Lehrstuhl für Aerodynamik
und Strömungmechanik
Prof. Dr.-Ing. N.A. Adams

Toolchain for Aerodynamic Characterization of a Rocket During Ascent using OpenFOAM

Félix Martí Valverde
Nr. /2023

Semesterarbeit

Betreuer: apl. Prof. Dr.-Ing. habil. Christian Stemmer

Ausgabe: 1st April 2023
Abgabe: 1st November 2023

Lehrstuhl für Aerodynamik und Strömungsmechnik
der Technischen Universität München

2023

Abstract

Computational Fluid Dynamics (CFD) tools from providers like ANSYS (Fluent) or Siemens (Star-CCM+) can be a financial burden on small businesses and startups alike. Thus, open-source alternatives like OpenFOAM are becoming increasingly appealing to reduce development costs. However, OpenFOAM has largely remained a research tool reserved for academia due to its lack of documentation, a high level of expertise required, and a lack of standardization and Graphical User Interface (GUI). This paper identifies these shortcomings and aims to streamline the process, which is generally reserved for academia, thus easing the commercial adoption of this tool. This document presents an OpenFOAM solution aimed at extracting the aerodynamic database of a rocket with all the relevant aerodynamic coefficients. Two solvers, *rhoPimpleFoam* a pressure-based solver and *rhoCentralFoam* a density-based solver, will be compared to determine which one better matches the expected aerodynamic coefficients. A wind-tunnel validated geometry for a rocket has been identified and will be used to assess the validity of the results obtained. Finally, lessons learned regarding the project's organizational structure, including automation, standardization, and scalability, will be discussed to showcase their potential application in other industrial cases. All necessary tools for this CFD toolchain are freely available.

Contents

List of Abbreviations

AMR Adaptive Mesh Refinement

CAD Computer Aided Design

CFD Computational Fluid Dynamics

ESI OpenFOAM ESI Group

FOs Functional Objects

GR Growth Ratio

GUI Graphical User Interface

LTS Local Time Stepping

NASA National Aeronautics and Space Administration

OF OpenFOAM Foundation

OS Operating System

PIFS Plume-Induced Flow Separation

PISO Pressure Implicit with Splitting of Operators

SA Spalart Allmaras 1-Equation Turbulence Model

SHM snappyHexMesh

SIMPLE Semi-Implicit Method for Pressure Linked Equations

SST Shear-Stress Transport 2-Equation Turbulence Model

VSC Visual Studio Code

WSL Windows Subsystem for Linux

List of Symbols

Sign	Description	Unit
A_c	Cross Sectional Area	m^2
AoA	Angle of Attack	deg
C_A	Aerodynamic Axial Coefficient	-
C_N	Aerodynamic Normal Coefficient	-
$C_{m_{\text{pitch}}}$	Aerodynamic Pitch Moment Coefficient	-
Co	Courant Number	-
D	Hydraulic Diameter	m
E	Wall Roughness Parameter	-
I	Turbulent Intensity	%
Ma	Mach Number	-
Pr	Prandtl Number	-
Re	Reynolds Number	-
U_∞	Free Stream Velocity	$\mathrm{m} \cdot \mathrm{s}^{-1}$
c_p	Pressure Coefficient	-
k	Turbulent Kinetic Energy	$\mathrm{m}^2 \cdot \mathrm{s}^{-2}$
l	Characteristic Length	m
y	Wall-Normal Height	m
y^+	Non-Dimensional Wall Spacing	-
$\frac{\mu_t}{\mu}$	Eddy Viscosity Ratio	-
κ	von Kármán Constant	-
ω	Specific Dissipation Rate	s^{-1}
α	Thermal Diffusivity	$\mathrm{m}^2 \cdot \mathrm{s}^{-1}$
δ	Boundary Layer Thickness	m
μ	Dynamic Viscosity	$\mathrm{kg} \cdot \mathrm{m}^{-1} \cdot \mathrm{s}^{-1}$
μ_t	Turbulent Dynamic Viscosity	$\mathrm{kg} \cdot \mathrm{m}^{-1} \cdot \mathrm{s}^{-1}$
ν	Kinetic Viscosity	$\mathrm{m}^2 \cdot \mathrm{s}^{-1}$
ρ_∞	Free Stream Density	$\mathrm{kg} \cdot \mathrm{m}^{-3}$

1 Introduction

1.1 Motivation

The primary objective of this project is to provide a guide on how to set up a complete CFD toolchain with the purpose of aerodynamically characterizing a given rocket geometry by providing aerodynamic coefficients for axial (C_A), normal (C_N) forces, and pitching moment ($C_{m_{\text{pitch}}}$). Additionally, considerations regarding the validity of other parameters such as surface temperatures and shock wave positions will be addressed, as they are important for assessing the validity of the CFD solution and providing valuable information during rocket design phases. The work presented exclusively employs open-source and freely available tools, thus eliminating the need for expensive CFD solutions such as ANSYS (Fluent) or Siemens (Star-CCM+). These licensed software options come at a substantial cost, making it necessary for small companies, individuals, and start-ups to seek more cost-effective alternatives. OpenFOAM has become the go-to alternative to expensive CFD analysis software, with many distributions available. For this work, the OpenFOAM ESI Group (ESI) distribution has been chosen.

After introducing the current state-of-the-art CFD solutions used for aerodynamic analysis of rockets in Section 1.2, a discussion will take place in Chapter 2 regarding the selected tools to carry out this project and the automation of the main steps in CFD analysis using Python and other programming tools. Chapter 3 will discuss the validation of CFD results obtained using the proposed OpenFOAM toolchain. To achieve this, a rocket geometry which has previously been tested in a wind-tunnel by NASA has been selected and reproduced using SolidWorks. Additionally, a set of atmospheric conditions has been selected to cover the main aerodynamic regimes (subsonic, transonic, and supersonic) and the flight envelope using different Angles of Attack (AoA), thus providing an accurate assessment of the CFD results obtained.

As the first step in CFD analysis, Chapter 4 will cover all the open-source and freely available meshing software in the market. The most compelling meshing solution will then be broken down into steps and discussed. Additionally, Section 4.2 in Chapter 4 will present a grid convergence study, as well as discuss the y^+ values achieved using the finest mesh in relation to the resolution of the boundary layer. Chapter 5 will address important parameters regarding OpenFOAM simulations using the two selected solvers identified in [1]: *rhoPimpleFoam* and *rhoCentralFoam*. First, boundary condition types for free-stream quantities (U, T, p) will be discussed, differentiating between use cases for supersonic and subsonic regimes. Then, a detailed discussion of turbulence modeling will address the turbulence model selected, the boundary condition types for modeling turbulence at the boundary layer, and calculations for turbulent quantities at the inlet. The chapter will continue with a discussion addressing time schemes for accelerated convergence to steady state for both unsteady solvers *rhoPimpleFoam* and *rhoCentralFoam*.

Finally, a discussion on the execution instructions followed by the solvers will address different techniques for faster convergence, such as mapping fields from a coarser mesh to initialize simulations. Additionally, other topics like monitoring algorithms to automatically stop simulations when convergence criteria are met will be covered. In the last chapter, Chapter 6, there will be an extensive analysis of the CFD results obtained from every simulation.

The comprehensive, modular, and expandable OpenFOAM toolchain presented here is made available to anyone with the intention of reducing the barrier to adopting OpenFOAM as a low-cost competitive alternative for CFD analysis. Interested parties can access the code at [2] and are invited to extend the scope of the project and share their findings.

1.2 State of Research

The current state-of-the-art CFD techniques are being implemented at National Aeronautics and Space Administration (NASA), and other aerospace companies and organization, to make informed design decisions with particular interest in further thermal and structural analysis. Papers [3, 4, 5, 6, 7] from renowned space agencies provide guidelines on the best practices to follow during the aerodynamic analysis of a rocket.

Of particular importance is [4], in which NASA presents a consistent methodology to validate the CFD results. In this paper, a comparative analysis between wind-tunnel data of a scaled-down rocket model and the CFD results is done. Moreover, the study summarizes the necessary convergence and sensitivity analyses to be done, such as grid-convergence and sensitivity for numerical parameters and turbulence models. In addition, the paper also contains a code-to-code comparison between the different solvers utilized. A similar comparative analysis among both selected OpenFOAM solvers can be found in Section 6, and sensitivity and convergence studies will also be discussed in later sections.

Previously mentioned studies [3, 7] already address the importance of the exhaust plumes on the aerodynamic performance of the vehicle and its thermal loads. Moreover, the paper [7] delves deeper than the aforementioned documents by comparing the effects of the plume on the relevant aerodynamic coefficients. Conclusions show a significant reduction in drag during powered flight up to Mach mumber (Ma) 5, at which point differences become negligible. Also, paper [3] conducts an extensive analysis on the impact of the plume and how to model it accurately. This document is of significant relevance as it compares different CFD setups and identifies the most significant parameters for each one. Among the proposed solvers, the dilemma between multi-species and single-species solvers and the turbulence model used (Spalart Allmaras 1-Equation Turbulence Model (SA) or Shear-Stress Transport 2-Equation Turbulence Model (SST)) are the main takeaways. Conclusions on [3] indicate that the turbulence model SST better predicted the Plume-Induced Flow Separation (PIFS) position compared to SA. In addition, the single-species model is able to predict the PIFS distance, even if multi-species solvers perform slightly better. As stated in [3], single-species solvers only have one equation of state, meaning that only two of the three thermodynamic quantities of the plume's exhaust will match reality. Investigation showed that the only good approximation had pressure and velocity set to real values, with the third thermodynamic quantity calculated to match density. Finally, an assessment on how to implement the engines concluded that nozzle interior is required to obtain valid PIFS results, as setting a uniform field at the exit plane of the engine nozzle is too crude of an approximation to match PIFS distances.

Advanced meshing techniques like Adaptive Mesh Refinement (AMR) are discussed in [4]. By using AMR, the mesh is automatically refined based on a user-provided function that analyzes the fields calculated from the previous iteration and indicates which cells need to be refined. In a Cartesian mesh, this refinement takes place by splitting selected cells into 8. Excessive refinement through AMR can result in shock waves being redirected into the refined areas, ultimately over-fitting the results to the mesh. Other meshing solutions include overset meshes, as presented in [3], which offer an interesting approach that should yield greater accuracy. An overset mesh consists of one or more sub-meshes that coexist superimposed on a background mesh. The division into multiple sub-meshes allows for tailoring each sub-mesh to specific components, enabling the use of structured meshes and reducing sensitivity to mesh quality. In the course of this work, overset meshes were explored due to their high potential and the apparent simplicity of the geometry at work. Nevertheless, the idea was dropped due to software instabilities of the overset solvers in the current versions of OpenFOAM. Alternatively, most of the benefits of using overset mesh could be reproduced using multi-block grids. Using *cyclicAMI* boundary condition types, meshes that shared patches could transfer information between them. Ultimately, this idea was not realized, and an unstructured Cartesian mesh was chosen instead.

2 Open Source Tools and Automated Workflow

Based on the available distributions of OpenFOAM (OpenFOAM Foundation (OF), OpenFOAM Extend, and OpenFOAM ESI Group (ESI) or other commercial releases) the best one for industrial use is ESI. OpenFOAM ESI Group releases are geared towards industrial environments with reliable and stable updates which come every 6 months contrary to OF. What follows is a list of tools used for this project, all being open-source or freely available:

- MobaXterm
- Visual Studio Code (VSC)
- Paraview
- GNUPLOT
- SALOME
- CfMesh
- Functional Objects (FOs)
- bash
- Python
- GitHub - GitLab
- Docker
- Windows Subsystem for Linux (WSL)
- XLaunch

2.1 Tools

Due to the large computational requirements of CFD analysis, development will take place on a local machine, while execution will happen on a server. To share the code between machines, the code will be placed on GitHub (or GitLab), helping track changes and enabling collaboration. To communicate with the server, MobaXterm for Windows offers an SSH communication protocol terminal with a capable Graphical User Interface (GUI) to navigate the server folders. Also, the ability to open a GUI from the server on the local machine, as with XLaunch. In combination with MobaXterm, Visual Studio Code (VSC) can be used to edit the server's code remotely and more, through SSH protocols.

Regarding monitoring tools, GNUPLOT is used to plot the convergence of residuals and aerodynamic coefficients as the CFD solutions are being computed. GNUPLOT reads a *dat* file, which is updated at every iteration of the simulation using Functional Objects (FOs) native to OpenFOAM. This system works coordinated by a bash script, which semi-automatizes the task, and plots all relevant residuals or aerodynamic coefficients based on needs. Finally, ParaView is used to visualize results, which can be downloaded through MobaXterm or VSC as the simulation computes. On the topic of OpenFOAM tools for semi-automated workflows, bash and Python scripts combined with FOs provide the building blocks for semi-automatization, more information will be provided in Section 2.3,

2.2 Docker for OpenFOAM

This work explores the use of Docker as a fast and reliable method to deploy OpenFOAM on any machine. Docker works on the principle of containers, a lightweight, standalone, and executable package that contains everything needed to run an application. Powerful servers can be rented on demand to compute scientific work on otherwise inaccessible machines due to their high acquisition costs. A server with Docker preinstalled can be rented, and accessed using the SSH protocol, and in a matter of minutes, OpenFOAM is ready to start computing the CFD solution. Once the work is done, the Docker container closes, liberating the server, and the user only

pays for the time used. Linode® offers a wide range of capable machines, that have already been used for similar work, e.g., training machine learning models. Performance drops due to the additional interface between the container and the Operating System (OS) should be considered.

A performance analysis was done using the development laptop (powered by a 4-core CPU) under minimal interference from other programs and under the same conditions. The test was performed using the Windows OS and also under the Linux OS through the use of dual-boot. When running the OpenFOAM toolchain using solver *rhoCentralFoam* on a coarse mesh, the results showed that Docker on Windows ran 5.13% worse compared to the same execution directly on WSL, with an execution time of 3208.35 s and 3051.76 s respectively. When the same simulation was executed in Linux using dual-boot, the results were worse with a time of 3243.62 s. This could be the result of optimized drivers installed in the Windows OS, but not present in Linux.

2.3 Workflow

The sum of all the simulations needed to aerodynamically characterize a rocket can be large, as seen in Table A.1 and Table A.2, and it can become difficult to manage that many simulations. Foreseeing this issue, a similar approach to CFD macros, already popular in other CFD tools, has been developed for OpenFOAM based on a Python library for templating called Jinja2. An OpenFOAM workspace can be tailored to a specific study-case by configuring parameters like flow speed, pressure, and more.

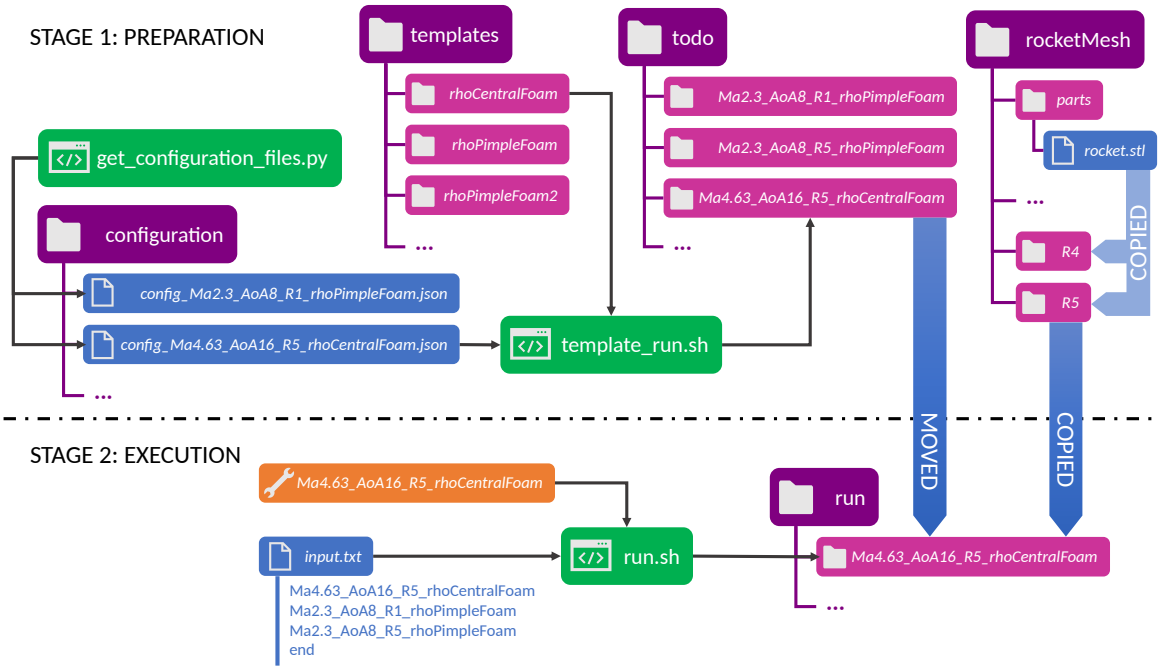


Figure 2.1 Semi-Automatized Workflow for Pre-Processing and Execution of OpenFOAM Simulations Using of Templating Enabled by Jinja2, Python, Bash, and *JSON*

Instead of creating an OpenFOAM workspace for each study-case, templates can be used as a base project from which specific parameters can be configured. Templates are OpenFOAM workspaces containing all required folders and scripts for execution (*constant, systems, 0.orig, Allrun, ...*) but which lack some parameters to be able to execute. Jinja2 fills in the gaps using a *JSON* configuration file with the additional parameters for each study-case. Jinja2 is executed using the application *jinja2-cli*, which in conjunction with *bash*, creates a semi-automatized workflow. Figure 2.1 depicts a schematic view of the semi-automatized workflow for OpenFOAM mentioned. During stage 1, a Python script (*get_configuration_files.py*) generates every *JSON* configuration file into the *configuration* folder. Then, the *bash* script *template_run.sh* takes three arguments: the path to the configuration file, the path to the template folder, and the location and name of the generated OpenFOAM

workspace. Note that each solver can have different versions with *rhoPimpleFoam* and *rhoPimpleFoam2* in the *template* folder being an example of it. Meshes are pre-calculated and stored in the folder *rocketMesh* to later be copied into the executing OpenFOAM simulation.

At execution, a bash script named *run.sh* controls the order of execution. All the simulations waiting to be executed are temporarily stored in the *todo* folder. The *run.sh* script takes two arguments: a list with the execution order and the first element to be executed (any element of the list). The script then moves the folder from *todo* to *run* and executes OpenFOAM scripts *Allclean* and *Allrun*. The latter instructions can be found in Section 5.

Some negative aspects of using template projects are that they can hinder the user's ability to personalize each study-case, as making changes in the template can become tedious or be incompatible with the necessities of other study-cases. Some extreme study-cases may require manual adjustments before execution. The *todo* folder fulfills this need by allowing user intervention before execution.

For automatized post-processing, Python is used to generate a series of images aimed at providing users with an initial insight into the flow solution without the need for Paraview. To do so, Python accesses ParaView through the *paraview.simple* API and saves the desired images to the OpenFOAM workspace. Additionally, convergence plots of residuals and force coefficients are exported as *PNG* files and saved primarily in the OpenFOAM workspace. These files are accompanied by a text file containing an simulation report generated automatically. An example of these files can be found in the Annex (refer to Section A). Figures A.1, A.2, and A.3 display all the images generated for preliminary evaluation of results. Figure A.4 presents the aforementioned convergence plots, which are not used for real-time monitoring as they are only available after solver convergence. Finally, Figure A.5 showcases the report automatically generated with all relevant parameters to assess the success of each simulation. During the automated post-processing step, all the information gathered in each report is compiled into a common table, which can be found in the Annex (refer to Table A.1 and Table A.2).

3 Validation Strategy

Papers [8, 9] provide the aerodynamic coefficients in a wind-tunnel environment for a meteorological rocket known as ARCAS, developed in the 1960s by NASA. The study tested two configurations of ARCAS, one 1.3 m long and another shorter version of 1 m in length. Both configurations are tested with fins at different incidence angles or without them. For the work's validation analysis, the fins are removed, and the shorter version is chosen, this geometry can be found in Figure 3.2. The support structures for the fins are also neglected, as it is difficult to conclude if they remained on the rocket once the fins were removed from the available pictures in [8, 9]. Three different aerodynamic regimes, subsonic, transonic, and supersonic, have been selected to benchmark the entirety of the flight envelope for unpowered flight, given ARCAS's available data. Table 3.1 shows the atmospheric conditions calculated for each Mach number (Ma). All calculations have been done while maintaining a constant Reynolds number (Re) of 562 000 , which remains constant through out all the wind-tunnel tests in [8, 9]. The aerodynamic coefficients used for the comparison with wind-tunnel data are: the axial force coefficient (C_A), the normal force coefficient (C_N), and the pitch moment coefficient ($C_{m_{\text{pitch}}}$). Calculations were done using a Cross Sectional Area (A_c) of 1.283×10^{-3} m^2 and Characteristic Length (l) of 57.15×10^{-3} m, and these values remained unchanged for every Angle of Attack (AoA) and Ma. The conditions considered for the benchmark can be found highlighted in the Ma column in Table 3.1. Each set of atmospheric conditions is studied at three different AoA of 0 deg, 8 deg, and 16 deg.

Height [m]	Temperature [K]	Pressure [Pa]	Velocity [m/s]	Mach Number [-]	Density [kg/m^3]
3763	263.71	63665.02	195.33	**0.60**	0.84
6618	245.18	43444.02	251.12	0.80	0.62
7730	237.96	37124.20	278.32	0.90	0.54
8229	234.73	34539.35	291.78	0.95	0.51
8699	231.68	32259.85	305.14	**1.00**	0.48
10334	222.81	25538.75	359.08	1.20	0.40
12185	220.37	20211.57	446.39	**1.50**	0.32
13411	218.75	16683.14	533.69	1.80	0.26
14737	217.00	12866.91	679.21	**2.30**	0.21
16634	216.65	9959.33	873.41	2.96	0.16
18544	216.65	7445.39	1168.49	3.96	0.12
19362	216.65	6368.74	1366.18	**4.63**	0.10

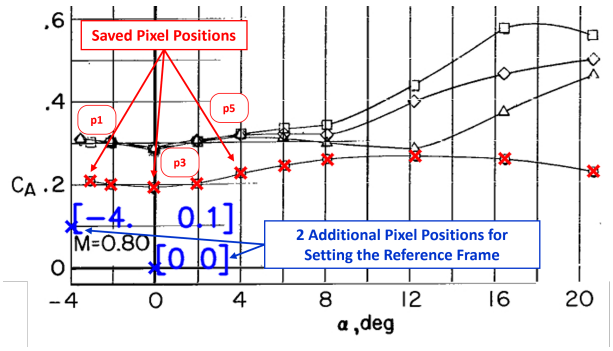
Table 3.1 Calculated Atmospheric Conditions (Temperature, Pressure, Density, Velocity) for each Ma Number based on Constant Re Number using a Python Script

The aerodynamic coefficients from [8, 9] are extracted using an image processing script due to the poor condition in which the documents presents the data, Figure 3.1 shows an example of this process. Pages with relevant quantities are selected and converted to images. Then relevant points in the plots are recorded in a separate document as pixel positions. Two additional pixels on the plot associate pixel positions with plot values, so that the recorded points in the document can be extrapolated. A Python script executes this function and uses splines to interpolate the remaining values.

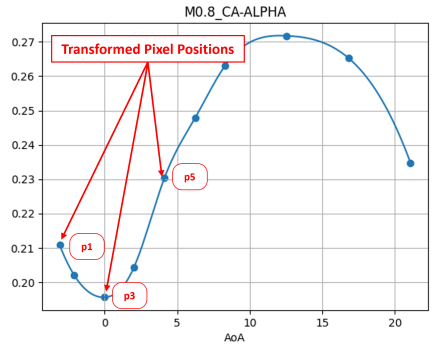
It should be noted that the covariance of the extracted data remains high for a variety of reasons. Starting with the pixel assignment, in this step each pixel assigned carries a big covariance due to the low resolution of the images. Moreover, plots in the images can be distorted or misaligned, which leads to increasing errors. The biggest variation during data extraction will happen in the $C_{m_{\text{pitch}}}$ coefficient plot, as the slope vs. AoA is the

highest. Finally, as discussed in the data sources [8, 9], wind-tunnel support structures have an impact on the recorded aerodynamic coefficients. The rocket model used for CFD analysis does not contain any wind-tunnel support structure, leading to a change in the aerodynamic coefficients obtained.

For analysis during engine ignition, a software named *RASaeroII* provides the C_A, C_N, and $C_{m_{\text{pitch}}}$ coefficients for powered flight for the ARCAS rocket. The values for unpowered flight do closely match the extracted data from [8, 9], thus providing a positive insight into the validity of the data for powered flight. The drag reduction is due to the addition of a rocket plume at the nozzle exit, resulting in higher pressures at the base of the rocket.



(a) Original C_A vs. AoA Plot of the wind-tunnel Results from [8]



(b) Resulting C_A vs. AoA Plot after Data Extraction Process

Figure 3.1 Example of the Data Extraction Process used to Retrieve the Aerodynamic Coefficients of the wind-tunnel Tested Rocket Geometry in [8] and [9] for Ma 0.8

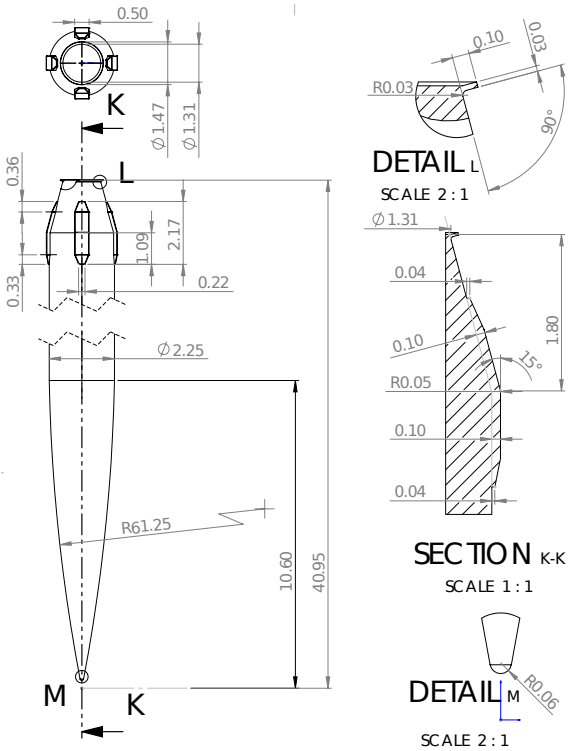


Figure 3.2 Schematics of the wind-tunnel Tested Rocket Geometry from [8, 9] Reproduced using SolidWorks - Rocket Configuration 1 with no Fins - **(units inches)**

4 Mesh Generation

4.1 Meshing Software Selection

Both preinstalled meshing tools in ESI were tested, snappyHexMesh (SHM) and CfMesh. Moreover, other free meshing tools were considered, such as SALOME, as OpenFOAM has the capability to import meshes from multiple sources, including those from paid-CFD solutions like ANSYS (Fluent) or Siemens (Star-CCM+). Table 4.1 contains a comparison between the two main meshing solutions considered, SHM and CfMesh.

SHM and CfMesh work on similar three-step principles made up of castellation, snapping, and refinement-layer addition. During castellation, cells with the biggest permissible size fill the study domain, then cells become progressively smaller around finer surface features by splitting in groups of 8 identically sized cells. This process increases the cell-level and can happen automatically based on feature refinement parameters, or the user can manually set the cell-level in indicated mesh regions. After that, snapping takes place, and cells adjacent to patches are snapped to smooth the surface. Snapping works in conjunction with an optimizer set to maximize mesh quality. Finally, refinement layers are added by splitting cells next to the walls based on the amount of refinement points and the provided Growth Ratio (GR). Close-by cells can be displaced to provide smoother resulting refinement layers.

Pros	Cons
• Default OpenFOAM meshing tool present in all distributions (OF and ESI). • Advanced castellation strategies that work together with blockMesh, allowing for greater control.	• Difficulty in adding refinement points to solve the boundary layer; more than 5, and the refinement points tend to collapse. • Prone to snapping imperfections. • Single-core computing, no multi-threading available.

(a) Pros and Cons of snappyHexMesh (SHM)

Pros	Cons
• Addition of refinement points is very reliable, with no refinement layer collapse. • Installed by default in ESI. • Multi-core enabled, very fast computing.	• Further software updates are only available in CfMesh+, which is a paid software. • Much fewer castellation controls compared to SHM, with no use of *blockMesh*.

(b) Pros and Cons of CfMesh

Table 4.1 Comparison between CfMesh and SHM

Meshing controls, particularly in CfMesh, are very limited, which can become an issue, particularly during the addition of refinement layers. As can be seen in Figure 4.2b, these poor control options result in an abrupt cell height change between the last refinement layer cell and its neighbor, something that cannot be avoided. Additionally, CfMesh does not currently support advanced settings for refinement layer addition (e.g. total refinement layer height), making the process highly reliant on the local cell-level. Smoothing parameters can help transition between different cell-levels at the refinement layer. This parameter gradually increases or decreases the size of elements as it transition to a different cell-level in a neighboring region of the refinement layer. For more information on CfMesh capabilities, see [10]. Automatic refinement in both SHM and CfMesh is not advisable if manual refinement is feasible. CfMesh presented issues when using the automatic refinement in junction points between patches, e.g. between the rocket and the symmetry plane or in the corners of the domain box. In these regions, the maximum cell-level is reached, leaving behind unnecessary small cells.

CfMesh offers a wide range of cell topologies, but based on a performance study discussed in [11], the most efficient calculations are provided by the cartesian topology. Based on this cell topology, the best domain geometry to fit the cells is a box. The dimensions of the domain are 10 m long, 2 m in front of the rocket tip and 8 m behind it, and 2 m in width on each side. Comparatively to the size of the rocket, based on Figure 3.2, the domain is 7.36 times the length of the vehicle and 70 times its width. Other domain geometries studied were cylindrical with and without spherical inlets to reduce unnecessary cells in the corners of the domain, which are geometrically the farthest from the rocket. CfMesh's cartesian cells cannot be tailored to these circular patches, leaving behind deformed cells. On the other hand, SHM can use *blockMesh* to optimally orient the cells before starting the three-step meshing process.

Independently of the selected option, both SHM and CfMesh require an STL file or, preferably, an FMS file for CfMesh. The initial geometry provided with the Computer Aided Design (CAD) software is a $STEP$ file, thus a triangulation process must take place to obtain the STL file. SALOME, apart from being an alternative to SHM and CfMesh, excels at surface preparation. Figure 4.1 shows the surface at the tip of the rocket that combines structured and unstructured strategies to obtain the best surface and keep the weight of the STL file low. A structured mesh is chosen where possible, while unstructured meshes are used for regions with complicated geometries, thus minimizing the deflection error and guaranteeing surface smoothness. Automatic triangulation tools can cause triangulation errors, leaving behind spikes that cause jumps in the deflection error along the surface. These anomalies are later the cause of bad snapping results in both SHM and CfMesh. During the generation of the STL files, it is possible to define regions that can be key during the meshing process later on.

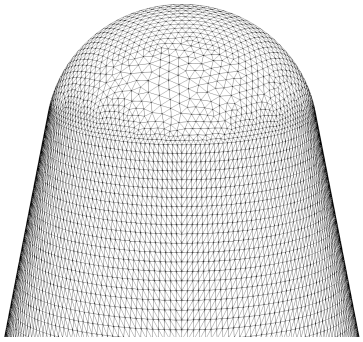
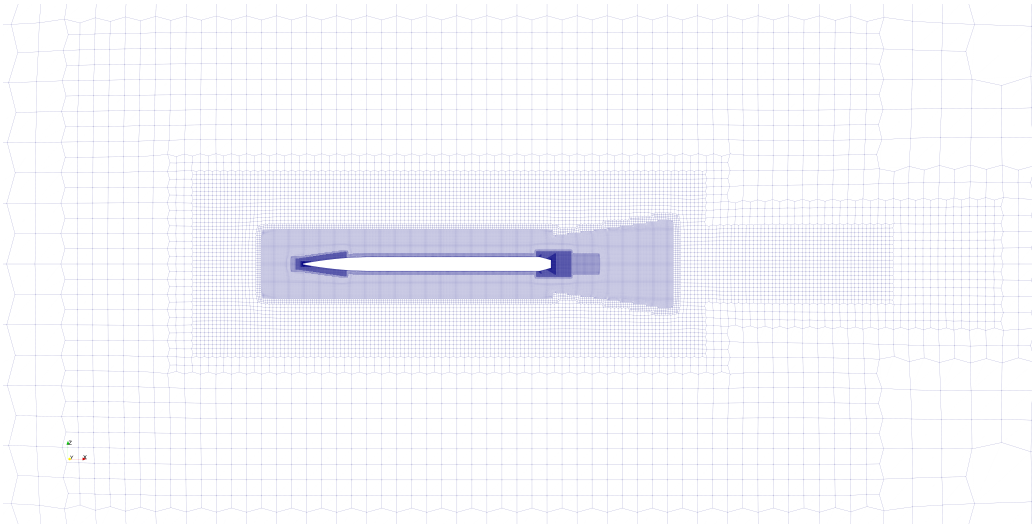
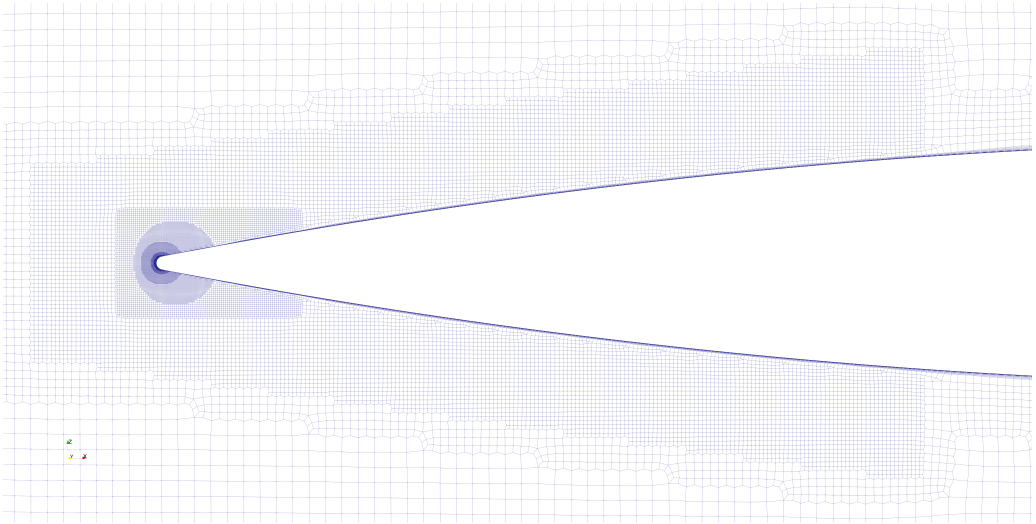


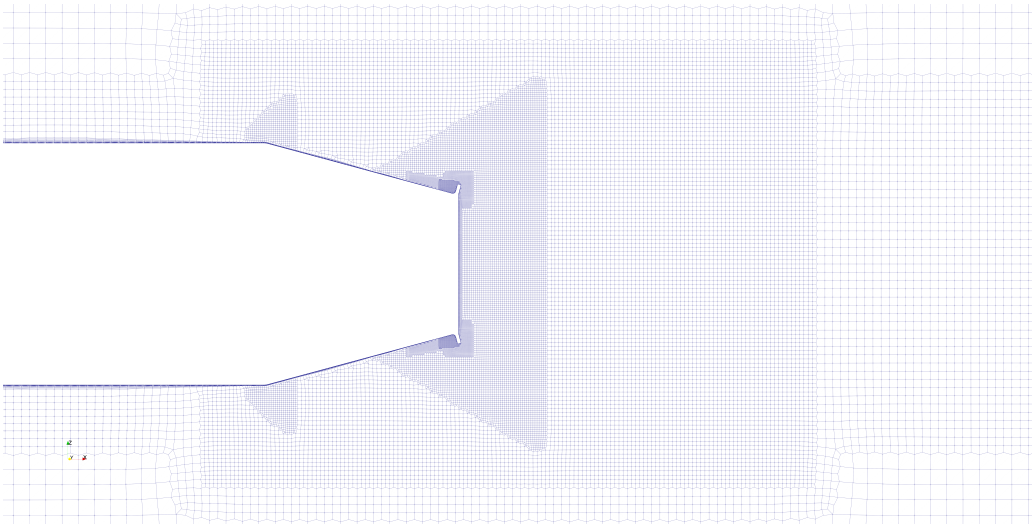
Figure 4.1 Zoom-In of the STL Surface Generated using SALOME at the Tip of the Rocket - Transition Between Structured and Unstructured Surface Mesh



(a) Zoomed-Out View



(b) Zoomed-In View Around the Tip of the Rocket



(c) Zoomed-In View Around Aft of the Rocket

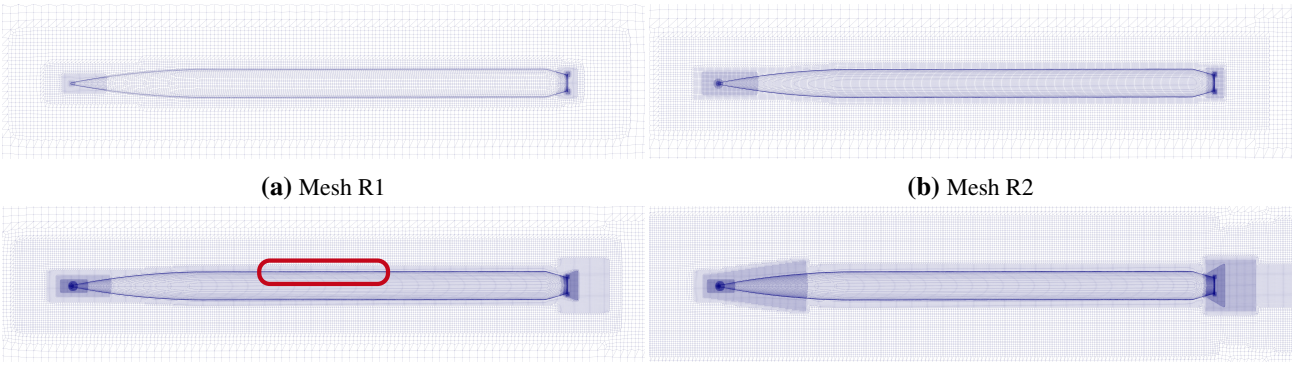
Figure 4.2 View of Cell Distribution Along the Symmetry Plane for Mesh R5

4.2 Mesh Convergence Study

A mesh convergence study is necessary to assess the refinement level required to accurately capture the aerodynamic characteristics of the rocket. However, due to computational limitations, a maximum cell refinement of approximately 7 million cells is set. Other mesh quality assessments to consider include the quality of the rocket's surface, feature size, and Non-Dimensional Wall Spacing (y^+) values. The main direct factors for controlling y^+ values are the number of refinement points and the GR.

Mesh	nº Cells	nº Ref. Points	GR	nº Surface Cells
R1	184.254	5	1.4	13.550
R2	1.129.954	15	1.4	38.228
R3	2.263.004	15	1.3	61.371
R4	5.246.305	20	1.2	101.394
R5	6.512.727	20	1.2	142.380

Table 4.2 Refinement Characteristic of all the Available Meshes from Coarser (R1) to Finer (R5)

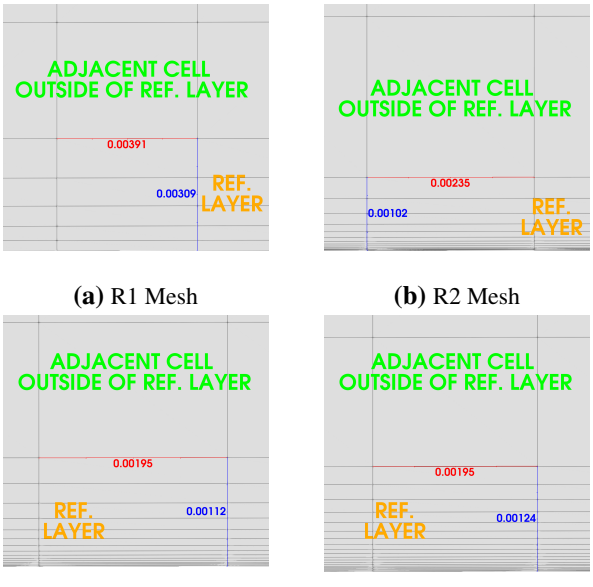


(a) Mesh R1 **(b)** Mesh R2

(c) Mesh R3 **(d)** Mesh R4

Figure 4.3 Zoomed-Out View of the Different Mesh Resolutions (R1, R2, R3, R4) - Red box in Mesh R3 indicates the zoom-in region for Figure 4.4

In Table 4.2 one can find the main parameters describing the level of refinement of each mesh resolution (R1, ..., R5) used in the convergence study. This parameters are the total number of cells used (nº Cells), and other parameters relative to the refinement layer resolution like GR, number of refinement points used (nº Ref. Points), and the number of cells found at the surface of the rocket (nº Surface Cells). The resolution increase between meshes can be observed in Figure 4.3 which shows meshes R1 to R4 viewed from the symmetry plane. Finally, a better understanding of the level of resolution at the refinement layer can be grasped at Figure 4.4. Note that images in Figure 4.4 are close-ups of the tubular region of the rocket indicated in Figure 4.3c using a red box. The dimensional relations between the refinement layers and the adjacent cells remain when observing the surface at other regions of the surface where the cell-level changes.



(a) R1 Mesh **(b)** R2 Mesh

(c) R3 Mesh **(d)** R4 and R5 Mesh

Figure 4.4 Close Up of Refinement Layers at the Tubular Region of the Rocket for all Mesh Resolutions **(in meters)**

Equation (4.1) is used to calculate the turbulent Boundary Layer Thickness (δ) along a flat plate channel. It results in a δ of 1.50×10^{-3} m for a given Re number of 562 000 and a Hydraulic Diameter (D) of 57.15×10^{-3} m. Based on this result, it is estimated that the smallest cell element should have a size on the order of 1×10^{-5} m to obtain ${y^+}_{max} < 1$.

$$\delta = 0.37\frac{D}{Re^{\frac{1}{5}}} \tag{4.1}$$

An example of the y^+ values obtained for a *rhoCentralFoam* simulation is available in Figure 4.5 also showing additional information on the cell-level at each zone for mesh R5. The size of elements can be calculated based on cell size for level 0, which is 0.5 m, and dividing by powers of two based on the number of cell-levels. Specific regions, like the nose tip and the rocket base, have refinement levels of 14 and 12 respectively, which correspond to element sizes of 3×10^{-5} m and 1×10^{-4} m, leading to very low y^+ values. On the other hand, the body of the rocket has y^+ values closer to 1 with element sizes of 2×10^{-3} m as seen in Figure 4.4d. Note that further refinement occurs during the refinement layer addition. For mesh R5, cell elements in the first layer of the refinement layer are 32 times smaller due to splitting at a GR of 1.2 to generate 20 refinement points. The observed ${y^+}_{max}$ across different atmospheric conditions and AoA shows a range from 0.001 to 1.0 for mesh R5. Moreover, the coarse mesh R1 sees a larger range from 0.7 to 70 Results can be found in Table A.1 and Table A.2 in the Annex (see Section A).

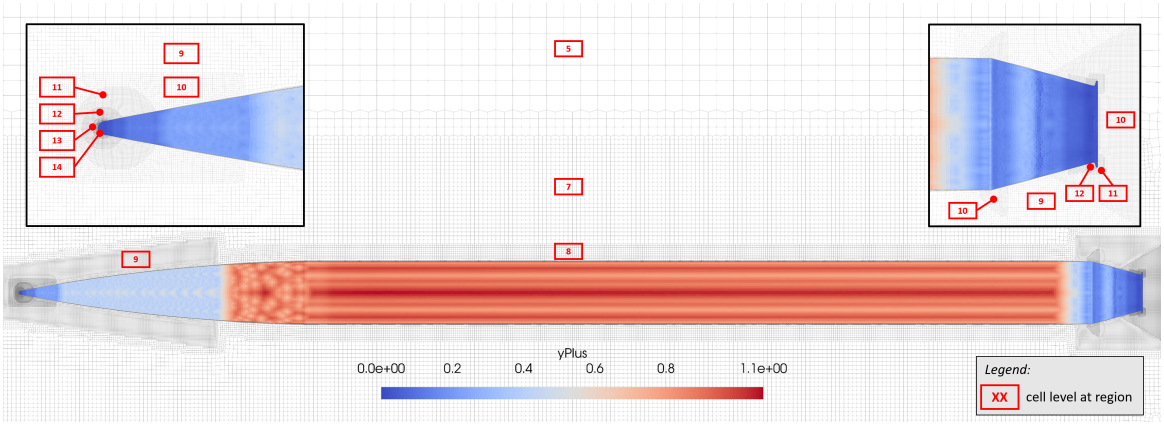
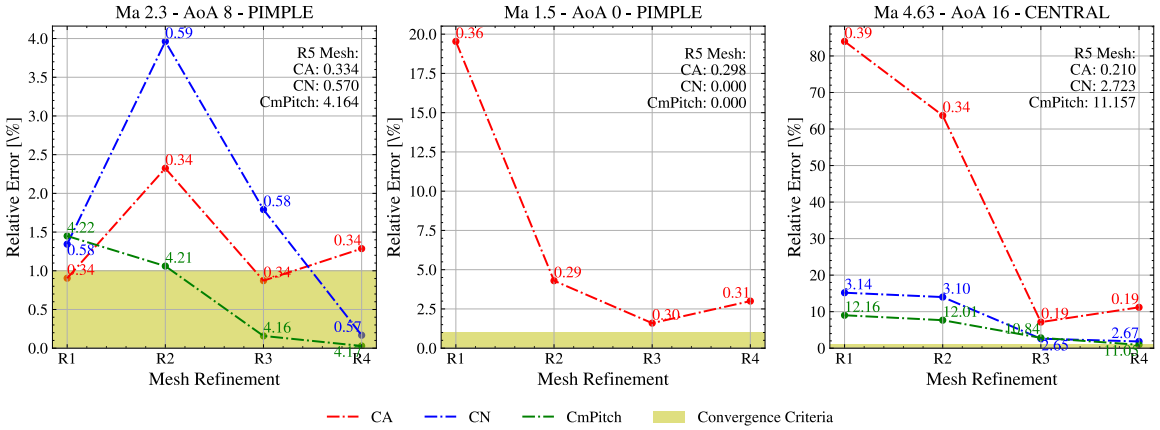
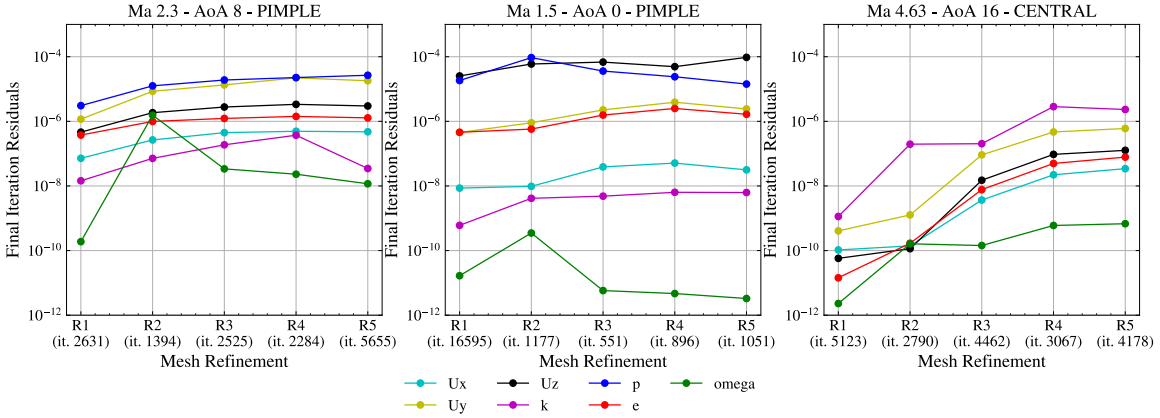


Figure 4.5 y^+ Values Along the Surface of the Rocket for Simulation Ma 1.0 - AoA 0 deg - Mesh R5 - *rhoCentralFoam* with Cell Refinement Levels around the Rocket for Mesh R5

Figure 4.6 shows the three convergence studies done, one using *rhoCentralFoam* and *rhoPimpleFoam* for the remaining. The atmospheric conditions chosen for the study are representative of the entire flight envelope. The aim of the mesh convergence study is to find the minimum cell refinement required to obtain a relative error in the aerodynamic coefficients (C_A, C_N, $C_{m_{\text{pitch}}}$) below 1%. The aerodynamic coefficients from the most refined mesh (R5) are set as the ground truth used to calculate the relative error shown in Figure 4.6a for the other meshes (R1, R2, etc.). Unfortunately, from the results obtained in Figure 4.6a, it is uncertain if the mesh R5 is good enough to accurately solve the cases, as none of the aerodynamic coefficients reaches a plateau. Nevertheless, as expressed previously, it is not possible to further increase the number of cells due to imposed computational limitations. To draw a conclusion on mesh R5, a 10-12 million cells mesh should be resolved, thus the relative error for mesh R5 would be computed using R6 results. Additionally, Figure 4.6b shows the final residual values for each mesh resolution, in the x-axis the number of iteration until convergence can be found.



(a) Progression of the Relative Error of Aerodynamic Coefficients (C_A, C_N, $C_{m_{\text{pitch}}}$) Calculated Based of R5 Values for Three Distinct Mesh Convergence Studies - Point labels indicate the aerodynamic coefficients at each mesh refinement



(b) Progression of the Residuals for Three Distinct Mesh Convergence Studies - Number of iterations to convergence in x-axis

Figure 4.6 Results of Convergence Study for Cases Ma 2.3 - AoA 8 deg - *rhoPimpleFoam* & Ma 1.5 - AoA 0 deg - *rhoPimpleFoam* & Ma 4.63 - AoA 16 deg - *rhoCentralFoam*

Regarding the results from Figure 4.6a, only the convergence study on the left (Ma 2.3 - AoA 8 deg - *rhoPimpleFoam*) shows a certain amount of convergence based on the 1% criteria previously discussed. In this case, the relative error for the C_A coefficients remains above the 1% threshold at 1.3%. The remaining studies in Figure 4.6a all show an increasing degree of convergence as relative errors tend to decrease. However, from all the aerodynamic coefficients measure, C_A stand out in all three studies as having a delayed convergence with respect to the other coefficients (C_N and $C_{m_{\text{pitch}}}$). For the convergence study found in the center of Figure 4.6a (Ma 1.5 - AoA 0 deg - *rhoPimpleFoam*), it initially shows a descending trend into the 1% convergence criterion region, but when reaching mesh R4 a final increase to a relative error of 2.5% occurs. In particular, the convergence study on the right (Ma 4.63 - AoA 16 deg - *rhoCentralFoam*) shows high relative errors for meshes R1 and R2, but as the mesh resolution increases to R3 and R4 a significant drop in relative error for all aerodynamic coefficients is experienced. However, the relative error for C_A in meshes R3 and R4 stays high at 10%, while the other coefficients (C_N and $C_{m_{\text{pitch}}}$) drop to values similar as the ones seen in the other convergence studies. The higher relative error observer for meshes R1 and R2 is the product of different interpolation schemes used as the ones for meshes R3 and R4. As will be explained in Section 5.5.1, solver *rhoCentralFoam* is initialized using more dissipating interpolation schemes. In the case of mesh R1 and R2 at Ma 4.63 and AoA 16 deg the transition to higher order interpolation schemes is not feasible due to the low mesh resolution. Moreover, the perceived drop in residuals for this convergence study, see Figure 4.6b, could be the results of using different interpolation schemes.

A hypothesis as to explain the tendency of C_A to higher relative errors would be the sensibility of the CFD analysis to unsteady flow cases. Based on post-processing analysis, the aft of the vehicle has an unsteady recirculation zone which might be the cause of larger relative errors for C_A than other coefficients as the pressure distribution in the aft would be affected by it, and the pressure distribution in the aft affect the C_A coefficient the most. The unsteady nature of the flow at the aft might not be fully captured using the steady state time scheme, thus leading to small variations in the relative error for C_A and no clear tendency to decrease its relative error in comparaison to the other coefficients (C_N and $C_{m_{\text{pitch}}}$), thus following a more chaotic approach to convergence. As a reminder the selection of a steady state time scheme will be discussed in Section 5.4. The use of steady state analysis is justified due to computational limitations and the foreseen addition of engine exhaust gases at the aft of the vehicle, which should mitigate the unsteadiness problem.

Further analysis would be required to prove the unsteady flow hypothesis, something further discussed in Section 5.4. Also, the necessity for finer meshes at greater Ma numbers where convergence results have been comparatively worse should be explored. Finally, the addition of rocket exhaust gases would require a new convergence study to take place following the procedure presented here. Current results are just an indication, but not an assurance of enough mesh resolution, even when using the finest mesh R5.

5 Simulation Setup

As stated during the introduction, the project aims to quantify the accuracy of two solvers: *rhoPimpleFoam* a pressure-based solver and *rhoCentralFoam* a density based-solver over the entirety of the flight envelope which cover three distinct aerodynamic regimes (subsonic, transonic, and supersonic). Both solvers are single-species ideal gas, unsteady, and will incorporate the turbulence model *k-ωSST*, which has been proven in [4] to offer the best results. Moreover, both solvers have a time-transient formulation, which contrasts with the need to study steady-state flow. To accelerate convergence, the use of Local Time Stepping (LTS) as a time scheme will be discussed in Section 5.4. For the viscosity model, Sutherland's law will be implemented.

5.1 Selected Solvers

In aims of providing a comprehensive toolchain capable of characterizing the flow in all three aerodynamic regimes aforementioned (subsonic, transonic, and supersonic), the project intends to use two distinct solvers working on pressure-based and density-based algorithms. While pressure-based algorithm have been intended in the past to solve subsonic flows, and density-based solver for supersonic flows, current algorithms in both families are extending their reaches to wider range of Ma numbers. For a more detailed description of both solvers reference [1], which dives deeper into the mathematical formulation of solvers *rhoPimpleFoam* and *rhoCentralFoam*.

In short, algorithms solve a system of three differential equations which express the physical laws of conservation of mass (continuity), conservation of momentum and conservation of energy. Pressure-based and density-based solvers alike, obtain the velocity field by solving the momentum equation. However, density-based algorithms obtain the density field from the continuity equation and the proceed to calculate the pressure field from it. On the contrary, pressure-based algorithms combine the continuity and the momentum equation to initially obtain the pressure, from which they derive the density using an equation of state.

The solver *rhoPimpleFoam* relays on a mix between two pressure-based algorithm, Semi-Implicit Method for Pressure Linked Equations (SIMPLE) and Pressure Implicit with Splitting of Operators (PISO), which combine to form a new algorithm named PIMPLE, name given based on the acronyms of its parent algorithms. The SIMPLE algorithm was introduced in 1972 by L. S. Caretto [12] and is a steady state solver for compressible and incompressible flows. While the PISO algorithm, which was introduced in 1982 by R. I. Issa [13], is a transient algorithm compatible with compressible and incompressible formulations of the transport equations. However, in OpenFOAM it is implemented under the solver *pisoFoam*, that only allows incompressible cases. PIMPLE remains a pressure-based algorithm which in its OpenFOAM implementation can solve transient flows for compressible and impressible fluids depending on the solver, with *pimpleFoam* and *rhoPimpleFoam* respectively. As will be discussed in Chapter 6 the limitation of pressure-based solvers become apparent at high Ma numbers, with $Ma < 2.3$, when solutions become nonphysical.

Alternatively, the proposed density-based solver, *rhoCentralFoam*, is a driven based on the central-upwind schemes of Kurganov and Tadmora. This algorithm aims to resolve a Riemann problems using approximations as to make the solution practical. Many types Reimann solver exist, this particular one belongs to the *central schemes* class, and is an improvement on the Lax-Friedrichs scheme.

5.2 Boundary Condition Types

Both *rhoPimpleFoam* and *rhoCentralFoam* simulations use the same boundary condition types. Pressure, velocity, and temperature values are set based on the calculations from Section 3 in Table 3.1. As seen in Figure 5.1, the domain is composed of five patches. A symmetry wall divides the computational domain into two parts, reducing the computational needs. Due to the lack of complex geometric features like fins or other potential asymmetries, the rocket is axially symmetrical, and thus all AoA can be studied within the symmetry wall.

As observed in Table 5.1a for supersonic regimes, *far field* and *outlet* patches require a *waveTransmissive* boundary condition to avoid issues with shock wave reflections at these patches. This boundary type works in conjunction with the *advective* type for temperature. The *symmetry* condition is preferred compared to *symmetryPlane*, which can cause problems if the patch is not perfectly flat. All boundary conditions of type *inletOutlet* and *freeStream* (*supersonicFreeStream* included) function as mixed boundary conditions. Initially, the patch is set to a user-specified *inletValue* using a boundary condition of type *fixedValue*. As long as the flow vector points inwards to the domain, this condition remains the same. If during the calculation, the flow vector points outwards of the domain, then the boundary condition switches to *zeroGradient*.

Table 5.1b presents the boundary conditions required for subsonic and transonic regimes.

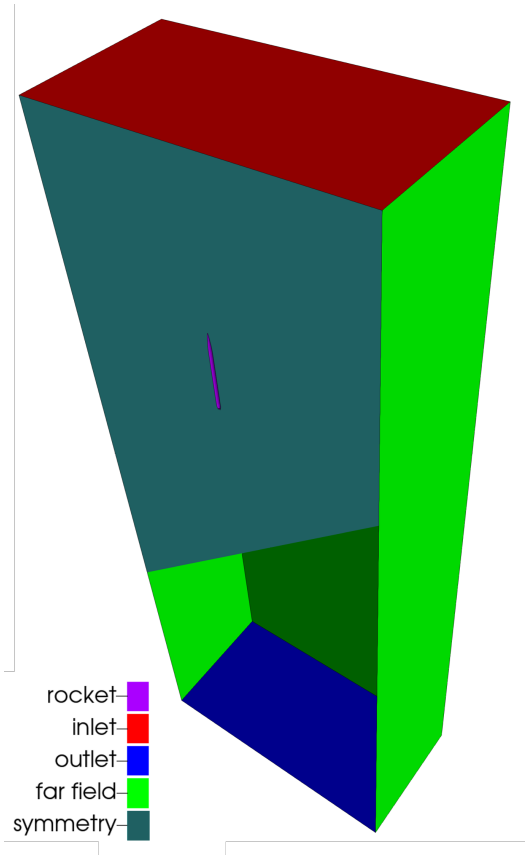


Figure 5.1 Schematic of Patches For All Meshes (R1, ..., R5) with Cut View of the *symmetry* Patch to Observe the *outlet* Patch

patch	pressure	temperature	velocity
inlet	*fixedValue*	*fixedValue*	*fixedValue*
outlet	*waveTransmissive*	*advective*	*inletOutlet*
rocket	*zeroGradient*	*zeroGradient*	*noSlip*
far field	*waveTransmissive*	*advective*	*supersonicFreestream*
symmetry	*symmetry*	*symmetry*	*symmetry*

(a) Boundary Condition Types for Solvers *rhoPimpleFoam* and *rhoCentralFoam* During Supersonic Simulations

patch	pressure	temperature	velocity
inlet	*freestreamPressure*	*inletOutlet*	*freestreamVelocity*
outlet	*freestreamPressure*	*inletOutlet*	*freestreamVelocity*
rocket	*zeroGradient*	*zeroGradient*	*noSlip*
far field	*freestreamPressure*	*inletOutlet*	*freestreamVelocity*
symmetry	*symmetry*	*symmetry*	*symmetry*

(b) Boundary Condition Types for Solvers *rhoPimpleFoam* and *rhoCentralFoam* During Subsonic and Transonic Simulations

Table 5.1 Two Configurations (Subsonic and Transonic, and Supersonic) for Pressure, Temperature, and Velocity Boundary Condition Types at Every Patch

For the series of simulations presented in this project, it is important to remark that the skin of the rocket is considered to be adiabatic, leading to a *zeroGradient* boundary type for the temperature at the rocket's surface. This boundary condition is apt for this study as the wind-tunnel tests were performed in similar conditions, but in a more realistic this approximation would significantly deviate from reality. In general, during launch, a rocket's wall temperature is not adiabatic. For instance, many sections of the rocket might contain liquid oxygen or other propellants at significantly low temperatures. In a more developed CFD model, the adiabatic consideration should be dropped in benefit of closer conditions to flight.

5.3 Turbulence Model Implementation

To model an external flow, the best RANS turbulence model is *k-ω*, with *k-ωSST* being an improved hybrid version that combines *k-ϵ* and *k-ω* formulations. Boundary conditions regarding turbulence modeling can be highly confusing in OpenFOAM due to the lack of clear documentation, updates, and a large number of options. This section aims to provide a general guide on which setup to use. Note that the turbulent boundary conditions presented in this section could be used with other turbulence models.

5.3.1 Free-stream Conditions

In addition to the boundary conditions specified in Tables 5.1a and 5.1b, other fields are required for *k-ωSST*. These fields include Turbulent Kinetic Energy (k), Specific Dissipation Rate (ω), Turbulent Viscosity (ν), and Turbulent Thermal Diffusivity (α).

The inlet values and initial internal field for k and ω are calculated based on a set of equations provided in [14]. The assumed atmospheric conditions include a low Turbulent Intensity (I) of 0.8% based on range provided by [14] and a uniform average wind speed of 15 m s [15], resulting in a k of 0.0314 $\frac{\mathrm{m}^2}{\mathrm{s}^2}$.

$$k = \sqrt{\frac{3}{2}}(U_\infty I)^2 \tag{5.1}$$

For ω, the Equation (5.2) for external flows is used [14]. Equation (5.2) is based on the Eddy Viscosity Ratio $\frac{\mu_t}{\mu}$, which takes a value of 0.7 based on range provided by [14] for external flows. Using this formula to compute all the different study-cases leaves ω ranging from 2300 to 300 $\frac{1}{\mathrm{s}}$ as the rocket ascents.

$$\omega = \frac{\rho_\infty k}{\mu}\left(\frac{\mu_t}{\mu}\right)^{-1} \tag{5.2}$$

Boundary conditions for *k-ωSST* will be discussed in the following section, where *inlet, outlet, far field, and symmetry* boundary types can be found in Tables 5.2 and 5.3. OpenFOAM provides alternatives to manually calculating the inlet conditions for the turbulence model. *turbulentIntensityKineticEnergyInlet* inherits from *inletOutlet* but automatically applies Equation (5.1) if provided with I by the user. Similarly, the boundary condition *turbulentMixingLengthFrequencyInlet* is used for ω and requires a mixing length. However, the initial internal fields for k and ω must be set separately.

5.3.2 Boundary Condition Types for Walls

Regarding the definition of boundary conditions on *noSplit* walls, it is imperative to consider the expected y^+ value. As seen in Figure 5.2, within the viscous sub-layer ($y^+ < 5$), the variation of y^+ matches the variation of u^+. This region is also known as the low Re region. For $y^+ > 30$, a logarithmic function better approximates the relation between y^+ and u^+. The buffer layer transitions between these two regions and can be challenging to model. Thus, to accurately model the boundary layer around the rocket, it is important to use a fine mesh with a ${y^+}_{max} < 5$.

In the search to maximize the result's accuracy, wall models should be avoided. A direct resolution of the boundary layer is preferred instead, even after accounting for the higher computational cost that this entails due to the use of finer meshes.

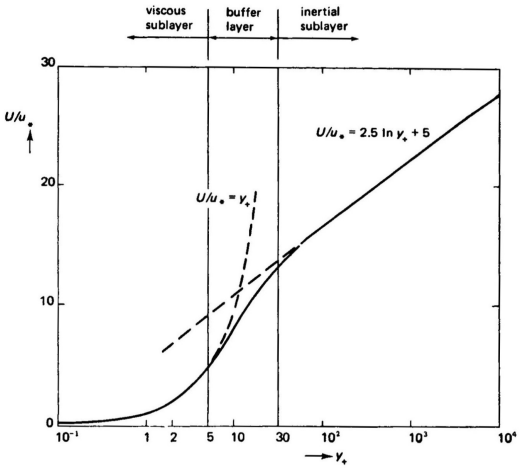


Figure 5.2 Law of the wall [16]

In this section, two candidate configurations for turbulent boundary condition types will be proposed. The configuration proposed in Table 5.3 should only be used in fine meshes which meet the criterion $y^+_{max} < 1$. On the other hand, Table 5.2 provides a configuration that fits all mesh qualities, including fine meshes and coarse meshes with $y^+_{max} > 30$. Initially, the configuration that uses wall models will be presented with all the mathematical functions used in conjunction with the *k-ωSST* turbulence model selected. After that, a set of boundary condition types that explicitly avoid the use of wall models will be discussed. Finally, the selected boundary condition types will be justified in the conclusion.

With a Wall Model

This set of boundary condition types is very necessary in OpenFOAM meshes due to reliance on castellation techniques by the main meshing solutions available in OpenFOAM, like SHM and CfMesh. This leads to y^+ values on coarse meshes to range between $y^+ < 1$ to $y^+ > 30$, thus the wall model needs to switch between the low and high *Re* regions as seen in Figure 5.2. To adapt the formulation, Table 5.2 shows a set of adaptable wall conditions for the *k-ωSST* turbulence model which can be used in coarse and fine meshes ($y^+_{max} < 1$).

patch	k	ω	ν	α
inlet	*inletOutlet*	*inletOutlet*	*calculated*	*calculated*
outlet	*inletOutlet*	*inletOutlet*	*calculated*	*calculated*
rocket	***kLowReWallFunction***	*omegaWallFunction*	***nutkWallFunction***	***calculated***
rocket ALT.				***alphatWallFunction***
far field	*inletOutlet*	*inletOutlet*	*calculated*	*calculated*
symmetry	*symmetry*	*symmetry*	*symmetry*	*symmetry*

Table 5.2 Boundary Condition Types for Turbulence Modeling using Wall Functions for *rhoPimpleFoam* and *rhoCentralFoam* in All Aerodynamic Regimes (Subsonic, Transonic, and Supersonic) - Valid for Low and High *Re*

The boundary condition *kLowReWallFunction* inherits from *fixedValue*. By identifying the region to which the cell belongs, *kLowReWallFunction* selects between two possible equations based on the law of the wall, as seen in Figure 5.2. For $y^+ < 11$, the viscous sub-layer approximation is used, while above this threshold, the log-law becomes a better model. From a theoretical perspective, the k becomes increasingly smaller as it approaches the wall. Thus, the implementation in *kLowReWallFunction* limits the value of k to a specified very small number to avoid numerical issues.

For ν, multiple boundary condition types exist, with the most common choice being *nutkWallFunction* as it makes use of k to calculate ν. This boundary condition adapts using the law of the wall, similar to *kLowReWallFunction*. ν is set to zero in the viscous layer, and Equation (5.3) takes over in the log-law region.

$$\nu_{t_{log}} = \nu_w \left(\frac{y^+ \kappa}{\ln(E y^+)} - 1 \right) \tag{5.3}$$

Finally, α uses the *alphatWallFunction*, which applies equation (5.4) directly.

$$\alpha_t = \frac{\nu_t}{Pr_t} \tag{5.4}$$

Both configurations make use of the same boundary condition, *omegaWallFunction* for ω, which makes use of the original Equation (5.5) first proposed in the *k-ω* model and found in [17]. Analogous to *kLowReWallFunction*, *omegaWallFunction* adapts its ω formulation according to the law of the wall. For the viscous-layer, ω is calculated as follows:

$$\omega_{vis} = \frac{6\nu_w}{\beta_1 y^2} \tag{5.5}$$

Note that this equation cannot be considered a wall model as it is an intrinsic part of the turbulence model *k-ωSST*. Finally, the boundary condition of type *calculated* present in both configurations for fields ν and α are used when a field is already fully defined by other boundary conditions. These fields have already been calculated in the previous steps of the solver.

Without a Wall Model

For optimal results, the use of a wall model needs to be avoided. This is only possible when the provided mesh is considered to have a high resolution at the boundary layer, thus being able to model the turbulence without the need for wall models. The boundary condition types in Table 5.3 deactivate all the wall models previously presented. However, in certain conditions which will be discussed later, the configuration aforementioned, in Table 5.2, becomes equivalent to the following configuration in Table 5.3.

patch	k	ω	ν	α
inlet	*inletOutlet*	*inletOutlet*	*calculated*	*calculated*
outlet	*inletOutlet*	*inletOutlet*	*calculated*	*calculated*
rocket	***fixedValue***	*omegaWallFunction*	***calculated***	***calculated***
rocket ALT.			***nutLowReWallFunction***	***alphatWallFunction***
far field	*inletOutlet*	*inletOutlet*	*calculated*	*calculated*
symmetry	*symmetry*	*symmetry*	*symmetry*	*symmetry*

Table 5.3 Boundary Condition Types for Turbulence Modeling Explicitly Removing the Use of Wall Functions for *rhoPimpleFoam* and *rhoCentralFoam* in All Aerodynamic Regimes (Subsonic, Transonic, and Supersonic) - Valid for Low *Re* Only

The k condition becomes a *fixedValue* boundary condition with a uniform extremely small number (1×10^{-20}). ν takes on the boundary condition type *nutLowReWallFunction*, which is a null uniform value with the benefits of having y^+ calculations implemented in the boundary conditions type, so to be able to measure y^+. Regarding the equivalency of the two proposed boundary condition sets in Table 5.3 and Table 5.2. In a fine enough meshes if ${y^+}_{max} < 1$, then Tables 5.2 and 5.3 are analogous since k takes an extremely small value in both cases and ν is set to null when used in the viscous sub-layer ($y^+< 5$) for the wall model in Table 5.2.

Conclusion

In conclusion, the setup presented in Table 5.2 can be considered equivalent to Table 5.3 for fine meshes (${y^+}_{max} < 1$). Even though the optimal results if achieved when no wall models are used, for this project, all simulations, independently of the mesh resolution used (R1, ..., R5), have made use of the configuration presented in Table 5.2 as its formulation can be applied to a larger range of mesh resolutions, leading to a more uniform OpenFOAM code between simulations. However, this choice does not undermine the solution of mesh R5 which intends to produce results without the use of wall models. As mesh resolution at the walls increases, wall models implemented in Table 5.2 transition to the configuration presented in Table 5.3. Based on the y^+ values presented in Chapter 4.2 for mesh R5 in Figure 4.5 it can be observed that wall models are not enabled during the execution of simulations using the R5 mesh.

5.4 Time Scheme - Euler vs. localEuler

With the objective of accelerating the convergence to steady-state, Local Time Stepping (LTS) is preferred compared to the transient time scheme alternative, *Euler*, a first order implicit time scheme. The LTS option is used in transient solvers like *rhoCentralFoam* and *rhoPimpleFoam* under the name *localEuler*, a first order time scheme activated in the *fvSchemes* configuration file. LTS transforms transient solvers into pseudo-steady-state solvers by applying different time steps to each cell. To control the time step, a maximum Courant (Co) number is selected (Co_{max}), and *maxDeltaT* sets the upper limit of the time-step. The time step difference between adjacent cells is limited to ensure continuity. To allow for faster convergence, more relaxed values for *rDeltaTSmoothingCoeff* and *rDeltaTDampingCoeff* are preferred. Default values for *rDeltaTSmoothingCoeff* and *rDeltaTDampingCoeff* can hinder the solver's ability to converge. For instance, this is observed when transitioning to finer meshes as cells take longer to diffuse into the new smaller mesh. Recommended values for LTS in *rhoPimpleFoam* at supersonic regimes are 0.1 for *rDeltaTSmoothingCoeff* and 0.9 for *rDeltaTDampingCoeff* [18].

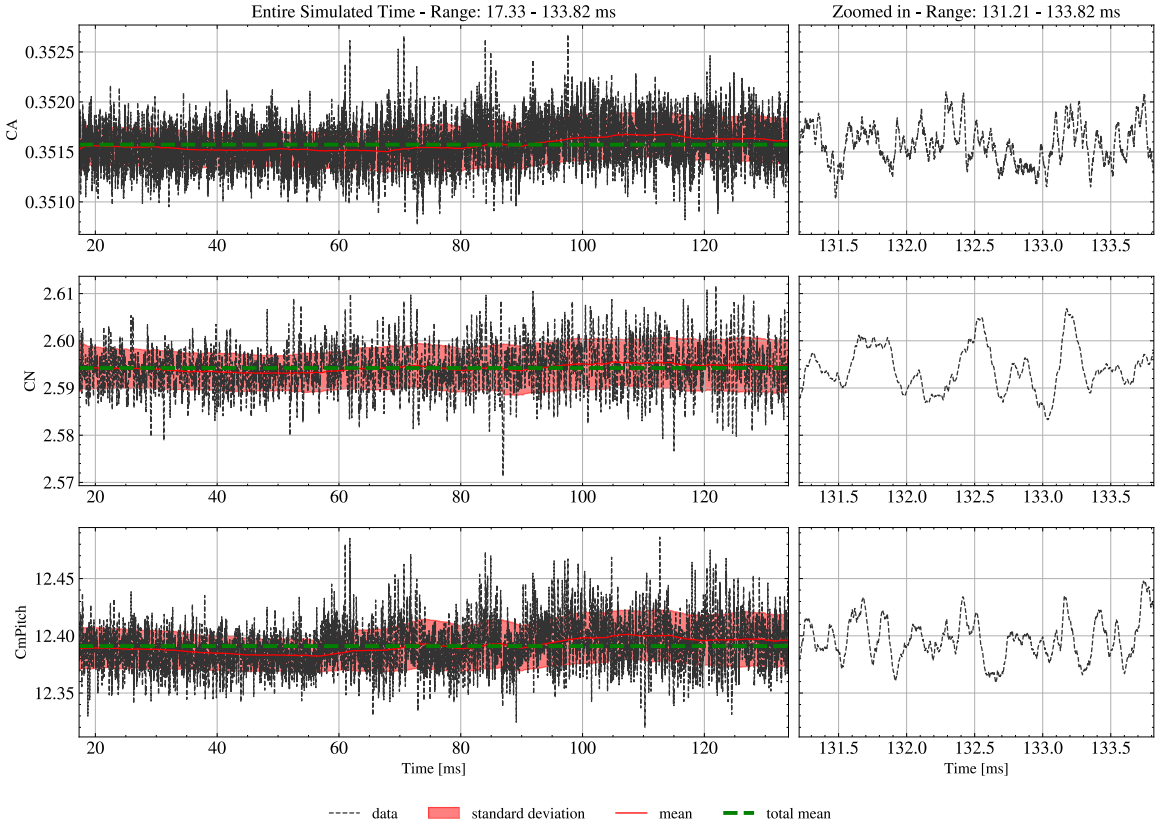


Figure 5.3 Change of Aerodynamic Coefficients (C_A, C_N, $C_{m_{pitch}}$) over Simulated Time for Simulation Ma 2.3 - AoA 16 deg - R1 Mesh - *rhoPimpleFoam* Using the Time Scheme *Euler* for Transient Solutions

To validate the hypothesis of steady-state, a transient-state analysis is performed using mesh R1 and replacing the *localEuler* time-scheme with *Euler*. *Euler* is an implicit time-scheme that allows Co numbers above 1. Using *adjustableTimeStep*, the time-step is adjusted based on a Co_{max} of 1.5, which grants the solver enough stability and accuracy, resulting in a mean time-step of 8.9×10^{-8} s. Bigger Co_{max} in the range of 8 to 10 have shown partial stability when using the linear solver *PBiCGStab* to solve the state equations. The simulation is initialized using the converged results of the *localEuler* analysis. Starting from there, Figure 5.3 shows the time dependency of aerodynamic coefficients C_A, C_N, and $C_{m_{\text{pitch}}}$ over a simulated time period of 133.82×10^{-3} s.

Figure 5.3 concludes that steady-state can be achieved for conditions at Ma of 2.3 and AoA of 16º using the R1 mesh. Due to the limited computational budget available, the transient response analysis is only performed once on the coarsest mesh (R1). A more accurate analysis would require using the R5 mesh, a longer simulated time in the order of seconds, and a wider range of Ma and AoA should be validated. Nevertheless, the conclusion of steady-state flow drawn from Figure 5.3 should be valid for the entire flight envelope, as there are no indications of the contrary either regarding the results in Section 6.

5.5 Execution Strategy

To accelerate the convergence, simulations are first executed using a very coarse mesh, mesh R1 in Table 4.2. This provides a rough estimate that can be used to initialize the fields in the final finer mesh (R5). The OpenFOAM application *mapFieldsPar* is used to map the fields (*U, p, T, ...*) from one mesh to the other using multi-threading for accelerated computation, drastically reducing time that it takes compared to *mapFields* (single-thread).

The code presented in Figure 5.4 shows the instructions followed by OpenFOAM when executing each simulation. The code complements the information provided in Section 2.3 regarding semi-automatized workflows, as this *Allrun* script contains Jinja2 components.

```
#!/bin/sh
cd "${0%/*}" || exit                                # Run from this directory
. ${WM_PROJECT_DIR:?}/bin/tools/RunFunctions        # Tutorial run functions
#------------------------------------------------------------------------------

MESH_DIR={{ mesh_file }}
MAP_DIR={{ map_file }}

mkdir constant
cp -r $MESH_DIR constant
cp $MESH_DIR/../../system/meshDict system/meshDict
touch case.foam

restore0Dir

runApplication decomposePar

{% if map_file != "none" %} # Jinja2 Templating
{{ "runParallel mapFieldsPar $MAP_DIR -sourceTime latestTime -consistent -fields '(U p T k omega alphat nut)'" }}
{% endif %}

runParallel -append "$(getApplication)"

runApplication reconstructPar -latestTime
#------------------------------------------------------------------------------
```

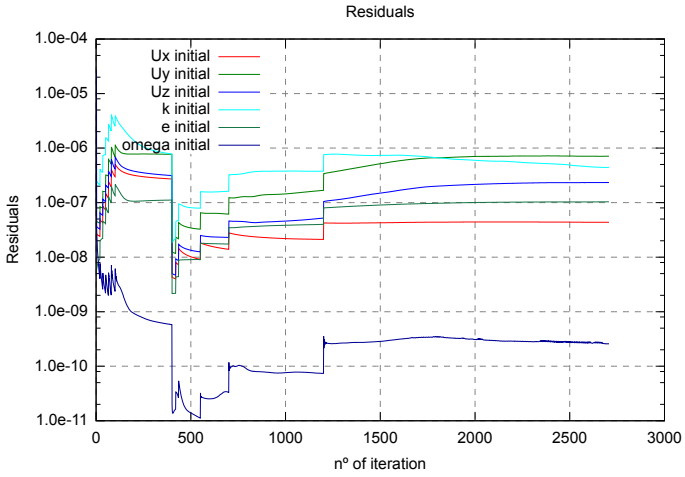
Figure 5.4 *Allrun* Script used To Execute Solutions for *rhoCentralFoam* and *rhoPimpleFoam* Solvers and Incorporating the Use of Templating Through Jinja2

To terminate simulations, the stabilization of aerodynamic coefficients C_A, C_N, and $C_{m_{\text{pitch}}}$ is monitored. This is implemented using an OpenFOAM FOs called *runTimeControl*, which stops simulations when convergence criterion are met. For this project, the only convergence criteria used is the stability of aerodynamic coefficients. If all aerodynamic coefficients fall within a moving average within a given tolerance, then the simulation is automatically stopped. Two parameters are provided: the tolerance and the size of the moving average window.

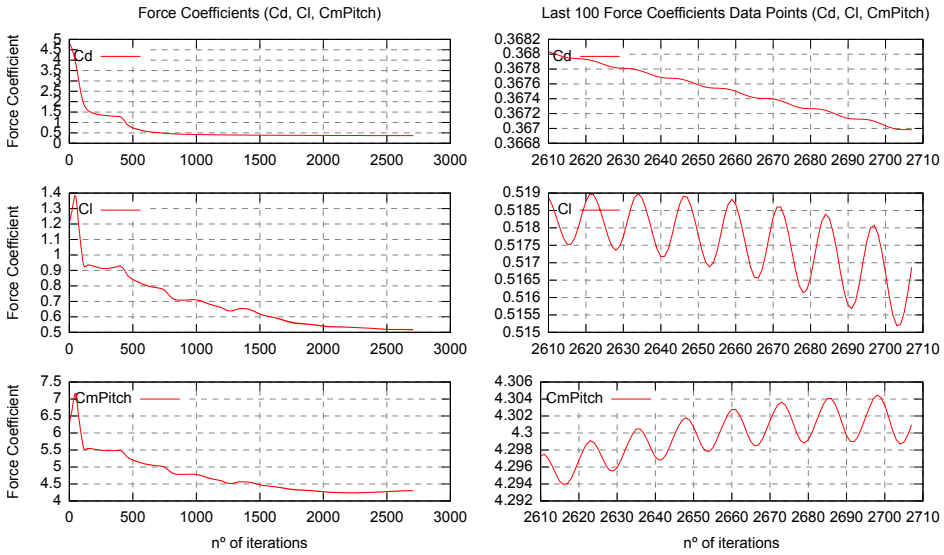
5.5.1 Specific for *rhoCentralFoam*

The solver *rhoCentralFoam* is notorious for being unstable and prone to crashing due to numerical issues. Multiple strategies have been tested to achieve stability. Initializing the flow using a basic solver like *potentialFlow*, which is inviscid and irrotational, has shown very limited success and, in some cases, has been detrimental, thus the idea was abandoned. On the other hand, incrementally updating the Co_{max} number in powers of 2 from a very low starting value (0.01, 0.02, 0.04, 0.08, ...) has proven very reliable. In addition to this strategy, the use of more dissipative interpolation schemes for velocity, i.e. *upwind*, has further resolved the convergence and numerical issues. *upwind* is a first-order bounded interpolation scheme, ideally the interpolation scheme preferred would be *vanLeer*, a second-order scheme that provides a more precise interpolation, leading to sharper shock waves and thus more exact results.

Using the FOs named *timeActivatedFileUpdate*, it is possible to update OpenFOAM configuration files (e.g. *controlDict*, *fvSolution*) based on the iteration number or the simulation time. The *timeActivatedFileUpdate* file contains a timeline for each file it has to modify, and the timelines have all the substitution files for each update. In this project, updates were done on the *controlDict* and *fvSchemes* files as the simulation progressed.



(a) Residuals Evolution - Steps Indicate Changing Parameters (Co_{max} and Interpolation Schemes) as the Simulation Progresses



(b) Aerodynamic Coefficients Evolution - Drop at 400 Iterations Indicates the Change From *upwind* to *vanLeer* Interpolation Scheme

Figure 5.5 Example of Real-Time Monitoring Plots used on Simulation Ma 2.3 - AoA 8 deg - R5 Mesh - *rhoCentralFoam* on a Converged Case

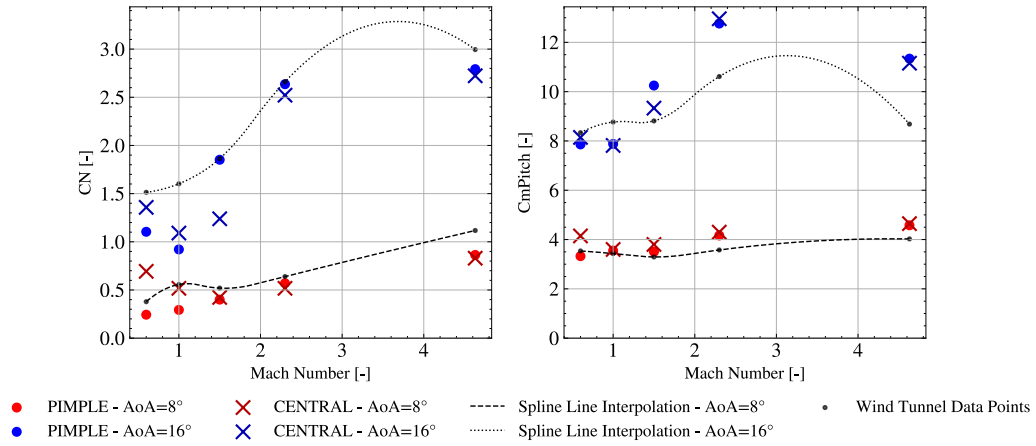
Figures 5.5 provide insight into how *rhoCentralFoam*'s convergence is achieved. Initially, the interpolation scheme is set to *upwind*, and the Co_{max} number is set to 0.01. As stated previously, the Co_{max} number increases in powers of 2 until it reaches 0.5. This process can be observed in Figure 5.5a up to iteration 100 with a set of incremental steps in the residuals. After the aerodynamic coefficients have plateaued in Figure 5.5b, the *upwind* scheme is upgraded to *vanLeer*, and the Co_{max} reverts to 0.01. The same process as before is reproduced but on a longer time scale. Once the new maximum Co_{max} at around 0.32 is achieved, then the minimum number of iterations will be accomplished, thus leading to the activation of the convergence criteria based on aerodynamic coefficient discussed in Section 5.5. The timing between Co_{max} transitions and interpolation schemes can be hard to find. Changes to the simulation's boundary conditions like speed, pressure, and temperature translate into different timing intervals. It is preferred to have longer simulation times which are stable across the full range of Ma and AoA, with larger Ma (i.e., Ma of 4.63) being harder to stabilize. Moreover, coarser meshes can also be more problematic as Ma increases.

Furthermore, Figure 5.5b shows an acute drop in the aerodynamic coefficients following the transition from *vanLeer* to *upwind*. Before the interpolation scheme change to *vanLeer*, coefficients converged to higher values for all C_A, C_N and $C_{m_{\text{pitch}}}$ quantities, and only upon changing the interpolation scheme the flow solution is modified and the coefficients change. As time progresses and the Co_{max} cycle starts again new convergence values are achieved. These results, visible in all the *rhoCentralFoam* simulations performed, are an indication of the high sensitivity of *rhoCentralFoam* to the interpolation scheme (*upwind* vs. *vanLeer*) and the significance of their impact on CFD solutions.

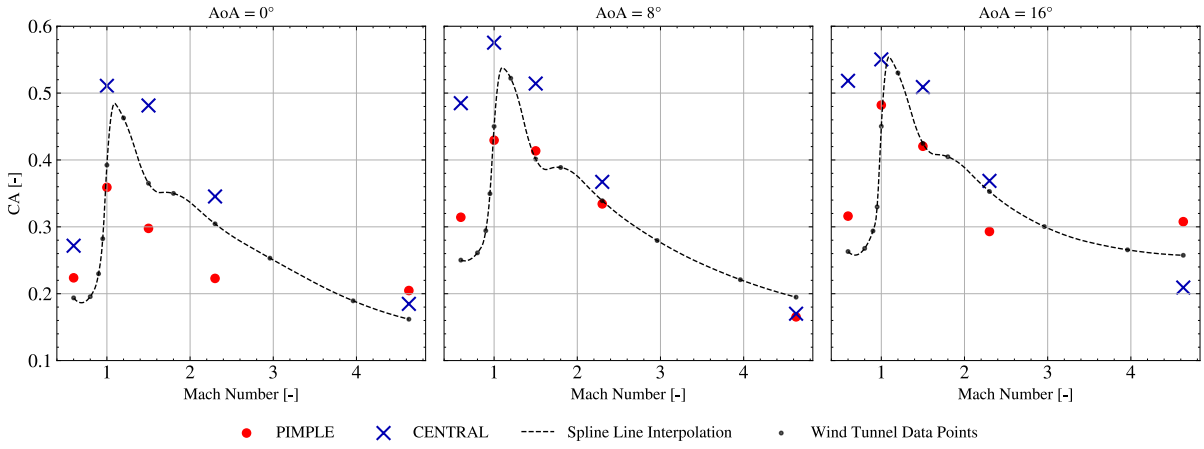
Note that, Figure 5.5a shows stable residuals before convergence with the Turbulent Kinetic Energy (k) continuing to drop. On the other hand, Figure 5.5b shows good converged results when looking at the totality of the iterations on the three plot on the left. However, when observing the last 100 iterations on the right, in particular for C_A named Cd in the plot, the coefficient continuous to drop further but at a very slow rate. The FOs *runTimeControl* has a big enough tolerance over a small enough window to deem the simulation as converged and stop the execution automatically. More strict parameters could be used to prolonged the number of iterations and access the convergence of this simulation.

6 Results

In this section, results obtained using the solvers *rhoPimpleFoam* and *rhoCentralFoam* are compared through a code-to-code analysis. These results are further compared to the available wind-tunnel data from [8, 9]. In the annex, Table A.1 and Table A.2 contains all the relevant information for each simulation. This information includes y^+ values, aerodynamic coefficients (C_A, C_N, $C_{m_{\text{pitch}}}$), the number of iterations, execution times, and more. All the simulations have been executed on a workstation powered by a 16-core CPU and are initialized using the coarse R1 mesh, resulting in a ${y^+}_{max}$ of approximately 60. Once converged, the results are mapped onto the finer mesh, resulting in a final ${y^+}_{max}$ of 0.8 to 1.5. It's important to note that the coarser mesh computes, on average, 100 times faster per iteration than the finer mesh. Even though the coarser mesh converges in a shorter time, it is not always possible to establish a global trend regarding the required number of iterations needed for convergence, as this varies significantly from case to case.



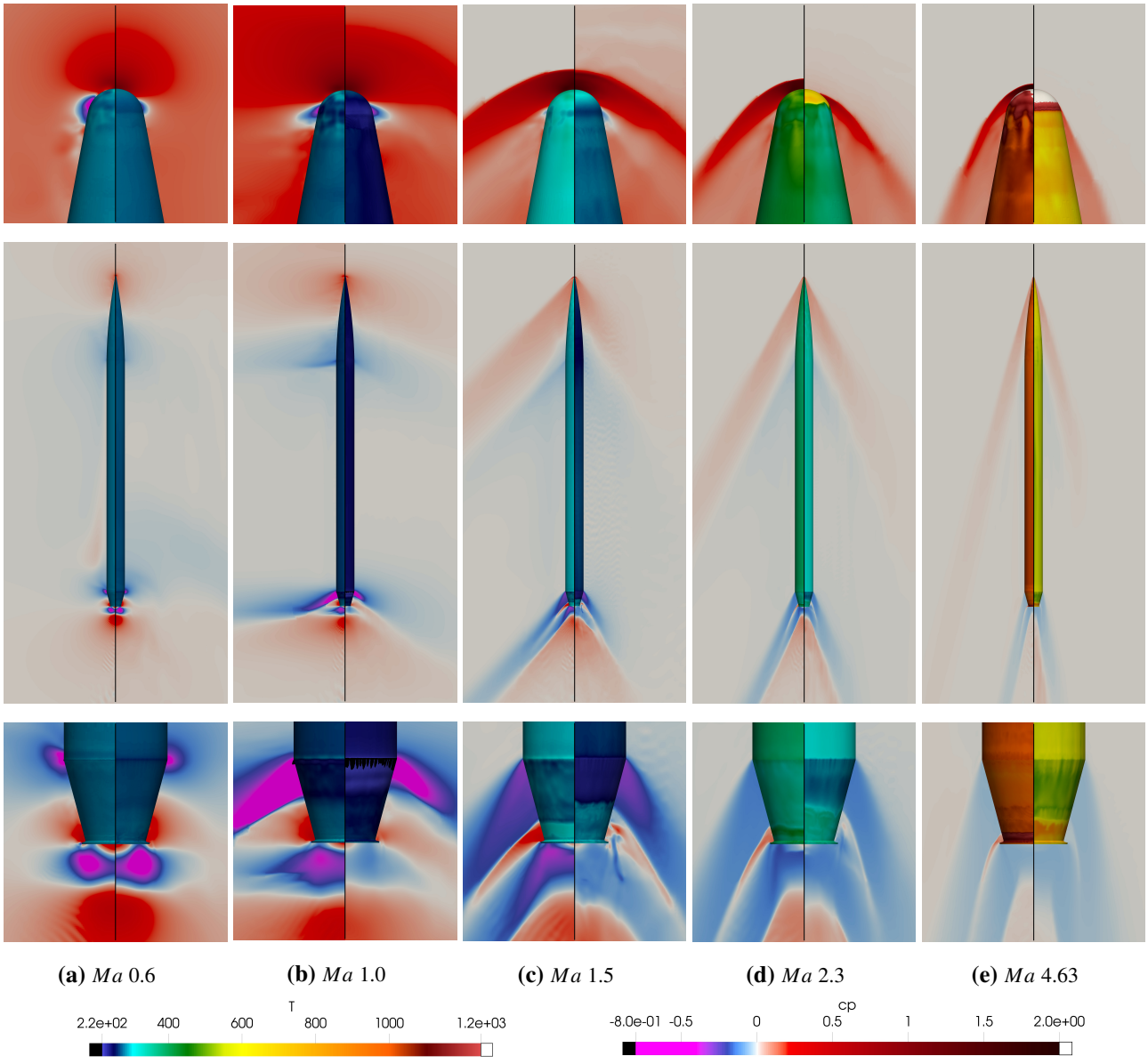
(a) Comparison of Aerodynamic Coefficients C_N and $C_{m_{\text{pitch}}}$ for Free-stream Conditions at AoA of 8 deg and 16 deg



(b) Comparison of the Drag Coefficient C_A for Free-stream Conditions at AoA of 0 deg, 8 deg, and 16 deg

Figure 6.1 Comparison of Results Obtained From *rhoPimpleFoam* and *rhoCentralFoam* across all Ma Numbers with wind-tunnel Data from [8, 9]

Initially, a comparative study of the aerodynamic coefficients from *rhoPimpleFoam*, *rhoCentralFoam*, and the wind-tunnel data is performed in Figures 6.1b and 6.1a. Conclusions do not show a tendency toward more accurate results from any of the two solvers at any free-stream condition (Ma and AoA). The relative error with respect to wind-tunnel data remains high in both cases. However, it must be pointed-out that for solver *rhoCentralFoam* the relative error with respect to wind-tunnel data for C_A at low Ma numbers is much higher at around 100% than for the pressure-based alternative, *rhoPimpleFoam*. For subsonic regimes ($Ma < 0.8$), *rhoPimpleFoam* appears to provide better aerodynamic results, particularly at AoA of 8 deg and 16 deg. When analyzing the aerodynamic coefficients with respect to wind-tunnel data, the relative errors for C_A in both solvers tends to be higher due to the spike in drag close to the transonic region. This spike in drag, seen in Figure 6.1b, should be studied further to assess the validity of the solvers around this region. Moreover, the unsteady nature of flow at the aft of the rocket could be the cause of higher relative errors observed in Figure 6.1b for C_A compared to errors for $C_{m_{\text{pitch}}}$ and C_N in Figure 6.1a, the latter showing good results for both coefficients, with no high uncertainty around the transonic region.



(a) Ma 0.6 **(b)** Ma 1.0 **(c)** Ma 1.5 **(d)** Ma 2.3 **(e)** Ma 4.63

Figure 6.2 Comparison of Solutions From *rhoCentralFoam* (left) and *rhoPimpleFoam* (right) Across All Studied Ma Numbers at an AoA of 0 deg Using c_p and Skin Temperature

The extraction of aerodynamic coefficients may not be the sole requirement for a solver to be deemed good at a specific aerodynamic regime. For instance, other factors like wall temperatures are remarkably important to make informed design decisions. As a reminder, the temperature boundary condition set at the wall of the rocket is adiabatic. Also, shock resolution is of the utmost importance to validate the solvers.
Figure 6.2 shows a code-to-code comparison of solutions provided by *rhoCentralFoam* on the right and *rhoPimpleFoam* on the left part of each column. Each column corresponds to one of the five Ma numbers selected in Chapter 3 at a fixed AoA of 0 deg. The Pressure Coefficient (c_p) normalizes the atmospheric pressure around the vehicle, which varies for every Ma as established in Table 3.1, with blue and purple zones showing a decrease in c_p while red zones mark an increase. Additionally, the temperature at the surface of the rocket is also indicated using a modified rainbow color scheme.

As the rocket from subsonic to supersonic, the two solutions start to diverge from each other. As can be seen in Figure 6.2c, the aft region shows two different solutions already at Ma 1.5. The solver *rhoCentralFoam* captures stronger shock waves and expansion fans compared to *rhoPimpleFoam*, the latter of which starts a shock wave at the aft of the vehicle sooner than its counterpart, upstream of the lip, and the expansion wave after is practically nonexistent. Moreover, shocks in *rhoPimpleFoam* dissipate sooner as they travel away from the rocket compared to *rhoCentralFoam*. At Ma 1.5, temperature differences remain relatively small at the surface of the rocket, around 20°C to 40°C, with the largest discrepancies at the nose tip of the rocket.
As the Ma number increases further, *rhoPimpleFoam* starts to behave in nonphysical ways. Results at Ma 2.3 and 4.63 show a shock wave at the tip of the rocket, which vanishes in *rhoPimpleFoam* compared to *rhoCentralFoam*, leaving behind a very high-temperature spot on the surface of the rocket. Effects at the aft of the rocket, which were already observed at Ma 1.5 persist. For *rhoPimpleFoam*, the generation in the aft of the rocket of a weaker shock wave at a higher position compared to *rhoCentralFoam* leaves the lip in the path of incoming high-velocity flow, leading to a substantial increase in surface temperature. Finally, surface temperature discrepancies between both solvers become extreme at Ma of 4.63, with the nose tip temperature in *rhoPimpleFoam* segregated into two regions, aligning with the collapse of the frontal shock wave.

A more in-depth analysis is performed regarding the differences between both solvers at the aft of the vehicle. Figure 6.3 depicts a comparison between both solvers, with *rhoCentralFoam* in the upper part and *rhoPimpleFoam* in the lower part, for a freestream condition with Ma of 2.3 and an AoA of 0 deg. For the analysis, two important parameters are considered: the wall shear stress to detect potential boundary separation regions and streamlines to track the flow evolution downstream. It's worth noting that streamlines initiate at the same positions in both cases. Based on additional information gathered in Figure 6.2d, it can be inferred that shock waves initiate at the drop in wall shear stress for both solutions. This information, in combination with Figure 6.3, leads to a hypothesis of shock-induced separation, which is evident from a sharp decrease in wall shear stress combined with an intense turbulent region afterward, as visualized through the streamlines for *rhoPimpleFoam*. This phenomenon is clearly observed in *rhoPimpleFoam*, but not in *rhoCentralFoam*. As a result, both solvers provide different results that should raise questions about the validity of at least one of them.

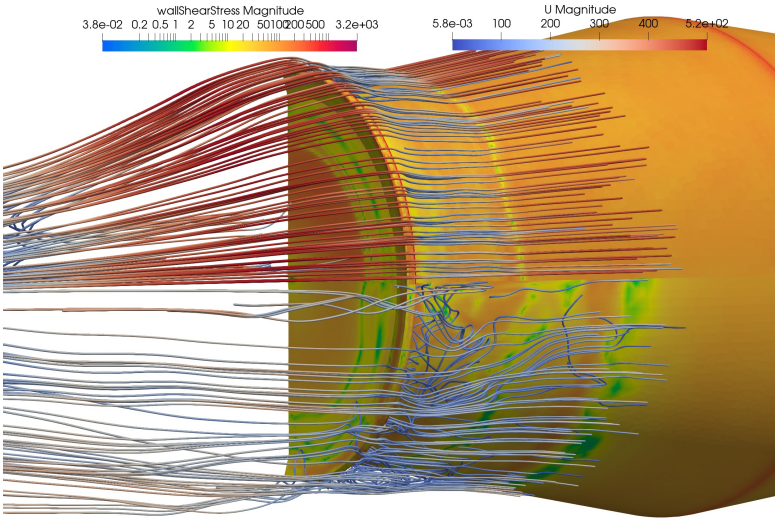


Figure 6.3 Comparison of *rhoCentralFoam* (upper) and *rhoPimpleFoam* (lower) Flow Separation at the Aft of the Rocket Using Wall Shear Stress and Streamlines

7 Further Developments and Conclusions

All project objectives have been successfully accomplished, first by providing an aerodynamic database for a given rocket geometry across a multitude of aerodynamic regimes, being those subsonic, transonic, and supersonic. Then, by establishing a repetitive and modular toolchain comprising of pre-processing, execution, and post-processing steps aimed to be used in an industrial environment. This toolchain was built with semi-automatization in mind, leading to great productivity gains like generation of OpenFOAM workspace using templates via Jinja2 and the creation of a execution pipeline that automatically feed simulation to the server without human intervention. The combination of these automatized processes in OpenFOAM enabled the user to ensure uniformity and repeatability across the multiple simulations. Finally, the toolchain was tested on a geometry and on two solvers thus accomplishing the third and final objective of the project which was to find a well suited solver for the aerodynamic characterization of a rocket.

Gained experience from this project shows that *rhoPimpleFoam* is numerically more stable than *rhoCentralFrom*. For instance, the LTS formulation is more permissive in *rhoPimpleFoam* with Co_{max} set at higher values of up to 0.8 without causing numerical instabilities. Moreover, *rhoPimpleFoam* is documented by OpenFOAM ESI [19] to be capable of resolving compressible problems, with our results showing good accuracy up to Ma numbers of 1.5, with further increases leading to nonphysical results notably at the tip of the vehicle. Even after further refining the mesh at the tip of vehicle the solver remains unable to solve the flow in that region. Overall, *rhoPimpleFoam* works out of the box compared to all the difficulties faced with *rhoCentralFoam* to avoid numerical instabilities during execution.
Nevertheless, conclusions regarding the preferred solver point to *rhoCentralFoam* as being the one better suited for supersonic flows. However, with the aim of transitioning from unpowered flight towards powered flight the use of multi-species solvers should be considered. In this context, *rhoCentralFoam* solver does not come in OpenFOAM ESI Group (ESI) with a multi-species equivalent, on the other hand, *rhoPimpleFoam* has it under the name *rhoReactingFoam*. Regardless of the observed lower accuracy of *rhoPimpleFoam* the potential to transition to multi-species within the officially recognized OpenFOAM solvers makes it worthy of further endeavours. Multiple parameters like the interpolation schemes and the linear equation solvers selected could be modified to better resolved the flow and potential. It remains doubtful that *rhoPimpleFoam* could match the results of *rhoCentralFoam* at high Ma where the density-based approach of the latter makes it more suited. It must be reminded that the aerodynamic database stretches over multiple aerodynamic regimes.

The addition of engines will further exacerbate the computational resources available by increasing the cell count. AMR techniques could make use of the results provided in this work to refine the mesh only where needed, thus reducing the size of the initial mesh of 6.5 million elements currently used. Further more, the use of a structured meshes to add the engines could reduce the number of cells added to the system, to accommodate such mesh to the already existing one the use of *ciclicAMI* could be beneficial. Further assessments on the sensibility of the solution towards CFD parameters like: element size in the mesh, select turbulence model, or cell interpolation schemes should be the focus of further study by expanding on the work provided here, limited by the computational resources available. Finally, to improve on the developed toolchain for this work the addition of Continuous Integration and Continuous Deployment (CI/CD) systems through the GitHub/GitLab software could help ensure the scalability of the toolchain and cement its use in industrial environments.

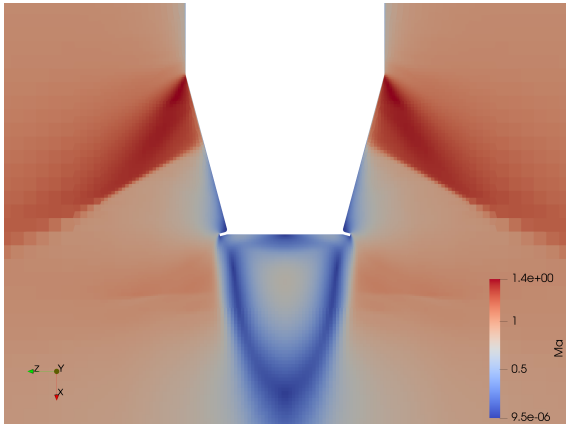
A Annex

Table A.1 Automatically Generated Report on All Simulations Performed for the Aerodynamic Characterization of the Rocket using *rhoCentralFoam* and *rhoPimpleFoam* - Page 1

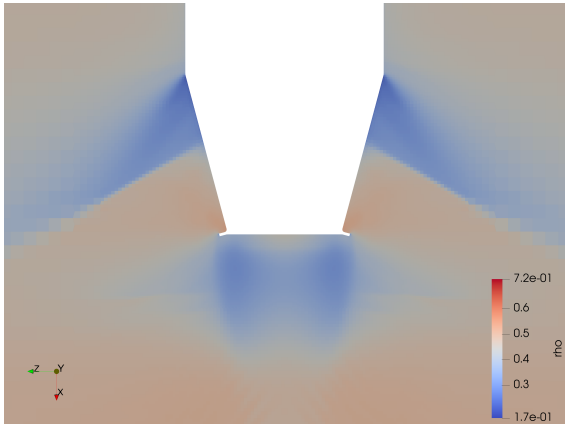
SIMULATION IDEN				YPLUS			FINAL COEFFICIENTS			FINAL COEFFCIENTS AVERAGE 50 LAST ITERATIONS			TIMING			RESIDUALS						
Ma	AoA	Mesh	Solver	min.	max.	avg.	CA	CN	CmPitch	CA.	CN.	CmPitch.	Execution Time [s]	Number of Iterrations	Iterrations per Second [1/s]	Ux	omega	p	Uy	k	e	Uz
0.6	0	R5	*rhoCentralFoam*	0.006	1.349	0.274	0.272	-0.001	-0.001	0.272	-0.001	0.001	66052	3661	0.055	9.06E-09	9.02E-11	-	1.76E-08	2.26E-06	9.95E-09	1.14E-06
0.6	0	R1	*rhoCentralFoam*	0.221	74.663	33.025	0.373	0.000	-0.002	0.373	0.001	-0.001	38350	5712	0.149	1.06E-09	9.61E-08	-	4.46E-10	1.64E-08	4.46E-10	1.56E-07
0.6	8	R5	*rhoCentralFoam*	0.004	1.517	0.317	0.484	0.696	4.138	0.485	0.693	4.147	152311	9221	0.061	4.60E-08	7.24E-10	-	3.04E-07	3.70E-07	1.23E-07	1.72E-07
0.6	8	R1	*rhoCentralFoam*	0.541	77.764	33.596	0.433	0.579	4.381	0.433	0.578	4.376	72481	12651	0.175	1.15E-09	2.45E-08	-	1.29E-09	5.41E-09	4.74E-10	2.66E-09
0.6	16	R5	*rhoCentralFoam*	0.004	1.554	0.324	0.518	1.324	8.070	0.518	1.357	8.144	163808	9191	0.056	5.52E-08	7.51E-10	-	3.04E-07	3.58E-07	1.56E-07	1.28E-07
0.6	16	R1	*rhoCentralFoam*	0.468	81.935	34.427	0.434	1.333	9.275	0.434	1.333	9.273	4283	15835	3.697	1.30E-09	1.29E-08	-	1.76E-09	4.34E-09	4.75E-10	1.41E-09
1	0	R5	*rhoCentralFoam*	0.003	1.218	0.257	0.511	0.000	0.004	0.511	0.000	0.001	37731	2685	0.071	2.52E-08	1.10E-10	-	1.29E-07	2.42E-07	4.80E-08	6.54E-06
1	0	R1	*rhoCentralFoam*	0.285	63.677	29.261	0.483	0.000	0.002	0.483	0.000	0.001	44990	8389	0.186	1.86E-09	2.01E-08	-	1.19E-09	4.08E-09	9.71E-10	3.55E-07
1	8	R5	*rhoCentralFoam*	0.003	1.215	0.243	0.575	0.516	3.599	0.575	0.517	3.598	61166	5430	0.089	4.62E-08	9.98E-11	-	3.00E-07	1.79E-07	9.87E-08	1.90E-07
1	8	R1	*rhoCentralFoam*	0.261	65.718	30.209	0.545	0.548	4.153	0.545	0.548	4.152	1637	6725	4.109	1.77E-09	2.56E-08	-	2.60E-09	8.11E-09	9.83E-10	5.13E-09
1	16	R5	*rhoCentralFoam*	0.003	1.274	0.242	0.550	1.090	7.821	0.550	1.091	7.817	87744	6676	0.076	5.64E-08	1.02E-10	-	3.09E-07	2.58E-07	1.35E-07	1.45E-07
1	16	R1	*rhoCentralFoam*	0.298	68.239	30.729	0.574	1.315	9.073	0.574	1.315	9.070	1566	5920	3.781	1.88E-09	2.64E-08	-	3.58E-09	9.96E-09	9.84E-10	3.29E-09
1.5	0	R5	*rhoCentralFoam*	0.002	0.895	0.189	0.481	0.000	0.001	0.481	0.000	0.001	32796	2602	0.079	2.67E-08	1.99E-10	-	3.14E-07	2.99E-07	5.69E-08	7.68E-06
1.5	0	R1	*rhoCentralFoam*	0.151	58.351	25.244	0.469	0.000	0.001	0.468	0.000	0.000	988	4768	4.824	3.20E-09	1.25E-08	-	2.42E-09	5.19E-09	1.89E-09	4.96E-07
1.5	8	R5	*rhoCentralFoam*	0.002	0.956	0.190	0.514	0.429	3.812	0.514	0.423	3.800	31313	2408	0.077	4.61E-08	2.25E-10	-	4.68E-07	4.73E-07	1.05E-07	2.10E-07
1.5	8	R1	*rhoCentralFoam*	0.217	62.742	26.082	0.495	0.553	4.319	0.495	0.553	4.317	993	4777	4.809	2.76E-09	1.12E-08	-	5.59E-09	8.01E-09	1.80E-09	4.96E-09
1.5	16	R5	*rhoCentralFoam*	0.003	1.017	0.181	0.509	1.244	9.337	0.509	1.239	9.332	32873	2490	0.076	6.13E-08	2.26E-10	-	4.34E-07	8.27E-07	1.42E-07	1.82E-07
1.5	16	R1	*rhoCentralFoam*	0.284	63.821	25.847	0.513	1.491	10.070	0.513	1.491	10.070	1186	5656	4.770	2.79E-09	6.94E-09	-	6.41E-09	1.13E-08	1.97E-09	4.50E-09
2.3	0	R5	*rhoCentralFoam*	0.002	0.527	0.114	0.345	0.000	0.000	0.345	0.000	0.000	39051	2923	0.075	2.37E-08	2.42E-10	-	8.18E-07	2.78E-07	5.52E-08	7.92E-06
2.3	0	R1	*rhoCentralFoam*	0.060	51.207	19.102	0.334	0.000	0.000	0.334	0.000	0.000	815	4011	4.921	4.22E-09	1.85E-09	-	1.01E-08	4.35E-09	2.60E-09	8.01E-07
2.3	8	R5	*rhoCentralFoam*	0.002	0.632	0.114	0.367	0.517	4.301	0.367	0.517	4.301	36616	2707	0.074	4.35E-08	2.57E-10	-	7.11E-07	4.42E-07	1.03E-07	2.34E-07
2.3	8	R1	*rhoCentralFoam*	0.247	61.360	19.807	0.355	0.614	4.530	0.355	0.613	4.529	987	4873	4.937	3.51E-09	1.32E-09	-	1.11E-08	6.35E-09	2.27E-09	7.60E-09
2.3	16	R5	*rhoCentralFoam*	0.002	0.778	0.106	0.369	2.522	12.954	0.369	2.522	12.958	34276	2525	0.074	6.28E-08	2.84E-10	-	5.58E-07	1.18E-06	1.30E-07	1.91E-07
2.3	16	R1	*rhoCentralFoam*	0.062	68.066	18.968	0.372	2.708	13.270	0.372	2.709	13.274	1030	4975	4.829	2.61E-09	9.19E-10	-	6.33E-09	1.03E-08	1.89E-09	3.52E-09
4.63	0	R5	*rhoCentralFoam*	0.001	0.142	0.031	0.185	0.000	0.000	0.185	0.000	0.000	31563	2104	0.067	1.22E-08	1.73E-09	-	5.61E-07	6.70E-07	3.14E-08	3.13E-06
4.63	0	R1	*rhoCentralFoam*	0.016	46.173	12.477	0.320	0.000	0.000	0.338	0.000	0.000	412	2062	5.010	4.35E-10	1.31E-11	-	2.65E-09	6.10E-09	5.46E-11	2.71E-07
4.63	8	R5	*rhoCentralFoam*	0.000	0.187	0.028	0.170	0.829	4.644	0.170	0.830	4.646	45633	2666	0.058	1.80E-08	1.89E-09	-	4.02E-07	6.60E-06	4.09E-08	1.07E-07
4.63	8	R1	*rhoCentralFoam*	0.018	62.912	12.351	0.324	1.106	5.330	0.339	1.121	5.364	411	2055	5.005	4.21E-10	1.07E-11	-	1.09E-09	4.75E-09	5.46E-11	3.39E-10
4.63	16	R5	*rhoCentralFoam*	0.000	0.360	0.034	0.210	2.723	11.157	0.209	2.723	11.155	60340	4178	0.069	3.43E-08	6.78E-10	-	6.01E-07	2.33E-06	7.82E-08	1.27E-07
4.63	16	R1	*rhoCentralFoam*	0.025	74.566	14.031	0.386	3.137	12.163	0.392	3.147	12.189	997	5124	5.137	1.05E-10	2.30E-12	-	4.07E-10	1.13E-09	1.44E-11	5.75E-11
0.6	0	R5	*rhoPimpleFoam*	0.001	1.180	0.240	0.223	-0.001	0.005	0.224	-0.001	0.006	46704	2051	0.044	1.22E-08	4.78E-12	5.56E-07	6.59E-07	9.95E-09	1.64E-07	8.79E-05
0.6	0	R1	*rhoPimpleFoam*	0.595	68.122	30.822	0.285	0.000	0.002	0.285	0.000	0.001	1682	4301	2.557	2.45E-08	1.55E-07	1.88E-06	3.84E-07	1.23E-08	2.87E-07	3.92E-04
0.6	8	R5	*rhoPimpleFoam*	0.002	1.311	0.246	0.315	0.244	3.325	0.314	0.243	3.327	24976	1081	0.043	1.04E-08	2.93E-12	4.07E-07	3.61E-07	1.31E-08	9.73E-08	1.01E-07
0.6	8	R1	*rhoPimpleFoam*	0.172	70.030	30.610	0.348	0.283	3.561	0.348	0.283	3.561	4619	11091	2.401	1.78E-08	6.46E-08	9.62E-06	2.56E-07	8.04E-09	1.12E-07	1.94E-07
0.6	16	R5	*rhoPimpleFoam*	0.001	1.489	0.262	0.316	1.104	7.860	0.316	1.104	7.860	103611	4237	0.041	9.66E-09	1.38E-12	9.17E-07	1.69E-07	1.90E-08	8.37E-08	4.16E-08
0.6	16	R1	*rhoPimpleFoam*	0.294	72.939	31.861	0.358	0.741	7.715	0.358	0.741	7.716	8532	20081	2.354	1.26E-08	3.05E-08	6.78E-07	9.32E-08	6.78E-09	7.54E-08	4.54E-08
1	0	R5	*rhoPimpleFoam*	0.001	1.107	0.215	0.359	0.000	-0.002	0.359	0.000	-0.002	24498	1051	0.043	6.87E-09	1.12E-12	2.59E-07	7.01E-07	2.95E-09	5.07E-08	1.97E-05
1	0	R1	*rhoPimpleFoam*	0.216	58.114	26.901	0.339	0.000	-0.001	0.339	0.000	-0.001	4212	10803	2.565	1.34E-08	2.68E-08	1.07E-06	5.96E-07	2.34E-09	6.41E-08	2.65E-05
1	8	R5	*rhoPimpleFoam*	0.001	1.239	0.225	0.429	0.292	3.549	0.430	0.293	3.553	24480	1051	0.043	6.92E-09	1.19E-12	2.25E-07	3.25E-07	5.27E-09	4.21E-08	6.35E-08
1	8	R1	*rhoPimpleFoam*	0.284	62.419	27.914	0.403	0.312	3.694	0.403	0.312	3.694	4648	11769	2.532	1.16E-08	1.96E-08	1.60E-06	2.38E-07	3.45E-09	4.51E-08	1.15E-07
1	16	R5	*rhoPimpleFoam*	0.001	1.321	0.241	0.482	0.921	7.873	0.482	0.921	7.878	27090	1186	0.044	8.76E-09	1.27E-12	7.63E-07	2.13E-07	8.47E-09	6.56E-08	5.60E-08
1	16	R1	*rhoPimpleFoam*	0.520	63.021	29.155	0.469	0.942	8.235	0.469	0.941	8.235	3425	8508	2.484	1.64E-08	2.00E-08	8.62E-07	1.63E-07	4.87E-09	8.47E-08	1.06E-07
1.5	0	R5	*rhoPimpleFoam*	0.001	0.929	0.161	0.298	0.000	0.004	0.298	0.000	0.004	23924	1051	0.044	3.15E-08	3.29E-12	1.43E-05	2.42E-06	6.26E-09	1.65E-06	9.55E-05
1.5	0	R1	*rhoPimpleFoam*	0.127	53.781	23.658	0.357	-0.001	0.011	0.357	-0.002	0.013	13092	16595	1.268	8.58E-09	1.67E-11	1.84E-05	4.53E-07	6.04E-10	4.59E-07	2.54E-05
1.5	8	R5	*rhoPimpleFoam*	0.002	1.129	0.194	0.413	0.400	3.509	0.414	0.400	3.503	37033	1706	0.046	2.59E-08	3.22E-12	3.32E-05	6.76E-07	2.89E-09	1.28E-06	1.47E-07
1.5	8	R1	*rhoPimpleFoam*	0.203	58.569	24.024	0.432	0.435	3.830	0.432	0.435	3.830	1085	2830	2.608	9.60E-08	4.15E-10	5.88E-06	1.06E-06	2.80E-08	5.97E-07	6.63E-07
1.5	16	R5	*rhoPimpleFoam*	0.001	1.233	0.204	0.420	1.852	10.258	0.420	1.852	10.248	51940	2501	0.048	2.96E-08	1.07E-11	4.32E-04	3.87E-07	3.42E-09	1.77E-06	8.36E-08
1.5	16	R1	*rhoPimpleFoam*	0.166	62.023	24.321	0.440	1.961	10.244	0.440	1.961	10.245	1148	3305	2.879	1.45E-07	6.39E-10	5.81E-06	8.80E-07	5.38E-08	1.01E-06	3.63E-07
2.3	0	R5	*rhoPimpleFoam*	0.001	0.441	0.098	0.223	0.002	-0.004	0.223	0.002	-0.004	47171	1755	0.037	7.17E-10	4.86E-13	4.35E-07	5.59E-07	4.21E-10	2.24E-08	1.33E-05
2.3	0	R1	*rhoPimpleFoam*	0.111	47.128	18.186	0.300	0.001	-0.007	0.300	0.001	-0.011	5063	13713	2.709	5.67E-09	1.25E-11	2.68E-06	5.31E-07	7.75E-10	1.65E-07	3.96E-05
2.3	8	R5	*rhoPimpleFoam*	0.001	0.973	0.168	0.334	0.570	4.164	0.334	0.570	4.164	97674	5655	0.058	4.75E-07	1.17E-08	2.67E-05	1.80E-05	3.47E-08	1.28E-06	2.99E-06
2.3	8	R1	*rhoPimpleFoam*	0.087	56.958	17.524	0.337	0.577	4.225	0.337	0.578	4.225	1202	2631	2.189	7.19E-08	1.89E-10	3.06E-06	1.17E-06	1.44E-08	3.76E-07	4.63E-07
2.3	16	R5	*rhoPimpleFoam*	0.001	0.688	0.106	0.293	2.635	12.762	0.293	2.636	12.763	38399	1351	0.035	1.53E-09	5.51E-13	2.98E-06	5.39E-08	7.26E-10	2.43E-08	8.49E-09
2.3	16	R1	*rhoPimpleFoam*	0.086	66.353	17.503	0.351	2.602	12.602	0.351	2.602	12.602	742	2272	3.062	1.74E-07	3.70E-10	5.85E-06	1.32E-06	4.29E-08	9.42E-07	4.15E-07

Table A.2 Automatically Generated Report on All Simulations Performed for the Aerodynamic Characterization of the Rocket using *rhoCentralFoam* and *rhoPimpleFoam* - Page 2

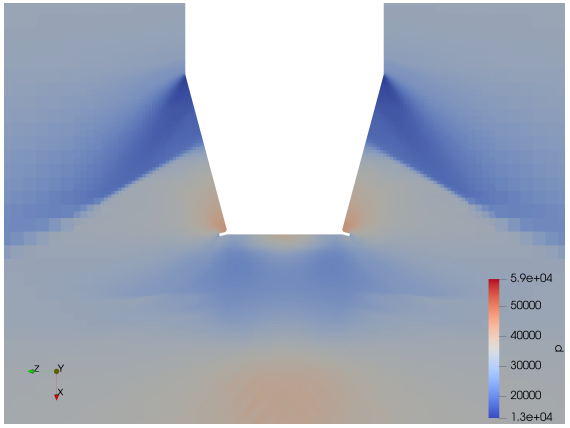
4.63	0	R5	*rhoPimpleFoam*	0.000	0.322	0.062	0.204	0.000	0.002	0.205	0.000	0.002	47938	2730	0.057	1.28E-10	4.38E-10	1.26E-07	3.22E-08	2.28E-10	1.68E-09	2.58E-06
4.63	0	R1	*rhoPimpleFoam*	0.007	44.326	7.696	0.181	0.000	0.000	0.181	0.000	0.000	3035	7608	2.506	1.62E-09	8.20E-12	2.34E-07	2.41E-07	6.87E-10	8.82E-09	7.25E-06
4.63	8	R5	*rhoPimpleFoam*	0.000	0.300	0.041	0.165	0.863	4.577	0.165	0.863	4.579	53003	1916	0.036	1.93E-10	8.85E-13	1.35E-06	2.55E-08	1.14E-10	3.93E-09	2.78E-09
4.63	8	R1	*rhoPimpleFoam*	0.032	47.763	6.975	0.195	0.864	4.462	0.195	0.864	4.461	7827	19630	2.508	1.87E-09	1.54E-12	9.37E-07	6.42E-08	6.15E-10	3.43E-08	2.09E-08
4.63	16	R5	*rhoPimpleFoam*	0.000	0.739	0.055	0.307	2.792	11.340	0.308	2.792	11.339	42738	1505	0.035	2.85E-10	5.79E-13	1.62E-07	3.47E-08	6.65E-09	4.01E-09	2.68E-09
4.63	16	R1	*rhoPimpleFoam*	0.025	128.496	9.155	0.264	2.772	11.168	0.264	2.772	11.167	10253	25645	2.501	1.07E-09	8.93E-13	4.06E-06	4.30E-08	3.68E-10	1.52E-08	7.82E-09



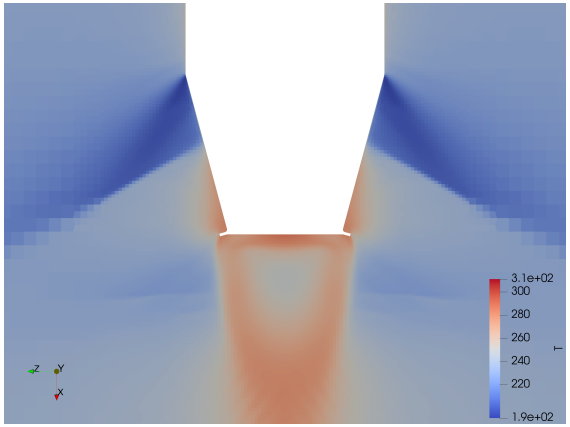
(a) Mach Number (Ma)



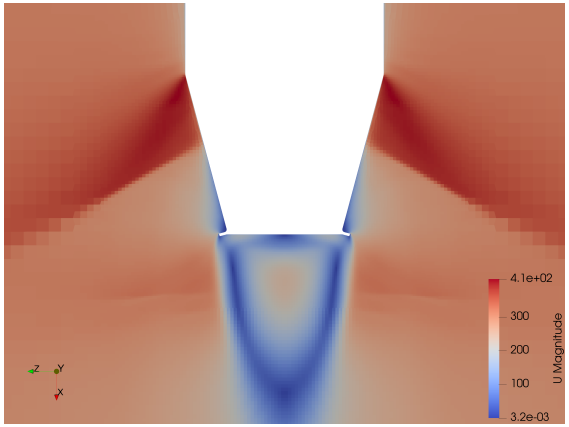
(b) Density (ρ)



(c) Pressure (p)

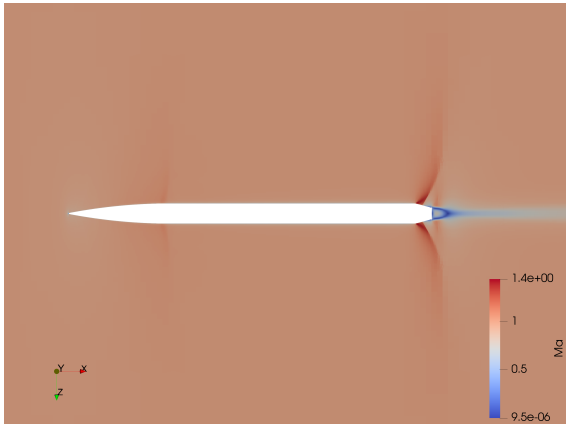


(d) Temperature (T)

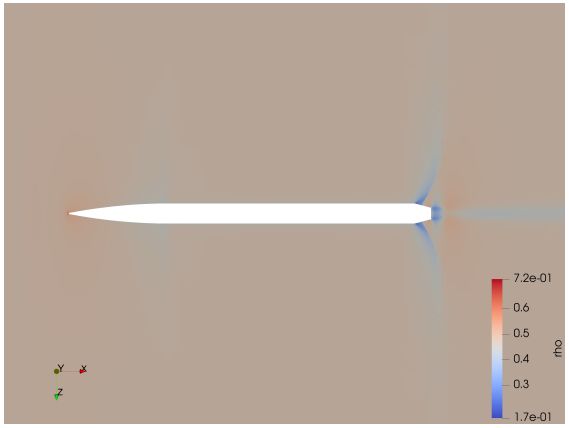


(e) Velocity (U)

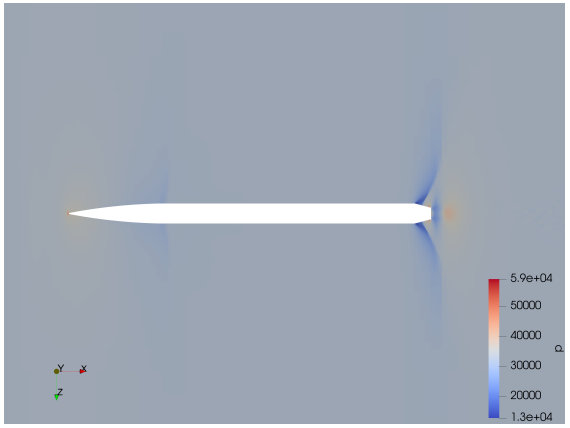
Figure A.1 Various Images for Ma, Density, Pressure, Temperature, and Velocity Generated Automatically After Convergence of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam* Using the ParaView Python API - Zoomed Region at the Aft of the Rocket



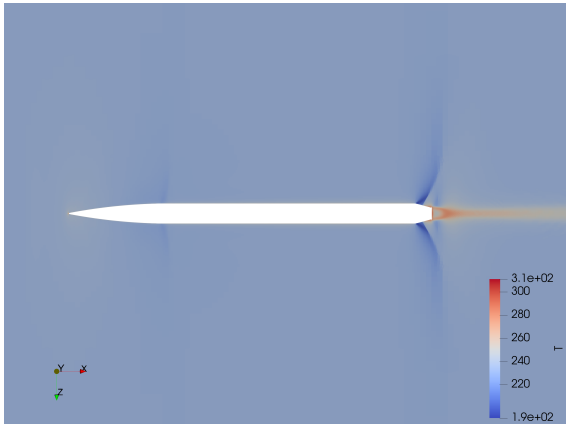
(a) Mach Number (Ma)



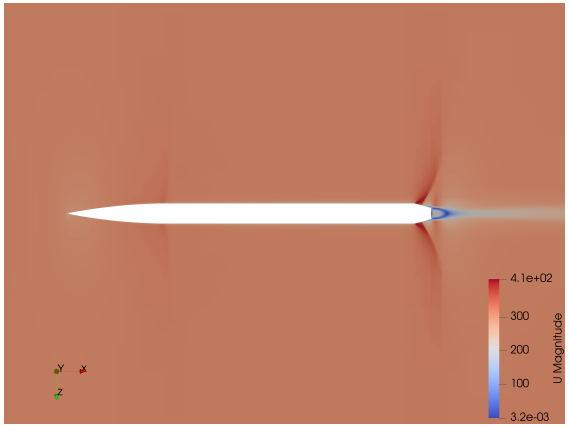
(b) Density (ρ)



(c) Pressure (p)

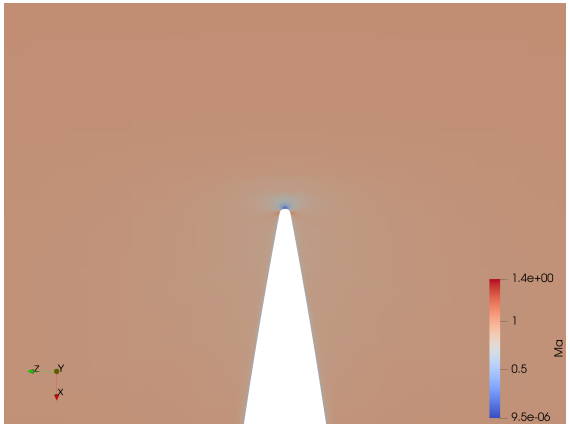


(d) Temperature (T)

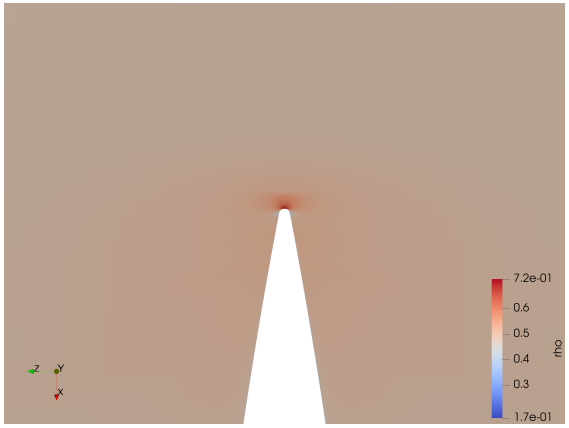


(e) Velocity (U)

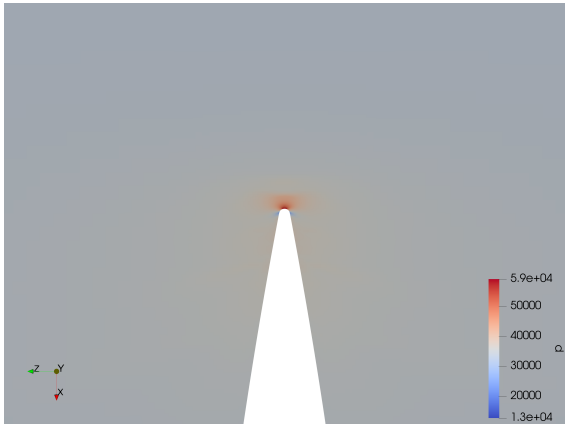
Figure A.2 Various Images for Ma, Density, Pressure, Temperature, and Velocity Generated Automatically After Convergence of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam* Using the ParaView Python API - General View of Flow Around the Rocket



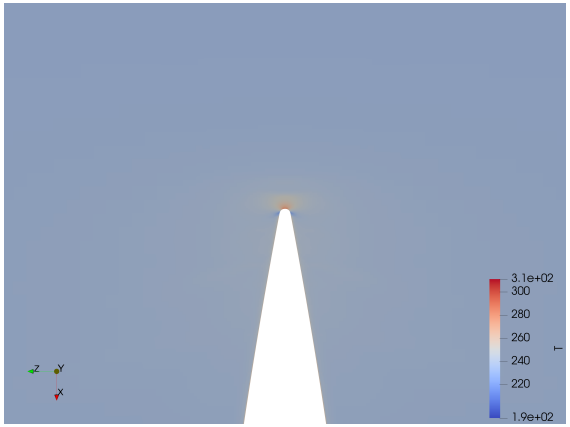
(a) Mach Number (Ma)



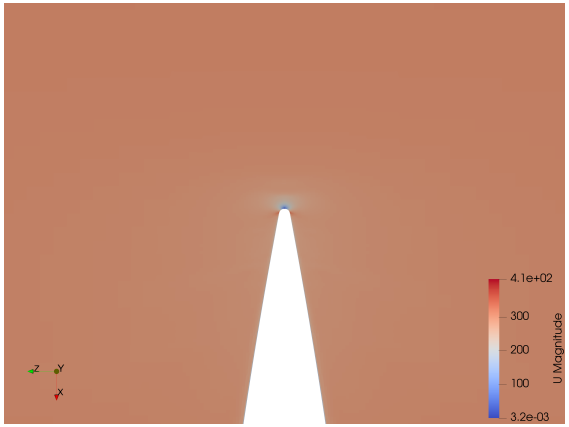
(b) Density (ρ)



(c) Pressure (p)

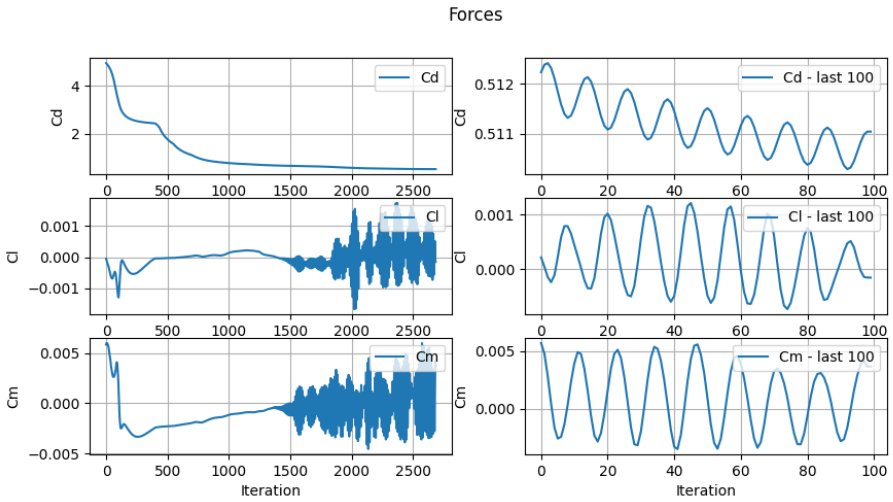


(d) Temperature (T)

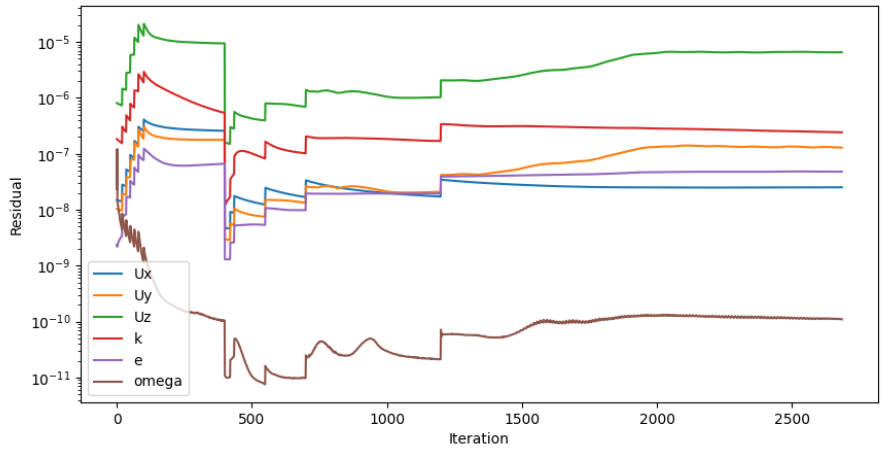


(e) Velocity (U)

Figure A.3 Various Images for Ma, Density, Pressure, Temperature, and Velocity Generated Automatically After Convergence of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam* Using the ParaView Python API - Zoomed Region at the Tip of the Rocket



(a) Convergence Plot for Aerodynamic Coefficients (C_A, C_N, and $C_{m_{\text{pitch}}}$) - Complete Evolution of the Coefficients at the Left, and Last 100 Data Points to the Right



(b) Convergence Plot for Residuals

Figure A.4 Automatically Generated Convergence Plots After Termination of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam*

```
Report
case: D:\good\rhoCentralFoam\R5\test\Ma1.0
     _AoA0_R5_rhoCentralFoam
------

yPlus
-----
min: 0.0026096484943
max: 1.2180784985
avg: 0.25654713293

Forces
------
Cd:  0.5110469
Cl:  -0.0001558524
Cm:  0.003654472

Cd_WINDOW:   0.510856416
Cl_WINDOW:   7.02600704e-05
Cm_WINDOW:   0.0005792470542
size window:   50

Final Residuals
---------
Ux:  2.51736e-08
Uy:  1.29264e-07
Uz:  6.5365e-06
k:   2.42441e-07
e:   4.79654e-08
omega:   1.09877e-10

Log Files
---------
total status: SUCCESS
log file log.decomposePar:   1
log file log.mapFieldsPar:   1
log file log.reconstructPar:   1
log file log.rhoCentralFoam:   1

Log Files Solver
----------------
execution time: [37730.6]
number iterations: [2685.0]
solver name: ['rhoCentralFoam']
iterations per execution time: [0.07116239868965774]
last modification date: ['2023-10-20 18:37:55']
```

Figure A.5 Automatically Generated Report with All Relevant Metrics After Termination of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam*

Bibliography

[1] Sean Bone. “Bachelor’s Thesis - Comparative Study of Density-based Versus Pressure-based Solvers for Supersonic Flow”. In: Zurich, Switzerland, Spring semester 2020.

[2] Félix Martí Valverde. *GitHub - OpenFOAM Toolchain for Rocket Aerodynamic Analysis*. https://github.com/WyllDuck/OpenFOAM-ToolChain-for-Rocket-Aerodynamic-Analysis. Accessed: 2023 Nov. 2023.

[3] Marshall Gusman, Jeffrey Housman, and Cetin Kiris. “Best Practices for CFD Simulations of Launch Vehicle Ascent with Plumes - OVERFLOW Perspective”. In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. DOI: 10.2514/6.2011-1054. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2011-1054. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2011-1054.

[4] Cetin Kiris et al. “Best Practices for Aero-Database CFD Simulations of Ares V Ascent”. In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. DOI: 10.2514/6.2011-16. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2011-16. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2011-16.

[5] Stuart E. Rogers, Derek J. Dalle, and William M. Chan. “CFD Simulations of the Space Launch System Ascent Aerodynamics and Booster Separation”. In: *53rd AIAA Aerospace Sciences Meeting*. DOI: 10.2514/6.2015-0778. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2015-0778. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2015-0778.

[6] Antonio Viviani, Giuseppe Pezzella, and Egidio D’Amato. “Aerodynamic Analysis with Separation Dynamics of a Launcher at Staging Conditions”. In: 2016. URL: https://api.semanticscholar.org/CorpusID:128355443.

[7] Antonio Viviani and Giuseppe Pezzella. “Launcher Aerodynamic Analysis with Plume Effects”. In: 2018. URL: https://api.semanticscholar.org/CorpusID:208003660.

[8] J. C. Ferris. *Static Stability Investigation of a Single-stage Sounding rocket at Mach Numbers from 0.60 to 1.20*. Tech. rep. 19670020050. Report/Patent Number: NASA-TN-D-4013, Accession Number: 67N29379. VA, United States: NASA Langley Research Center Hampton, July 1967. URL: https://ntrs.nasa.gov/citations/19670020050.

[9] C. D. Babb and D. E. Fuller. *Static Stability Investigation of a Sounding-rocket Vehicle at Mach Numbers from 1.50 to 4.63*. Tech. rep. 19670020031. Report/Patent Number: NASA-TN-D-4014, Accession Number: 67N29360. VA, United States: NASA Langley Research Center Hampton, June 1967. URL: https://ntrs.nasa.gov/citations/19670020031.

[10] Assist. Prof. Managing Director Dr. Franjo Juretić M. Eng. and Ltd. Founding Partner Creative Fields. “User Guide v1.1”. In: May 2015. URL: https://cfmesh.com/wp-content/uploads/2015/09/User_Guide-cfMesh_v1.1.pdf.

[11] *Technical Report - Advanced meshing using OpenFOAM® technology: cfMesh*. Tech. rep. summer 2017.

[12] L. Caretto et al. “Two Calculation Procedures for Steady, Three-Dimensional Flows With Recirculation”. In: vol. 2. Mar. 2007, pp. 60–68. ISBN: 978-3-540-06171-7. DOI: 10.1007/BFb0112677.

[13] R.I Issa. “Solution of the implicitly discretised fluid flow equations by operator-splitting”. In: *Journal of Computational Physics* 62.1 (1986), pp. 40–65. ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(86)90099-9. URL: https://www.sciencedirect.com/science/article/pii/0021999186900999.

[14] *Technical Report - Turbulence and CFD models: Theory and applications*. Tech. rep. Accessed on November 1st, 2023. URL: http://www.dicat.unige.it/guerrero/turbulence2021/slides/lecture4/4practical_estimates.pdf.

[15] Moritz Limpinsel, Dawei Kuo, and Aarohi Vijh. "SMARTS Modeling of Solar Spectra at Stratospheric Altitude and Influence on Performance of Selected III-V Solar Cells". In: Oct. 2018. DOI: 10.1109/PVSC.2018.8547665.

[16] "Wall-Bounded Shear Flows". In: *A First Course in Turbulence*. The MIT Press, Mar. 1972, p. 180. ISBN: 9780262310901. DOI: 10.7551/mitpress/3014.003.0007. eprint: https://direct.mit.edu/book/chapter-pdf/280761/9780262310901_cah.pdf. URL: https://doi.org/10.7551/mitpress/3014.003.0007.

[17] F. Menter. "Zonal Two Equation k-w Turbulence Models For Aerodynamic Flows". In: *23rd Fluid Dynamics, Plasmadynamics, and Lasers Conference*, p. 5. DOI: 10.2514/6.1993-2906. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.1993-2906. URL: https://arc.aiaa.org/doi/abs/10.2514/6.1993-2906.

[18] *Technical Report - Supersonic flow past a wedge*. Tech. rep. Accessed on September 21th, 2023. URL: http://www.wolfdynamics.com/wiki/tut_2D_supersonic_wedge.pdf.

[19] OpenFOAM ESI. *Documentation - User Guide*. Accessed on September 26th, 2023. OpenFOAM ESI. Accessed on 23/09/2023. URL: https://www.openfoam.com/documentation/guides/latest/doc/index.html.