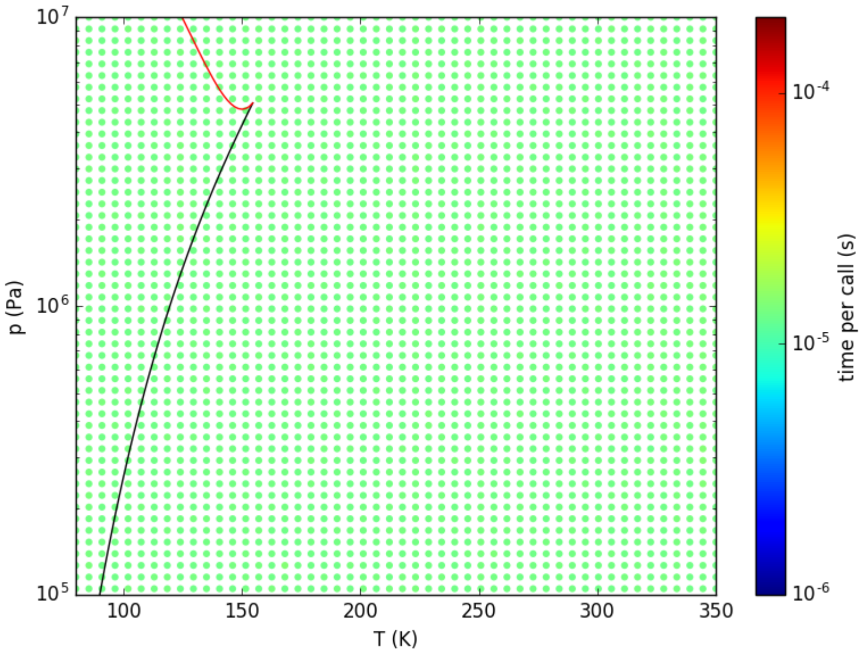
Figure 3.17: Error Validation (Log-Linear plot)

3.7 Computational cost comparison

The increase in speed of thermodynamic properties evaluation compared to cubic EOS is one of the primary motivations of this thesis, here we show an example of the speedup of computations. When (ρ, e) pair is evaluated from uniformly generated (P, T) combination in the nearcritical regime. For each pair iteration, 100000 loops/calls are executed and the slowest run time is tabulated. A contour plot of such calls is presented in figure 3.18. In comparison, the tabular lookup calls are 10 times faster than the calls to cubic EOS which in this case is chosen as Peng-Robinson EOS, it involves an iterative root-finding process and for this comparison its convergence tolerance is set to $1E{-}06$. Whereas in the tabular EOS, the searching algorithm (for a binary search tree it is $\mathcal{O}(n)$) and interpolation scheme (here it is a bilinear) influence the lookup time, the adaptive tabulation is comparably denser in the near critical regime, because of high non linearity, comparatively higher lookup search times are observed for those regions as shown in the figure 3.18a



(a) Block based tabulated AMR lookup computational cost in μ sec



(b) PR EOS evaluation computational Cost in μ sec

Figure 3.18: Computational cost (μ sec) comparison over Density($\frac{Kg}{m^3}$) evaluation calls.

3.8 Harmonic Acoustic wave test case

To test the computational performance and significance of using a tabular EOS, a one-dimensional acoustic wave propagation in supercritical fluid is considered, Using periodic boundary conditions at both sides a harmonic wave is initialized in supercritical conditions. The computational domain is $x \in [0, 10]$ m and 100 grid points are used, giving a uniform grid spacing $\Delta x = 0.1m$. With fourth order accurate schemes in both spatial and temporal scales, both cubic (PREOS) and tabular EOS are used. The sound pressure level $L_p = 23$ (dB) is calculated according to:

$$L_p = 20log_{10}(\frac{\Delta p}{p_{ref}\sqrt{2}}) \tag{3.8}$$

Where Δp is the amplitude of the pressure harmonic wave (Pa).

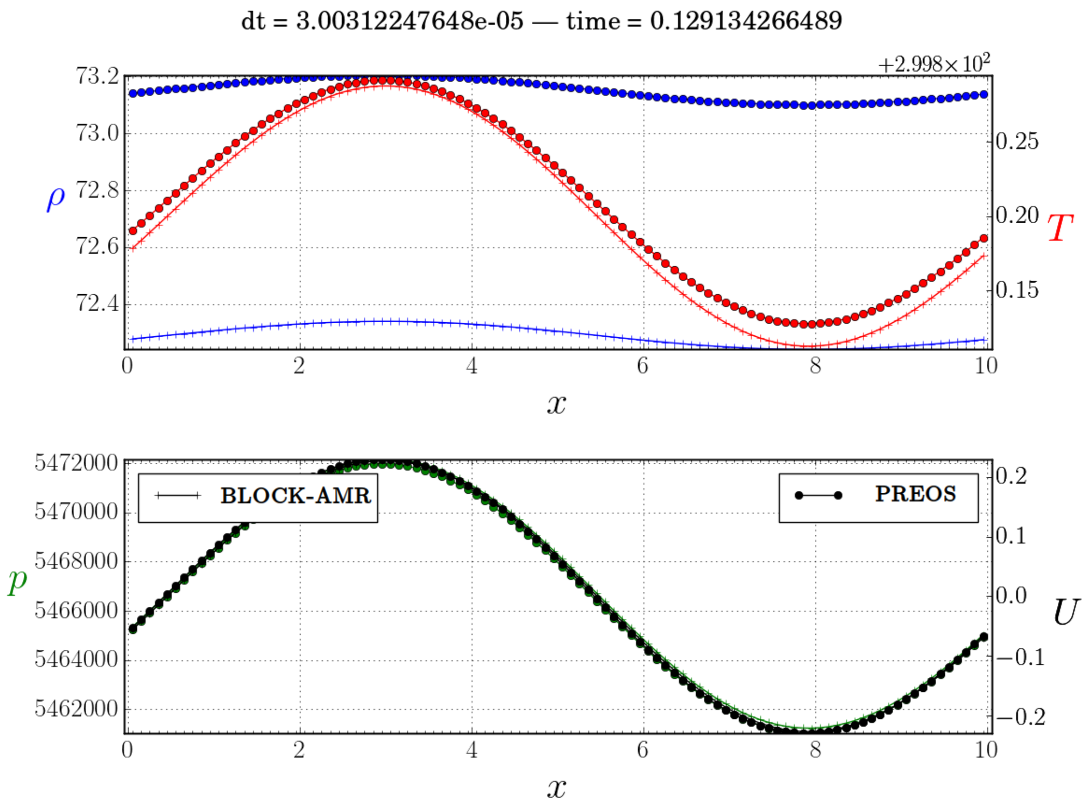


Figure 3.19: Harmonic Acoustic wave propagation; PREOS (•) vs AMR (-)

As shown in the figure 3.19, there is a distinction in the form of density by using accurate tabular real fluid EOS (Block-AMR) rather than conventional cubic PREOS. It is within the usually observed relative error, but it is important to mention that slight variations in temperature and pressure can cause significant change in density in the nearcritical regime.

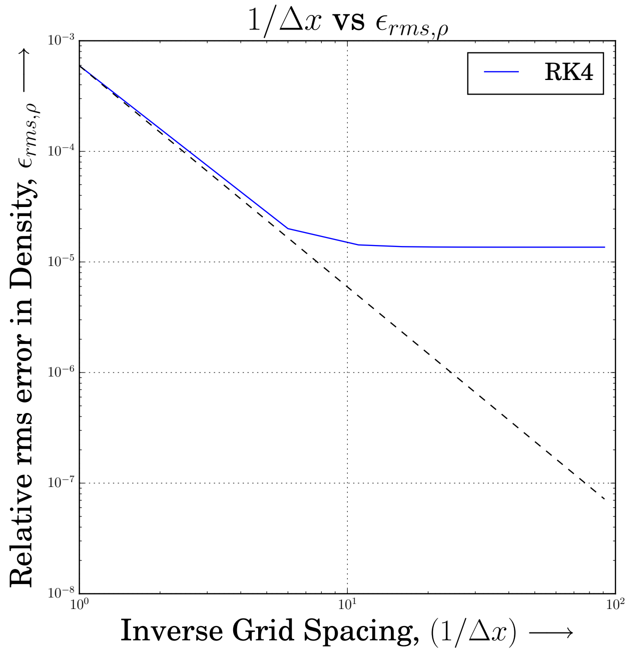


Figure 3.20: Grid convergence plot for the acoustic wave propagation case.

3.8.1 Computational cost comparison

To investigate how the choice of EOS affects the CFD computational cost (wall time in seconds) an experiment is performed with the above mentioned acoustic test case, the simulation is repeated using the available different EOS such as cubic (PREOS), tabular (Block-AMR, Bezier patch) and perfect gas law. The computational cost in seconds in plotted against increasing grid size in figure 3.22. It is important to mention here that the simulation code was written in Python, parallelized and ran on a CPU (Intel Core i5-4590 CPU 3.30GHz $\times$ 4) and utilized fourth order accurate schemes in both spatial and temporal scales. Hence the computational cost time is different from what is observed in a typical 1D simulation written in compiled languages such as C/C++. When we compare tabular lookup based thermodynamic property evaluation and cubic polynomial equations. There are two important aspects that play a key role in influencing the time complexity in AMR and cubic EOS, they are 'lookup approach' and 'iterative solution method' respectively. The lookup approach here refers to the procedure to search/query the table for a given physical conditions (thermodynamic variables) and the iterative solution method refers to the repetitive root finding method such as Newton-Raphson solver, Bi-section method that is employed to obtain the root of the polynomial expression.

$$X_{n+1} = X_n - \left[\frac{f(X_n)}{f'(X_n)}\right] \tag{3.9}$$

In the latter case, the time complexity depends on how good the initial guess value (x_0) is, i.e., how close the initial guess value is to the root value. Further, during each iteration there will be a reduction of guess value into half $\frac{x_0}{2}$ over and over. So approximately, it takes $\frac{1}{2}log_2(x_0)$ steps to reach within the vicinity of the supposed solution value of $\sqrt{x_0}$. After which, there will be a convergence process to zero-in onto the exact solution value. Here, the number of iteration steps that are required would be depended on the error tolerance value (not on the "Initial guess"). Thereby, the time complexity scales in the $\mathcal{O}(log(n))$

In the case of lookup search/query, the conventional data structures are multidimensional array, binary tree, quad-tree. Only the multidimensional array has a least worst case time complexity in access ($\mathcal{O}(1)$). Whereas for the binary, quad-trees, K-D tree the worst case time complexity in accessing an element is $\mathcal{O}(n)$. k-d tree has order of complexity $\mathcal{O}(k.N^{1-1/k})$ in the worst case, where 'k' being the number of dimensions.

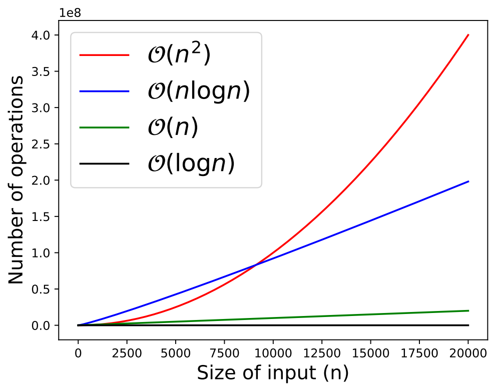


Figure 3.21: Big-O complexity plot

In our experiment we ran harmonic acoustic wave simulations with different types of EOS available namely, cubic, tabular and perfect gas. With a timestep size of $\Delta t = 3.003E-05s$, we ran until time $t = 0.129s$ starting with a grid size of 100 grid points, we increased grid size and plotted computational cost vs grid size as shown in figure 3.22. The key findings are that the cubic EOS involving iterative process of root finding had its computation cost vary quadratic with respect to the grid size. Upon an approximate logarithmic scaling analysis, the computation cost varies as

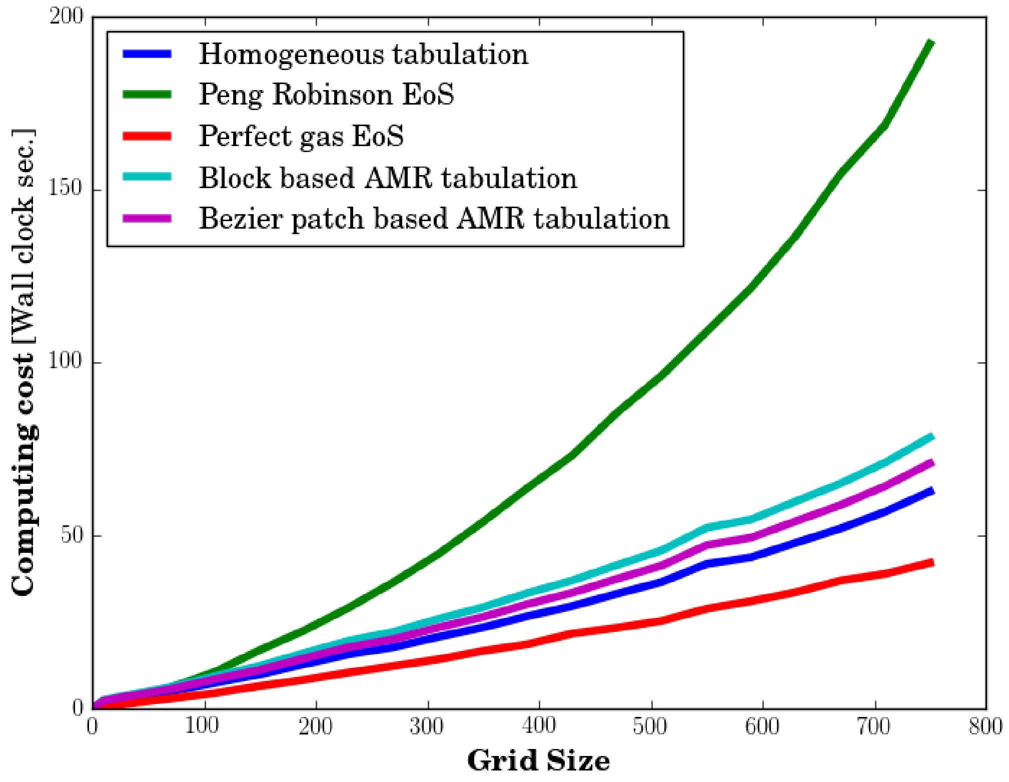


Figure 3.22: Computational cost (CC) comparison with respect to grid size $[Nx]$ for 1D-Acoustic wave problem with real fluid thermodynamics

$$\text{Computational cost}(CC) \approx \begin{cases} 0.000252[Nx]^2 + 0.058[Nx], \text{PREOS}, \\ 0.000023[Nx]^2 + 0.0394[Nx], \text{Block-AMR Tabulation }, \\ 0.000023[Nx]^2 + 0.0372[Nx], \text{Bezier-AMR Tabulation}, \\ 0.000023[Nx]^2 + 0.0352[Nx], \text{Homogeneous tabulation}, \\ 0.000003[Nx]^2 + 0.0324[Nx], \text{Perfect gas law} \end{cases}$$

3.9 Conclusions

Thermodynamic properties evaluation using tabulated EOS is implemented and various forms of tabulation such as homogeneous, adaptive grid based, higher degree polynomial based versions are developed, tested for their computational performance, accuracy and compared to the existing cubic EOS. In homogeneous tabulation the accuracy is heavily constrained to the properties range within which the table is being constructed, this is due to the limitations from cache memory perspective due to a uniform fine grid.

To avoid such limitations, adaptive tabulation methods are researched and efficient tabulation in the form of Block-structured AMR is presented. Although, it overcomes the table range limitations and being able to lookup with minimal computation cost, it requires additional storage in the form of predefined uniform index arrays (figure 3.5), overlapping index files. By avoiding the need to use any iterative solution methods to obtain thermodynamic properties, tabulation methods save a lot of computational cost as the experiment on the harmonic acoustic wave supports the claim.

Chapter 4

Implementation details of CFD Solver

In this chapter we provide a detailed overview of numerical methods involved in the TwoDThermoCode, the CFD solver extended as part of this thesis. The code is developed on top of Pyro2 [?], an Astrophysics simulation code. Additional features that was added for this thesis work are:

- Real fluid thermodynamic equation of state. (CoolProp database, Tabulated EOS)
- HLLC Riemann solver adapted to the real fluid thermodynamics
- Higher order time stepping scheme (RK4)

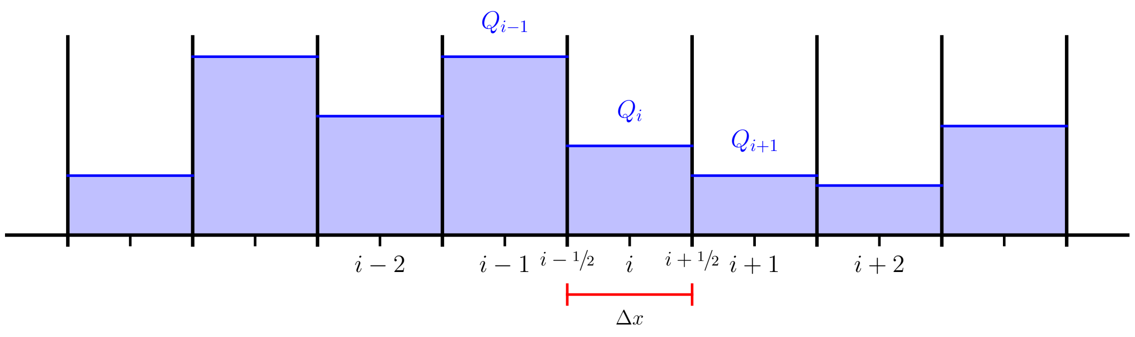


Figure 4.1: A representation of finite volume grid where an averaged value represents the variable throughout each cell [Figure reproduced based on [68]]

4.1 Conservative discretization of governing equations using Finite Volume methods

Assuming a conservation law in the form,

$$q_t + f(q)_x = 0 \tag{4.1}$$

where, q is a flux quantity, f is a flux function. We can discretize this in a one dimensional domain with the cell edges at $[x_{i-1/2}, x_{i+1/2}]$, as shown in the figure 4.1. The numerical solution (Q_i^n) will be the volume average of the true solution $q(x, t_n)$ over the cell. In simple terms, when we run a finite volume CFD solver we are not getting a pointwise value, instead we obtain the cell averaged value to the solution. By defining cell averages over the interval $C_i = [x_{i-1/2}, x_{i+1/2}]$, we can write the numerical solution as,

$$Q_i^n = \frac{1}{\Delta x}\int_{C_i} q(x, t_n)dx \tag{4.2}$$

The time rate of change of the true solution $q(x, t_n)$ cell average is,

$$\frac{d}{dt}\int_{C_i} q(x,t)dx = -\int_{C_i}\frac{d}{dx}f(q(x,t))dx = \underbrace{f(q(x_{i-1/2},t))}_{\text{Flux in}} - \underbrace{f(q(x_{i+1/2},t))}_{\text{Flux out}} \tag{4.3}$$

On integrating equation 4.3 in time, we obtain the time average value at time t_{n+1} as

$$\int_{C_i} q(x, t_{n+1})dx = \int_{C_i} q(x, t_n)dx + \int_{t_n}^{t_{n+1}} [f(q(x_{i-1/2}, t) - f(q(x_{i+1/2}, t)]dt \tag{4.4}$$

Using numerical fluxes, we can write the update as,

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}\left[F_{i+1/2}^n - F_{i-1/2}^n\right] \tag{4.5}$$

$$\frac{Q_i^{n+1} - Q_i^n}{\Delta t} + \frac{F_{i+1/2}^n - F_{i-1/2}^n}{\Delta x} = 0 \tag{4.6}$$

Which resembles the conservation law equation 4.1 and we want to approximate the numerical fluxes $(F_{i+1/2}^n, F_{i-1/2}^n)$ for a given flux function $f(q(x,t))$ and explicit time stepping scheme at cell interfaces $[x_{i-1/2}, x_{i+1/2}]$, in the form of,

$$F_{i-1/2}^n = \mathbf{F}(Q_i^n, Q_{i-1}^n), F_{i+1/2}^n = \mathbf{F}(Q_{i+1}^n, Q_i^n) \tag{4.7}$$

where $Q_{i+1}^n, Q_i^n, Q_{i-1}^n$ are the cell averages. So, in simple terms we are trying to find the flux across the edges using the pointwise cell average values. This leads us to a classic Riemann problem of solving the hyperbolic conservation law with an initial constant piecewise data to obtain an analytic solution, explained in further sections (4.2, 4.3)

4.2 The Riemann problem

As we know that the Euler equations are a non-linear system of hyperbolic partial differential equations representing the conservation laws, obtaining an exact solution is a complex task, yet Godunov [23] proposed a scheme to obtain exact solutions to such governing equations using the Riemann problem, which can be defined as an Initial Value Problem for the hyperbolic conservation laws 4.1

$$U_t + \left(F(U)\right)_x = 0 \tag{4.8}$$

where,

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho E \end{pmatrix} \quad F(U) = \begin{pmatrix} \rho u \\ \rho u u + p \\ \rho u E + up \end{pmatrix} \tag{4.9}$$

with initial conditions,

$$U(x, t = 0) = \begin{cases} U_l & x < 0 \\ U_r & x > 0 \end{cases} \tag{4.10}$$

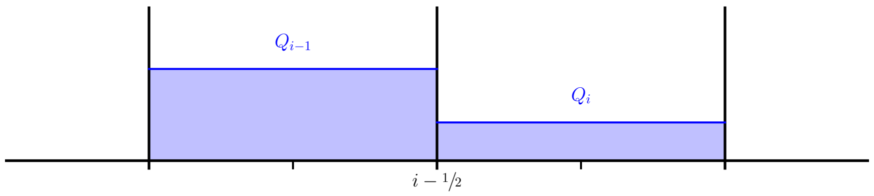


Figure 4.2: At time t = 0. (set from initial condition)

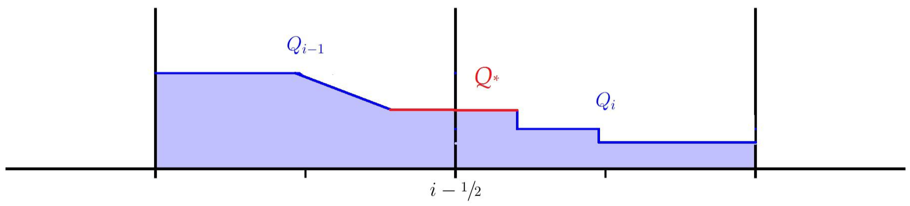


Figure 4.3: At time $t > 0$

After time $t > 0$, if we look at the solution along the red line highlighted in figure, due to the initial condition assumed, the solution $Q*$ will be constant for a small finite amount of time Δt. We use this $Q*$ to evaluate the numerical flux at the interface, i.e., $F_{i-1/2} = f(Q*)$. This process is known as the classical Godunov [23] approach to solve the hyperbolic conservation law. Although it may be able to resolve shocks and rarefactions, it does so by solving conservation law at each and every cell's interface causing huge computational overhead. So, a typical CFD code doesn't employ Godunov's approach to solve Riemann problem.

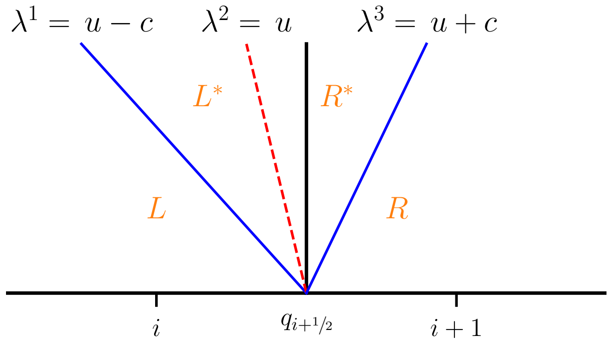


Figure 4.4: Representation of three wave patterns $(\lambda^{(+)}, \lambda^{(o)}, \lambda^{(-)})$ of approximate Riemann solution, 4 different regions. Where x-axis represents space and the y-axis represents time. [Figure reproduced based on [68]]

The Riemann problem has three waves as shown in the figure 4.4, corresponding to the three eigenvalues of the matrix $\lambda^1, \lambda^2, \lambda^3$ and the these waves separate four constant states namely L, L^*, R^*, R. The region between two non-linear waves (L, R) is known as "Star Region" [68], which contains the contact discontinuity. If the flow is supersonic, the two non-linear acoustic waves can be either shocks or rarefactions leading to four possible combinations of wave patterns as shown in figure 4.6.

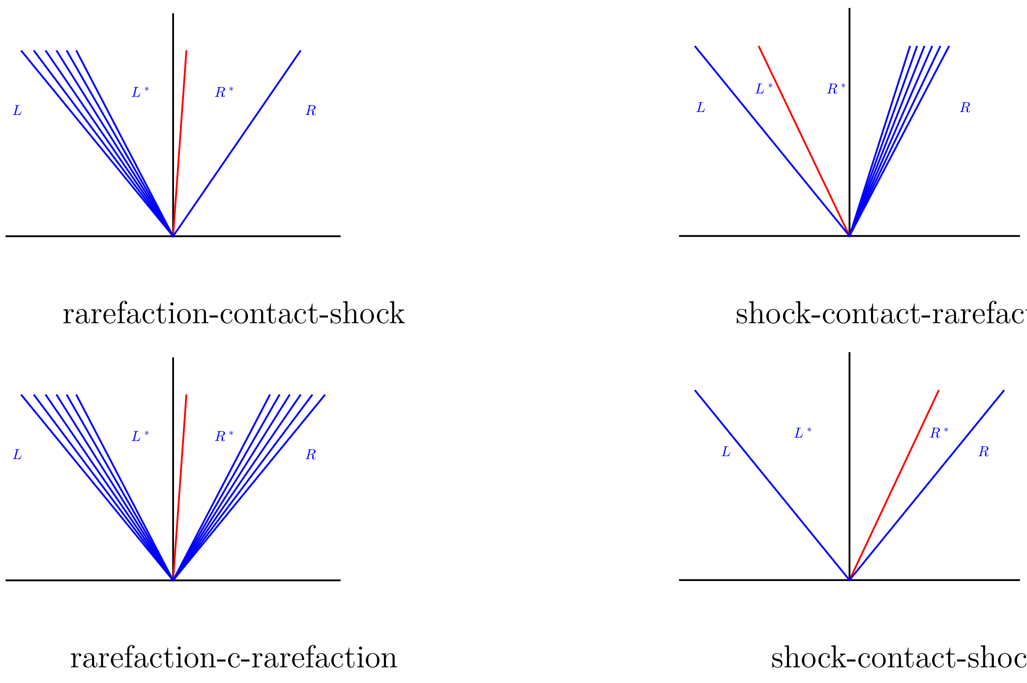

L
L^*
R^*
R
rarefaction-contact-shock
L
L^*
R^*
R
shock-contact-rarefaction
L
L^*
R^*
R
rarefaction-c-rarefaction
L
L^*
R^*
R
shock-contact-shock

Figure 4.5: The possible wave pattern combinations to the solution of Riemann problem. (Figure reproduced from [68])

4.3 Riemann solvers with real fluid thermodynamics

At the heart of many finite volume compressible CFD codes is a Riemann solver, using which the numerical inviscid fluxes are computed. The Riemann problem described in 4.9 with suitable boundary conditions 4.10, can be solved by an explicit conservative method as in equation 4.3. In which the flux terms $F^n_{i+1/2}$ is classified into two types based on the approach in which the wave propagation information is being used, either explicitly or implicitly. They are Godunov or Upwind type fluxes and non-upwind or centered fluxes. In the Godunov type fluxes we use the prior wave propagation information explicitly. Although we know how to obtain an exact solution to the Riemann problem using Godunov method, the reason why we introduce an approximate Riemann solver is because of the requirement to solve the conservation law at every cell interface over each iteration thereby making the solution computationally expensive. According to Toro[68], there are two ways to get an approximate solution to the Riemann problem, the first one involves obtaining an "approximation to the numerical flux" and the second is to obtain an approximate interface state and solve for the flux function at that state. The methods HLL, HLLC solvers are categorized in this manner.

4.3.1 HLL Riemann solver

Proposed by Harten, Lax, Van Leer[29], this solution method involves a direct approximation of the flux at the interface state (single state in between two cells). The solution is assumed to be in a wave configuration consisting of two waves separating three constant states. The approximating numerical interface flux is given [68] as,

$$Q_{i+\frac{1}{2}} = \begin{cases} Q_L & 0 \leq S_L \\ Q_{hll} & S_L \leq 0 \leq S_R \\ Q_R & 0 \geq S_R \end{cases} \tag{4.11}$$

Here S_L, S_R are the wave speeds of two regions that separate star region denoted in figure 4.4. Q_L, Q_R are left, right fluxes and Q_{hll} is the HLL intercell flux calculated as,

$$Q_{hll} = \frac{S_R F_L - S_L F_R + S_L S_R (Q_R - Q_L))}{S_R - S_L} \tag{4.12}$$

4.3.2 HLLC Riemann solver

By restoring the absent contact and shear waves in the Euler equations, the HLLC scheme ('C' refers to Contact) is a modification of HLL scheme, and it is a three wave model consisting of S_L, S_*, S_R. The solution in the star region consists of two intermediate fluxes Q_{*L}, Q_{*R}. Thereby, the HLLC intercell flux is calculated [68] as,

$$Q_{i+\frac{1}{2}} = \begin{cases} Q_L & 0 \leq S_L \\ Q_{*L} & S_L \leq 0 \leq S_* \\ Q_{*R} & S_* \leq 0 \leq S_R \\ Q_R & 0 \geq S_R \end{cases} \tag{4.13}$$

The Q_{*L}, Q_{*R} fluxes are calculated as,

$$Q_{*L} = Q_L + S_L(U_{*L} - U_L) \tag{4.14}$$
$$Q_{*R} = Q_R + S_R(U_{*R} - U_R) \tag{4.15}$$

Here, the U_{*L}, U_{*R} denotes the two separate regions of star state. From equation 4.15

$$U_{*\omega} = \begin{pmatrix} \rho_{*\omega} \\ \rho_{*\omega} u_{*\omega} \\ \rho_{*\omega} E_{*\omega} \end{pmatrix} = \rho_\omega \left(\frac{S_\omega - u_\omega}{S_\omega - S_*} \right) \begin{pmatrix} 1 \\ S_* \\ (\frac{E_\omega}{\rho_\omega} + (S_* - u_\omega)[S_* + \frac{\rho_\omega}{\rho_\omega (S_\omega - u_\omega)}]) \end{pmatrix} \tag{4.16}$$

Here, $\omega \in L, R$ and for a given star region pressure, velocity u_*, p_*, the wave speeds (S_L, S_R, S_*) are computed as follows,

$$S_L = \begin{cases} u_L - c_L & p_L \geq p_* \\ u_L + \frac{p_L - p_*}{\rho_L (u_L - u_*)} & p_L \leq p_* \end{cases} \tag{4.17}$$

$$S_* = u_* \tag{4.18}$$

$$S_R = \begin{cases} u_L + c_L & p_R \geq p_* \\ u_R + \frac{p_R - p_*}{\rho_L (u_R - u_*)} & p_R \leq p_* \end{cases} \tag{4.19}$$

4.3.3 Pressure oscillations in Supercritical flows

Supercritical fluids have non-linear variation of thermophysical properties near the critical point, the simulations associated with such sort of phase transition from the sub-critical to super-critical, the so called 'Transcritical' simulations always experience spurious pressure oscillations that inhibit the ease of modelling the fluid flow in these thermodynamic conditions. In the past decade there has been a significant improvement in tackling this problem especially from the numerical scheme perspective. Outlined below are some of those approaches.

Double flux method: The double flux method was first proposed by Abgrall[3] for a simple numerical scheme with perfect gas assumption, later this was extended into the real fluid simulations[43],[44]. In the context of these real fluid simulations, the spurious pressure oscillations appear when the specific heat ratio (γ) and internal energy is not constant over the cell, therefore by keeping this specifc heat constant in both space and time during the time integration over each time step and each individual cell we can restrain these oscillations. Also, in the primitive variables have to be updated in each sub-step of the time integration.

Additional transport equations: When simulating reacting compressible flows or multifluid flows, the pressure oscillations are frequent at the contact interfaces, even though with the help of high order accurate Riemann solvers we can capture the shocks and be able to resolve the discontinuities in the case of single phase fluid. Johnsen and Colonius[34] presented a novel way of oscillation suppressing interface-capture methods based on Roe solver reconstruction of the primitive variables (ρ, u, p) so as to preserve the continuity and thereby eliminate the pressure imbalance in each cell. They extended these methods to the higher-order accurate Weighted Essentially Non-Oscillatory (WENO) schemes and HLLC approximate Riemann solver[34]. They also included a stiffened equation of state to model the individual components of the multiphase fluid. Shyue[57] used a quasi-conservative formulation of specific-heat ration (γ) model (conservative, primitive equations that are rewritten in terms of the specific heat) and volume-fraction models (fraction of each liquid, gaseous component in each cell) to model multiphase fluids with a high resolution wave propagation method.

Baer-Nunziato type models: Contrary to the belief that having a strong non-linear variation in thermodynamic properties would lead to pressure oscillations Pantano [47] claims that the convexity of the equation of state is not the source of oscillations but rather the discontinuities that present across various interfaces cause these oscillations and no matter how non-linear the density due to the equation of state, these pressure oscillations are not easy to eliminate not even under the grid refinement. Thus causing a ”violation of mechanical equilibrium”. In their work, using a conservative formulation of the compressible Euler equations, an extra evolution equation[47] is added to the governing equations in order to carry the nonlinearity of the equation of state. *Entropy stable hybrid scheme:* Aiming at the transcritical turbulent flows, this method[44] uses double flux method[3, 44, 43] for the flux reconstruction and adaptively couples the higher-order non-dissipative and lower-order dissipative finite volume schemes with the help of a relative solution sensor, that flags the cells, usually that are characterized by large density gradients. *Non-conservative pressure formulation:* In order to maintain the pressure equilibrium across the fluid interfaces in the case of mixing, Terashima and Koshi[63, 64] proposed a pressure evolution based governing equation that will reconstruct the pressure and velocity across the interfaces instead as in the fully conservative method of evolving the total energy. Their approach[63, 64] also employed consistent numerical diffusion terms as their scheme was based on finite-difference formulation. Matheis and Hickel[45] addressed this oscillations problem in transcritical and supercritical fluid mixing in the case of a high pressure liquid-fuel injection[45]. They replaced the quasi-conservative, total energy based governing equations to non conservative pressure evolution equation[64]. They compared the both formats i.e., quasi-conservative and fully conservative simulations including the effect of diffusion induced pressure variation[45] solving for energy and pressure evolution equations respectively.

Ghost fluid methods: This particular method[20] focus heavily towards the oscillations at the material interfaces, by dividing the computational domain into individual fluids the Ghost fluid method[20] is developed based on a Eulerian scheme that treats the material interfaces in a Lagrangian format they used a level set function[20] to track the material interface along with the ghost cells keeping a constant pressure and velocity across the fluid interfaces as they remain constant irrespective of type of fluid. Thus, the scheme[20] is non-conservative at the material interface.

4.3.4 Freezing the specific heat capacity ratio (γ)

The original HLLC approximate Riemann solvers is formulated based on a constant specific heat ratio (γ), as we know that the thermodynamic properties vary rapidly as we approach the near critical temperature, pressure making the perfect gas assumption invalid. Also, there is a fundamental assumption[68] that the wave speed estimates that are based on speed of sound are valid for a given equation of state. Hence, the equation of state would only be needed to compute the speed of sound thereby the wave speeds (both left & right). In order to incorporate the real fluid effects into the HLLC solver, there are two solutions, one[4] is carrying thermodynamic index γ_e using an evolution equation and another is to carry ρe which is calculated at the interface states in addition to the primitive variables ρ, u, p. Although this approach overfits the thermodynamics, it prevents the need to calculate γ_e as described in the first method. In the current project, the Colella and Glaz method[18] is used and the effective thermodynamic index is written as,

$$\gamma_e = \frac{p}{\rho e} + 1 \tag{4.20}$$

And then the characteristics for the Euler equations are solved where the speed of sound is calculate using the above mentioned thermodynamic index in perfect gas assumption for a given cell. A 1D advection simulation is setup to validate the significance of this alternative numerical technique, Note that the Euler system is solved for all test cases in this subsection, which enables a direct comparison with exact analytical solutions. The test case involves nitrogen as the working fluid. The pressure is set to 5 MPa, which is above the critical pressure of nitrogen. The computational domain is x ∈ [0, 1] m and a uniform mesh with 512 grid points is used. Periodic boundary conditions are applied. The initial conditions involve a sharp jump of density, given as

$$\rho = \begin{cases} 800 & 0.25 \le x < 0.75 \\ 144.3, & otherwise \end{cases} \tag{4.21}$$

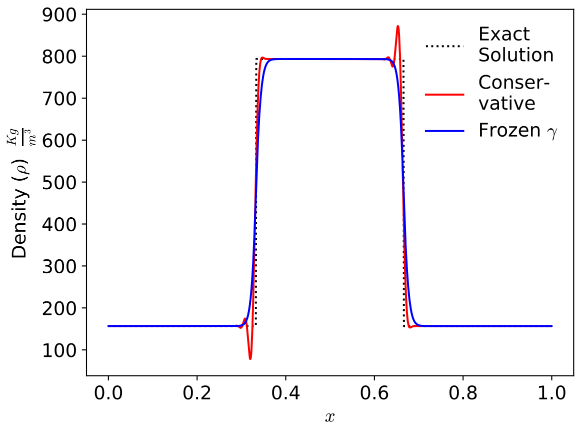


Figure 4.6: Density plot for the 1D real fluid Nitrogen advection case with initial condition as a sharp jump. Mesh size = 512.

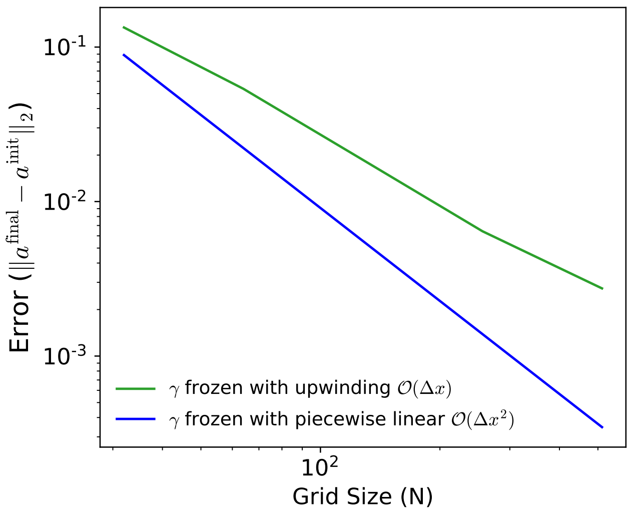


Figure 4.7: Convergence plot for the 1D real fluid Nitrogen Advection case with initial condition as a sharp jump. Mesh size = 512.

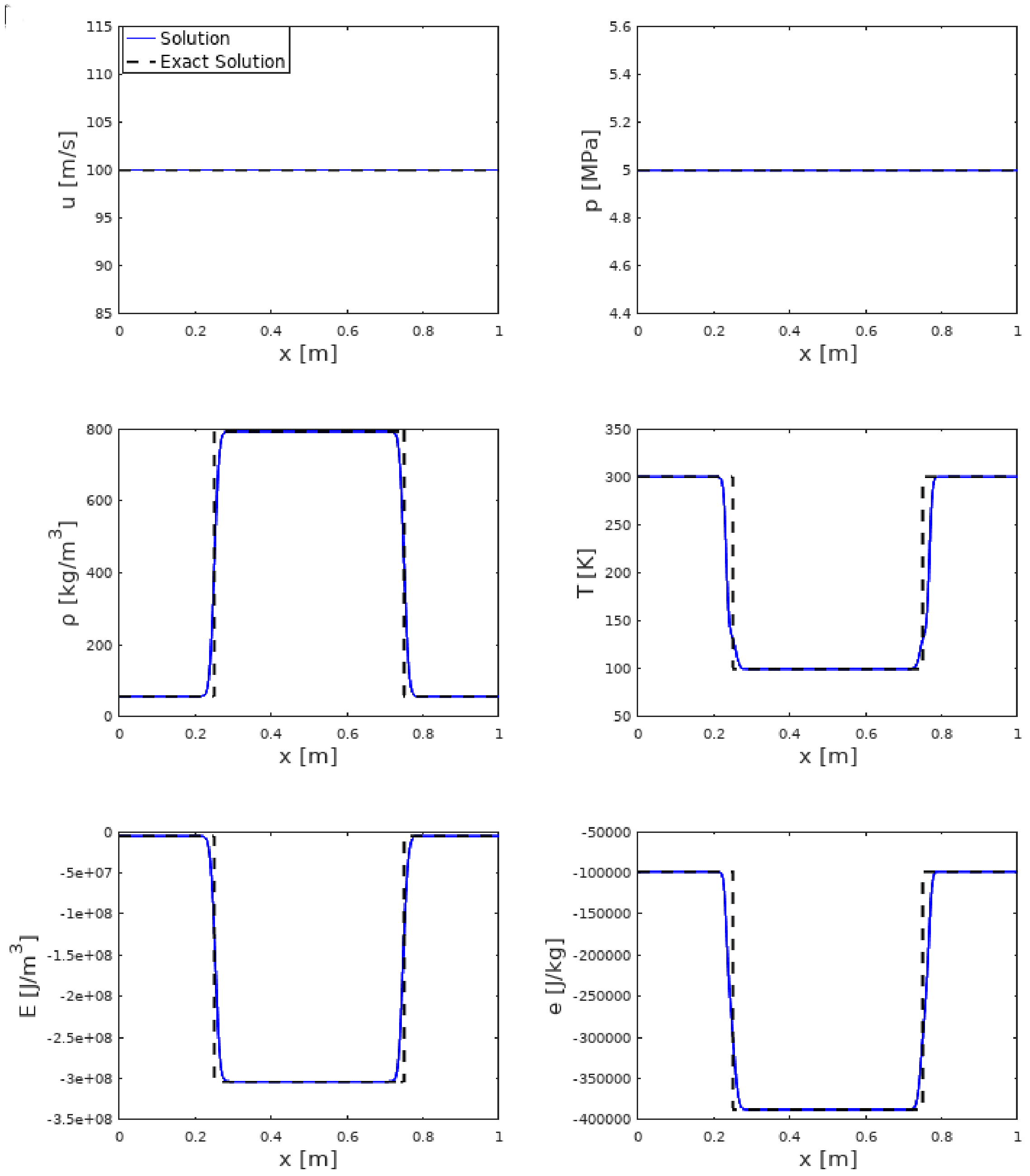


Figure 4.8: Plots of Density, velocity, temperature, total, internal energy for the 1D real fluid Nitrogen advection case with initial condition as a sharp jump. Mesh size = 512.

4.4 Fourth-order Runge-Kutta scheme (RK4)

The time advancement method is followed from [44], and implemented as described below.

- Compute the frozen γ, total energy across all the cells in mesh and store it to keep it as a constant over the entire timestep, across each cell.
- Starting with a defined conservative state, evaluate a single substep by reconstructing the fluxes and solving the Riemann problem with HLLC solver (adapted to real fluid thermodynamics by carrying the pre-computed gamma, total energy in above step).
- The RK substages are evaluated (k_1, k_2, k_3, k_4) during which the conservative variables are updated and after all stages of time integration are completed, a final update where all the substages are summed i.e., $(k_1 + 2k_2 + 2k_3 + k_4/6)$ and y_{t+1} is computed, primitive variables are updated from the conservative variables.

Here's the code snippet, that implements the above mentioned steps.

```python
    sos = myd.get_var("soundspeed")
    rho = myd.get_var("density")
    pres = myd.get_var("pressure")
    gammaf = (sos**2)*rho/pres
    e = eos.rhoe(rho, pres)/rho
    e = e - (pres/(rho*(gammaf - 1.0)))
    for s in range(4):
        ytmp = rk.get_stage_start(s)
        ytmp.fill_BC_all()
        k = self.substep(ytmp, gammaf, eOS)
        self.cons_to_prim()
        rk.store_increment(s, k)
    rk.compute_final_update()
```

Listing 4.1: RK4 with frozen specific heat ratio

4.5 Integration with tabulated thermodynamic equation of state

As we discussed in section 2.1 the governing equations are closed by an EOS (a function that maps the thermodynamic variables (rho, e) $\rightarrow$ (P, T) or vice versa). Here, in the current CFD solver, EOS is used in context of transforming conservative variables (ρ, e) that are obtained by solving the flux vectors of conserved quantities (ρ, $\rho\vec{\mathbf{u}}$, ρE) to primitive variables (P, T), which are necessary to calculate the flux. The code is implemented with both CoolProp, tabular EOS, for the latter the table generation and lookup process is described in chapter 2. It covers the super-critical, near-critical regime except the thermally equilibrium wet-region. In this thesis, for the transformation of conservative (ρ, e) $\rightarrow$ primitive (P, T) and vice versa, we proceed with following method.

- During the solution initialization, to initialize conserved variables (ρ, e) based on (P, T) we use a multi-variate Newton-Raphson method with initial guess values (E.g; T = 300 K, P = 101325 Pa) to iterate and obtain corresponding density, internal energy values. This method is robust and efficient. Since, we run this iterative method only during the initialization, there is not much of an effort in terms of computation. Although we can use another table to lookup, i.e., (ρ, e) tabulated in (P, T), storing it in cache memory during runtime would make it ill-efficient.
- As we know that for a typical conservative finite volume method of Godunov type, once we set the conservative variables based on initial conditions at time, t= 0, we need not to transform (P, T) $\rightarrow$ (ρ, e) at any point later on in the simulation, since the (ρ, e) pair is directly evolved from the numerical method thereby always known. So, we directly lookup in the pre-loaded EOS tabulated in (ρ, e)

Although this method is robust and accurate, a relatively simpler approach is to use thermodynamic database (CoolProp NIST) functions directly. But, this can cause severe bottleneck when executed thousands of times during the higher order schemes. It is important to mention here that the previously mentioned inversion i.e., (ρ, e) $\rightarrow$ (P, T) using multi-variate Newton method is not valid in the wet-steam region due to the fact that the pressure and temperature become dependent variables and coupled by a relation. $p = p_{saturation}(T)$, here $p_{saturation}(T)$ is saturation pressure at a given temperature T.

Chapter 5

Evaluation of tabulated supercritical thermodynamics

5.1 Real fluid thermodynamics significance in shock tube flow at supercritical conditions

Shock tube problems are of great use when it comes to validating numerical codes/schemes for various characteristics like shock capturing, dissipation, preventing oscillations etc. A fluid is said to be in a supercritical state, if its thermodynamic temperature and pressure are above its critical point. Since it is difficult to use experiments to validate the supercritical flow behaviour, often times simulations are widely used. Especially in the cases of supercritical fuel injection in diesel engines, mixing and injection of cryogenic propellants in rocket engines, or organic Rankine cycle turbines. The present chapter aims at understanding the influence of supercritical, near-critical, subcritical thermodynamic conditions on the shock tube problem and more specifically the shock wave, expansion fan locations in each case. It is well known that supercritical fluids have special thermodynamic properties by having a liquid like high density and gas like high diffusivity. In the proximity of critical point, specific heat increases drastically as shown in the figure 1.2 which causes a large expansion even for small increase in temperature. A perfect gas law cannot capture such non-linear variation thereby an accurate calculation of thermodynamic properties is required.There have been several attempts at studying shock tube flows in supercritical and nearcritical conditions. Near the critical point, the compression due to shock introduces a phase transition causing subsequent thermo-physical property changes resulting in an occurrence of non-classical waves [6]. The application of accurate equation of states to the gas dynamics simulation revealed the formation of non-classical supersonic wave[27]. Arina [7] developed a numerical method for investigating supercritical fluid flow in nozzles, shock tube using cubic equation of states. Terashima[63],[64] presented a high resolution numerical method for simulating supercritical flows with large density variations and was able to resolve the flow interfaces over a wide range of scales. There have been several extensions of Riemann solvers to the real fluid thermodynamic conditions, specifically for van der Waals EOS [47],[6], stiff-gas EOS[4], cubic EOS[51].

5.1.1 Fundamental derivative of gas dynamics

In the process of understanding the key implications of using a real fluid thermodynamic equation of state for gas dynamics problems, we need to discuss about an important variable known as the "fundamental derivative of gas dynamics"[65, 46] which governs the non-linear dynamics of compressible fluids. It relates the change in speed of sound with reference to the change in density [65] at a given constant entropy, it is mathematically expressed as,

$$\tau(\rho, c) \equiv 1 + \frac{\rho}{c}\left(\frac{\partial c}{\partial \rho}\right)_s = \frac{\nu^3}{2c^2}\left(\frac{\partial^2 P}{\partial \nu^2}\right)_s \tag{5.1}$$

where, ρ is the density, c is the speed of sound, $\nu = \frac{1}{\rho}$ is the specific volume and P is the pressure.

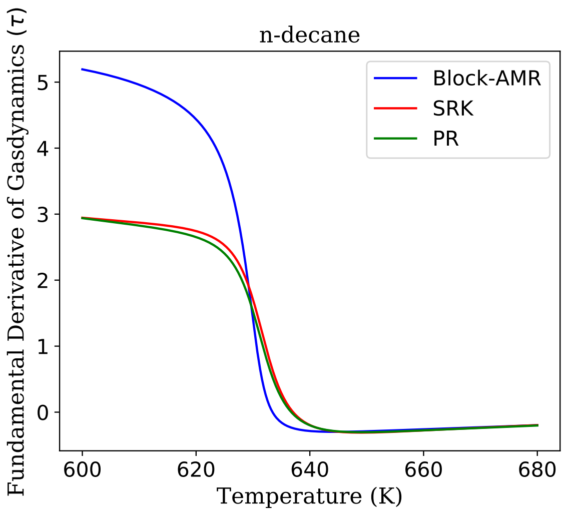


Figure 5.1: Fundamental derivative of gas dynamics evaluated using different equations of state for n-decane ($C_{10}H_{22}$)

This property denotes the direction in which the non-linearity propagates, when $\tau > 0$ the non-linearity propagates in a conventional direction i.e., disturbances travel forward leading to compression shocks and when $\tau < 0$, the disturbances travel backward leading to rarefactions [65] and it is also been suggested [38] that this is required in order to satisfy the second law of thermodynamics. The fluids that fall into this category of negative non-linearity[46] are mainly comprised of dense gases, Bethe-Zel'dovich-Thompson fluids[38], complex organic compounds and the term "dense" here refers to the region close to its thermodynamic critical point or liquid-vapour curve.

τ	Change in $\left(\frac{\partial c}{\partial \rho}\right)$	Behaviour	Remark
$\tau > 1$	$\left(\frac{\partial c}{\partial \rho}\right) > 0$	Speed of sound increases with pressure	Ideal
$\tau = 1$	$\left(\frac{\partial c}{\partial \rho}\right) = 0$	Speed of sound is constant and pressure varies linearly with density (ρ)	
$0 < \tau < 1$	$\frac{\rho}{c} < \left(\frac{\partial c}{\partial \rho}\right) < 0$	Speed of sound decreases with pressure	Real
$\tau < 0$	$\left(\frac{\partial c}{\partial \rho}\right) < \frac{\rho}{c}$	Unusual	non-classical[40]

Table 5.1: Description of fundamental derivative of gas dynamics behaviour for different values [40, 65]

If we use a perfect gas EOS to compute this fundamental derivative (τ) where $c = \sqrt{\gamma RT}$ we obtain, $\tau = (\gamma + 1)/2$, which is a constant for a given specific heat ratio (γ). Which means, there is no chance for us to observe negative fundamental derivative ($\tau < 0$) for any given fluid with perfect gas EOS assumption. This warrants the use of multiparametric EOS for such evaluations however as we discussed in chapter 2 with regard to the error associated with cubic EOS, using such equations to compute speed of sound and density would lead to further amplification of erroneous values. As can be seen in the figure 5.1 there is a significant difference in the fundamental derivative (τ) values computed using cubic EOS and tabular EOS

Before simulating such non-classical gas dynamics problems with real fluid EOS we try to obtain a validation for the CFD solver with the help of established testcases that employ a higher order accurate spatial and temporal schemes with real fluid EOS. The next two sections discuss the validation cases in two such test cases namely; Sod shock tube and Shu-Osher shock tube problems.

5.1.2 Sod Shock Tube Real gas validation case

Previously Terashima[64] carried out a 1D shock tube simulation under supercritical thermodynamic conditions, for this problem carbon dioxide is considered as the working fluid whose critical pressure, temperature and density [?] are $T_{crit} = 304.22K$, $p_{crit} = 7.37$ MPa, $\rho_{crit} = 348.8\ \frac{kg}{m^3}$.

The initial conditions for this Sod shocktube simulation are

$$(\rho, u, p) = \begin{cases} (\rho_0, 0.0, 10p_0) & 0 \leq x \leq 0.5 \text{ m}, \\ (0.01\rho_0, 0.0, 0.1p_0) & 0.5 \leq x \leq 1.0 \text{ m} \end{cases}$$

Here, $\rho_0 = \rho_{crit} = 348.8\frac{Kg}{m^3}$, and $p_0 = p_{crit} = 7.37MPa$, the grid consists of 2001 x 4 uniformly distributed cells.

The below plot is taken at t = 2.745E-04 s. There is a good agreement in between the present solver and the reference simulation data obtained from the paper[63], no spurious pressure oscillations are found mainly due to the fact that the initial conditions are in supercritical regime. However, the transcritical case showed significant pressure oscillations.

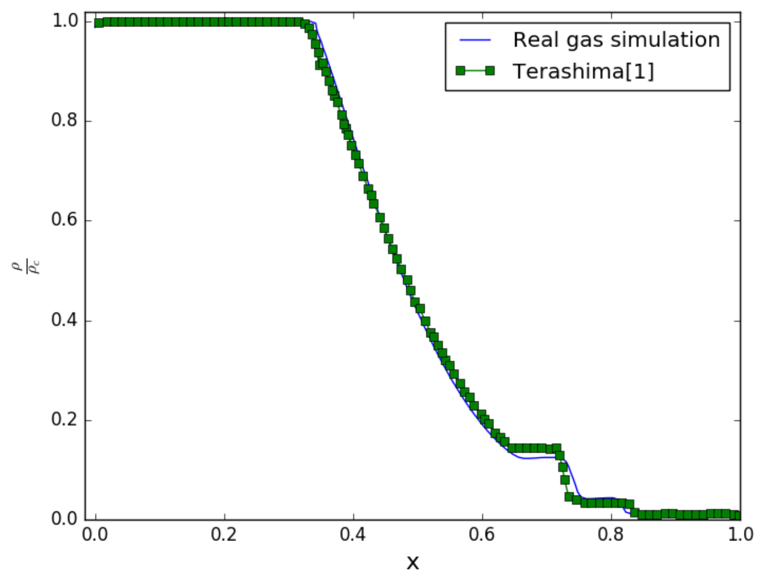
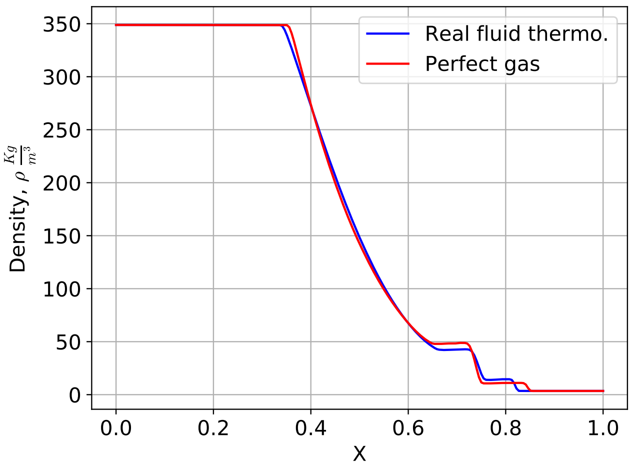


Figure 5.2: Sod Shock tube problem with real fluid thermodynamics Terashima validation case[63], Normalized Density $\left(\frac{\rho}{\rho_{critical}}\right)$ vs distance plot taken at $t = 2.745E-04$ s, CFL = 0.4, Mesh size = 201, working fluid = Carbon dioxide.



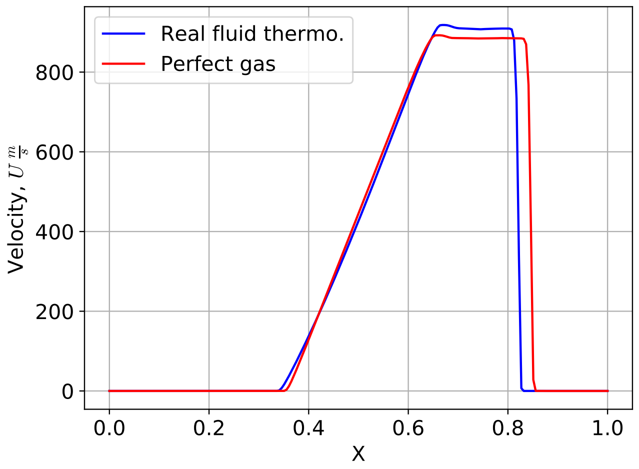


Figure 5.3: Sod Shock tube with real fluid thermodynamics compared with Ideal gas law

From the above figure on density distribution plot comparing real, ideal thermodynamics, we can see tat the real gas case predicts the faster expansion fan when compared to the perfect gas case's expansion fan.

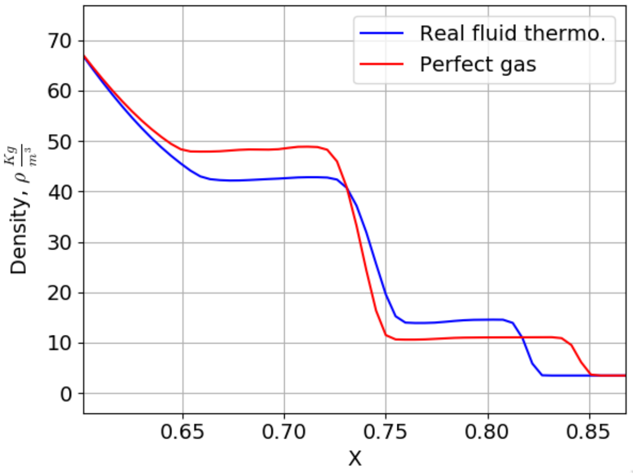


Figure 5.4: Sod Shock tube with real fluid thermodynamics compared with Ideal gas law

5.1.3 Shu-Osher shock tube real gas validation case

Another case used for reference is the Shu-Osher problem[64] that is modified to the supercritical regime (initial conditions). The purpose of testing with this case is to validate the code's dissipation characteristics. This test case is taken from Terashima[64], nitrogen is used as working fluid for this case and the initial conditions are,

$$(\rho, u, p) = \begin{cases} (3.857\rho_0, M_{ref}, 10.333p_0) & -0.5 \leq x \leq -0.4 \text{ m}, \\ ((1.0 + 0.2sin5x)\rho_0, 0.0, p_0) & -0.4 \leq x \leq 0.5 \text{ m} \end{cases}$$

Here, $\rho_0 = 50.0\frac{Kg}{m^3}$, $p_0 = 4.0MPa$ and M_{ref} is the reference sound speed on the left side. Similarly as previous case, the grid consists of 2001 uniformly distributed cells. The below plot is taken at 5.903E-04 s.

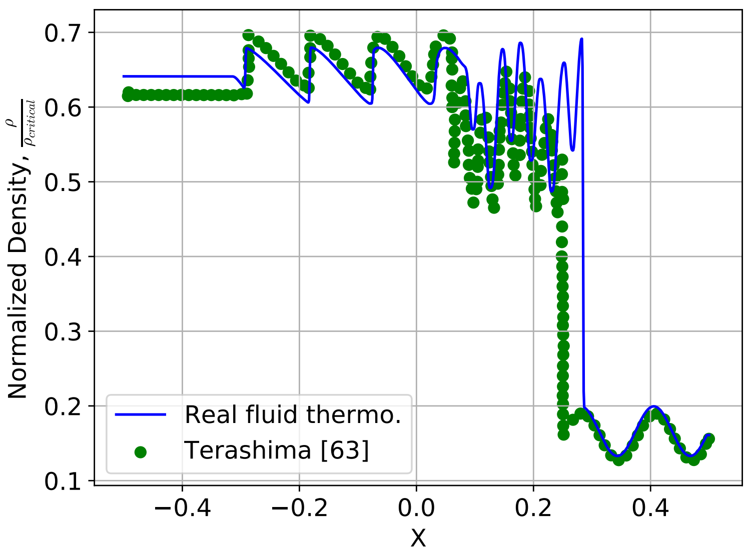


Figure 5.5: Shu-Osher shocktube problem with real fluid thermodynamics validation case[63], Normalized density $(\frac{\rho}{\rho_{critical}})$ vs distance (x) plot

Figure 5.5 shows the normalized density $(\frac{\rho}{\rho_{crit}})$ plot for the current real fluid modified solver vs the reference case. It is important to mention here that the reference case was obtained by smoothening the initial interface with an adjustable free parameter $c_\epsilon = 3.0$. This could be a reason for slight mismatch in the density profiles.

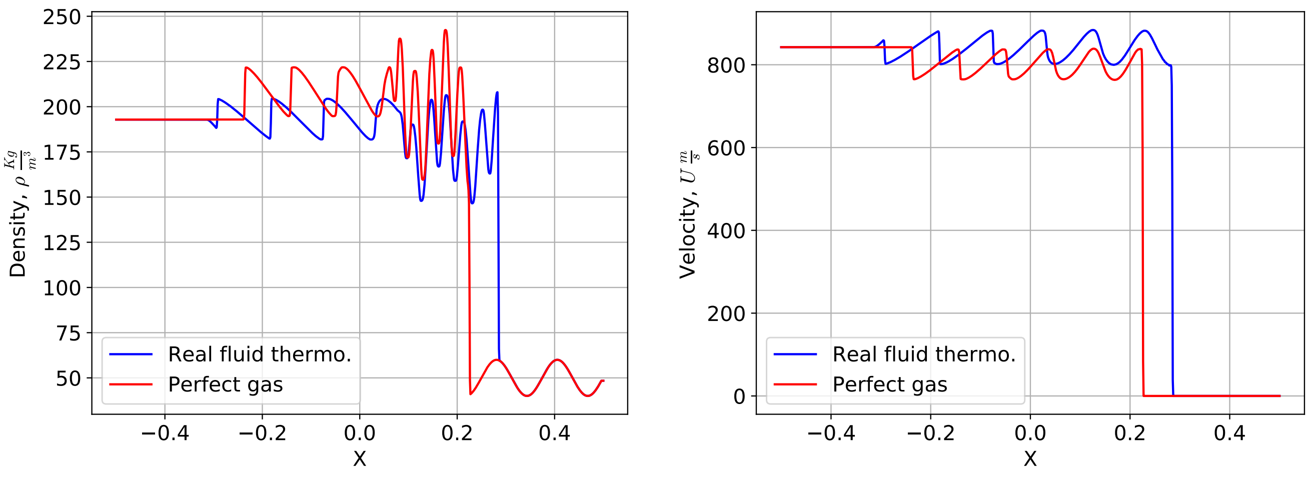


Figure 5.6: Shu-Osher shock tube problem comparison between real fluid thermodynamics and perfect gas assumption, density (ρ) , velocity plots vs distance. Mesh size (N) = 201, working fluid is Nitrogen.

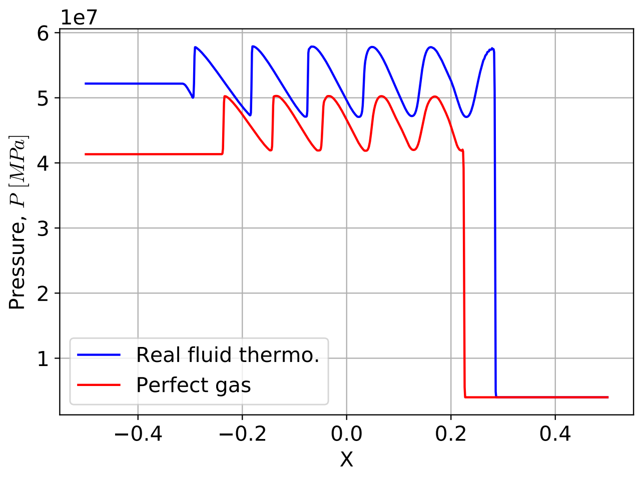


Figure 5.7: Shu-Osher shock tube problem comparison between real fluid thermodynamics and perfect gas assumption, Pressure (P) , vs distance plot. Mesh size (N) = 201, working fluid is Nitrogen.

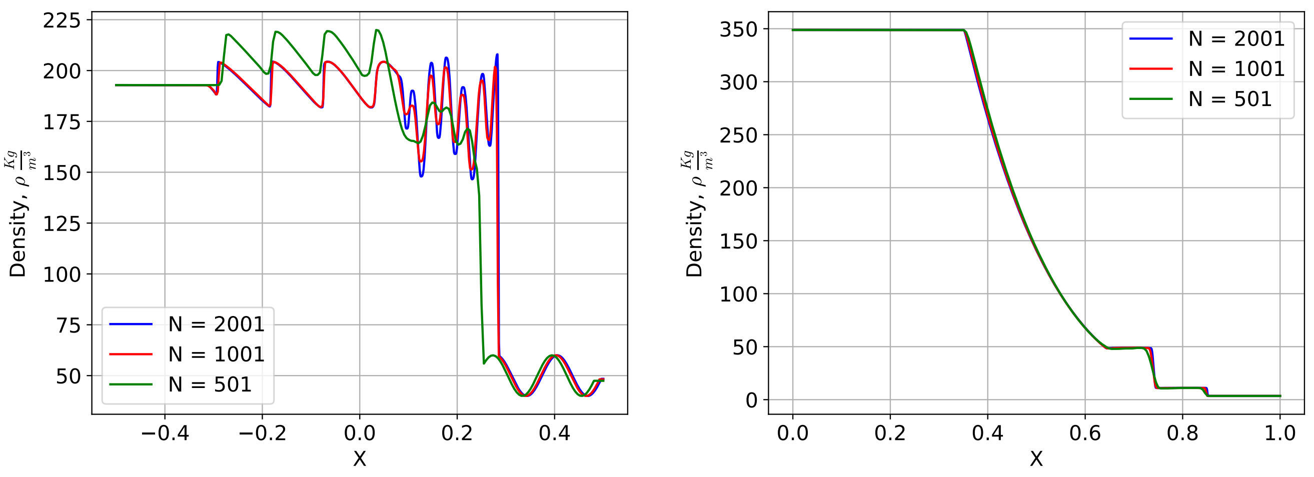


Figure 5.8: Mesh Independence density profile plot of Shu-Osher shock tube problem with real fluid thermodynamics, taken at $t = 2.745E - 04$ s,CFL = 0.4 Mesh sizes (N) = 501, 1001, 2001,

5.1.4 Accurate real-fluid thermodynamics significance in shock tube problem

In order to study the real-fluid effects on shock tube wave propagation, three cases are simulated at supercritical, near-critical and the subcritical thermodynamic conditions. The initial conditions are shown in the table 5.2 for each case, in all the cases the left to right pressure ratios ($\frac{p_{left}}{p_{right}}$) are set to 2 with an initial temperature (T) of 310 K. Figure 5.10

Table 5.2: Initial conditions for Sod-problem.

	P$_{left}$	P$_{right}$	P$_{left}/P_{right}$	T, K	ρ_{left}	ρ_{right}
Supercritical	4p$_{cr}$	2p$_{cr}$	2	310	845	696
Nearcritical	1.5p$_{cr}$	0.75p$_{cr}$	2	310	610	134
Subcritical	0.5p$_{cr}$	0.25p$_{cr}$	2	310	76.9	34.4

shows the pressure, temperature, density distribution plots, in which the variables (ρ, P, T) are non-dimensionalized with respect to the initial values on left side. For the case of nearcritical, comparatively smaller pressure, temperature jump ($\frac{p}{p_{left}}$) is observed at the contact interface, while the density jump is largest. This can be attributed to the strong non-linear variation of thermodynamic properties in the proximity of critical point.

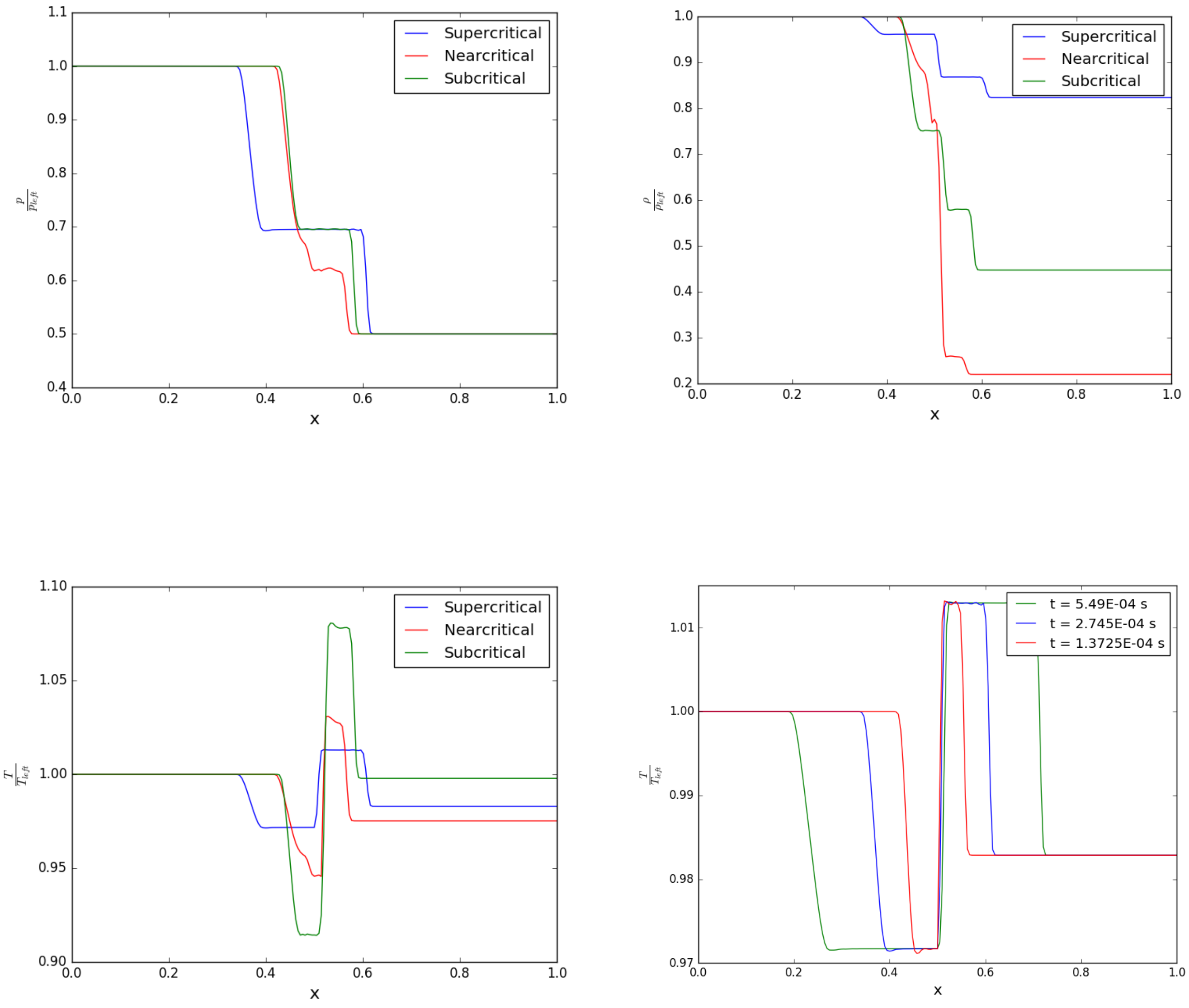


Figure 5.10: Plots of non-dimensionalized pressure, density, temperature and time histories of temperature profile for the near critical case . Here the variables are non-dimensionalized using the initial left side values.

A small temperature overshoot can be observed next to the contact discontinuity. We have discussed in section 4.3.4 on various numerical techniques to overcome this oscillations.

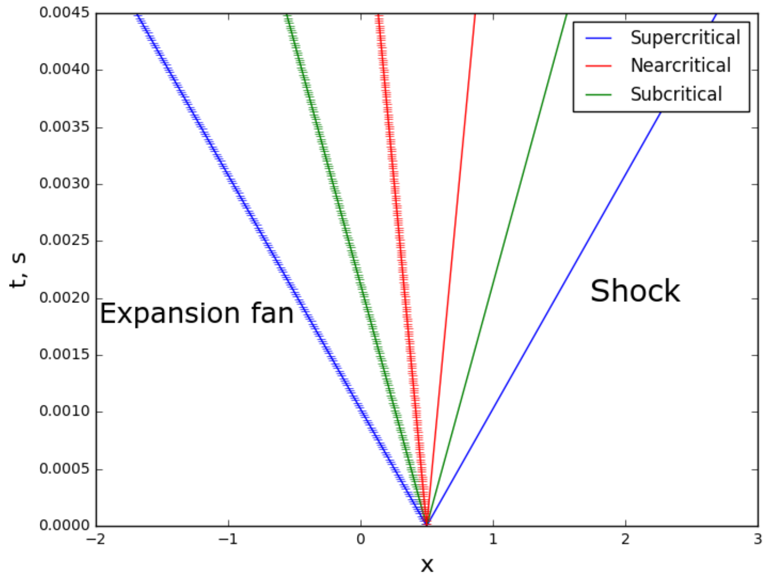


Figure 5.11: Sod shocktube problem with three different real fluid thermodynamics cases, plot of time histories of shock wave (—) , expansion fan (+++) locations.

Figure 5.11 showcases the shows time histories of the wave positions, in the real fluid supercritical case, both the shock and the expansion fan propagate faster than in the ideal gas assumption. Though, we couldn't verify whether this is purely due to the calculation mismatch in speed of sound or there is really a certain physical significance.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Paving way towards accurate real-fluid CFD simulations, this thesis provides improvised method in a key area, real fluid thermodynamic property evaluation using an adaptive tabulation, given the fact that the CFD simulations with complex thermodynamic systems involve expensive simulations due to the non-linear variation of thermodynamic properties and numerical scheme to model large density gradients. CFD solvers require EOS to transform the conservative variables to their primitive state. Typically in case of cubic EOS such as Peng-Robinson this is accomplished through an iterative algorithm to find the correct (P, T) primitive pair for given conservative pair (ρ, e). The emphasis in obtaining such accurate thermodynamic information is due to the strong non-linear thermodynamic property gradients observed in the vicinity of critical point i.e., any minute changes in temperature or pressure drastically changes the density. As an alternative to such iterative algorithm, this thesis presented a detailed framework on Tabulated real fluid thermodynamics, starting from the simplest form of tabulation i.e., homogeneous or logarithmic tabulation to various other forms of tabulation such as Block structured adaptive mesh refinement AMR and Cubic Bezier surface based adaptive mesh refinement AMR are described and implemented in detail. When compared to the homogeneous tabulation, adaptive tabulation gives efficient memory storage according for a given user defined error tolerance, generating a homogeneous EOS table with very low error tolerances generates large sized storage files thereby limits the physical and thermodynamic range within which the table is going to support. When the tabular EOS is compared with the cubic equations in terms of accuracy, evaluation (look-up) speed there was a significant improvement. To observe the speedup, the tabular EOS is implemented into to a 1D CFD code and the grid size is scaled incrementally. The cubic EOS based simulations increased the computational cost non-linearly with respect to the grid size. To further investigate the significance of using real fluid thermodynamics in a gas dynamics simulation, a CFD code with higher order spatial and temporal schemes is developed and its numerical schemes were discussed in chapter 4 In order to overcome the oscillations at the sharp interfaces, a method in which the specific heat ratio is kept constant over a time step across each cell is discussed and implemented. It helped in damping the oscillations and enabled to obtain stable simulations. Later the process through which the adaptive tabular EOS is integrated into this CFD code is described. The relevance of computing accurate thermodynamic properties in gasdynamics problems is described in chapter 5 coupling both the tabulated thermodynamic equation of state framework and the CFD solver together, two test cases are used for this study, sod-shock tube and the shu-osher shock tube. A validation result is obtained by simulating a near critical testcase taken from Terashima [64]. Three different thermodynamic initial conditions namely subcritical, supercritical and near critical regions are simulated. It was found that the accurate thermodynamics evaluation resulted in a faster expansion fan compared to the perfect gas assumption. In the test case where three different thermodynamic conditions where tested, among them the nearcritical case had a significantly smaller temperature, pressure jumps, whereas the density jump was the largest among all the three cases. Also, the expansion fan was faster while using real fluid thermodynamics than perfect gas assumption.

6.2 Future Work

There are two aspects to which the current thesis can be extended, in terms of tabular EOS in present work there is no consideration of the region below the vapour dome (wet-steam region). Also, the present tabulation is limited to single phase, pseudo-pure fluids. Most of the industrial applications involve the use of mixtures, hence this current two dimensional tabulation can be extended into three dimensions, where the third dimension can be mixture fraction.

The second aspect is the uncertainty quantification, as with every EOS there will be an uncertainty associated within the output values, in the case of tabular EOS apart from a parametric uncertainty derived from the underlying empirical EOS there will be an additional model uncertainty due to the interpolation, tabulation. Understanding and quantifying this tabulation uncertainty is key to have a reliable simulation. In terms of test cases, the tabular EOS can be tested with Large Eddy Simulation (LES) softwares,

References

[1] Phase diagram. Matthieumarechal[CCBY-SA3.0(http://creativecommons.org/licenses/by-sa/3.0/)]. Accessed: 2018-08-23.

[2] TwoDThermoCode git repository. https://git.uwaterloo.ca/spmuppar/TwoDThermoCode/network/master. Accessed: 2018-10-31.

[3] Rémi Abgrall and Smadar Karni. Computations of compressible multifluids. *Journal of Computational Physics*, 169(2):594–623, 2001.

[4] AS Almgren, VE Beckner, JB Bell, MS Day, LH Howell, CC Joggerst, MJ Lijewski, A Nonaka, M Singer, and M Zingale. Castro: A new compressible astrophysical solver. i. hydrodynamics and self-gravity. *The Astrophysical Journal*, 715(2):1221, 2010.

[5] George Anitescu, Thomas J Bruno, and Lawrence L Tavlarides. Dieseline for supercritical injection and combustion in compression-ignition engines: volatility, phase transitions, spray/jet structure, and thermal stability. *Energy & Fuels*, 26(10):6247–6258, 2012.

[6] BM Argrow. Computational analysis of dense gas shock tube flow. *Shock Waves*, 6(4):241–248, 1996.

[7] Renzo Arina. Numerical simulation of near-critical fluids. *Applied Numerical Mathematics*, 51(4):409–426, 2004.

[8] Daniel T Banuti, Volker Hannemann, Klaus Hannemann, and Bernhard Weigand. An efficient multi-fluid-mixing model for real gas reacting flows in liquid propellant rocket engines. *Combustion and Flame*, 168:98–112, 2016.

[9] Ian H Bell, Jorrit Wronski, Sylvain Quoilin, and Vincent Lemort. Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical

property library coolprop. *Industrial & Engineering Chemistry Research*, 53(6):2498–2508, 2014.

[10] E Bender. An equation of state for predicting vapour-liquid equilibria of the system N2-Ar-O2. *Cryogenics*, 13(1):11–18, 1973.

[11] Manson Benedict, George B Webb, and Louis C Rubin. An empirical equation for thermodynamic properties of light hydrocarbons and their mixtures i. methane, ethane, propane and n-butane. *The Journal of Chemical Physics*, 8(4):334–345, 1940.

[12] Bertrand Berche, Malte Henkel, and Ralph Kenna. Critical phenomena: 150 years since cagniard de la tour. *Revista Brasileira de Ensino de Física*, 31(2):2602–1, 2009.

[13] Francesco Bonelli, Annarita Viggiano, and Vinicio Magi. A numerical analysis of hydrogen underexpanded jets under real gas assumption. *Journal of Fluids Engineering*, 135(12):121101, 2013.

[14] Thierry Buffard, Thierry Gallouët, and Jean-Marc Hérard. A sequel to a rough godunov scheme: application to real gases. *Computers & Fluids*, 29(7):813–847, 2000.

[15] John H Carpenter. Adaptive tabulation for verified equations of state. In *AIP Conference Proceedings*, volume 1426, pages 808–811. AIP, 2012.

[16] Ting Horng Chung, Mohammad Ajlan, Lloyd L Lee, and Kenneth E Starling. Generalized multiparameter correlation for nonpolar and polar fluid transport properties. *Industrial & Engineering Chemistry Research*, 27(4):671–679, 1988.

[17] Ting Horng Chung, Lloyd L Lee, and Kenneth E Starling. Applications of kinetic gas theories and multiparameter correlation for prediction of dilute gas viscosity and thermal conductivity. *Industrial & Engineering Chemistry Fundamentals*, 23(1):8–13, 1984.

[18] Phillip Colella and Harland M Glaz. Efficient solution algorithms for the riemann problem for real gases. *Journal of Computational Physics*, 59(2):264–289, 1985.

[19] Eric M Collins and Edward A Luke. Fast evaluation of complex equations of state. *Electronic Journal of Differential Equations*, 2013(20):27–37, 2013.

[20] Ronald P Fedkiw, Tariq Aslam, Barry Merriman, and Stanley Osher. A non-oscillatory eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of Computational Physics*, 152(2):457–492, 1999.

[21] Charles L Fefferman. Existence and smoothness of the navier-stokes equation. *The Millennium Prize Problems*, 57:67, 2006.

[22] Theodore E Fessler. Fluid: A numerical interpolation procedure for obtaining thermodynamic and transport properties of fluids. 1977.

[23] Sergei Konstantinovich Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Matematicheskii Sbornik*, 89(3):271–306, 1959.

[24] Anthony RH Goodwin, Jan V Sengers, and Cor J Peters. *Applied thermodynamics of fluids*. Royal Society of Chemistry, 2010.

[25] Kalyana C Gottiparthi, Ramanan Sankaran, Anthony M Ruiz, Guilhem Lacaze, and Joseph C Oefelein. Large eddy simulation of a supercritical fuel jet in cross flow using gpu-acceleration. *54th AIAA Aerospace Sciences Meeting*, page 1939, 2016.

[26] Sigal Gottlieb and Chi-Wang Shu. Total variation diminishing runge-kutta schemes. *Mathematics of Computation of the American Mathematical Society*, 67(221):73–85, 1998.

[27] A Guardone, L Vigevano, and BM Argrow. Assessment of thermodynamic models for dense gas dynamics. *Physics of Fluids*, 16(11):3878–3887, 2004.

[28] A Hamzehloo and PG Aleiferis. Large eddy simulation of highly turbulent underexpanded hydrogen and methane jets for gaseous-fuelled internal combustion engines. *International Journal of Hydrogen Energy*, 39(36):21275–21296, 2014.

[29] Amiram Harten, Peter D Lax, and Bram van Leer. On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM review*, 25(1):35–61, 1983.

[30] F Hempert, S Boblest, T Ertl, F Sadlo, P Offenhäuser, CW Glass, M Hoffmann, A Beck, C-D Munz, and U Iben. Simulation of real gas effects in supersonic methane jets using a tabulated equation of state with a discontinuous galerkin spectral element method. *Computers & Fluids*, 145:167–179, 2017.

[31] Mortada Mohammad Yao Matthew Hickey Jean-Pierre, Devaud Cecile. Locally-adaptive tabulation of low-dimensional manifolds using bezier patch reconstruction. *Annual Spring Meet, Combustion Institute Canadian Section*, 2017.

[32] Takashi Ishihara, Toshiyuki Gotoh, and Yukio Kaneda. Study of high–reynolds number isotropic turbulence by direct numerical simulation. *Annual Review of Fluid Mechanics*, 41:165–180, 2009.

[33] Richard T Jacobsen, Richard Byron Stewart, Majid Jahangiri, and SG Penoncello. A new fundamental equation for thermodynamic property correlations. In *Advances in Cryogenic Engineering*, pages 1161–1168. Springer, 1986.

[34] Eric Johnsen and Tim Colonius. Implementation of weno schemes in compressible multicomponent flow problems. *Journal of Computational Physics*, 219(2):715–732, 2006.

[35] WC Johnson and PW Voorhees. Coherent phase diagrams. *Bulletin of Alloy Phase Diagrams*, 9(3):208–215, 1988.

[36] R Khaksarfard, MR Kameshki, and Marius Paraschivoiu. Numerical simulation of high pressure release and dispersion of hydrogen into air with real gas model. *Shock Waves*, 20(3):205–216, 2010.

[37] Erdogan Kiran, Pablo G Debenedetti, and Cor J Peters. *Supercritical fluids: fundamentals and applications*, volume 366. Springer Science & Business Media, 2012.

[38] A Kluwick. Non-ideal compressible fluid dynamics: A challenge for theory. In *Journal of Physics: Conference Series*, volume 821, page 012001. IOP Publishing, 2017.

[39] Matthias Kunick, Hans-Joachim Kretzschmar, Francesca di Mare, and Uwe Gampe. Cfd analysis of steam turbines with the iapws standard on the spline-based table look-up method (sbtl) for the fast calculation of real fluid properties. In *ASME Turbo Expo 2015: Turbine Technical Conference and Exposition*, pages V008T26A037–V008T26A037. American Society of Mechanical Engineers, 2015.

[40] Konstantine C Lambrakis and Philip A Thompson. Existence of real fluids with a negative fundamental derivative γ. *The Physics of Fluids*, 15(5):933–935, 1972.

[41] Eric W Lemmon, Marcia L Huber, and Mark O McLinden. Nist reference fluid thermodynamic and transport properties—refprop. *NIST Standard Reference Database*, 23:v7, 2002.

[42] Zhiqi Liu, Jianhan Liang, and Yu Pan. Efficient thermodynamic properties reconstruction method with adaptive triangular mesh. *Journal of Thermophysics and Heat Transfer*, 29(1):83–89, 2014.

[43] Yu Lv and Matthias Ihme. Discontinuous galerkin method for compressible viscous reacting flow. In *21st AIAA Computational Fluid Dynamics Conference*, page 3067, 2013.

[44] Peter C Ma, Yu Lv, and Matthias Ihme. An entropy-stable hybrid scheme for simulations of transcritical real-fluid flows. *Journal of Computational Physics*, 340:330–357, 2017.

[45] Jan Matheis and Stefan Hickel. Multi-component vapor-liquid equilibrium model for les of high-pressure fuel injection and application to ecn spray a. *International Journal of Multiphase Flow*, 99:294–311, 2018.

[46] NR Nannan, ALBERTO Guardone, and P Colonna. On the fundamental derivative of gas dynamics in the vapor–liquid critical region of single-component typical fluids. *Fluid Phase Equilibria*, 337:259–273, 2013.

[47] Carlos Pantano, Richard Saurel, and T Schmitt. An oscillation free shock-capturing method for compressible van der waals supercritical fluid flows. *Journal of Computational Physics*, 335:780–811, 2017.

[48] Hulya Peker, MP Srinivasan, JM Smith, and Ben J McCoy. Caffeine extraction rates from coffee beans with supercritical carbon dioxide. *AIChE Journal*, 38(5):761–770, 1992.

[49] Ding-Yu Peng and Donald B Robinson. A new two-constant equation of state. *Industrial & Engineering Chemistry Fundamentals*, 15(1):59–64, 1976.

[50] M Pini, A Spinelli, G Persico, and S Rebay. Consistent look-up table interpolation method for real-gas flow simulations. *Computers & Fluids*, 107:178–188, 2015.

[51] Marco Pizzarelli. Finite-volume solver of the euler equation for real fluids. *Journal of Computational Physics*, 17:68–91, 2010.

[52] Otto Redlich and Joseph NS Kwong. On the thermodynamics of solutions. v. an equation of state. fugacities of gaseous solutions. *Chemical Reviews*, 44(1):233–244, 1949.

[53] Robert C Reid, John M Prausnitz, and Bruce E Poling. The properties of gases and liquids. 1987.

[54] Jörg-Rüdiger Sack and Jorge Urrutia. *Handbook of Computational Geometry*. Elsevier, 1999.

[55] Stefan Schlamp and Thomas Rösgen. Flow in near-critical fluids induced by shock and expansion waves. *Shock Waves*, 14(1-2):93–101, 2005.

[56] JS Shang. Three decades of accomplishments in computational fluid dynamics. *Progress in Aerospace Sciences*, 40(3):173–197, 2004.

[57] Keh-Ming Shyue. An efficient shock-capturing algorithm for compressible multicomponent problems. *Journal of Computational Physics*, 142(1):208–242, 1998.

[58] Giorgio Soave. Equilibrium constants from a modified redlich-kwong equation of state. *Chemical Engineering Science*, 27(6):1197–1203, 1972.

[59] Roland Span. *Multiparameter equations of state: an accurate source of thermodynamic property data.* Springer Science & Business Media, 2013.

[60] Kenneth E Starling. *Fluid thermodynamic properties for light petroleum systems.* Gulf Pub. Co., 1973.

[61] F Douglas Swesty. Thermodynamically consistent interpolation for equation of state tables. *Journal of Computational Physics*, 127(1):118–127, 1996.

[62] Viacheslav Vladimirovich Sychev, AA Vasserman, AD Kozlov, GA Spiridonov, and VA Tsymarnyi. Thermodynamic properties of oxygen. In *Washington, DC, Hemisphere Publishing Corp.(National Standard Reference Data Service of the USSR. Volume 5), 1987, 322 p. Translation.*, volume 5, 1987.

[63] Hiroshi Terashima, Soshi Kawai, and Nobuhiro Yamanishi. High-resolution numerical method for supercritical flows with large density variations. *AIAA journal*, 49(12):2658–2672, 2011.

[64] Hiroshi Terashima and Mitsuo Koshi. Strategy for simulating supercritical cryogenic jets using high-order schemes. *Computers & Fluids*, 85:39–46, 2013.

[65] Philip A Thompson. A fundamental derivative in gasdynamics. *The Physics of Fluids*, 14(9):1843–1849, 1971.

[66] James Thomson. *A quantitative investigation of certain relations between the gaseous, the liquid, and the solid states of water-substance.* Publisher not identified, 1873.

[67] Eleuterio F Toro. The hll and hllc riemann solvers. In *Riemann Solvers and Numerical Methods for Fluid Dynamics*, pages 315–339. Springer, 1999.

[68] Eleuterio F Toro. *Riemann solvers and numerical methods for fluid dynamics: a practical introduction.* Springer Science & Business Media, 2013.

[69] Eleuterio F Toro, Michael Spruce, and William Speares. Restoration of the contact surface in the hll-riemann solver. *Shock Waves*, 4(1):25–34, 1994.

[70] Jacobo Troncoso, Diego González-Salgado, Claudio A Cerdeiriña, Enrique Carballo, and Luis Romaní. Griffiths-wheeler geometrical picture of critical phenomena: Experimental testing for liquid-liquid critical points. *Physical Review E*, 71(2):021503, 2005.

[71] Johannes Diderik Van der Waals. *Over de Continuiteit van den Gas-en Vloeistoftoestand*, volume 1. Sijthoff, 1873.

[72] Bram Van Leer. Towards the ultimate conservative difference scheme iii. upstream-centered finite-difference schemes for ideal compressible flow. *Journal of Computational Physics*, 23(3):263–275, 1977.

[73] Guoping Xia, Ding Li, and Charles L Merkle. Consistent properties reconstruction on adaptive cartesian meshes for complex fluids computations. *Journal of Computational Physics*, 225(1):1175–1197, 2007.

[74] Michael Zingale. pyro: A teaching code for computational astrophysical hydrodynamics. *Astronomy and Computing*, 6:52–62, 2014.

APPENDICES

Appendix A

A.1 Code repositories

The adaptive tabulation based thermodynamic equation of state code is available in git repository `https://github.com/spmuppar/Adaptive-Tabulated-real-fluid-thermo` and the cfd code developed for this thesis is available at `https://github.com/spmuppar/TwoDThermoCode`

A.2 HLLC Riemann solver adapted for Real fluid thermodynamics module

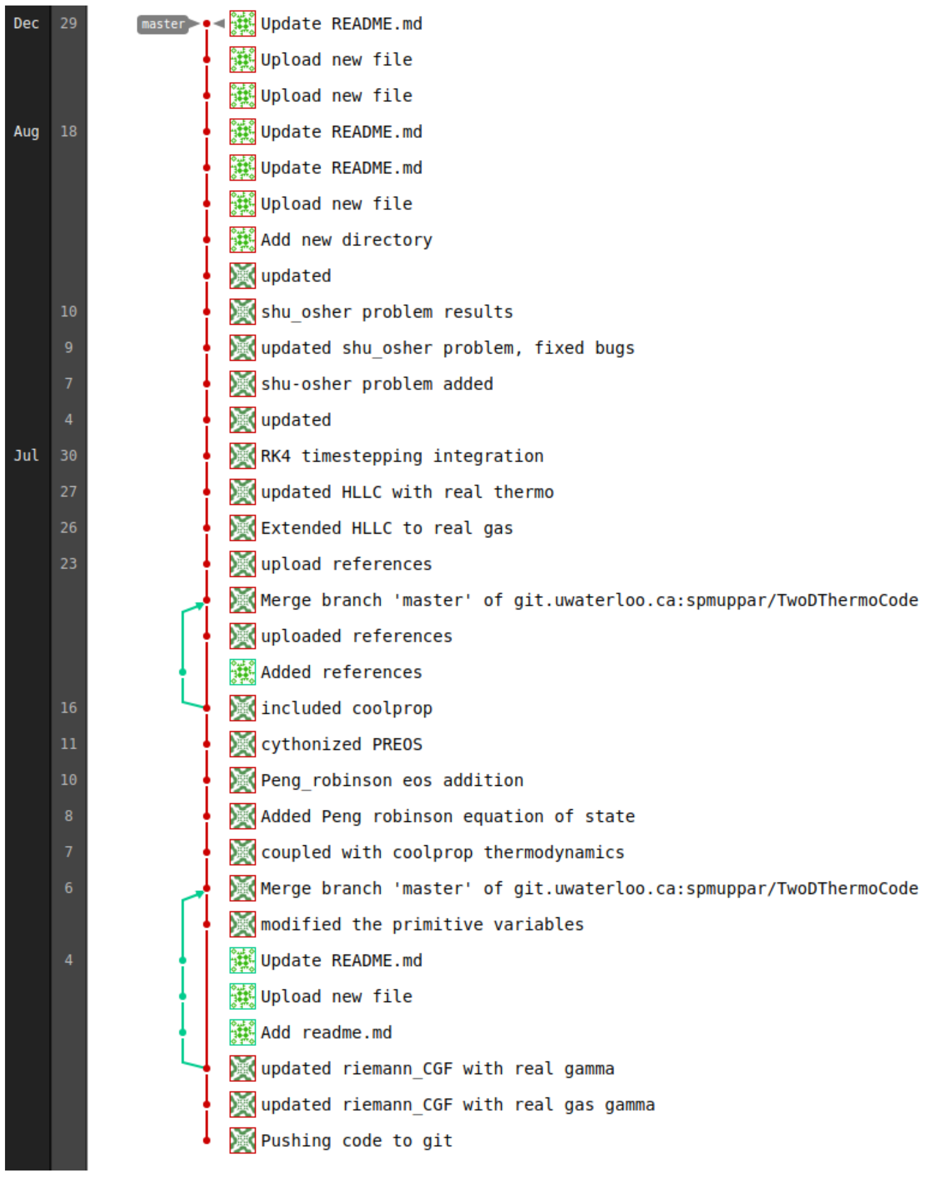
```fortran

SUBROUTINE riemann_HLLC_real(idir, qx, qy, ng, &
                       nvar, idens, ixmom, iymom, iener, &
                       lower_solid, upper_solid, &
                       real_gamma, pres, U_l, U_r, F)

!implicit none
EXTERNAL real_gamma
EXTERNAL pres

integer, intent(in) :: idir
integer, intent(in) :: qx, qy, ng
integer, intent(in) :: nvar, idens, ixmom, iymom, iener
integer, intent(in) :: lower_solid, upper_solid
```

```fortran

double precision, intent(inout) :: U_l(0:qx-1,0:qy-1,0:nvar-1)
double precision, intent(inout) :: U_r(0:qx-1,0:qy-1,0:nvar-1)
double precision, intent(  out) :: F(0:qx-1,0:qy-1,0:nvar-1)
!F2PY (CALLBACK) real_gamma
!F2PY (CALLBACK) pres
!f2py depend(qx, qy, nvar) :: U_l, U_r
!f2py intent(in) :: U_l, U_r
!f2py intent(out) :: F


integer :: ilo, ihi, jlo, jhi
integer :: nx, ny
integer :: i, j

double precision, parameter :: smallc = 1.e-10
double precision, parameter :: smallrho = 1.e-10
double precision, parameter :: smallp = 1.e-10
double precision, dimension(0:1) :: temp
real :: temp1

double precision :: rho_l, un_l, ut_l, rhoe_l, p_l
double precision :: rho_r, un_r, ut_r, rhoe_r, p_r

double precision :: rhostar_l, rhostar_r, rhoestar_l,
   rhoestar_r
double precision :: ustar, pstar, cstar_l, cstar_r
double precision :: lambda_l, lambdastar_l, lambda_r,
   lambdastar_r
double precision :: W_l, W_r, c_l, c_r, sigma
double precision :: alpha
double precision :: eint_l, eint_r

double precision :: rho_state, un_state, ut_state, p_state,
   rhoe_state
double precision :: rho_avg
double precision :: Q, p_min, p_max, p_lr, p_guess
double precision :: factor, factor2
double precision :: g_l, g_r, A_l, B_l, A_r, B_r, z
double precision :: S_l, S_r, S_c
double precision :: c_avg

double precision :: U_state(0:nvar-1)
double precision :: HLLCfactor

nx = qx - 2*ng; ny = qy - 2*ng
ilo = ng; ihi = ng+nx-1; jlo = ng; jhi = ng+ny-1

do j = jlo-1, jhi+1
 do i = ilo-1, ihi+1
    ! primitive variable states
    rho_l  = U_l(i,j,idens)

    ! un = normal velocity; ut = transverse velocity
    if (idir == 1) then
       un_l    = U_l(i,j,ixmom)/rho_l
       ut_l    = U_l(i,j,iymom)/rho_l
    else
       un_l    = U_l(i,j,iymom)/rho_l
       ut_l    = U_l(i,j,ixmom)/rho_l
    endif

    rhoe_l = U_l(i,j,iener) - 0.5*rho_l*(un_l**2 + ut_l**2)
    eint_l = rhoe_l/rho_l !this will be negative

    temp = [rho_l, eint_l]
    temp1 = 0.0d0 + pres(temp)
    p_l = temp1
    !p_l   = rhoe_l*(gamma - 1.0d0)
    p_l = max(p_l, smallp)

    rho_r  = U_r(i,j,idens)

    if (idir == 1) then
       un_r    = U_r(i,j,ixmom)/rho_r
       ut_r    = U_r(i,j,iymom)/rho_r
```

```fortran
   else
      un_r     = U_r(i,j,iymom)/rho_r
      ut_r     = U_r(i,j,ixmom)/rho_r
   endif

   rhoe_r = U_r(i,j,iener) - 0.5*rho_r*(un_r**2 + ut_r**2)
   eint_r = rhoe_r/rho_r

   temp = [rho_r, eint_r]
   temp1 = 0.0d0 + pres(temp)
   p_r = temp1
   !p_r    = rhoe_r*(gamma - 1.0d0)
   p_r = max(p_r, smallp)

   ! define the Lagrangian sound speed
   temp = [rho_l, p_l]
   temp1 = 0.0d0 + real_gamma(temp)
   gamma_l = temp1

   temp = [rho_r, p_r]
   temp1 = 0.0d0 + real_gamma(temp)
   gamma_r = temp1

   W_l = max(smallrho*smallc, sqrt(gamma_l*p_l*rho_l))
   W_r = max(smallrho*smallc, sqrt(gamma_r*p_r*rho_r))

   ! and the regular sound speeds
   c_l = max(smallc, sqrt(gamma_l*p_l/rho_l))
   c_r = max(smallc, sqrt(gamma_r*p_r/rho_r))

   ! define the star states
   pstar = (W_l*p_r + W_r*p_l + W_l*W_r*(un_l - un_r))/(W_l +
  W_r)
   pstar = max(pstar, smallp)
   ustar = (W_l*un_l + W_r*un_r + (p_l - p_r))/(W_l + W_r)

   ! now compute the remaining state to the left and right
   ! of the contact (in the star region)
```

```
    rhostar_l = rho_l + (pstar - p_l)/c_l**2
    rhostar_r = rho_r + (pstar - p_r)/c_r**2

    ! step II: wave speed estimate
    ! estimate the nonlinear wave speeds

    ! use the simplest estimates of the wave speeds
    S_l = min(un_l - sqrt(gamma_l*p_l/rho_l),
            un_r - sqrt(gamma_r*p_r/rho_r))
    S_r = max(un_l + sqrt(gamma_l*p_l/rho_l),
            un_r + sqrt(gamma_r*p_r/rho_r))

    ! Eqn 10.70 in Toro
    S_c = (p_r - p_l + rho_l*un_l*(S_l - un_l) -
          rho_r*un_r*(S_r - un_r))/ &
         (rho_l*(S_l - un_l) - rho_r*(S_r - un_r))

    ! step III: Compute the HLLC flux
    ! figure out which region we are in and compute the state
  and
    ! the interface fluxes using the HLLC Riemann solver
    if (S_r <= 0.0d0) then
       ! R region
       U_state(:) = U_r(i,j,:)
       call consFlux(idir, pres, idens, ixmom, iymom,
                    iener, nvar, U_state, F(i,j,:))
    else if (S_r > 0.0d0 .and. S_c <= 0) then
       ! R* region
       HLLCfactor = rho_r*(S_r - un_r)/(S_r - S_c)
       U_state(idens) = HLLCfactor
       if (idir == 1) then
          U_state(ixmom) = HLLCfactor*S_c
          U_state(iymom) = HLLCfactor*ut_r
       else
          U_state(ixmom) = HLLCfactor*ut_r
          U_state(iymom) = HLLCfactor*S_c
       endif
       U_state(iener) = HLLCfactor*(U_r(i,j,iener)/rho_r + &
               (S_c - un_r)*(S_c + p_r/(rho_r*(S_r - un_r))))
          ! find the flux on the right interface
          call consFlux(idir, pres, idens, ixmom, iymom,
                        iener, nvar, U_r(i,j,:), F(i,j,:))
          ! correct the flux
          F(i,j,:) = F(i,j,:) + S_r*(U_state(:) - U_r(i,j,:))
       else if (S_c > 0.0d0 .and. S_l < 0.0) then
          ! L* region
          HLLCfactor = rho_l*(S_l - un_l)/(S_l - S_c)
          U_state(idens) = HLLCfactor
          if (idir == 1) then
             U_state(ixmom) = HLLCfactor*S_c
             U_state(iymom) = HLLCfactor*ut_l
          else
             U_state(ixmom) = HLLCfactor*ut_l
             U_state(iymom) = HLLCfactor*S_c
          endif
          U_state(iener) = HLLCfactor*(U_l(i,j,iener)/rho_l + &
               (S_c - un_l)*(S_c + p_l/(rho_l*(S_l - un_l))))

          ! find the flux on the left interface
          call consFlux(idir, pres, idens, ixmom, iymom, iener,
    nvar, &
                         U_l(i,j,:), F(i,j,:))
          ! correct the flux
          F(i,j,:) = F(i,j,:) + S_l*(U_state(:) - U_l(i,j,:))
       else
          ! L region
          U_state(:) = U_l(i,j,:)

          call consFlux(idir, pres, idens, ixmom, iymom, iener,
    nvar, &
                         U_state, F(i,j,:))
       endif
    enddo
 enddo
 end subroutine riemann_HLLC
```

A.3 CFD code development git history for this study



```
Dec 29  master  Update README.md
                Upload new file
                Upload new file
Aug 18          Update README.md
                Update README.md
                Upload new file
                Add new directory
                updated
    10          shu_osher problem results
     9          updated shu_osher problem, fixed bugs
     7          shu-osher problem added
     4          updated
Jul 30          RK4 timestepping integration
    27          updated HLLC with real thermo
    26          Extended HLLC to real gas
    23          upload references
                Merge branch 'master' of git.uwaterloo.ca:spmuppar/TwoDThermoCode
                uploaded references
                Added references
    16          included coolprop
    11          cythonized PREOS
    10          Peng_robinson eos addition
     8          Added Peng robinson equation of state
     7          coupled with coolprop thermodynamics
     6          Merge branch 'master' of git.uwaterloo.ca:spmuppar/TwoDThermoCode
                modified the primitive variables
     4          Update README.md
                Upload new file
                Add readme.md
                updated riemann_CGF with real gamma
                updated riemann_CGF with real gas gamma
                Pushing code to git
```