CSE6730 Final Project

JET ENGINE FLOW SIMULATION

December 6, 2023

Team 24:
Sijian Tan,
Kaiqun Peng,
Cheng Zhang
Github Repository

CSE6730 Project: Jet Engine Flow Simulation Final Report

Team 24: Sijian Tan, Kaiqun Peng, Cheng Zhang

Abstract

This project conducts a simulation of supersonic flow through a 2D scramjet engine inlet, utilizing an adaptive, first-order finite-volume method implemented in Python. Emphasizing total pressure recovery calculation at the engine inlet's end, the project also offers insightful visualizations of Mach and total pressure contours. The study aims to advance supersonic inlet aerodynamics and integrate computational science and engineering (CSE) to solve complex aerospace challenges. Our implementation is available on the course GitHub repository.

1 Background

1.1 Aerospace Engineering

Jet engines are widely used in today's aerospace industry. They generate immense thrust, ensuring sufficient support for aircraft to sustain flight in the sky. The simulation of airflow through jet engines perpetually captivates the aerospace industry, with the modeling of supersonic jet engine inlet flow presenting notable challenges. The supersonic inlet serves the crucial function of transitioning external supersonic flow to subsonic flow prior to its entry into the combustion section [1]. Due to the inherent properties of supersonic flow, shock waves are generated during ingress into the engine, inducing substantial alterations in pressure and temperature. These characteristics notably elevate the complexity and challenges of the issue.

An essential metric for assessing the performance of a supersonic inlet is the inlet pressure recovery ratio. This is computed by determining the ratio of the total pressure at the compressor entrance to the free stream total pressure [2]. Numerous studies have been conducted in the realm of supersonic inlet flow simulation. For instance, B. Alekhya employed the ANSYS CFX tool to execute simulations of 2D cone and ramp inlets, conducting a performance analysis of the supersonic inlet that encompassed a discussion on inlet pressure recovery [3]. Similarly, N. Om Prakash Raj also conducted a numerical evaluation on the performance of scramjet inlet using the commercial software ANSYS Fluent [4].

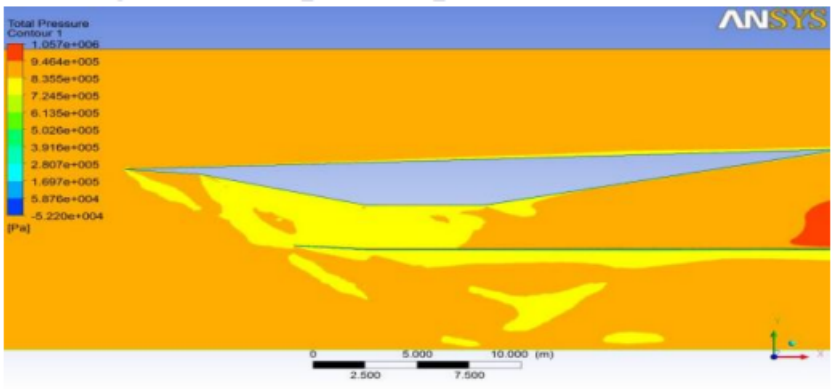

Figure 1: Ramp inlet total pressure contour [3]

1.2 Computational Science and Engineering

Supersonic flow simulation through two-dimensional scramjet engines has been a topic of interest within the computational fluid dynamics (CFD) community. Various methodologies, including the finite volume method (FVM), have been explored to address the challenges associated with accurately capturing shock and expansion wave structures in supersonic flows. Our approach employs the first-order, adaptive FVM, drawing inspiration from previous studies on the simulation of a supersonic laminar flow [5]. The FVM, a domain discretization method, divides the flow domain into numerous control volumes, applying the conservation laws of mass, momentum, and energy on each volume [6]. By solving the two-dimensional Euler equations, we aim to obtain the distribution of velocity, pressure, and temperature in the flow field.

Our implementation involves several steps. Initially, we will generate an appropriate grid to divide the computational domain, providing sufficient resolution to capture the flow characteristics, similar to the mesh adaptation techniques discussed in the aforementioned ANSYS study [6]. Then, we initialize the flow field and apply suitable boundary and initial conditions. In each time step, evaluate the residuals of the flow field, update the flow variables, and check the convergence criteria. To evaluate variations and convergence in the flow field, we will monitor the undivided L_1 norm of the residual vector while calculating and monitoring the output of average total pressure recovery (ATPR). Furthermore, we plan to conduct mesh adaptation iterations under different angles of attack conditions to improve solution quality, as explored in Choubey's work [7]. The post-processing steps to analyze the results include plotting and analyzing the convergence of the residual, the Mach number, total pressure fields, and the relationship between ATPR output and the number of cells in the mesh under various conditions. At the same time, our work seeks to provide a deeper understanding of the relationship between mesh resolution and ATPR output, potentially uncovering new insights into optimizing mesh configurations for improved simulation accuracy and efficiency.

2 Motivations and Objectives

Most of the simulations of supersonic inlet flow employed commercial CFD software, such as ANSYS Fluent, and CFX. However, most of the software would not show details of the simulation, which makes the process like going through a black box. Our group is interested in applying the computational method learned from CSE 6730, including FVM, first-order approximation, and initial and boundary conditions, to replicate the CFD results in commercial software using Python. We will perform a CFD analysis on a simplified ramp supersonic inlet to evaluate the inlet performance, starting from mesh to final properties calculations, and validate our result with ANSYS Fluent simulation.

3 Conceptual Model

The system under research consists of the scramjet engine inlet and the incoming supersonic airflow. Figure 2 presents a conceptual scheme of the system.

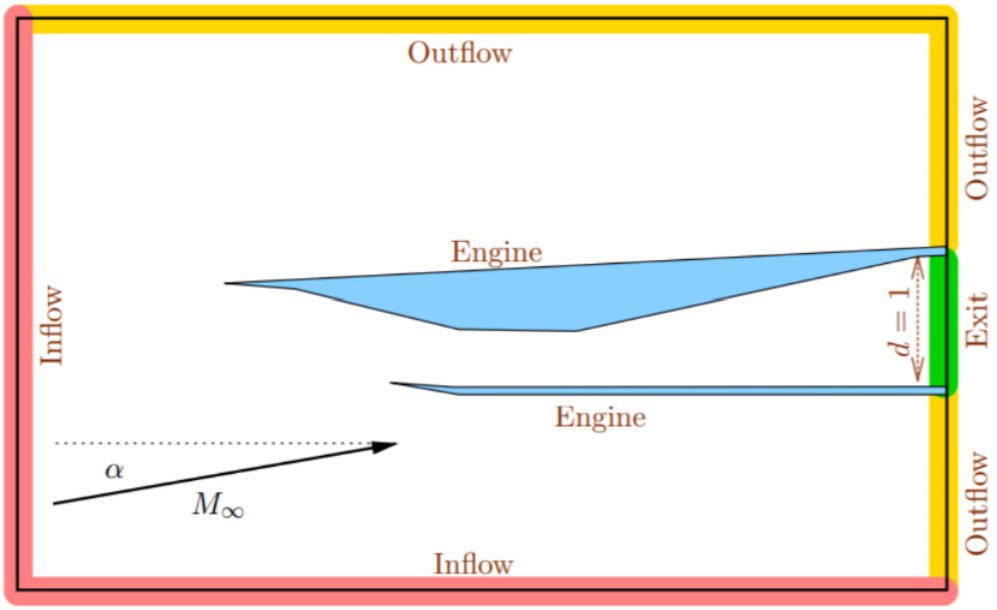

Figure 2: Problem setup

As the figure above illustrates, the system under investigation consists of a rectangular computational domain, wherein the geometry of the scramjet engine inlet is colored blue. The entire white region is filled with dry air and the focal point of this project is to figure out some key properties of the air. The boundary conditions are also crucial to the simulation and need to be cautiously defined. The left and bottom boundaries are defined as the inflow boundary, from where the air will enter the computational domain with prescribed velocity. The top and right sides are defined as the outflow boundaries, allowing the air to exit the computational domain. All the surfaces of the scramjet inlet (the blue region) are treated as adiabatic walls, making the blue region impermeable to the air. Besides, the adiabatic wall precludes the heat of air transfer into the wall.

The goal of this project in a technical way is to use CFD simulation, specifically the Finite Volume Method (FVM) to solve the Euler Equations in fluid dynamics, which is a simplification of Naiver-Stokes Equations under the assumptions of **inviscid, adiabatic and incompressible** flow.

$$\frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} = 0, \tag{1}$$

$$\frac{\partial(\rho u^2)}{\partial x} + \frac{\partial(\rho uv)}{\partial y} = -\frac{\partial P}{\partial x}, \tag{2}$$

$$\frac{\partial(\rho uv)}{\partial x} + \frac{\partial(\rho v^2)}{\partial y} = -\frac{\partial P}{\partial y} \tag{3}$$

Here:

- ρ is the fluid density.
- $\mathbf{V} = [u, v]^T$ is the velocity vector.
- P is the pressure.

There are 3 unknowns in total: u, v, P. The density ρ is assumed to be constant for the entire computational domain. After solving this, we can use the stagnation relations to calculate the total pressure before and after the engine inlet, and then get the total pressure recovery ratio.

4 Platforms

Initially, SolidWorks (Dassault Systèmes, 2023 [8]) is a computer-aid-design (CAD) software and will be employed in this project to create the CAD file of the scramjet engine geometry. Following this, the grid-generating software Gridpro (Program Development Company, 2023 [9]) will be chosen to convert the CAD file into the computational grid file. The grid file is a fundamental basis for the CFD simulation of the scramjet engine inlet. The core of this project is the CFD simulation of the flow around the scramjet engine inlet, and the code for the simulation will be written and executed in Python. Besides, Python also serves the purpose of the post-processing and visualization of the results from CFD simulation.

5 Simulation Model

5.1 CAD to Mesh

A 2D CAD file is generated using SolidWorks and then exported to Gridpro to generate a .grd file. Triangular grids are used for mesh.

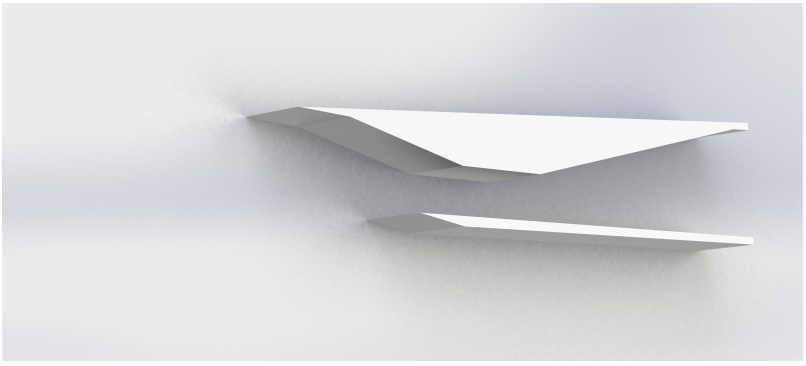

Figure 3: CAD File Generated from Solidworks

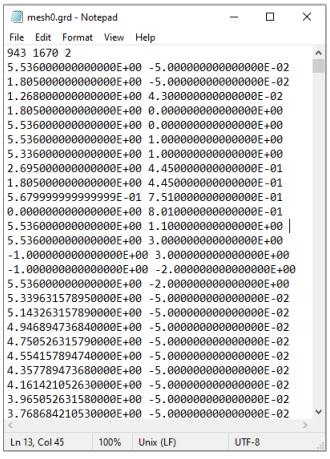

Figure 4: Grid File

The grid file contains all the information for mesh generation, including:

1. Number of nodes
2. Number of edges

3. XY coordinates of each node

4. The node combination of each edge

Then we use *plotmesh.py* file to plot the initial coarse mesh, as shown below:

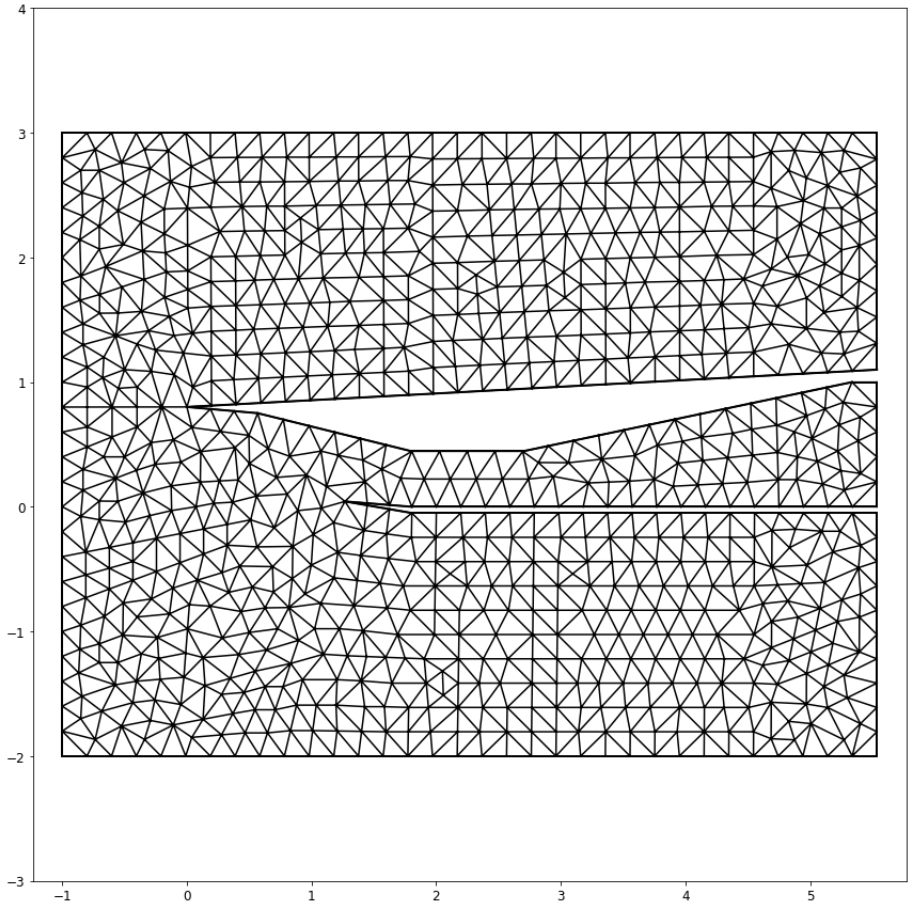

Figure 5: Coarse Mesh

In CFD, the need for fine mesh at sharp regions arises from the complex nature of fluid flow in these areas. Sharp regions, such as edges or corners, are where the fluid experiences significant changes in direction and speed, often resulting in high gradients of flow properties like velocity and pressure. A finer mesh in these regions allows for a more detailed and accurate representation of the flow, ensuring that these critical variations are captured. Without a fine mesh, the simulation may oversimplify these variations, leading to a loss of important details and potentially incorrect predictions of the fluid behavior. Therefore, fine meshing in sharp regions is essential for reliable and precise CFD simulations. Gridpro is utilized for mesh adaption, and the fine mesh is shown below:

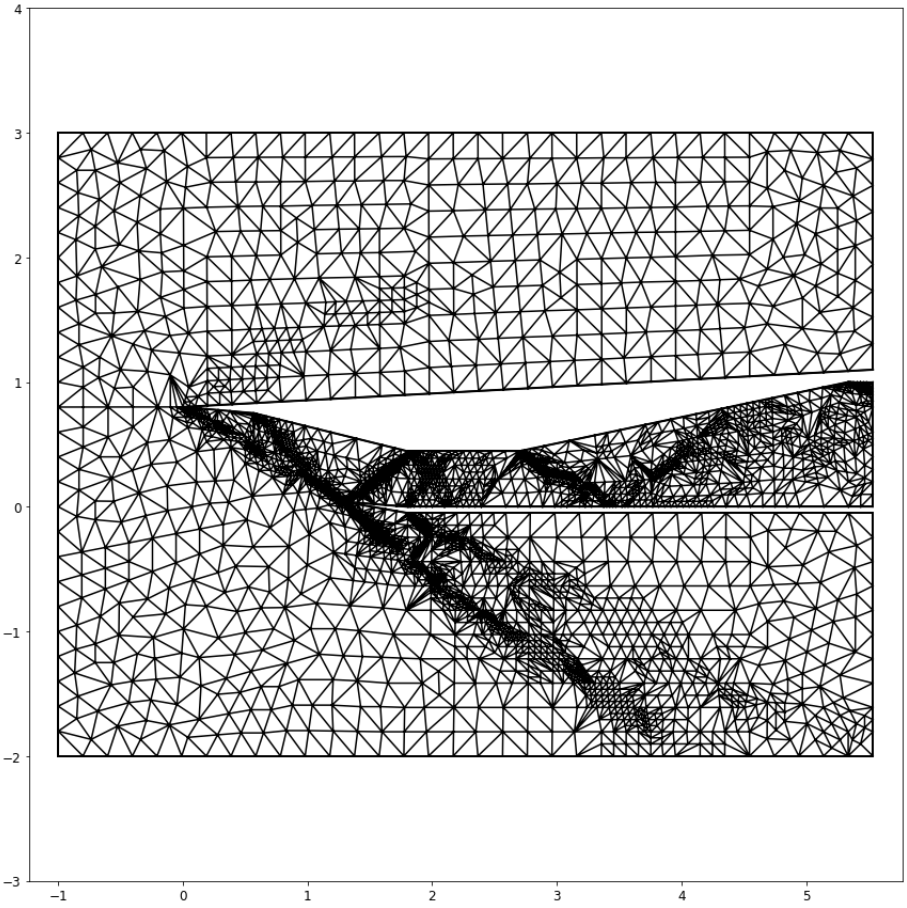

Figure 6: Fine Mesh

The fine mesh will be used for more accurate results in the following steps.

5.2 2D FVM Implementation

As mentioned in the introduction section, flux represents the rate at which each quantity crosses the boundaries of each control volume. This method discretizes the fluid dynamics conservation laws over the control volumes. In this project, we implement the FVM using Python and call the Roe Flux (mentioned in the next section) to calculate the final total pressure recovery ratio of the supersonic inlet. The procedures are listed below:

5.2.1 Import the Mesh

Read the mesh file (*.grd*), save the mesh information in $Mesh$ structure, and save the internal edges and boundary edges in IE and BE matrices. This is implemented in *readgrd.py* file. One row in IE includes:

$$[n_1, n_2, elem_1, elem_2] \tag{4}$$

where:

1. n_1, n_2: nodes 1 and 2
2. $elem_1, elem_2$: elements 1 and 2

One row in BE includes:

$$[n_1, n_2, elem, bgroup] \tag{5}$$

where:

1. *elem*: boundary element
2. *bgroup*: boundary type label, 0 as *engine*, 3 as $inflow$, others as normal boundary edges.

5.2.2 Initialization

Define the constants and initialize arrays and matrices. First, we define a state vector as:

$$\rho U = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \tag{6}$$

Then initialize the state matrix U by free stream condition:

$$U_i = \begin{bmatrix} 1 \\ M_{inf}\cos(\alpha) \\ M_{inf}\sin(\alpha) \\ \frac{1}{(\gamma-1)\gamma} + \frac{M_{inf}^2}{2} \end{bmatrix} \tag{7}$$

5.2.3 FVM Iterations

Now we start the loop over the number of iterations (total as n, index as i). The steps include (this is high-level explanation, may not be specific to each variable):

1. Residual matrix and CFL helper matrix initialization.
2. Loop over the number of boundary edges (index as j):
   (a) If boundary edges are labeled as *engine*:
      i. This boundary condition is *Inviscid Wall*
      ii. Calculate the tangential boundary velocity:

$$\mathbf{V^b} = \mathbf{V}^+ - (\mathbf{V}^+\mathbf{n})\mathbf{n} \tag{8}$$

      Here, $\mathbf{V}^+$ is the interior state velocity and $\mathbf{n}$ is the normal vector of the edge (pointing from left to right cell)

      iii. Calculate the boundary pressure:

$$p^b = (\gamma - 1)(\rho E^+ - \frac{1}{2}\rho^+|\mathbf{V^b}|^2) \tag{9}$$

      iv. Calculate the flux at the boundary:

$$\mathbf{F_b} = [0; p^b n_x; p^b n_y; 0] \tag{10}$$

      v. Calculate the speed of sound:

$$c = \sqrt{(\gamma - 1)(\rho E^+ - \frac{1}{2}\rho^+|\mathbf{V}^+|^2)/\rho^+} \tag{11}$$

vi. Calculate the maximum wavelength:

$$\mathbf{V}^{+}\mathbf{n} = ((\mathbf{V}^{+}\mathbf{n})\mathbf{n}) \tag{12}$$

$$smax = norm(\mathbf{V}^{\mathbf{n}}) + c \tag{13}$$

(b) If the boundary edges are labeled as $Inflow$: then we need to use Roe_Flux function to calculate the flux and maximum wavelength. Here, the left cell is the interior state and the right cell is the free stream state.

$$[\mathbf{F}, smax] = Roe\_Flux(U[:, BE[j,2]], U_{inf}, \mathbf{n}) \tag{14}$$

(c) Else: also use Roe_Flux function to calculate the flux and maximum wavelength. However, here both cells are in the interior states:

$$[\mathbf{F}, smax] = Roe\_Flux(U[:, BE[j,2]], U[:, BE[j,2]], \mathbf{n}) \tag{15}$$

3. Calculate the edge length (suppose there are two points A and B):

$$dl = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2} \tag{16}$$

4. Update the residual matrix:

$$R[:, BE[j,2]] = R[:, BE[j,2]] + \mathbf{F}dl \tag{17}$$

5. Update the CFL helper matrix:

$$s_{dl}[:, BE[j,2]] = s_{dl}[:, BE[j,2]] + smax \cdot dl \tag{18}$$

6. Loop over the internal edges (index as j):

(a) Use Roe_Flux function to calculate the flux and smax:

$$[\mathbf{F}, smax] = Roe\_Flux(U[:, IE[j,2]], U[:, IE[j,3]], \mathbf{n}) \tag{19}$$

(b) Update the residual matrix and CFL helper matrix for the left cell:

$$R[:, IE[j,2]] = R[:, IE[j,2]] + \mathbf{F}dl \tag{20}$$

$$s_{dl}[:, IE[j,2]] = s_{dl}[:, IE[j,2]] + smax \cdot dl \tag{21}$$

(c) Update the residual matrix and CFL helper matrix for the left cell:

$$R[:, IE[j,3]] = R[:, IE[j,3]] - \mathbf{F}dl \tag{22}$$

$$s_{dl}[:, IE[j,3]] = s_{dl}[:, IE[j,3]] + smax \cdot dl \tag{23}$$

7. Loop over all the mesh elements (index as j):

(a) Calculate:

$$\frac{\Delta t_j}{A_j} = \frac{2CFL_j}{s_{dl_j}} \tag{24}$$

(b) Calculate the L1 norm of residual:

$$R_j = |R[0,j]| + |R[1,j]| + |R[2,j]| + |R[3,j]| \tag{25}$$

$$R_{tot} = R_{tot} + R_j \tag{26}$$

8. Calculate the ATPR:

(a) Get the total pressure of the free stream:

$$\mathbf{V_{inf}} = [U_{inf}[1]/U_{inf}[0]; U_{inf}[2]/U_{inf}[0]] \tag{27}$$

$$p_{inf} = (\gamma - 1) \cdot (U_{inf}[3] - \frac{1}{2} U_{inf}[0] \cdot norm(\mathbf{V_{inf}})^2) \tag{28}$$

$$p_{t,inf} = p_{inf}(1 + \frac{\gamma - 1}{2} M_{inf}^2)^{\frac{\gamma}{\gamma - 1}} \tag{29}$$

(b) Loop over the boundary edge:

i. Find the boundary edges labeled as $Exit$

ii. Calculate the edge length

iii. Calculate the total pressure of the element, same formulas as those of free stream

iv. Calculate the integral helper:

$$p_{t,intg} = \frac{p_t}{p_{t,inf}} dl \tag{30}$$

(c) Calculate ATPR:

$$ATPR = \frac{1}{d} \cdot sum(p_{t,intg}) \tag{31}$$

5.3 Roe Flux Implementation

As mentioned in previous sections, one important step of the FVM implementation is the computation of flux at the surface of each control volume. In this project, the Roe Flux Method was employed for the calculation of flux.

Roe Flux Method, named after Philip L. Roe, is a commonly used numerical method in the FVM implementation. In this subsection, the implementation method of Roe Flux will be discussed. As mentioned in the FVM implementation section, the elements in the vector above represent primary variables in Euler's Equation: ρ is the density, u and v are x,y components of the velocity, and E is the total energy per unit mass. The values of the vectors are stored in the center of each control volume and will be updated in each iteration. To update the values, the flux of those variables at the surface of each control volume is needed and the flux is defined as:

$$\vec{F} = \begin{bmatrix} \rho u & \rho v \\ \rho u^2 + P & \rho uv \\ \rho uv & \rho v^2 + P \\ \rho uH & \rho vH \end{bmatrix} \tag{32}$$

Here, P is the pressure:($\vec{v}$ is the velocity vector (u,v), γ=1.4) and P is calculated from:

$$P = \rho RT = (\gamma - 1)[\rho E - \frac{1}{2}\rho[\vec{v}]^2] \tag{33}$$

And H is the total enthalpy per unit mass and is defined as:

$$H = E + \frac{P}{\rho} \tag{34}$$

The details of the Roe Flux Method are complex and it is unnecessary to quota all the details in this report. In this subsection, we will present the equations involved in the Roe Flux Method and also discuss the basic principles behind the theory.

As mentioned above, the flux of the state vector at the surface of each control value is needed in the FVM implementation. As the mesh figure (Figure 5) illustrates, two adjacent triangles are separated by an edge. In the FVM, the variables were assumed to be constant inside one control volume. Therefore, the problem now becomes solving the Flux of variables from a piece-wise constant condition. In other words, FVM aims to estimate the flux on the edge based on the values on both sides of the control volume.

Therefore, the Roe Flux Method is a numerical approach that approximates the flux based on values from both sides of an edge. Instead of solving for the exact Riemann Problem, the Roe Flux Method solves the flux for a linearized Riemann Problem. The flux calculated from the Roe Method is close to the exact flux but requires less computational resources. The following parts are the equations that the Roe Flux Method uses to construct flux:
The speed of sound can be calculated through the following relation:

$$c = \sqrt{\gamma RT} = \sqrt{\gamma P/\rho} \tag{35}$$

The input of our Roe Flux function are the states from both the left cell and right cell (U_L,U_R) and the normal vector $\vec{n}$ pointing from the left cell to the right cell. The core equation of the Roe Flux is shown below:

$$\hat{F} = \frac{1}{2}(F_L + F_R) - \frac{1}{2}\begin{bmatrix} |\lambda_3|\Delta\rho + C_1 \\ |\lambda_3|\Delta\rho\vec{v} + C_1\vec{v} + C_2\vec{n} \\ |\lambda_3|\Delta\rho E + C_1 H + C_2 u \end{bmatrix} \tag{36}$$

Here, $F_L = \vec{F_L} \times \vec{n}$, same as F_R, which are shown in previous parts. And:

$$\begin{gathered} \Delta\rho = \rho_R - \rho_L \\ \Delta\rho\vec{v} = \rho\vec{v}_R - \rho\vec{v}_L \\ \Delta\rho = \rho E_R - \rho E_L \end{gathered} \tag{37}$$

$\vec{v}$ in $C_1\vec{v}$ is defined as the roe-average velocity vector, and it is equal to:

$$\vec{v}_{ra} = \frac{\sqrt{\rho_L}\vec{v}_L + \sqrt{\rho_R}\vec{v}_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{38}$$

And H in $C_1 H$ is also the roe-average enthalpy:

$$H_{ra} = \frac{\sqrt{\rho_L}H_L + \sqrt{\rho_R}H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{39}$$

The u in $C_2 u$ calculation is also a roe-average term:

$$u_{ra} = \vec{v}_{ra}\vec{n} \tag{40}$$

The magnitude of $\vec{v}_{ra}$ and speed of sound are shown below:

$$q_{ra} = |\vec{v}_{ra}| \tag{41}$$

$$c = \sqrt{(\gamma - 1)(H_{ra} - \frac{q_{ra^2}}{2})} \tag{42}$$

Also, the eigenvalues are defined as:

$$[\lambda_1\ \lambda_2\ \lambda_3] = [u_{ra} + c, u_{ra} - c, u_{ra}] \tag{43}$$

Based on the requirement of the project, we set ϵ=0.1c for the entropy fix. So if —λ_i—¡ϵ,

$$\lambda_i = \frac{\epsilon^2 + {\lambda_i}^2}{2\epsilon} \tag{44}$$

Then, all the other parameter definitions are shown below:

$$s_1 = 0.5 * (|\lambda_1| + |\lambda_2|) \tag{45}$$

$$s_2 = 0.5 * (|\lambda_1| - |\lambda_2|) \tag{46}$$

$$G_1 = (\gamma - 1) * (\frac{q^2}{2}\Delta\rho - \vec{v}_{ra}\Delta\rho\vec{v} + \Delta\rho E) \tag{47}$$

$$G_2 = -u_{ra}\Delta\rho + \Delta\rho\vec{v}\vec{n} \tag{48}$$

$$C_1 = \frac{G_1}{c^2}(s_1 - |\lambda_3|) + \frac{G_2}{c}s_2 \tag{49}$$

$$C_2 = \frac{G_1}{c}s_2 + (s_1 - |\lambda_3|)G_2 \tag{50}$$

At the end, for convenience, we also return the maximum wave number:

$$smax = max[|u_{ra}| + c, |u_{ra}| - c, |u_{ra}|] \tag{51}$$

All the equations above present the implementation of the Roe Flux Method.

6 Results Discussion

6.1 Convergence Analysis

In this project, the convergence is mainly determined by the residue. The equations used for the computation of the residual are presented in the FVM section above. We define the residual as the percentage change of the state variable from its value in the last iteration. We have 4 state variables and therefore the l norm was performed to help average the residuals for all variables. The following figure presents how the residual change as the code progress:

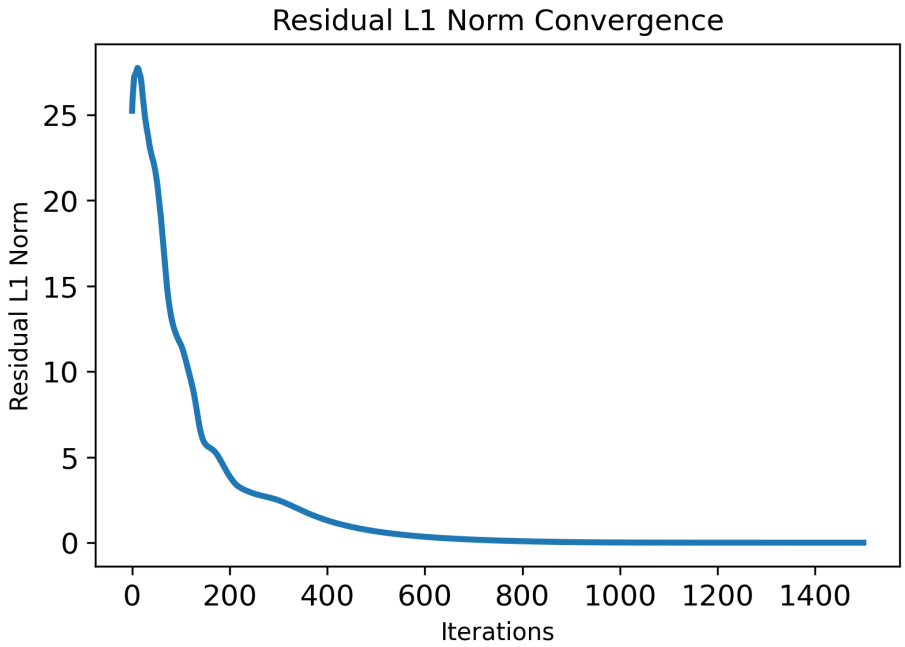

Figure 7: Mesh0Convergence of L1 norm of residual

As the residual figure illustrates, the L_1 norm of the residual decays as the code proceeds. Initially, the L_1 norm is large and it decays into zero as the code goes to a larger number of iterations. The decaying L_1 norm of residual indicates that our code is computing the fluid field successfully and the variables are converging.

Besides the L_1 norm of the residual, the convergence of ATPR was also investigated and the following plot presents the results:

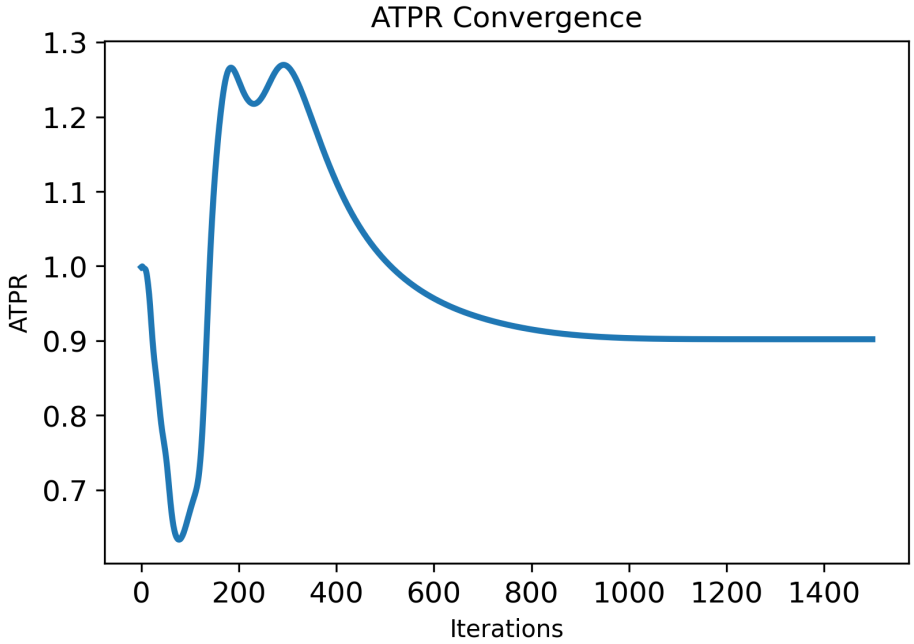

Figure 8: Convergence of Total Pressure Recovery Ratio

As the Figure shows, the value of ATPR initially vibrates. But as the number of iterations becomes larger, the value of ATPR eventually stabilizes and converges into a value.

6.2 Field Plots

After confirming the convergence of our code, we then visualized the fluid properties, and the following field plots were created. We create field plots for the Mach number and the total pressure for both the fine and coarse mesh. Note Mach Number is a non-dimension number that represents the magnitude of the fluid velocity.

6.2.1 Coarse Grid Field Plots

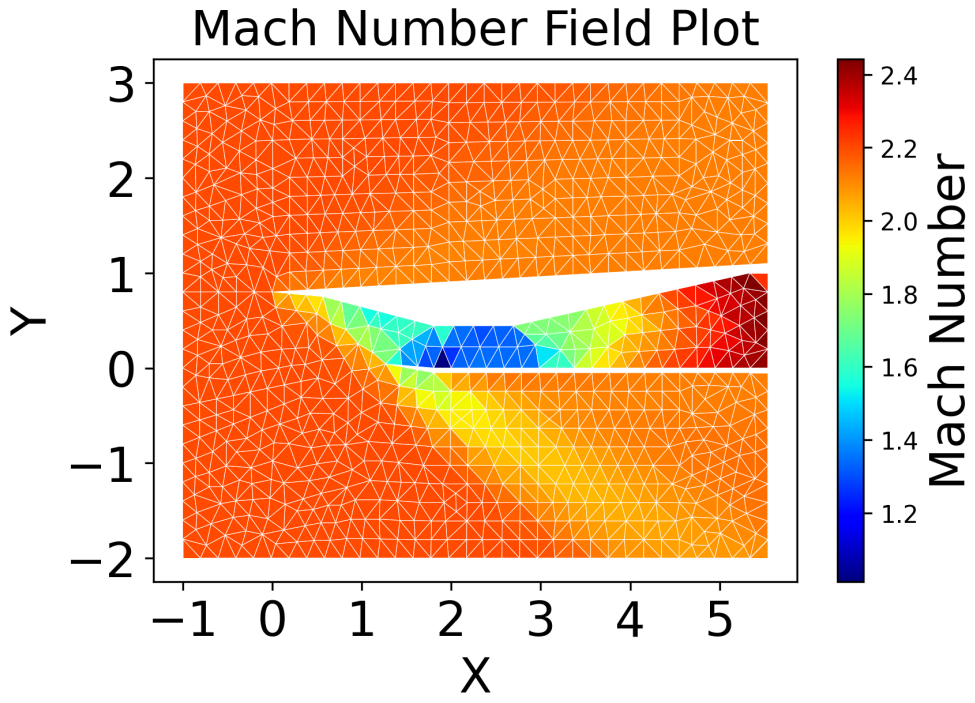

Figure 9: Mach Number Field Plot

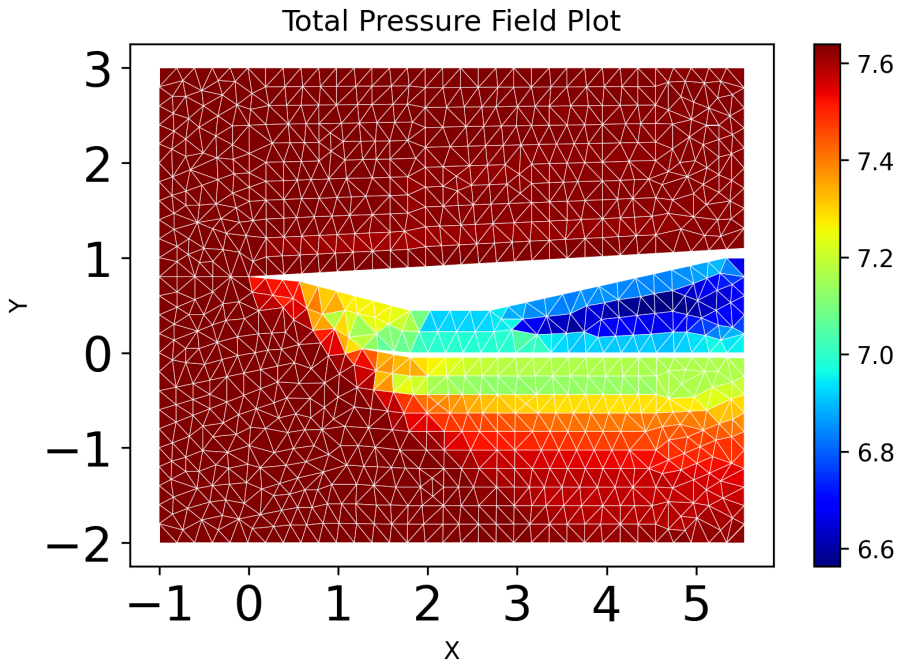

Figure 10: Total Pressure Field Plot

6.2.2 Fine Grid Field Plots

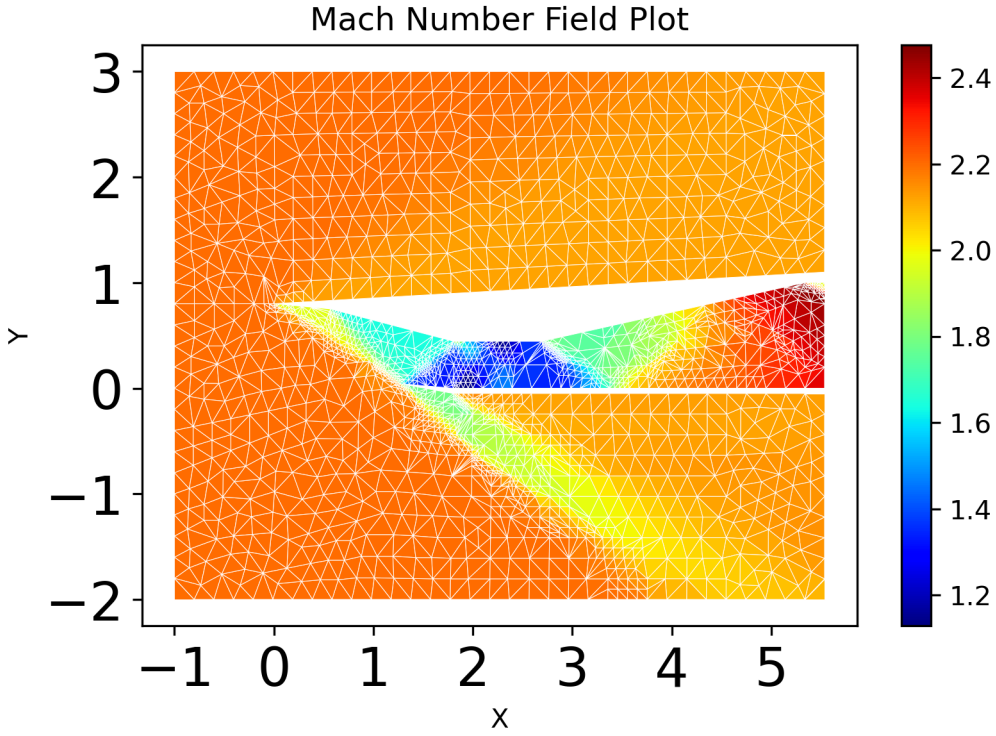

Figure 11: Mach Number Field Plot

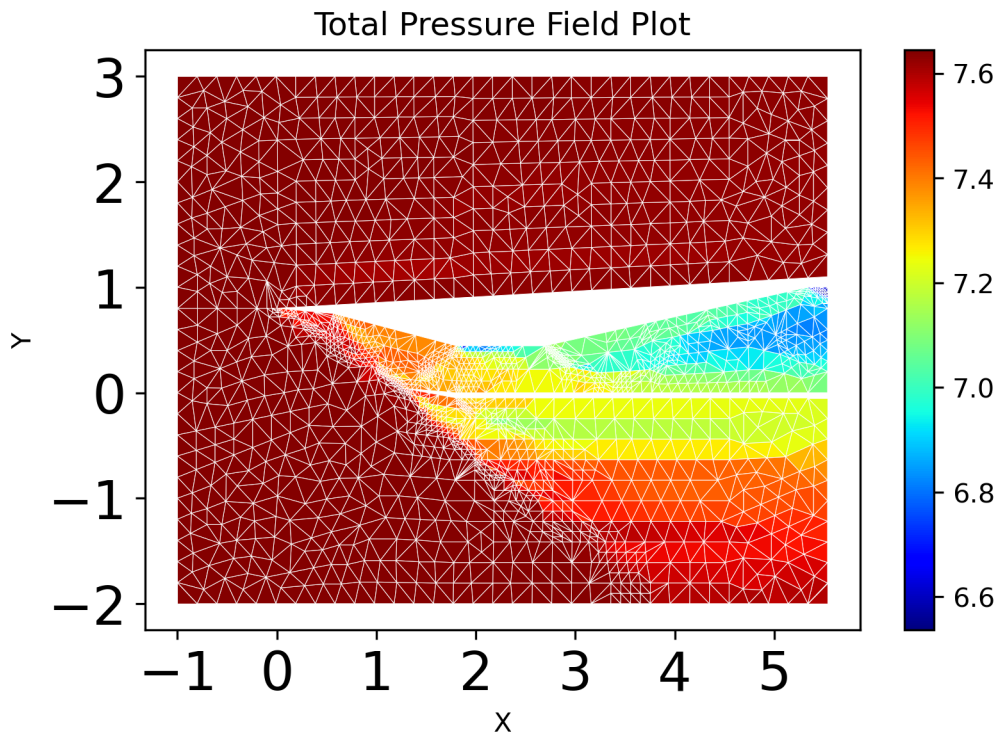

Figure 12: Total Pressure Field Plot

The figures above present the field plots of the fluid properties around the scramjet engine inlet in supersonic inflow conditions. Both the coarse mesh and the fine mesh can generate the field plots for Mach number and total pressure. The validation of the results will be discussed in the next section. From both fine mesh and coarse mesh plots, the shock wave can be viewed. From the plot, the position and shape of the expected leading shock ahead of the inlet can be clearly viewed. Besides, the complicated reflection of shock inside the scramjet inlet was also visually demonstrated.

Comparing the fine grid and coarse grid results, the fine grid can generate a graphical visualization with superior quality. Although the computational cost is higher, fine mesh can simulate the fluid with greater details.

Overall, our code successfully simulated the fluid around the engine inlet, and the results agree with other research qualitatively. From our simulation, the pressure recovery ratio was found to be 0.88, meaning 88 percent of total pressure was preserved as the fluid passed the scramjet inlet.

7 Validation

Various investigations exist on the fluid dynamic simulation around scramjet inlets, providing the expectations of the data from our code. Javad and Safa [10] have performed a simulation on the scramjet inlet, resulting in a graphical visualization that contains what we expect to see from our simulation:

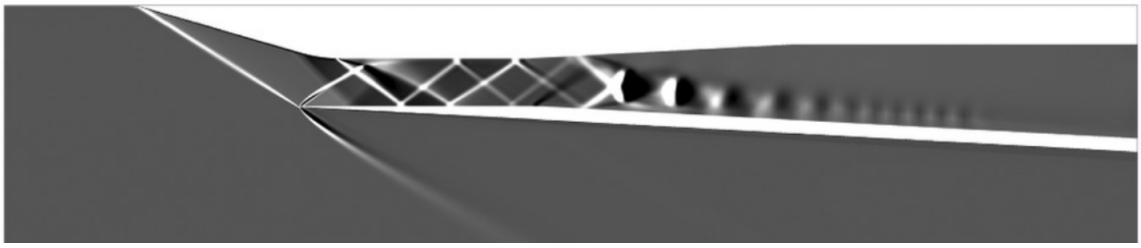

Figure 13: Graphical Visualization Expected Results

From the Figure above, details of the flow field around the scramjet engine inlet can be clearly seen. Note that this Figure presents the static pressure contour around the engine inlet, the lighter the color is, the larger the pressure is. Therefore, the white lines in front of and inside the inlet represent a strong compression wave which is a shock wave. From the figure, We can see a strong leading shock sitting at the entrance of the scramjet inlet, and the leading shock wave also extends into space below the inlet. Inside the inlet, a complex reflection of the shock wave occurs and creates a "shock train".

Comparing our four figures with the figures presented above, the leading shock wave sitting at the entrance can be viewed in our simulation. Besides, although not in the same level of detail, the complex reflection of the shock waves inside the engine inlet was also displayed in our field plot. Qualitatively, our code can simulate the physical fluid motion in supersonic conditions for the scramjet inlet. The distribution of the total pressure also agrees with what other researchers computed.

8 Conclusion

In this project, we started with an extensive literature review and deeply investigated the latest advancements in CFD areas and the finite volume method in CSE. We are motivated to implement the CFD simulation method hidden behind the commercial software using the knowledge learned from the course. At first, we discretized the domain into mesh and implemented the Roe Flux crossing the boundaries of the mesh. Then, we implemented the finite volume method with multiple iterations, including the interrelations between adjacent cells. Finally, we solved the fluid dynamics equations set numerically and calculated the crucial flow parameters. The Mach number and total pressure contour plots are presented in the result section, and the convergence analysis is conducted to prove our model's performance.

For future work, we are interested in implementing the mesh adaption process in meshing software. Creating fine mesh is crucial for CFD simulation, but the judgment process (determining which region should be fined) behind the fining process is very worth exploring. By doing this, we could gain a more thorough understanding of the CFD simulation process and the knowledge of modeling and simulation behind the process.

References

[1] Hongjun Ran and Dimitri Mavris. Preliminary design of a 2d supersonic inlet to maximize total pressure recovery. *AIAA 5th ATIO and16th Lighter-Than-Air Sys Tech. and Balloon Systems Conferences*, 2005. doi: 10.2514/6.2005-7357.

[2] URL https://www.grc.nasa.gov/www/k-12/airplane/inleth.html.

[3] Neela Manoj Kumar B.Alekhya. Design and performance analysis of supersonic inlets using computational fluid dynamics. *International Journal Magazine of Engineering, Technology, Management and Research*, 2016. ISSN 2348-4845.

[4] N. Om Prakash Raj and K. Venkatasubbaiah. A new approach for the design of hypersonic scramjet inlets. *Physics of Fluids*, 24(8):086103, 2012. doi: 10.1063/1.4748130.

[5] Bahia Dris and Najim Salhi. A finite volume simulation of a supersonic laminar flow: Application to a flat plate and compression corner model. *Journal of Materials and Environmental Science*, 8, 01 2017.

[6] Krishna Zore, Isik Ozcer, Luke Munholand, and John Stokes. Ansys cfd simulations of supersonic and hypersonic flows. 2020.

[7] Gautam Choubey and K.M. Pandey. Effect of variation of angle of attack on the performance of two-strut scramjet combustor. *International Journal of Hydrogen Energy*, 41(26):11455–11470, 2016. ISSN 0360-3199. doi: https://doi.org/10.1016/j.ijhydene.2016.04.048.

[8] Dassault Systèmes. Solidworks. URL https://www.solidworks.com/.

[9] Program Development Company. Gridpro. URL https://www.gridpro.com/.

[10] Javad Sepahi-Younsi and Safa Esmaeili. Performance enhancement of a supersonic air intake by applying a heat source. *Journal of Aerospace Engineering*, 33(5):04020048, 2020. doi: 10.1061/(ASCE)AS.1943-5525.0001170. URL https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29AS.1943-5525.0001170.

A Division of Labor

1. Sijian Tan: Roe flux implementation, Mach/pressure contour plot generation, mesh adaption, ANSYS validation
2. Kaiqun Peng: CAD, mesh generation, total pressure recovery implementation, mesh adaption
3. Cheng Zhang: Github version control, FVM implementation, convergence analysis