
TOOLCHAIN FOR AERODYNAMIC CHARACTERIZATION OF A ROCKET DURING ASCENT USING OPENFOAM

TECHNICAL REPORT

• **M.Sc. Félix Martí Valverde***
Department of Aerospace
Technical University of Munich
felix.marti@tum.de

• **Prof. Dr.-Ing. habil. Christian Stemmer**
Aerodynamics and Fluid Mechanics
Technical University of Munich
christian.stemmer@tum.de

April 27, 2026

ABSTRACT

Computational Fluid Dynamics tools from providers such as ANSYS (Fluent) or Siemens (Star-CCM+) can be a financial burden for small businesses and startups alike. Thus, open-source alternatives such as OpenFOAM are increasingly appealing for reducing 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. 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 for generating an aerodynamic database for a rocket, including all relevant aerodynamic coefficients. Two solvers, *rhoPimpleFoam*, a pressure-based solver, and *rhoCentralFoam*, a density-based solver, are compared to determine which one better matches the expected aerodynamic coefficients. A wind-tunnel-validated geometry for a rocket has been identified and used to assess the results' validity. Finally, lessons learned regarding the project's organizational structure, including automation, standardization, and scalability, are discussed to showcase their potential application in other industrial cases. All necessary tools for this Computational Fluid Dynamics toolchain are freely available.

Keywords OpenFOAM · supersonic · transonic · subsonic · CFD · automation · aerodynamics · rhoPimpleFoam · rhoCentralFoam · rocket

1 Introduction

1.1 Motivation

The primary objective of this project is to establish a reproducible workflow for setting up a complete Computational Fluid Dynamics (CFD) toolchain to aerodynamically characterize a given rocket geometry. The aerodynamic characterization is performed through the computation of axial (C_A), normal (C_N), and pitching moment ($C_{m_{pitch}}$) coefficients. Surface temperature distributions and shock-wave locations are additionally evaluated to assess solution quality and to support subsequent design phases.

The present work exclusively employs open-source software, thereby avoiding reliance on proprietary CFD packages such as ANSYS Fluent or Siemens Star-CCM+. Due to the substantial licensing costs associated with commercial software, open-source alternatives represent a viable option for small companies, start-ups, and independent developers. Among these, OpenFOAM has gained widespread adoption within the CFD community. In this study, the OpenFOAM ESI Group (ESI) distribution is used.

Section 1.2 reviews current state-of-the-art CFD methodologies for rocket aerodynamic analysis. Chapter 2 presents the selected software tools and describes the automation of key workflow steps using Python. Validation of the proposed

*visit code repository of the project at *GitHub* - OpenFOAM ToolChain for Rocket Aerodynamic Analysis

OpenFOAM framework is addressed in Chapter 3. A rocket geometry previously tested in a NASA wind tunnel is reproduced in SolidWorks and simulated under atmospheric conditions covering subsonic, transonic, and supersonic regimes, as well as a range of AoA values representative of the flight envelope.

Chapter 4 discusses open-source meshing approaches and details the selected meshing strategy. Section 4.2 presents a grid convergence study and evaluates the achieved y^+ values in relation to boundary-layer resolution requirements. Chapter 5 describes the simulation setup in OpenFOAM using the solvers *rhoPimpleFoam* and *rhoCentralFoam*. Boundary conditions for free-stream quantities (U , T , p) are defined separately for subsonic and supersonic regimes. Turbulence modeling considerations are addressed, including model selection, inlet turbulence specification, and boundary-layer treatment. Temporal discretization strategies for accelerating convergence toward steady-state solutions are also discussed. The final chapter presents a detailed analysis of the simulation results obtained across all test cases.

The developed OpenFOAM toolchain is modular and extensible. The associated source code is made publicly available [1] to facilitate reproducibility and further development.

1.2 State of Research

The current state-of-the-art CFD methodologies are implemented at National Aeronautics and Space Administration (NASA) and other aerospace organizations to support aerodynamic, thermal, and structural design processes. Several studies [2, 3, 4, 5, 6] provide methodological guidelines for the aerodynamic analysis of launch vehicles.

Reference [3] presents a systematic validation methodology for CFD simulations through comparison with wind-tunnel measurements of a scaled rocket model. The study emphasizes grid convergence, sensitivity analyses of numerical parameters, evaluation of turbulence models, and code-to-code comparisons.

Previous investigations [2, 6] highlight the influence of exhaust plumes on aerodynamic coefficients and thermal loads. In particular, [6] quantifies plume-induced drag reduction during powered flight up to Mach 5, beyond which the differences become negligible. The modeling approach discussed in [2] compares multi-species and single-species solvers as well as turbulence models (SA and SST). The results indicate improved prediction of the Plume-Induced Flow Separation (PIFS) location when using the SST model. Although multi-species solvers provide improved thermodynamic consistency, single-species solvers were shown to yield acceptable PIFS predictions when pressure and velocity are prescribed consistently.

Advanced meshing techniques such as Adaptive Mesh Refinement (AMR) are discussed in [3]. While adaptive refinement improves local resolution, excessive refinement may introduce numerical artifacts when shocks align with refined regions. Overset mesh approaches [2] offer enhanced geometric flexibility and the advantages of a structured grid. However, due to solver stability limitations in the current OpenFOAM version, an unstructured Cartesian mesh was selected for the present work.

2 Open Source Tools and Automated Workflow

Based on the currently available OpenFOAM distributions (the OpenFOAM Foundation (OF) Foundation release, OpenFOAM Extend, and the OpenFOAM ESI Group (ESI) distribution, as well as other commercially supported variants), the ESI release was selected for this work. The ESI distribution is developed with industrial applicability in mind, providing regularly scheduled releases and long-term maintenance versions. In contrast to the Foundation release, which follows a different development and release model, the ESI version emphasizes stability and structured update cycles, typically occurring every 6 months.

Due to the high computational requirements of CFD analysis, development occurs on a local machine, while execution runs on a server. The code is hosted on a git repository, helping track changes and enabling collaboration. To communicate with the server, MobaXterm for Windows offers an SSH terminal with a full Graphical User Interface (GUI) to navigate the server's folders. Visual Studio Code (VSC) is used to edit the server's code remotely via SSH. For monitoring, GNUPLOT is used to plot the convergence of residuals and aerodynamic coefficients as the CFD solutions are computed. GNUPLOT reads a *dat* file, which is updated at every simulation iteration using Functional Objects (FOs) native to OpenFOAM. This system is coordinated by a bash script that semi-automates the task and plots all relevant residuals or aerodynamic coefficients as needed. Finally, ParaView is used to visualize results, which can be downloaded through MobaXterm or VSC as the simulation computes. Both bash and Python scripts, combined with FOs, provide the building blocks for a semi-automated OpenFOAM framework. More information is provided in Section 2.2.

2.1 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: lightweight, standalone, and executable packages that contain everything needed to run an application. Powerful servers can be rented on demand to run scientific work. A server with Docker preinstalled can be rented and accessed via SSH, and within minutes, OpenFOAM is ready to start computing. Once the work is done, the Docker container closes, liberating the server. 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 conducted on the development laptop (powered by a 4-core CPU) under the same conditions, with minimal interference from other programs. The test was performed on Windows OS and on Linux OS via 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 on Linux in dual-boot mode, 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.

2.2 Workflow

The total number of simulations required to aerodynamically characterize a rocket is substantial, as shown in Table 9 and Table 10. Managing such a large number of simulations can be challenging. To address this issue, a method similar to CFD macros, already popular in other CFD tools, has been developed for OpenFOAM using the Jinja2 templating library. Allowing the OpenFOAM workspace to be customized for a specific case study by adjusting parameters such as flow speed, pressure, and more.

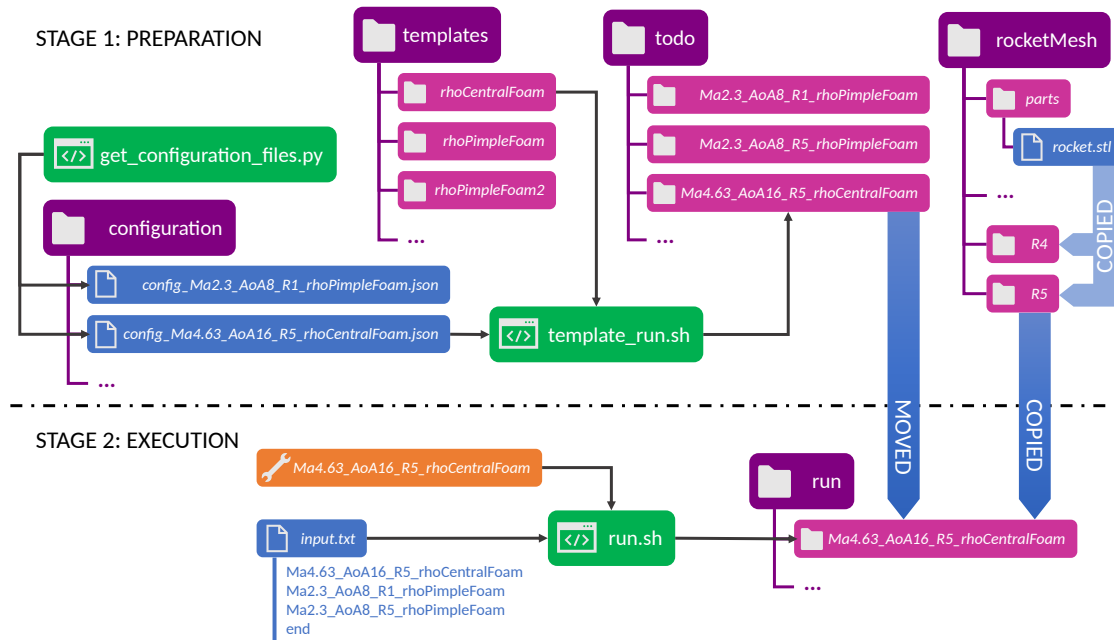


Figure 1: Semi-Automatized Workflow for Pre-Processing and Execution of OpenFOAM Simulations Using Templating Enabled by Jinja2, Python, Bash, and JSON

Instead of creating an OpenFOAM workspace for each case study, templates are base projects from which specific parameters are configured. Templates are OpenFOAM workspaces containing all required folders and scripts for execution (*constant*, *systems*, *0.orig*, *Allrun*, ...), but they lack some parameters needed to execute. Jinja2 fills in the gaps using a JSON configuration file with the additional parameters for each study case. Jinja2 is executed via the *jinja2-cli* application, which, in conjunction with *bash*, creates a semi-automated workflow. Figure 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, such as *rhoPimpleFoam* and *rhoPimpleFoam2*, in the *template* folder. Meshes are pre-calculated and stored in the *rocketMesh* folder for later copying into the executing OpenFOAM simulation.

At execution, a bash script named *run.sh* controls the execution order. 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 are in Section 5.

A limitation of using template projects is that they hinder the user's ability to personalize each case study, as making changes to the template can be tedious or incompatible with the needs of other case studies. Some extreme study cases require manual adjustments before execution. The *todo* folder fulfills this need by allowing user intervention before execution.

For automated post-processing, Python generates a series of images to provide 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. Convergence plots of residuals and force coefficients are exported as *PNG* files and primarily saved in the OpenFOAM workspace. These files are accompanied by a text file containing an automatically generated simulation report. An example of these files can be found in the Annex (refer to Section A). Figures 18, 19, and 20 display all the images generated for preliminary evaluation of results. Figure 21 presents the aforementioned convergence plots, which are not used for real-time monitoring as they are only available after solver convergence. Finally, Figure 22 shows the automatically generated report, including all relevant parameters, to assess the success of each simulation. During the automated post-processing step, all information gathered in each report is compiled into a single table, available in the Annex (see Tables 9 and 10).

3 Validation Strategy

Papers [7, 8] 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 a shorter version 1 m long. 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 is shown in Figure 3. 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 [7, 8]. 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 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 throughout all the wind-tunnel tests in [7, 8]. 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 ($C_{m_{pitch}}$). Calculations were done using a Cross Sectional Area (A_c) of $1.283 \times 10^{-3} \text{ m}^2$ and Characteristic Length (l) of $57.15 \times 10^{-3} \text{ m}$, and these values remain unchanged for every Angle of Attack (*AoA*) and *Ma*. The conditions considered for the benchmark are highlighted in the *Ma* column of Table 1. Each set of atmospheric conditions is studied at three different *AoA* of 0 deg, 8 deg, and 16 deg.

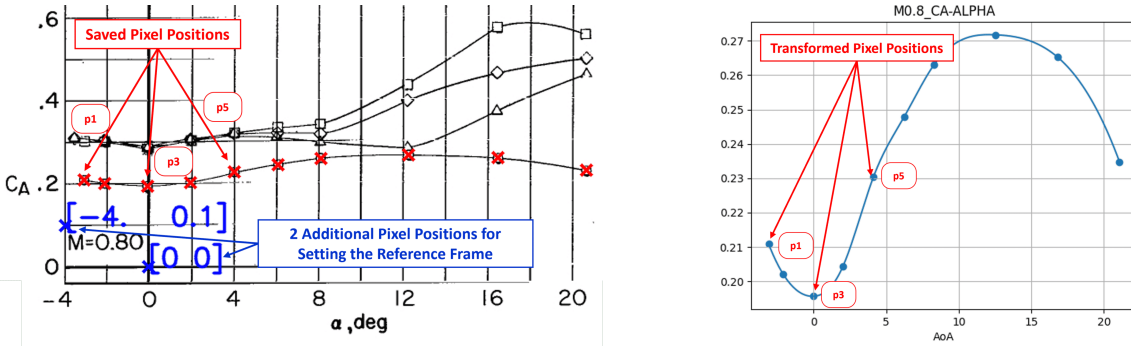
Height [m]	Temperature [K]	Pressure [Pa]	Velocity [m/s]	Mach Number [-]	Density [kg/m ³]
3763	263.71	63665.02	195.33	0.60	0.84
6618	245.18	43444.02	251.12	0.80	0.62
7730	237.96	37124.20	278.32	0.90	0.54
8229	234.73	34539.35	291.78	0.95	0.51
8699	231.68	32259.85	305.14	1.00	0.48
10334	222.81	25538.75	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 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 [7, 8] are extracted using an image processing script due to the poor condition in which the documents present the data. Figure 2 shows an example of this process. Pages with relevant quantities are selected and converted to images. The 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, allowing the recorded points in the document to be extrapolated. A Python script executes this function and uses splines to interpolate the remaining values.

The extracted data's covariance remains high for several reasons. Starting with the pixel assignment, each assigned pixel shows significant covariance due to the low image resolution. Moreover, plots in the images can be distorted or misaligned, leading to increased errors. The most important variation during data extraction occurs in the $C_{m_{pitch}}$ coefficient plot, where the slope vs. AoA is the highest. As discussed in the data sources [7, 8], wind-tunnel support structures affect the recorded aerodynamic coefficients. The rocket model used for CFD analysis lacks a wind-tunnel support structure, leading to changes in the aerodynamic coefficients obtained.

For analysis during engine ignition, the software *RASaeroII* provides the C_A , C_N , and $C_{m_{pitch}}$ coefficients for powered flight of the ARCAS rocket. The values for unpowered flight closely match the data extracted from [7, 8], thus providing a positive indication of the validity of the data for powered flight. The drag reduction is due to the addition of a rocket plume at the nozzle exit, which increases pressure at the rocket's base.



(a) Original C_A vs. AoA Plot of the wind-tunnel Results from [7] (b) Resulting C_A vs. AoA Plot after Data Extraction Process

Figure 2: Example of the Data Extraction Process used to Retrieve the Aerodynamic Coefficients of the wind-tunnel Tested Rocket Geometry in [7] and [8] for Ma 0.8

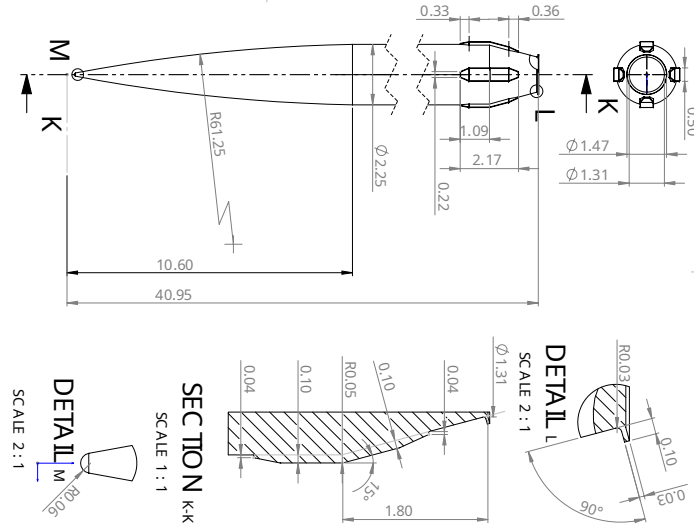


Figure 3: Schematics of the wind-tunnel Tested Rocket Geometry from [7, 8] Reproduced using SolidWorks - Rocket Configuration 1 with no Fins - (units inches)

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 can import meshes from multiple sources, including those from paid acrsort cfd solutions like ANSYS (Fluent) or Siemens (Star-CCM+). Table 2 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 into 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 the 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 number of refinement points and the provided Growth Ratio (GR). Close-by cells can be displaced to provide smoother resulting refinement layers.

Pros	Cons
snappyHexMesh (SHM)	
<ul style="list-style-type: none"> • Default OpenFOAM meshing tool present in all distributions (OF and ESI). • Advanced castellation strategies that work together with blockMesh, allowing for greater control. 	<ul style="list-style-type: none"> • 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.
CfMesh	
<ul style="list-style-type: none"> • Addition of refinement points is very reliable, with no refinement layer collapse. • Installed by default in ESI. • Multi-core enabled, very fast computing. 	<ul style="list-style-type: none"> • Further software updates are only available in CfMesh+, which is a paid software. • Much fewer castellation controls compared to SHM, with no use of <i>blockMesh</i>.

Table 2: Comparison between CfMesh and SHM

Meshing controls, particularly in CfMesh, are minimal, which can be an issue, especially when adding refinement layers. As shown in Figure 17b, these poor control options result in an abrupt change in cell height between the last refinement layer cell and its neighbor, which cannot be avoided. Additionally, CfMesh currently does not support advanced settings for adding refinement layers (e.g., total refinement layer height), making the process highly reliant on local cell-level settings. Smoothing parameters help transition between different cell levels at the refinement layer. This parameter gradually increases or decreases the size of elements as they transition to a different cell level in a neighboring region of the refinement layer. For more information on CfMesh capabilities, see [9]. Automatic refinement in both SHM and CfMesh is not advisable if manual refinement is feasible. CfMesh encountered issues with automatic refinement at junction points between patches, e.g., between the rocket and the symmetry plane or at the corners of the domain box. In these regions, the maximum cell-level is reached, leaving behind unnecessarily small cells.

CfMesh offers a wide range of cell topologies, but based on a performance study discussed in [10], 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. Compared to the size of the rocket, based on Figure 3, 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

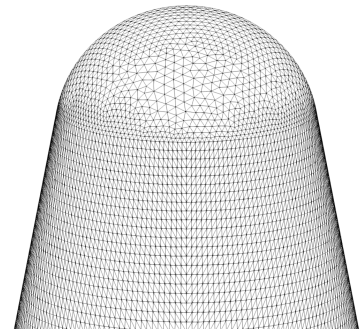


Figure 4: Zoom-In of the *STL* Surface Generated using SALOME at the Tip of the Rocket - Transition Between Structured and Unstructured Surface Mesh

spherical inlets to reduce unnecessary cells in the domain corners, 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.

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 in *STEP* format; thus, a triangulation must be performed to generate the *STL* file. SALOME, apart from being an alternative to SHM and CfMesh, excels at surface preparation. Figure 4 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 used where possible, while unstructured meshes are used for regions with complex geometries, thereby minimizing deflection error and ensuring surface smoothness. Automatic triangulation tools can cause triangulation errors, leaving behind spikes that cause jumps in the deflection error along the surface. These anomalies cause poor snapping results in both SHM and CfMesh. During *STL* file generation, it is possible to define regions that can be key during later meshing.

4.2 Mesh Convergence Study

A mesh-convergence study is necessary to determine the required refinement level to accurately capture the rocket's aerodynamic characteristics. However, due to computational limitations, a maximum cell refinement of approximately 7 million cells is set. Other mesh quality assessments to consider include the rocket's surface quality, 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	n° Cells	n° Ref. Points	GR	n° 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 3: Refinement Characteristic of all the Available Meshes from Coarser (R1) to Finer (R5)

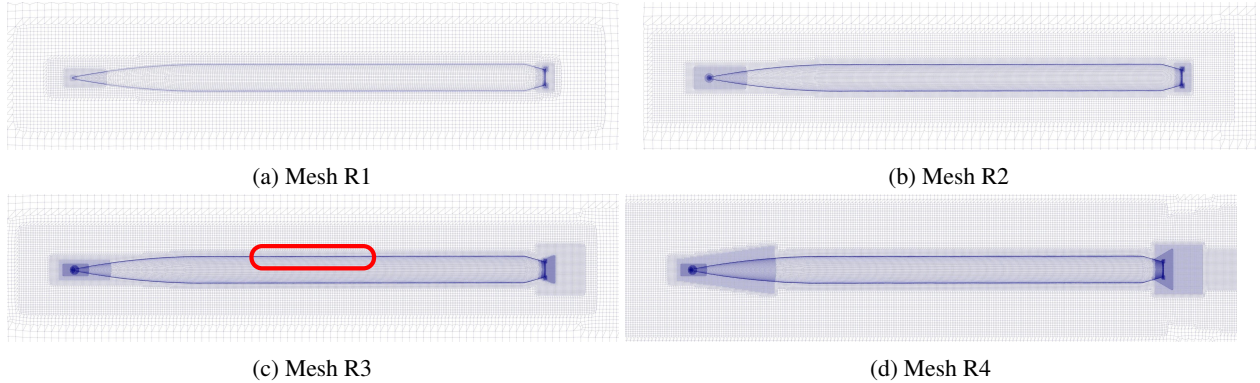


Figure 5: Zoomed-Out View of the Different Mesh Resolutions (R1, R2, R3, R4) - Red box in Mesh R3 indicates the zoom-in region for Figure 6

In Table 3, one can find the main parameters describing the level of refinement of each mesh resolution (R1, ..., R5) used in the convergence study. These parameters are the total number of cells used (n° Cells), and other parameters relative to the refinement layer resolution like GR, number of refinement points used (n° Ref. Points), and the number of cells found at the surface of the rocket (n° Surface Cells). The increase in resolution between meshes can be observed in Figure 5, 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 6. Note that images in Figure 6 are close-ups of the tubular region of the rocket indicated in Figure 5c using a red box. The dimensional relations between the refinement layers and the adjacent cells remain when observing the surface at other regions where cell-level changes occur.

Equation (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 $y^+_{max} < 1$.

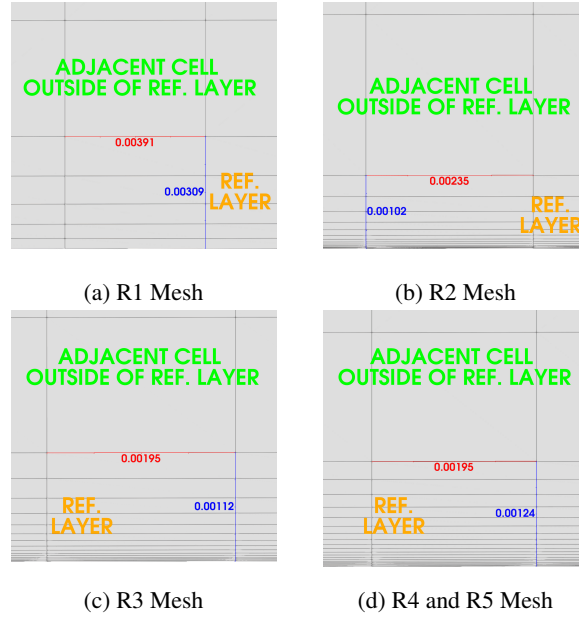


Figure 6: Close Up of Refinement Layers at the Tubular Region of the Rocket for all Mesh Resolutions (**in meters**)

$$\delta = 0.37 \frac{D}{Re^{\frac{1}{5}}} \quad (1)$$

An example of the y^+ values obtained for a *rhoCentralFoam* simulation is shown in Figure 7, which also provides 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 divided by powers of two based on the number of cell levels. Specific regions, such as the nose tip and the rocket base, have refinement levels of 14 and 12, respectively, corresponding to element sizes of 3×10^{-5} m and 1×10^{-4} m, resulting in very low y^+ values. The body of the rocket has y^+ values closer to 1 with element sizes of 2×10^{-3} m as seen in Figure 6d. Further refinement occurs during the addition of the refinement layer. 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 spans a larger range, from 0.7 to 70 Results can be found in Table 9 and Table 10 in the Annex (see Section A).

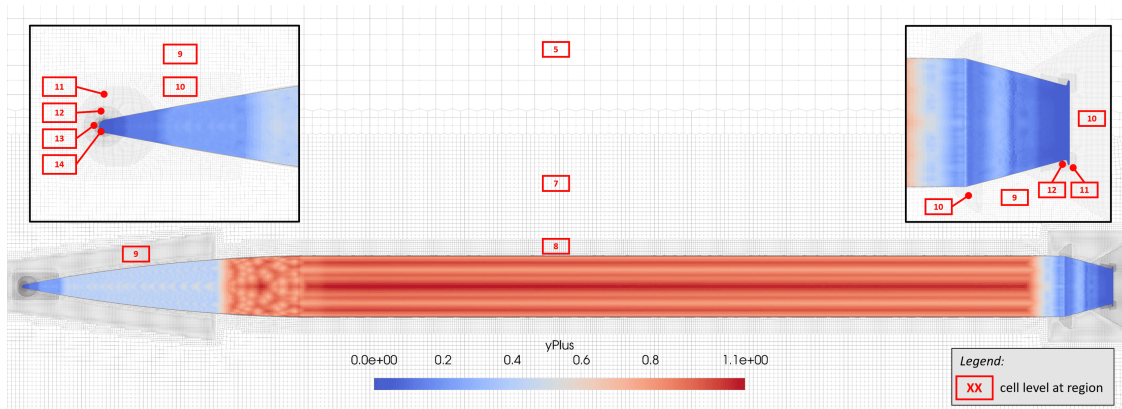
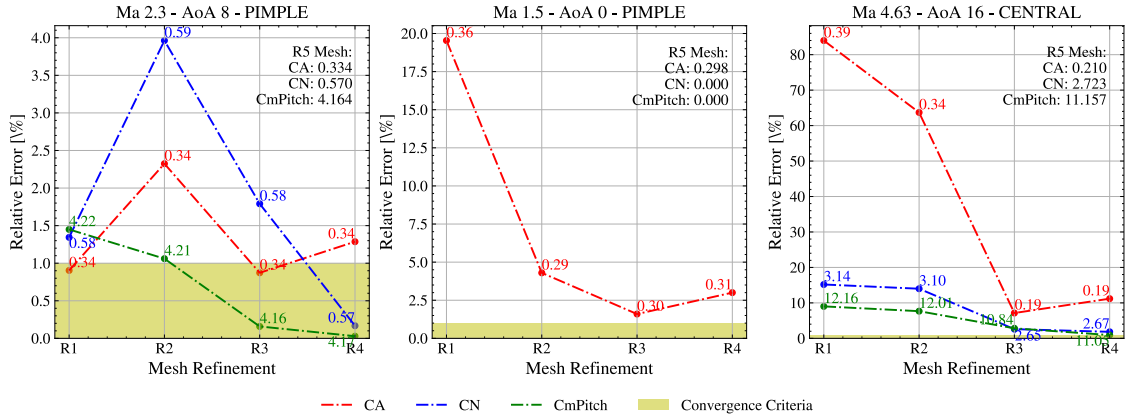


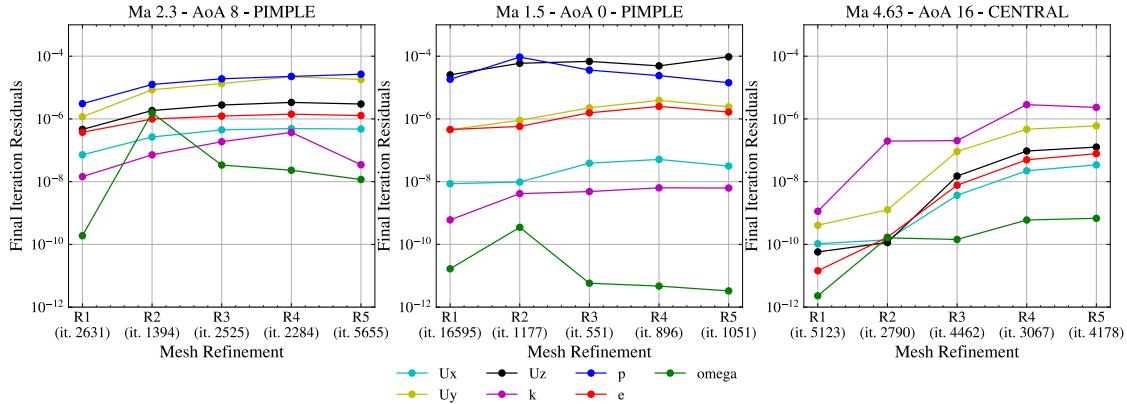
Figure 7: y^+ Values Along the Surface of the Rocket for Simulation Ma 1.0 - AoA 0 deg - Mesh R5 - *rhoCentralFoam* with Cell Refinement Levels around the Rocket for Mesh R5

Figure 8 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 , $C_{m_{pitch}}$) 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 8a for the other meshes (R1, R2, etc.). Unfortunately, from the results obtained in Figure 8a, it is uncertain if the mesh R5 is good enough to accurately solve the cases, as none of the aerodynamic coefficients reach 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 cell mesh should be resolved; thus, the relative error for mesh R5 would be computed using R6 results. Additionally, Figure 8b shows the final residual values for each mesh resolution. On the x-axis, the number of iterations to convergence is shown.



(a) Progression of the Relative Error of Aerodynamic Coefficients (C_A , C_N , $C_{m_{pitch}}$) Calculated Based on 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 8: Results of Convergence Study for Cases Ma 2.3 - AoA 8 deg - ρ oPimpleFoam & Ma 1.5 - AoA 0 deg - ρ oPimpleFoam & Ma 4.63 - AoA 16 deg - ρ oCentralFoam

Regarding the results from Figure 8a, only the convergence study on the left (Ma 2.3 - AoA 8 deg - ρ oPimpleFoam) 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 8a all show increasing convergence as relative errors decrease. However, from all the aerodynamic coefficients measured, C_A stands out in all three studies as having a delayed convergence with respect to the other coefficients (C_N and $C_{m_{pitch}}$). For the convergence study found in the center of Figure 8a (Ma 1.5 - AoA 0 deg - ρ oPimpleFoam), 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 - ρ oCentralFoam) 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 $C_{m_{pitch}}$) drop to values similar to the ones seen in the other convergence studies. The higher relative error observed for meshes R1 and R2 is due to the different interpolation schemes used, unlike those for meshes R3 and R4. As explained in Section 5.4.1, solver ρ oCentralFoam is initialized

using more dissipating interpolation schemes. For meshes 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 in this convergence study (see Figure 8b) could be due to the use of different interpolation schemes.

A hypothesis to explain the tendency of C_A to higher relative errors would be the sensitivity of the CFD analysis to unsteady flow cases. Based on post-processing analysis, the aft of the vehicle exhibits an unsteady recirculation zone, which may be the cause of larger relative errors in C_A than in other coefficients, as the pressure distribution in the aft region is affected by it. The pressure distribution in the aft affects the C_A the most. The unsteady nature may cause the flow at the aft to be fully captured by the steady-state time scheme, leading to slight variations in the relative error for C_A and no apparent tendency to decrease it. The reason to the other coefficients (C_N and $C_{m_{pitch}}$), thus leading to a more chaotic convergence. As a reminder, the selection of a steady state time scheme is discussed in Section 5.3. 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 5.3. Also, the need for finer meshes at higher Ma values, where convergence has been comparatively worse, should be explored. Finally, the addition of rocket exhaust gases would require a new convergence study, following the procedure presented here. Current results are just an indication, not an assurance of sufficient mesh resolution, even with the finest mesh (R5).

5 Simulation Setup

A central project goal is to provide a comprehensive toolchain capable of characterizing the flow in all three aforementioned aerodynamic regimes (subsonic, transonic, and supersonic). The project intends to use two distinct solvers, one based on a pressure-based algorithm and the other on a density-based algorithm. While pressure-based algorithms have been intended in the past to solve subsonic flows, and density-based solvers for supersonic flows, current algorithms in both families are extending their reaches to a broader range of Ma numbers.

CFD algorithms solve a system of three differential equations that express the conservation laws of continuity, momentum, and 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 proceed to calculate the pressure field from it. Pressure-based algorithms combine the continuity and the momentum equations to initially obtain the pressure, from which they derive the density using an equation of state. For a more detailed description of both solvers, reference [11], which dives deeper into the mathematical formulation behind solvers *rhoPimpleFoam* and *rhoCentralFoam*.

Both *rhoPimpleFoam*, a pressure-based solver, and *rhoCentralFoam*, a density-based solver, are used to simulate three distinct aerodynamic regimes (subsonic, transonic, and supersonic). Both are single-species, ideal gas, unsteady, and incorporate the turbulence model $k-\omega SST$. The turbulence model $k-\omega SST$ has been shown in [3] to yield the best results. These solvers are time-transient; however, a first-order time scheme, *localEuler*, transforms them into pseudo-steady-state solvers by applying different time steps to each cell (see Section 5.3). For the viscosity model, Sutherland's law is implemented.

The solver *rhoPimpleFoam* relies on a mix between two pressure-based algorithms, 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, named 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 in the solver *pisoFoam*, which only supports incompressible cases. PIMPLE remains a pressure-based algorithm that, in its OpenFOAM implementation, can solve transient flows for compressible and incompressible fluids, depending on the solver: *pimpleFoam* and *rhoPimpleFoam*, respectively. As discussed in Chapter 6, the limitations of pressure-based solvers become apparent at high Ma numbers, with $Ma < 2.3$, when solutions become nonphysical.

The density-based solver *rhoCentralFoam* is driven based on the central-upwind schemes of Kurganov and Tadmor. This algorithm solves a Riemann problem using approximations to make the solution practical. Many types of Riemann solvers exist; this particular one belongs to the *central schemes* class, and is an improvement on the Lax-Friedrichs scheme.

5.1 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 1. As seen in Figure 9, the domain is composed of five patches. A symmetry wall divides the computational domain into two parts, thereby reducing computational requirements. Due to the lack of complex geometric features such as fins or other potential asymmetries, the rocket is axially symmetric, and thus all *AoA* can be studied within the symmetry wall.

As observed in Table 4a 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 *fixedValue* boundary condition. As long as the flow vector points inwards to the domain, this condition remains the same. If, during the calculation, the flow vector points outward from the domain, then the boundary condition switches to *zeroGradient*. Table 4b presents the boundary conditions required for subsonic and transonic regimes.

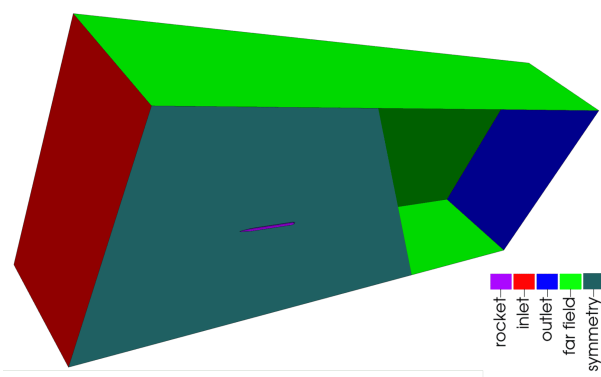


Figure 9: Schematic of Patches For All Meshes (R1, . . . , R5) with Cut View of the *symmetry* Patch to Observe the *outlet* Patch

patch	pressure	temperature	velocity
inlet	<i>fixedValue</i>	<i>fixedValue</i>	<i>fixedValue</i>
outlet	<i>waveTransmissive</i>	<i>advective</i>	<i>inletOutlet</i>
rocket	<i>zeroGradient</i>	<i>zeroGradient</i>	<i>noSlip</i>
far field	<i>waveTransmissive</i>	<i>advective</i>	<i>supersonicFreeStream</i>
symmetry	<i>symmetry</i>	<i>symmetry</i>	<i>symmetry</i>

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

patch	pressure	temperature	velocity
inlet	<i>freestreamPressure</i>	<i>inletOutlet</i>	<i>freestreamVelocity</i>
outlet	<i>freestreamPressure</i>	<i>inletOutlet</i>	<i>freestreamVelocity</i>
rocket	<i>zeroGradient</i>	<i>zeroGradient</i>	<i>noSlip</i>
far field	<i>freestreamPressure</i>	<i>inletOutlet</i>	<i>freestreamVelocity</i>
symmetry	<i>symmetry</i>	<i>symmetry</i>	<i>symmetry</i>

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

Table 4: Two Configurations (Subsonic and Transonic, and Supersonic) for Pressure, Temperature, and Velocity Boundary Condition Types at Every Patch

The rocket's skin is assumed adiabatic, leading to a *zeroGradient* boundary condition for the temperature at the rocket's surface. This boundary condition is appropriate for this study, as the wind-tunnel tests were performed under similar conditions; however, in a more realistic setting, this approximation would deviate significantly from reality. In general, during launch, a rocket's wall temperature is not adiabatic. Sections of the rocket containing liquid oxygen or other

propellants can be at significantly lower temperatures. In a more advanced CFD model, the adiabatic assumption is dropped in favor of more realistic conditions closer to those in flight.

5.2 Turbulence Model Implementation

To model an external flow, the best RANS turbulence model is $k-\omega$, with $k-\omega SST$ being an improved hybrid version that combines $k-\epsilon$ and $k-\omega$ formulations. Boundary conditions for turbulence modeling can be highly confusing in OpenFOAM due to a lack of clear documentation, infrequent 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.

5.2.1 Free-stream Conditions

In addition to the boundary conditions specified in Tables 4a and 4b, other fields are required for $k-\omega SST$. 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 $0.0314 \frac{m^2}{s^2}$.

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

For ω , the Equation (3) for external flows is used [14]. Equation (3) is based on the Eddy Viscosity Ratio μ_t/μ , which takes a value of 0.7 based on the range provided by [14] for external flows. Using this formula to compute all the different study cases leaves ω ranging from 2300 to $300 \frac{1}{s}$ as the rocket ascends.

$$\omega = \frac{\rho_\infty k}{\mu} (\mu_t/\mu)^{-1} \quad (3)$$

Boundary conditions for $k-\omega SST$ are discussed in the following section, where *inlet*, *outlet*, *far field*, and *symmetry* boundary types can be found in Tables 5 and 6. OpenFOAM provides alternatives to manually calculating the inlet conditions for the turbulence model. *turbulentIntensityKineticEnergyInlet* inherits from *inletOutlet* but automatically applies Equation (2) 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.

5.2.2 Boundary Condition Types for Walls

For boundary conditions on *noSplit* walls, it is imperative to consider the expected y^+ value. As seen in Figure 10, within the viscous sublayer ($y^+ < 5$), 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, which transitions between these two regions, can be challenging to model. To accurately model the boundary layer around the rocket, it is important to use a fine mesh with a $y^+_{max} < 5$.

To maximize accuracy, wall models should be avoided. A direct resolution of the boundary layer is preferred, even though it incurs a higher computational cost due to the use of finer meshes. Two candidate configurations for turbulent boundary condition types are proposed. The configuration in Table 6 should only be used in fine meshes which meet the criterion $y^+_{max} < 1$. On the other hand, Table 5 provides a configuration that fits all mesh qualities, including fine meshes and coarse meshes with $y^+_{max} > 30$.

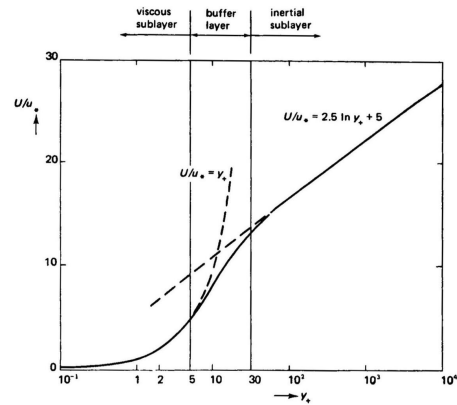


Figure 10: Law of the wall [16]

With a Wall Model

Because OpenFOAM meshes rely on castellation techniques, y^+ values on coarse meshes range between $y^+ < 1$ to $y^+ > 30$. The boundary condition needs to switch between the low and high Re regions as seen in Figure 10. To adapt the formulation, Table 5 shows a set of adaptable wall conditions for the k - ω SST turbulence model used in coarse and fine meshes ($y^+_{max} < 1$) regions.

patch	k	ω	ν	α
inlet	<i>inletOutlet</i>	<i>inletOutlet</i>	<i>calculated</i>	<i>calculated</i>
outlet	<i>inletOutlet</i>	<i>inletOutlet</i>	<i>calculated</i>	<i>calculated</i>
rocket	<i>kLowReWallFunction</i>	<i>omegaWallFunction</i>	<i>nutkWallFunction</i>	<i>calculated</i>
<i>rocket ALT.</i>				<i>alphatWallFunction</i>
far field	<i>inletOutlet</i>	<i>inletOutlet</i>	<i>calculated</i>	<i>calculated</i>
symmetry	<i>symmetry</i>	<i>symmetry</i>	<i>symmetry</i>	<i>symmetry</i>

Table 5: 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 10. For $y^+ < 11$, the viscous sublayer approximation is used; above this threshold, the log-law is 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 model, similar to *kLowReWallFunction*. ν is set to zero in the viscous layer, and Equation (4) takes over in the log-law region.

$$\nu_{tlog} = \nu_w \left(\frac{y^+ \kappa}{\ln(Ey^+)} - 1 \right) \quad (4)$$

Finally, α uses the *alphatWallFunction*, which applies equation (5) directly.

$$\alpha_t = \frac{\nu_t}{Pr_t} \quad (5)$$

Both configurations make use of the same boundary condition, *omegaWallFunction* for ω , which makes use of the original Equation (6) 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:

$$\omega_{vis} = \frac{6\nu_w}{\beta_1 y^2} \quad (6)$$

Note that this equation cannot be considered a wall model as it is an intrinsic part of the turbulence model k - ω SST. 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

To eliminate the need for wall models, the mesh must have a high resolution in the boundary layer, allowing turbulence to be modeled without them. The boundary condition types in Table 6 deactivate all the wall models previously presented. However, under certain conditions, the aforementioned configuration in Table 5 becomes equivalent to the following configuration in Table 6.

The k condition becomes a *fixedValue* boundary condition with a uniform minimal 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, to be able to measure y^+ . Regarding the equivalency of the two proposed boundary condition sets in Table 6 and Table 5. In a fine enough mesh if $y^+_{max} < 1$, then Tables 5 and 6 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.

patch	k	ω	ν	α
inlet	<i>inletOutlet</i>	<i>inletOutlet</i>	<i>calculated</i>	<i>calculated</i>
outlet	<i>inletOutlet</i>	<i>inletOutlet</i>	<i>calculated</i>	<i>calculated</i>
rocket	<i>fixedValue</i>	<i>omegaWallFunction</i>	<i>calculated</i>	<i>calculated</i>
<i>rocket ALT.</i>			<i>nutLowReWallFunction</i>	<i>alphatWallFunction</i>
far field	<i>inletOutlet</i>	<i>inletOutlet</i>	<i>calculated</i>	<i>calculated</i>
symmetry	<i>symmetry</i>	<i>symmetry</i>	<i>symmetry</i>	<i>symmetry</i>

Table 6: 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

In conclusion, the setup presented in Table 5 is equivalent to Table 6 for fine meshes ($y^+_{max} < 1$). All simulations, regardless of the mesh resolution used (R1, ..., R5), have used the configuration presented in Table 5, as it can be applied to a wider range of mesh resolutions, leading to a more uniform OpenFOAM code across simulations. However, this choice does not undermine the solution of mesh R5, which aims to produce results without using wall models. As mesh resolution at the walls increases, wall models implemented in Table 5 transition to the configuration presented in Table 6. Based on the y^+ values presented in Chapter 4.2 for mesh R5 in Figure 7, it can be observed that wall models are not enabled during simulations using the R5 mesh.

5.3 Time Scheme - Euler vs. localEuler

To accelerate convergence to steady-state, Local Time Stepping (LTS) is preferred over the alternative transient-time scheme, *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. 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].

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 11 shows the time dependency of aerodynamic coefficients C_A , C_N , and $C_{m_{pitch}}$ over a simulated time period of 133.82×10^{-3} s.

Figure 11 concludes that steady-state can be achieved for conditions at Ma of 2.3 and AoA of 16° using the R1 mesh. Due to the limited computational budget, the transient response analysis is performed only 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 11 should be valid for the entire flight envelope, as there are no indications of the contrary, either regarding the results in Section 6.

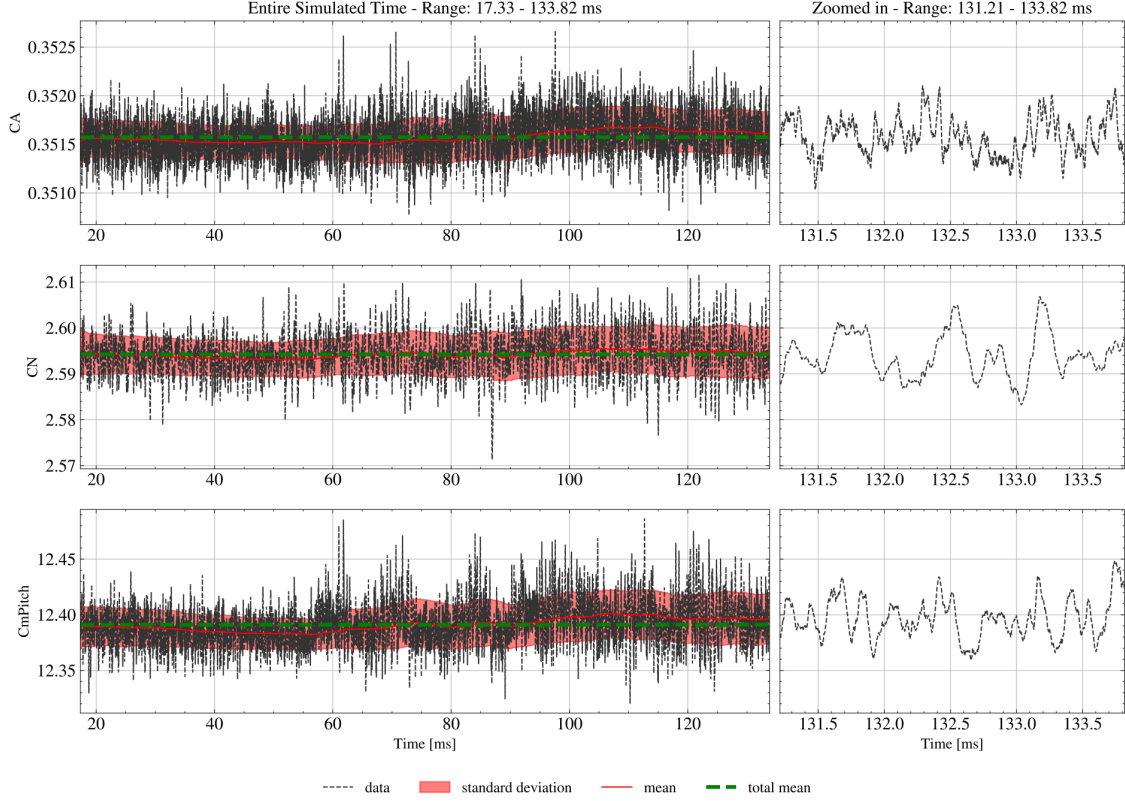


Figure 11: Change of Aerodynamic Coefficients (C_A , C_N , $C_{m_{pitch}}$) over Simulated Time for Simulation Ma 2.3 - AoA 16 deg - R1 Mesh - $\rho PimpleFoam$ Using the Time Scheme *Euler* for Transient Solutions

5.4 Execution Strategy

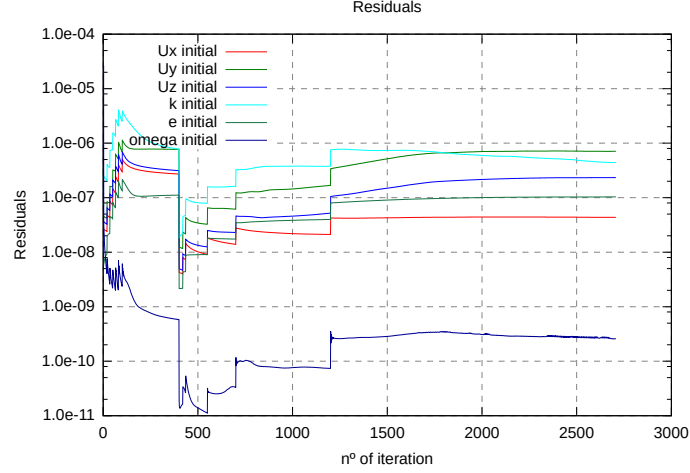
To accelerate the convergence, simulations are first executed using a very coarse mesh, mesh R1 in Table 3. This provides a rough estimate to initialize the fields in the final, finer mesh (R5). The OpenFOAM application *mapFieldsPar* maps fields (U , p , T , ...) from one mesh to another using multi-threading for accelerated computation, drastically reducing the time required compared to *mapFields* (single-thread).

To terminate simulations, the stabilization of aerodynamic coefficients C_A , C_N , and $C_{m_{pitch}}$ is monitored. This is implemented using an OpenFOAM FOs called *runTimeControl*, which stops the simulation when the convergence criteria are met. For this project, the only convergence criterion used is the stability of the 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.

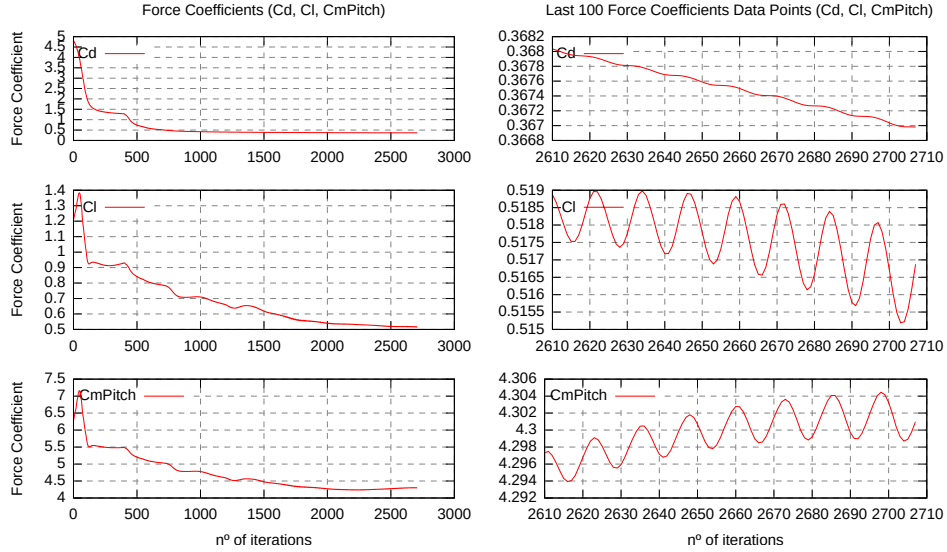
5.4.1 Specific for *rhoCentralFoam*

The solver *rhoCentralFoam* is known to be unstable and prone to crashing due to numerical issues. Multiple strategies have been tested to achieve stability. Initializing the flow using a basic solver such as *potentialFlow*, which is inviscid and irrotational, has yielded minimal success and, in some cases, 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 velocity interpolation schemes, i.e., *upwind*, has further resolved convergence and numerical issues. *upwind* is a first-order bounded interpolation scheme; ideally, the preferred interpolation scheme would be *vanLeer*, a second-order scheme that provides more precise interpolation, leading to sharper shock waves and thus more accurate 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 needs to modify, and each timeline lists the substitution files for that 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 12: Example of Real-Time Monitoring Plots used on Simulation Ma 2.3 - AoA 8 deg - R5 Mesh - *rhoCentralFoam* on a Converged Case

Figures 12 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 12a up to iteration 100 with a set of incremental steps in the residuals. After the aerodynamic coefficients have plateaued in Figure 12b, 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 is accomplished, thus leading to the activation of the convergence criteria based on aerodynamic coefficient discussed in Section 5.4. The timing between Co_{max} transitions and the interpolation schemes can be difficult to determine. Changes in the simulation's boundary conditions, such as speed, pressure, and temperature, result in 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.

Figure 12b shows an acute drop in the aerodynamic coefficients following the transition from *vanLeer* to *upwind*. Before the interpolation scheme changed to *vanLeer*, coefficients converged to higher values for all C_A , C_N , and $C_{m_{pitch}}$ quantities; only after the scheme change did the flow solution 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 12a shows stable residuals before convergence with the Turbulent Kinetic Energy (k) continuing to drop. On the other hand, Figure 12b shows good converged results when looking at the totality of the iterations on the three plots on the left. However, when observing the last 100 iterations on the right, in particular for C_A named C_d in the plot, the coefficient continues 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 prolong the number of iterations and assess 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 [7, 8]. The extraction of aerodynamic coefficients is not the only requirement for a solver at a specific aerodynamic regime. Factors such as wall temperatures and shock position are also to be observed. In the annex, Tables 9 and 10 contain all the relevant information for each simulation. This information includes y^+ values, aerodynamic coefficients (C_A , C_N , $C_{m_{pitch}}$), the number of iterations, execution times, and more. All simulations have been executed on a workstation with a 16-core CPU and are initialized with the coarse R1 mesh, resulting in a y^+_{max} of approximately 60. Once converged, the results are mapped onto the finer mesh, yielding a final y^+_{max} of 0.8-1.5. The coarser mesh computes, on average, 100 times faster per iteration than the finer mesh. It has not been possible to establish the number of iterations required before solution convergence.

A comparative study of the aerodynamic coefficients from *rhoPimpleFoam*, *rhoCentralFoam*, and the wind-tunnel data is performed in Figures 14 and 13. 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, 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 against wind-tunnel data, the relative errors for C_A are higher in both solvers due to a spike in drag near the transonic region. This spike in drag, seen in Figure 14, should be studied further to assess the validity of the solvers around this region. The unsteady nature of flow at the aft of the rocket could be the cause of higher relative errors observed in Figure 14 for C_A compared to errors for $C_{m_{pitch}}$ and C_N in Figure 13, the latter showing good results for both coefficients, with no high uncertainty around the transonic region.

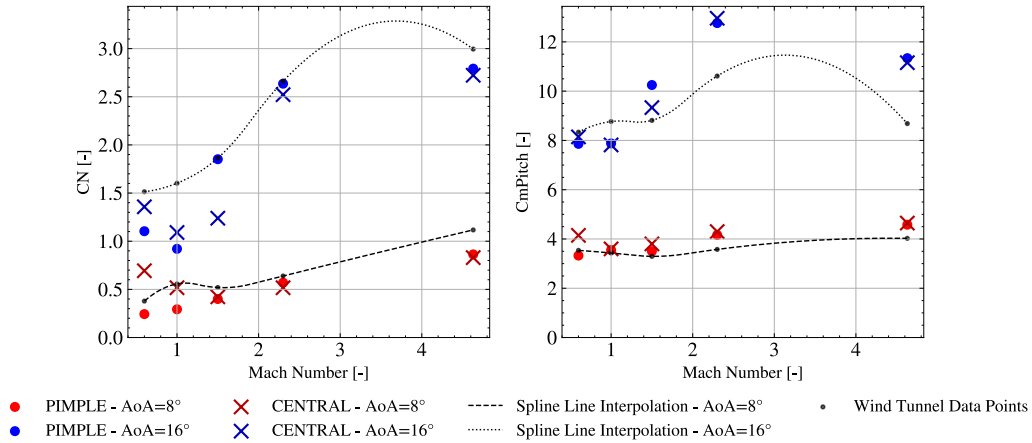


Figure 13: Comparison of Aerodynamic Coefficients C_N and $C_{m_{pitch}}$ for Free-stream Conditions at AoA of 8 deg and 16 deg Obtained From *rhoPimpleFoam* and *rhoCentralFoam* across all Ma Numbers with wind-tunnel Data from [7, 8]

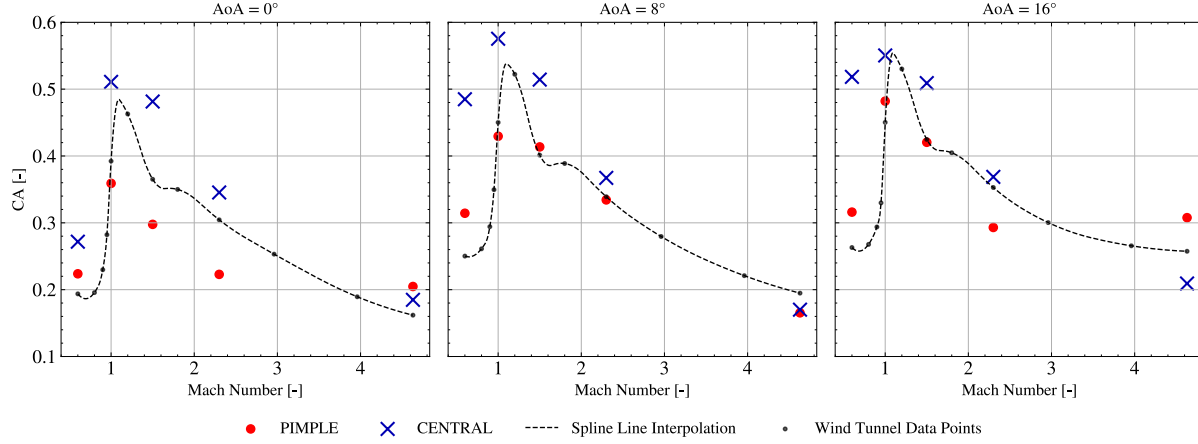


Figure 14: Comparison of the Drag Coefficient C_A for Free-stream Conditions at AoA of 0 deg, 8 deg, and 16 deg Obtained From *rhoPimpleFoam* and *rhoCentralFoam* across all Ma Numbers with wind-tunnel Data from [7, 8]

Figure 15 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 1, with blue and purple zones showing a decrease in c_p while red zones mark an increase. Additionally, the rocket's surface temperature is indicated using a modified rainbow color scheme. As the rocket transitions from subsonic to supersonic, the two solutions begin to diverge. As shown in Figure 15c, the aft region exhibits two distinct 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 most significant 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. 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.

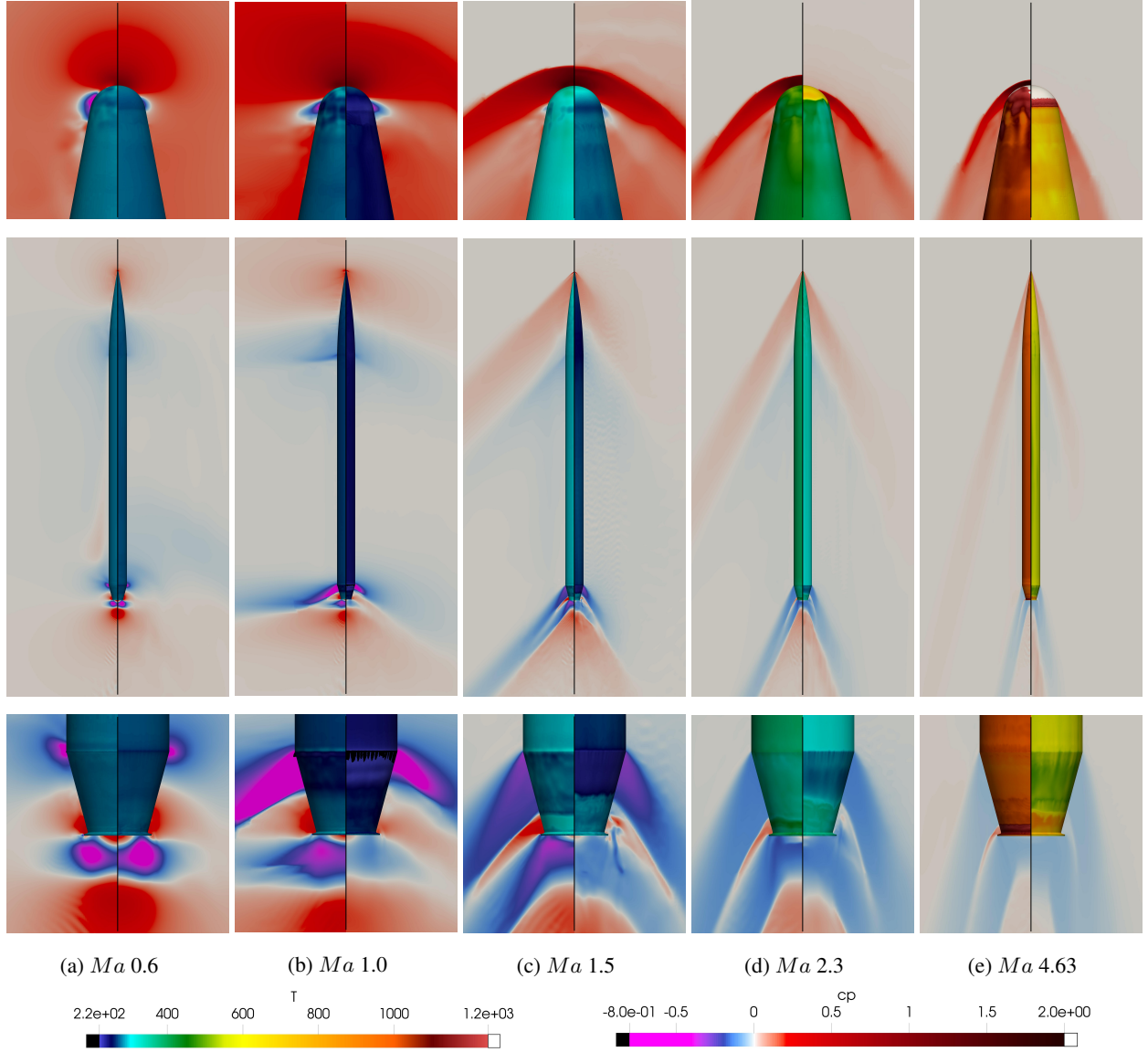


Figure 15: Comparison of Solutions From *rhoCentralFoam* (left) and *rhoPimpleFoam* (right) Across All Studied Ma Numbers at an AoA of 0 deg Using c_p and Skin Temperature

A more in-depth analysis of the differences between the two solvers is performed at the aft of the vehicle. Figure 16 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 critical parameters are considered: wall shear stress to detect potential boundary separation regions, and streamlines to track flow evolution downstream. It's worth noting that streamlines initiate at the same positions in both cases. Based on additional information gathered in Figure 15d, it can be inferred that shock waves initiate at the drop in wall shear stress for both solutions. This information, in combination with Figure 16, 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 yield different results, which should raise questions about the validity of at least one of them.

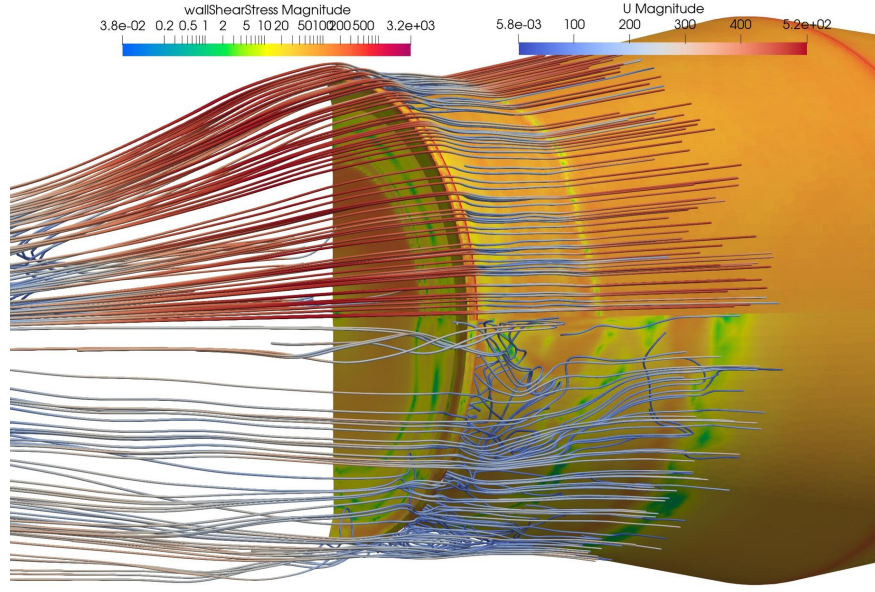


Figure 16: Comparison of *rhoCentralFoam* (upper) and *rhoPimpleFoam* (lower) Flow Separation at the Aft of the Rocket Using Wall Shear Stress and Streamlines

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, including subsonic, transonic, and supersonic. Then, by establishing a repetitive, modular toolchain comprising pre-processing, execution, and post-processing steps, aimed for use in an industrial environment. This toolchain was built with semi-automated workflows in mind, leading to significant productivity gains, including generating an OpenFOAM workspace with Jinja2 templates and creating an execution pipeline that automatically feeds simulations to the server without human intervention. The combination of these automated processes in OpenFOAM enabled the user to ensure uniformity and repeatability across multiple simulations. Finally, the toolchain was tested on a geometry and two solvers, thereby accomplishing the third and final objective of the project: finding a well-suited solver for the aerodynamic characterization of a rocket.

Gained experience from this project shows that *rhoPimpleFoam* is numerically more stable than *rhoCentralFoam*. 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 vehicle's tip, the solver still cannot 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, to transition 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*. Despite 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 resolve 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 remembered that the aerodynamic database stretches over multiple aerodynamic regimes.

Adding engines will further strain computational resources by increasing the number of cells. AMR techniques could leverage the results of this work to refine the mesh only where needed, thereby reducing the size of the initial 6.5 million-element mesh currently used. Furthermore, using a structured mesh to add the engines could reduce the number of cells required in the system. To accommodate such a mesh to the already existing one, the use of *cyclicAMI* could be

beneficial. Further assessments of the solution's sensitivity to CFD parameters, such as mesh element size, turbulence model selection, and cell interpolation schemes, should be the focus of further study by expanding on the work presented here, which is limited by the computational resources available. Finally, to improve the developed toolchain for this work, the addition of Continuous Integration and Continuous Deployment (CI/CD) systems via git could help ensure its scalability and cement its use in industrial environments.

References

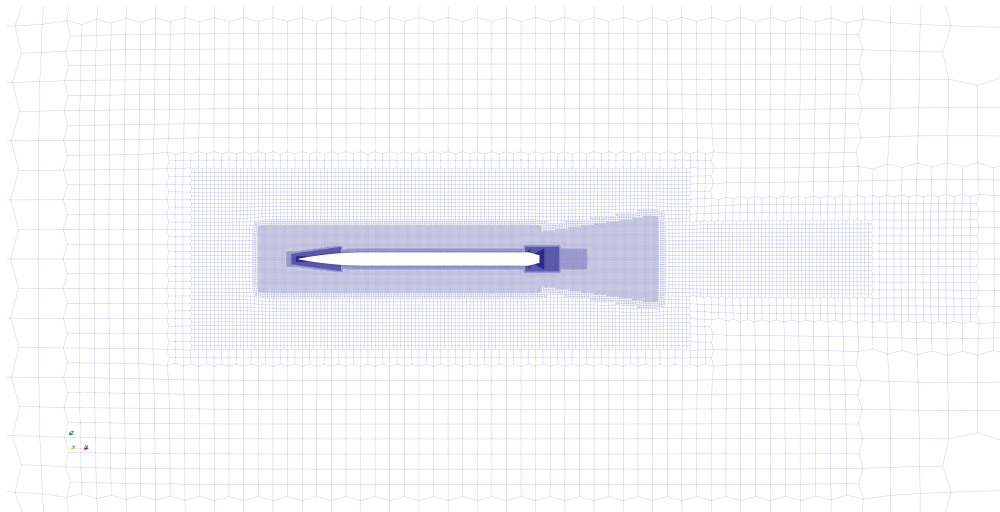
- [1] Félix Martí Valverde. *GitHub - OpenFOAM Toolchain for Rocket Aerodynamic Analysis*. <https://github.com/WyllDuck/OpenFOAM-ToolChain-for-Rocket-Aerodynamic-Analysis>. Accessed: 2023 Nov. 2023.
- [2] Marshall Gusman, Jeffrey Housman, and Cetin Kiris. "Best Practices for CFD Simulations of Launch 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>.
- [3] 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>.
- [4] 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>.
- [5] 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>.
- [6] Antonio Viviani and Giuseppe Pezzella. "Launcher Aerodynamic Analysis with Plume Effects". In: 2018. URL: <https://api.semanticscholar.org/CorpusID:208003660>.
- [7] 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>.
- [8] 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>.
- [9] 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.
- [10] *Technical Report - Advanced meshing using OpenFOAM® technology: cfMesh*. Tech. rep. summer 2017.
- [11] Sean Bone. "Bachelor's Thesis - Comparative Study of Density-based Versus Pressure-based Solvers for Supersonic Flow". In: Zurich, Switzerland, Spring semester 2020.
- [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](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>.

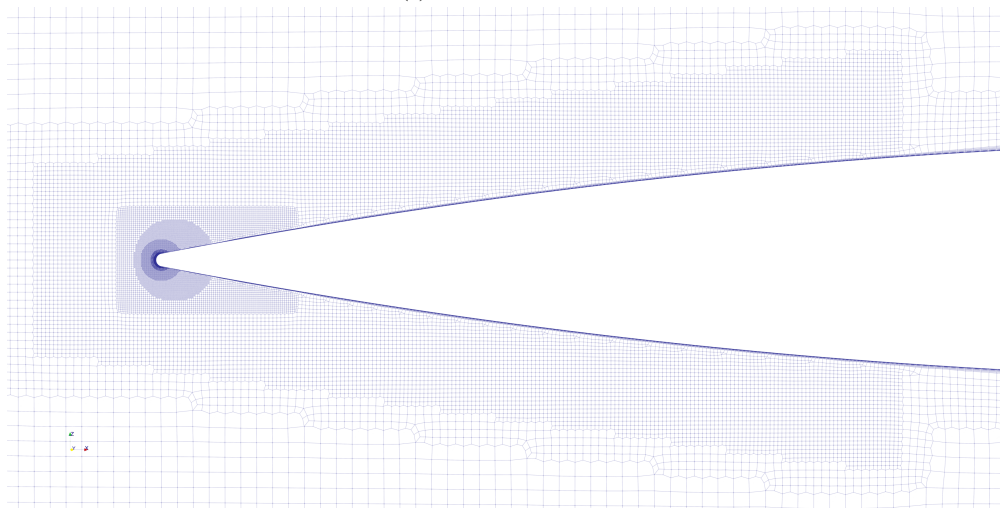
A Annex

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

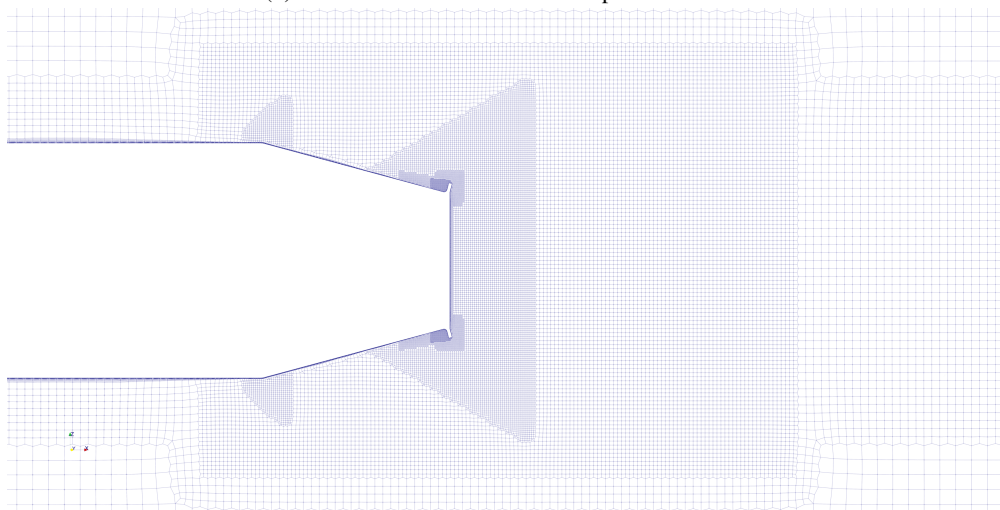
Acronym	Description
AMR	Adaptive Mesh Refinement
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
ESI	OpenFOAM ESI Group
FOs	Functional Objects
GR	Growth Ratio
GUI	Graphical User Interface
LTS	Local Time Stepping
NASA	National Aeronautics and Space Administration
OF	OpenFOAM Foundation
OS	Operating 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
TUM	Technical University of Munich
VSC	Visual Studio Code
WSL	Windows Subsystem for Linux



(a) Zoomed-Out View



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



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

Figure 17: View of Cell Distribution Along the Symmetry Plane for Mesh R5

Table 9: 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 COEFFICIENTS AVERAGE 50 LAST ITERATIONS			TIMING			RESIDUALS						
Ma	AoA	Mesh	Solver	min.	max.	avg.	CA	CN	CmPitch	CA.	CN.	CmPitch.	Execution Time [s]	Number of Iterations	Iterations 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	2.54.

Table 10: Automatically Generated Report on All Simulations Performed for the Aerodynamic Characterization of the Rocket using *rhoCentralFoam* and *rhoPimpleFoam* - Page 2

4.63	0	R1	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

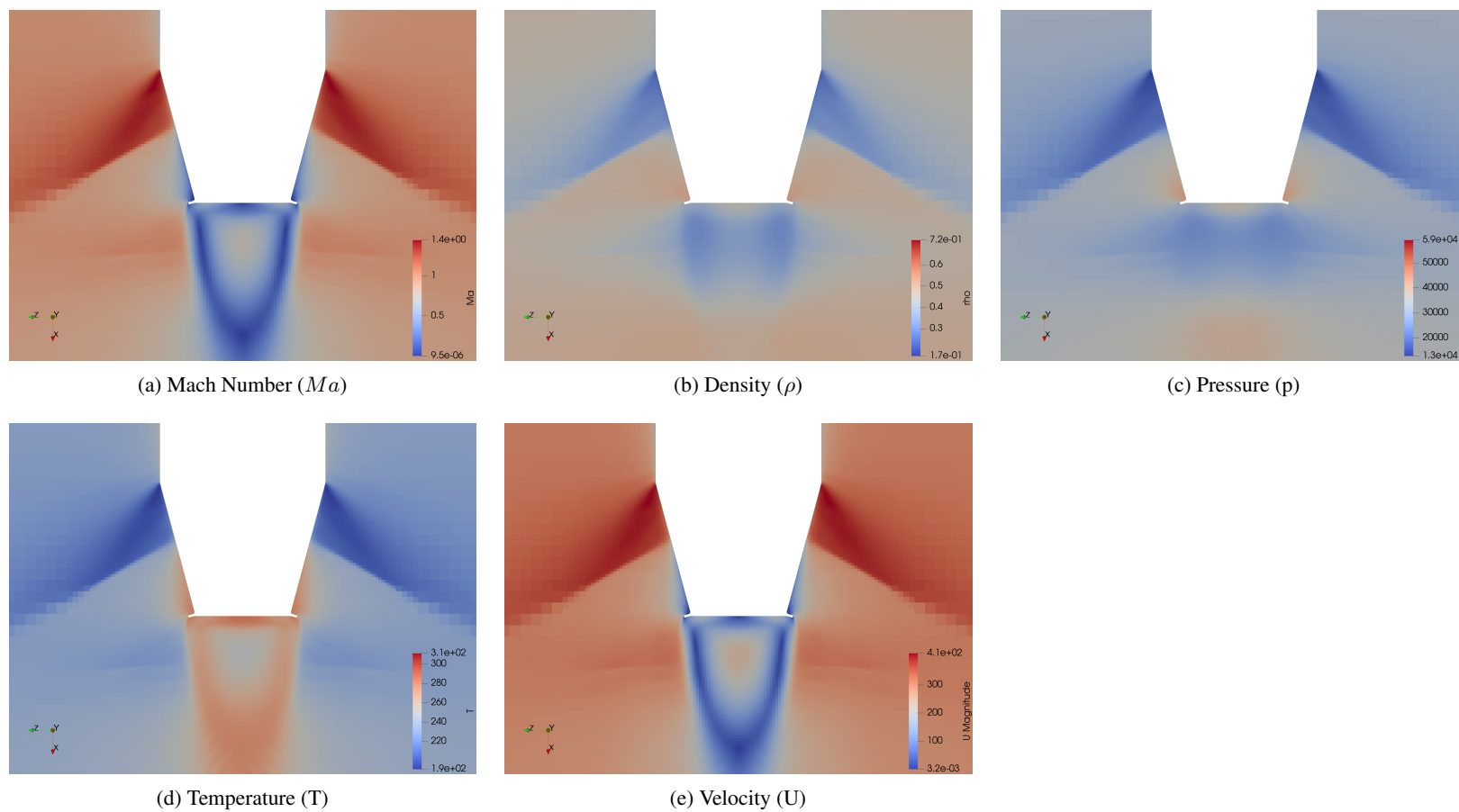


Figure 18: Various Images for Ma , Density, Pressure, Temperature, and Velocity Generated Automatically After Convergence of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam* Using the ParaView Python API - Zoomed Region at the Aft of the Rocket

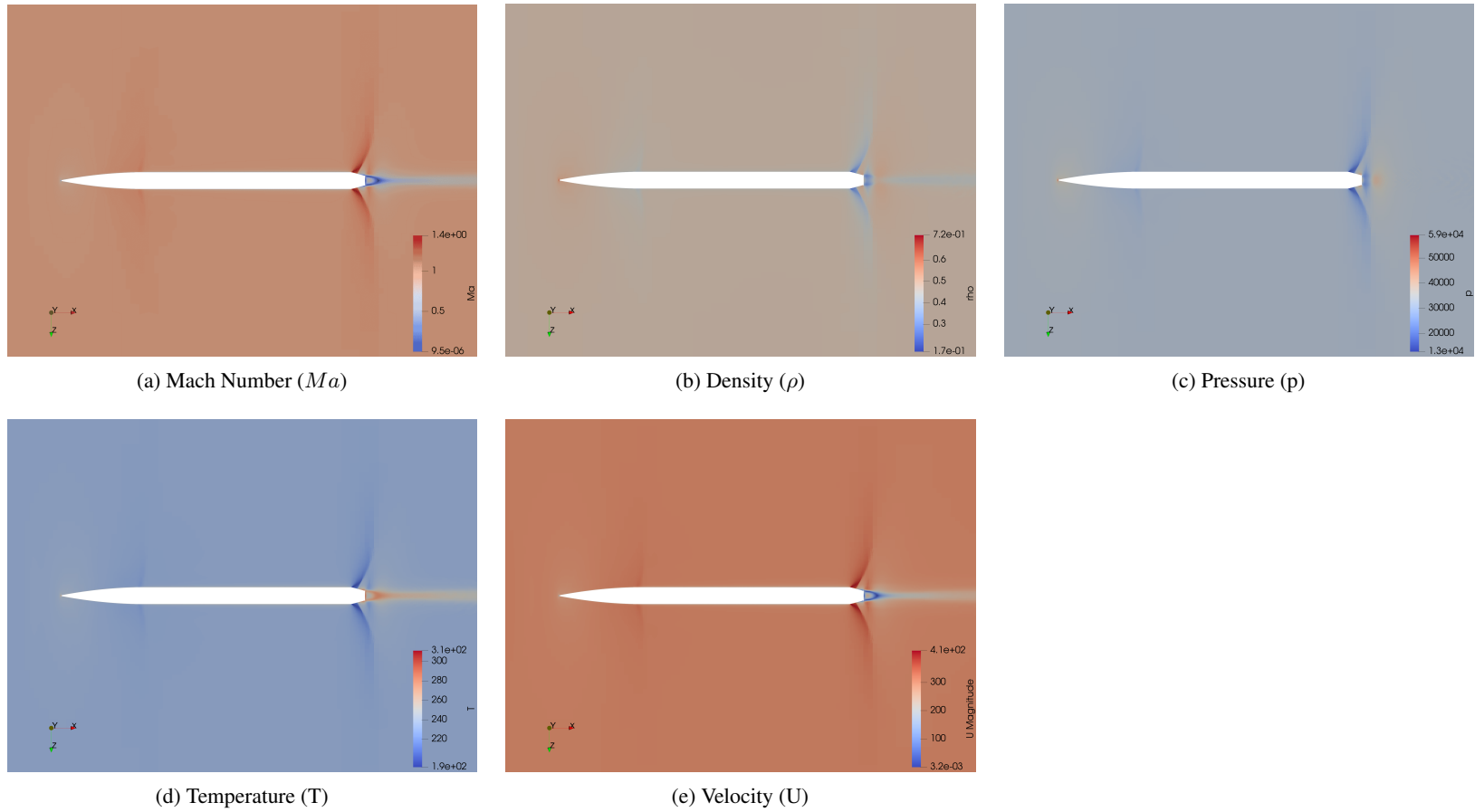


Figure 19: Various Images for Ma , Density, Pressure, Temperature, and Velocity Generated Automatically After Convergence of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam* Using the ParaView Python API - General View of Flow Around the Rocket

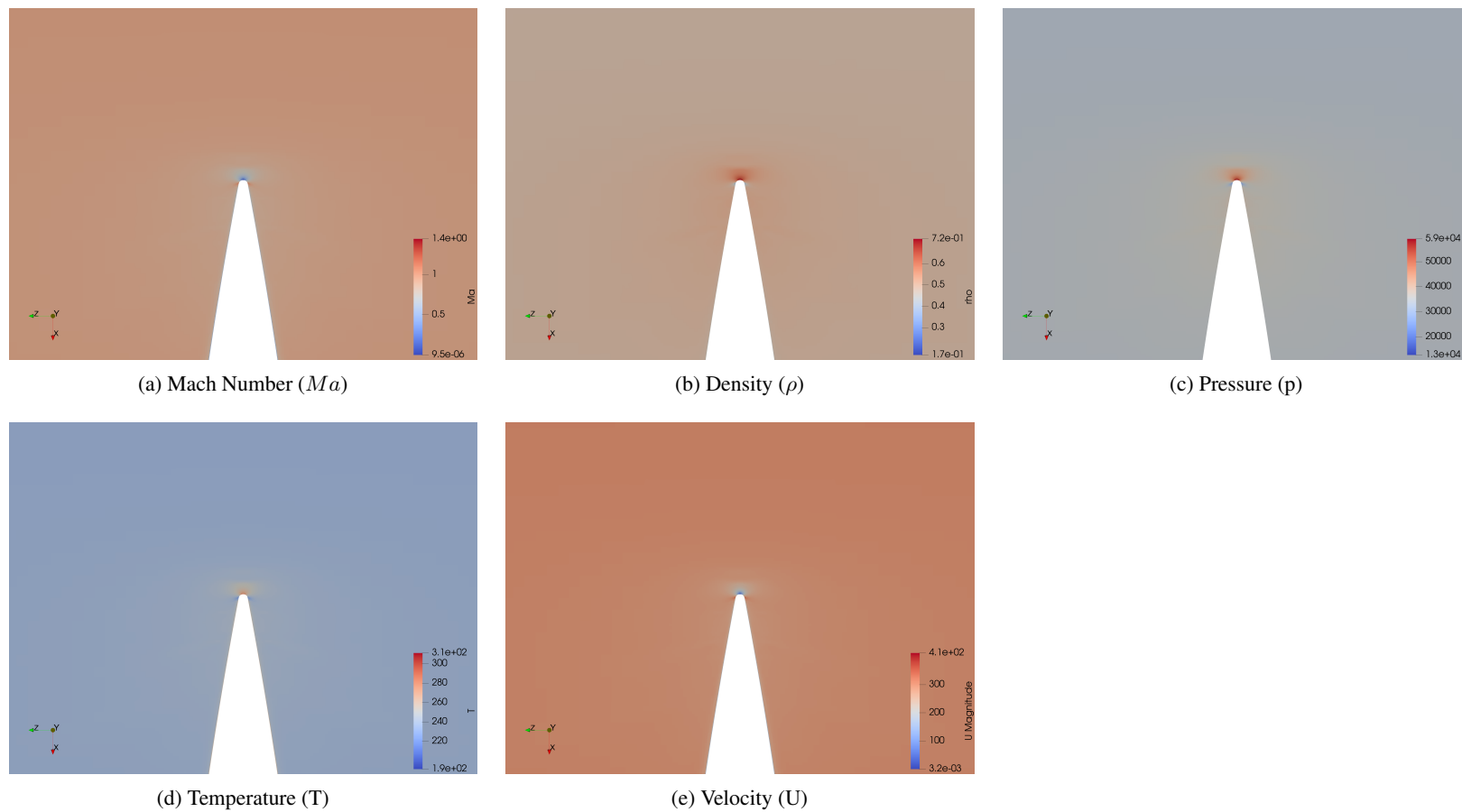


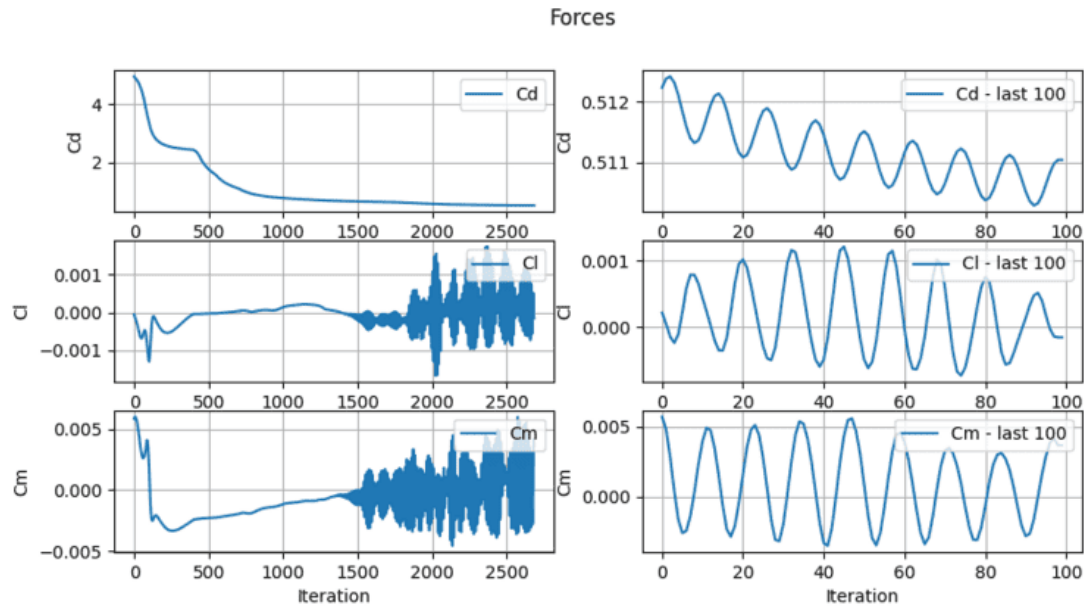
Figure 20: Various Images for Ma , Density, Pressure, Temperature, and Velocity Generated Automatically After Convergence of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - *rhoCentralFoam* Using the ParaView Python API - Zoomed Region at the Tip of the Rocket

```

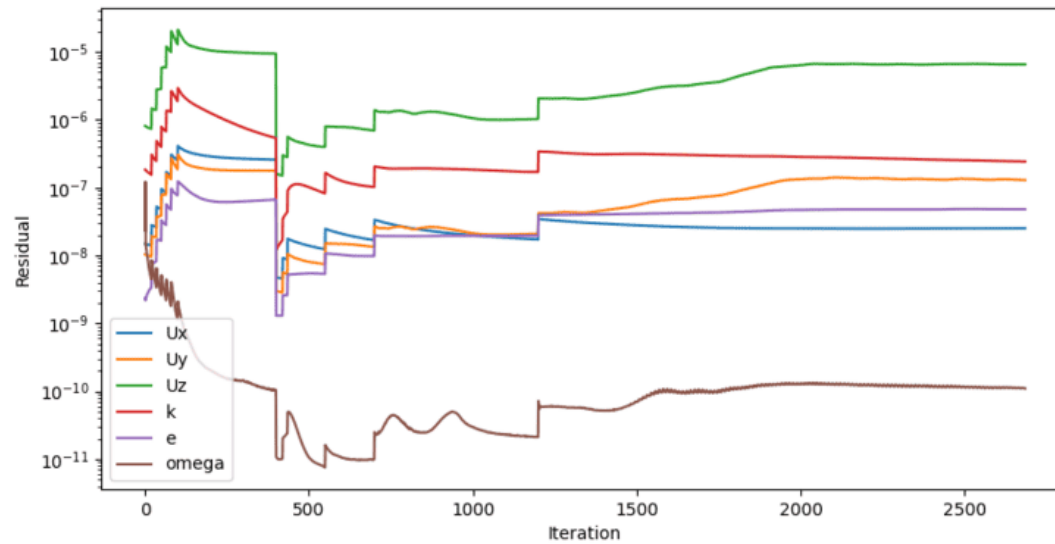
1 Report
2 case: D:\good\rhoCentralFoam\R5\test\Ma1.0
   _AoA0_R5_rhoCentralFoam
3 -----
4
5 yPlus
6 -----
7 min: 0.0026096484943
8 max: 1.2180784985
9 avg: 0.25654713293
10
11 Forces
12 -----
13 Cd: 0.5110469
14 Cl: -0.0001558524
15 Cm: 0.003654472
16
17 Cd_WINDOW: 0.510856416
18 Cl_WINDOW: 7.02600704e-05
19 Cm_WINDOW: 0.0005792470542
20 size window: 50
21
22 Final Residuals
23 -----
24 Ux: 2.51736e-08
25 Uy: 1.29264e-07
26 Uz: 6.5365e-06
27 k: 2.42441e-07
28 e: 4.79654e-08
29 omega: 1.09877e-10
30
31 Log Files
32 -----
33 total status: SUCCESS
34 log file log.decomposePar: 1
35 log file log.mapFieldsPar: 1
36 log file log.reconstructPar: 1
37 log file log.rhoCentralFoam: 1
38
39 Log Files Solver
40 -----
41 execution time: [37730.6]
42 number iterations: [2685.0]
43 solver name: ['rhoCentralFoam']
44 iterations per execution time: [0.07116239868965774]
45 last modification date: ['2023-10-20 18:37:55']
46

```

Figure 22: Automatically Generated Report with All Relevant Metrics After Termination of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - ρ CentralFoam



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



(b) Convergence Plot for Residuals

Figure 21: Automatically Generated Convergence Plots After Termination of Simulation Ma 1.0 - AoA 0 deg - R5 Mesh - ρ CentralFoam