Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

Erlend Austad Wesseltoft

A Numerical Investigation of Turbulent Combustion in Pool Fires using OpenFOAM

Master's thesis in Mechanical Engineering
Supervisor: Ivar Ståle Ertesvåg
Co-supervisor: Bima A Putra
August 2022

NTNU
Norwegian University of
Science and Technology

Erlend Austad Wesseltoft

A Numerical Investigation of Turbulent Combustion in Pool Fires using OpenFOAM

Master's thesis in Mechanical Engineering
Supervisor: Ivar Ståle Ertesvåg
Co-supervisor: Bima A Putra
August 2022

Norwegian University of Science and Technology
Faculty of Engineering
Department of Energy and Process Engineering

NTNU
Norwegian University of
Science and Technology

Abstract

Fires are a category of hazards that pose a large danger to human life and cause tremendous economic harm. Improved knowledge in the area of accurate modeling of fires is a first step towards gaining insight into its mechanisms and allows better decisions to be taken in an engineering context. Pool fires are one of the larger canonical fire categories and are the most common of all process industry accidents. This thesis aims to investigate the feasibility of conducting simulations of pool fires using the OpenFOAM simulation toolbox.

The numerical solver used in OpenFOAM is the *buoyantReactingFoam* solver, which includes the treatment of buoyancy. This is of importance for pool fires since buoyancy is the dominating factor driving the flow. The models used in the numerical setup include a Reynolds averaged Navier-Stokes (RANS) simulation using a detailed finite rate chemistry reaction mechanism for methanol with the eddy dissipation concept (EDC) model for combustion. The geometry and parameters in the case setup are based on previous experimental work conducted by Weckman and Strong. Details of the setup include a pan with a diameter of 30.5 cm filled with methanol, kept at a constant level with a feed rate of 1.35 cm^3/s for a total heat release rate is 24.6 kW.

The geometry of the case has been assumed axisymmetrical and is simulated as a wedge. Several different case setups have been created as problems with sustaining the combustion were experienced. Three partial cases were presented. The first case presented includes a change of combustion model to infinitely fast, the result of which was non-physical as the temperature field greatly exceeded the adiabatic flame temperature for methanol. The second case presented included no simplifications but proved numerically unstable, likely influenced by the ignition method. The final case presented included no simplifications and did not sustain combustion, being equivalent to a cold non-reacting flow. The results showed poor agreement with experimental- and numerical data available. Possible reasons for this are discussed.

Keywords: OpenFOAM, buoyantReactingFoam, EDC, finite-rate chemistry, pool fire, buoyancy, RANS

Sammendrag

Brann er en faretype som kan påføre store økonomiske tap og føre til tap av liv. Økt kunnskap rundt nøyaktig modellering av brann er ett av de første stegene for å få en bedre innsikt i dens mekanismer og gir bedre beslutningsgrunnlag i ingeniørsammenheng. Væskedamsbrannn er en av de større forbrennings kategoriene og er den vanligste ulykkesårsaken i prosessindustrien. Denne masteroppgaven har som formål å undersøke gjennomførbarheten av simulering av væskedamsbrann i simuleringsverktøyet OpenFOAM.

Den numeriske løseren brukt i OpenFOAM heter buoyantReactingFoam og inkluderer modeller for oppdrift, noe som er viktig for væskedamsbrann da dette er den dominante faktoren som driver strømningen. Modellene anvendt i det numeriske oppsettet inkluderer Reynolds midlet Navier-Stokes simulering med bruk av en detaljert kjemisk reaksjonsmekanisme, og eddy dissipation concept (EDC) som forbrenningsmodell. Geometrien og parameterene benyttet i det numeriske oppsettet er basert på tidligere eksperimentelt arbeid presentert av Weckman og Strong. Detaljer i oppsettet inkluderer en skål med diameter på 30.5 cm fylt med metanol, der væskenivået holdes i en konstant høyde ved en innmatingsrate på 1.35 cm^3/s, noe som gir en total effekt på 24.6 kW.

Geometrien i det numeriske oppsettet har blitt antatt å være aksesymmetrisk og simulert som formen av en kile. Flere forskjellige numeriske oppsett har blitt fremstilt da det oppsto problemer med å opprettholde forbrenning i modellen. Tre ulike oppsett er presentert. I den første er forbrenningsmodellen byttet til en uendelig rask forbrenningsmodell. Resultatene fra dette er ikke-fysiske da forbrenningstemperaturen oversteg den adiabatiske flammetemperaturen for metanol. Den andre oppsettet inkluderte ingen forenklinger, men viste seg å være numerisk ustabilt, noe som mest sannsynlig skyldes metoden brukt til påtenning. Det siste oppsettet presentert inkluderte heller ingen forenklinger, men forbeningen kunne ikke opprettholdes etter påtenning. Det tilsvarer da et tilfelle med kald, ikke reagerende strømning. Resultatene presentert var lite forenlig med tidligere eksperimentelle- og numeriske data tilgjengelig. Mulige grunner til dette er diskutert.

Nøkkelord: OpenFOAM, buoyantReactingFoam, EDC,
Detaljert kjemi, Væskedamsbrann, oppdrift, RANS

Acknowledgements

For their insight, technical knowledge, and helpful guidance, I would like to thank my supervisor professor Ivar S. Ertesvåg and my co-supervisor, PhD candidate Bima A. Putra. I would also like to thank my parents for their encouragement throughout my degree and my dog, Fia, for always lifting my spirit. Finally, I would like to thank Torunn Bakken for her unconditional support and patience.

Erlend Austad Wesseltoft

Trondheim, August 2022

Nomenclature

Roman Letters

C	Courant number	–
c	Speed of light	$\mathrm{m\,s^{-1}}$
c_p	Specific heat capacity	$\mathrm{J\,kg^{-1}\,K}$
$\mathcal{D}$	Diffusion coefficient	$\mathrm{m^2/s}$
D	Pool diameter	m
Da	Damkohlner numbe	–
f	Body forces	$\mathrm{kg/m^2s^2}$
G	Incident radiation	$\mathrm{W/m^2}$
g	Gravitational constant	$9.81\,\mathrm{m/s^2}$
h	Enthalpy	$\mathrm{J\,kg^{-1}}$
h	Planck constant	$6.626\,070\,15 \times 10^{-34}\,\mathrm{J\,s}$
H_f	Flame height	m
Δh_f^0	Formation enthalpy	$\mathrm{J\,kg^{-1}}$
I_λ	Spectral radiative intensity	W/mSr
I_b	Black body incident radiation	W/mSr
k	Turbulent kinetic energy	$\mathrm{m^2/s^2}$
k_b	Boltzmann constant	$1.380\,649 \times 10^{-23}\,\mathrm{m^2kg/s^2K}$
L	Length	m
$\dot{m}$	Mass exchange rate	$\mathrm{kg\,s^{-1}}$
MW	Molecular weight	kg/kmol
P	Production term	–
p	Pressure	Pa
Pr_t	Turbulent Prandtl number	–
$\dot{Q}$	Heat release rate	$\mathrm{W/m^3}$
$\dot{R}$	Reaction rate on mass basis	$\mathrm{s^{-1}}$
q	Radiative heat flux	$\mathrm{W/m^2}$
R_u	Universal gas constant	$8.314\,32\,\mathrm{J\,K^{-1}\,mol^{-1}}$

S_h	Source term for chemical reactions	$\mathrm{J/m^3s}$
S_{rad}	Source term for radiation	$\mathrm{J/m^3s}$
Sc_t	Turbulent Schmidt number	–
T	Temperature	K
u	Velocity	$\mathrm{m\,s^{-1}}$
Y	Mass fraction	–

Greek Letters

γ_*	Ratio of total mass to the mass in the fine structure	–
γ_λ	Ratio of total mass to the mass in the regions of fine structure	–
δ_{ij}	The Kronecker-Delta function	–
ϵ	The rate of dissipation of turbulent kinetic energy	$\mathrm{m/s^3}$
χ	Reacting fraction	–
μ_B	Bulk viscosity	$\mathrm{Pa\,s}$
ν_t	Turbulent kinematic viscosity	$\mathrm{m^2/s}$
ρ	Density	$\mathrm{kg/m^3}$
σ	Stefan-Boltzmann constant	$5.67 \times 10^{-8}\,\mathrm{W/m^2K^4}$
σ_s	Scattering coefficient	$\mathrm{m^{-1}}$
τ_{eqi}	Timescale to obtain equilibrium	s
τ_{res}	Residence timescale	s
φ	Any given variable	–

Subscripts and superscripts

$'$	Fluctuating value (Reynolds decomposition)
$''$	Fluctuating value (Favre decomposition)
$\tilde{}$	Favre average
$\bar{}$	Reynolds average
$*$	Fine structure
o	Surroundings
$fuel$	Fuel
k	For species k

max Maximum

min Minimum

oxi Oxidant

pr Product

Abbreviations

CFD Computational fluid dynamics

CFL The Courant–Friedrichs–Lewy condition

CPU Central processing unit

DAC Dynamic adaptive chemistry

DOM Discrete ordinance method

EDC Eddy dissipataion concept

FDS Fire dynamics simulator

FVM Finite volume method

GPU General processing unit

ISAT In situ adaptive tabulation

LES Large eddy simulations

NIST National Institute of Standards and Technology

ODE Ordinary differential equation

OpenFOAM Open source field operation and manipulation

PDE Partial differential equation

PSR Perfectly stirred reactor

RANS Reynolds averaged Navier Stokes

RHS Right hand side

RSM Reynold stress model

RTE Radiative transfer equation

TDAC Tabulation of dynamic adaptive chemistry

Table of Contents

1 Introduction

1.1 Background and Motivation

Fire

Combustion systems come in many different forms with differing properties, some examples of which can be seen in Figure 1. In this thesis, the type of combustion system investigated is pool fires, which can be categorised as non-premixed and turbulent.

Pool fires

Pool fires are fires that include a liquid phase of fuel on a surface and where the evaporation gases are ignited. In the real world, this can occur when fuel is spilt and ignited, which is a real safety concern and an explosion risk if the evaporated gasses are given enough time to mix with an oxidiser. According to Miao *et al.*[3] pool fires are the most common of all process industry accidents. Experiments investigating pool fires are often done in a circular or square pan, where the fuel is kept at a constant level by a head device. The existence of a rim on the pan's circumference also affects the flame's characteristics by changing the entrainment of air, and consequently, the induced turbulence [4].

Some early experimental work conducted by Blinov and Khudiakov (1957) remains some of the most complete, where various hydrocarbons and pool diameters were tested. Their conclusions are summarised in [4]. It was found that pools with diameters below 0.03 m were fully laminar, and pools in the range of $0.03 < D < 1.0$ m were in transitional behaviour. Where pools above $D > 1$ m were fully turbulent.

Pool fires are a category of fires with some good experimental data available such as in the main experiment being compared to in this paper 'Experimental investigation of the turbulence structure of medium-scale methanol pool fires' by Weckman and Strong [1] and some numerical investigations using OpenFOAM with LES, Sikic *et al.* [5], Chen *et al.* [6], Wang *et al.* [8], Razeghi *et al.* [9] among others.

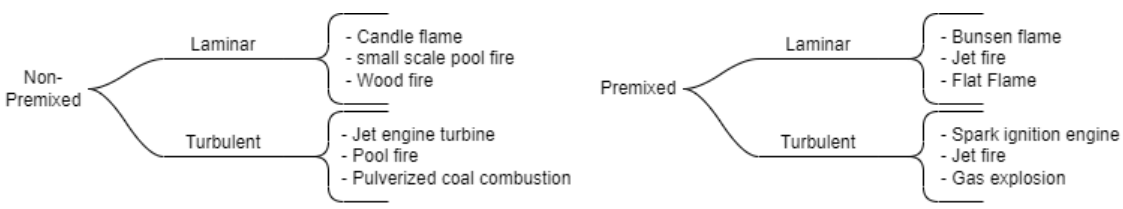


Figure 1: Examples of combustion systems ordered by turbulence and if fuel/oxidizer is mixed before burning (pre-mixed) or mixed continuously in the combustion zone (non-premixed) adapted from Table 1.2 in *Combustion* [2]

Pool fires have certain defining features that need special consideration when modelling. Pool fires are buoyancy dominated with almost zero initial momentum, in stark contrast to jet fires where momentum dominates the flow. Furthermore, they are influenced heavily by external factors such as wind. Being buoyancy driven with a large surface area also means that the flame behaves erratically and is inherently unsteady for diameters above 0.03 m.

The rate of energy release is the most important factor in determining the characterising behaviour of a pool fire, as found by Babrauskas and Peacock[11] among others. This energy release rate is closely tied to the liquid fuel's evaporation rate into gaseous form. The rate of which is mainly influenced by radiation and, in turn, the temperature field. For numerical simulations, this highly coupled problem can pose difficulty in modelling accurately. The oscillating nature of pool fires makes it so that a fixed point in space will often have large variations in temperature, especially near the top of the flame height and near the perimeter, [10]. This behaviour is driven by turbulent mixing. The resulting oscillations affect the height and shape of the flame, which goes through distinct phases. First, a plume starts to build and grow until it reaches the maximum height, after which it detaches, and the flame height is at its minimum. A sketch of which can be seen in Figure 2. This detaching frequency is described as the puffing frequency in the literature. This results in most of the studies on pool fires reporting temperatures along the centerline or temperature averages. Flame height has been determined to be where the flame is observed at least 50 % of the time. Empirical predictions of which can be found in engineering handbooks such as National Fire Protection Association and Society of Fire Protection Engineers [12] which states;

$$H_f = 0.235\dot{Q}^{\frac{2}{5}} - 1.02D \tag{1}$$

Where H_f denotes the flame height, $\dot{Q}$ is the heat release rate, and D is the pool's diameter. Other empirical correlations are presented by Thomas 1963 [13] among others.

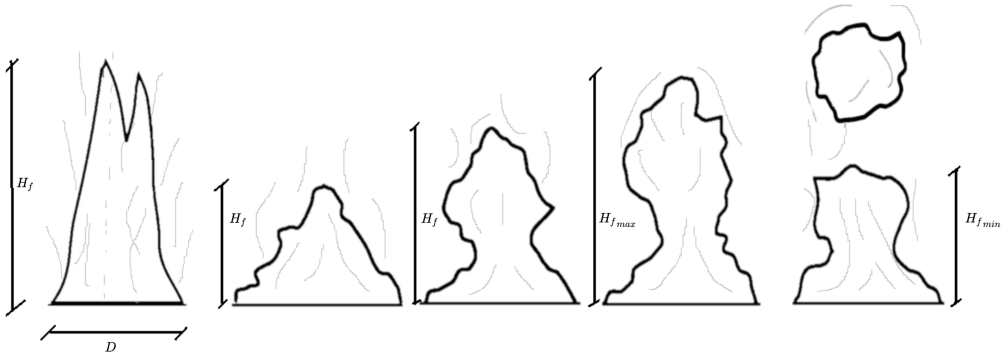


Figure 2: Flame height fluctuations for pool fire, adapted from *FDS-SMV*[10]

Context for fire modeling

Different approaches to the modelling of fires have a long and extensive history, and much of the complexity and scope has been expanded and developed with the pace of increasing computational power.

Some of the earliest approaches focused on correlations between the rate of heat release and flame height. Such as the work conducted by Thomas *et al.* (1961) [14], (1963) [13], Quintiere *et al.* (1979) [15]. The experiments conducted were limited to the scope of simple and well-behaving fires, such as fully developed and well-ventilated. More involved models that included mass and energy were developed in the 1970s and onward. Such as the zone model, where the compartment was subdivided into zones with uniform conditions. The work presented by Babrauskas and Williamson (1978) [16] among others details different approaches to this. The result includes average temperatures and plume heights but lacks calculations of the spatial distributions such as velocity profiles. This is due to it being 0-dimensional. The zone model can be a useful tool in a simplified analysis such as the fire engineering design handbook presented by Pettersson *et al.* (1976) [17].

A more detailed modelling approach is the use of computational fluid dynamics (CFD) models. This involves dividing the modelling domain into discrete subpartitions, such as finite volume cells of a given resolution and shape, and imposing conservation equations for mass, energy and momentum. In addition, the conditions for the boundaries of the domain and initial time have to be described. This can then be solved numerically with respect to each cell and, as such, produce results with the spatial resolution of the initial grid generated. This provides a much more detailed result compared to the earlier models at the cost of computational complexity. CFD methods are widely used in both an academic and commercial setting and provide precision and detail that is crucial to engineering decisions and academic insight.

With the momentous increase in computational power since the earliest models were proposed, in accordance with Moore's observation CFD based methods are dominating. Moreover, the ease with which a model can be implemented in CFD code and its numerical cost become important factors in the model choice. Compared to experimental data gathering, CFD allows for the investigation of high-resolution fields and full-scale geometries. On the other side, an experiment is limited to a finite set of probe points and restrictions on length scales to be able to be conducted in a laboratory environment.

Although the governing equations can be precisely described with a simple set of equations, accurately solving the fundamental conservation equations with the spatial and temporal resolution to capture the smallest scales of turbulence and chemistry is infeasible. A rough estimation of the time scale required for a direct simulation of a combustion flow is presented by Ertesvåg [18, p. 170], where 100 species who react on an average of 10 times each, with length and time scales of 1. The required computational operations are on the order of 10^{30}. In other terms, at 10^{20} operations per second, the simulation time would be roughly the age of the universe. Therefore, many simplifications and models must be applied to use in an engineering

context. Some of which is discussed in chapter 2 and 3.

1.2 Problem Description

The question investigated in this thesis is the viability of conducting combustion simulations in OpenFOAM with the use of detailed chemistry, the eddy dissipation concept (EDC) for combustion modelling, and Reynolds averaged Navier Stokes (RANS) for buoyancy dominated flow. With the goal that this knowledge can be used in more detailed simulations of compartment fires. As this can be an alternative to the more computationally expensive LES models. This is done in the following way;

1. Conduct a literature study of pool fires, including available measurements, numerical models presented by others and physical phenomena such as turbulent flow and combustion.
2. Getting familiar with a suitable version of OpenFOAM to simulate fires.
3. Conduct numerical simulations, compare with available experimental data. Present and discuss the findings.

1.3 Goal and Scope of the Thesis

The goal of this thesis is to explore the use of RANS simulations with finite rate detailed chemistry and the application of the EDC model in OpenFOAM to model a pool fire with special treatment of buoyancy. This is done to be able to compare to the experimental data presented by Weckman and Strong. This combination has not been explored in the literature, and there is a severe lack of documentation on how one would approach this in an engineering context. Therefore the goal is to get a working case setup to be able to model a pool fire accurately.

The scope of this thesis is to conduct numerical simulations with the given parameters and compare these to numerical data available in the literature. The initial choice of Weckman and Strong well documented pool fire provides experimental data with greater fidelity than many others. The scope of the geometry is limited to an axisymmetrical wedge, and the fuel is limited to methanol.

It can be noted that the scope has been continuously altered throughout the work on this thesis due to technical problems and a large amount of troubleshooting with the software package OpenFOAM. Therefore the resulting final case setup is somewhat lacking in terms of stability and accuracy, and further work needs to be done to validate the results produced by OpenFOAM. This is further discussed in Chapter 4.

1.4 State of the Art

Many commercial solutions are available for fire modelling, such as Ansys FLUENT and Siemens STAR-CCM packages. These software packages have a long development history and have been extensively validated against experimental data and, as such, represent a good option in modelling in an engineering context. The rapid integration of new methods and models from the literature is an upside of the commercial software packages. An example of this from Ansys is the implementation of numerical solvers in a way that has better parallel scaling. This is done to better align with the architecture of a GPU and, as such, can utilise the massive computational improvements GPUs have experienced in the last decades. One drawback with commercial solutions is the closed "black box" nature of the software and solver codes, as this can present an integrity problem, especially for academic endeavours. As the numerical results can be hard to impossible to reproduce in the future due to a lack of portability of work conducted, as the specifics of a simulation are locked in proprietary formats, and the numerical implementations are obfuscated to the user. Another drawback is the perpetual licensing cost of the software.

The open source and free solutions to CFD modelling are often plagued by a lack of maintainers and, as such, quickly fall into obscurity for the vast amount of users. One project that has continuously developed and expanded its capabilities is the OpenFOAM project, born out of a PhD project by Henry Weller and released publicly in the early 2000s. OpenFOAM has seen wide adoption in both academia and commercially. Many commercial actors opt to develop solvers and publicly share them to further their adoption alongside academics who conduct validation experiments of the numerical results. This symbiosis of differing motivations has improved OpenFOAM's capabilities and validation cases over the last decade and has gained widespread adoption. Since the software is released under the GLPv3 license, any modifications of the code have to be made publicly available. A survey of software usage on the top 50 HPC computers in 2017 [19] shows that OpenFOAM is among the top and is a worthy contender to the commercial solutions that exist.

There exist highly specialised implementations of sophisticated CFD, such as the work presented by Stephen Jones and Adam Lichtl in their "GPUs to Mars" talk[20], both working in rocket design simulation at SpaceX. They detail a novel approach to combustion modelling based on wavelets and a dynamic mesh. Designed from the ground up to be run on GPUs, and as such, presents impressive results with regards to detail and computational time. Approaches such as this heavily leverage the strength and weaknesses of modern hardware architectures. Such as solving the stiff set of chemistry ODEs on a parallel architecture such as GPUs while leaving sequential algorithms to be solved by the CPUs. This greatly reduces the time required for computations and offers higher fidelity results for the same cost.

Another open source alternative for fire simulations is the Fire Dynamics Simulator (FDS), released by the National Institute of Standards and Technology (NIST). Which, as stated on their website[10] '(LES) code for low-speed flows, with an emphasis on smoke and heat transport from fires'. This alternative is free to use and developed as a public service. FDS has excellent documentation and validation tests, but the software is limited in its application. Most academic endeavours studying

fires are currently utilising large eddy simulation (LES) simulations, in contrast to RANS simulations which are much less computationally expensive. Investigating the use of RANS simulations in OpenFOAM is of interest in an engineering context.

The work conducted by Weckman and Strong in their 'Experimental investigation of the turbulence structure of medium-scale methanol pool fires' (1996) was chosen as the starting point for experimental data for comparison since it is of high quality, and provides data for many different aspects of turbulent pool fires. The experiment consists of a 30.5 cm diameter pan, filled at constant rate of 1.35 cm^3/s with methanol (CH_3OH). This results in a total heat release rate of 24.6 kW. The error analysis presents that rms and mean values for velocities and temperature fields are accurate to $\pm 5\%$ at a 95% confidence. The temperate and flow field measurements were taken at nine increments in the radial directions, along with ten different heights above the fire. The paper also presents correlations for turbulent energy fields with iso-surfaces.

This paper has also been the basis for some numerical simulations in OpenFOAM such as previous work conducted by Sedano *et al.* [21] and Chen *et al.* [6] utilizing the fireFoam solver with LES. This was conducted using four different meshes ranging from 1.62 to 2.84 million cells, applying concepts such as EDC and using a one equation model for the chemistry. The authors found that the numerical results were in good agreement with the experimental data from Weckman and Strong. However, for the average and normalised results, the authors state in their conclusion; 'Nevertheless, when looking at the time averaged results and particularly temperature contours, the results show some discrepancy, once again proving the random nature of fires and turbulent reactive flows. Again, this is proved by comparing the numerical results of the contours of average temperature in the midplane with the experimental ones; this shows not only numerical but also geometrical discrepancies' [21].

A comprehensive report on evaluating the fireFoam solver in OpenFOAM is provided by the RISE fire research institute along with contributors from Lund university [22]. Where they give a detailed overview of the capabilities of OpenFOAM in fire research. Some notable findings in their conclusion include; 'In general, predictions of gas temperatures, gas velocities, gas concentrations, and heat fluxes by FireFOAM correlate with test results reasonably well for all the cases, although the flame temperatures were generally overestimated', which aligns with the mean temperature results from Sikic *et al.* for measurements 10cm above the pool. They also state that; 'However, it should be noted that heat loss to thick walls and smoke radiation are not well accounted for in the present version of FireFOAM. Caution needs to be taken when using FireFOAM to simulate scenarios relevant to these two phenomena.', Some research has been conducted on the radiation part since, such as Sun *et al.*[23], but have not yet been implemented. Its also worth noting that although the authors found excellent scaling on multiple compute nodes with OpenFOAM, NISTs FDS solver preformed 2.5 times faster in comparable simulation setups. Finally, the paper concludes with; 'Therefore, at present, FireFOAM is more a tool for researchers wishing to exploit some of the special capabilities of the code, than for the consultants wanting to use it for doing standard fire safety engineering analysis.'

1.5 Report Structure

This thesis is structured with a theory part, introducing the physical concepts relevant and the theory behind modelling approaches. A short introduction to the theory relating to the numerical implementation is then presented, alongside more detailed information on a few numerical components used. After this, the specifics of the implementation in OpenFOAM are presented, with relevant parameters used in the simulations. Results are presented for three different cases with comparisons to the experimental data. The results are then discussed with an emphasis on their erroneous nature. After this, a chapter on possible sources of error and possible solutions is presented, which detail some approaches taken in an attempt to remedy the problems experienced with the implementation in OpenFOAM. After this, a short conclusion is presented. Attached in the appendix are the relevant source code for the two solvers used, the code used to generate the mesh, alongside some brief details on the computer used, so that the work can be replicated by the reader.

2 Theoretical Background

Turbulent combustion is a field of process engineering involving a wide variety of different physical phenomena, such as radiation, chemistry, heat transfer, density changes and a complex flow field, among many others. This happens over a wide variety of length and time scales, from the smallest chemical scales to the large scales in the fluid flow. This presents a highly complex and coupled problem to model. Some of the fundamental theory and modelling approaches are presented in this chapter.

2.1 Averaging of Transport Equations

One approach to solving the governing fluid equations is decomposing the different quantities into a mean and fluctuating component. Where the mean quantity is only a function of spatial location and the fluctuating component is a function of both space and time. This solution strategy allows for different treatment of the terms when the main interest is the mean flow characteristics, which is often the case in an engineering design context.

Reynolds time averaging

Time averaging of the traditional transport equations allows for separation of the variable part from the mean. For a given quantity φ, we can introduce the relationships;

$$\varphi' = \varphi - \overline{\varphi} \tag{2}$$

$$\overline{\varphi} = \frac{1}{\Delta t} \int_{t-\frac{1}{2}\Delta t}^{t+\frac{1}{2}\Delta t} \varphi(\mathbf{x}, t) dt \tag{3}$$

$$\overline{u_i'} \equiv 0, \overline{p'} \equiv 0, \overline{\varphi'} \equiv 0 \tag{4}$$

Here the bar annotation represents the mean quantity.

However, this approach introduces many correlations between the quantity φ and density, such as in the momentum equation where ρ' and u' have to be modelled. For combustion and other reacting flows, large variations in density are present, and as such, the assumption of a constant density ρ is invalid [18]. To avoid additional modelling terms, the Favre mass-weighed average approach is used.

Favre mass-weighted averaging

The limitations with the use of Reynolds averaging occur when modelling combustion and other reacting flows. The assumption of constant density is invalid, which gives rise to additional terms that include both density fluctuations and velocity fluctuations. When dealing with flow of variable density, introducing the Favre mass-weighted averaging is advantageous. By introducing the relationship; [18]

$$\tilde{\varphi} = \frac{1}{\overline{\rho}} \frac{1}{\Delta t} \int_{t-\frac{1}{2}\Delta t}^{t+\frac{1}{2}\Delta t} \rho(\mathbf{x}, t)\varphi(\mathbf{x}, t)dt = \frac{\overline{\rho\varphi}}{\overline{\rho}} \tag{5}$$

$$\varphi = \tilde{\varphi} + \varphi'' \tag{6}$$

For a given transport quantity φ were both the mean and fluctuating components are separated into density weighted averaging. The bar annotation represents the same operation as in Reynolds averaging. This removes the density fluctuations from the averaged part of the equation, removing the hard-to-model terms. The density effects in turbulence still remain.

2.2 Conservation Equations

Using Favre mass-weighed averaging, as discussed in the previous section, the governing conservation equations can be rewritten. The equations are stated without detailed derivations, which can be found in textbooks such as *Turbulent strøyming og forbrenning*[18] and *An introduction to computational fluid dynamics*[24] among others.

Conservation of mass

The conservation of mass equation with Favre mass-weighted averaging can be found from the continuity equation as; [18]

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_j} \left(\overline{\rho} \tilde{u}_j \right) = 0 \tag{7}$$

When there is no net mass change the RHS of equation (7) reduces to 0. [18, p. 227]

Conservation of momentum

The momentum equation with Favre mass-weighted averaging can be stated as; [18, p. 228]

$$\frac{\partial}{\partial t}\left(\overline{\rho}\tilde{u}_i\right) + \frac{\partial}{\partial x_j}\left(\overline{\rho}\tilde{u}_i\tilde{u}_j\right) = -\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\overline{\tau}_{ij} - \overline{\rho u_i'' u_j''}\right) + \overline{\rho f}_i \tag{8}$$

Where $\overline{\tau}_{ij}$ is the viscous stress tensor;

$$\overline{\tau}_{ij} = \mu\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) + \left(\mu_B - \frac{2}{3}\mu\right)\frac{\partial \tilde{u}_k}{\partial x_k}\delta_{ij} \tag{9}$$

Using Stoke's hypothesis, the bulk viscosity μ_B is set to 0. The remainder of the second term also becomes 0 for flows with constant density.

Sutherland's equation (10) is used to model the viscosity. Values for T_s and A_s are determined empirically, and example of which can be found in Chapter 3.

$$\mu = \frac{A_s T^{3/2}}{T + T_s} \tag{10}$$

Conservation of species

The transport equations for species using Favre mass-weighted averaging from [18, p. 229];

$$\frac{\partial}{\partial t}\left(\overline{\rho}\tilde{Y}_k\right) + \frac{\partial}{\partial x_j}\left(\overline{\rho}\tilde{Y}_k\tilde{u}_j\right) = \frac{\partial}{\partial x_j}\left(\overline{\rho}\mathcal{D}_k\frac{\partial \tilde{Y}_k}{\partial x_j}\right) \tag{11}$$

$$-\frac{\partial}{\partial x_j}\left(\overline{\rho u_j'' Y_k''}\right) + \overline{R_k} + \frac{\partial}{\partial x_j}\left(\mathcal{D}_k \cdot \rho\overline{\frac{\partial Y_k''}{\partial x_j}}\right), \text{for k=1,N}$$

Where $\tilde{Y}_k$ is the Favre averaged mass fraction for species k. The $\overline{R_k} = \overline{\rho}\tilde{\omega}_k$ term is the Favre averaged chemical reaction rate which require modeling. This is discussed in section 2.6. The diffusion coefficient $\mathcal{D}_k$ for species k is often set to constant for all species to reduce complexity [18, p. 202]. The last term in Equation (11) is non-zero, but often omitted [18]. By applying the gradient diffusion assumption to the flux term $\overline{\rho u_j'' Y_k''} = \overline{\rho}\widetilde{u_j'' Y_k''}$ it can be restated with known quantities. By introducing the turbulent Schmidt number $Sc_k^t = \frac{\mu_t}{\rho\mathcal{D}_k^t} = \frac{\nu_t}{\mathcal{D}_k^t}$ for species k, the flux term can be expressed as;

$$-\widetilde{u_j'' Y_k''} = \mathcal{D}_k^t\frac{\partial \overline{Y}_k}{\partial x_j} = \frac{\nu_t}{Sc_k^t}\frac{\partial \overline{Y}_k}{\partial x_j} \tag{12}$$

Where the turbulent viscosity ν_t is determined from the turbulence model discussed in section 2.4. The turbulent Schmidt number is determined from experimental work to be set as a constant in the range of 0.7-1.0.

Conservation of enthalpy

The enthalpy can be split into a sensible and chemical component, as this is advantageous for CFD applications. This avoids terms that include heat flux and chemistry. As defined in [24, p. 347];

$$h = \sum_{k=1}^{N} Y_k \left(\int_{T_0}^{T} \underbrace{c_{pk} dT}_{\text{sensible}} + \underbrace{\Delta h^0_{f,k}}_{\text{chemical}} \right) \tag{13}$$

Where the heat capacity at constant pressure c_{pk} and the formation enthalpy $\Delta h^0_{f,k}$ are given for species k.

The enthalpy conservation equation with Favre mass-weighted averaging from [18, p. 229];

$$\frac{\partial}{\partial t}(\overline{\rho}\tilde{h}) + \frac{\partial}{\partial x_j}\left(\overline{\rho}\tilde{h}\tilde{u}_j\right) = \frac{\partial}{\partial x_j}\left(\overline{\rho}\alpha\frac{\partial \tilde{h}}{\partial x_j} - \overline{\rho u_j'' h''}\right) + \overline{S_{rad}} + \overline{S_h} \tag{14}$$

The term $\overline{S_{rad}}$ includes contributions from radiation, the $\overline{S_h}$ includes heat released from combustion and other source terms. By introducing the gradient diffusion assumption the turbulent enthalpy flux term $\overline{\rho u_j'' h''} = \overline{\rho}\widetilde{u_j'' h''}$ and by introducing the turbulent Prandtl number $\mathrm{Pr}_t = \frac{\mu_t}{\lambda_t / C_p} = \frac{\nu_t}{\alpha_t}$, The turbulent enthalpy flux term can be expressed as;

$$-\widetilde{u_j'' h''} = \alpha_{\mathrm{t}} \frac{\partial \tilde{h}}{\partial x_j} = \frac{\nu_{\mathrm{t}}}{Pr_t}\frac{\partial \tilde{h}}{\partial x_j} \tag{15}$$

Where the turbulent viscosity ν_t is determined from the turbulence model discussed in section 2.4, and the turbulent Prandtl number is set in the same way as the turbulent Schmidt number, as a constant in the range of 0.7-1.

2.3 Ideal Gas Relations

For determining the thermodynamic equation of state, the ideal gas law has been used. from [24];

$$\rho = \frac{p}{R_u T \sum_{k=1}^{N} \frac{Y_k}{(MW)_k}} \tag{16}$$

Here $(MW)_k$ is the molecular weight of species k, and R_u is the universal gas constant.

Enthalpy from Empirical Data

This relation can then be used to determine the specific heat capacity c_p as a function of T. This in turn is used to determine enthalpy by the relation $h_s = h(T) - h(T_0)$. A widely used method based on polynomial curve fitting, where the constants are found empirically, is on the form;

$$c_p = R_u[a_1 + a_2T + a_3T^2 + a_4T^3 + a_5T^4] \tag{17}$$

$$H^\circ = R_uT[a_1 + \frac{a_2}{2}T + \frac{a_3}{3}T^2 + \frac{a_4}{4}T^3 + \frac{a_5}{5}T^4 + \frac{a_6}{T}] \tag{18}$$

Further discussion in determining the values can be found in Chapter 3.

2.4 Turbulence Modeling

Turbulence remains one of the largest unsolved problems to model in classical physics. Although it is fully mathematically described by the work of Navier and Stokes in the middle of the 19th century, describing it in discrete terms imposes large requirements on computational power and storage size, which makes it infeasible. Modelling approaches, therefore, have to be applied.

> " Big whirls have little whirls which feed on their velocity, And little whirls have lesser whirls And so on to viscosity. "
>
> *Lewis Fry Richardson, 1922*

The problem arises in modeling the mass-weighted Reynold stress term $-\overline{\rho}\widetilde{u_i''u_j''}$ from the Favre-averaged momentum in Equation (8). Non-linear convection gives rise to velocity fluctuations and shear stress, which makes 6 additional unknowns that need transport equations to be closed. Reynold stress model (RSM) is a comprehensive model involving 6 additional PDEs to achieve closure and is the most computationally expensive option. Simplified 2-equation models have shown good agreement with experimental data at a reasonable computational cost. One such example is the k-ε 2-equation model chosen, where the turbulence energy k and its dissipation rate ε are given their own transport equations.

Using the Boussinesq eddy viscosity assumption for the Favre stresses;

$$-\overline{\rho u_i'' u_i''} = -\overline{\rho}\widetilde{u_i''u_j''} = \mu_\mathrm{t}\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) - \frac{2}{3}\left(\overline{\rho}\tilde{k} + \mu_\mathrm{t}\frac{\partial \tilde{u}_l}{\partial x_l}\right)\delta_{ij} \tag{19}$$

Where δ_{ij} denotes the Kronecker-Delta function.

The k-ε Turbulence Model

Using the Boussinesq assumption in equation (19) and the the k-ε turbulence model. The Favre mass-weighted averaged turbulent kinetic energy $\tilde{k}$ is defined as;

$$\tilde{k} = \frac{1}{2}\frac{1}{\overline{\rho}}\overline{\rho u_i'' u_i''} \tag{20}$$

The turbulent viscosity μ_t can be expressed as

$$\mu_t = \overline{\rho}\nu_t = C_\mu \overline{\rho}\frac{\tilde{k}^2}{\tilde{\varepsilon}} \tag{21}$$

Where the scalar quantity μ_t can be expressed in terms of $\tilde{k}$ and $\tilde{\varepsilon}$.

Which are found by the transport equations for the Favre mass-weighted averaged $\tilde{k}$ and $\tilde{\varepsilon}$ from [18, p. 55];

$$\frac{\partial}{\partial t}(\overline{\rho}\tilde{k}) + \frac{\partial}{\partial x_j}\left(\overline{\rho}\tilde{k}\tilde{u}_j\right) = \frac{\partial}{\partial x_j}\left(\left(\mu + \frac{\mu_\mathrm{t}}{\sigma_k}\right)\frac{\partial \tilde{k}}{\partial x_j}\right) + P_k - \overline{\rho}\tilde{\varepsilon} \tag{22}$$

$$\frac{\partial}{\partial t}(\overline{\rho}\tilde{\varepsilon}) + \frac{\partial}{\partial x_j}\left(\overline{\rho}\tilde{\varepsilon}\tilde{u}_j\right) = \frac{\partial}{\partial x_j}\left(\left(\mu + \frac{\mu_\mathrm{t}}{\sigma_\varepsilon}\right)\frac{\partial \tilde{\varepsilon}}{\partial x_j}\right) + C_{\varepsilon 1}\frac{\tilde{\varepsilon}}{\tilde{k}}\overline{\rho}P_k - C_{\varepsilon 2}\frac{\tilde{\varepsilon}}{\tilde{k}}\overline{\rho}\tilde{\varepsilon} \tag{23}$$

Where the production term P_k is defined as [18, p. 55];

$$\overline{\rho}P_k = \mu_\mathrm{t}\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right)\frac{\partial \tilde{u}_i}{\partial x_j} - \frac{2}{3}\left(\overline{\rho}\tilde{k} + \mu_\mathrm{t}\frac{\partial \tilde{u}_l}{\partial x_l}\right)\frac{\partial \tilde{u}_i}{\partial x_i} \tag{24}$$

Where C_μ, $C_{\varepsilon 1}$, $C_{\varepsilon 2}$, σ_k and σ_ϵ are modeling constants found empirically. The coefficients proposed by Launder and Spalding are listed in Table (1).

Table 1: The modeling constants for $k-\epsilon$ from Launder and Spalding 1974[25]

C_μ	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$	σ_k	σ_ϵ
0.09	1.44	1.92	1.0	1.3

2.5 Combustion Modeling

Determining the reaction rate is fundamental to combustion modelling. Relating the flow characteristics and the chemistry often poses a complex task. Modelling of the mean reaction rate found in equation (11) has many different approaches, such as the EDC model presented here. This is a method that is advantageous and widely used in CFD code.

The Eddy Break-up Model proposed by Spalding 1971 is one such combustion model. Where it is assumed that turbulent mixing determines the reaction rate. Such that in a small control volume, the timescale to achieve equilibrium τ_{eqi} is much less than the residence timescale τ_{res} before being transported out of the control volume by convection. The dimensionless Damkohlner number (Da) can be used to evaluate this assumption.

$$Da = \frac{\text{reaction rate}}{\text{convective mass transport rate}} \propto \frac{\tau_{\text{res}}}{\tau_{eqi}}$$

Such that when the Da number is large, the mixture has enough time to reach equilibrium before further mixing occurs, and the assumption is valid. The Eddy Break-up Model reliance on flow properties can cause unwanted effects such as local extinction for large amounts of turbulence and requires determining these quantities accurately.

2.6 Eddy Dissipation Concept

The Eddy Dissipation Concept (EDC) first introduced by Magnussen and Hjertager (1981) builds on the Eddy Break-up Model by Spalding. EDC is a detailed chemistry model and differs from the eddy break-up model by still being applicable in cases where chemical kinetics are dominating. The EDC combustion model assumes that the molecular mixing required for reactions to take place happens in the smallest eddies. This is on the order of the Kolmogorov scale at sufficiently high temperatures.

The EDC combustion model can be split into two parts, the cascade and the reactor model. The cascade model relates the lower frequencies of turbulence from the mean flow in a causal relationship all the way down to the highest frequencies in the fine structures. Where it is assumed that all chemical reactions and most of the turbulent dissipation happen (Re dependent) in the fine structures. The fine structures are modelled as perfectly stirred reactors, where the reactants must have a sufficient residence time and mixture composition for combustion to occur before the control volume exchanges mass balance within the surroundings. The quantities in the fine structures are denoted by $(*)$, and the surrounding quantities are denoted by (o).

By defining the characteristic scales for velocity and length in the fine structure as; [18]

$$\begin{aligned} L^* &= \frac{2}{3}\left(\frac{3C_{D2}^3}{C_{D1}^2}\right)^{1/4}\left(\frac{v^3}{\epsilon}\right)^{1/4} \\ u^* &= \left(\frac{C_{D2}}{3C_{D1}^2}\right)^{1/4}(v\epsilon)^{1/4} \end{aligned} \tag{25}$$

Where $C_{D1} = 0.134$ and $C_{D2} = 0.50$ are constants given by Magnussen.

In the areas occupied by smaller frequency turbulence, regions of fine structure volumes will form. Within the volume of the fine structure region, the fine structures can be found. Some of the areas within the structure have many such fine structures, and others few. The ratio of total mass to the mass in the regions of fine structure can be defined as;[18]

$$\gamma_\lambda = \frac{u^*}{u'} = 2.1 \left(\frac{v\epsilon}{k^2}\right)^{1/4} \tag{26}$$

Where the ratio between total mass to the mass in the fine structure, γ^*, can be defined as; [18]

$$\gamma^* = \left(\frac{u^*}{u'}\right)^3 = 9.8 \left(\frac{v\epsilon}{k^2}\right)^{3/4} \tag{27}$$

The relation between the quantities therefore become $\gamma_\lambda = (\gamma^*)^{1/3}$. The mass exchange rate from the fine structure to the and the surroundings, divided by the mass of the fine structure, be defined as; [18]

$$\dot{m}^* = 2\frac{u^*}{L^*} = 2,5 \left(\frac{\varepsilon}{\nu}\right)^{1/2} \tag{28}$$

Where the residence time in the fine structure τ^* is the inverse of the mass exchange, such that $\tau^* = \frac{1}{\dot{m}^*}$. The relation between the Favre mass-weighed average mean state and the fine structure state for a given quantity φ is given as;

$$\tilde{\varphi} = \gamma^* \chi \varphi^* + (1 - \gamma^* \chi)\, \varphi^o \tag{29}$$

Applying a mass balance to the control volume of the fine structure and Favre mass-weighed averaging, one can obtain an expression for the reaction rate; [18]

$$-\bar{R}_k = \frac{\bar{\rho}\dot{m}^*\gamma^*\chi}{1 - \gamma^*\chi}\left(\tilde{Y}_k - Y_k^*\right) \tag{30}$$

The reaction rate $\bar{R}_k$ is seen as a source term in the species equation (11). For species $k = 1...N$. The mass fraction in the fine structure Y_k^* and the reacting fraction in the fine structure χ can be modelled in multiple ways. The three main ways of determining Y_k^* are categorised in both fast- and detailed chemistry approach and the local extinction approach[28].

For the simpler case of a fast chemistry approach, a simple stoichiometric balance can be applied. Such that

$$1\text{kg Fuel } + r\text{kg Oxidizer } = (1 + r)\text{kg Products}$$

Where r is the stoichiometric Oxidiser. The reaction rate of the fuel can then be stated as;[18]

$$-\bar{R}_{fuel} = \frac{\bar{\rho}\dot{m}^*\gamma^*\chi}{1-\gamma^*\chi}\tilde{Y}_{min} \tag{31}$$

Where $\tilde{Y}_{min}$ is defined as;[18]

$$\tilde{Y}_{\min} = \min\left(\tilde{Y}_{\text{fuel}}, \frac{1}{r}\tilde{Y}_{\text{oxidiser}}\right). \tag{32}$$

The reacting fraction in the fine structure χ is dependent on three factors, the model version presented by Magnussen (1989) and later reformulated by Gran (1994);

$$\chi = \chi_1 \cdot \chi_2 \cdot \chi_3 \tag{33}$$

$$\begin{aligned}
\chi_1 &= \frac{\left(\tilde{Y}_{\min}+\tilde{Y}_{\text{pr}}/(1+r)\right)^2}{\left(\tilde{Y}_{\text{fuel}}+\tilde{Y}_{\text{pr}}/(1+r)\right)\left(\tilde{Y}_{\text{oxi}}/r+\tilde{Y}_{\text{pr}}/(1+r)\right)} \\
\chi_2 &= \min\left\{\frac{1}{\gamma_\lambda}\cdot\frac{\tilde{Y}_{\text{pr}}/(1+r)}{\tilde{Y}_{\text{pr}}/(1+r)+\tilde{Y}_{\min}}, 1\right\} \\
\chi_3 &= \frac{1}{\gamma_\lambda}\min\left\{\frac{\gamma_\lambda\left(\tilde{Y}_{\text{pr}}/(1+r)+\tilde{Y}_{\min}\right)}{\tilde{Y}_{\min}}, 1\right\}
\end{aligned}$$

Where the different χ can be interpreted as; χ_1 Probability of reactants coexisting, χ_2 The degree of heating and χ_3 is a limiter due to a lack of reactants [18].

For the detailed chemistry approach, the fine structure is considered as an adiabatic homogeneous perfectly stirred reactor (PSR) and solved with a set of ordinary differential equations (ODE). When applying the detailed chemistry approach, the value of the reacting fraction can χ can usually be set to 1 for simplicity, as this has been found to give similar results. [29]

By setting the reacting fraction to 1 in Equation (30), in addition to using γ_λ from Equations (26) and (27), the expression for the reaction rate can be restated as;

$$-\bar{R}_{fuel} = \bar{\rho}\dot{m}^*\frac{\gamma_\lambda^3}{1-\gamma_\lambda^3}\tilde{Y}_{min} \tag{34}$$

Where the exponents for γ_λ are modified to a value of 2 in the 2005 version of the model proposed by Magnussen[30].

Assuming the fine structure can be described as a PSR, the system of ODEs for describing the mass fraction used in the reaction rate Equation (30) becomes; [29]

$$\frac{dY_k^*}{dt} = \frac{R_k^*}{\rho^*} + \frac{1}{\tau^*}\left(Y_k^o - Y_k^*\right) \tag{35}$$

Where the reaction rate within the fine structure R_k^* is given from the detailed chemical kinetics mechanism, the mass fraction entering the fine structure from the surroundings Y_k^o can be found from the relation given in Equation (29). The inverse of the residence time scale $\frac{1}{\tau^*}$ is equal to the mixing rate $\dot{m}^*$. [31]

2.7 Radiation

Radiation differs from other energy transfer mechanisms in combustion due to the energy transfer happening at the speed of light in the electromagnetic spectrum and not by molecular interactions at close distances. Additionally, it is strongly temperature dependent, being proportional to the temperature to the fourth power as seen in equation (38). This makes radiation often a significant aspect of heat transfer in combustion, where it often is on the same order of magnitude as convection. This differs from other engineering applications where radiation is often neglected, where temperature differences are insignificant.

Different wavelengths of radiation are emitted from all matter above absolute zero (0 K), with the dominating wavelength being determined by the temperature. The sum of these wavelengths is the energy transfer. This spectral distribution is absorbed and emitted with different affinities by the species involved, and this is also temperature dependent.

Radiation being a complex energy transfer involving a wide array of wavelengths in all directions, including the effects of absorption, emission and scattering, makes it a challenge to discretise. This makes the implementation in CFD modelling important, and the simplifications made can greatly affect the computational cost. The most accurate models include models based on ray–tracing, where a discrete amount of radiative rays are modelled as travelling in a given direction and interacting with the medium a given amount of reflections each time–step. This approach can be prohibitively expensive in most applications. A simpler variant is the discrete ordinance method (DOM). This approach is similar in nature but differs in limitations on directions. An even simpler model is the PN model, being based on expanding spherical harmonics, where the N denotes how many terms are included in a series expansion. The simplest is the P1 model.

Radiative Transfer Equation

From the enthalpy equation (14) we can see that radiation show up as a source term. In this section, the radiative transfer equation (RTE) is introduced to model this term. The RTE describes a time dependent change in radiation intensity for a ray moving in a given direction through a medium. From *Radiative Heat Transfer* [32, p. 270];

$$\frac{dI_\lambda(\mathbf{r},\mathbf{s})}{ds} = \underbrace{-\kappa_\lambda I_\lambda(\mathbf{r},\mathbf{s})}_{\text{absorption}} + \underbrace{\kappa_\lambda I_{b,\lambda}(\mathbf{r})}_{\text{emmission}} \underbrace{-\sigma_{\lambda,s} I_\lambda(\mathbf{r},\mathbf{s})}_{\text{Out-scattered intensity}} + \underbrace{\frac{\sigma_{s\lambda}}{4\pi}\int_{4\pi} I_\lambda(\mathbf{s}_i)\,\Omega(\mathbf{s},\mathbf{s}_i)\,\Omega_i}_{\text{In-scattered intensity}} \tag{36}$$

Where spectral radiation intensity I_λ for a given coordinate r, propagating in a given direction $\mathbf{s}$ is defined in terms of the absorption, emission and scattering terms, where κ and σ_s are the absorption and scattering coefficients for a given wavelength λ, the b denotes radiative blackbody. Ω is the function describing the scattering phase.

The solid angle from an arbitrary point within a sphere is given as 4π steradians, and as such, the area integrated over in the scattering term represents a unit sphere.

The scattering term in the RTE describes the scattering interactions in the control volume, and the $\mathbf{s_i}$ denotes the directions differing from $\mathbf{s}$. This term is mostly neglected in combustion where the only species are gaseous and is neglected in further discussion. This leaves the absorption and emission terms. The spectral radiance I_λ is given in terms of effect per area per solid angle per wavelength and is a directional quantity.

Both the absorption and emission terms include the spectral absorption coefficient κ_λ with the unit of per meter. This can be viewed as the inverse of the mean distance of travel before absorption for a given radiation beam. This coefficient is a property of the medium that is traversed. These properties are related to the structure on a molecular scale and, therefore, a function of temperature, species ratio and pressure for each wavelength. It should be mentioned that while neglecting scattering, κ_λ is proportional to the rate change of spectral radiation intensity.

The absorption term represents the attenuation of the spectral radiation intensity. The emission term represents the radiative energy increase due to radiative emission from the medium, this being proportional to the black body incident radiation I_b.

The RTE definition of I can then be used further to derive the total radiation intensity G and divergence of the heat flux vector $\mathbf{q}$. These are relevant quantities in the energy equations by showing up in the enthalpy equation as source terms.

Expressing the spectral radiance $I_{b\lambda}$ as a function of temperature can be done by introducing Planck's law for blackbody radiation; [32]

$$I_{b\lambda}(T) = \frac{2hc^2}{\lambda^5}\frac{1}{e^{\frac{hc}{\lambda k_B T}} - 1} \tag{37}$$

Where h and k_b are the Planck and Boltzmann constants, respectively, and c is the speed of light in the medium.

Assuming that the properties of the medium are independent of wavelength, also called the grey-gas assumption, Equation (37) can be integrated over all wavelengths for blackbody approximation;

$$I_b = \int_0^\infty I_{b\lambda} d\lambda = \frac{\sigma}{\pi} T^4 \tag{38}$$

Where σ denotes the Stefan-Boltzmann constant given as $5.67 \times 10^{-8} [\frac{W}{m^2 K^4}]$.

In a similar manner, the total radiation intensity G can be found from integrated over all wavelengths and directions;

$$G = \int_0^\infty \int_{4\pi} I_\lambda d\Omega \tag{39}$$

Using the definition of the radiative heat flux vector $\mathbf{q}$ from Modest [32, p. 473];

$$\mathbf{q} = \int_0^\infty \int_{4\pi} I_\lambda s d\Omega \tag{40}$$

Where $\mathbf{s}$ is the direction vector. Using the assumption of the absorption coefficient κ being independent of wavelength in addition to the medium is non-scattering. This simplifies it to a grey medium assumption, resulting in the subscript of λ being removed such that $\kappa_\lambda = \kappa$.

With these assumptions, The RTE (36) can be restated as;

$$-\nabla \cdot \mathbf{q} = \underbrace{\kappa G}_{\text{source}} - \underbrace{4\kappa\sigma T^4}_{\text{sink}} \tag{41}$$

This gives an expression of the flux $\mathbf{q}$ equal to the source term S_{rad} in the entalpy equation (14).

The unknown quantity in equation (41) that remains is the total radiation intensity G. In order to create an expression for G with known quantities, a modelling approach has to be applied, and a transport equation for G has to be formulated.

P1 model

One approach to formulating a transport equation for G is the PN approach. Where in the radiative intensity field $I_\lambda(\mathbf{r}, \mathbf{s})$ is described as a two-dimensional Fourier series. From Modest[32, p. 466];

$$I(\mathbf{r}, \mathbf{s_i}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} I_l^m(\mathbf{r}) Y_l^m(\mathbf{s_i}) \tag{42}$$

Expanding this series and truncating it at N number of terms, where the simplest model is truncated at the first term. This is the P1 model. Without further derivations, using the assumption of grey gas and non-scattering mediums the RTE equation (36) with the Fourier series (42) reduces to; [32, p. 466];

$$q = -\frac{1}{3\kappa}\nabla G \tag{43}$$

Which when combined with equation (41) gives the following transport equation for the total radiation intensity G;

$$\nabla \cdot \left(\frac{1}{3\kappa}\nabla G\right) = \kappa G - 4\kappa\sigma T^4 \tag{44}$$

Other Radiation Models

The P1 is most accurate in situations where the medium can be approximated as an optically thick medium. An optically thick medium has an isotropic radiation intensity field [32]. In situations where this approximation falls short, the PN model lacks accuracy and can overpredict certain interactions.

More involved models include the discrete ordinance method (DOM), where the RTE equation is solved for a given amount of solid angles, where the number of solid angles solved can be chosen. This method provides a more accurate prediction of the temperature field and is widely used in combustion modelling, especially where radiation is dominating. The cost of this method is a large increase in computational requirement compared to the P1 model.

The P1 radiation model has been used in the numerical setup seen in chapter 3.

2.8 Numerical Implementations

This section aims to give a very brief introduction and overview of some of the numerical schemes and special numerical implementations used in OpenFOAM. Implementation of which is discussed in the next chapter.

Discretization

In order to implement the continuous equations described in the previous chapter, they first need to be limited to the discrete realm. There are many methods to achieve this with differing stability and accuracy characteristics.

Discretization of time

The discretisation of time is done with the implicit Euler Method. For a given quantity φ in discrete time steps is given as;

$$\frac{\partial}{\partial t}(\varphi) = \frac{\varphi - \varphi^0}{\Delta t} \tag{45}$$

Where the superscript of 0 denotes the time step. This method has the properties of being implicit, with a first-order truncation error.

Discretization of Domain

The discretisation of the domain is done by means of the finite volume method (FVM). This method divides the domain into discrete, non-overlapping volumes. Volumes are defined by vertices surrounding the cell centre, with cell faces in every direction. This method is preferred in fluid flow since it allows for easier implementation of the conservation equations by imposing strict conservation across the cells. The collection of these cells across the domain is referred to as the mesh. The way these cells are constructed is discussed in section 3.2.

CFL condition

The Courant–Friedrichs–Lewy condition (CFL) condition gives an upper bound for the stability of numerical schemes. For the n-dimensional case; [24]

$$C = \Delta t \left(\sum_{i=1}^{n} \frac{u_i}{\Delta x_i} \right) \leq C_{\max} \tag{46}$$

Where C is the Courant number, Δt is the time step. u_i and Δx_i is the velocity and length step in direction i.

2.9 TDAC

Solving the equations relating to chemistry involves solving a set of ODEs. This set of equations relating to chemistry is often a significant part of the computational cost for each time step when dealing with reacting fluids. Therefore optimisation in the solving strategy can have a large impact on computational requirements for each time step. The reason for the high cost of solving this set of equations comes from large gradients over a short time, and as such traditional ODE solvers need to reduce their time step to be able to capture the rapid change and not become numerically unstable.

A fairly new solution strategy named tabulation of dynamic adaptive chemistry (TDAC) was introduced and applied by Contino *et al.*Contino *et al.* It details a method applying partial tabulation and mechanism reduction at run time. This is a combination of the more well known In situ adaptive tabulation (ISAT) by Pope (1997)[34], which uses tabulations from previous timesteps. And dynamic adaptive chemistry (DAC) by Liang *et al.*(2009)[35], which is a method to reduce the reaction mechanism at runtime. This has shown speedups in the range of 1 or 2 orders of magnitude[36] compared to normal chemistry solving. Where very detailed mechanisms on the order of 1000 species have shown immense speedups,

as shown summarised in [37] a 500x speedup on the solution time for the chemistry component was achieved. This method is implemented in OpenFOAM and has been tested for the detailed chemistry approach.

2.10 PIMPLE Algorithm

The PIMPLE algorithm is a mix between the well known PISO and SIMPLE algorithms for pressure correction. This pressure-velocity coupling method is widely used in OpenFOAM. For a given timestep, the algorithm solves the pressure field and imposes mass conservation. This is done iteratively while correcting velocity to impose conservation. The momentum predictor can alternatively be turned on, where the momentum equation is solved first. This is the default behaviour in OpenFOAM. A flow chart detailing its operation is shown in Figure 3.

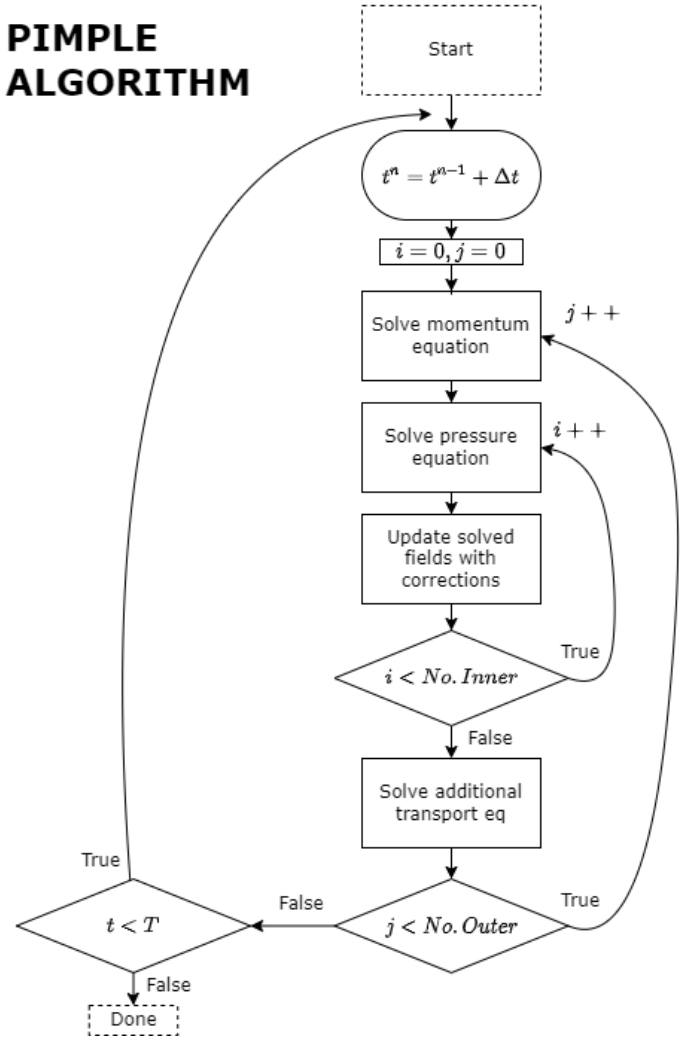


Figure 3: PIMPLE algorithm solution flow

3 Computational setup

This chapter gives an overview of the OpenFOAM toolbox used for the numerical simulations, emphasising the utilities utilised. Along with the details of the case setups and the parameters used.

3.1 OpenFoam

The main numerical tool used is OpenFOAM, which is a C++ based library toolbox. OpenFOAM comes pre-packaged with multiple numerical solvers and example case setups to allow the user a starting point for their own implementations. This differs from many other commercial solutions by being open sourced and licensed under the GPL-v3 licence. This allows the user to modify and recompile the software to fit their needs.

It is worth mentioning that there are two competing releases of OpenFOAM with the same name. One of them is released by the ESI-Group with a biannual release target. This version can be identified by the YYMM naming convention for versioning, the latest being v2206. The other is released by the original team under the name The OpenFOAM Foundation Ltd, targeting an annual release. Version naming is a simple increment from the last version. The latest being v10, released in July 2022. Compatibility between the two releases is good, with some manual renaming of functions and enumeration objects often being required.

The main version used for this project is version 9, released by The OpenFOAM Foundation Ltd. This is mainly due to the old *fireFoam* solver getting an overhaul with better buoyancy treatment, also being renamed to *buoyantReactingFoam* for this version onward. Some work has also been conducted in version 7 due to previous related work being done in this version and help received from PhD candidate Bima A Putra.

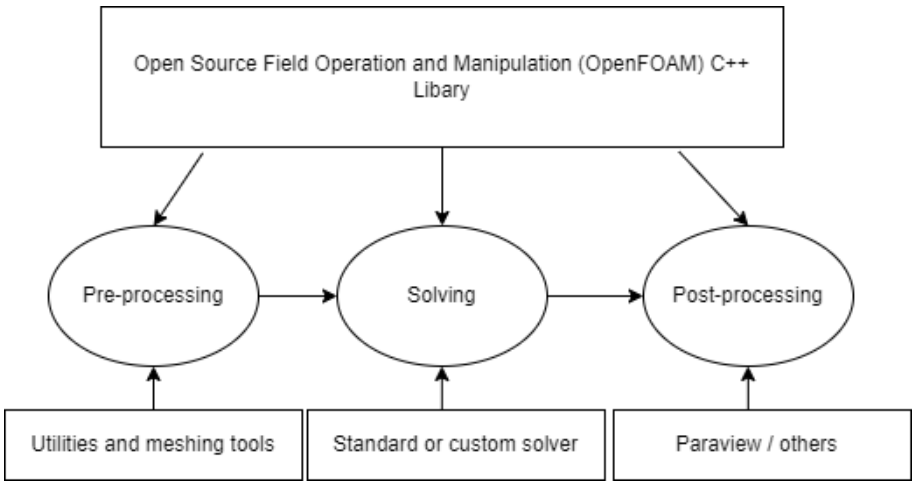


Figure 4: Overview of the OpenFOAM application workflow

OpenFOAM documentation

The documentation available for OpenFOAM is released alongside the version release in the form of a user guide [38], giving an overview and introduction to many of the features included. Another form of documentation released is the annotation in the source code, and this can be compiled to a more readable form with a tool like Doxygen [39]. The support and documentation required beyond this are mainly crowd sourced. This comes in the form of people detailing their work in papers or blog posts, alongside CFD forums such as CFD-online, where people can discuss various aspects or details relating to OpenFOAM is specific forum boards.

OpenFOAM file structure

When an OpenFOAM related function or solver is called, it is designed to act upon a set of folders where it expects user defined parameters and to be in predetermined locations. The basic file structure of an OpenFOAM case is shown in Figure 5.

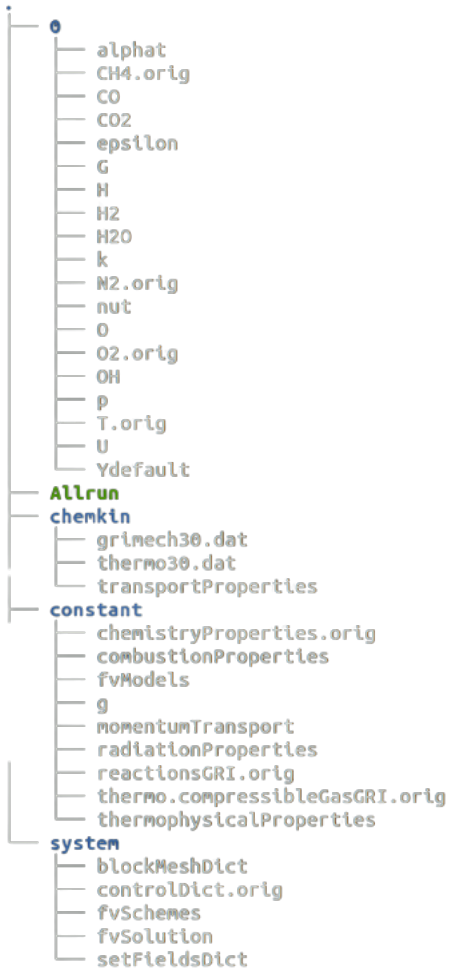


Figure 5: Example of an OpenFOAM folder structure. Generated from the example case for the SandiaD flame in OpenFOAM version 9.

The */0* folder contains the values and type of boundary conditions for the different

fields included. The boundaries are prescribed for the different patches defined in the mesh. The fields will normally be initialised as zero in the domain, but this can be altered with the help of the *setFields* utility to initialise arbitrary regions and values. The fields can also be initialised with a previous solution using the *mapFields* utility. Both are methods which have been utilised.

The *constant* folder contains the different models utilized. The *chemistryProperties* file contains the user defined ODE solver to be used, alongside other relevant parameters. The *combustionProperties* file contains the combustion model, in this case, the 2005 version of the EDC model. The *fvModels* file contains imports to use the radiation library, where the radiation model and parameters are defined in *radiationProperties*. Where in this case, the P1 model is defined alongside the absorption/emissivity parameters used in the grey gas assumption. The *momentumTransport* file contains the turbulence model, in this case the k-ε model, with the standard parameters shown in Table 1. The *thermo* files define the thermodynamic properties to be used in equation (13). This is defined with JANAF coefficients. The values of which are gathered from the *GRI-Mech 3.0,* thermodynamic properties file [40]. The *reactions* file contains the definition for the reactions taking place in OpenFOAM format, with units of [K, Kmol, J]. The methanol reaction used is based on the work presented by Held and Dryer (1998) [41], and implemented in the OpenFOAM reaction format.

The *system* folder contains definitions for the solution procedure and different models used. The vertices and grid spacing for the mesh is defined in *blockMeshDict* for use with the *blockMesh* utility, in addition mesh extrusion definitions are defined in *extrudeMeshDict*. Both of with have been used to create the wedge shape mesh seen in Figure 7. The *fvSolution* defines the fields to be solved and how, including tolerances and pre-conditioners. The *fvSchemes* contains the definition for the discretisation schemes used for time, gradient, divergence, Laplacian, and interpolation schemes. In the *controlDict* file definitions for runtime control and data writing intervals are defined. Of note in this dictionary is the ability to define runtime modifiable, which allows parameters and even discretisation schemes to be changed while the solver is running. In addition this folder contains definitions for *fvModels* applied, this includes *probe* definitions and *fvConstraints*. Both of which have been used. A full list of compatible options can be found by passing either *-listFvModels* or *-listFvConstraints* as arguments to the solver.

ParaView

Paraview is an open source utility to process scientific data built on the VTK libraries. Paraview has been used in this thesis to process the files generated by OpenFOAM and create a graphical representation of the data. Version 5.60 has been used as it is pre-packaged with version 9 of OpenFOAM. The pre-packaged version includes reader plugins specifically designed for the OpenFOAM workflow, such as viewing the *blockMeshDict* file before generating the mesh and viewing a decomposed case. This is invoked by using the *-builtin* argument from the command line. This tool has been used to process the results from OpenFOAM presented.

OpenFOAM Utilities

OpenFOAM includes many smaller utilities that improve the workflow, such as *foamInfo* and *foamSearch* to gather specific information on parts of the code. Some notable ones that have been used are mentioned. The *chemkinToFoam* utility has been used to convert the reaction written in chemkin format to an OpenFOAM supported one. The units used by OpenFOAM differ from the standard chemkin units, that being *kmol* instead of *mol* and *Joule* instead of *calories*.

The *decomposePar / reconstructPar* utility has been used to decompose and recompose the computational domain for running in parallel.

3.2 Geometry

The geometry of the case is similar to the experimental setup by Weckman and Strong in their 'Experimental investigation of the turbulence structure of medium-scale methanol pool fires' 1996 [1]. Differing in that the domain simulated handles the inlet closer to the ground. The geometry consists of a 30.5 cm diameter D inlet, where half of this diameter is a part of the axisymmetrical computational domain, which was extruded at an angle of 3°. The bottom of the domain is modelled as a solid wall. The sides and top of the domain are modelled as an open atmosphere. A sketch of the outline can be seen in Figure 6.

The domain is equal to 9 pool diameters in height and diameter. Out of which, an axisymmetrical wedge is created and used for the computational domain. This is using the assumption that the problem can be modelled as axisymmetrical due to RANS mean values being simulated.

Meshing

OpenFOAM supports meshing with two included tools, *blockMesh* and *snappyHexMesh*. In addition to having built-in mesh converters from other software packages like Ansys fluent and STAR-CCM. The first built-in tool *snappyHexMesh* is based on a hex dominant algorithm that creates unstructured meshes with split hex and hex cells, with the main use case being creating meshes around 3D geometries in the STL format. The second built-in tool *blockMesh* is the most basic mesh generator and is suitable for simple geometries, where it creates high quality meshes. The workflow consists of defining special vertices and creating blocks with different grading anchored in the vertices. Boundary patches are defined in a similar manner with normal vector direction based on the order of the vertices definition. This information is defined in the *blockMeshDict* file in the systems directory. The definitions used to generate the mesh can be found in Appendix D. The scope of the dictionary file rapidly increases for complex geometries, and as such, more appropriate tools should be used. A preview of the definitions in the *blockMeshDict* file can be viewed without mesh generation using the visualisation tool Paraview in conjunction with the *PVfoamreader* plugin when its called with the *-block* argument.

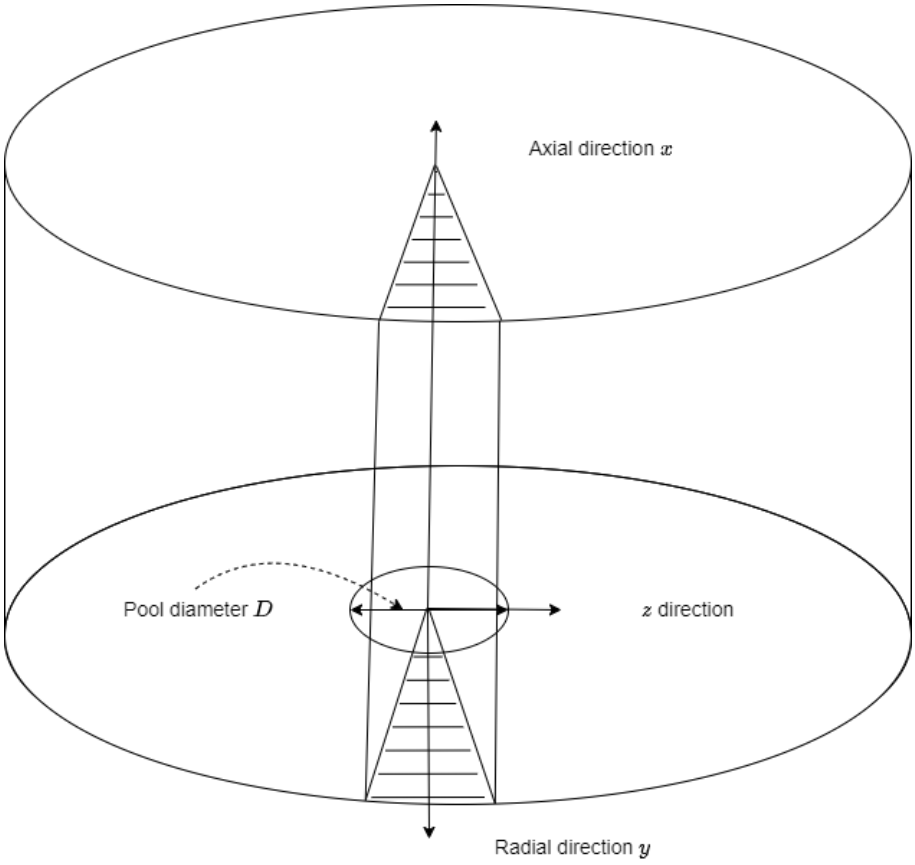


Figure 6: Sketch of the domain

After initial mesh generation is complete, OpenFOAM supports an array of different mesh manipulation tools, such as mesh rotation and topology definition. Other notable mesh manipulation tools are *extrudeMesh*, which extrudes the mesh based on boundary patches, with parameters store in the *extrudeMeshDict* file. This has been used in conjunction with *blockMesh* to generate the wedge-shaped mesh. The code used with *extrudeMesh* can be found in Appendix E. A full overview of the different mesh generation and manipulation tools featured can be found in *OpenFOAM v9 user guide* [38].

Mesh resolution

An initial estimation of the required mesh resolution to adequately resolve the fire plume was taken from the NIST handbook [42]. This value is not directly transferable but should give a good indication of the required mesh resolution. By defining the characteristic diameter D^* of the pool;

$$D^* = \left(\frac{\dot{Q}}{\rho_\infty c_p T_\infty \sqrt{g}} \right)^{\frac{2}{5}} \tag{47}$$

Where $\dot{Q}$ is the heat release rate of the pool fire, in this case 24.6 kW as reported by Weckman and Strong [1]. ρ_∞, c_p and T_∞ are the density, specific heat capacity and temperature of the ambient air and g is the gravitational constant. Using the values of the case gives $D^* \approx 0.22$. By introducing the dimensionless plume resolution index, $D^*/\delta x$, where δx is the nominal cell diameter. Using 110 total cells in the radial direction across the domain of 1.3725 m gives an average cell length of 1.25 cm. This, in turn, gives a plume resolution index of 17.5. Comparing this to the value given for somewhat similar case setups like the McCaffrey Plume summarised in *FSD Technical Reference Guide v4* where the values are in the range of 5-20, and is therefore used as a starting point for the mesh resolution.

The mesh density is also not constant, being biased towards the area of interest near the inlet. The mesh above the inlet is constant in the radial direction and biased with a 1:10 ratio in the axial direction towards the inlet. The entertainment area next to the inlet is biased with a 1:4 ratio towards the inlet in the radial direction, with the same 1:10 ratio in the axial direction. This can be seen in Figure 7 and Figure 8. The mesh biasing influences the plume resolution index by increasing the value near the area of interest and should, therefore, still be valid. Testing of which is further discussed in the next section.

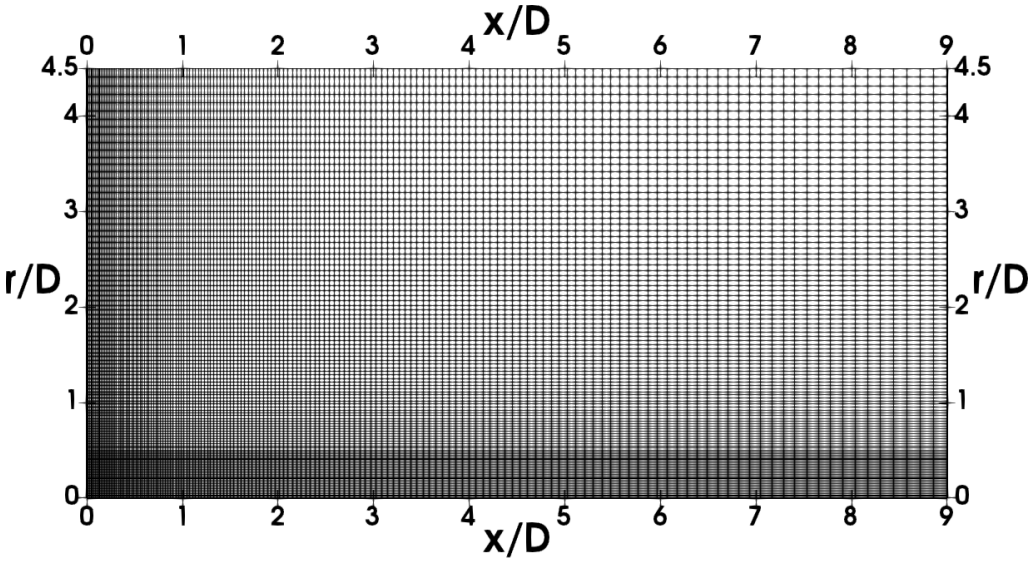


Figure 7: Overview of the computational domain, where the inlet is defined from the origin to half a pool diameter in the radial direction, representing half the pool. D = 30,5 cm

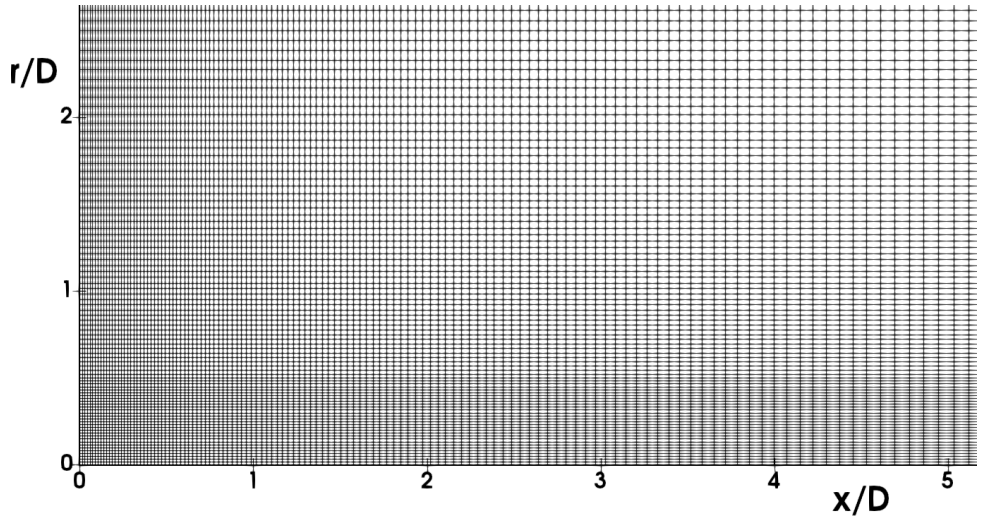


Figure 8: Zoomed in view of the mesh near the inlet. D = 30,5 cm

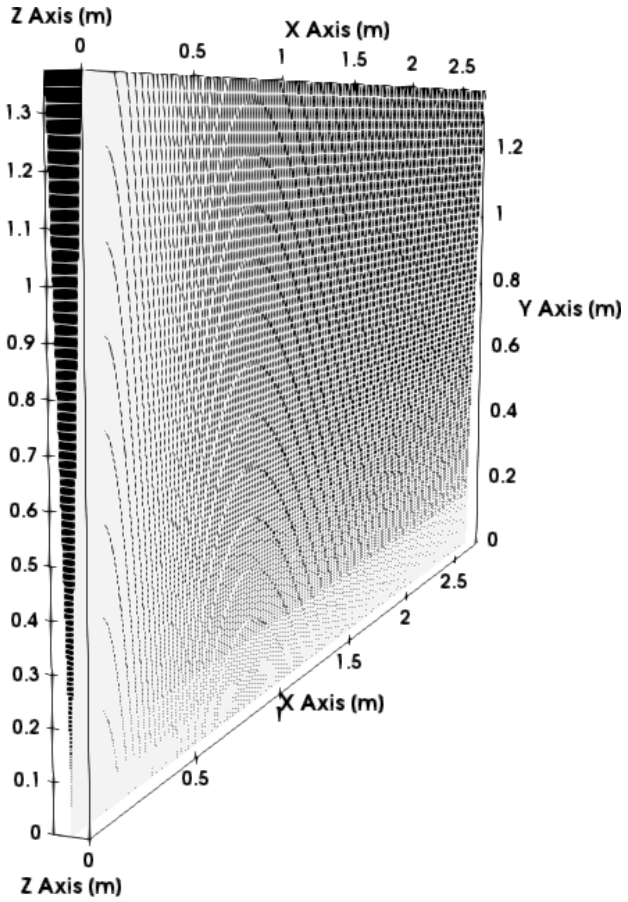
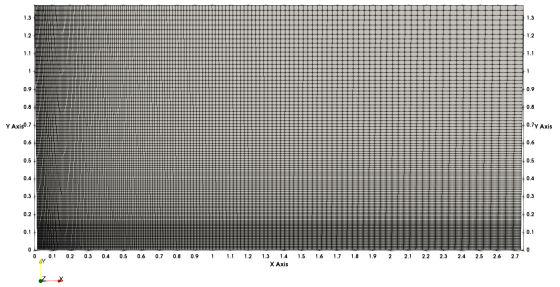


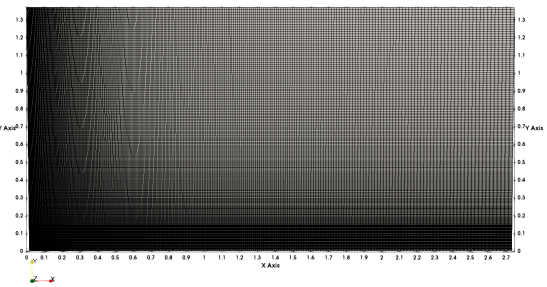
Figure 9: Axisymmetric mesh with an angle of 3°, 1 cell thick.

Mesh refinement

The resolution of the mesh and its general quality is of high importance for both accuracy and stability, and as such, a mesh refinement study has been conducted. To adequately capture the physical phenomena happening and handle the step gradients introduced by turbulent combustion. The goal of the refinement is to determine the grid spacing where further refinement would not result in any numerical accuracy gain, and the solution would be unaffected. This was conducted by constructing a case with infinitely fast chemistry and refining the mesh. The mesh was refined with a factor of 1.5 and 2 in both radial and axial directions. The result of which provided no difference to the physical quantities and, as such, was kept at its initial resolution. A visual illustrating this refinement can be seen in Figure 10b, where the original mesh resolution can be seen in Figure 10a.



(a) Mesh with initial grid spacing.



(b) Mesh refined by double the amount of cells in the radial and axial directions.

3.3 OpenFOAM solver

OpenFOAM provides different solvers to simulate reacting flows in their official release, in addition to highly specialised solvers available from the research community or commercial sectors. The two most general solvers with the most use, as found in the literature study, are the *fireFoam* and *reactingFoam* solvers. The *fireFoam* solver was a joint development between FM Global and CFD-direct, based on LES simulations of industrial-sized fires. This solver provides the most additional features and includes the handling of buoyant terms. This solver has also been extensively validated in previous work [43][44]. This is, however, with LES simulations with very limited chemistry modelling, and little to no documentation exists on running RANS simulations or using the EDC combustion model. This is, however, stated as supported in the documentation. This solver has seen a partial rework with version 9 of OpenFOAM, including changing the name to *buoyantReactingFoam*, and from here on, referred to as *buoyantReactingFoam* to avoid confusion.

The other solver relevant for the case is *reactingFoam*. This solver handles detailed chemistry well as some example documentation and validation has shown [45]. This solver has some examples available for running with detailed chemistry and using the EDC combustion models. It should be noted that these example cases are mostly premixed and include flue gas far above the auto-igniting temperature for the fuel.

And for preliminary testing provided difficulties in sustaining the combustion for a non-premixed buoyant case. In addition this solver handles buoyant forces poorly as noted in previous testing and by authors such as Trefall [46]. This is also evident from the example cases provided in OpenFOAM, where most of the work done has been examining heavily momentum driven flows. One way to examine its applicability to buoyancy driven flow is the dimensionless Froude number which is a ratio between momentum forces and gravitational forces;

$$Fr = \frac{u_{\text{fuel},0}}{\sqrt{gD}} = \frac{\text{momentum forces}}{\text{gravity forces}} \tag{48}$$

Where $u_{\text{fuel},0}$ is the fuel inlet velocity, g is the gravitational constant, and D is a characteristic length scale, which is taken as the hydraulic depth. The length scale D is taken as the cross sectional area of the flow divided by the width of the cross section. Values of the Froude number below unity indicate buoyant forces are dominating, and above unity indicates momentum dominated flow. A rough estimation of the Froude number for the case setup puts it in the order of 10^{-2}, and as such, it is heavily buoyancy driven. An overview of the Froude number of case setups utilizing both the *reactingFoam* and *buoyantReactingFoam* solver is presented by Paulasalo [47]. Where it is found that flows using the *reactingFoam* solver have Froude number in the magnitude of 10^{1}, while flows in the same range as this case setup utilises the *buoyantReactingFoam* solver.

And as such, the *buoyantReactingFoam* solver is the most applicable solver in modelling the physical phenomena.

3.4 Modeling parameters

The values given for the constant feed rate of methanol in Weckman and Strong (1996) [1] were 1.35 cm^3/s for a total heat release rate of 24.6 kW. This is in liquid form. Given the density of methanol as 1.202 kg/m^3 at its boiling point of 338 K, with the burner diameter of 30.5 cm. Applying a mass balance gives an inlet velocity of approximately 0.0108 m/s. The modelling of the pan lip described in the experimental setup by Weckman and Strong is neglected, as this would introduce a large amount of complexity. This assumption has also been assumed by others [6] [21] for the same experimental setup.

An overview of the models used in OpenFOAM can be seen in Table 2.

3.5 Chemistry

The liquid phase of the pool is simplified to a constant influx of gas at the boiling point temperature of methanol, as mentioned in the previous section. And as such, the only phase that needs to be considered when modelling the chemistry is the gaseous phase.

Table 2: Modeling parameters for OpenFOAM

Type	Name	Description
Solver	*buoyant ReactingFoam*	Using RANS momentum model
Equation of state	*perfectGas*	Ideal gas model in OpenFOAM
Thermo physical model	*hePsiThermo*	General model based on enthalpy or internal energy and compressibility
Transport	*sutherland Transport*	Sutherland's formula for transport properties based on temperature, viscosity
Thermo physical properties	*janafThermo*	Calculate C_p from JANAF coefficients. Values of which are gathered from GRI-MECH termo [40]
Mixture properties	*multi Component Mixture*	OpenFOAM model for multi specie reactions
Reaction ODE-solver	*seulex*	Based on the linearly implicit Euler method with step size control
Radiation	*P1*	Based on expanding first order spherical harmonics
Soot	*none*	neglected due to methanol burning almost without soot
Combustion model	*EDC*	version from 2005 [30][38]
Turbulence model	*k-ϵ*	With coefficients shown in table (1)

The thermodynamic values for the different species are based on the data provided by *GRI-Mech 3.0,* [40].

Chemical reaction mechanism

The chemical reaction mechanism models used to describe the oxidation of hydrocarbons range from the simplest 1-equation models to hundreds of species and reactions. The model was chosen on the basis of accuracy with regard to flame and turbulence quantities. Therefore the large and complex models that specialise in NOx could be discarded. By accounting for the system of equations stemming from the chemical mechanisms tendency to be very stiff and computationally expensive, a simpler model was chosen.

The model used is presented by Held and Dryer (1998) [41]. This model includes 22 species and 97 reactions and has been validated in the temperature and pressure ranges relevant to the case setup. Another simpler mechanism by Mosisa Wako *et al.* (2021) [48] using 19 reactions has been tested, as this model has been implemented in CFD. A further discussion of which can be found in Chapter 4.

Methanol combustion

Methanol (CH3OH) has similar properties to other alcohols in combustion. Methanol fires are almost imperceptible when viewed and give off a slight shimmer of blue. This is mainly due to it burning extremely clean, and soot formation is almost non-existent [6]. Methanol also has a boiling point above ambient temperature under normal atmospheric pressure, at 337.85 K [49]. This results in methanol fires often having multiple phases included, such as a liquid phase with evaporation.

Ignition

The ambient conditions prescribed in the modelling case are not sufficient for the methanol fuel to auto-ignite. Therefore, a source of ignition is needed to start combustion. This numerical ignition is achieved with different methods. One method used is to alter the ambient conditions of the boundaries and inlet to above the auto-ignition point of methanol, which under normal atmospheric conditions is found to be approximately 870K [49]. In this case, an inlet temperature of 1000K is prescribed, and the atmospheric boundary conditions are changed to 600K. Since these conditions are above the auto-ignition point, the fuel and oxidiser readily react, and combustion happens. This solution is then allowed to converge, and the different fields are then mapped back to a case with the original boundary conditions. This is done with the hope of introducing enough energy into the domain with a distribution like the one expected from the original boundary conditions to prevent the extinction of the flame and sustain further combustion. In addition, both of the field initialisation methods mentioned earlier are used to achieve ignition. The setFields utility can be used by initialising a small field near the inlet to a very high temperature for only the initial time step. The mapFields utility is also used as a starting point for some of the simulations where the cold non-reacting flow is allowed to develop before being mapped to a new case running with chemistry turned on. This is done to significantly speed up the time before convergence.

3.6 Boundary Conditions

The boundary conditions used for the case setup can be seen in Table 3.

Where the abbreviations are the following;

- zG - *zeroGradient* - Applies a zero gradient condition.
- fFP - *fixedFluxPressure* - Sets pressure gradient such that face flux is equal to the velocity bondary condition.
- pIOV - *pressureInletOutletVelocity* - Applies a zero gradient for the outflow and assigns a value for the inflow based on the face flux.
- prghTHP - *prghTotalHydrostaticPressure* - Static pressure condition such that $p_rgh = ph_rgh - 0.5\rho|U|^2$, Where $ph_rgh = \rho g\,(h - h_{ref})$ and $hRef = 0$.

Table 3: Overview of the different boundary conditions used in the case setup

	Outlet	**Sides**	**Base**	**Inlet**
α_t	zG	zG	zG	zG
ν_t	calculated	calculated	zG	calculated
ϵ	inletOutlet 10^{-4}	inletOutlet 10^{-4}	zG	fixedValue 10^{-4}
k	inletOutlet $3.94 * 10^{-6}$	inletOutlet $3.94 * 10^{-6}$	zG	fixedValue $3.94 * 10^{-6}$
U	pIOV	pIOV	noSlip	(0.0108 0 0)
T	inletOutlet 300 K	inletOutlet 300 K	zG	uniform 338
p	calculated	calculated	calculated	calculated
p_rgh	prghTHP	prghTHP	fFP	fFP
ph_rgh	fixedValue	fFP	fFP	fFP
G	MR	MR	MR	MR
CH3OH	inletOutlet 0	inletOutlet 0	zG	uniform 1.0
O2	inletOutlet 0.23301	inletOutlet 0.23301	zG	uniform 0
N2	inletOutlet 0.76699	inletOutlet 0.76699	zG	uniform 0
Ydefault	uniform 0	uniform 0	zG	uniform 0

- MR - Marshak Radiation boundary condition

The *inletOutlet* boundary condition acts as a outlet for flow out of the domain and as an inlet with a defined value for flow into the domain. The values for k are found from the turbulent intensity I relation $k = 1.5(UI)^2$, where the intensity for pool fires are in the range of 5% to 15% [21]. Using 15% intensity with the inlet velocity gives a value of $3.94 \cdot 10^{-6}$. The front and back all have the boundary condition of wedge.

It should be noted that the boundaries for ph_rgh differ slightly for the outlet and sides. This is done for stability. [50]

3.7 Running in Parallel

To be able to efficiently run the numerical simulations utilising the computational load need to be able to be spread across multiple processing cores. A prerequisite for this is decomposing the computational domain. OpenFOAM has built-in support for this type of scaling computational load and is found to have excellent scaling across nodes [51]. Running the computation in parallel comes with some inherent overhead as the communication between the decomposed domains has a computational cost.

The way domains can be decomposed differs from the simplest partitions based on cell counts to more complicated algorithms accounting for boundaries and mesh

density. The computational requirements between different areas in the domain often differ drastically, and the more naive algorithms have large inefficiencies. A more advanced decomposition algorithm implemented in OpenFOAM is the Scotch algorithm, based on. This seeks to minimise the interface between cells and provides generally improved performance over the geometric approach. This method has been used for all the case setups. An example of the domain decomposed with the Scotch algorithm can be seen in Figure 11.

For dividing the job across multiple processor cores, the message protocol interface (MPI) protocol has been used, as this is implemented in OpenFOAM with openMPI. Other proprietary alternatives exist, such as Intel's MPI library, which has shown to be slower in OpenFOAM [52]. The domain was decomposed into 10 subdomains and run without hyper-threading enabled.

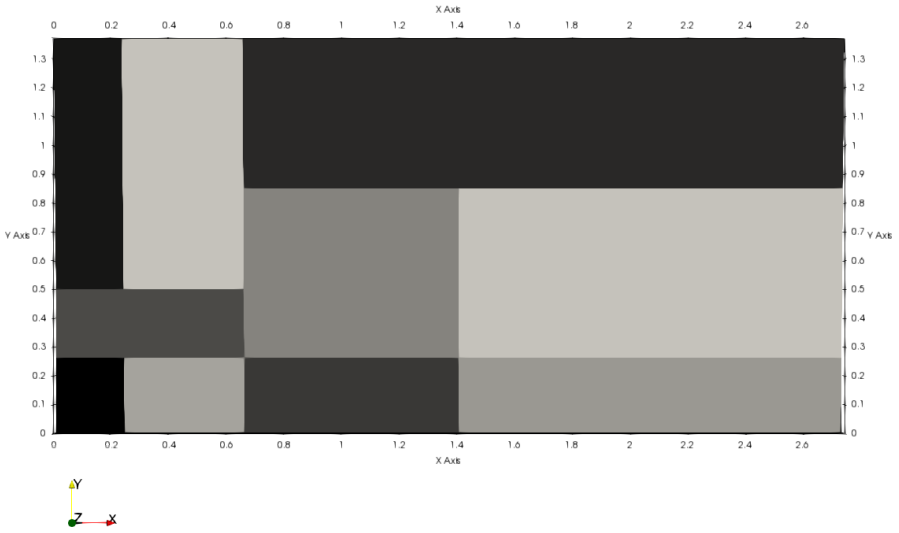


Figure 11: domain decomposed into 10 sub-domains, using the Scotch algorithm

Where the different colours represent a different decomposed domain, the boundary conditions between the sub-domains are evaluated between each node, and transfer across the boundaries is handled with the MPI protocol.

Determining Convergence

Convergence can be difficult to determine accurately, as no rigid definition exists. The approach taken included monitoring the residual in addition to physical quantities and determining convergences based on when all the values were stable for a given length of time. The residuals were monitored using the built-in *foamMonitor* tool in addition to defining the residuals in the case files. OpenFOAM supports the use of probes as a function object. This is implemented by including the probes library in the *controlDict* file and defining the details in a probes file in the systems directory. The physical quantities were plotted with data from the log file of the solver in addition to the probes that were defined in the setup. This was achieved

with a custom written script utilising *gnuplot*. The different fields of interest are then determined to have converged when there is no significant change in their value or residuals with an increased number of iterations.

4 Results and Discussion

Due to large amounts of problems encountered in producing a working case setup using the *buoyantReactingFoam* solver with the parameters discussed in previous chapters, no satisfactory results have been produced. The case had large problems sustaining combustion and would diverge before convergence when combustion did sustain. And as such, the results presented are partial representations of the full case. This includes results from three simplified cases. In the first case **(A)** the combustion model was changed to infinitely fast (mixed is burned). The second simplified case **(B)** includes elevated inlet temperatures for a given time above the auto-ignition point of methane to sustain the combustion. This case is numerically unstable as it would suddenly diverge after a long period of stable residual and physical values. Therefore it is unlikely that the results of this case represent the converged solution discussed further in its section. And lastly, case **(C)** where combustion extinguished with no changes to the setup. This is presented in the hope that it might prove valuable in further work and as a basis for further discussion in the next sub-sections.

All the plots have been mirrored across the vertical axis when presented as per the axisymmetrical assumption. This is done for easier viewing and comparison.

Infinitely Fast Combustion (A)

The results presented in this section are using the infinitely fast combustion model, referred to as case **A**. This over predicts the flame temperatures drastically and is well above the adiabatic flame temperature of approximately 2222 K at normal conditions. This is highly non-physical and likely hints at something wrong with the case setup beyond the combustion model.

Figure 12 shows the velocity magnitude field for the entire mirrored domain of 9 pool diameters in both directions. The largest magnitude of the velocity vector is 5.5 [m/s] and as such is vastly higher than than the values presented by Sedano *et al.*[21] and Weckman and Strong[1]. The largest velocity magnitudes found by Sedano *et al.* were 3.14 [m/s]. And the largest reported by Weckman and Strong were 2.5 [m/s]. This large discrepancy is likely due to the much higher temperature field, as can be seen in Figure 13 resulting in larger density differences. This, in turn, would cause the flow to speed up. The velocity field seen in Figure 12 is also very thin compared to one presented in Sedano *et al.* [21] [Their Figure 6(b)]. This is likely also a result of the increased velocities.

Figure 13 shows the temperature field for the entire domain. From this, we can see that the highest temperature is 3000 [K] and likely above this as 3000 [K] is a limitation of the thermodynamics file in OpenFOAM. This is above the adiabatic temperature for methanol and likely stem from incorrect boundary definitions or thermodynamic data. Discussed further in the subsection of possible error sources.

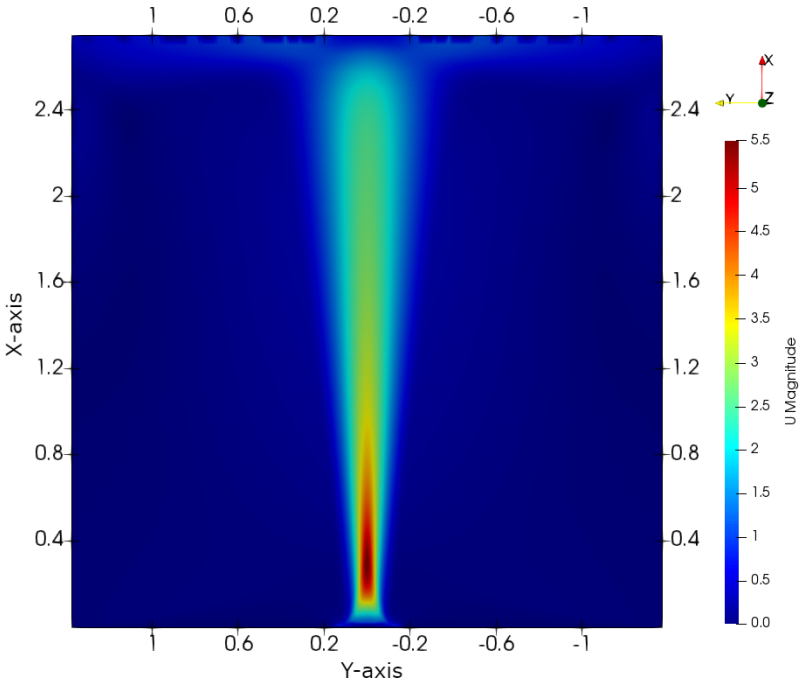


Figure 12: Velocity magnitude field for the entire computational domain of **Case (A)**. With units of [m/s^2] for the velocity magnitude scale, and meters [m] for the length scale. The domain has been mirrored around the centreline for easier comparisons.

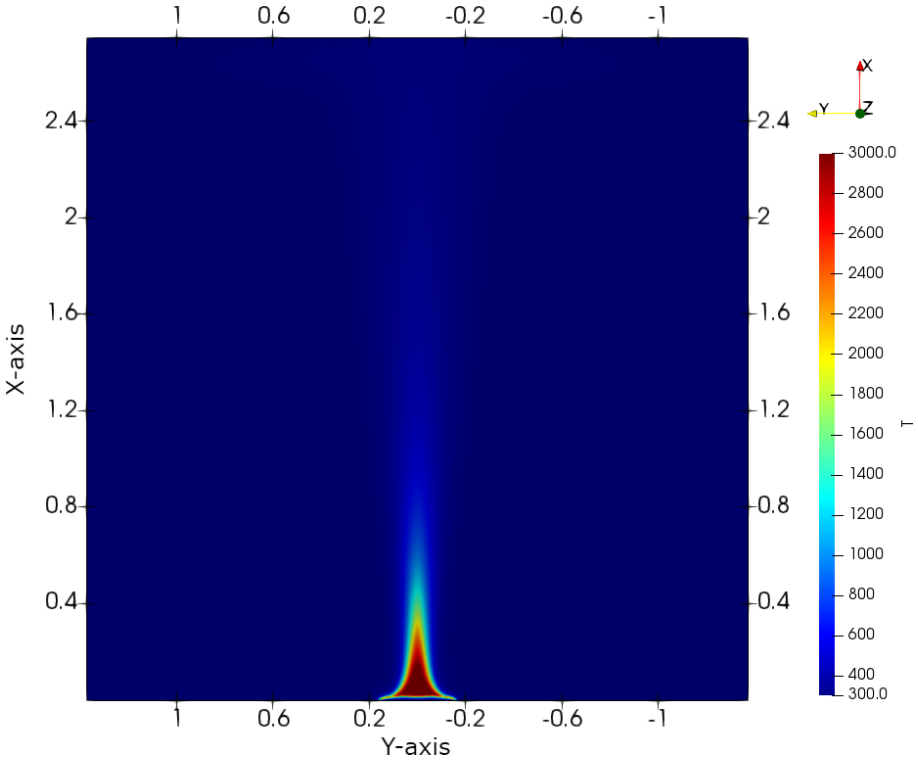


Figure 13: Temperature field for the entire computational domain for case **(A)**. With units of [K] for the temperature scale and meters [m] for the length scale. The domain has been mirrored around the centreline for easier comparisons.

Elevated inlet temperature results (B)

In this section, the results from the case **(B)** are presented. Where the inlet temperatures are elevated to 900 [K] for a given amount of time to start the combustion process, after which the inlet temperature is changed to 338 [K] equal that to the boiling point of methanol, this case is numerically unstable as mentioned in the introduction and unlikely to have reached convergence. Therefore the results presented are likely not accurate but presented as the basis for discussion.

When kept running after an initial quasi-steady period where the physical quantities remained largely unchanged with residuals on the order of 10^{-4}, the case would diverge. This occurred at approximately an order of magnitude more iterations than required to achieve the quasi-converged state. Equivalent to approximately 20 hours of CPU time with the specifications listed in Appendix A. Attempts to stabilise the numerical schemes from a previous time step were taken. This included changing the schemes to more numerically robust ones and reducing the CFL number constraint by order of magnitude. In addition to mapping the quasi-steady solution to a finer grid. Both of these strategies proved unsuccessful as the solution still diverged.

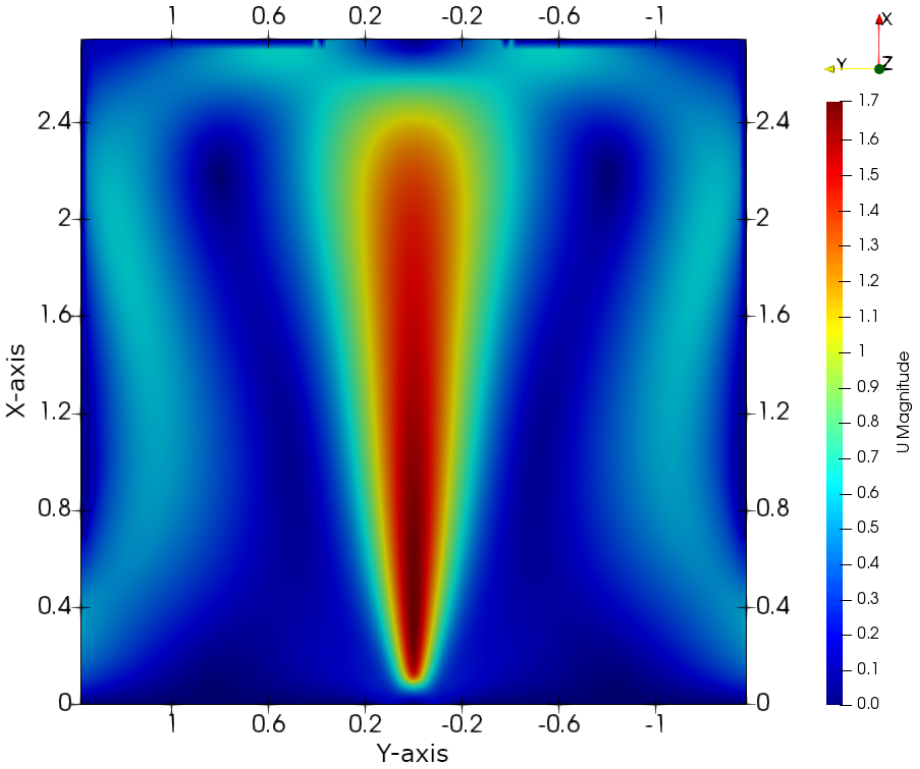


Figure 14: Velocity magnitude field for the entire computational domain of **Case (B)**. With units of [m/s^2] for the velocity magnitude scale, and meters [m] for the length scale. The domain has been mirrored around the centreline for easier comparisons.

In Figure 14 the velocity magnitude field for the entire domain is shown. Where the maximum velocity occurs at the centreline with a magnitude of 1.7 [m/s]. This is significantly less than the values found by Sedano *et al.*[21] where the highest velocities magnitudes are reported as 3.14 [m/s]. The shape of the velocity field

might also indicate an obstruction near the boundaries of the domain. The values of the velocity field seen in conjunction with the temperature give inconsistencies with previously reported values. Where the temperature reported by Sedano *et al.* for the case with the largest velocity magnitude is found to be 1100 [K], significantly less than the 1618 [K] reported here, one should expect the opposite relation between velocity and temperature based of pressure differences from the density gradient. From Weckman and Strong[1] the maximum reported temperatures are approximately 1420 [K] and the velocity magnitude of 2.5 [m/s]. This is also in poor agreement with the results presented here. From the temperature field shown in Figure 15 it is evident that the area of active combustion is very small, and the combustion process happens above the pool in an unexpected way compared to the temperature field reported by Sedano *et al.*[21][Their Figure 6(a)]. The mass fraction for selected species near the inlet, seen in Figure 16c shows this in detail. Where the reaction byproducts are clustered around the area of combustion and likely not transported away in a proper way. The temperature field shown in 15 also shows poor heat transfer down to the inlet which might indicate incorrect handling of thermal transfer or radiation. With the large discrepancies shown, this case setup is likely not modelling the interactions in a correct way.

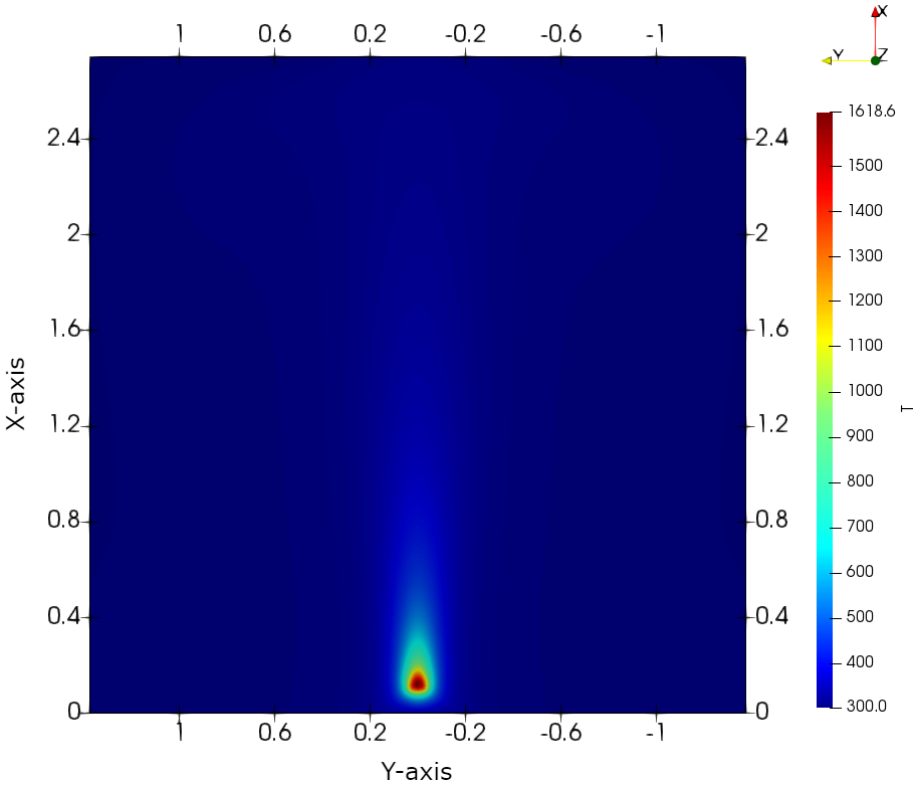
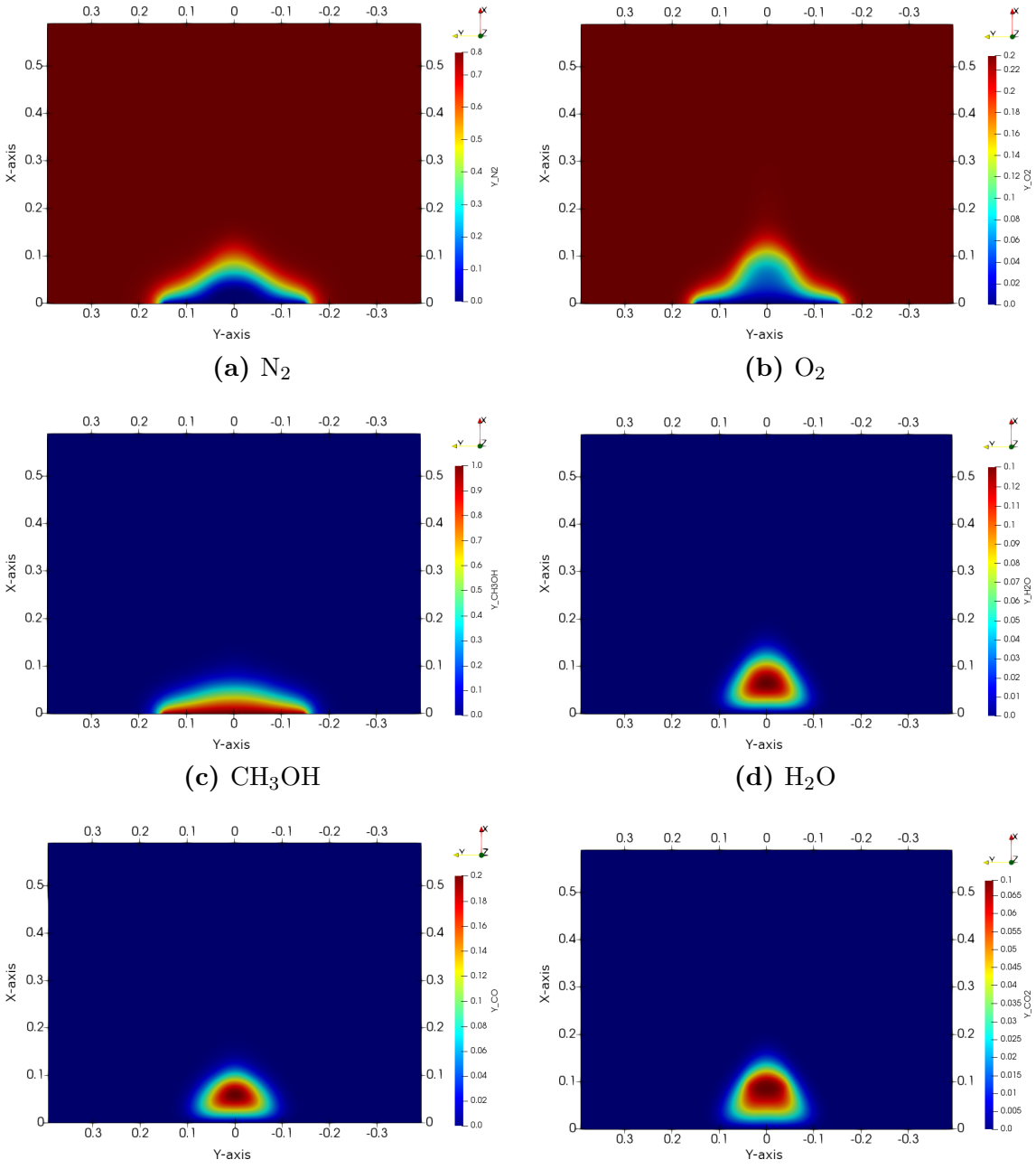


Figure 15: Temperature field for the entire computational domain of **Case (B)**. With units of [K] for the temperature scale and meters [m] for the length scale. The domain has been mirrored around the centreline for easier comparisons.



(a) N_2 **(b)** O_2

(c) CH_3OH **(d)** H_2O

(e) CO **(f)** CO_2

Figure 16: Mass fraction for selected species in case **(B)**. With units of [m] for both axis.

Full case setup with combustion extinction (C)

In this setup, referred to as case **(C)** all the parameters are equivalent to the ones shown in Table 3 and represent the full case setup. This includes the EDC combustion model, with values given by Magnussen (2005) [30], and detailed chemistry based on the reaction mechanism given by Held and Dryer (1998) [41]. The flow was attempted ignited multiple times during the simulation run. This was done by setting the temperature to a high value, between 1000 to 2000 [K] near the burner inlet. This was achieved with the *setFields* utility. This would make the species react, but further combustion did not sustain itself. This case is equivalent to a cold, non-reacting case setup.

The velocity magnitude field is shown in Figure 17 has a maximum value of 0.4 [m/s]. This velocity component is much greater than the inlet velocity of 0.0108 [m/s]. This show that acceleration due to buoyant forces are treated. The shape of the velocity magnitude field is also an indication that buoyant forces are active. Near the inlet of the domain, the characteristic necking behaviour can be seen, with the entertainment of air from the surroundings. The temperature difference in this example is rather small at 38 [K], as can be seen in Figure 18. From where it can also be seen that the temperature drops further up the centreline as expected with cooling and mixing with the ambient air.

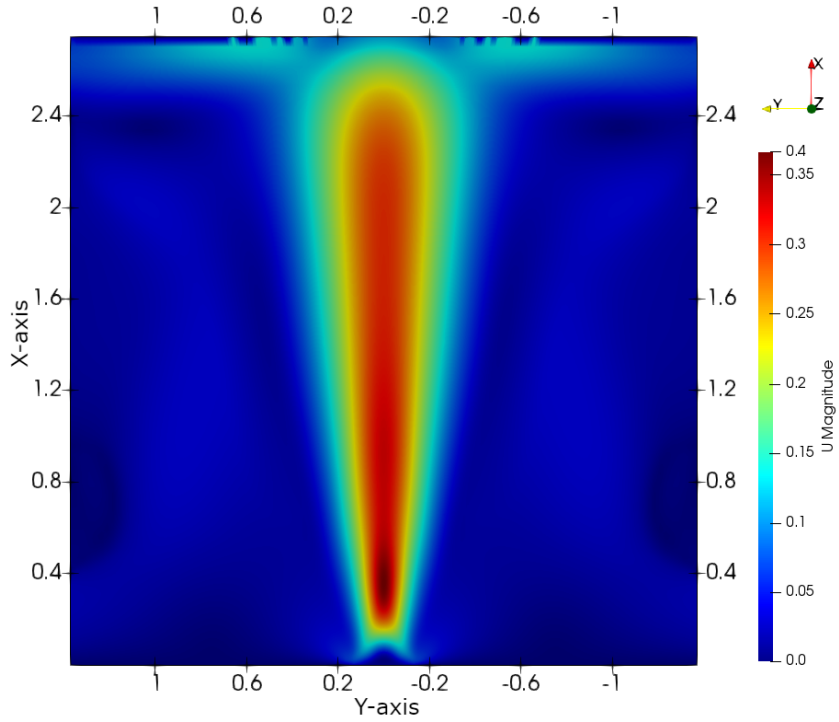


Figure 17: Temperature field for the entire computational domain **Case (C)**. With units of [K] for the temperature scale and meters [m] for the length scale. The domain has been mirrored around the centreline for easier comparisons.

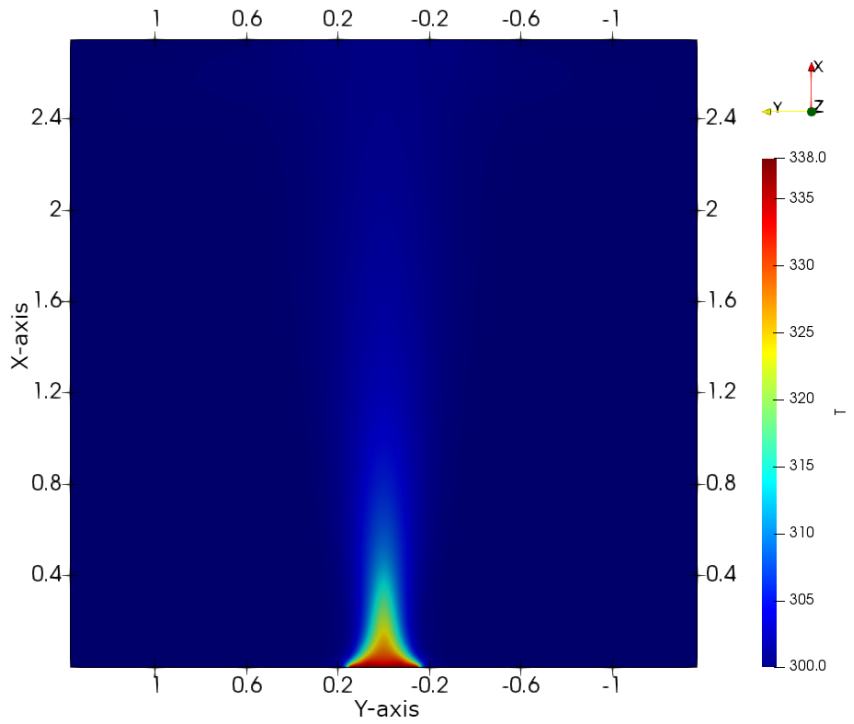


Figure 18: Temperature field for the entire computational domain **Case (C)**. With units of [m/s^2] for the velocity magnitude scale, and meters [m] for the length scale. The domain has been mirrored around the centreline for easier comparisons.

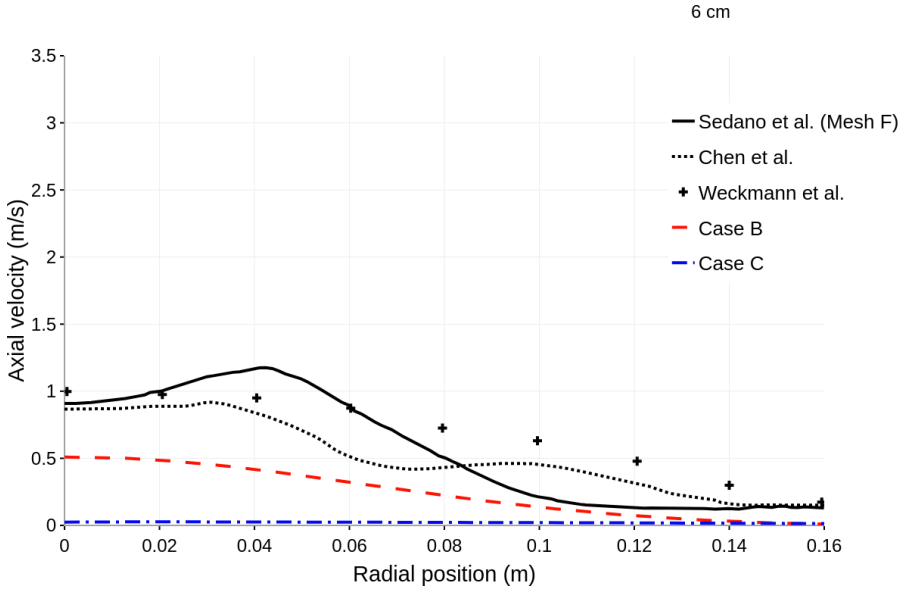
Comparison of the axial velocity

Comparisons of the axial velocity component is shown in Figure 20 where case (**B**) and (**C**) are compared to the numerical data presented by both Chen *et al.* (2014) [6], and the "Mesh F" data presented by Sedano *et al.* (2017) [21]. Both of which utilise the fireFoam solver in OpenFOAM with LES, EDC and infinitely fast chemistry. In addition to data from the experimental work presented by Weckman and Strong (1996) [1]. The setup for both of the numerical studies mentioned are based on the experimental setup by Weckman and Strong[1]. The values for case (**A**) is omitted as the high values of the temperature field, and in turn, the axial velocity values are highly non-physical and would make the other presented data harder to differentiate.

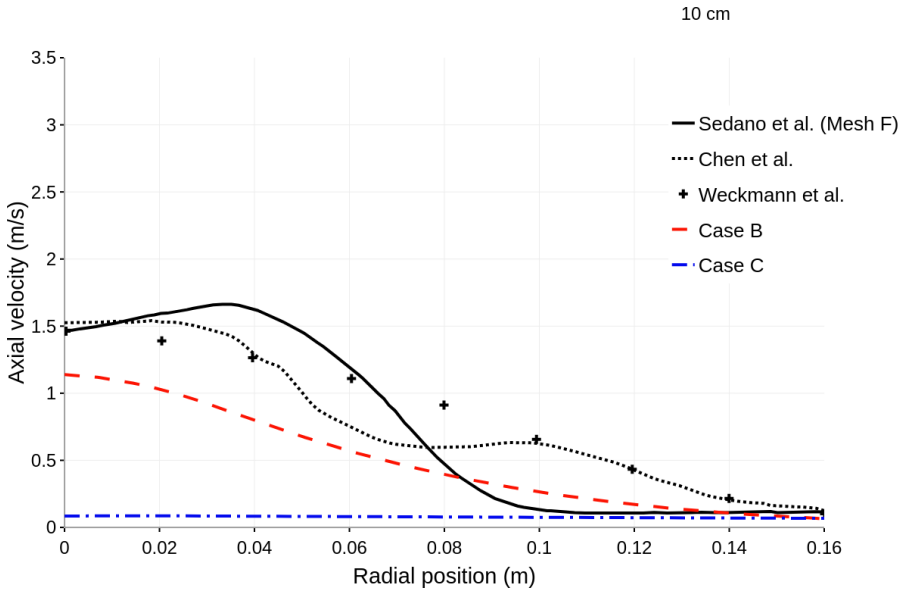
The radial position describes the distance from the centreline of the circular pool acting as the inlet, where the radius is 15.25 [cm]. The measurements are taken at a vertical distance of 6, 10, 18 and 30 [cm] above the pool.

While the numerical data presented by both Chen *et al.* (2014) [6] and Sedano *et al.* (2017) [21] are in generally in agreement with the experimental data, the data is not completely in agreement and some trends can be seen. While the velocities very close to the pool centre are in agreement with the experimental data for the lower heights of 6 and 10 [cm], the general trend is over prediction of the velocities a quarter to halfway out in the pool. While under predicting the axial velocities from halfway out to the rim of the pool. This indicates that the numerical data reflects a much skinnier and more stretched flame than the experimental data. The results

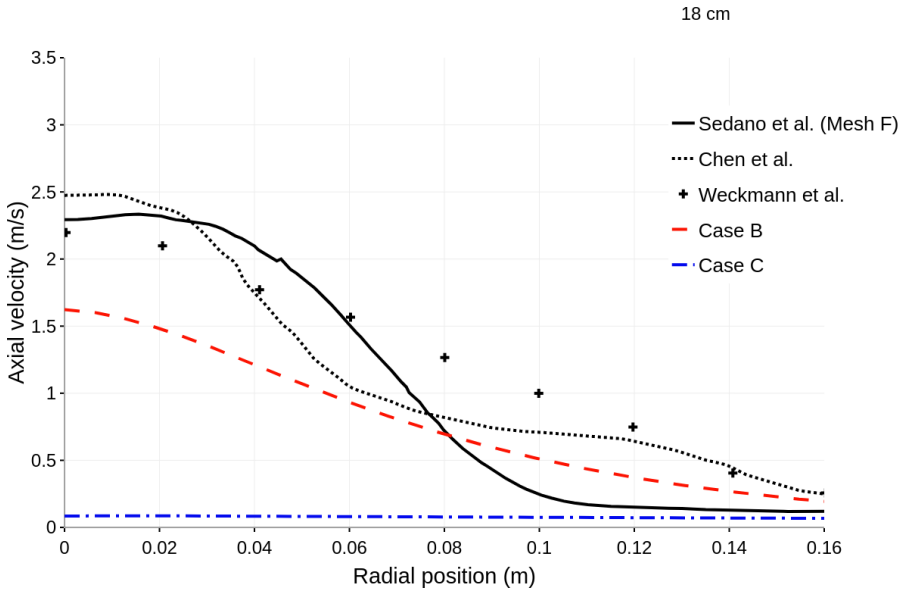
from case (**B**) show a significant underprediction of the velocities near the centre of the pool for all heights, and it is likely that this results in combustion byproducts not being properly transported away and interfering with combustion. The shape of the axial velocity profile for case (**B**) shows a general trend of larger velocities near the centre of the pool, in agreement with the general trend of the other data. The falloff of velocity with increasing distance is also much more gradual than the other data presented, which seen in conjunction with Figure 15 might indicate a significant difference in the shape of the combustion area. The axial velocities shown in case (**C**) are an order of magnitude smaller than the rest due to a very small temperature gradient.



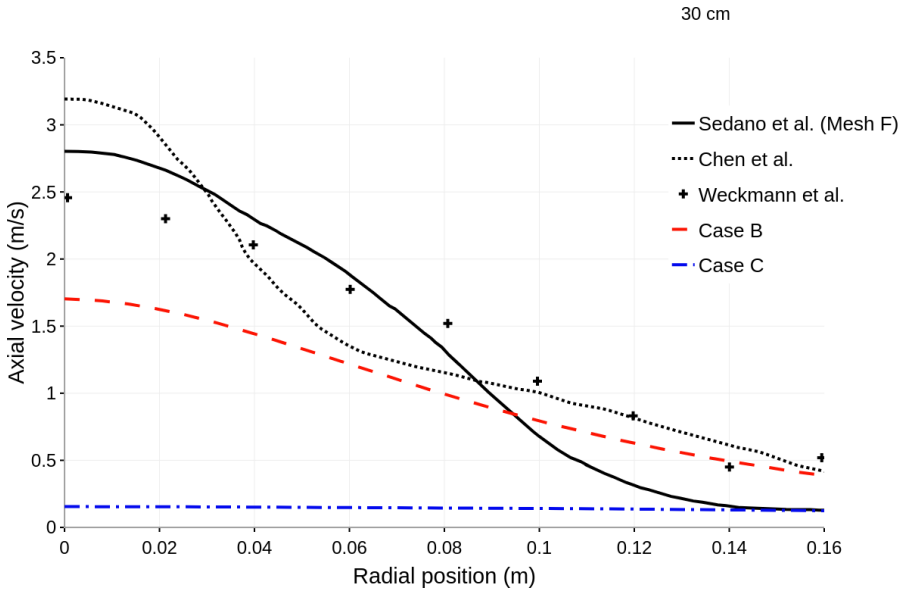
(a) 6 cm above the inlet



(b) 10 cm above the inlet



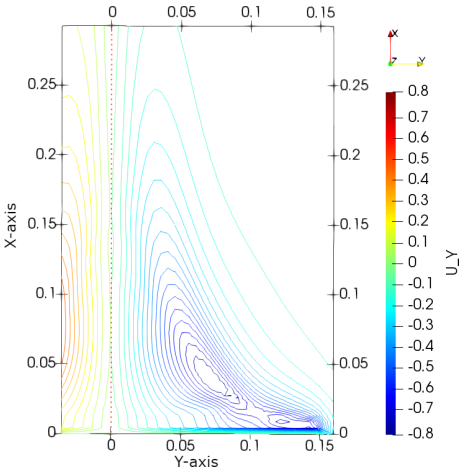
(c) 18 cm above the inlet



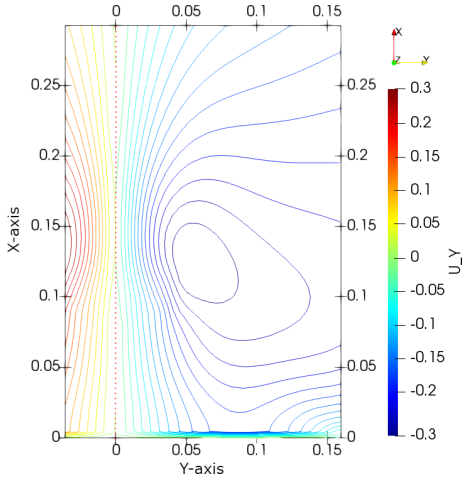
(d) 30 cm above the inlet

Figure 20: Radial position plotted against axial velocity for selected heights above the pool.

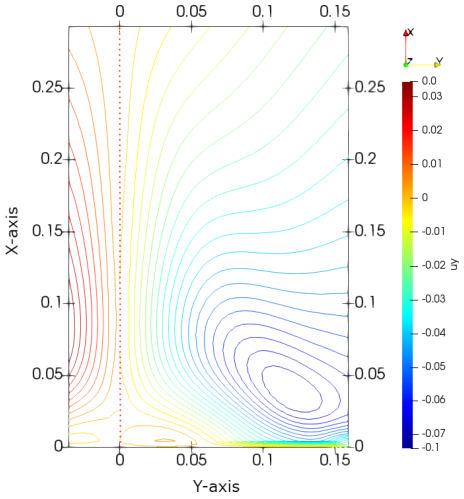
Comparison of the radial velocity



(a) Isovel of radial velocity for case (**A**)



(b) Isovel of radial velocity for case (**B**)



(c) Isovel of radial velocity for case (**C**)

Figure 21: Isovels of the radial velocity for the different case setups, the dotted red line denotes the centreline. The axis has the unit of meters [m] and the radial velocity has the unit of [m/s^2]. The area shown has the same dimensions as Weckman and Strong (1996) [1] for comparison.

Comparing the isovels of case (**A**), (**B**) and (**C**) to data presented by Weckman and Strong (1996) [1] and Chen *et al.* (2014) [6] show poor agreement. Comparing case (**B**) to the data by Chen *et al.*[6][Their Figure 15b] shows the values of the radial velocity to be in approximately the same range, but large differences in shape and direction can be seen where the area of largest radial velocity in case (**B**) is located significantly higher up than the data presented by Chen *et al.* This is likely due to the similarities in temperature more than the shape of the flame. Comparing to the data presented by Weckman and Strong[Their Figure 9] shows similar trends as the comparison to Chen *et al.*, where the area of high radial velocity is moved significantly up.

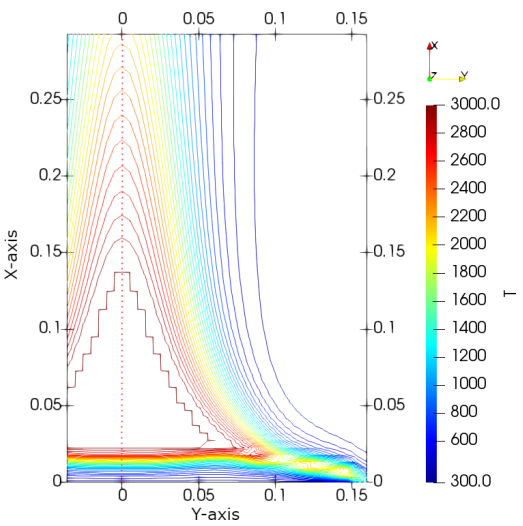
Comparison of the temperature field

The temperature fields for all the cases are shown in Figure 22. In case (**A**), a clearer picture of the non-physical nature of the temperature field can be seen by the jagged line with a temperature of 3000 [K]. This is limited by the thermodynamics file in OpenFOAM and would likely climb higher if allowed. This behaviour indicates incorrect handling of the energy from combustion, although the correct heat of combustion at approximately 22.7 [MJ/kg][49] is output from the solver. Comparing case (**B**) to Weckman and Strong (1996) [1][Their Figure 3] shows poor agreement, where the area of high heat is found to be much larger in the experimental data. The shape of the temperature field shown for case (**B**) looks highly suspect as it resembles a radiating body more than the profile one would expect of a flame. In addition to a lack of iso-lines near the inlet. Comparisons to Sedano *et al.* (2017) [21] shows poor agreement, with the flame here being significantly skinnier and concentrated near the centreline. The Temperature field shown for case (**C**) shows the expected behaviour of a buoyancy driven flow with temperature gradients. This again indicates that the buoyant forces are simulated.

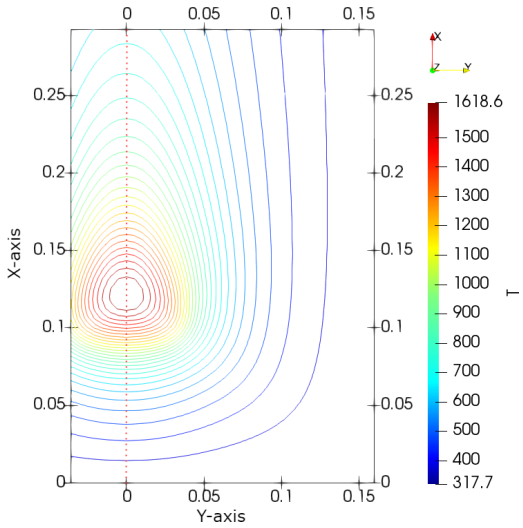
Comparison of the turbulent kinetic energy field

Figure 23 shows the of the turbulent kinetic energy k, illustrated as isocontours. Comparing case (**A**) and (**B**) to Weckman and Strong (1996) [1][Their Figure 17] shows both plots to have values of k approximately in the same order of magnitude as the experimental data. The largest values of k found by Weckman and Strong is near the centreline above the pool, the same can be found for both case (**A**) and (**B**). Where case (**B**) has the area of high turbulent kinetic energy at a lower height. This is in agreement with the axial velocity data shown in Figure 20, as the areas of high velocity should see a relative higher turbulence. Comparing to the results presented by Chen *et al.* (2014) [6][Their Figure 23] shows similar results as the comparison to Weckman and Strong, differing in that the results presented by Chen *et al.* have the region of high turbulent kinetic energy moved further up with slightly larger values. It should be mentioned that in Chen *et al.* the isolines of k are more well defined and orderly, likely a byproduct of the numerical computation. The turbulent kinetic energy for case (**C**) shows turbulent interactions at the edge of the pool, on the interface between the ambient and inlet.

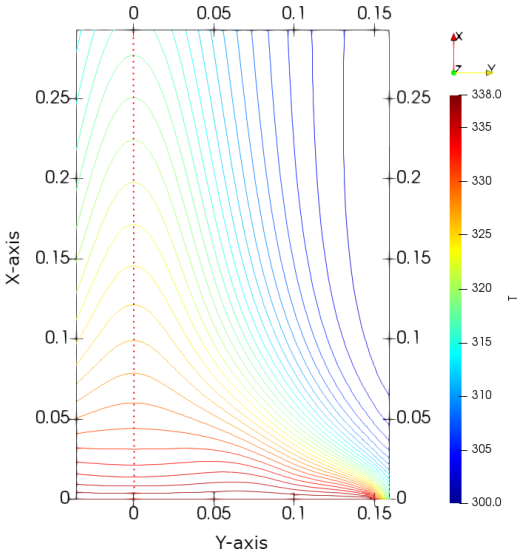
In general are the results presented in poor agreement with the experimental and numerical results presented by Chen *et al.* and Weckman and Strong. Likely a byproduct of the large differences in temperature and velocity. The turbulent kinetic energy is an important part of achieving mixing in the EDC model. This could be a part of the reason for the non-sustaining combustion experienced in the case setup.



(a) Temperature field for case (**A**)

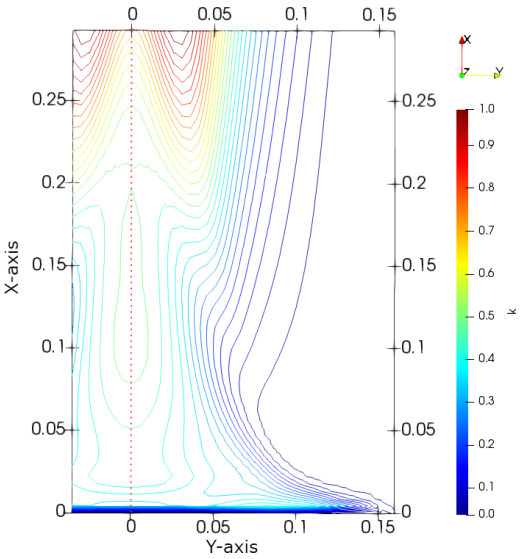


(b) Temperature field for case (**B**)

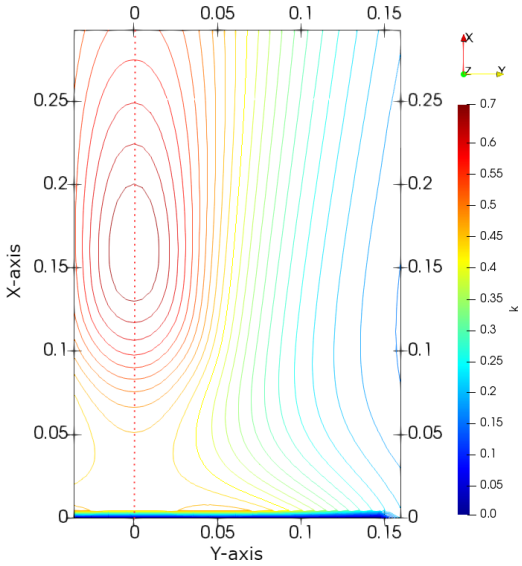


(c) Temperature field for case (**C**)

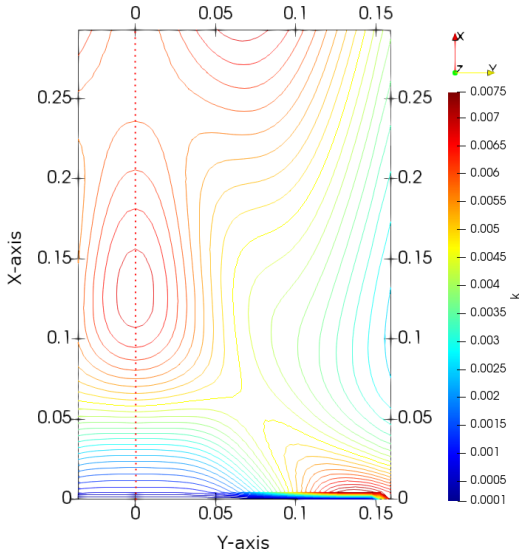
Figure 22: Temperature field for the different case setups, shown as isocontours. The dotted red line denotes the centreline. The axis has the unit of meters [m], and the temperature has the unit of [K]. The area shown has the same dimensions as Weckman and Strong (1996) [1] for comparison.



(a) Turbulent kinetic energy for case (**A**)



(b) Turbulent kinetic energy for case (**B**)



(c) Turbulent kinetic energy for case (**C**)

Figure 23: Turbulent kinetic energy for the different case setups, shown as isocontours. The dotted red line denotes the centreline. The axis has the unit of meters [m], and the turbulent kinetic energy has the unit of $[m^2/s^2]$. The area shown has the same dimensions as Weckman and Strong (1996) [1] for comparison.

Possible Sources of Error

The results from the different case setups shows a large discrepancy from both the experimental results by Weckman and Strong (1996) [1] and the other numerical investigations using LES by Sedano *et al.* (2017) [21] and Wang *et al.* (2011) [53]. This is likely due to incorrect modelling in the case setup or the solver of choice. Producing a working case setup with correct treatment of buoyancy, combustion model, detailed chemistry and utilising RANS proved to be a difficult task in OpenFOAM.

Producing a working case setup has included iterative work. In this section, some possible sources of errors are listed, in addition to some of the steps taken to isolate and test different aspects of the model. Due to the nature of turbulent combustion being a highly coupled and complex problem to model, the approach involved reducing the number of variables to test specific parts of the model.

To test if there was something elementary wrong with the software or hardware used, some known good validation cases were run. This included the example cases for the Sandia-D[45] flame using the *reactingFoam* solver, including detailed chemistry, EDC for combustion and being a heavily momentum driven flow. The other case run was a LES based pool fire example, utilising infinitely fast chemistry and the *buoyantReactingFoam* solver. Both the cases produced results in agreement with previous works. The source of the discrepancy is likely in the numerical solver or the case setup.

To test different aspects of the geometry setup for the case and if the boundaries were influencing the flow, the domain was extended beyond its initial values of 9 times the pool diameter in increments up to 20 times the pool diameter. This was to ensure that the atmosphere boundary was correct and didn't interfere with the flame development. This produces some minor improvement for the initial extension but not an appreciable difference overall. The domain was kept at 9 times the pool diameter D, as this was among the upper end of what was reported in the literature, such as [47]. The assumption of axisymmetrical should be valid due to simulating mean quantities and the geometric layout of the problem. The numerical solvers in OpenFOAM are inherently 3D, and 2D cases are treated with the boundary condition *empty* to not evaluate quantities at those faces. The wedge boundary condition used is evaluated similarly to a symmetrical boundary. To test the validity of the axisymmetrical assumption, the case was recreated in a fully 3D cylinder with the same dimensions. This resulted in no difference from the axisymmetrical case setup. A 2D case was also created as some instabilities and high residuals for velocities in the third dimension, and this also resulted in no difference from the axisymmetrical setup.

Another aspect of the geometry that could influence the result is the mesh. As the current resolution could be inadequate to fully capture the steep gradients or some length scale relating to the cells could be interfering with the combustion model. To test if the mesh resolution were influencing the result, a further mesh refinement study was conducted with a basis in the dimensionless number D^* as seen in equation (47). This gives a starting point for the required grid resolution to resolve the plume in pool fires, frequently used for LES simulations. Comparisons of previous works

summarized by *FSD Technical Reference Guide v4* [42] concluded that this would adequately resolve the fire plume with a value on the order of 10^1. The mesh was initially set up to conform with this as a value for the grid resolution. This was further refined in steps to an order of magnitude higher, where the total grid cells were on the order of $5 \cdot 10^5$. This produced no difference in the result of the physical quantities for the initial refinements, and the significantly refined grids produced some numerical instabilities and a very high increase in computational cost. The mesh resolution was kept at a value such that the characteristic diameter D^* was twice that of the ones reported to resolve the fire plume, as this incurred little extra computational cost.

The aspect ratio of the cells and the mesh grading values were also investigated to see if this impacted the solution, as this could cause numerical inaccuracies when interpolating between faces. The grid spacing was initially heavily biased towards the area of interest, near the areas where combustion occurred. Changing this to a uniform grid with spacing equal to that of the finest cells produced no difference in the physical quantities of interest, although it produced more stable values near the boundaries. This could be of interest in further study as some of the convergent states experience with case setup (**B**) began at the boundaries. This, in turn, also tested the aspect ratios of the cells from square cells to a rectangle with a length to width ratio of $1 : 18$ as this was the largest aspect ratio in the original mesh. The number of cells in the axial direction was kept constant in the entertainment and burner area to avoid face discontinuities. Since this produced no remarkable difference from the original case setup, it is likely not the source of error in the model. Therefore the grid spacing was kept at a 1:10 bias in the axial direction and a 1:4 bias in the entertainment area towards the burner, as discussed in chapter 3 for further testing.

Further aspects of the model that might influence its poor agreement with experimental data are the handling of chemistry and other energy based terms. Tests were conducted to determine the sensitivity to this and if there are some non-physical modelling parameters included in the model. To determine if there was something erroneous in the implementation of Held and Dryer (1998) [41] methanol oxidation mechanism into OpenFOAM the coefficients were reviewed manually. This was done to see if something went wrong with the *chemkinToFoam* utility since the units differ between the standard in chemkin format and the OpenFOAM format. The manual review produced the same values as implemented, and as such, conversion errors were ruled out. Further testing the chemical mechanism aspect of the model, the mechanism by Held and Dryer was changed by implementing a simpler mechanism proposed by Mosisa Wako *et al.* (2021) [48] containing 19 reactions and validated in CFD work. This provided erroneous results and large changes in the combustion byproducts and temperature field, but it should be noted that it provided a moderate computational speedup compared to the more complex model. In reviewing the chemical mechanism, the thermodynamic data were also validated against the values provided by *GRI-Mech 3.0,* and were found to be in agreement. The heat of combustion reported by the solver was also in agreement with tabulated data for methanol. As there is likely an error in the energy related to the chemistry, as seen in the results from the infinitely fast combustion model results, a different approach

was taken, as the publicly available methanol reaction mechanisms are less validated than for other hydrocarbons.

Further isolating parts of the model for testing the reaction mechanism and fuel were substituted for both a case with the reduced DRM-22[54] methane reaction mechanism and the full *GRI-Mech 3.0,* mechanism [40]. This was done since available and tested reaction mechanisms for methanol were much fewer than for methane, and the reaction mechanism files for *GRI-Mech 3.0,* were already implemented in OpenFOAM and validated against experimental data in the Sandia-D flame simulation. Implementing a test using the geometry proposed by Tieszen *et al.* (2002)[55], methane as the fuel and with the complete *GRI-Mech 3.0,* mechanism for chemistry interactions was done. Where the turbulent properties were recalculated using the new length and velocity scales, all other parameters of the model were kept consistent with the initial case. This geometry and setup was chosen since previous comparisons have been done using LES by Sedano *et al.* (2018)[44] and the flow is buoyancy dominated. The results from this test provided many of the same problems experienced with the initial case setup and were in generally poor agreement with both the experimental results [55], and numerical results using LES [44]. This isolation testing indicates that the discrepancies in the model might lay somewhere else, such as the numerical implementation in the solver.

As can be seen from the results of the model in case (**A**), (**B**) and (**C**), the setup was very sensitive in sustaining combustion and many approaches were taken in doing so. Introducing artificially high amounts of energy into the domain to initiate or sustain the combustion might introduce non-physical behaviour. As a prerequisite for this, the TDAC method for solving the chemistry was disabled as this tabulation becomes non-physical when additional energy is introduced. In testing aspects of the ignition process, the different approaches introduced in Chapter 3 were analysed. The simplest approach utilised is setting the temperature field to a high value defined in the dictionary utilising the *setFields* utility. This approach was largely unsuccessful in sustaining combustion, but in the cases where this was used, it should not introduce any error since the extra energy applied is quickly carried out of the domain by convection. One reason this was largely unsuccessful is likely due to the energy dissipating before enough mixing has occurred in the buoyant case, as this was successfully used in the Sandia-D flame setup that included hot flume gasses, which likely slowed down the dissipation enough for combustion to sustain itself and mixing to occur. Another approach taken is to force certain regions to be at a constant temperature for a given amount of time, achieved by using the *fvConstraint* model. This was successfully applied to ignite the buoyant case. A closer inspection of the result of this approach showed that the method was unable to deactivate the temperature constraint and, as such, was highly non-physical. The function has some incorrect handling of definitions, as noted on the bug tracker for the OpenFOAM code. Therefore the results using this approach for ignition were discarded.

Another approach to achieving ignition is to start the solution with solved fields from a simpler setup. This was done using three ways, the first being cold, non-reacting flow. The second is hot non-reacting flow, as this creates a more similar velocity field and, finally, the solution from an infinitely fast chemistry model. This

provided more stability and allowed for faster convergence by allowing the velocity field to develop. This method proved unsuccessful in sustaining the ignition with the given case but should not introduce any error in the modelling of the case.

The final approach taken to achieve ignition was setting artificially high temperatures in the domain via the boundary conditions. Where the temperatures of the domain were set at 600 K ambient and 900 K for the inlet, this introduces enough energy to force combustion to occur since it is above the auto-ignition temperature of methanol. The converged solution of this was then mapped as an initial field for an identical case with reduced boundary condition values for temperature. This was done in decreasing increments of 100 to 200 K to ensure stability. Where the new converged solution would be mapped as an initial field for every decrement, down to the prescribed values of 300 K ambient and 338 K for the inlet, equal to the evaporation temperature of methanol, this method was successful in achieving and sustaining combustion for a given time, but for the latter cases close to ambient the solution would inexplicably diverge. This occurred after a large number of iterations and computational time, in the range of 20 hours of CPU time. This last method of ignition is likely contributing in some unforeseen way to incorrectness in the case setup. Although multiple people have reported success using methods similar to this, the specifics of the case setups are likely a large factor in its success. And as such, more documentation and testing are needed in ways to sustain the combustion in RANS simulations in OpenFOAM. One aspect only given a cursory look has been the source code and following how the energy and chemistry terms interact in the code. And this requires further investigation.

Another aspect of the model that might be the issue is the numerical schemes used and their parameters. This could be the underlying reason for the incorrect handling of physics and chemistry involved. Investigating this statement began with using an older version of OpenFOAM version 7. With the old *fireFoam* solver being renamed *buoyantReactingFoam* and receiving an overhaul differences might have occurred. After translating the case setup to the older format and conducting multiple simulations, no appreciable differences were found. The known good validation cases, as mentioned previously, were also run in version 7 with their respective solver. They produced equivalent results as in version 9, and as such, the difference in version was ruled out as a source of error.

The numerical schemes used to solve and discretise the equations were also altered. An overview of the different schemes used for combustion in the example cases can be gathered with the *foamSearch* utility with the *-c* argument. Some notable changes were testing the *localEuler* local time stepping method, where the Courant number restriction is imposed locally [56]. This allows for different time steps for each cell, and such is only applicable when looking for a steady-state solution. This method is implemented in the example cases for *reactingFoam* a similar solver to the one being used. Testing this introduced faster convergence but was discarded as it introduced large amounts of instabilities. Changing the schemes used to solve the divergence of different variables were also tested. This provided small changes to the way the solution converged but no overall no difference in the final result.

In determining other modelling parameters that might be the cause of non-physical

behaviour, the boundary conditions were examined. The boundary conditions for the pressure fields relating to buoyancy were changed in accordance with recommendations in [50]. This produced unstable and non-physical behaviour. The boundary conditions for the turbulence were also changed to increase the inflow of turbulence, as this could help the mixing and is directly related to the combustion in terms of the EDC model. This provided little change as the turbulence field after convergence remained largely the same. The boundary conditions for the species N2 and O2 and temperature T were changed at the ambient boundary to boundaries similar to *zeroGradient* in the hope of containing more energy in the domain. This sustained combustion but quickly lead to rising temperatures in the domain, likely not reflective of the real world. The recommendations for boundary conditions in OpenFOAM are in large parts coupled for stability, especially pressure and velocity conditions. This were in tested with variations and found in [50] and in the example files provided for both the *reactingFoam* and *buoyantReactingFoam* solvers. This was largely unsuccessful but should be of high interest in future work as the reason for the non-physical behaviour is likely boundary conditions that do not reflect reality.

Many of the problems experienced working with OpenFOAM were likely a result of a lack of documentation on combustion case setups. This resulted in many iterations of trial and error to determine how the models and parameters interacted, along with understanding the consensus of best practices. The source code annotations, along with the details provided in the example case setups, proved to be one of the better sources of correct information. The source code on the exact actions taken was examined, but getting a full understanding of the interactions between the submodels in the code proved to be a very complex task. The OpenFOAM software package has shown to be useful in simulating turbulent combustion, and many of its shortcomings are likely to get resolved with time when larger parts of the academic and commercial sectors see its value.

5 Conclusion

In this thesis, an investigation into OpenFOAM's capabilities for simulating pool fires has been conducted. Where a partially working case setup to model buoyancy dominated pool fires with methanol as the fuel, using RANS modelling, detailed chemistry and the EDC combustion model were presented. This was compared to available experimental data by Weckman and Strong[1] and numerical data produced with LES simulations and fast chemistry by Wang *et al.*[53] and Sedano *et al.*[21]. The result of the simulations using the setup deviates significantly from experimental and experimental data. This deviation came from issues with sustaining combustion and case setups which were numerically unstable. The results from the case setups presented showed large differences in both velocity and temperature fields compared to the numerical and experimental results. Some general trends can be seen, such as higher axial velocities near the centreline with a reduction towards the edge of the pool. When viewed in conjunction with the result from the case without combustion, it likely indicates that buoyant forces are active in the model. It was found that the modelling parameters describing the case are likely incorrect, and this needs further work to determine parameters that better reflect reality. The combustion model was changed to infinitely fast for one of the case setups, resulting in a highly non-physical temperature field above the adiabatic flame temperature methanol. This is likely the result of incorrect thermodynamics definitions.

Altering the case setup to identify possible sources of incorrect modelling has been conducted, focusing on testing singular aspects of the model in isolation, where many of the relevant modelling parameters were tested. The results of the testing showed no obvious cause for the lack of agreement with the numerical and experimental results, and further investigation is required. It was also found that the model used is highly sensitive to sustaining combustion, and more robust methods of ignition are required. While the results presented likely fails to capture the real physical phenomena occurring, the case setup might be helpful as a starting point for further investigations of OpenFOAM's capabilities in simulating buoyant flames. Many of the steps outlined for testing different aspects of the model are transferable to other case setups. The parameters presented are sufficient for recreating the numerical setup in OpenFOAM. Further, this thesis also provides documentation which is lacking for this combination in OpenFOAM.

Many of its shortcomings became apparent during the use and development of the simulation case in OpenFOAM. This includes a significant lack of documentation and example cases, especially in the field of combustion. Therefore, most of the work was spent on testing different aspects of the software and models included to better understand how they interacted. Although the problems experienced when working with OpenFOAM were significant, and many remain unsolved, the tool has still proved valuable in numerical investigations of fire.

Bibliography

[1] E. J. Weckman and A. B. Strong, 'Experimental investigation of the turbulence structure of medium-scale methanol pool fires', *Combustion and Flame*, vol. 105, 1996. DOI: 10.1016/0010-2180(95)00103-4.

[2] J. Warnatz, U. Maas and R. W. Dibble, *Combustion: Physical and Chemical Fundamentals, Modeling and Simulation, Experiments, Pollutant Formation*, 4th ed. Springer Berlin Heidelberg, 2012, ISBN: 978-3-642-98029-9.

[3] Z. Miao, S. Wenhua, W. Ji and C. Zhen, 'Accident Consequence Simulation Analysis of Pool Fire in Fire Dike', *Procedia Engineering*, vol. 84, 2014. DOI: 10.1016/j.proeng.2014.10.469.

[4] D. Drysdale, *An Introduction to Fire Dynamics*. Wiley, Jul. 2011, vol. 3, ISBN: 978-1-119-97547-2.

[5] I. Sikic, S. Dembele and J. Wen, 'Non-grey radiative heat transfer modelling in LES-CFD simulated methanol pool fires', *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 234, pp. 78–89, 2019. DOI: 10.1016/j.jqsrt.2019.06.004.

[6] Z. Chen, J. Wen, B. Xu and S. Dembele, 'Large eddy simulation of a medium-scale methanol pool fire using the extended eddy dissipation concept', *International Journal of Heat and Mass Transfer*, vol. 70, 2014. DOI: 10.1016/j.ijheatmasstransfer.2013.11.010.

[7] C. Wang, J. Wen and Z. Chen, 'Simulation of large-scale LNG pool fires using firefoam', English, *Combustion Science and Technology*, vol. 186, no. 10-11, pp. 1632–1649, 2014, ISSN: 0010-2202. DOI: 10.1080/00102202.2014.935615.

[8] C.-J. Wang and J. Wen, 'Numerical Study of Heptane Pool Fires on a Hollow Square Pan', ISSN: 1877-7058, vol. 211, 2018, pp. 689–698. DOI: 10.1016/j.proeng.2017.12.065.

[9] S. Razeghi, M. Safarzadeh and H. Pasdarshahri, 'Comparison of combustion models based on fast chemistry assumption in large eddy simulation of pool fire', *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 42, no. 4, 2020. DOI: 10.1007/s40430-020-02291-9.

[10] *FDS-SMV*. [Online]. Available: https://pages.nist.gov/fds-smv/ (visited on 15th Aug. 2022).

[11] V. Babrauskas and R. D. Peacock, 'Heat release rate: The single most important variable in fire hazard', *Fire Safety Journal*, vol. 18, no. 3, pp. 255–272, 1992. DOI: 10.1016/0379-7112(92)90019-9.

[12] National Fire Protection Association and Society of Fire Protection Engineers, *SFPE handbook of fire protection engineering*. National Fire Protection Association ; Society of Fire Protection Engineers, 1995, ISBN: 978-0-87765-354-7.

[13] P. H. Thomas, 'The size of flames from natural fires', in *Ninth International Symposium on Combustion*, Academic Press, 1963, pp. 844–859.

[14] P. Thomas, C. Webster and M. Raftery, 'Some experiments on buoyant diffusion flames', *Combustion and Flame*, vol. 5, pp. 359–367, 1961. DOI: 10.1016/0010-2180(61)90117-1.

[15] J. Quintiere, B. McCaffrey and K. Braven, 'Experimental and theoretical analysis of quasi-steady small-scale enclosure fires', *Symposium (International) on Combustion*, vol. 17, no. 1, pp. 1125–1137, 1979. DOI: 10.1016/S0082-0784(79)80107-1.

[16] V. Babrauskas and R. Williamson, 'Post-flashover compartment fires: Basis of a theoretical model', English, *Fire and Materials*, vol. 2, no. 2, pp. 39–53, 1978, ISSN: 0308-0501. DOI: 10.1002/fam.810020202.

[17] O. Pettersson, S. E. Magnusson and J. Thor, *Fire Engineering Design of Steel Structures*, ser. Bulletin of Division of Structural Mechanics and Concrete Construction, Bulletin 52. Lund Institute of Technology, 1976, vol. Bulletin 52.

[18] I. S. Ertesvåg, *Turbulent strøyming og forbrenning: frå turbulensteori til ingeniørverkty*, nno. Trondheim: Tapir akademisk forl, 2000, Book Title: Turbulent strøyming og forbrenning : frå turbulensteori til ingeniørverkty, ISBN: 978-82-519-1568-7.

[19] A. Snell and L. Segervall, 'HPC Application Support for GPU Computing', p. 10, 2017.

[20] S. Jones and A. Lichtl, *GPUs TO MARS*. [Online]. Available: https://on-demand.gputechconf.com/gtc/2015/presentation/S5398-Stephen-Jones-Adam-Lichtl.pdf (visited on 15th Aug. 2022).

[21] C. A. Sedano, O. D. López, A. Ladino and F. Muñoz, 'Prediction of a Small-Scale Pool Fire with FireFoam', *International Journal of Chemical Engineering*, vol. 2017, pp. 1–12, 2017. DOI: 10.1155/2017/4934956.

[22] Y. Z. Li *et al.*, *Verification, validation and evaluation of FireFOAM as a tool for performance design*. 2017.

[23] Y. Sun, H. Shen, S. Zheng and L. Jiang, 'A hybrid non-gray gas radiation heat transfer solver based on OpenFOAM', *Journal of Quantitative Spectroscopy and Radiative Transfer*, vol. 281, 2022. DOI: 10.1016/j.jqsrt.2022.108105.

[24] H. K. Versteeg and W. Malalasekera, *An introduction to computational fluid dynamics: the finite volume method*, eng, 2nd ed. 2007, Book Title: An introduction to computational fluid dynamics : the finite volume method, ISBN: 978-0-13-127498-3.

[25] B. E. Launder and D. B. Spalding, 'The numerical computation of turbulent flows', en, p. 21, 1974.

[26] D. B. Spalding, 'Mixing and chemical reaction in steady confined turbulent flames', *Symposium (International) on Combustion*, Thirteenth symposium (International) on Combustion, vol. 13, no. 1, pp. 649–657, 1971. DOI: 10.1016/S0082-0784(71)80067-X.

[27] B. Magnussen, 'On the structure of turbulence and a generalized eddy dissipation concept for chemical reaction in turbulent flow', in *19th Aerospace Sciences Meeting*, American Institute of Aeronautics and Astronautics, 1981. DOI: 10.2514/6.1981-42.

[28] E. Ghasemi, S. Soleimani and C. Lin, 'RANS simulation of methane-air burner using local extinction approach within eddy dissipation concept by OpenFOAM', *International Communications in Heat and Mass Transfer*, vol. 54, pp. 96–102, 2014. DOI: 10.1016/j.icheatmasstransfer.2014.03.006.

[29] M. T. Lewandowski and I. S. Ertesvåg, 'Analysis of the Eddy Dissipation Concept formulation for MILD combustion modelling', *Fuel*, vol. 224, pp. 687–700, 2018. DOI: 10.1016/j.fuel.2018.03.110.

[30] B. F. Magnussen, 'THE EDDY DISSIPATION CONCEPT A BRIDGE BETWEEN SCIENCE AND TECHNOLOGY', en, p. 25, 2005.

[31] D. A. Lysenko, I. S. Ertesvåg and K. E. Rian, 'Numerical Simulation of Non-premixed Turbulent Combustion Using the Eddy Dissipation Concept and Comparing with the Steady Laminar Flamelet Model', *Flow, Turbulence and Combustion*, vol. 93, no. 4, pp. 577–605, 2014. DOI: 10.1007/s10494-014-9551-7.

[32] M. F. Modest, *Radiative Heat Transfer*, English, 2nd edition. Amsterdam ; Boston: Academic Press, Mar. 2003, ISBN: 978-0-12-503163-9.

[33] F. Contino, H. Jeanmart, T. Lucchini and G. D'Errico, 'Coupling of in situ adaptive tabulation and dynamic adaptive chemistry: An effective method for solving combustion in engine simulations', *Proceedings of the Combustion Institute*, vol. 33, no. 2, pp. 3057–3064, 2011. DOI: 10.1016/j.proci.2010.08.002.

[34] S. Pope, 'Computationally efficient implementation of combustion chemistry using **in situ** adaptive tabulation', *Combustion Theory and Modelling*, vol. 1, no. 1, pp. 41–63, 1997. DOI: 10.1080/713665229.

[35] L. Liang, J. G. Stevens and J. T. Farrell, 'A dynamic adaptive chemistry scheme for reactive flow computations', *Proceedings of the Combustion Institute*, vol. 32, no. 1, pp. 527–534, 2009. DOI: 10.1016/j.proci.2008.05.073.

[36] F. Contino, J.-B. Masurier, F. Foucher, T. Lucchini, G. D'Errico and P. Dagaut, 'CFD simulations using the TDAC method to model iso-octane combustion for a large range of ozone seeding and temperature conditions in a single cylinder HCCI engine', *Fuel*, vol. 137, pp. 179–184, 2014. DOI: 10.1016/j.fuel.2014.07.084.

[37] F. Contino, H. Jeanmart, T. Lucchini and G. D'Errico, 'Tabulation of Dynamic Adaptive Chemistry: A global approach to include detailed mechanisms in engine simulations', *how it works*, p. 32,

[38] C. Greenshields, *OpenFOAM v9 user guide*. London, UK: The OpenFOAM Foundation, 2021. [Online]. Available: https://doc.cfd.direct/openfoam/user-guide-v9 (visited on 15th Aug. 2022).

[39] *OpenFOAM: Free, Open Source Software from the OpenFOAM Foundation.* [Online]. Available: https://cpp.openfoam.org/v9/ (visited on 15th Aug. 2022).

[40] G. P. Smith, D. M. Golden, M. Frenklach and N. W. Moriarty, *GRI-Mech 3.0,* [Online]. Available: http://combustion.berkeley.edu/gri-mech/.

[41] T. J. Held and F. L. Dryer, 'A comprehensive mechanism for methanol oxidation', *International Journal of Chemical Kinetics*, vol. 30, no. 11, pp. 805–830, 1998. DOI: 10.1002/(SICI)1097-4601(1998)30:11⟨805::AID-KIN4⟩3.0.CO;2-Z.

[42] K. B. McGrattan, 'FSD v4 :: Technical Reference Guide', National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. NIST SP 1018, 2006, Edition: 0. DOI: 10.6028/NIST.SP.1018.

[43] R. Zamorano, 'FIREFOAM (CFD SOLVER) VALIDATION IN COMPARTMENT FIRE SCENARIO USING HIGH RESOLUTION DATA', en, p. 178, 2018.

[44] C. Sedano, O. López, A. Ladino and F. Muñoz, 'Prediction of a methane circular pool fire with fireFoam', *MATEC Web of Conferences*, vol. 240, S. Gradziel, S. Łopata, T. Sobota and W. Zima, Eds., 2018. DOI: 10.1051/matecconf/201824005026.

[45] H. Norouzi, *Sandia/TUD Piloted CH4/Air Jet Flames — TNF Workshop*. [Online]. Available: https://tnfworkshop.org/data-archives/pilotedjet/ch4-air/ (visited on 15th Aug. 2022).

[46] S. Trefall, 'TEP4540 - Engineering Fluid Mechanics, Specialization Project', en, p. 51,

[47] J. Paulasalo, 'CFD MODELLING OF INDUSTRIAL SCALE GAS FLAME WITH OPENFOAM SOFTWARE', en, p. 115, 2019.

[48] F. Mosisa Wako, G. Pio and E. Salzano, 'Reduced Combustion Mechanism for Fire with Light Alcohols', *Fire*, vol. 4, no. 4, 2021, Number: 4 Publisher: Multidisciplinary Digital Publishing Institute. DOI: 10.3390/fire4040086.

[49] PubChem, *Methanol*, en. [Online]. Available: https://pubchem.ncbi.nlm.nih.gov/compound/887 (visited on 15th Aug. 2022).

[50] C. J. Greenshields and H. G. Weller, *Notes on computational fluid dynamics: general principles*, eng. Reading, UK: CFD Direct Limited, 2022, ISBN: 978-1-399-92078-0.

[51] D. Le, J. Labahn, T. Beji, C. B. Devaud, E. J. Weckman and A. Bounagui, 'Assessment of the capabilities of FireFOAM to model large-scale fires in a well-confined and mechanically ventilated multi-compartment structure', *Journal of Fire Sciences*, vol. 36, no. 1, pp. 3–29, 2018, Publisher: SAGE Publications Ltd STM. DOI: 10.1177/0734904117733427.

[52] S. Keough, 'Optimising the Parallelisation of OpenFOAM Simulations', en, p. 48,

[53] Y. Wang, P. Chatterjee and J. L. de Ris, 'Large eddy simulation of fire plumes', *Proceedings of the Combustion Institute*, vol. 33, no. 2, pp. 2473–2480, 2011. DOI: 10.1016/j.proci.2010.07.031.

[54] *Reduced Reaction Sets based on GRI-Mech 1.2*. [Online]. Available: http://combustion.berkeley.edu/drm/.

[55] S. Tieszen, T. O'Hern, R. Schefer, E. Weckman and T. Blanchat, 'Experimental study of the flow field in and around a one meter diameter methane fire', *Combustion and Flame*, vol. 129, no. 4, pp. 378–391, 2002. DOI: 10.1016/S0010-2180(02)00352-8.

[56] C. Greenshields, *OpenFOAM 2.0.0: Steady-State VoF — OpenFOAM*, en-GB. [Online]. Available: https://openfoam.org/release/2-0-0/steady-state-vof/ (visited on 17th Aug. 2022).

[57] A. H. (Kadar, 'Modelling Turbulent Non-Premixed Combustion in Industrial Furnaces', en, Ph.D. dissertation, Delft University of Technology, 2015. (visited on 15th Aug. 2022).

[58] M. O. Bolstad, 'Jet fires', eng, *136*, 2016, Accepted: 2016-09 Publisher: NTNU.

Appendix

A System properties

Architecture:	x86_64
CPU op-mode(s):	32-bit, 64-bit
Byte Order:	Little Endian
Address sizes:	39 bits physical, 48 bits virtual
CPU(s):	20
On-line CPU(s) list:	0-19
Thread(s) per core:	2
Core(s) per socket:	10
Socket(s):	1
Vendor ID:	GenuineIntel
CPU family:	6
Model:	165
Model name:	Intel(R) Core(TM) i9-10900K CPU @ 3.70GHz
Stepping:	5
CPU MHz:	3700.000
CPU max MHz:	5300,0000
Virtualization:	VT-x
L1d cache:	320 KiB
L1i cache:	320 KiB
L2 cache:	2,5 MiB
L3 cache:	20 MiB
System Memory	DIMM DDR4 Synchronous 2133 MHz (0,5 ns)
Size	32GiB
Operating System	Ubuntu 20.04.4 LTS x86_64
OpenFOAM version	v7, v9 as released by The OpenFOAM Foundation Ltd

Table 4: System Info for the computer used to run OpenFOAM

B Source code for *buoyantReactingFoam* solver

```
/*---------------------------------------------------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website: https://openfoam.org
    \\  /    A nd           | Copyright (C) 2012-2021 OpenFOAM Foundation
     \\/     M anipulation  |
-------------------------------------------------------------------------------
License
    This file is part of OpenFOAM.

    OpenFOAM is free software: you can redistribute it and/or modify it
    under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    OpenFOAM is distributed in the hope that it will be useful, but
        WITHOUT
    ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
    FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License
    for more details.

    You should have received a copy of the GNU General Public License
    along with OpenFOAM. If not, see <http://www.gnu.org/licenses/>.

Application
    buoyantReactingFoam

Description
    Transient solver for turbulent flow of compressible reacting fluids
        with
    enhanced buoyancy treatment and optional mesh motion and mesh topology
    changes.

    Uses the flexible PIMPLE (PISO-SIMPLE) solution for time-resolved and
    pseudo-transient simulations.

\*---------------------------------------------------------------------------*/

#include "fvCFD.H"
#include "dynamicFvMesh.H"
#include "fluidReactionThermo.H"
#include "combustionModel.H"
#include "dynamicMomentumTransportModel.H"
#include "fluidReactionThermophysicalTransportModel.H"
#include "multivariateScheme.H"
#include "pimpleControl.H"
#include "pressureReference.H"
```

```
#include "hydrostaticInitialisation.H"
#include "CorrectPhi.H"
#include "fvModels.H"
#include "fvConstraints.H"
#include "localEulerDdtScheme.H"
#include "fvcSmooth.H"

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

int main(int argc, char *argv[])
{
    #include "postProcess.H"

    #include "setRootCaseLists.H"
    #include "createTime.H"
    #include "createDynamicFvMesh.H"
    #include "createDyMControls.H"
    #include "initContinuityErrs.H"
    #include "createFields.H"
    #include "createFieldRefs.H"
    #include "createRhoUfIfPresent.H"

    turbulence->validate();

    if (!LTS)
    {
        #include "compressibleCourantNo.H"
        #include "setInitialDeltaT.H"
    }

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

    Info<< "\nStarting time loop\n" << endl;

    while (pimple.run(runTime))
    {
        #include "readDyMControls.H"

        // Store divrhoU from the previous mesh so that it can be mapped
        // and used in correctPhi to ensure the corrected phi has the
        // same divergence
        autoPtr<volScalarField> divrhoU;
        if (correctPhi)
        {
            divrhoU = new volScalarField
            (
                "divrhoU",
                fvc::div(fvc::absolute(phi, rho, U))
            );
        }
```

```
if (LTS)
{
    #include "setRDeltaT.H"
}
else
{
    #include "compressibleCourantNo.H"
    #include "setDeltaT.H"
}

runTime++;

Info<< "Time = " << runTime.timeName() << nl << endl;

// --- Pressure-velocity PIMPLE corrector loop
while (pimple.loop())
{
    if (!pimple.flow())
    {
        if (pimple.models())
        {
            fvModels.correct();
        }

        if (pimple.thermophysics())
        {
            #include "YEqn.H"
            #include "EEqn.H"
        }
    }
    else
    {
        if (pimple.firstPimpleIter() || moveMeshOuterCorrectors)
        {
            // Store momentum to set rhoUf for introduced faces.
            autoPtr<volVectorField> rhoU;
            if (rhoUf.valid())
            {
                rhoU = new volVectorField("rhoU", rho*U);
            }

            fvModels.preUpdateMesh();

            // Do any mesh changes
            mesh.update();

            if (mesh.changing())
            {
                gh = (g & mesh.C()) - ghRef;
                ghf = (g & mesh.Cf()) - ghRef;
```

```
                MRF.update();

                if (correctPhi)
                {
                    #include "correctPhi.H"
                }

                if (checkMeshCourantNo)
                {
                    #include "meshCourantNo.H"
                }
            }
        }

        if (pimple.firstPimpleIter() && !pimple.simpleRho())
        {
            #include "rhoEqn.H"
        }

        if (pimple.models())
        {
            fvModels.correct();
        }

        #include "UEqn.H"

        if (pimple.thermophysics())
        {
            #include "YEqn.H"
            #include "EEqn.H"
        }

        // --- Pressure corrector loop
        while (pimple.correct())
        {
            #include
                "../../../heatTransfer/buoyantPimpleFoam/pEqn.H"
        }

        if (pimple.turbCorr())
        {
            turbulence->correct();
            thermophysicalTransport->correct();
        }
    }
}

rho = thermo.rho();

runTime.write();
```

```
        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
            << "  ClockTime = " << runTime.elapsedClockTime() << " s"
            << nl << endl;
    }

    Info<< "End\n" << endl;

    return 0;
}


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

C Source code for *reactingFoam* solver

```
/*---------------------------------------------------------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website: https://openfoam.org
    \\  /    A nd           | Copyright (C) 2011-2021 OpenFOAM Foundation
     \\/     M anipulation  |
-------------------------------------------------------------------------------
License
    This file is part of OpenFOAM.

    OpenFOAM is free software: you can redistribute it and/or modify it
    under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    OpenFOAM is distributed in the hope that it will be useful, but
        WITHOUT
    ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
    FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License
    for more details.

    You should have received a copy of the GNU General Public License
    along with OpenFOAM. If not, see <http://www.gnu.org/licenses/>.

Application
    reactingFoam

Description
    Transient solver for turbulent flow of compressible reacting fluids
        with
    optional mesh motion and mesh topology changes.

    Uses the flexible PIMPLE (PISO-SIMPLE) solution for time-resolved and
    pseudo-transient simulations.

\*---------------------------------------------------------------------------*/

#include "fvCFD.H"
#include "dynamicFvMesh.H"
#include "fluidReactionThermo.H"
#include "combustionModel.H"
#include "dynamicMomentumTransportModel.H"
#include "fluidReactionThermophysicalTransportModel.H"
#include "multivariateScheme.H"
#include "pimpleControl.H"
#include "pressureReference.H"
#include "CorrectPhi.H"
```

```
#include "fvModels.H"
#include "fvConstraints.H"
#include "localEulerDdtScheme.H"
#include "fvcSmooth.H"

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

int main(int argc, char *argv[])
{
    #include "postProcess.H"

    #include "setRootCaseLists.H"
    #include "createTime.H"
    #include "createDynamicFvMesh.H"
    #include "createDyMControls.H"
    #include "initContinuityErrs.H"
    #include "createFields.H"
    #include "createFieldRefs.H"
    #include "createRhoUfIfPresent.H"

    turbulence->validate();

    if (!LTS)
    {
        #include "compressibleCourantNo.H"
        #include "setInitialDeltaT.H"
    }

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

    Info<< "\nStarting time loop\n" << endl;

    while (pimple.run(runTime))
    {
        #include "readDyMControls.H"

        // Store divrhoU from the previous mesh so that it can be mapped
        // and used in correctPhi to ensure the corrected phi has the
        // same divergence
        autoPtr<volScalarField> divrhoU;
        if (correctPhi)
        {
            divrhoU = new volScalarField
            (
                "divrhoU",
                fvc::div(fvc::absolute(phi, rho, U))
            );
        }

        if (LTS)
```

```
{
    #include "setRDeltaT.H"
}
else
{
    #include "compressibleCourantNo.H"
    #include "setDeltaT.H"
}

runTime++;

Info<< "Time = " << runTime.timeName() << nl << endl;

// --- Pressure-velocity PIMPLE corrector loop
while (pimple.loop())
{
    if (!pimple.flow())
    {
        if (pimple.models())
        {
            fvModels.correct();
        }

        if (pimple.thermophysics())
        {
            #include "YEqn.H"
            #include "EEqn.H"
        }
    }
    else
    {
        if (pimple.firstPimpleIter() || moveMeshOuterCorrectors)
        {
            // Store momentum to set rhoUf for introduced faces.
            autoPtr<volVectorField> rhoU;
            if (rhoUf.valid())
            {
                rhoU = new volVectorField("rhoU", rho*U);
            }

            fvModels.preUpdateMesh();

            // Do any mesh changes
            mesh.update();

            if (mesh.changing())
            {
                MRF.update();

                if (correctPhi)
                {
```

```
                        #include "correctPhi.H"
                    }

                    if (checkMeshCourantNo)
                    {
                        #include "meshCourantNo.H"
                    }
                }
            }

            if (pimple.firstPimpleIter() && !pimple.simpleRho())
            {
                #include "rhoEqn.H"
            }

            if (pimple.models())
            {
                fvModels.correct();
            }

            #include "UEqn.H"

            if (pimple.thermophysics())
            {
                #include "YEqn.H"
                #include "EEqn.H"
            }

            // --- Pressure corrector loop
            while (pimple.correct())
            {
                #include "../../compressible/rhoPimpleFoam/pEqn.H"
            }

            if (pimple.turbCorr())
            {
                turbulence->correct();
                thermophysicalTransport->correct();
            }
        }
    }

    rho = thermo.rho();

    runTime.write();

    Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
        << "  ClockTime = " << runTime.elapsedClockTime() << " s"
        << nl << endl;
}
```

```
    Info<< "End\n" << endl;

    return 0;
}


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

D Code used to generate the mesh with *blockMesh*. Based on wedge mesh examples in OpenFOAM

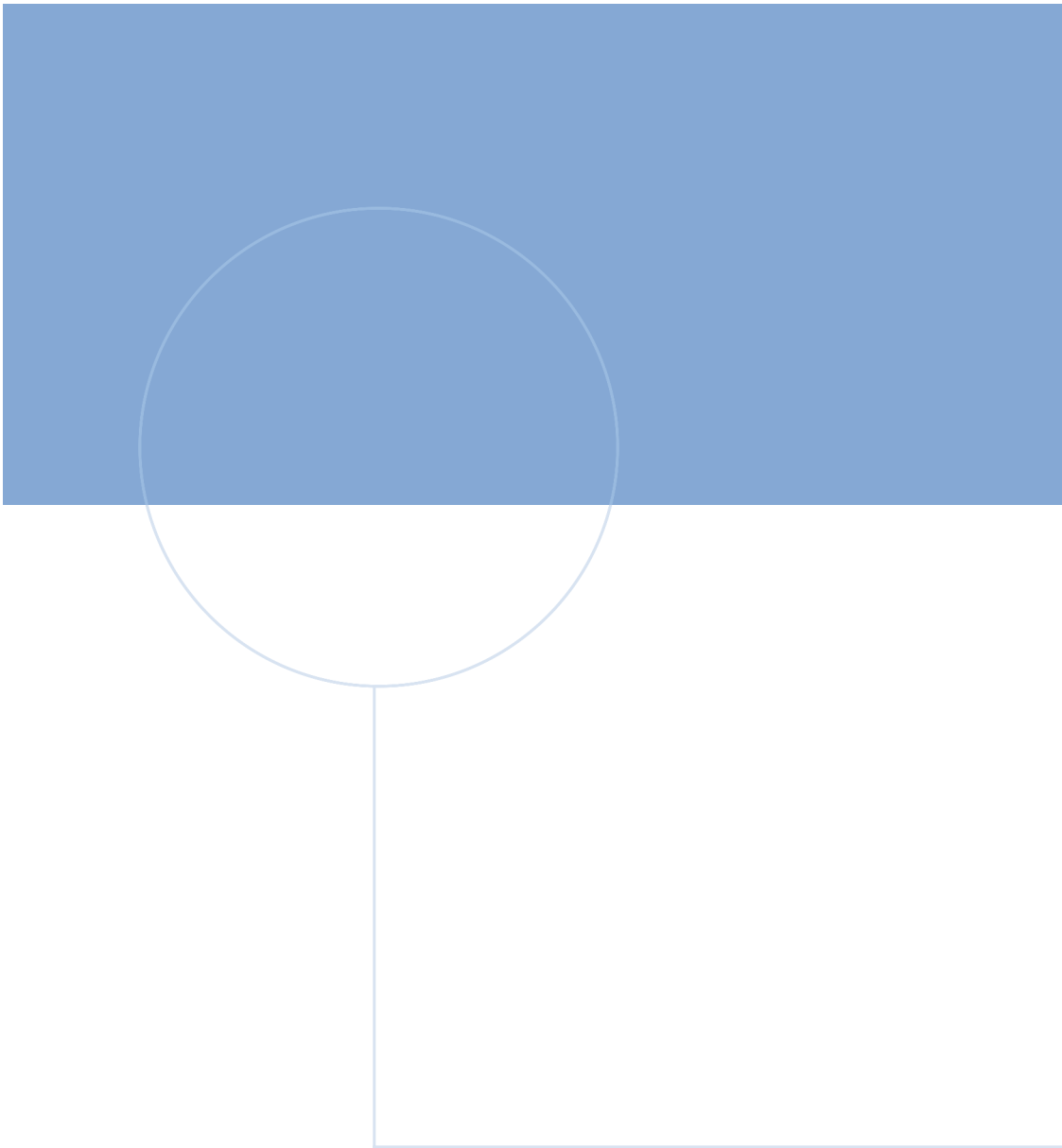
```
/*--------------------------*- C++ -*----------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website: https://openfoam.org
    \\  /    A nd           | Version: 9
     \\/     M anipulation  |
\*---------------------------------------------------------------*/
FoamFile
{
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * ** * * * * * * * * * * * *
    //

    length 2.745;
    rA     0.1525; // radius of inlet patch
    rB      1.3725; // outer radius 9x rA
    lengthCells 160;
    rAcells 30;
    rBcells 80;

convertToMeters 1;

vertices
(
    (                    0                  0 -1)
    ($length                 0 -1)
    (                    0 $rA -1)
    ($length $rA -1)
    (                    0 $rB -1)
    ($length $rB -1)

    (                    0                  0  0)
    ($length                 0  0)
    (                    0 $rA  0)
    ($length $rA 0)
    (                    0 $rB  0)
    ($length $rB 0)
);

blocks
(
    hex (0 1 3 2 6 7 9 8)
    ($lengthCells $rAcells 1)
    simpleGrading (10 1 1)
```

```
    hex (2 3 5 4 8 9 11 10)
    ($lengthCells $rBcells 1)
    simpleGrading (10 4 1)
);

edges
(
);

boundary
(
    inlet
    {
        type patch;
        faces
        (
            (0 6 8 2)
        );
    }

    front
    {
        type symmetry;
        faces
        (
            (6 7  9 8)
            (8 9 11 10)
        );
    }

    back
    {
        type symmetry;
        faces
        (
            (0 1 3 2)
            (2 3 5 4)
        );
    }

    sides
    {
        type patch;
        faces
        (
            (4 5 11 10) //thick side of the wedge
        );
    }

    outlet
```

```
    {
        type patch;
        faces
        (
            (5 3  9 11) //top
            (3 1  7  9) //above inlet inlet
        );
    }

    base
    {
        type patch;
        faces
        (
            (2 8 10 4)
        );
    }

);

mergePatchPairs
(
);

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

E Code used to extrude to a wedge shape based on mesh from *blockMesh*.

```
/*--------------------------------*- C++ -*----------------------------*\
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website: https://openfoam.org
    \\  /    A nd           | Version: 9
     \\/     M anipulation  |
\*---------------------------------------------------------------------*/
FoamFile
{
    format      ascii;
    class       dictionary;
    object      extrudeProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

constructFrom patch;
sourceCase "$FOAM_CASE";

sourcePatches (front);
exposedPatchName back;

extrudeModel      wedge;

sectorCoeffs
{
    axisPt          (0 0 0);
    axis            (1 0 0);
    angle           3;
}

flipNormals false;
mergeFaces false;

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



NTNU

Norwegian University of
Science and Technology