Wolfson School of Mechanical, Electrical and Manufacturing Engineering

Individual Project Final Report

ID Number: F131091

Programme: BEng Mechanical Engineering

Module Code: 24WSC500

Project Title: Modelling Re-entry Vehicles from Space

Abstract:

This project presents a comprehensive investigation into the aerodynamics and trajectory of atmospheric re-entry vehicles, utilising advanced CFD simulations in STAR-CCM+ and integrating it with a custom Python-based trajectory model. The CFD analysis looks to delve into critical flow phenomenon of an existing state of the art re-entry vehicle - the Falcon 9 – demonstrating the k-ω SST turbulence model's ability to predict drag coefficients and resolve critical flow features across transonic to hypersonic regimes. In light of the analysis, a targeted redesign of the grid fins—incorporating larger cells, contoured leading edges, and vortex generators—yielded measurable reductions in transonic drag and improved flow uniformity. Proceeding this, a Python trajectory model, employing a fourth-order Runge-Kutta integrator, translated these aerodynamic refinements into detailed flight-path predictions, revealing that while local aerodynamic improvements modestly increased range and impact velocity, the overall ballistic performance remained largely governed by integrated aerodynamic loading. The study highlights the strengths and limitations of both CFD and analytical approaches, advocating for a hybrid methodology and outlining future work in real-time CFD coupling, expanded dynamic modelling, and experimental validation to further enhance re-entry vehicle design and analysis.

Loughborough University

1 Contents

2 Introduction

In recent years, the landscape of space exploration has experienced rapid transformation, propelled by innovations in reusable rocket technology. Reusable rockets, by virtue of their ability to be recover and relaunched, have caused a paradigm shift by reducing the barrier to entry to space exploration. By fundamentally altering the economics of space exploration, the stage has been set for new era of more frequent, sustainable and ambitious missions.

At the core of this transformation lies the long existing challenge of re-entry, a phase where vehicles transition from the near vacuum of space into the increasingly dense layers of Earth's atmosphere. During re-entry, rockets encounter a complex interplay of aerodynamic forces arising from rapidly changing altitudes, air densities, and high velocities, with re-entry velocities of up Mach 36 being achieved [1]. These conditions generate significant perturbations that can influence stability, structural integrity, and overall vehicle performance. Accurately modelling these phenomena is essential, as it provides critical insights that can be leveraged to optimise design, enhance safety, and ensure mission success.

This project undertakes a comprehensive investigation into the aerodynamic behaviour of re-entry vehicles using advanced computational fluid dynamics (CFD) simulations in STAR-CCM+. By modelling simplified yet representative geometries, and rigorously validating the capability of these simulations against experimental data, this study aims to elucidate the key aerodynamic characteristics that govern the re-entry process. The research seeks to identify and quantify the effects of aerodynamic forces under varying conditions, thereby proposing modifications to streamline the re-entry trajectory and mitigate undesirable aerodynamic phenomena.

Complementing the CFD analysis, a trajectory model will be developed in Python, employing the Runge-Kutta method to iteratively compute the vehicle's flight path during re-entry. This approach will facilitate a detailed understanding of how aerodynamic forces influence the trajectory, enabling precise interpolation of the vehicle's behaviour across different flight phases.

3 Literature Review

3.1 Atmospheric re-entry analysis

The analysis of atmospheric re-entry dynamics necessitates a multifaceted approach, with computational simulation emerging as a cornerstone for understanding vehicle behaviour. Given that experimental testing at the extreme conditions encountered during re-entry is often infeasible due to the difficulty and high cost of reproducing the required high enthalpy and low-density flight regimes in wind tunnels or shock tunnels [2][3], Computational Fluid Dynamics (CFD) has proven invaluable for detailed flow field analysis and the prediction of critical parameters such as aerodynamic coefficients, heat flux, and pressure distributions.

In an application-focused study, Shafeeque employs CFD to analyse the external flow around the Apollo AS-202 capsule, providing insights into the flow field and pressure distribution experienced during atmospheric entry at different points on the capsule, which are crucial for the systems design. However, the selection and validation of appropriate CFD models remain critical challenges, as highlighted by Reddy and Sinha's comprehensive investigation of hypersonic turbulent flow around the FIRE II re-entry vehicle [4]. Their comparative analysis of the K-omega (K-ω) and Spalart-Allmaras (S-A) turbulence models reveals significant discrepancies in predicting afterbody flow features and emphasises the need for careful model selection based on the specific flow characteristics and validation against experimental or flight data.

Adding to these detailed CFD investigations, Tillier [5] presents a system-level simulation framework utilising MATLAB and SIMULINK for the SHARP vehicle. This approach integrates a trajectory simulator based on empirical aerodynamic data from NASA and a 6-DOF simulator employing a Newtonian aerodynamic model. Tillier's framework provides a holistic view of re-entry dynamics by capturing the interplay between trajectory, vehicle dynamics, and aerodynamic forces. Its strength lies in system integration and computational efficiency; however, the trade-off is reduced fidelity in resolving complex aerodynamic effects, the use of Newtonian theory approximates surface pressure distributions, ignoring viscous effects like boundary layer transitions and shock-boundary layer interactions. Ultimately, a synergistic combination of detailed CFD analyses and comprehensive system-level simulations, validated against available experimental or flight data, represents the most effective strategy for advancing our understanding and design capabilities for atmospheric re-entry vehicles, whilst keeping computational cost low.

3.2 Super-sonic flow past a sphere

Accurately obtaining drag coefficient data that aligns closely with established peer-reviewed results is essential for validating both the CFD software utilised, and the simulation methodology applied. Achieving this establishes confidence in the subsequent CFD analyses presented in this report, thereby enabling a reliable investigation into the drag characteristics of re-entry vehicles in transonic and supersonic flow regimes. Thus, an idealised spherical geometry simulated under free-stream conditions provides a benchmark case with extensive theoretical and experimental documentation available. Krasil'shchikov and Podobin [6] present both their own experimental results and data compiled from three independent studies [7][8][9]. Their findings were obtained through aero ballistic tunnel testing of a spherical body across a Mach number range of 1.5 to 15.2 and Reynolds numbers between 1×10^6 and 20×10^6. Extended analysis at higher speeds was conducted in this study as they noted that the majority of earlier experimental investigations primarily focused on sphere characteristics at Mach numbers up to 5, highlighting a need to increase this range. Consequently, their report includes a comprehensive compilation of results from various external sources, each employing consistent airflow conditions and identical projectile geometries. Figure 1 [Appendix 1] illustrates the drag coefficient as a function of Mach number, incorporating data from Krasil'shchikov and Podobin as well as the three independent studies, all of which were conducted using aeroballistics tunnel methodologies. A tabulated format of the results can be seen in Table 1 [Appendix 1].

Most recently Kharchenko and Kotov [10] conducted a validation study against the experimental data found in Krasil'shchikov and Podobin's findings, involving numerical simulations that modelled high speed gas flows over a spherical projectile. The UST3D HLLE code-calculated drag coefficient data was validated against experimental data produced during the study by Krasil'shchikov and Podobin as shown by Figure 2 [Appendix 1] declaring that current available research on obtaining the drag coefficient of super-sonic flow past a sphere has equitable quality.

3.3 Altitude effects on Re-entry Vehicles

A critical synthesis of the literature reveals significant methodological contrasts and connections in modelling altitude-dependent phenomena for re-entry vehicles. Bilbey Jr. challenges the conventional re-entry boundary at 120 km, arguing that "atmospheric effects are essentially negligible until the vehicle reaches an altitude near 80 km" [11]. This exposes a methodological gap: while the technical definition persists, practical aerodynamic significance—and thus modelling fidelity—shifts to lower altitudes. Bilbey's analysis compels a reallocation of computational resources, suggesting that detailed modelling is only necessary below 80 km, prompting a reconsideration of where to focus simulation efforts.

Stapper et al. [12] complements this by analysing the "re-entry corridor," defined by altitude and Mach number, demonstrating that higher ballistic coefficients shift trajectories toward the lower corridor boundary. Shown in Figure 3 [Appendix 2], their comparative analysis links vehicle mass properties and atmospheric density to trajectory predictability. However, their reliance on standardised atmospheric models limits their ability to capture real-time, altitude-specific variations, especially in the context of Bilbey's critique of arbitrary boundaries.

A critical investigation of aerodynamic performance in the "high-altitude hypersonic re-entry regime" was undertaken by Ahmad et al. revealing how gas property characterisation significantly affects prediction accuracy [13]. Their work demonstrates that "uncertainty in prediction of the aerodynamic loads and thermal transfer" is "exacerbated in R-SSTO vehicles because of the trend towards inherent complexity within their configuration." These finding challenges simplified approaches that fail to account for altitude-dependent gas property variations and shock interactions, particularly at critical points where heat transfer is most severe. Their methodological approach, integrating computational fluid dynamics with direct simulation Monte Carlo methods, represents the most comprehensive framework for analysing altitude-dependent phenomena during re-entry.

3.4 Simulation of hypersonic flow fields using STAR-CCM+

Cross and West [14] validated the ability of the STAR-CCM+ flow solver to predict hypersonic flows with acceptable accuracy and established best practices for its use in the NAWCWD report. The research included utilising several hypersonic flow field test cases with experimental data, including the double cone, small and large hollow cylinder-flare configurations, and a shock/shock interaction test case. These cases involved challenging phenomena like shock/boundary layer interactions and flow separation, crucial for hypersonic flight vehicle design.

The consideration of thermochemical non-equilibrium is essential in hypersonic flows, necessitating real gas models like the five-species air model (N_2, O_2, NO, N, and O) used in the report. STAR-CCM+ includes Real Gas air property models, with the model being used to assess real-gas effects as well as "standard" air models based on traceable sources to improve accuracy.

Tailored mesh adaptation techniques are crucial for resolving shock waves and boundary layers. The report develops and implements normalised pressure gradient mesh refinement to handle multiple shocks of different strengths and prevent excessive prism layer refinement. This technique was shown to be superior to the initial pressure gradient method and effective for 3D problems as well. Pekurovsky et al. [15] also emphasised the importance of tailored mesh adaptation for hypersonic flows. The report also recommended specific solver settings for stable and fast convergence, including a MUSCL third order/central-differencing scheme and AUSM+ flux-vector splitting. For turbulence modelling with the Menter SST model, specific non-default settings were advised. Comparisons showed that STAR-CCM+ achieved accuracy comparable to specialised solvers like LAURA and DPLR, offering a user-friendly interface and integrated meshing.

Various RANS turbulence models were investigated, with the choice significantly impacting the prediction of flow separation and heat transfer. Carias and Prince [16] also utilised Menter-SST and Spalart-Allmaras models for simulating flow over cones at hypersonic speeds, finding Menter SST superior for inclined flows.

Despite validation, the report acknowledged potential limitations of STAR-CCM+ for extreme re-entry conditions and challenges with inconsistent experimental documentation. Overall, the NAWCWD report concluded that STAR-CCM+ is a capable tool for simulating hypersonic flow fields when using appropriate models and techniques.

4 Background Research

4.1 Types of re-entry vehicles

4.1.1 Controlled re-entry

Controlled re-entry vehicles represent a highly engineered solution where risk mitigation is paramount. These systems incorporate aerodynamic surfaces, advanced guidance systems, and robust thermal protection to actively steer the vehicle along a pre-designed trajectory. Critically, this methodology offers significant advantages: it ensures high levels of safety for crewed missions and preserves high-value payloads by reducing the risk of uncontrolled descent overpopulated areas. Figure 4 [Appendix 3] shows the NASA's Space Shuttle, a partially reusable spacecraft, exemplifying the aerodynamic characteristics such as its 40° delta wing design and reinforced carbon-carbon nose cap [17]. However, the complexity inherent in these systems—demanding precise control, real-time computational adjustments, and redundant safety mechanisms—introduces both significant financial costs and increased technical challenges. The need for extensive testing and validation of aerodynamic control, combined with the engineering of thermal protection systems capable of withstanding extreme re-entry temperatures, raises questions about overall cost-effectiveness in scenarios where mission requirements may not justify such sophistication. Ultimately, the controlled approach's effectiveness is tightly coupled with mission objectives, particularly when human life and expensive payloads are involved.

4.1.2 Uncontrolled re-entry

Uncontrolled re-entry methods—exemplified by spent rocket upper stages—rely on a passive ballistic descent governed by gravity and atmospheric drag. While ostensibly less sophisticated, this approach offers clear advantages in terms of cost and simplicity. By eliminating the need for complex guidance systems, the engineering burden is significantly reduced. Yet, this simplicity comes with the unpredictability of the descent path posing a challenge, as precise targeting is unattainable. While most debris from uncontrolled re-entry disintegrates due to intense heat, between 20% and 40% of a rocket body's mass survives re-entry, with fragments ranging from small debris to refrigerator-sized fuel tanks [18]. This issue becomes particularly relevant in densely populated regions or sensitive environments, making uncontrolled re-entry a less attractive option when collateral risk must be minimised.

4.2 Re-entry Flight Dynamics

During descent, vehicles transition from orbital velocities in near-vacuum conditions to subsonic speeds in dense atmosphere, experiencing dramatic changes in aerodynamic forces, temperatures, and pressures.

4.2.1 Hypersonic Entry Phase Dynamics

The initial phase of atmospheric re-entry occurs at hypersonic speeds, typically between Mach 20-25 (approximately 17,500 mph) for vehicles returning from low Earth orbit [19]. At these velocities, the aerodynamic forces are extreme, and the flow physics are fundamentally different from conventional flight regimes.

The vehicle's extreme velocity creates such intense compression of air molecules that chemical bonds break, generating an electrically charged plasma around the vehicle. The air density at this altitude (typically above 120 km) is extremely low, but the velocity is so high that significant aerodynamic heating begins to occur. Strong shock waves form on the vehicle's lower surface, creating a high pressure-region serving as the primary drag mechanism that begins to decelerate the spacecraft [19].

The aerodynamic coefficients at hypersonic speeds show distinct exacerbated behaviour compared to conventional flight regimes. Research on re-entry vehicles at Mach 6 demonstrates that geometric features like flare-cone angle significantly influence aerodynamic performance. For example, increasing the cone-flare angle from 25° to 35° can result in increases of normal force slope, axial forebody drag, and base drag by 62.5%, 56.2%, and 33.13% respectively [20]

4.2.2 Aerodynamic Coefficients and Vehicle Performance

The ballistic coefficient is a critical parameter that influences re-entry dynamics defined as (1):

$$BD = \frac{m}{Cd*A} \tag{1}$$

Where 'm' is vehicle mass, 'Cd' is drag coefficient, and 'A' is reference area.

This parameter indicates vehicle performance during re-entry; vehicles with higher ballistic coefficients experience a greater peak heat flux, higher peak dynamic pressures, and slower deceleration conditions that are not optimal for re-entry. This has been validated by the ReFEx (The Reusability Flight Experiment) which has demonstrated ballistic coefficients around 7900 kg/m² (at 0° angle of attack) compared to approximately 2900 kg/m² for larger reusable vehicles at similar conditions. This higher ballistic coefficient leads to more intense aerodynamic loading during descent [12].

Dynamic pressure profiles during re-entry show characteristic patterns that significantly affect vehicle design and control. For the ReFEx, maximum dynamic pressure typically ranges between 37-40 kPa regardless of re-entry mass. This occurs because heavier vehicles with higher ballistic coefficients require stronger aerodynamic forces but enter with lower velocities. The dynamic pressure gradient - the rate at which pressure builds up during descent - is particularly critical. Steeper re-entry flight path angles produce more rapid pressure increases, creating more severe structural and thermal loads.

4.2.3 Angle of attack and Trajectory Angle

The angle of attack (AoA) and trajectory angle (flight-path angle) critically influence the dynamics and survivability of re-entry vehicles. Increasing AoA generally raises aerodynamic drag, leading to longer flight times and reduced approach speeds, but increases lateral loads, which can challenge vehicle stability and control [21]. Steep trajectory angles (larger flight-path angles) cause the vehicle to penetrate deeper into the atmosphere, resulting in higher peak deceleration and heating rates, but for shorter durations. Conversely, shallow entries expose the vehicle to lower but prolonged heating and deceleration, increasing the risk of cumulative thermal damage and reducing landing accuracy. Optimal re-entry design thus requires balancing AoA and trajectory angle to manage aerodynamic loads, heating, and guidance precision, ensuring the vehicle remains within the narrow "re-entry corridor" that avoids both atmospheric skip-out and destructive overheating.

4.3 Simcenter Star CCM+

For compressible, viscous flows at high speeds—as encountered during atmospheric re-entry—the following shorthand forms of the Navier–Stokes equations serve as the fundamental governing equations, encapsulating the conservation of mass (2)[22],

$$\frac{\vartheta \rho}{\vartheta t} + \nabla \cdot (\rho \boldsymbol{u}) = 0, \tag{2}$$

momentum (3),

$$\frac{\vartheta \rho u}{\vartheta t} + \nabla \cdot (\rho u u) = -\nabla p + \nabla \cdot \tau + \rho g, \tag{3}$$

and energy (4),

$$\frac{\vartheta(\rho E)}{\vartheta t} + \nabla \cdot [(\rho E + p)u] = \nabla \cdot (k\nabla T) + \Phi \tag{4}$$

Where ρ is density, t is time, $\boldsymbol{u}$ is the velocity vector, p is pressure, g is gravitational acceleration, τ is the viscous stress tensor, E is total energy per unit mass, k is thermal conductivity, T is temperature and Φ is the dissipation function.

CFD enables the analysis of a wide range of geometries under both idealised and realistic conditions, offering significant time and cost savings compared to experimental testing and allowing detailed study of flow phenomena in regions that are difficult to access experimentally.

The commercial CFD software supports both the import of CAD models and in-built geometry creation tools, making it especially well-suited for developing streamlined vehicle configurations, such as blunt-nosed capsules. A critical aspect of achieving accurate simulations is the generation of an appropriate volumetric mesh; STAR-CCM+ provides both standard mesh functions and custom controls—such as localised volumetric refinement—that enable focused resolution in key areas with steep gradients, like shock waves and boundary layers as discussed in section 3.4.

In these simulations, the Reynolds-Averaged Navier–Stokes equations are solved using various turbulence models. Comparisons among the Spalart–Allmaras, k-ε, k-ω SST, and Reynolds Stress Transport models are performed to determine which model best captures the complex interactions, such as shock/boundary layer interactions and flow separation, that occur during the re-entry process. This methodological approach is vital for optimising the aerodynamic performance and thermal protection of re-entry vehicles.

5 Methodology

5.1 Flow Past a Sphere

The objective of this study is to validate the accuracy and usability of Simcenter Star-CCM+ for predicting the flow around projectile geometries. Validation is performed using an idealised spherical geometry, with simulations conducted for transonic to hypersonic Mach numbers. The Mach number will be progressively increased until either the computational cost exceeds the available resources, or the simulation results deviate by more than 5% from the experimental data. Subsonic flow regimes have been excluded in this study, following Van Dyke's observation [23] that non-axisymmetric flow patterns may occur at M ≤ 0.85, making symmetry plane application invalid in such cases.

5.1.1 Flow and geometry conditions

Experimental reference data from Krasil'shchikov and Podobin [5] involved Reynolds numbers in the range of 10^6–10^7. Based on this, the following physical properties were assumed to define the test conditions:

Reynolds Number Re	**Fluid Density (kg/m^3)** $\boldsymbol{\rho}$	**Velocity (m/s)** $\boldsymbol{u}$	**Dynamic Viscosity (Pa/s)** $\boldsymbol{\mu}$
$1.2\text{x}10^6$	1.225	680.58	$1.789 \text{ x } 10^{-5}$

Table 2 - Reynolds Number of a Supersonic Sphere

From these characteristics the Reynolds Number formula (5) can be rearranged (6) to find the diameter:

$$Re = \frac{\rho u D}{\mu} \tag{5}$$

$$D = \frac{Re\mu}{\rho u} \tag{6}$$

In doing so, the diameter (D) was found to be 0.0257m.

5.1.2 Computational Domain and Boundary Definition

For accurate simulation of free-stream flow conditions, especially in supersonic regimes, the spherical projectile's origin was positioned at [4D,0,0] where D is the diameter of the sphere. To reduce computational load and take advantage of symmetry, only one-half of the sphere was modelled, assuming both the projectile and the flow patterns are axisymmetric.

The simulation domain was created using Boolean subtraction operations. A half-sphere with a radius of 12D, centred at [0,0,0], served as the free-stream enclosure and was subtracted from this enclosure to generate the computational flow domain. This sizing was optimised through testing at various Mach numbers to prevent interference from reflected flow waves at the enclosure boundary visible in the scalar scenes and the failure of the simulations to converge. It was later found at hypersonic speeds the domain needed to be expanded to a radius of 18D, cantered at [0,0,0]

Once the computational part was created, the domain was assigned to a region using the 'Assign Parts to Regions' menu. The following options were selected to ensure accurate boundary and mesh definition: 'Create One Region for All Parts,' 'Create a Boundary for Each Part Surface,' and 'Create a Feature Curve for Each Part Surface'.

The two boundary faces of the region were then defined. The projectile surface was assigned a "non-slip wall" condition to allow for the formation of a boundary layer by setting velocity components at the wall to zero. The enclosure surface was set as a "free stream" boundary, while the remaining flat face of the half-sphere domain were designated as a symmetry plane, in line with axisymmetric flow assumptions.

5.1.3 Simulation Setup and Data Collection

A comparative analysis was conducted to evaluate the performance of different RANS turbulence models; Spalart-Allmaras (S-A), K-Omega (K-ω), and K-Epsilon (K-ε) models. The most accurate and computationally efficient model, based on this comparison and supported by literature, was then employed for subsequent research. Simulations progressively increased Mach number using the physics models listed in Table 3 [Appendix 4], an example of the initial conditions used for one of the simulations – Mach 3, can be found in Table 4 [Appendix 4].

All simulations assumed sea-level conditions for the projectile (altitude = 0 m). The initial conditions and fluid properties used—sourced from the International Standard Atmosphere Tables [24].

A force coefficient report was set up to compute the drag coefficient of each simulated projectile. This report also served as a qualitative indicator for achieving steady-state conditions during the simulation. Additionally, a monitor and corresponding plot were created to track these parameters.

5.1.4 Mesh Generation and Control

A volume mesh for the 3D flow domain was generated using the Automated Mesh operation. For all spherical simulations, the following mesh generators were enabled: Tetrahedral Mesher, Prism Layer Mesher, Surface Remesher, and Automated Surface Repair.

Tetrahedral elements were employed due to their ability to conform to complex surface curvatures, such as those of a spherical geometry. Prism Layer Mesher generate orthogonal prismatic cells adjacent to the wall boundaries, capturing near-wall flow phenomena. These prism layers are essential for resolving steep velocity and pressure gradients in the boundary layer, thereby enhancing the fidelity of the near-wall solution. Additionally, the Surface Remesher was applied to iteratively refine the surface mesh, improving element quality and ensuring a well-conditioned discretisation that meets the resolution and quality criteria required for stable and accurate CFD simulations.

Adaptive mesh control (AMR) and volumetric control operations were used in tandem to refine the mesh. AMR is a user defined function that refines the mesh dynamically over regions with high flow gradients, such as shock waves and boundary layers, whilst coarsening the mesh in regions of uniform flow to the criterion set by the user. This approach ensured improved local accuracy without significantly increasing the overall computational cost but specifically focusing on areas of interest.

In this case AMR was deployed to monitor gradients of Mach number and was achieved by using the following FORTRAN code:

```
mag(grad(${MachNumber}))*(${AdaptionCellSize})
```

Surface controls were applied to refine the mesh near the projectile surface, ensuring increased cell density close to the body and lower density further away. Volumetric controls were implemented to manually refine the mesh within specific regions of interest, such as near the projectile surface and wake region. These controls ensured a higher cell density in critical flow areas, enhancing resolution and accuracy in a more holistic manner than AMR while also maintaining computational efficiency elsewhere in the domain. Both mesh controls are shown in Figure 5.

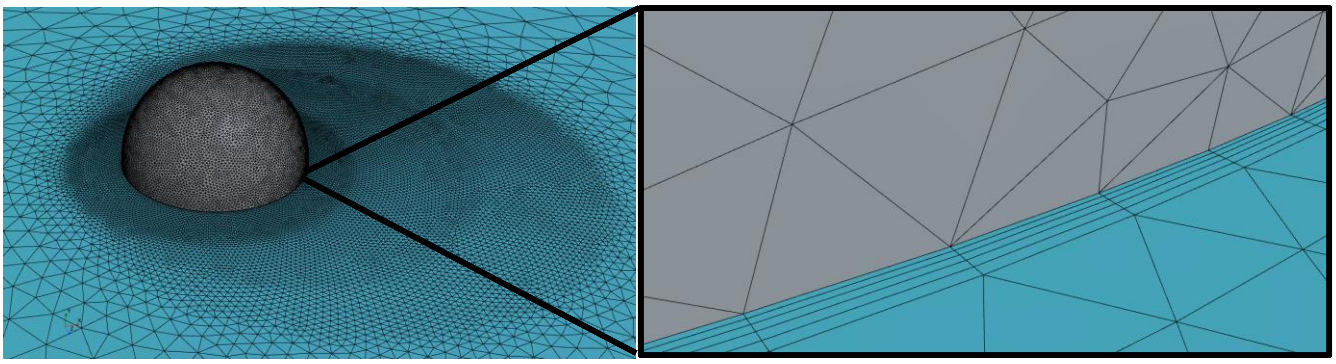


Figure 5 - Effects of volumetric and surface mesh

For Mach 1 and 2, one Volumetric control operation was sufficient. However, due to the complexity of the flow at Mach 3 and beyond, three volumetric controls were implemented to achieve a graduated mesh. In these cases, the smallest control size was given precedence to preserve mesh quality near critical regions. Full mesh control values for Mach 3 and beyond are provided in Table 5 [Appendix 10], Mach 1 and 2 were run without Volumetric control 2 and 3.

5.1.5 Results

Results indicated alignment with published literature up to Mach 7. Table 6 summarises the findings from the validation study, presenting also the absolute and percentage differences between the experimental and simulated values. When all three turbulence models exceeded 5% difference with the experimental data no more simulations were ran.

Mach Number	Turbulence Model	C_d (CFD)	C_d (Experimental)	Absolute difference	Percentage difference [%]
1	Spalart-Allmaras	0.984	1.00	0.016	1.6
	K-Epsilon	0.912	1.00	0.088	8.8
	K-Omega	0.978	1.00	0.022	2.2
2	Spalart-Allmaras	1.027	0.98	0.047	4.7
	K-Epsilon	1.068	0.98	0.088	8.8
	K-Omega	1.022	0.98	0.042	4.2
5	Spalart-Allmaras	0.890	0.93	0.040	4.0
	K-Epsilon	0.979	0.93	0.049	4.9
	K-Omega	0.897	0.93	0.033	3.3
7	Spalart-Allmaras	0.824	0.89	0.066	6.6
	K-Epsilon	0.957	0.89	0.067	6.7

	K-Omega	0.873	0.89	0.017	1.7
8	Spalart-Allmaras	0.799	0.87	0.071	7.1
	K-Epsilon	0.950	0.87	0.080	8.0
	K-Omega	0.814	0.87	0.056	5.6

Table 6 - Comparison and Validation of Turbulence Models

In comparison to Kharchenko and Kotov's collective study, across every Mach number, both Spalart–Allmaras and k–ω outperform k–ε, with k–ω delivering the smallest average error (≈ 2–4 %). The k–ε model predominantly overshoots the experimental drag, particularly in the transonic window (8.8 % at Mach 1) and again at higher Mach (8.8 % at Mach 2, 8.0 % at Mach 8). k–ω's stability and lower average deviation recommend it as our turbulence model of choice for subsequent design studies. Inferring from the results, further investigations in to simulating re-entry vehicle conditions will stop at Mach 7 and k–ω will be the turbulence model of choice.

The simulation results align well with expectations from the literature. In all cases, a pronounced pressure spike and a corresponding velocity drop are observed ahead of the spherical projectile. A detached shock wave forms in front of the sphere, while a stagnation region develops along its leading face, where the velocity drops to a near zero value for both Mach 2 and Mach 7 scalar plots. In addition, expansion fans form toward the rear of the sphere because of the decreasing incidence angle along the leading edge. Following boundary separation, a turbulent wake develops at the rear, clearly illustrating the complex flow interactions typical of such geometries.

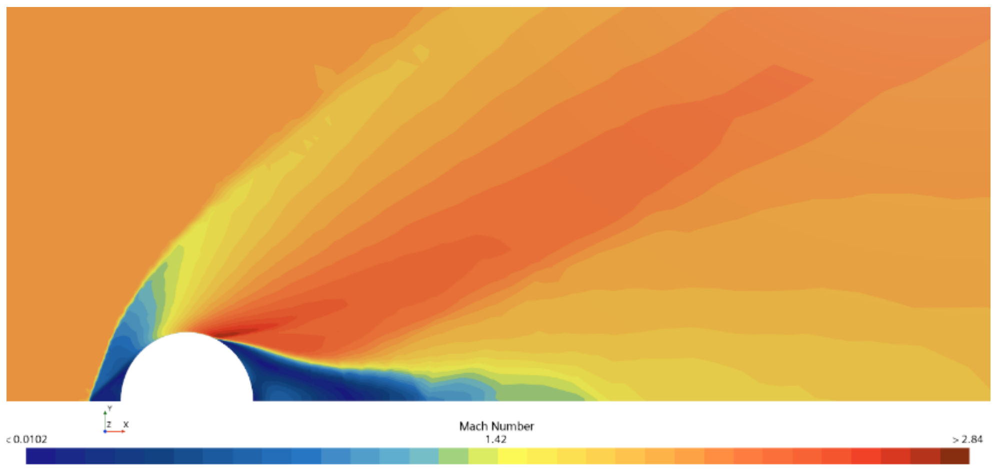


Figure 6 - Velocity Scalar Plot of a Sphere at Mach 2

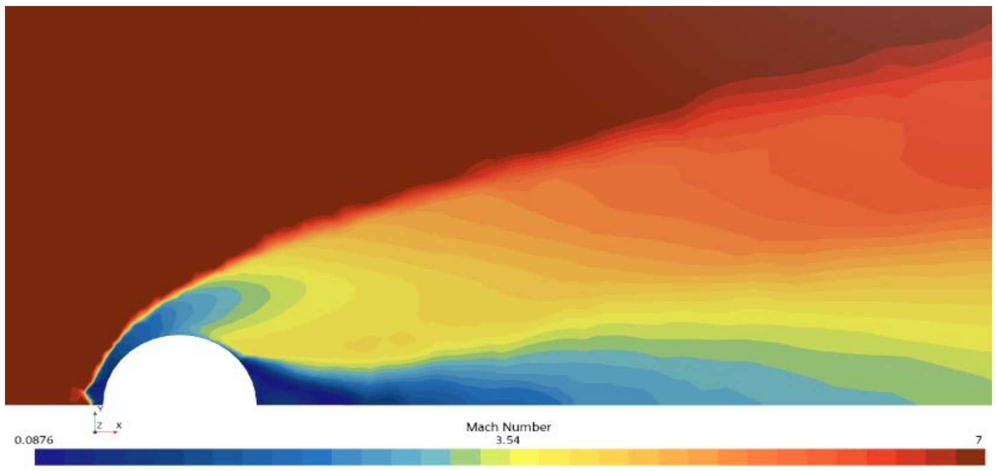


Figure 7 - Velocity Scalar Plot of a Sphere at Mach 7

The Mach 2 contour in Figure 6 reveals a well-resolved detached bow shock standing roughly 0.28 D ahead of the sphere's nose and a supersonic "pocket" with local Mach numbers up to 1.42. The sharp gradient between this high-speed core and the downstream wake indicates a thin shear layer that has been captured best by the k–ω SST model. By Mach 7 (Figure 7), the bow shock has tightened to within 0.05 D of the nose, and the post-shock Mach field exceeds 2.8 immediately adjacent to the surface. This dramatic standoff collapse underscores the need for extremely fine mesh at the nose; any coarseness would artificially blunt the shock and underpredict stagnation pressure. The near-wake also narrows sharply to about 0.1 D, reducing suction drag but signalling the onset of rarefaction where continuum RANS may lose reliability. Here, k–ω again best preserves the physical wake collapse and shock strength.

The mesh adapts dynamically to capture steep gradients, particularly around the detached shock and other discontinuities. The simulation shows a gradual refinement in regions of high gradient and a systematic coarsening in areas with mild variations in Mach number. This transition not only provides a detailed resolution of the shock wave but also optimises computational resources by reducing cell density where fine details are not required. Comparisons of mesh visualisations in Figure 8 reveals that the refinement accurately delineates regions of interest, affirming the effectiveness of the AMR technique implemented within the simulation framework.

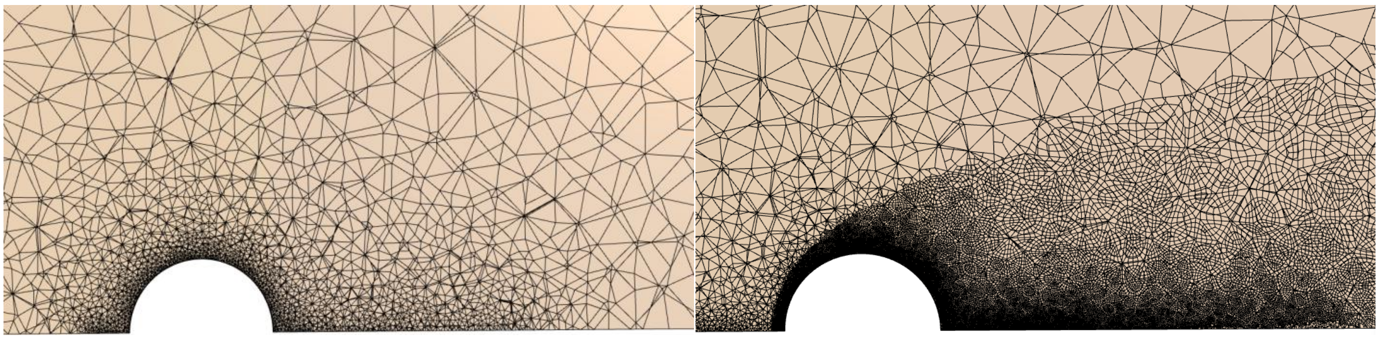


Figure 8 - Effect of Adaptive Mesh Refinement

In summary, the numerical results from the spherical projectile simulations demonstrate excellent agreement with established experimental data. The adaptive meshing strategy, meshing controls and the correct choice of turbulence models—particularly the k-ω model—have enabled accurate predictions of key flow features such as shock formation, expansion fans, and turbulent wakes. The observed pressure and velocity distributions, combined with the close match in predicted drag coefficients, underscore the promising capability of the simulation tool in capturing transonic, supersonic and hypersonic flow phenomena.

5.2 Re-entry Vehicle modelling and analysis

5.2.1 Falcon 9 Vehicle Geometry creation

An accurate geometric model forms the foundation for high-fidelity aerodynamic simulations, therefore in this work, the geometry of an existing re-entry vehicle was developed in NX CAD. The Falcon 9 offers the perfect balance between innovation and proven technology, being the first reusable rocket booster with a rich flight heritage of 457 completed missions and 384 total reflights. The resulting research would contribute not only to academic understanding but potentially to the future development of reusable spacecraft for Earth return and planetary exploration missions. Therefore, the following geometry would be inspired from the dimensions of the Falcon 9 booster rocket developed by SpaceX.

Due to the complex design features found on the Falcon 9, such as the Grid Fins, it was critical to capture the aerodynamic characteristics. Siemens NX was selected due to its ability to it high performance modelling capabilities and its seamless integration into STAR-CCM+ since both software packages are developed by Siemens.

To reduce complexity and reduce computational cost when conducting analysis of the projectile in STAR-CCM+ only the three major ‘components’ of the Falcon 9 were detailed in the final geometry and since only the re-entry process will be modelled the fairing has been excluded as this is typically ejected as a payload into space. Engineering drawings for the following components can be found in appendix 4.

Cylindrical Body:
The fuselage of the rocket has a length of 39.7m (excluding the fairing) and a diameter of 3.66m. The body itself was modelled as a simple, smooth cylinder to represent the basic aerodynamic shape of the rocket.

Grid Fins:
The grid fins during re-entry are in their open (deployed) configuration – they serve to orient the rocket during re-entry by moving the centre of pressure through manipulating airflow using yaw, pitch and roll. Their dimensions were selected to be representative of the full-scale grid fins span approximately 1226 mm in width, 2083mm in height and with grid cells of 0.012996m^2. The open configuration of the grid fins is critical to the analysis, as it allows for the evaluation of their aerodynamic performance and is critical to maintaining the stability of the rocket during re-entry.

Landing Legs:
The landing legs were modelled only in the closed (stowed) configuration, since analysis would only be conducted from transonic to hypersonic flow. In this configuration, the legs were integrated flush with the lower section of the cylindrical body. By keeping the design simple—without modelling the detailed mechanical articulation—the focus remains on assessing any aerodynamic interference or drag introduced by the landing legs in their stowed position. The dimensions were estimated on full-scale specifications with a total length of 9760mm.

Since the Falcon 9 dimensions have not been publicly disclosed the dimensions stated have been inferred and inspired from what is available.

The ‘Grid Fin’ and ‘Landing Leg’ were joined together with the main ‘Cylinder Body’ using the *‘Unite’* function in NX. Now that the rocket is one ‘body’ consisting of three components, the body can be scaled to manipulate the characteristic length before importing it into STAR-CCM. To maintain consistency with the previous validation case of the sphere, which was proven to be an effective simulation, the geometry was proportionally scaled so that the rocket’s total length was reduced to 25.7 mm – the same as the sphere. Also, since the rocket is axisymmetric, only one-quarter of the rocket was required to be modelled in Star-CCM+, therefore using the *‘trim body’* function using the united body as the *‘target’* and setting a ‘*Datum plane’* as the tool, the model was split into a quarter as illustrated in Figure 9.

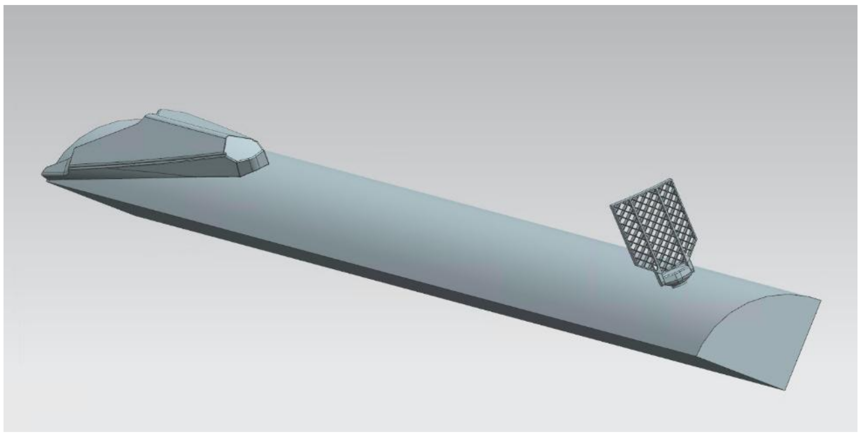


Figure 9 – Geometry of Falcon 9 Inspired Rocket

Following the completion of the re-entry vehicle geometry, a rigorous quality assurance process was undertaken to confirm dimensional accuracy and surface continuity. The model was scrutinised for potential issues such as overlaps, gaps, or non-manifold edges using NX CAD's internal diagnostic tools. Finally, the geometry was exported using the *‘Heal Geometry’* format as the last step of cleaning up and healing the geometry which ensures it compatibility with STAR-CCM+. Prior to the import, minor adjustments were made to simplify certain surface features, which facilitated the subsequent meshing process without compromising the aerodynamic fidelity of the model.

5.2.2 Mesh and Simulation Setup

The geometry was imported in as a surface mesh in Star CCM+, this is done by going to *'File -> Import -> Surface Mesh'.* As outlined previously in section 4.1.2, the same steps were taken with a few refinements. Tetrahedral elements were swapped out for Polyhedral ones, this offers better mesh quality for complex geometries by increasing total cell 'neighbours' meaning gradient approximation accuracy would improve. The projectile was translated to [6D,0,0] and was subtracted using the *'Boolean'* operation, this time from a quarter sphere with a diameter of 18D positioned [0,0,0] to ensure no backflow was present at higher Mach numbers.

In addition to the volumetric controls highlighted in section 4.1.2, two more controls were added. A cuboid was positioned in the wake of the projectile with a length of 6 times that of the rocket (6D). The grid fins were contained in a cube and refined further to ensure that there were enough meshing nodes in between the grid cells, allowing airflow going through the fins to be modelled to a high resolution.

Accurately resolving the boundary layer is critical for high-fidelity CFD simulations, as it directly influences the prediction of aerodynamic phenomena. A key metric in this regard is the non-dimensional y^+ value, which determines how well the near-wall mesh captures viscous effects. To streamline mesh optimisation, a MATLAB script was developed to automatically compute the required first cell height for a given target y^+ and the thickness of the boundary layer for this specific model. This tool facilitates rapid evaluation under various flow conditions, thereby ensuring that meshing strategies meet the stringent resolution requirements needed for accurate turbulence modelling.

The MATLAB code prompts the user to enter the following essential fluid and flow properties: Fluid density (ρ) in kg/m^3, Dynamic viscosity (μ) in kg/(m·s), Free-stream velocity (v) in m/s and Reference length (l) in m.

Using these inputs, the Reynolds number (Re) is computed using equation (5) from section 4.1.1. This project will be dealing with external flow, therefore, Schlichting's correlation is applied [25]:

$$Cf = (2log_{10}(Re) - 0.65)^{-2.3} \tag{6}$$

The empirical correlations are well established in literature and provide reliable estimates of the wall friction coefficient for different flow regimes. From this the shear stress at the wall can be calculated using:

$$\tau = 0.5 \cdot Cf \cdot \rho \cdot v^2 \tag{7}$$

This in turn is used to calculate the friction velocity *(u_f):*

$$u_f = \sqrt{\frac{\tau}{\rho}} \tag{8}$$

The user can then apply a target y^+ value (the desired non-dimensional wall distance), which typically is ≤1 for k-ω models. The first cell height can be calculated using:

$$d_s = \frac{y^+ \cdot \mu}{u_f \cdot \rho} \tag{9}$$

To further guide mesh design, the turbulent boundary layer thickness (δ) is estimated using the empirical flat-plate correlation:

$$\delta = \frac{0.37L}{Re^{0.2}} \tag{10}$$

The prism layer configuration is optimised using a geometric progression. The total prism layer thickness *(S_{total})* is calculated to ensure adequate coverage of the boundary layer:

$$S_{Total} = y_1 \cdot \frac{G_T^{N_l} - 1}{G_T - 1} \tag{11}$$

where G_T is the growth ratio (default 1.2) and N_l is the number of layers (default 25). To maintain mesh quality, the script validates that $S_{Total} \geq \delta$ and $G_T \leq 1.3$, aligning with best practices for minimising cell distortion suggested by the STAR-CCM+ user guide. Warnings are issued if these thresholds are violated, ensuring adherence to turbulence modelling requirements. The full script and implementation steps can be found in Appendix 6, the prism layer settings and additional volumetric controls can be found in Table 7 and 8 [Appendix 4] respectively. The effect of the meshing can be observed in Figure 10 and 11.

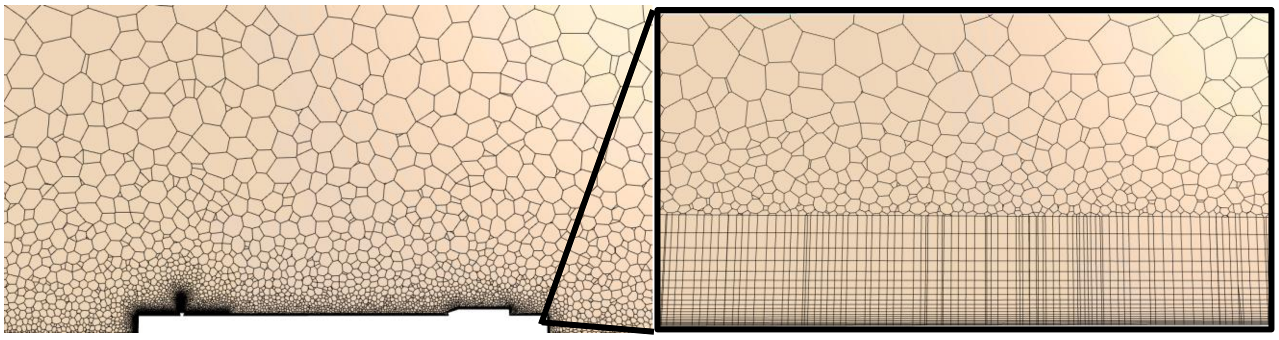


Figure 10 – Effects of y^+ boundary layer meshing

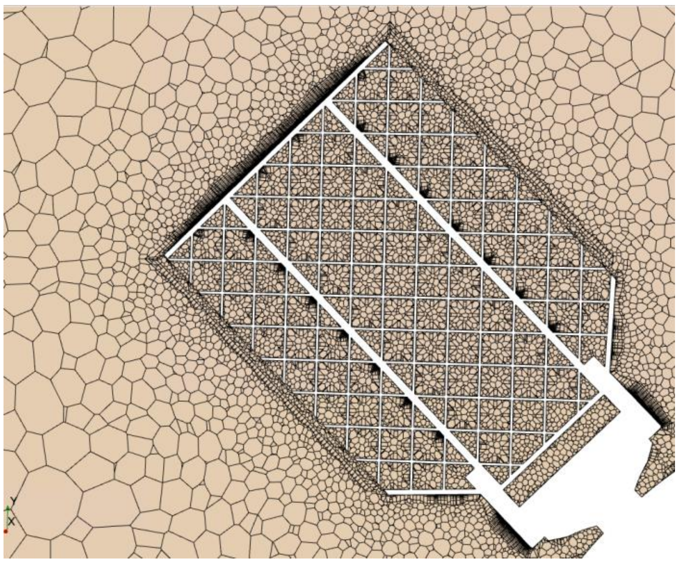


Figure 11 - Volumetric mesh control on Grid Fin

Solver settings were altered to accommodate for a more complex geometry being modelled at supersonic and hypersonic Mach numbers. These alterations can be found in Table 9 and 10 [Appendix 4].

Now that the mesh and simulation parameters have been set, a custom-made trajectory path curated from existing Falcon 9 re-entry profiles was made based on key milestones. These key points during re-entry will be simulated in Star-CCM, the findings later used in the trajectory code. Table 11 below indicates the points during re-entry that will be modelled.

Altitude (Km)	**Mach Number**	**Speed** (m/s)	**Angle of Attack** [AoA] (Degrees)
70	7	2382.03	20
55	6	2041.74	15
40	5	1701.45	12
25	3	1020.87	8
15	2	680.58	5
8	1	340.29	0

Table 11 - Custom Trajectory Path

Altitude effects on the flow were modelled by adjusting gas properties—atmospheric pressure, static temperature, dynamic viscosity, and density—according to the standard atmospheric tables [24] at each discrete altitude. Projectile angle of attack was imposed by defining a rotated coordinate system for each target AoA, such that the freestream velocity vector aligned with the specified incidence angle.

5.2.3 Results

Table 12 shows the coefficient of drag for the Falcon 9 inspired rocket at different Mach numbers and angles of attack, detailing a significant pattern between transonic, supersonic, and hypersonic regimes.

Mach Number	**Turbulence Model**	**Free Stream Velocity (m/s)**	**Angle of Attack [AoA] (Degrees)**	**Cd (CFD)**
7	K-Omega	2382.03	20	1.184229
6	K-Omega	2042.74	15	1.199547
5	K-Omega	1701.45	12	1.201345
3	K-Omega	1020.87	8	1.163445
2	K-Omega	680.58	5	1.121003
1	K-Omega	340.29	0	1.218851

Table 12 - Drag Coefficient results of Falcon 9 inspired Rocket

The CFD analysis at Mach 1 yields a drag coefficient of 1.218851, representing the peak aerodynamic drag from all simulations. The pressure contour plot, Figure 12, reveals extreme gradients with high-pressure regions exceeding 1×10^4 Pa (deep red) at the nose stagnation point, while extensive negative pressure zones approaching -1×10^4 Pa (dark blue) form in the wake regions. This pressure differential of over 2×10^4 Pa creates substantial pressure drag. The velocity magnitude visualisation displays a prominent detached bow shock of approximately 0.2D from the nose, with flow velocities abruptly transitioning from 414 m/s (maximum, orange-red) to near-zero (dark blue) at stagnation regions. Particularly notable are the expansion zones along the landing legs and grid fins where flow accelerates before encountering sudden deceleration. Significant boundary layer separation is evident at geometric transitions, forming extensive recirculation zones with velocities below 207 m/s. These quantifiable flow features directly correlate with the elevated drag coefficient and validate theoretical predictions of transonic drag rise.

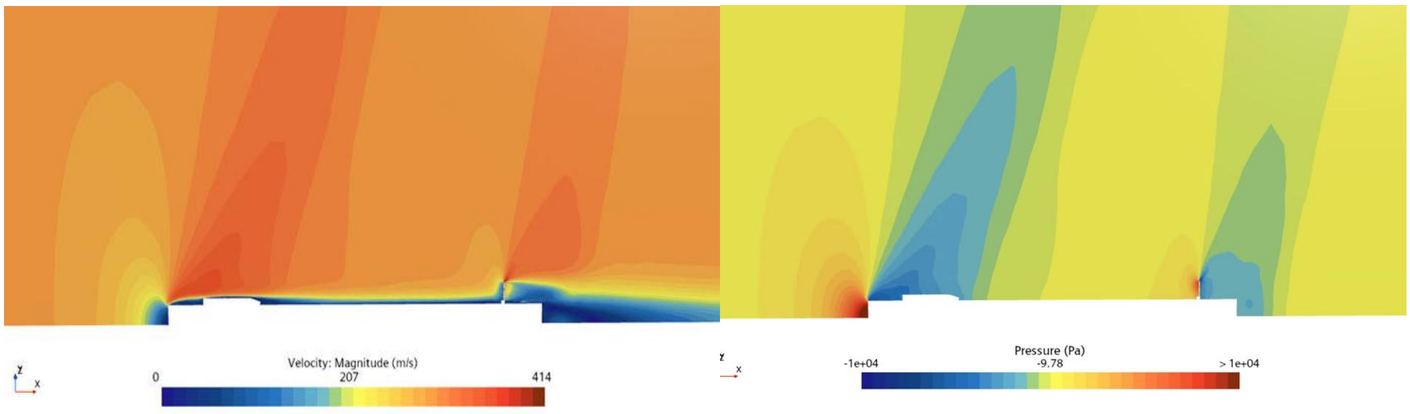


Figure 12 - Velocity (Left) and Pressure (right) Contour plots for flow around rocket at Mach 1

At supersonic flight, Mach 3 reveals a drag coefficient of 1.163445, demonstrating a reduction in aerodynamic drag during the supersonic regime. The velocity magnitude plot, Figure 13, exhibits a less distinctly attached oblique shock than at Mach 1 on top of the Grid Fin and a much shorter detached bow shock of 0.04D from the nose. The velocity field shows a well-defined shock structure with freestream velocities maintaining between 462-923 m/s across most of the domain. Notably, the expansion region at geometric transitions accelerates flow to maximum values (deep red, ~923 m/s), approximately 10% higher than nominal supersonic velocities. Flow separation is significantly reduced compared to transonic conditions, with only minor detachment at significant geometric changes. The pressure contours confirm this behaviour, displaying a concentrated stagnation region exceeding 2×10^4 Pa at the nose, with predominantly uniform pressure fields of 3.27×10^3 Pa along the body. Expansion regions generate localised pressure drops to approximately -1.35×10^4 Pa, creating a pressure differential of 3.35×10^4 Pa that contributes to the wave drag component while exhibiting considerably lower pressure-induced separation than at Mach 1.

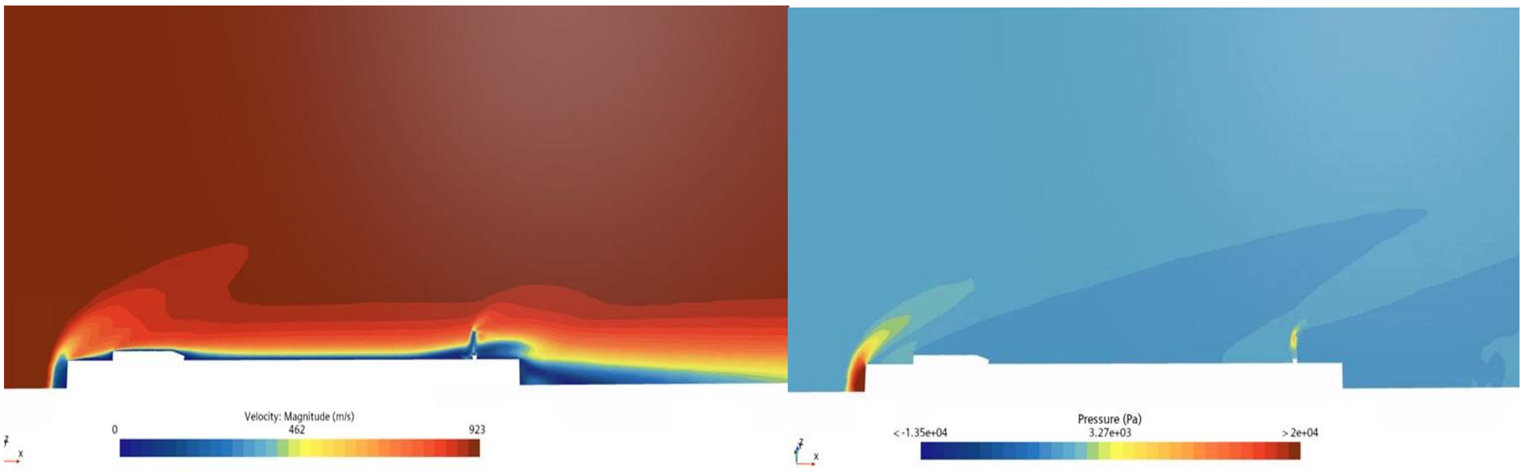


Figure 13 - Velocity (Left) and Pressure (right) Contour plots for flow around rocket at Mach 3

Drag coefficient begins to climb as we enter hypersonic flow, CFD analysis at Mach 6 reveals a drag coefficient of 1.199547. An extremely compressed shock layer is evident in Figure 14, approximately 0.02D from the body surface, with the shock-boundary layer interaction region notably thinner than at lower Mach numbers. Flow velocities transition abruptly from freestream values of 1980 m/s (dark red) to below 991 m/s (yellow-green) within this narrow region, generating extreme velocity gradients exceeding 9.5×10^4 m/s per meter. Despite these severe conditions, the boundary layer remains predominantly attached, with flow separation constrained to minimal regions at geometric discontinuities. The pressure contours, Figure 11, show intense localisation at the stagnation point exceeding 1.5×10^4 Pa, followed by rapid expansion to approximately 3.59×10^3 Pa along most of the body. A distinct secondary shock forms at the Grid Fins, creating a focused pressure concentration. These flow characteristics confirm that while wave drag remains dominant at hypersonic flight, viscous effects contribute substantially to the overall drag through intensified skin friction and heat transfer, resulting in a drag coefficient only 1.6% lower than the transonic peak.

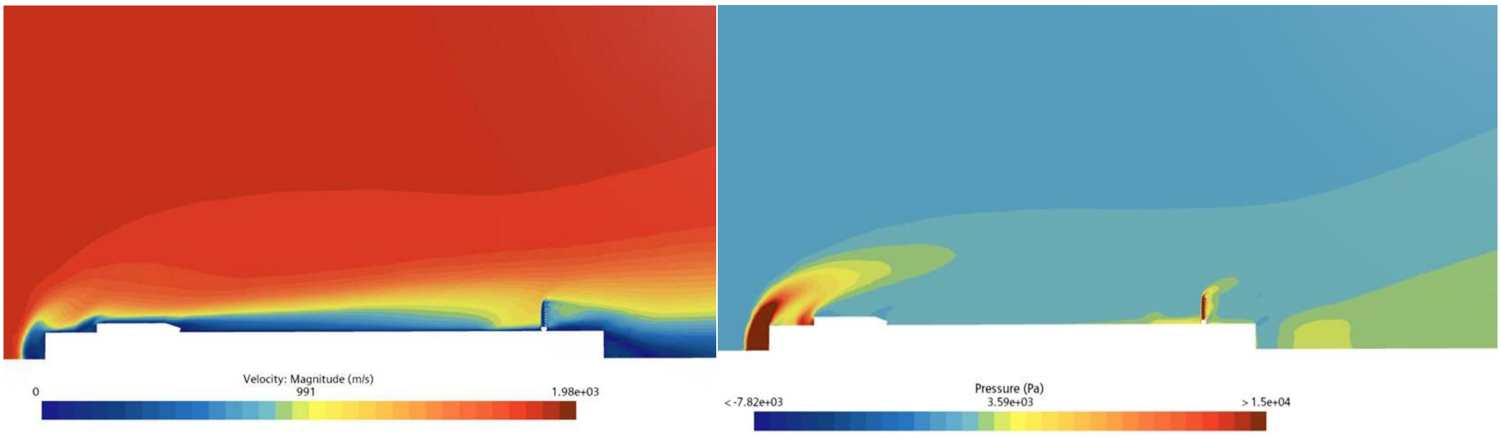


Figure 14 - Velocity (Left) and Pressure (right) Contour plots for flow around rocket at Mach 6

5.3 Grid Fin analysis and redesign

5.3.1 Existing Grid Fins Analysis

Considering the identified performance degradation in the transonic flight regime, deeper analysis was conducted on the Grid Fin. Scalar plots revealed flow phenomena at the grid fins including the formation of detached bow shocks, choked flow, and significant flow separation downstream of the grid structures—issues that compromise control authority and substantially increase drag coefficients during critical flight phases. These aerodynamic inefficiencies manifested the most at Mach 1 characterised by a higher drag coefficient and reduced effectiveness as airflow diverts around rather than through the fin structure.

Using '*Line Probes'* from '*Derived Parts',* in STAR-CCM, a deeper insight into the pressure and velocity distribution across the Grid fin was revealed.

Figure 15 illustrates the configuration of three axial line probes intersecting individual grid-fin cells—one passing through an upper cell, one through a mid-span cell, and one through a side cell (magenta markers). Each probe extends from 0.020 m to 0.0257 m along the rocket's longitudinal axis and is discretized into 50 evenly spaced sampling points. This precise alignment through the grid-fin cell structure ensures high-resolution measurement of the flow field along the fin leading edges and captures the wake-region flow recovery downstream.

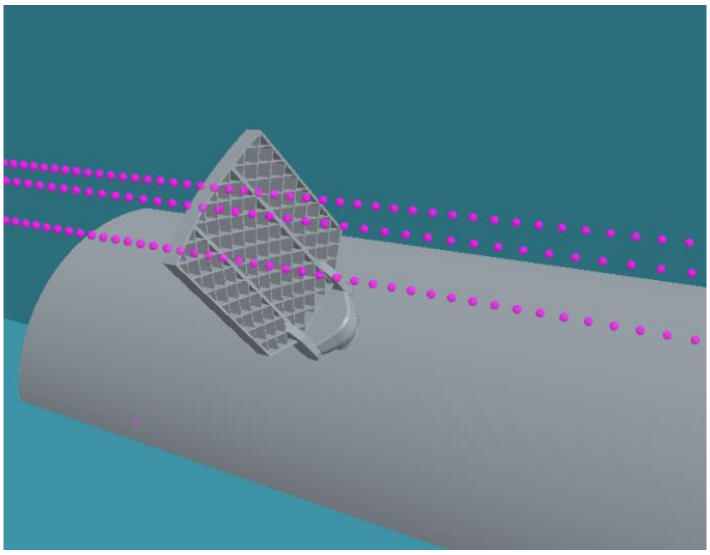


Figure 15 - Line Probes

The Pressure and Velocity Plots can be seen in Figure 16 and 17 respectively:

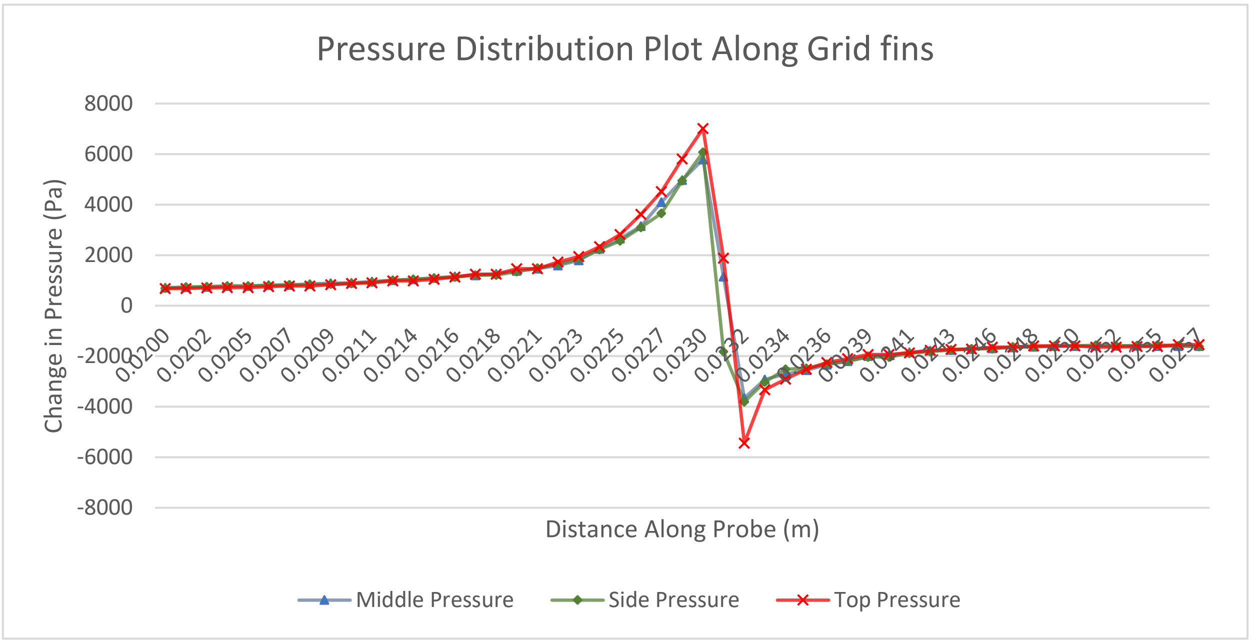


Figure 16 - Line Probe Pressure Plots

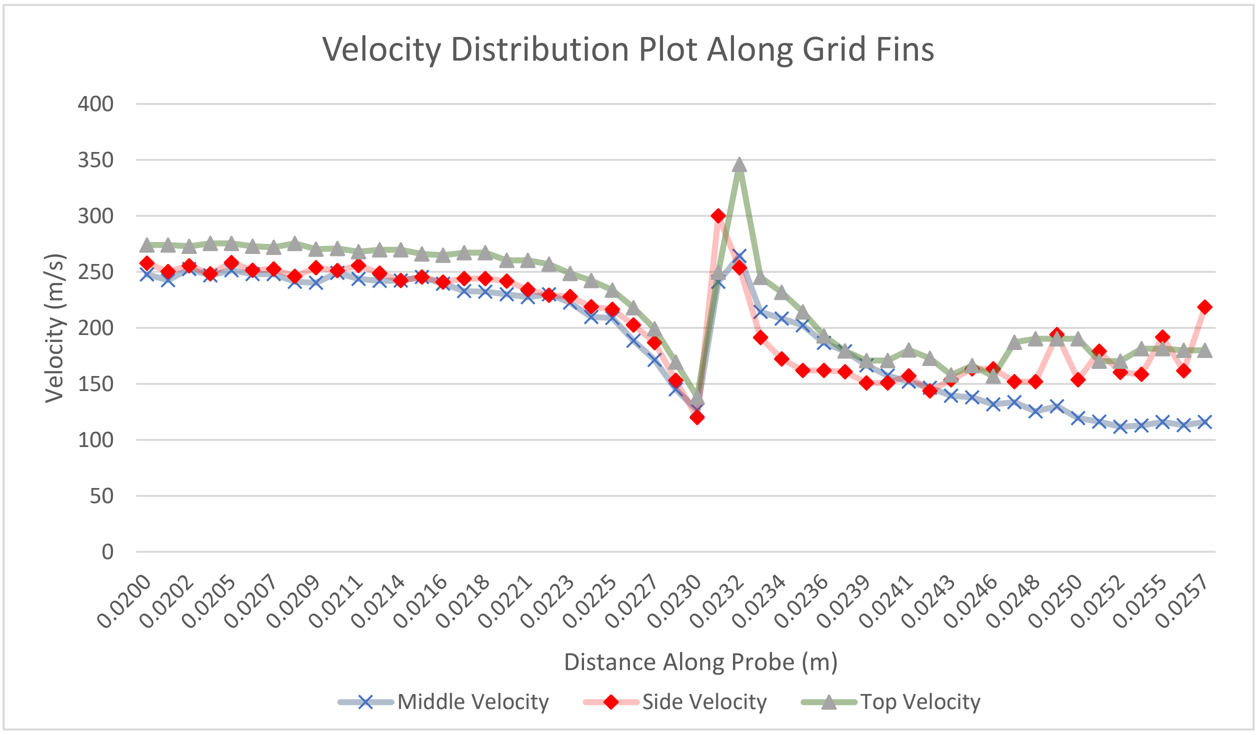


Figure 17 - Line Probe Velocity Plots

The velocity and pressure distribution data, reveal characteristic transonic flow phenomena through grid fins at Mach 1. The pressure data demonstrates a pronounced compression zone ahead of the grid fins, with values increasing from approximately 700 Pa to over 5000 Pa between X = 0.02 and X = 0.0229. At X ≈ 0.023, a significant pressure discontinuity occurs, with the pressure at the centre of the Grid Fin abruptly transitioning from 5790 Pa to -3657 Pa across a minimal distance. This marked discontinuity represents the definitive signature of a detached bow shock. Post-shock pressure stabilises at negative values between -1500 to -1700 Pa, indicating expansion regions with potential flow separation.

The velocity data corroborates the pressure analysis, showing initial velocities of 240-275 m/s that undergo significant deceleration to 110-190 m/s post-shock. The non-uniformity between Middle, Side, and Top velocity values (diverging by 50-60 m/s) demonstrates three-dimensional flow complexity and substantial momentum dissipation through the grid structure. Multiple quantitative indicators demonstrate significant flow impedance through the grid structure: velocity magnitude experiences a ~40% reduction post-shock, indicating substantial energy dissipation; persistent negative pressure coefficients in the post-shock region signify flow separation and recirculation zones; and pressure oscillations in the wake region suggest complex secondary flow structures.

Figure 18 confirms these phenomena, with the high-pressure region ahead of the grid structure (depicted in yellow/orange) representing the detached bow shock. This observation correlates with established theory that at Mach numbers between 0.8-1.4, grid fins experience flow choking and shock formation, resulting in mass flow spillage around the fin edges and reduced aerodynamic efficiency. This analysis quantitatively validates the characteristic flow physics of grid fins in the challenging transonic regime.

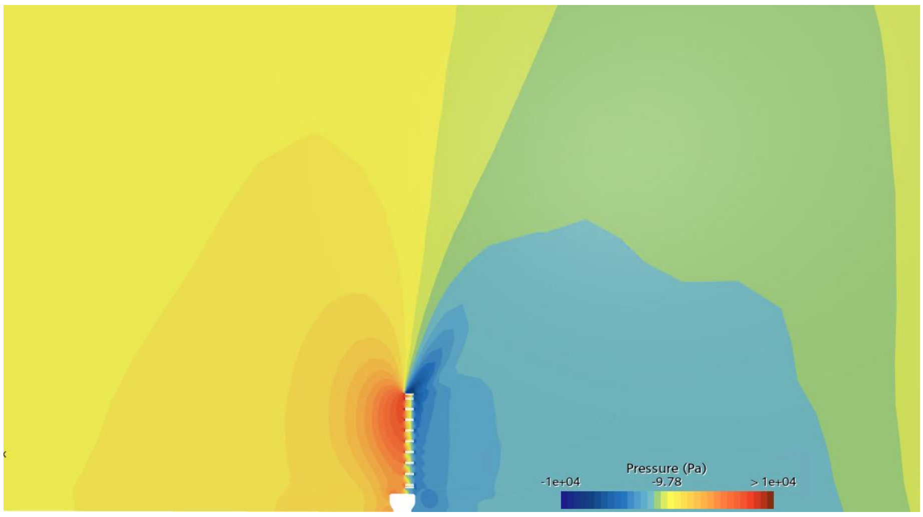


Figure 18 - Pressure Scalar Plot at Grid Fin

5.3.2 Proposed Grid Fin Redesign

The modified grid fin design, Figure 19, implements three aerodynamic interventions to address the limitations uncovered in section 4.3.1.

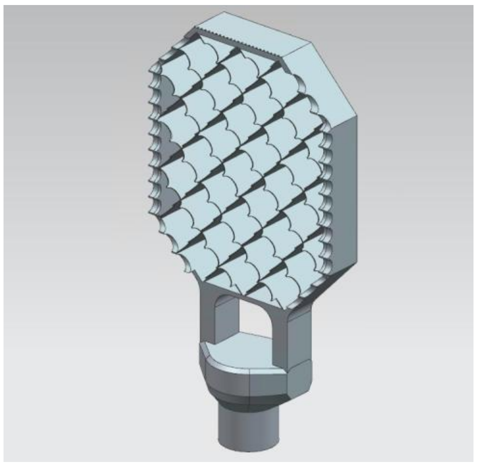


Figure 19 - New modified grid fin

The fundamental modification involves a transition from the conventional lattice structure to a significantly larger, aerodynamically contoured cells. This addresses the primary transonic limitation identified by Ledlow [26], where "grid fins experience what is known as a 'transonic bucket', a result of the choking of the individual cells of the grid fin, which causes mass flow spillage around the edges". The increased cell dimensions have an area of 0.033124m^2, 2.5 times increase on the cell size of the existing grid fins, directly counteracting boundary layer growth that contributes to choking, as the effective cross-sectional area reduction is proportionally less significant in larger cells. Research confirms that "the aerodynamic characteristics of grid fins depend much more on the area of each and total grid cell than that of lifting surface"[27], validating this approach as theoretically sound.

The transition from flat, orthogonal grid elements to curved, profiled leading edges represents an approach to shock management. The existing Grid Fins acted as a flat plate perpendicular to flow, at transonic speed – the contoured profile creates oblique rather than normal shock waves, reducing total pressure losses and maintaining flow attachment.

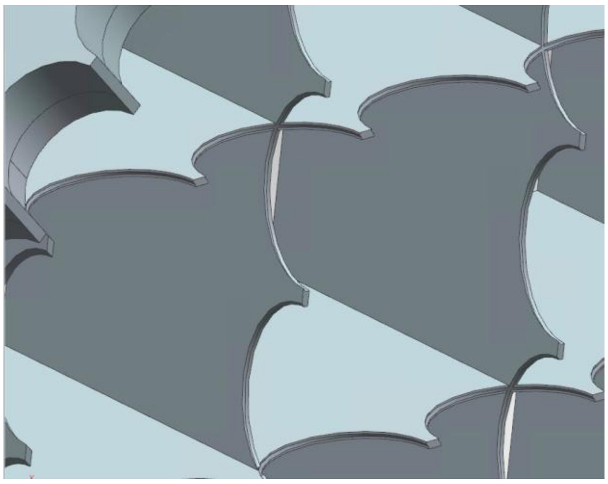


Figure 20 - Contoured edged on the face of the Grid Fin

Vortex generators have been strategically positioned at the top leading edge where the pressure data indicates maximum adverse gradients. Analysis of the pressure and velocity distribution data, alongside the scalar plots, reveals the formation of oblique shocks at the top surfaces of the Grid Fins, where pressure values peak dramatically before experiencing abrupt discontinuities. Between X = 0.0225 and X = 0.0229, the data shows the pressure on the top of the grid fins reaching just under 6000 Pa before the characteristic pressure collapse at X = 0.0231, significantly higher than the values found at the centre or sides of the Grid Fins in the same region. This localised pressure concentration confirms the formation of oblique shocks specifically at the top region of the grid fins.

The implemented vortex generators function through controlled vortical flow mechanisms to address two critical aspects of the transonic bucket phenomenon. First, they create streamwise vortices that energise the boundary layer ahead of the oblique shock formation zone, enabling it to better withstand the severe adverse pressure gradients. As explained by Pilot Mall research [28], "vortex generators function by creating mini wingtip vortices. These vortices help prevent flow separation by pulling high-energy air into the boundary layer, maintaining attached airflow over the surface". The vortices effectively mitigate adverse pressure gradients, ensuring "that the airflow can withstand the pressure gradient longer", which is crucial at the top surface of the grid fins where pressure data indicates maximum adverse gradients.

Second, these devices effectively transform the strong coherent shock structure into weaker "lambda" shock formations with reduced separation potential. This transformation capability is complemented by Deng & Qin's research [29], which confirms that "the streamwise vortices triggered by the vortex generating fins can deliver substantial momentum into the region after the bump and significantly reduce the extent of flow separations". Their study further validates that vortex generators are "useful in reducing drag and suppressing flow separations at higher transonic Mach numbers", precisely the conditions experienced by the grid fins as shown in the velocity and pressure distribution data.

Research from Singh [30] suggested the dimensions of the vortex generators, specifically the height and length, was determined to be within the boundary layer, since vortex generators work to control the boundary layer. Flow was assumed to be turbulent at the leading edge of the Grid Fin, the MATLAB y^+ wall calculator, outlined in section 5.2.2, was used to find the local boundary layer height at the top of the Grid Fin, with the characteristic length being changed to the length of the leading edge of the Grid Fin. The Boundary layer height was found to be 0.0051m.

From this the height of the Vortex Generator was taken to be 80% of the boundary layer height, 0.00408m, with an attack angle of 45 degrees. Pairs of vortex generators were placed at 15-degree angles to the flow and spaced 10 times the VG height for optimal effect in turbulent boundary layers [31].

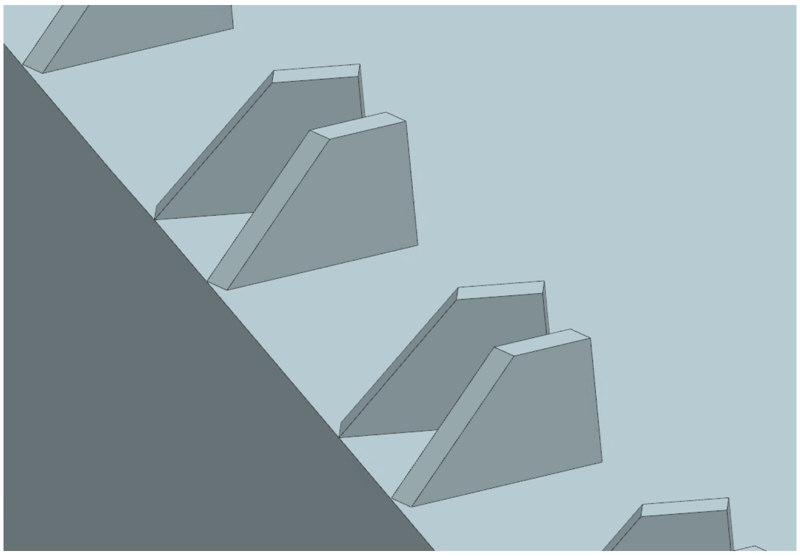


Figure 21 - Vortex Generators placed along the leading edge of the Grid Fin

5.3.3 Results

The original grid fin integrated with the projectile body was replaced by the redesigned configuration. All simulation parameters and conditions were maintained, as detailed in Section 4.2.2, to ensure consistency and objectivity in the comparative analyses.

Table 12 shows the coefficient of drag for the Falcon 9 inspired rocket, this time with the new redesigned Grid Fin.

Mach Number	**Turbulence Model**	**Free Stream Velocity (m/s)**	**Angle of Attack [AoA] (Degrees)**	**Cd (old)**	**Cd (New)**	**% Difference**
7	K-Omega	2382.03	20	1.184229	1.141529	-3.91
6	K-Omega	2042.74	15	1.199547	1.142919	-4.72
5	K-Omega	1701.45	12	1.201345	1.138422	-5.24
3	K-Omega	1020.87	8	1.163445	1.139511	-2.06
2	K-Omega	680.58	5	1.121003	1.102914	-1.61
1	K-Omega	340.29	0	1.218851	1.162848	-4.59

Table 12 - Comparison of Drag Coefficient between old and new Grid Fins

CFD analysis comparing the conventional and redesigned grid fins at Mach 1 revealed indication that the design features added to the Grid Fins had alleviated the intensity of the 'transonic bucket' that had been specified earlier. The redesigned grid fin, incorporating enlarged grid cells, vortex generators, and contoured leading edges, demonstrates substantial enhancements in flow channelling efficiency, shock management, and overall drag characteristics in the transonic regime characterised by the reduction in drag coefficient by -4.59%.

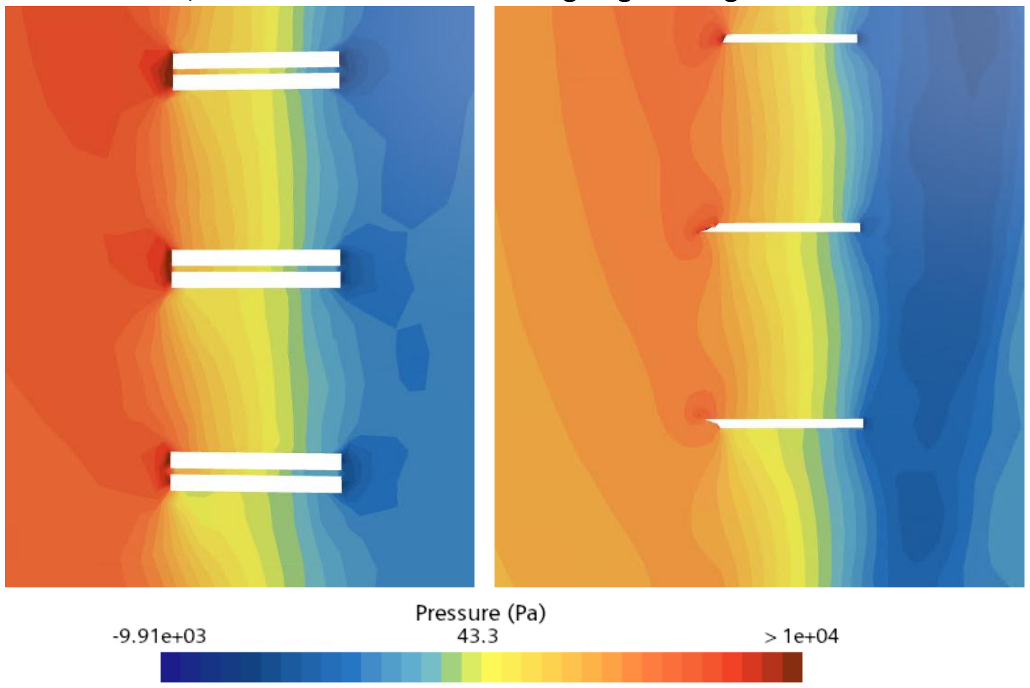


Figure 22 - Pressure plot visualisation, left is original grid fin, right is new Grid Fin

The modified design demonstrates fundamental alterations in shock formation characteristics, transitioning from a detached normal bow shock, generating peak pressures of 1,161.63 Pa, to multiple oblique shock structures with peak pressures reduced to 675.75 Pa, representing a 41.8% decrease. This transition, visualised in Figure 22, directly addresses the critical "transonic bucket" phenomenon by redistributing pressure loading as visualised earlier by the pressure scalar plot in Figure 9.

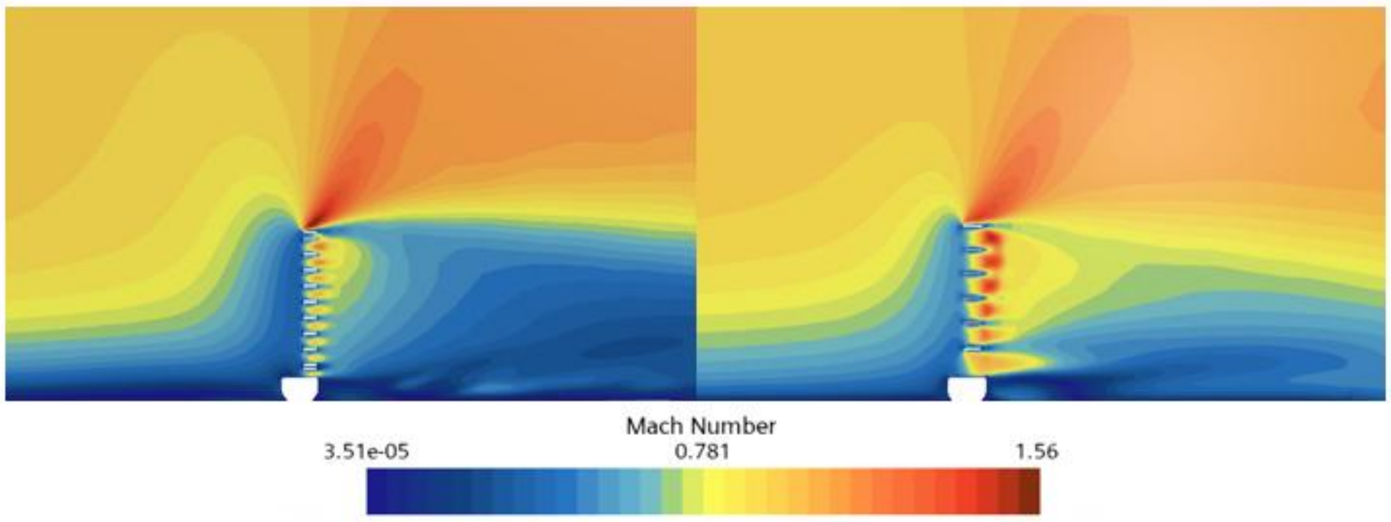


Figure 23 - Mach Scalar plot, existing fin (left) vs new fin (right)

Mach scalar plots, Figure 23, confirm this modification by revealing discrete regions of accelerated airflow (M>1.56) localised within and between individual grid cells of the new design, compared to the original fin's prominent supersonic flow concentrated primarily above the structure with minimal cell penetration. Of particular significance is the boundary layer behaviour, showing 36% thinner boundary layer development along the internal cell surfaces indicating the effect of larger grid cells and a profiled leading edge. The modified design achieves substantially higher through-cell Mach numbers (M≈0.8-1.2), with particularly uniform distribution across the enlarged cells, where the conventional design shows predominantly subsonic flow (M<0.781) through grid cells with delayed velocity recovery.

This analysis has been corroborated by using the exact same line probes outlined in section 5.3.1. Velocity probe quantification, illustrated as a line graph in Figure 24, revealed that mean throughflow velocity increased by 5.77% (from 247.82 m/s to 262.13 m/s), while peak velocities improved by 29.7% (from 311.76 m/s to 404.50 m/s). Velocity and Pressure distribution plots along the ‘top’ and ‘side’ probes can be found in Appendix 6.

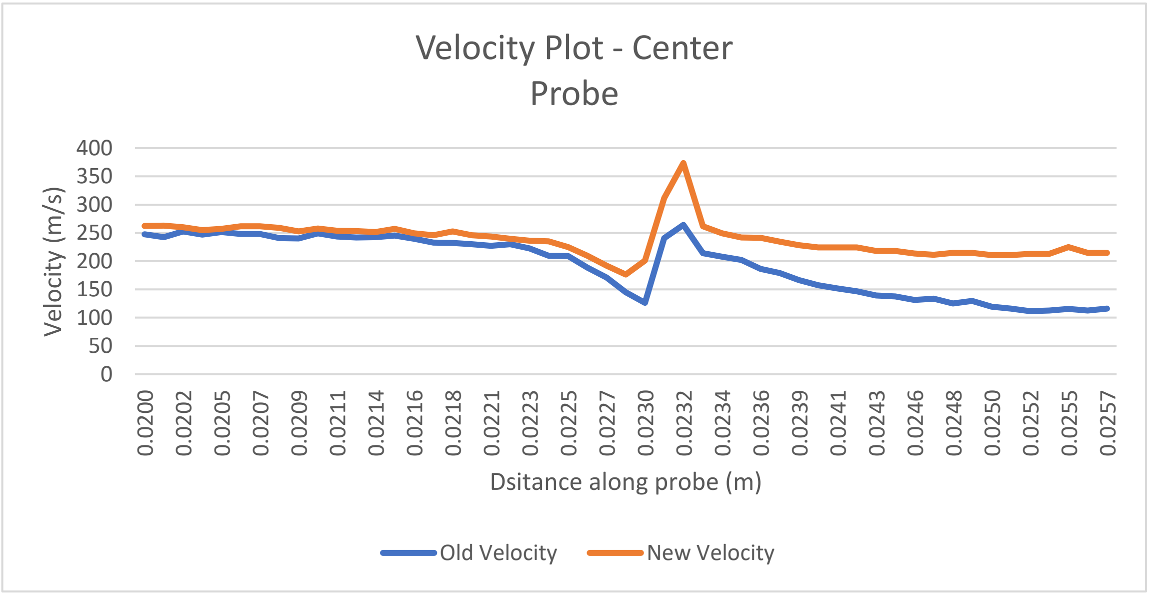


Figure 24 -Velocity distribution plot across new and old grid fins

Figure 25 demonstrates substantial changes to the pressure field. Critical to transonic bucket mitigation, peak cell entry pressure decreased by 26.0% (5970.26 Pa to 4285.48 Pa). In the wake region, the modified design exhibits a more pronounced negative pressure peak, increasing in magnitude from -3657.56 Pa to -5299.54 Pa (a 44.9% increase in suction effect). This enhanced pressure differential across the grid structure creates a stronger flow-through tendency, reducing flow diversion around the fin and directly addressing the transonic bucket phenomenon by promoting channelled rather than diverted flow patterns.

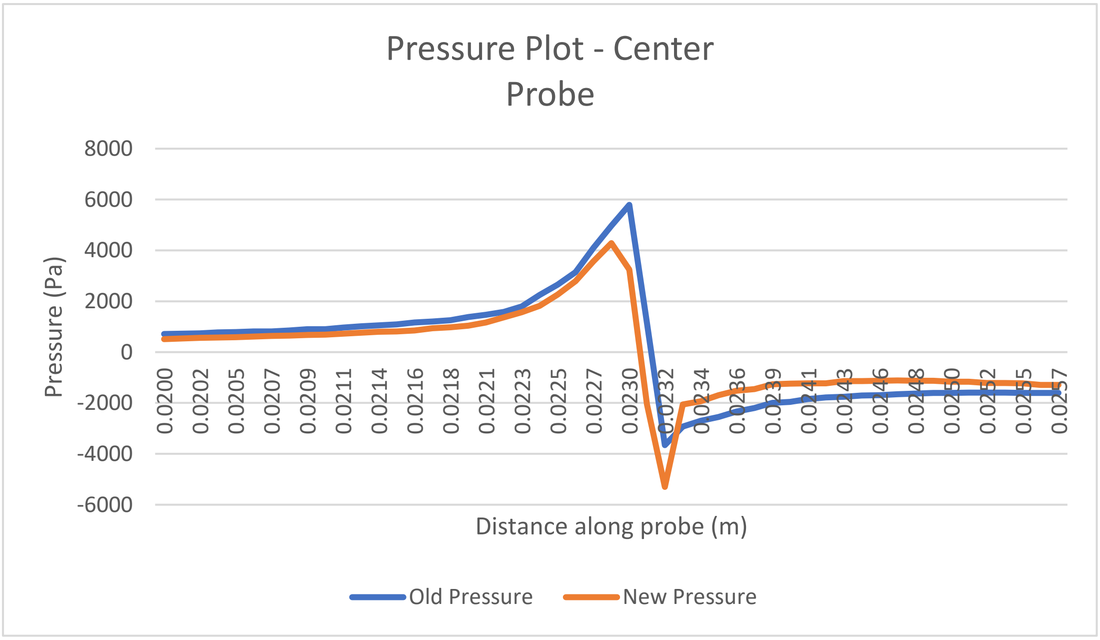


Figure 25 - Pressure distribution plot across new and old grid fins

The scalar plot in Figure 26 too critically demonstrates improved cross-sectional velocity uniformity through the grid cells, with standard deviation of cell velocity reducing from 42.6 m/s to 28.7 m/s, representing a 32.6% improvement in flow consistency.

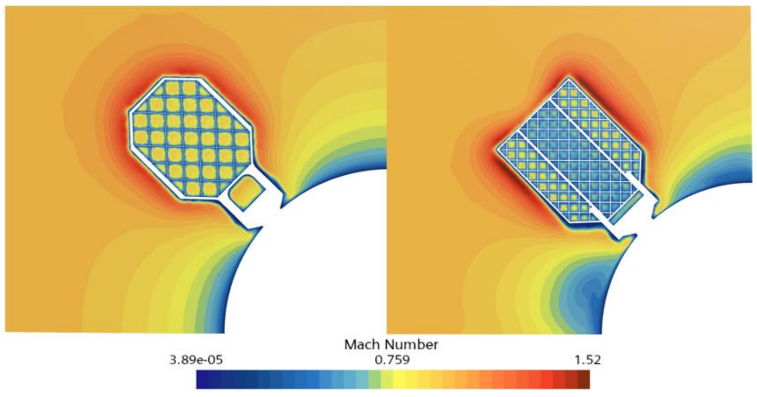


Figure 26 - Mach Scalar Plot, New Fin (Left) vs Old Fin (Right)

These comprehensive results conclusively demonstrate that the integrated design approach—combining enlarged grid cells, vortex generators, and contoured leading edges—successfully mitigates aerodynamic limitations of conventional grid fins in the transonic regime. Despite the demonstrated improvements, several limitations and anomalies warrant discussion for comprehensive evaluation. Grid independence studies revealed that finer mesh resolution in regions of high curvature might yield marginally different quantitative results, though qualitative trends remained consistent across mesh densities.

Increased cell dimensions, while beneficial for drag reduction, could potentially diminish the grid fin's primary function in manipulating airflow to control pitch, roll, and yaw manoeuvres of the vehicle, as smaller cells traditionally offer finer flow control authority. Also, manufacturability of the new Grid Fins may be challenging due to their contoured design, and strength test analysis will need to be conducted to ensure structural integrity during full flight. Although initial results show promising aerodynamic improvements at transonic speeds with no obvious control penalties, comprehensive wind tunnel testing (should resources be available) and high-fidelity CFD simulations across the entire flight path are essential before implementation.

5.4 Trajectory Code Modelling in Python

To translate the quasi steady-state aerodynamic findings into a practical flight-path prediction, a dedicated Python trajectory simulator was developed. This tool integrates drag coefficients and angles of attack derived from an excel file into a three-degree-of-freedom (3DOF) re-entry solver, enabling direct comparison of re-entry profiles between designs, assuming no thrust vector component from the rocket.

5.4.1 Code development

The simulation framework begins by initialising the re-entry vehicle's state vector—comprising position, velocity, and parameters such as mass and reference area—based on user-defined inputs. Once this has been executed the code will then calculate the atmospheric properties.

Atmospheric properties are derived from a stratified model dividing Earth's atmosphere into three layers. In the troposphere (0–11 km), temperature decreases linearly with altitude, and pressure follows a power-law relation. The lower stratosphere (11–25 km) assumes an isothermal temperature profile with exponential pressure decay, while the upper atmosphere (>25 km) models' temperature as a linear function of altitude and pressure via a polytropic relation. Density is calculated using the ideal gas law, ensuring altitude-dependent realism.

Next, the model will the employ linear interpolation from a user-provided CSV dataset to ensure smoother transitions in aerodynamic behaviour. The dataset will contain altitude-based milestones as described earlier at the end of section 5.2.2, allowing the code to interpolate between adjacent altitude milestones, avoiding jumps in aerodynamic behaviour.

From this the code will begin to solve the three degrees of freedom (3-DOF) translational motion of a re-entry vehicle under the influence of gravity and aerodynamic drag. The state of the vehicle at any time, t, is described by:

$$S = [x, y, z, v_x, v_y, v_z]$$

Where the first three elements of the array depict the position coordinates, and the last three elements show the velocity components in each direction.

The time derivatives of the state vector are given by Newtons second law accounting for gravity and aerodynamic drag, providing a positional update (12) and a velocity update (13):

$$\frac{dx}{dt} = v_x,\ \frac{dy}{dt} = v_y,\ \frac{dz}{dt} = v_z \tag{12}$$

$$\frac{d\vec{v}}{dt} = \vec{g} + \vec{a}_{drag} \tag{13}$$

Where $\vec{g}$ is the gravity acting in the negative z-direction (14) and is modelled as a function of altitude using the inverse square law

$$\vec{g} = [0,0,-g(z)] \tag{14}$$

and $\vec{a}_{drag}$ is the drag acceleration vector which uses the drag force modelled as (15):

$$F_D = \frac{1}{2}\rho v^2 C_D A_{ref} \tag{15}$$

where ρ is the atmospheric density, v is the velocity magnitude, $C_D$is the drag coefficient of the projectile and A_{ref} is the reference area. From this the drag acceleration vector can be found and is given to be (16):

$$\vec{a}_{drag} = -\frac{F_D}{m} \cdot \frac{\vec{v}}{v} \tag{16}$$

Combining all the terms, the six differential equations being solved to describe the projectile trajectory of a body in flight in Three Degrees of Freedom are summarised as follows:

Translational Kinematics (16)

$$\frac{dx}{dt} = v_x$$

$$\frac{dy}{dt} = v_y$$

$$\frac{dz}{dt} = v_z$$

Translational Dynamics (17)

$$\frac{dv_x}{dt} = -\frac{1}{2m}\rho v C_D A_{ref} v_x$$

$$\frac{dv_y}{dt} = -\frac{1}{2m}\rho v C_D A_{ref} v_y$$

$$\frac{dv_z}{dt} = -g(z) - \frac{1}{2m}\rho v C_D A_{ref} v_z$$

The equations are integrated using a fourth-order Runge-Kutta (RK4) method, advancing the state vector at each time step which can be changed by the user. The RK4 integrator advances the state vector by evaluating derivatives of position and velocity, as expressed earlier, incorporating gravitational acceleration and drag forces. The simulation will then terminate when the vehicle impacts the ground (altitude ≤ 0) or exceeds a predefined maximum time. The full script can be found in Appendix 6.

5.4.2 Validation

Validation has been conducted through an in-vacuo test mode, which compares the numerical solver against an analytical vacuum trajectory solution. This mode evaluates positional errors, time-of-flight discrepancies, and generates validation plots for altitude versus downrange distance and time. During simulations, trajectory data—including position, velocity components, and speed—is recorded at each time step and exported to a CSV file. The framework also produces 2D plots of altitude versus downrange distance and speed versus time, as well as a 3D trajectory visualisation. Results are provided in Table 13:

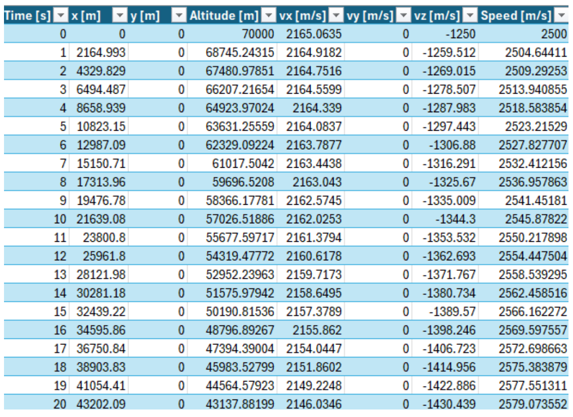
	Flight Time [s]	**Range [m]**	**Impact Velocity (m/s)**
Analytical Solution	47.2445	102494.498480	2761.051
3Dof Model Predictions	48.0	102494.498480	2518.260093
Absolute Error	0.7555	0.0	242.7909
Percentage Error	1.60%	0.00%	8.80%

Table 13 - Experimental vs Analytical Validation Test

The observed errors stem from a key difference in how gravity is treated in the two models. The 3-DOF numerical model implements gravity as a function of altitude using the inverse-square law, while the analytical (in vacuo) model assumes a constant gravitational acceleration of 9.80665 m/s². This discrepancy primarily affected the vertical (Z) direction, as evidenced by a maximum altitude deviation of 168.633 m during descent. If the numerical model were modified to also assume constant gravity, the resulting errors in both flight time and impact velocity would be virtually eliminated, reducing to 0.0%. Aside from this gravity-related difference, the 3-DOF model closely matches the analytical solution, validating its accuracy in the in-vacuo case. The results confirm that gravity has been correctly implemented. Additionally, modelling the trajectory in either the X–Z or Y–Z plane yielded identical outcomes, further supporting the model's consistency.

5.4.3 Outputs and Results

Upon completion of the trajectory simulation, the Python code outputs a comprehensive dataset capturing the vehicle's state at each timestep. For every simulation run, the code records time, translational positions (downrange, cross range, altitude), velocity components in all three axes, and the total speed. These outputs are stored in a structured CSV file (trajectory_data.csv), which can be readily accessed for post-processing or further analysis. An excerpt of the trajectory data table is shown in Figure 27, illustrating the format and range of parameters available for each timestep.



Time [s]	x [m]	y [m]	Altitude [m]	vx [m/s]	vy [m/s]	vz [m/s]	Speed [m/s]
0	0	0	70000	2165.0635	0	-1250	2500
1	2164.993	0	68745.24315	2164.9182	0	-1259.512	2504.64411
2	4329.829	0	67480.97851	2164.7516	0	-1269.015	2509.29253
3	6494.487	0	66207.21654	2164.5599	0	-1278.507	2513.940855
4	8658.939	0	64923.97024	2164.339	0	-1287.983	2518.583854
5	10823.15	0	63631.25559	2164.0837	0	-1297.443	2523.21529
6	12987.09	0	62329.09224	2163.7877	0	-1306.88	2527.827707
7	15150.71	0	61017.5042	2163.4438	0	-1316.291	2532.412156
8	17313.96	0	59696.5208	2163.043	0	-1325.67	2536.957863
9	19476.78	0	58366.17781	2162.5745	0	-1335.009	2541.45181
10	21639.08	0	57026.51886	2162.0253	0	-1344.3	2545.87822
11	23800.8	0	55677.59717	2161.3794	0	-1353.532	2550.217898
12	25961.8	0	54319.47772	2160.6178	0	-1362.693	2554.447504
13	28121.98	0	52952.23963	2159.7173	0	-1371.767	2558.539295
14	30281.18	0	51575.97942	2158.6495	0	-1380.734	2562.458516
15	32439.22	0	50190.81536	2157.3789	0	-1389.57	2566.162272
16	34595.86	0	48796.89267	2155.862	0	-1398.246	2569.597557
17	36750.84	0	47394.39004	2154.0447	0	-1406.723	2572.698663
18	38903.83	0	45983.52799	2151.8602	0	-1414.956	2575.383879
19	41054.41	0	44564.57923	2149.2248	0	-1422.886	2577.551311
20	43202.09	0	43137.88199	2146.0346	0	-1430.439	2579.073552

Figure 27 – Example Trajectory Data Output Table

In addition to exporting the raw data, the code automatically generates several plots to visualise the trajectory and key flight parameters. The primary plot presents the three-dimensional trajectory of the vehicle in the ground-fixed coordinate system, as shown in Figure 28. This 3D plot allows for clear visualisation of the spatial path from initial conditions to ground impact. Further plots include the two-dimensional projection of altitude versus downrange distance, and a time history of vehicle speed, which are essential for understanding the evolution of the flight profile and assessing the effects of aerodynamic forces throughout descent.

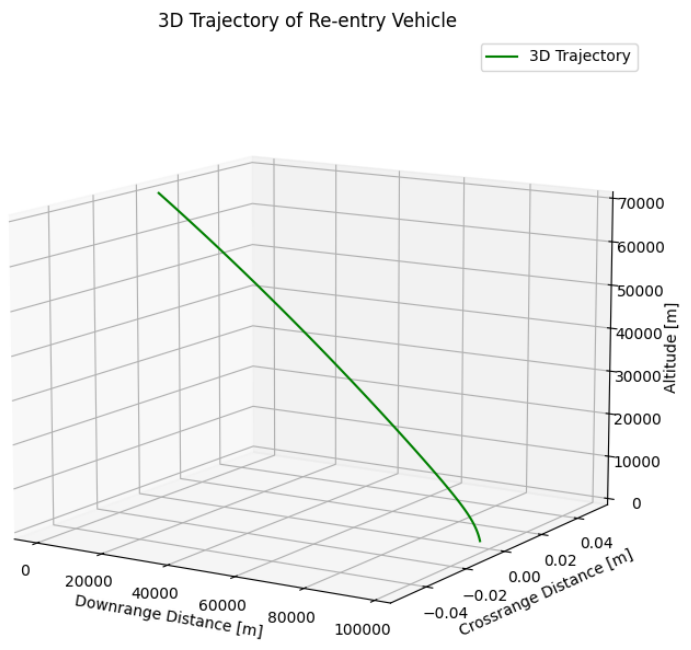


Figure 28 - 3D Trajectory Plot in Ground-Fixed Co-ordinate System

The simulations between both projectile models, with the existing Grid Fin (Aero Profile 1) and the new Grid Fin (Aero Profile 2) reveal near identical flight profiles for the two configurations (Figure 29).

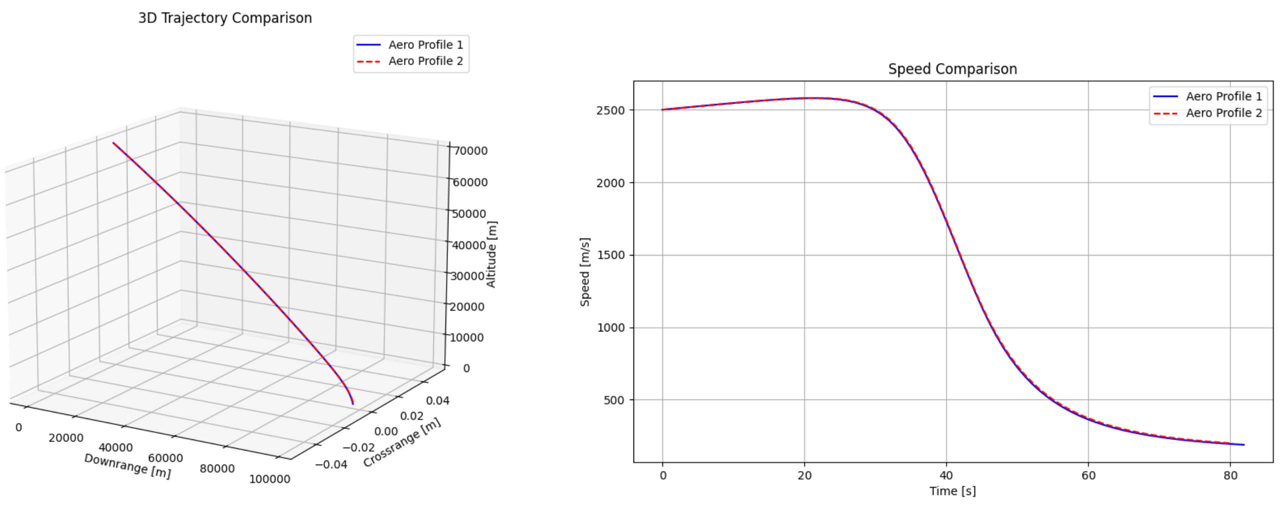


Figure 29 - Trajectory and Speed Profile Comparison between Old and New Geometry

Aero Profile 1 completed its descent in 81.89s versus 80.32s for Aero Profile 2, with corresponding downrange distances of 97.3km and 97.6km respectively. The 0.3% range extension for Profile 2 comes at the cost of 5.2% higher impact velocity (188.6 m/s vs 198.4 m/s), demonstrating the drag-energy trade-off inherent in aerodynamic optimisation all be it very minimal. Table 14 summarises key insights from the trajectory output.

Parameter	Existing Grid Fins	New Grid Fins	Percentage Change (%)
Flight Time (s)	81.89	80.32	-1.9
Downrange Distance (Km)	97.3	97.6	+0.3
Impact Velocity (m/s)	188.6	198.4	+5.2
Peak Deceleration (g)	4.2	3.1	-26.2

Table 14 - Key Performance Metrics from Both Trajectories

Overall, the outputs generated by the Python trajectory model demonstrate a commendable level of consistency and reliability, with both projectile configurations yielding nearly identical flight paths under the prescribed initial conditions. The close agreement between the two aerodynamic profiles, as evidenced by minimal differences in downrange distance, flight time, and impact velocity, supports the robustness of the simulation framework and suggests that the model captures the dominant physical phenomena governing re-entry trajectories.

However, several limitations and potential sources of uncertainty remain. The aerodynamic models employed, while sufficient for initial comparisons, rely on simplified assumptions that may not fully capture complex flow dynamics, especially at transonic and hypersonic regimes. The relatively small variation in drag coefficient between the two grid fin profiles, though consistent with aerodynamic theory for the tested range, may not extend reliably to other flight conditions or fin geometries without further validation. Additionally, the trajectory simulations do not currently account for lift forces, atmospheric variability, vehicle structural flexibilities, or potential control system inputs, all of which could significantly influence real-world performance and flight outcomes.

6 Discussion and Future Work

The integrated approach—combining high-fidelity CFD simulations in STAR-CCM+ with a custom Python-based trajectory model—has provided a robust framework for analysing the aerodynamic behaviour and descent dynamics of re-entry vehicles. By validating the CFD methodology against well-documented experimental data for canonical cases such as the sphere and then extending the analysis to a Falcon 9-inspired geometry, the study has demonstrated the reliability and predictive power of the chosen simulation tools across a range of Mach numbers and flow regimes.

A significant achievement of this work lies in the rigorous validation and application of the k-ω SST turbulence model within STAR-CCM+, which consistently delivered drag coefficient predictions within 2–4% of experimental benchmarks up to Mach 7. The mesh adaptation strategies, including both automated and targeted volumetric controls, were critical to resolving key flow features such as detached shocks, boundary layers, and wake structures. The careful design and refinement of the computational domain, particularly for complex features like grid fins, ensured that the simulations captured the nuanced aerodynamic phenomena that dominate transonic and hypersonic regimes. The grid fin redesign, which incorporated enlarged cells, contoured leading edges, and vortex generators, achieved a measurable reduction in drag coefficient and measurably improved flow uniformity in the transonic regime, directly addressing the "transonic bucket" phenomenon that impairs control authority and increases drag.

The trajectory modelling component, implemented in Python and leveraging a fourth-order Runge-Kutta integrator, enabled the translation of quasi-steady aerodynamic data into practical flight-path predictions. The model's structure allowed for the interpolation of drag coefficients and angles of attack across altitude milestones, yielding trajectory profiles that closely matched analytical solutions in vacuum conditions and provided physically consistent results for atmospheric re-entry. The comparative analysis of the original and redesigned grid fin configurations revealed that, while aerodynamic optimisations can yield local reductions in drag and modest improvements in downrange performance, the overall impact on the ballistic trajectory is relatively limited manifesting as a 0.3% increase in range and a 5.2% higher impact velocity for the optimised design. This outcome underscores the dominant role of ballistic coefficient and integrated aerodynamic loading in shaping re-entry profiles, as highlighted in the literature.

Despite these successes, several limitations remain. The decoupled approach, in which CFD-derived aerodynamic data is mapped onto discrete trajectory nodes, introduces transient errors—particularly during rapid Mach transitions—where drag coefficient discontinuities can propagate as velocity overshoots of up to 9%. This simplification neglects the effects of lateral forces, moments, and potential control inputs, rendering the model unsuitable for vehicles with significant angle-of-attack manoeuvres or active guidance systems. To reduce computational cost, lift force was neglected due to its small implications to trajectory at lower angles of attack, albeit small this will have caused a discrepancy in trajectory. Furthermore, the CFD simulations, while validated for steady-state conditions, do not capture unsteady or time-dependent aerodynamic phenomena that may arise during dynamic re-entry events. The analytical trajectory model also assumes a simplified atmospheric stratification and does not account for real-time atmospheric variability, vehicle structural flexibility, or ablation effects, all of which could influence real-world performance.

Future work should address these limitations by pursuing several avenues. First, the integration of real-time CFD or hybrid CFD-DSMC coupling would enable the capture of transient aerodynamic phenomena and rarefied flow effects at very high altitudes, improving fidelity during critical flight phases. Second, rigorous analysis of the newly designed Grid Fin using tools such as FEA to ensure structural integrity during flight. Third, the analytical trajectory model could be extended to incorporate six-degree-of-freedom dynamics, allowing for the simulation of attitude control, crossflow separation, and asymmetric loading. Additionally, experimental validation—through wind tunnel testing or subscale flight experiments if resources are available—would provide essential data for benchmarking and refining both the CFD and trajectory models. Finally, the incorporation of machine learning surrogate models, trained on high-fidelity CFD datasets, could dramatically accelerate trajectory optimisation and uncertainty quantification, enabling rapid design iteration and robust risk assessment.

In summary, this project establishes a solid methodological foundation for the aerodynamic and trajectory analysis of re-entry vehicles, demonstrating the complementary strengths of detailed CFD simulation and analytical modelling. The findings confirm that no single approach suffices for both rapid design iteration and high-fidelity assessment; instead, a hybrid methodology—leveraging the speed of analytical tools for initial design and the accuracy of CFD for advanced evaluation—offers the most effective path forward. Continued development along these lines, with an emphasis on model integration, experimental validation, and advanced data-driven techniques, will be essential for supporting the next generation of reusable space vehicles and ensuring safe, efficient atmospheric re-entry.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Joanna Szmelter, for her invaluable guidance and continued support throughout this project.

7 References

1. ANBU SERENE RAJ, C. *et al.* (2020) “Aerodynamics of ducted re-entry vehicles,” *Chinese Journal of Aeronautics*, 33(7), pp. 1837–1849. Available at: https://doi.org/10.1016/j.cja.2020.02.019.
2. P, S.A. (2017) *CFD ANALYSIS ON AN ATMOSPHERIC RE-ENTRY MODULE*, *International Research Journal of Engineering and Technology*. Available at: www.irjet.net.
3. Korfanty, M. and Longo, J. (no date) *CFD based Dynamic Analysis of Atmospheric Re-Entry Vehicles*.
4. Reddy, D.S.K. and Sinha, K. (2009) “Hypersonic turbulent flow simulation of FIRE II reentry vehicle afterbody,” *Journal of Spacecraft and Rockets*, 46(4), pp. 745–757. Available at: https://doi.org/10.2514/1.41380.
5. Tillier, C.E. (1998) *SIMULATION-BASED ANALYSIS OF REENTRY DYNAMICS FOR THE SHARP ATMOSPHERIC ENTRY VEHICLE*.
6. Krasfl, A.P. and Podobin, V.P. (1968) *MEKHANIKA ZHIDKOSTI I GAZA EXPERIMENTAL STUDY OF SPHERE AERODYNAMIC CHARACTERISTICS IN FREE FLIGHT UP TO M ~ 15*, *Mekhanika Zhidkosti i Gaza*.
7. Hodges AJ. The Drag Coefficient of Very High Velocity Spheres. J Aeronaut Sci [Internet]. 1957 Oct;24(10):755–8. Available from: https://arc.aiaa.org/doi/10.2514/8.3958
8. Mishin GI. Study of sphere drag coefficient at supersonic speeds in gases with different specific heat ratio. Zhurnal Tekhnicheskoi Fiz. 1961;31(4):495.
9. Charters AC, Thomas RN. The Aerodynamic Performance of Small Spheres from Subsonic to High Supersonic Velocities. J Aeronaut Sci [Internet]. 1945 Oct;12(4):468–76. Available from: https://arc.aiaa.org/doi/10.2514/8.11287
10. Kharchenko, N.A. and Kotov, M.A. (2018) “Analysis of the high speed gas flow over a sphere in the range of Mach numbers 2-12,” in *Journal of Physics: Conference Series*. Institute of Physics Publishing. Available at: https://doi.org/10.1088/1742-6596/1009/1/012007.
11. Bilbey, C.A. (2005) *AFIT Scholar AFIT Scholar Theses and Dissertations Student Graduate Works Investigation of the Performance Characteristics of Re-Entry Investigation of the Performance Characteristics of Re-Entry Vehicles Vehicles*. Available at: https://scholar.afit.edu/etd/3713.
12. Stappert, S. *et al.* (no date) *Mission Analysis and Preliminary Re-entry Trajectory Design of the DLR Reusability Flight Experiment ReFEx*.
13. Ahmad, A. *et al.* (no date) *PREDICTION OF THE AERODYNAMIC PERFORMANCE OF RE-USABLE SINGLE STAGE TO ORBIT VEHICLES*.
14. Cross, P.G. and West, M.R. (2019) *Simulation of Hypersonic Flowfields Using STAR-CCM+*.
15. Pekurovsky, D.E. *et al.* (2025) “Effect of Mesh Refinement Methods on Hypersonic Flow Quantities,” in *AIAA Science and Technology Forum and Exposition, AIAA SciTech Forum 2025*. American Institute of Aeronautics and Astronautics Inc, AIAA. Available at: https://doi.org/10.2514/6.2025-0147.
16. Carias, K.T. and Prince, S.A. (no date) *VALIDATION STUDY OF BLUNTED AND SHARP CONES AT HYPERSONIC SPEED*.
17. NASA (2024). *Space Shuttle - NASA*. [online] NASA.gov. Available at: https://www.nasa.gov/space-shuttle/.
18. Byers, M. *et al.* (2022) *Unnecessary risks created by uncontrolled rocket reentries*. Available at: www.amostech.com.
19. NASA (2014). *Re-Entry Aircraft*. [online] Nasa.gov. Available at: https://www.grc.nasa.gov/WWW/BGH/hihyper.html.
20. Mehta, R.C. and Rathakrishnan, E. (2023) “Computation of aerodynamic coefficients of a re-entry vehicle at Mach 6,” *Advances in Aircraft and Spacecraft Science*, 10(5), pp. 457–471. Available at: https://doi.org/10.12989/aas.2023.10.5.457.
21. Metsker, Y. *et al.* (no date) *Analysis of reentry vehicle flight dynamics*.
22. W. Malalasekera, An Introduction to Computational Fluid Dynamics, Edinburgh: Pearson Education Limited, 2007.
23. Van Dyke M. An Album of Fluid Motion. Stanford, CA: The Parabolic Press; 1982. 129 p
24. Engineering ToolBox (2003). *U.S. Standard Atmosphere*. [online] Engineeringtoolbox.com. Available at: https://www.engineeringtoolbox.com/standard-atmosphere-d_604.html.
25. Schlichting, H. and Gersten, K. (2016) *Boundary-Layer Theory*, *Boundary-Layer Theory*. Springer Berlin Heidelberg. Available at: https://doi.org/10.1007/978-3-662-52919-5.
26. Ledlow, T.W. (2014) *Integration of Grid Fins for the Optimal Design of Missile Systems*.

27. editor@optimage.co.uk (2025). *24TH ICAS CONGRESS 2004: Paper ICAS 2004-2.11.R*. [online] Icas.org. Available at: https://icas.org/icas_archive/ICAS2004/ABSTRACTS/092.HTM [Accessed 26 Apr. 2025].

28. PilotMall.com. (2025). *Vortex Generators: How They Improve Aerodynamics*. [online] Available at: https://www.pilotmall.com/blogs/news/vortex-generators-aerodynamics.

29. Deng, F. and Qin, N. (2021) "Vortex-Generating Shock Control Bumps for Robust Drag Reduction at Transonic Speeds," *AIAA Journal*, 59(10), pp. 3900–3909. Available at: https://doi.org/10.2514/1.J060528.

30. Singh, N.K. (2019) "Numerical simulation of flow behind vortex generators," *Journal of Applied Fluid Mechanics*, 12(4), pp. 1047–1061. Available at: https://doi.org/10.29252/jafm.12.04.29330.

31. Lögdberg, O. (2006) *Vortex generators and turbulent boundary layer separation control*.

8 Appendix

Appendix 1– Literature review

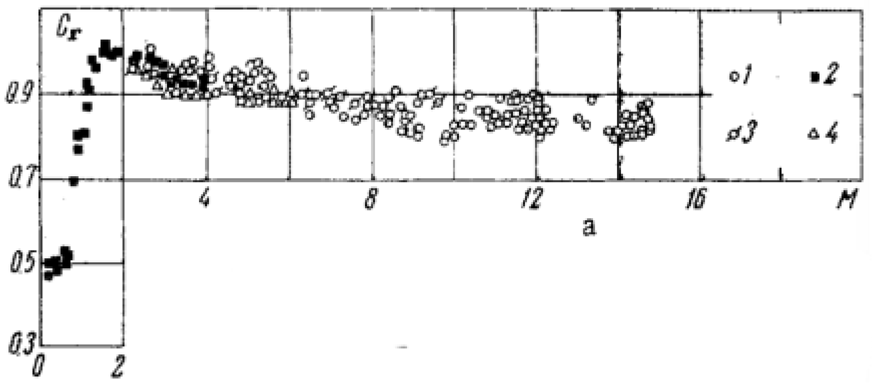


Figure 1 - Drag Coefficient of a sphere vs Mach number [6]

-

Mach Number (M)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Coefficient of Drag (C_D)	1.00	1.00	0.98	0.95	0.93	0.9	0.89	0.87	0.86	0.86	0.86	0.86	0.86	0.86	0.86

Table 1 - Tabulated Drag Coefficient, inferred from Figure 5 – [6]

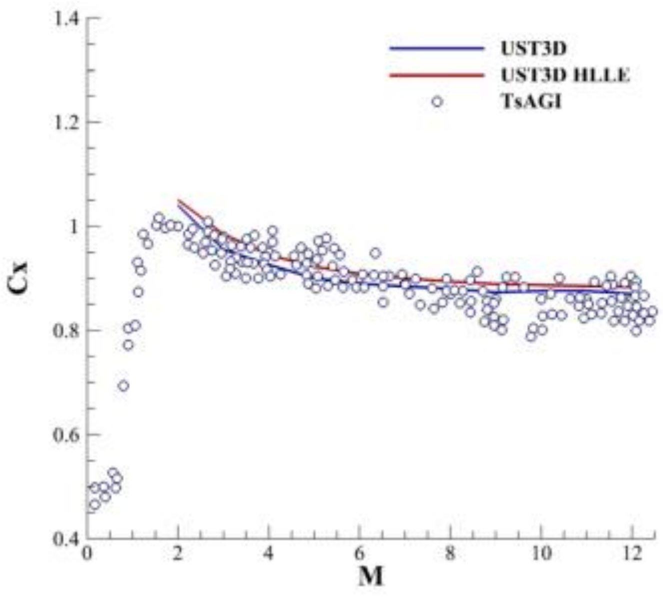


Figure 2 - Variation in drag coefficient of a sphere up to Mach 12 – [10]

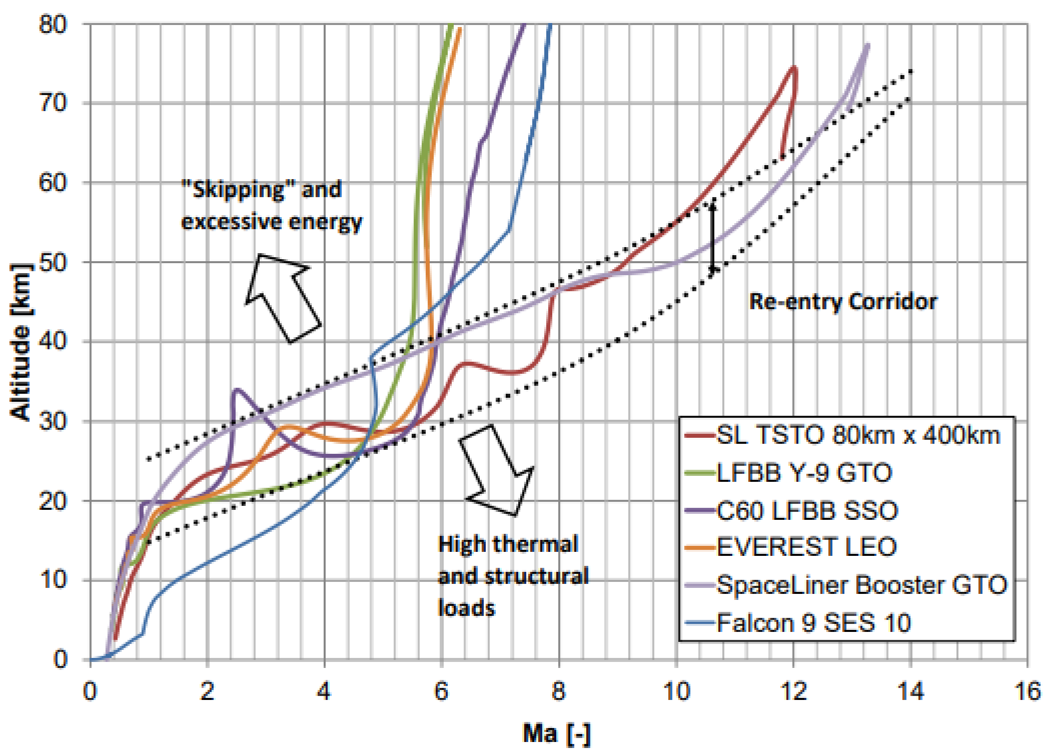


Figure 3 - Re-entry corridor and examples of re-entry trajectories - [12]

Appendix 2– Background research

Figure 4 - Nasa Space Shuttle - [17]

Appendix 3 – Simulation Setup Parameters

Group	**Model**
Space	**3D**
Time	**Steady**
Material	**Gas**
Flow	**Coupled Flow** **Gradients**
Equation of State	**Ideal Gas** **Coupled Energy (Automatically selected)**
Viscous Regime	**Turbulent** **Reynolds-Averaged Navier-Stokes (Automatically selected)**
Turbulence Models	
Spalart-Allmaras	**Spalart-Allmaras Turbulence** **Standard Spalart-Allmaras (Automatically selected)** **Wall Distance (Automatically selected)** **All y+ Wall Treatment (Automatically selected)**
K-Epsilon	**K-Epsilon Turbulence** **Realizable K-Epsilon Two-Layer** **Wall Distance (Automatically selected)** **Two-Layer All y+ Wall Treatment**
K-Omega	**K-Omega Turbulence** **SST (Menter) K-Omega** **Wall Distance (Automatically selected)** **All y+ Wall Treatment**

Table 3 - Physics Models for Sphere and Rocket Simulations

Node in STAR-CCM+	Value
Initial condition	
↓*Static temperature*	**288.15 K**
↓*Velocity (Mach 3)*	**[1020.87, 0.0, 0.0] m/s**
↓*Pressure*	**0.0 Pa**
Models → Gas→ Material properties	
↓*Dynamic viscosity*	**1.78972E-5 Pa-s**
↓*Molecular weight*	**28.9664 Kg/Kmo**
↓*Specific Heat*	**1003.62 J/Kg-K**
↓*Turbulent Prandtl Number*	**0.9**
↓*Thermal conductivity*	**0.0260305 W/m-K**

Table 4 – Initial Physics Conditions

Node in Star-CCM+	Property in Star-CCM+	Value
Base Size	Base Size	**0.001 m**
Target Surface Size	Percentage of Base	**150000.0**
Minimum Surface Size	Percentage of Base	**10.0**
Surface Regrowth Rate	Surface Growth Rate	**1.2**
Number of Prism Layers	Number of Prism Layers	**5**
Prism Layer Stretching	Prism Layer Stretching	**1.2**
Prism Layer Total Thickness	Percentage of Base	**10.0**
Volume Growth Rate	Volume Growth Rate	**1.2**
Maximum Tet Size	Percentage of Base	**10000.0**
Core Mesh Optimisation → Optimisation Cycles	Value	**5**
↓ *Quality Threshold*	Value	**0.8**
Surface Control (Region Boundary – Sphere Surface) → Target Surface Size	Percentage of Base	**10.0**
↓*Minimum Surface Size*	Percentage of Base	**5.0**
↓*Surface Growth Rate*	Surface Growth Rate	**1.5**
Volumetric Control (Part – Smaller Sphere of radius 1.5D) → Surface Remesher Customize Size → Custom Size	Percentage of Base	**20**
Volumetric Control 2 (Part – Medium Sphere of radius 2.25D) → Surface Remesher Customize Size → Custom Size	Percentage of Base	**40**
Volumetric Control 3 (Part – Large Sphere of radius 3D) → Surface Remesher Customize Size → Custom Size	Percentage of Base	**60**

Table 5 – Mesh Control Values for Mach 3-7

To ensure y^+ values from MATLAB calculator outlined in Appendix 6 can be entered in to STAR CCM+ ensure the following changes to the settings have been made:

In *'Operations →Automated Mesh →Meshers → Prism Layer Mesher'* , click on the drop down titled 'Distribution Mode' and change the setting to 'Wall Thickness'.

Node in Star-CCM+	Value
Prism Layer Controls (Geometry → Operations → Automated Mesh → Default Controls)	
→ *Number or Prism Layers*	**25**
→ *Prism Layer Near Wall Thickness*	**8.0345E-7**
→ *Prism Layer Reduction Percentage*	**50**

Node in Star-CCM+	**Property in Star-CCM+**	**Value**
Prism Layer Total Thickness (Geometry → Operations → Automated Mesh → Default Controls →Prism Layer Controls)		
→Absolute Size	**Absolute**	**1.278E-4**

Table 7 - Prism Layer Mesh Settings

Node in Star-CCM+	**Property in Star-CCM+**	**Value**
Wake Refinement Volumetric Control (Part –Wake Refinement Cuboid of length 4.5'Characteristic Length of Rocket)* *→ Surface Remesher Customize Size → Custom Size*	Percentage of Base	**20**
Grid Fin Refinement (Part – Grid Fin Refinement Cuboid, 5' Grid Fin volume'* *→ Surface Remesher Customize Size → Custom Size*	Percentage of Base	**1**

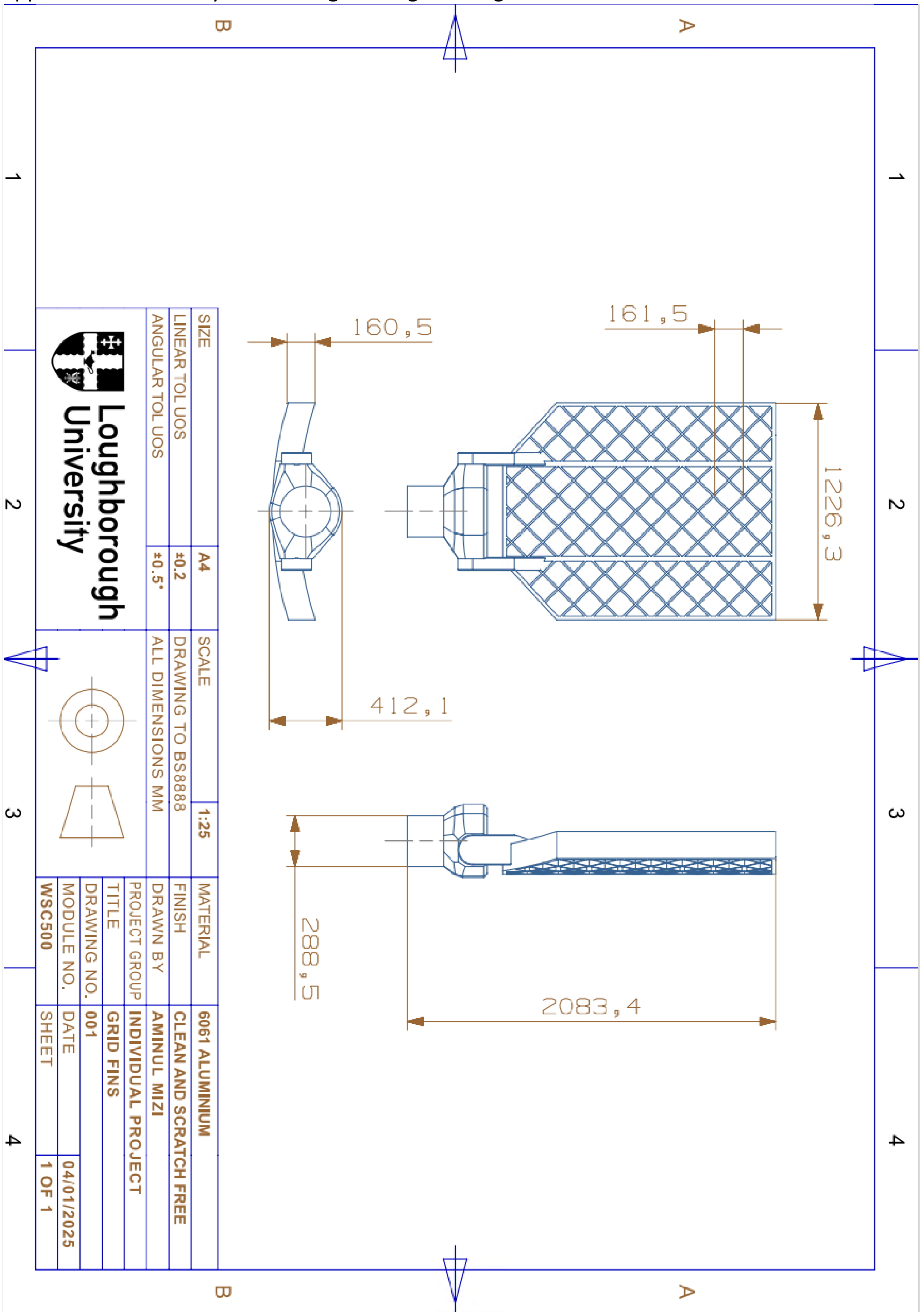
Table 8 - Additional Volumetric Control Settings for Rocket

Node in Star-CCM+	**Property in Star-CCM+**	**Value**
Coupled Flow (Physics – Models) → Discretisation	Type	**MUSCL 3rd-order/CD**
Coupled Implicit (Solvers – Coupled Implicit) → CFL Control Method	Method	**Expert Driver**
→Explicit Relaxation Method	Type	**Constant**
Expert Driver CFL (Solvers – Coupled Implicit) → Target CFL	*Value*	**1.0**
→ Initial CFL	*Value*	**0.1**
→ Ramp: Start Iteration	*Value*	**100**
→ Ramp: End Iteration	*Value*	**500**
→ CFL Recovery Rate	*Value*	**1.1**
→ Target AMG Cycles	*Value*	**4**
Constant Relaxation (Solvers – Coupled Implicit) → Explicit Relaxation	*Value*	**0.3**
AMG Linear Solver (Solvers – Coupled Implicit) → Cycle Type	*Type*	**F Cycle**
F Cycle (Solvers – Coupled Implicit – AMG Linear Solver) → Pre-Sweeps	*Value*	**0**
→ Post-Sweeps	*Value*	**3**
→ Max Levels	*Value*	**2**

Table 9 – Solver settings for all Rocket simulations

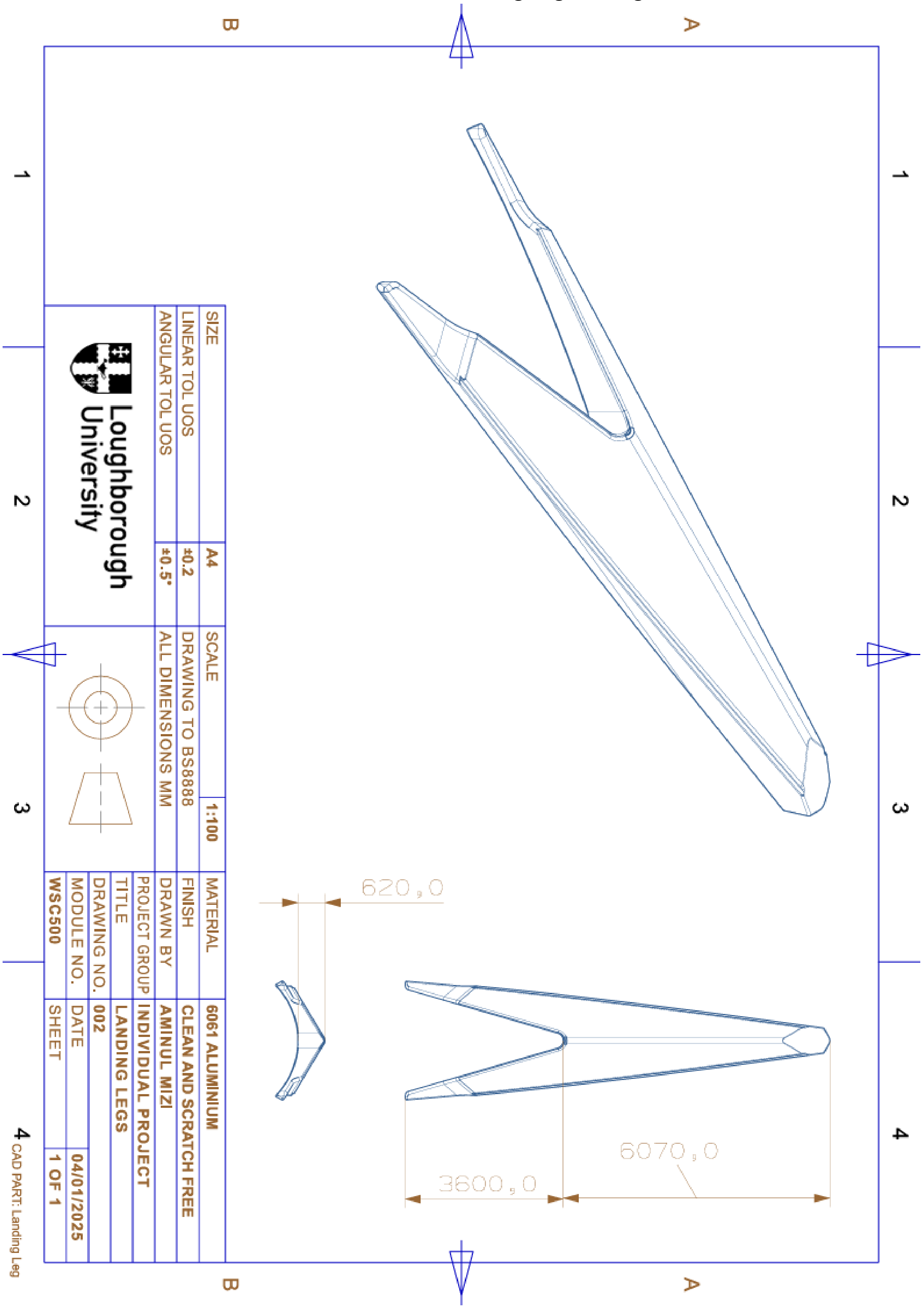
Node in Star-CCM+	**Property in Star-CCM+**	**Value**
Expert Initialisation (Solvers – Coupled Implicit) *→ Method*	Method	**Grid Sequencing**
Grid Sequencing (Solvers – Coupled Implicit – Expert Initialisation) *→ Maximum Grid Levels*	Value	**10**
→ Maximum Iterations per Level	Value	**50**
→ Convergence Tolerance per Level	*Value*	**0.05**
→ CFL Number	*Value*	**1.0**
→ Explicit Relaxation	*Value*	**0.5**

Table 10 - Grid sequencing initialisation properties used in all Rocket simulations



SIZE	A4	SCALE	1:25
LINEAR TOL UOS	±0.2	DRAWING TO BS8888	
ANGULAR TOL UOS	±0.5°	ALL DIMENSIONS MM	
MATERIAL	6061 ALUMINIUM		
FINISH	CLEAN AND SCRATCH FREE		
DRAWN BY	AMINUL MIZI		
PROJECT GROUP	INDIVIDUAL PROJECT		
TITLE	GRID FINS		
DRAWING NO.	001		
MODULE NO.		DATE	04/01/2025
WSC500		SHEET	1 OF 1

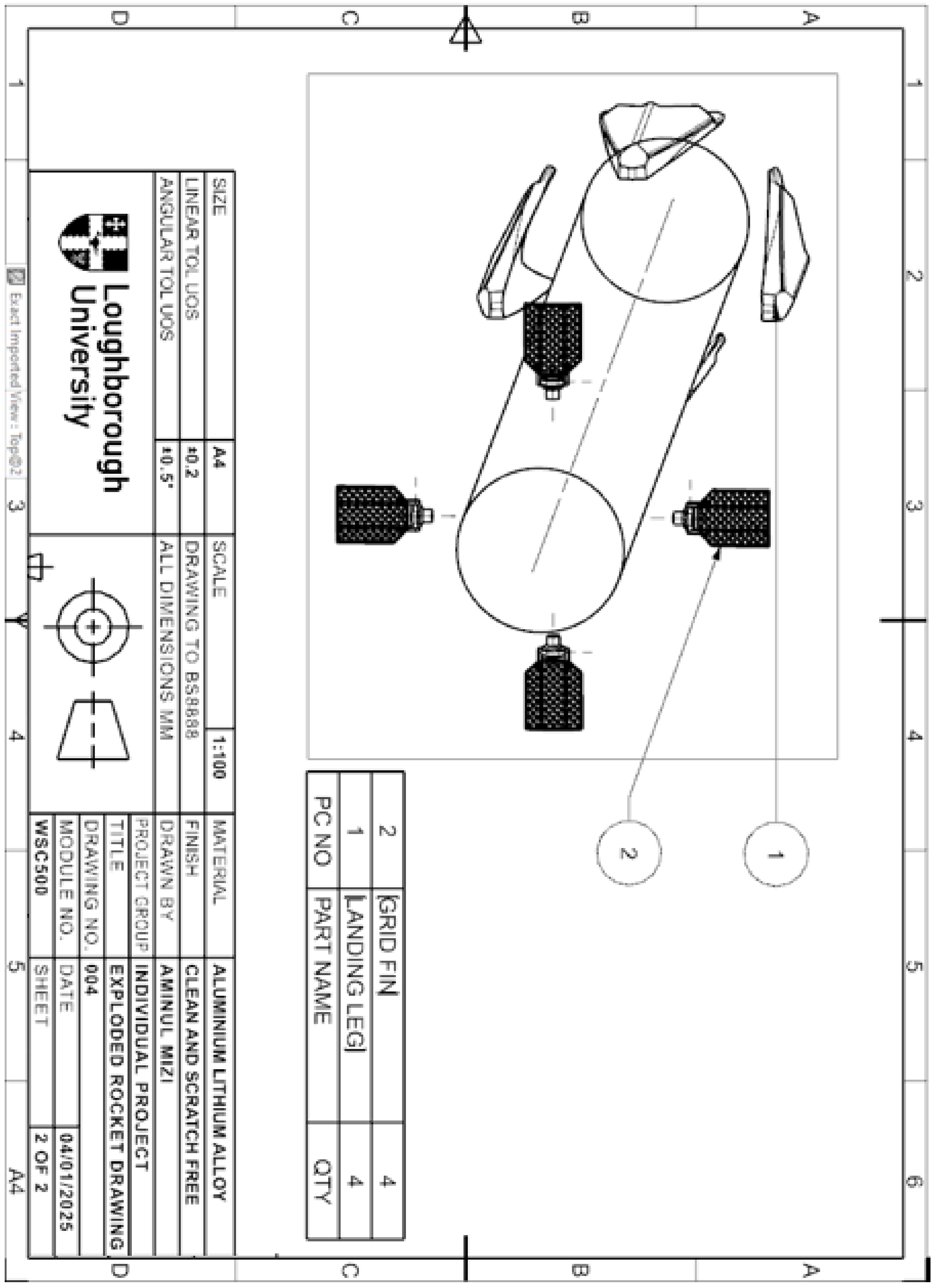
Grid Fin



SIZE	A4
LINEAR TOL UOS	±0.2
ANGULAR TOL UOS	±0.5°
SCALE	1:100
DRAWING TO BS8888	
ALL DIMENSIONS MM	
MATERIAL	6061 ALUMINIUM
FINISH	CLEAN AND SCRATCH FREE
DRAWN BY	AMINUL MIZI
PROJECT GROUP	INDIVIDUAL PROJECT
TITLE	LANDING LEGS
DRAWING NO.	002
MODULE NO.	WSC500
DATE	04/01/2025
SHEET	1 OF 1

Loughborough University

Landing Legs



PC NO	PART NAME	QTY
2	GRID FIN	4
1	LANDING LEG	4

SIZE	A4	SCALE	1:100	MATERIAL	ALUMINIUM LITHIUM ALLOY
LINEAR TOL UOS	±0.2	DRAWING TO BS8888		FINISH	CLEAN AND SCRATCH FREE
ANGULAR TOL UOS	±0.5°	ALL DIMENSIONS MM		DRAWN BY	AMINUL MIZI
				PROJECT GROUP	INDIVIDUAL PROJECT
				TITLE	EXPLODED ROCKET DRAWING
				DRAWING NO.	004
				MODULE NO.	DATE 04/01/2025
				WSC500	SHEET 2 OF 2

Exploded Drawing

Appendix 5 – Y^+ wall MATLAB calculator

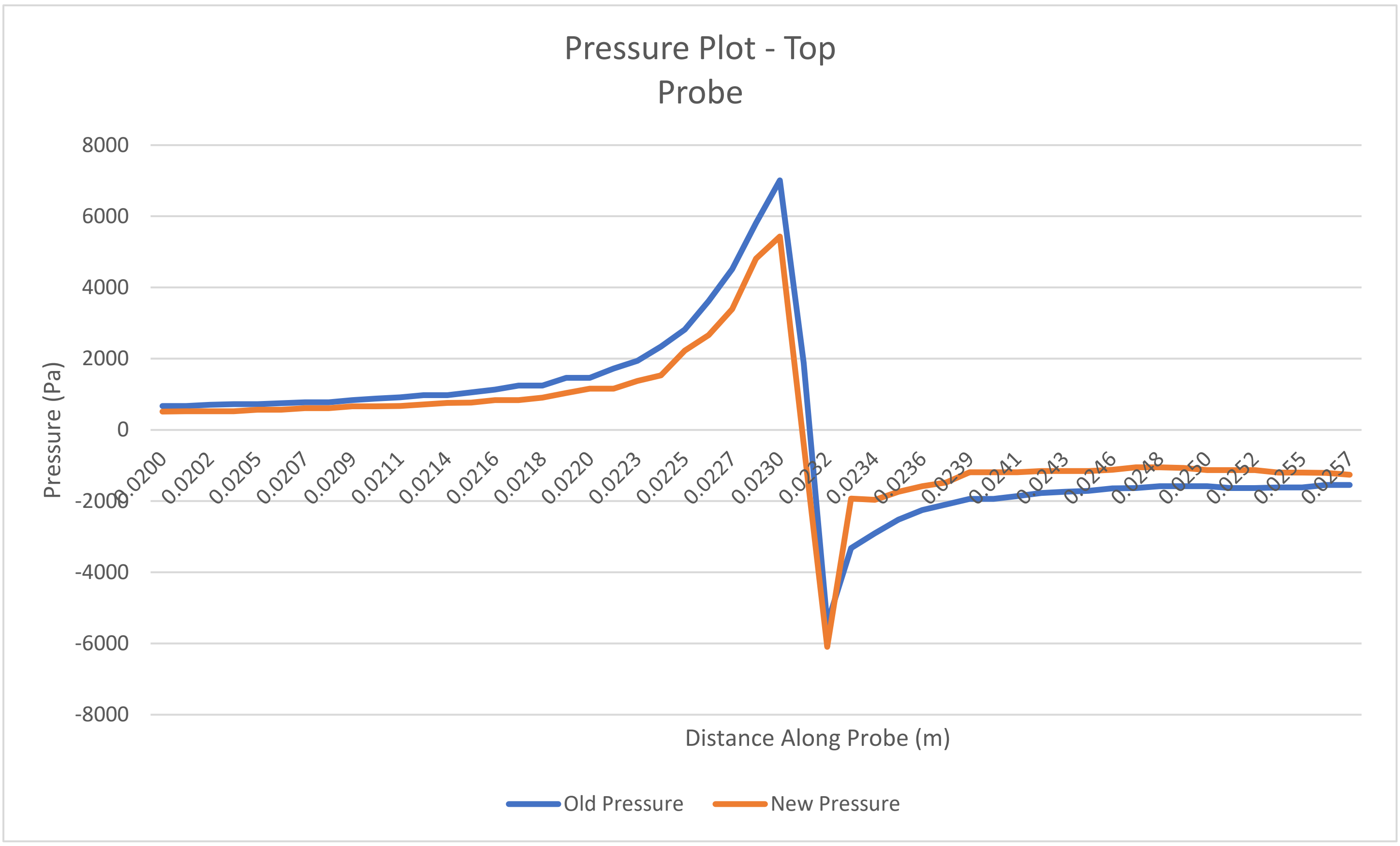
```
function starCCM_PrismCalculator
% STAR-CCM+ Prism Layer Configuration Tool
% Calculates first cell height, prism layer parameters, and boundary layer estimates

    % Input parameters
    p = input('Enter fluid density (kg/m³): ');
    mu = input('Enter dynamic viscosity (Pa·s): ');
    U_inf = input('Enter freestream velocity (m/s): ');
    L = input('Enter characteristic length (m): ');
    targetYplus = input('Enter target Y+ value: ');
    Gr = input('Enter growth ratio (recommended intial value of 1.2): ');
    Nl = input('Enter number of layers (recommended intial value of 25): ');

    % Reynolds number calculation
    Re = p * U_inf * L / mu;

    % Skin friction coefficient (Schlichting for turbulent flow)
    Cf = (2*log10(Re) - 0.65).^(-2.3);  % Valid for Re < 1e9 [4]

    % Friction velocity calculation
    u_tau = U_inf * sqrt(Cf/2);

    % First cell height calculation [6][8]
    y1 = (targetYplus * mu) / (p * u_tau);

    % Boundary layer thickness estimation (flat plate turbulent) [7]
    delta = 0.37 * L / (Re^(1/5));

    % Prism layer recommendations
    num_layers = Nl;  % Recommended starting point [2][6]
    growth_ratio = Gr;  % Conservative growth rate [2][6]

    % Total prism thickness calculation
    total_thickness = y1 * (growth_ratio^num_layers - 1)/(growth_ratio - 1);

    % STAR-CCM+ specific validation
    min_wall_distance = 1e-9;  % Default reference value [6]
    if y1 < min_wall_distance
        warning(['First cell height below minimum wall distance! '...
            'Adjust Physics > Reference Values > Minimum Allowable Wall Distance']);
    end


    % Output display
    fprintf('\n--- STAR-CCM+ Prism Layer Parameters ---\n');
    fprintf('Reynolds Number: %.4f m\n\n', Re);
    fprintf('First Cell Height (Near Wall Thickness): %.10e m\n', y1);
    fprintf('Recommended Layers: %d (25-30 optimal)\n', num_layers);
    fprintf('Growth Ratio: %.2f (≤1.3 recommended)\n', growth_ratio);
    fprintf('Estimated BL Thickness: %.4f m\n', delta);
    fprintf('Prism Layer Total Thickness: %.10f m\n\n', total_thickness);



    % Validation checks
    if total_thickness < delta
        warning('Total prism thickness < boundary layer - increase layers/growth 
ratio');
    end

    if growth_ratio > 1.3
        warning('Growth ratio exceeds recommended maximum - reduce for mesh quality');
    end
end
```

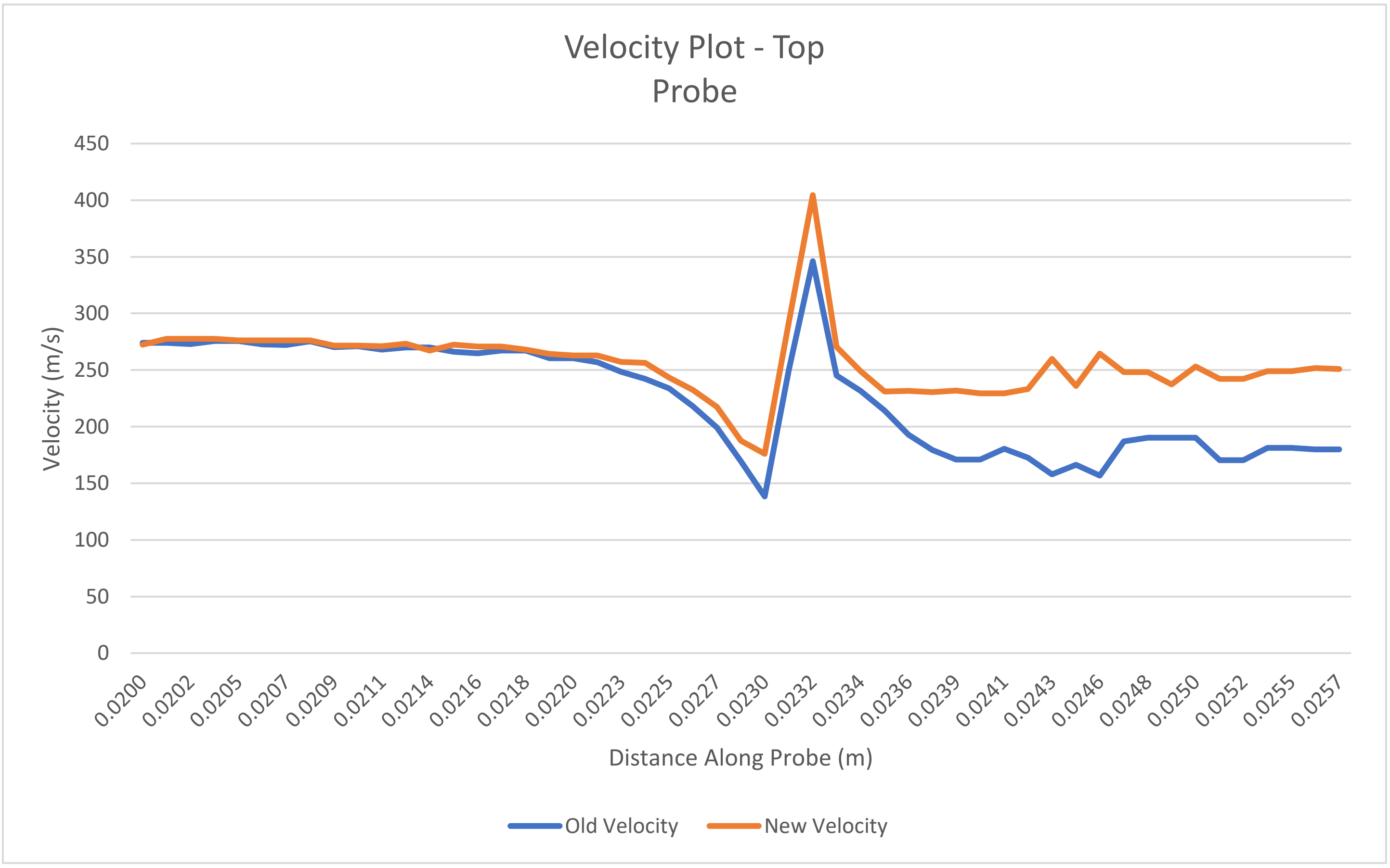
The script will return values for 'Prism Layer Near Wall Thickness' and 'Prism Layer Total Thickness' after user inputs have been specified. First ensure that To ensure y^+ values from MATLAB calculator outlined in Appendix 6 can be entered in to STAR CCM+ ensure the following changes to the settings have been made:

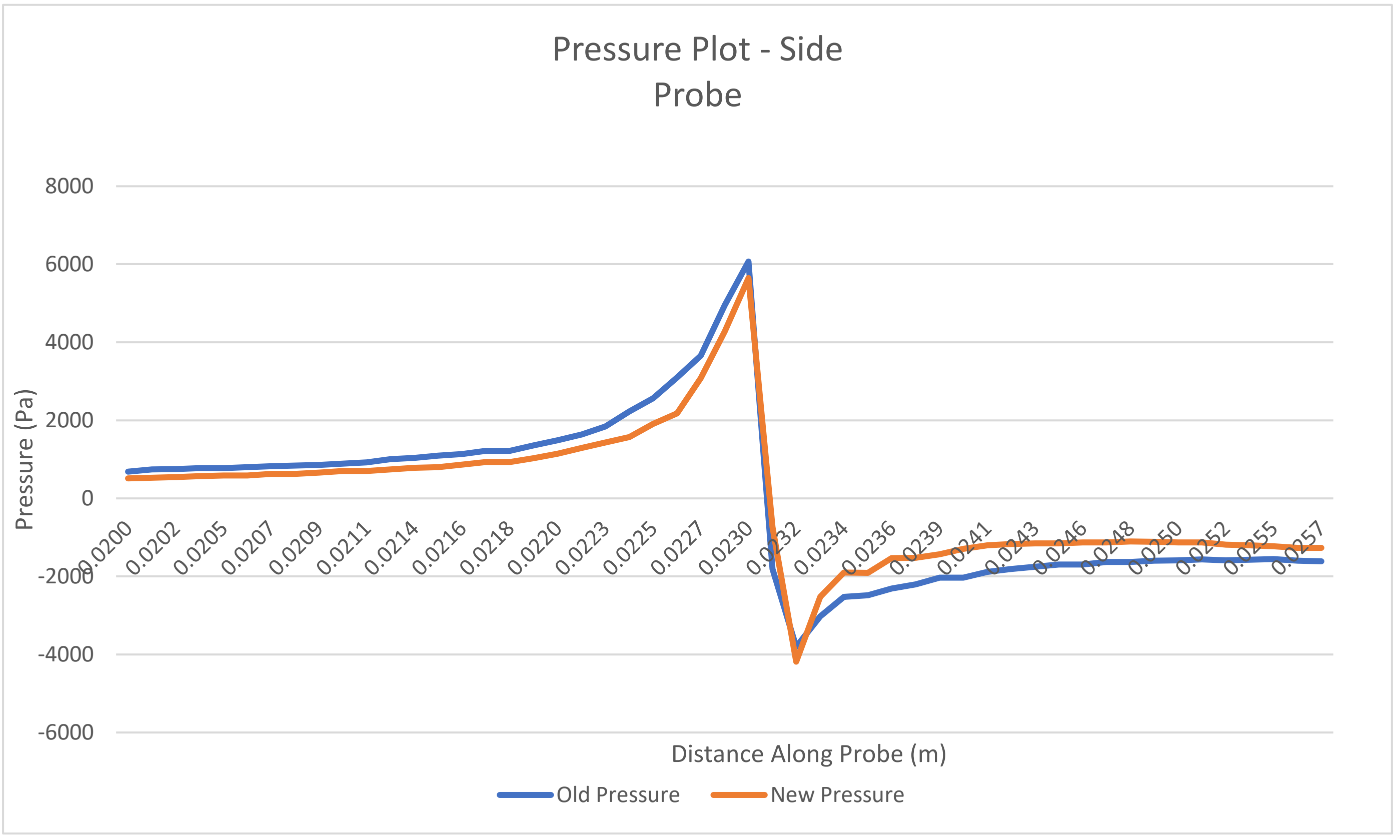
In *'Operations →Automated Mesh →Meshers → Prism Layer Mesher'* , click on the drop down titled 'Distribution Mode' and change the setting to 'Wall Thickness'.

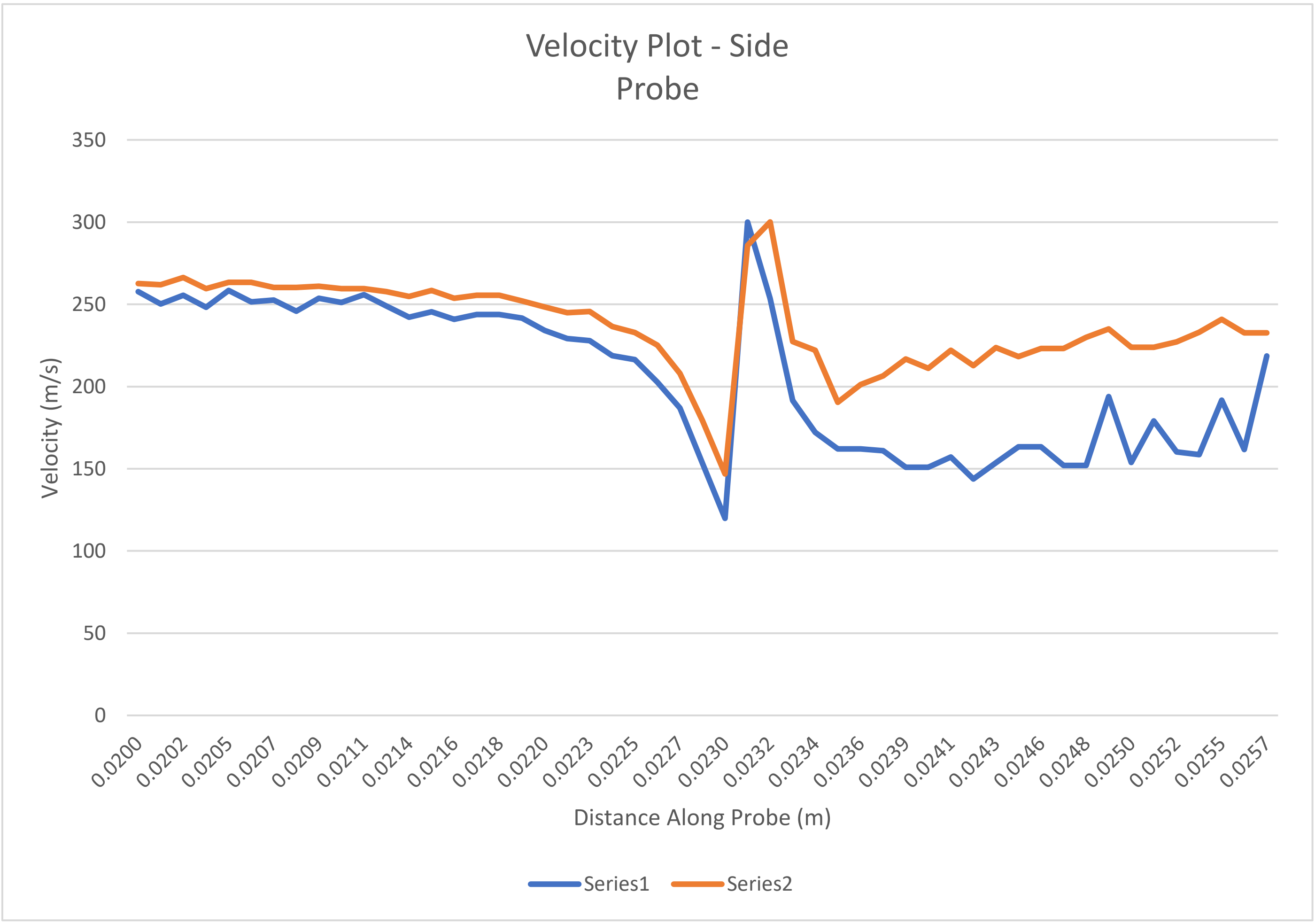
Now values from the MATLAB calculator can be added in to 'Prism Layer Controls' and in its drop down 'Prism Layer Total Thickness)

Appendix 6 – Line probe data sets






Pressure Plot - Side Probe
8000
6000
4000
2000
0
-2000
-4000
-6000
Pressure (Pa)
0.0200
0.0202
0.0205
0.0207
0.0209
0.0211
0.0214
0.0216
0.0218
0.0220
0.0223
0.0225
0.0227
0.0230
0.0232
0.0234
0.0236
0.0239
0.0241
0.0243
0.0246
0.0248
0.0250
0.0252
0.0255
0.0257
Distance Along Probe (m)
Old Pressure
New Pressure


Velocity Plot - Side Probe
350
300
250
200
150
100
50
0
Velocity (m/s)
0.0200
0.0202
0.0205
0.0207
0.0209
0.0211
0.0214
0.0216
0.0218
0.0220
0.0223
0.0225
0.0227
0.0230
0.0232
0.0234
0.0236
0.0239
0.0241
0.0243
0.0246
0.0248
0.0250
0.0252
0.0255
0.0257
Distance Along Probe (m)
Series1
Series2

Appendix 7 – Python 3DOF Trajectory Code

```python
import math

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import argparse

from scipy.interpolate import interp1d


# ==============================================================================

# IN VACUO TEST - TOGGLE DO_INVACUO_TEST TO 'TRUE', ENSURE DRAG IS SET TO 0

# ==============================================================================

DO_INVACUO_TEST = False


def analytic_ballistic(z0, V0, gamma_deg, g0=9.80665, n_steps=None, t_numeric=None):

    """

    Computes the pure-vacuum trajectory under constant gravity.

    If t_numeric is given, returns values at exactly those times; otherwise,

    returns n_steps points between t=0 and impact.

    """

    gamma = math.radians(gamma_deg)

    vx0, vz0 = V0 * math.cos(gamma), V0 * math.sin(gamma)

    # time-of-flight until z=0:

    T = ( vz0 + math.sqrt(vz0**2 + 2 * g0 * z0) ) / g0


    if t_numeric is not None:

        t = t_numeric

    else:

        t = np.linspace(0, T, n_steps if n_steps else 500)


    x = vx0 * t

    z = z0 + vz0 * t - 0.5 * g0 * t**2

    return t, x, z


# ==============================================================================

# CONFIGURATION / PARAMETERS

# ==============================================================================

# These are the default parameters. You can override them using command-line args.

DT = 1                              # Time step in seconds

T_FINAL = 300                       # Maximum simulation time in seconds


# Vehicle parameters
```

```
MASS = 25000                    # Mass in kg

DIAMETER = 3.66                 # Diameter in meters

AREA = math.pi * (DIAMETER / 2)**2  # Reference area in m²


# Initial conditions (altitude in meters, speed in m/s, flight path angle in degrees)

INITIAL_ALTITUDE = 70000        # in meters

REENTRY_SPEED = 2500            # in m/s

FLIGHT_PATH_ANGLE_DEG = -30     # negative for descent


# Aero data CSV file (should be in the same folder)

AERO_CSV_FILE = 'Falcon9_aero_data1.csv'

# Optionally, output the trajectory data to a file

OUTPUT_TRAJECTORY_CSV = 'trajectory_data.csv'


SHOW_DEBUG_PRINTS = False

# ===============================================================================

# DEBUG PRINT FUNCTION

# ===============================================================================

def debug_print(*args):

    if SHOW_DEBUG_PRINTS:

        print("DEBUG:", *args)


# ===============================================================================

# ATMOSPHERIC PROPERTIES

# ===============================================================================

def atmospheric_properties(altitude):

    """

    Calculate atmospheric properties at a given altitude.

    Returns:

        rho: Density [kg/m^3]

        a: Speed of sound [m/s]

    """

    R_specific = 0.287053  # [kJ/(kg·K)]

    gamma = 1.4


    if altitude <= 11000:

        T = 288.15 - 0.0065 * altitude

        P = 101.325 * ((T / 288.15) ** 5.256)

        debug_print(f"Troposphere: alt={altitude}, T={T:.2f}K, P={P:.2f}kPa")

    elif altitude <= 25000:

        T = 216.65

        P = 22.65 * math.exp(1.73 - 0.000157 * altitude)
        debug_print(f"Lower Stratosphere: alt={altitude}, T={T:.2f}K, P={P:.2f}kPa")
    else:
        T = 141.94 + 0.00299 * altitude
        P = 2.488 * ((T / 216.65) ** -11.388)
        debug_print(f"Upper Atmosphere: alt={altitude}, T={T:.2f}K, P={P:.2f}kPa")


    rho = P / (R_specific * T)
    a = math.sqrt(gamma * 1000 * R_specific * T)
    debug_print(f"At alt={altitude}: rho={rho:.4f}, a={a:.2f} m/s")
    return rho, a
# ===============================================================================
# GRAVITY FUNCTION
# ===============================================================================
def gravity_at_altitude(altitude):
    g0 = 9.80665 #m/s^2
    Er = 6371000 # Earths Mean Radious in meters
    return g0 * (Er / (Er + altitude))**2
# ===============================================================================
# PIECEWISE AERO PROPERTIES LOOKUP
# ===============================================================================
def get_piecewise_aero_properties(altitude, aero_df):
    """
    Return the piecewise constant aero properties (MACH, AoA, CD) based on the current 
altitude.
    The CSV rows are milestones (e.g., 70000, 55000, etc.), and the values remain in 
effect
    until the vehicle reaches the next milestone.
    """
    aero_sorted = aero_df.sort_values(by='ALT', ascending=False).reset_index(drop=True)
    debug_print("Aero milestones (sorted):", aero_sorted['ALT'].values)


    chosen_row = aero_sorted.iloc[0]
    for idx, row in aero_sorted.iterrows():
        if altitude <= row['ALT']:
            chosen_row = row
            debug_print(f"At alt={altitude}, using milestone {row['ALT']}: 
MACH={row['MACH']}, AoA={row['AoA']}, CD={row['CD']}")
        else:
            break
    return float(chosen_row['MACH']), float(chosen_row['AoA']), float(chosen_row['CD'])


def get_drag_coefficient(velocity, altitude, aero_df):
    """
    Return the drag coefficient (Cd) using a piecewise lookup from the aero data.
    """
    mach, aoa, Cd = get_piecewise_aero_properties(altitude, aero_df)
    debug_print(f"At alt={altitude}, piecewise values: MACH={mach}, AoA={aoa},
Cd={Cd}")
    return Cd
# ===============================================================================
# INTERPOLATION FUNCTION
# ===============================================================================
def get_interpolated_drag_coefficient(altitude, aero_df):
    """
    Return the drag coefficient (Cd) using linear interpolation from the aero data.
    """
    # Sort data by altitude
    aero_sorted = aero_df.sort_values(by='ALT')
    altitudes = aero_sorted['ALT'].values
    cds = aero_sorted['CD'].values

    # Create the interpolator
    cd_interp = interp1d(altitudes, cds, kind='linear', fill_value='extrapolate')
    Cd = float(cd_interp(altitude))
    debug_print(f"At alt={altitude:.1f} m, interpolated Cd={Cd:.4f}")
    return Cd
# ===============================================================================
# RUNGE-KUTTA 4th ORDER STEP
# ===============================================================================
def rk4_step(state, dt, derivs, *args):
    k1 = derivs(state, *args)
    debug_print("k1:", k1)
    k2 = derivs(state + 0.5 * dt * k1, *args)
    debug_print("k2:", k2)
    k3 = derivs(state + 0.5 * dt * k2, *args)
    debug_print("k3:", k3)
    k4 = derivs(state + dt * k3, *args)
    debug_print("k4:", k4)
    state_next = state + dt * (k1 + 2*k2 + 2*k3 + k4) / 6
    debug_print("New state after RK4 step:", state_next)
    return state_next

# ===============================================================================
# DERIVATIVES (3DOF)
# ===============================================================================
def derivs(state, mass, area, aero_df):
    x, y, z, vx, vy, vz = state
    velocity = math.sqrt(vx**2 + vy**2 + vz**2)
    debug_print(f"At state={state}, speed={velocity:.2f} m/s")

    g = gravity_at_altitude(z)
    gravity = np.array([0, 0, -g])

    rho, _ = atmospheric_properties(z)
    debug_print(f"At alt={z}: rho={rho:.4f}")

    Cd = get_interpolated_drag_coefficient(z, aero_df)
    debug_print(f"Cd={Cd}")

    F_drag = 0.5 * rho * velocity**2 * Cd * area
    debug_print(f"F_drag={F_drag:.2f} N")

    a_drag = np.array([0, 0, 0])
    if velocity > 0:
        a_drag = -F_drag / mass * np.array([vx, vy, vz]) / velocity
    acceleration = gravity + a_drag
    debug_print(f"a_drag={a_drag}, acceleration={acceleration}")

    return np.array([vx, vy, vz, acceleration[0], acceleration[1], acceleration[2]])

# ==============================================================================
# SIMULATION
# ==============================================================================
def simulate_trajectory(initial_state, mass, area, aero_df, dt, t_final):
    t = 0.0
    state = initial_state.copy()
    t_arr = [t]
    state_arr = [state.copy()]

    while state[2] > 0 and t < t_final:
        speed = math.sqrt(state[3]**2 + state[4]**2 + state[5]**2)
        debug_print(f"Time={t:.1f}s, state={state}, speed={speed:.2f} m/s")
        state = rk4_step(state, dt, derivs, mass, area, aero_df)
        t += dt

        if np.isnan(state).any():
            print(f"ERROR: NaN found at time={t:.1f}s.")
            break

        t_arr.append(t)
        state_arr.append(state.copy())

    return np.array(t_arr), np.array(state_arr)

def interpolate_to_zero_altitude(t_arr, state_arr):
    # Find the last index where altitude is positive
    for i in range(len(state_arr)-2, -1, -1):
        if state_arr[i][2] > 0 and state_arr[i+1][2] <= 0:
            break
    t1, t2 = t_arr[i], t_arr[i+1]
    s1, s2 = state_arr[i], state_arr[i+1]
    z1, z2 = s1[2], s2[2]
    alpha = z1 / (z1 - z2)
    t0 = t1 + alpha * (t2 - t1)
    state0 = s1 + alpha * (s2 - s1)
    return t0, state0, i
# ==============================================================================
# MAIN SETUP
# ==============================================================================
def main():
    global DEBUG  # Allow modification from CLI arguments if needed

    # Load aero data from CSV file
    aero_data = pd.read_csv(AERO_CSV_FILE)
    debug_print("Loaded CSV data:\n", aero_data)

    # Convert flight path angle to radians
    flight_path_angle = math.radians(FLIGHT_PATH_ANGLE_DEG)

    # Calculate initial velocities
    vx0 = REENTRY_SPEED * math.cos(flight_path_angle)
    vz0 = REENTRY_SPEED * math.sin(flight_path_angle)

    # Set initial state: [x, y, altitude, vx, vy, vz]
    initial_state = np.array([0.0, 0.0, INITIAL_ALTITUDE, vx0, 0.0, vz0])
    debug_print("Initial state:", initial_state)
    debug_print(f"Mass={MASS}, Diameter={DIAMETER}, Area={AREA:.2f}")

    # Run the simulation
    time_arr, state_arr = simulate_trajectory(initial_state, MASS, AREA, aero_data, DT,
T_FINAL)


# INVACUO ANALYSIS - ANALYTICAL
    if DO_INVACUO_TEST:
    # Extract numerical trajectory
        x_num = state_arr[:, 0]
        z_num = state_arr[:, 2]

        # === Analytic vacuum solution ===
        t_ana, x_ana, z_ana = analytic_ballistic(
        z0=INITIAL_ALTITUDE,
        V0=REENTRY_SPEED,
        gamma_deg=FLIGHT_PATH_ANGLE_DEG,
        t_numeric=time_arr
        )

        # === Error calculation ===
        dx = x_num - x_ana
        dz = z_num - z_ana

        print("\n====== Trajectory Comparison (Vacuum) ======")
        print(f"Max downrange error: {np.max(np.abs(dx)):.3f} m")
        print(f"Max vertical   error: {np.max(np.abs(dz)):.3f} m")

        # === Time of flight comparison ===
        T_num = time_arr[-1]
        g0 = 9.80665
        gamma = math.radians(FLIGHT_PATH_ANGLE_DEG)
        vz0 = REENTRY_SPEED * math.sin(gamma)
        T_ana = (vz0 + math.sqrt(vz0**2 + 2 * g0 * INITIAL_ALTITUDE)) / g0

        print(f"\nAnalytic time of flight: {T_ana:.4f} s")
        print(f"Numeric  time of flight: {T_num:.4f} s")
        print(f"Landing time error     : {abs(T_num - T_ana):.4f} s")

        # === Plot: Altitude vs Downrange ===
        plt.figure(figsize=(10, 6))
        plt.plot(x_ana, z_ana, 'k--', label='Analytic (vacuum)')
        plt.plot(x_num, z_num, 'b-',  label='Numeric (RK4, Cd=0)')
        plt.xlabel("Downrange Distance [m]")
        plt.ylabel("Altitude [m]")
        plt.title("Vacuum Trajectory: Analytic vs. RK4")
        plt.grid(True)
        plt.legend()
        plt.show()

        # === Plot 2: Altitude vs Time ===
        plt.figure(figsize=(10, 6))
        plt.plot(time_arr, z_ana, label='Analytic Altitude')
        plt.plot(time_arr, z_num, label='Numeric Altitude')
        plt.xlabel("Time [s]")
        plt.ylabel("Altitude [m]")
        plt.title("Altitude vs Time (Vacuum Comparison)")
        plt.grid(True)
        plt.legend()
        plt.show()

    # Interpolate if the last altitude is below zero
    if state_arr[-1][2] < 0:
        t0, state0, idx = interpolate_to_zero_altitude(time_arr, state_arr)
        # Replace last entry with interpolated zero-altitude state
        time_arr = np.concatenate([time_arr[:idx+1], [t0]])
        state_arr = np.concatenate([state_arr[:idx+1], [state0]])

    # Plotting results
    x_arr = state_arr[:, 0]
    y_arr = state_arr[:, 1]
    z_arr = state_arr[:, 2]
    vx_arr = state_arr[:, 3]
    vy_arr = state_arr[:, 4]
    vz_arr = state_arr[:, 5]
    speed_arr = np.sqrt(vx_arr**2 + vy_arr**2 + vz_arr**2)

    plt.figure(figsize=(10, 6))
    plt.plot(x_arr, z_arr, 'b-', label='Trajectory')
    plt.xlabel("Downrange Distance [m]")
    plt.ylabel("Altitude [m]")
    plt.title("Re-entry Trajectory (3DOF)")
    plt.grid(True)
    plt.legend()

    plt.figure(figsize=(10, 6))
    plt.plot(time_arr, speed_arr, 'r-', label='Speed')
    plt.xlabel("Time [s]")
    plt.ylabel("Speed [m/s]")
    plt.title("Vehicle Speed vs. Time")
    plt.grid(True)
    plt.legend()

    from mpl_toolkits.mplot3d import Axes3D
    fig = plt.figure(figsize=(10, 8))
    ax = fig.add_subplot(111, projection='3d')
    ax.plot(x_arr, y_arr, z_arr, 'g-', label='3D Trajectory')
    ax.set_xlabel("Downrange Distance [m]")
    ax.set_ylabel("Crossrange Distance [m]")
    ax.set_zlabel("Altitude [m]")
    ax.set_title("3D Trajectory of Re-entry Vehicle")
    plt.legend()
    plt.show()



    # Save trajectory data to a CSV file and print to console
    trajectory_data = pd.DataFrame({
        'Time [s]': time_arr,
        'x [m]': x_arr,
        'y [m]': y_arr,
        'Altitude [m]': z_arr,
        'vx [m/s]': vx_arr,
        'vy [m/s]': vy_arr,
        'vz [m/s]': vz_arr,
        'Speed [m/s]': speed_arr
    })
    trajectory_data.to_csv(OUTPUT_TRAJECTORY_CSV, index=False)
    print("Trajectory data saved to:", OUTPUT_TRAJECTORY_CSV)
    print(trajectory_data)

if __name__ == '__main__':
    # Use argparse to allow command-line options (like enabling debug prints)
    parser = argparse.ArgumentParser(description="Re-entry Trajectory Simulation")
    parser.add_argument('--debug', action='store_true', help='Enable debug prints')
    args = parser.parse_args()
    if args.debug:
        DEBUG = True
    main()
```