Various Flow in a Slender Tube: Mach Number and Domain Shape Influenced by Boundary Obstacles Via Small Disturbance Equations

Juankai Wu

School of Mechanical and Manufacturing Engineering, University of New South Wales, Sydney, 2052, Australia

juankai.wu@student.unsw.edu.au

Abstract. With the exponential advancements in computer hardware and software in recent years, computational fluid dynamics (CFD) has emerged as a highly efficient alternative to traditional fluid mechanics studies. Its prowess in accurately predicting and analyzing flow mechanics has led to widespread adoption across diverse engineering disciplines. This research delves into a numerical analysis of flow conditions utilizing the small disturbance equation (SDE). The study bridges the gap between mathematical formulations and computational language, exemplified through Python code, by harnessing the successive over-relaxation (SOR) method, Neumann boundary conditions (BC), and the ghost point technique. The research underscores its predictions' precision in subsonic and supersonic scenarios by meticulously evaluating the resultant errors. This accuracy is further corroborated by examining overarching flow patterns and Mach number distributions. However, the program's outputs indicate a discernible lack of precision in the transonic case, with errors surpassing acceptable thresholds. This discrepancy is hypothesized to stem from potential mischaracterizations of points at the inflow and outflow boundaries. This study, thus, offers invaluable insights while also highlighting areas necessitating further refinement.

Keywords: Hypersonic flow, slender tube, small disturbance equations.

1. Introduction

Computational fluid dynamics (CFD) is a branch of fluid dynamics that solves and analyzes fluid problems with numerical analysis. Compared with a rigid body, fluid cannot resist any shear force or stress without moving, making the traditional studies of fluid properties and conditions the most challenging part, which CFD can easily solve. With the rapid development of hardware and software, generating fluid solutions with numerical analysis has become highly efficient in recent years. With end-to-end deep learning, which was used to improve approximations inside CFD, compared with baseline solvers, the finer resolution was 8 to 10 times higher, resulting in 40- to 80-fold computational speedups with similar accuracy [1]. CFD has been widely employed in various engineering fields, such as aerospace, automobile, biomedical, etc. It is also used in industrial research and design. For example, CFD plays a significant role in the design process of aerospace, where CFD is used to predict the drag, lift, combustion, etc. CFD also helps engineers reduce the amount of physical testing. In the biomedical field, CFD analyzes blood flow in arteries, simulates airflow in the respiratory passages, etc. Compared with experimental studies, CFD has better capability and is more cost-effective for predicting and simulating flow conditions [2]. In addition, CFD contributes to safety in industrial research and design [3]. In the future, CFD may play an important role in reactor design and thermal hydraulics analysis [4]. The following paragraph will predict and show fluid conditions through CFD and apply small-disturbance theory. Small-disturbance theory mainly studies the variety of hypersonic flow for slender bodies [5]. Previous studies frequently used the theory to analyze incompressible, subsonic, transonic, and supersonic flows. For the hypersonic vehicles that are designed as generally slender bodies, the theory can be applied to create an accurate and effective method for approximate calculating a hypersonic flow field [6].

This paper will develop the Python code and show how to predict the condition of hypersonic flow through the small disturbance equation (SDE) in the slender tube with some small obstacles. Firstly, it will explain how to implement the equation with coding under two scenarios, subsonic and supersonic flow. Then, it will illustrate the combination of the two scenarios and show the relative application of transonic flow. Finally, it will state the prediction's accuracy and discuss the method's limitations and improvements.

2. Methodology

This paper will mainly focus on the scenario of two dimensional. The SDE would be:

$$k\varphi_{xx} + \varphi_{yy} = 0 \tag{1}$$

Where K was the function of Mach number, M.

$$k = 1 - M^2 \tag{2}$$

With different Mach numbers greater or smaller than 1, there will be a slight difference in the equation. When considering the supersonic flow, where the Mach number was greater than 1, applying the Euler method, the small-disturbance equation would be [7]:

$$k_{i-1,j}\left(\varphi_{ij} - 2\varphi_{i-1,j} + \varphi_{i-2,j}\right) + \left(\varphi_{i,j+1} - 2\varphi_{i,j} + \varphi_{i,j-1}\right) = 0 \tag{3}$$

Where φ_{ij} was the function of the flow, and i, j represented the row and column. To solve the equation with the given Mach number, SOR was employed. The SOR method was usually applied to decrease the number of iterations needed to solve the system for a large and sparse system of equations. Then, a new variable, dp, was assumed to be the difference between the current solution and the exact solution, where [8]:

$$\varphi_{ij\ exact} = \varphi_{ij} + dp \tag{4}$$

By substituting the above relationship in equation 1, the equation can then be rewritten as:

$$k_{i-1,j}\left((\varphi_{ij} + dp) - 2\varphi_{i-1,j} + \varphi_{i-2,j}\right) + \left(\varphi_{i,j+1} - 2(\varphi_{ij} + dp) + \varphi_{i,j-1}\right) = 0 \tag{5}$$

Following the SOR method, the formula of the dp can be defined by rearranging the equation 3, where:

$$dp = -\varphi_{ij} - \frac{\varphi_{i,j+1} + \varphi_{i,j-1} - 2(1-M^2)\varphi_{i-1,j} + (1-M^2)\varphi_{i-2,j}}{k_{i-1,j} - 2} \tag{6}$$

To make the prediction more accurate, M should represent the local Mach number that depends on the local speed of the flow. The function of M can be written as:

$$M(x) = M_{inf}^2 + (\gamma + 1)M_{inf}^2\varphi_x \tag{7}$$

Where M_{inf} represents the Mach number of the uniform inlet flow, γ represents the specific weight, and φ_x represents the velocity of flow in the x-direction. Applying the Euler method, φ_x can be recovered by:

$$\varphi_x = \frac{\left(\varphi_{i+1,j} + \varphi_{i-1,j}\right)}{2h} \tag{8}$$

After that, the iteration can then be written as:

$$\varphi_{ij}^{n+1} = \varphi_{ij}^{n} + dp \tag{9}$$

Where n represents the numbers, it has been iterated.

Before starting the iteration to solve the equation of the system, the initial velocity of the inlet flow and the boundary condition should be defined. The uniform velocity of the inlet flow, U_{inf}, related to the relative Mach number and the specific weight. According to the definition of the Mach number, it was the proportion of a fluid velocity to the relevant sound speed, the formula of the U_{inf} should then be:

$$U_{inf} = c \cdot M_{inf} \tag{10}$$

Where c is the speed of sound, and it is also related to the condition of the flow, where the function of c then is:

$$c^2 = (\gamma - 1)\left(H_0 - \frac{1}{2}U_{inf}^2\right) \tag{11}$$

Combining the equation 10 and 11, the formula of U_{inf} would then be defined as:

$$U_{inf} = \sqrt{\frac{(\gamma-1)H_0 M_{inf}^2}{1+\frac{1}{2}(\gamma-1)M_{inf}^2}} \tag{12}$$

To solve the boundary condition, for example, the lower boundary, φ_{ij} cannot be defined through equation 6 where $j = 0$ while i, j cannot be negative. The Neumann BC and Ghost point method would be applied in this case. According to the Euler method, the velocity of the flow in y-direction can be defined as [9]:

$$\varphi_y = \frac{(\varphi_{i,j+1} - \varphi_{i,j-1})}{2h} \tag{13}$$

Then, the $j - 1$ term can be considered as:

$$\varphi_{i,j-1} = \varphi_{i,j+1} - 2h\varphi_y \tag{14}$$

The velocity of the flow could be recovered from the potential using the formula:

$$v = U_{inf}\varphi_y \tag{15}$$

Applying the small-disturbance approximation, the velocity near the wall BC becomes:

$$v \approx U_{inf} f'(x) \tag{16}$$

Where $f(x)$ was the shape of the boundary, thus, for the $j - 1$ term at the boundary, the equation would be:

$$\varphi_{i,j-1} = \varphi_{i,j+1} - 2hf'(x) \tag{17}$$

Similarly, for the upper boundary, where the $j + 1$ term does not appear, $\varphi_{i,j+1}$ could be defined as:

$$\varphi_{i,j+1} = \varphi_{i,j-1} + 2hf'(x) \tag{18}$$

For the subsonic flow, where the Mach number was smaller than 1, the equation would be:

$$k_{i,j}(\varphi_{i+1,j} - 2\varphi_{i,j} + \varphi_{i-1,j}) + (\varphi_{i,j+1} - 2\varphi_{i,j} + \varphi_{i,j-1}) = 0 \tag{19}$$

Similar to the supersonic scenario, the formula of the dp can then be defined as:

$$dp = -\varphi_{ij} + \frac{\varphi_{i,j+1} + \varphi_{i,j-1} + (1-M^2)\varphi_{i-1,j} + (1-M^2)\varphi_{i+1,j}}{2k_{i,j} + 2} \tag{20}$$

Then, the equation of the transonic system can be solved by combining equations 3 and 19. Since the equation depends on the Mach number, for combination, a step function of M is created:

$$u_{ij}(M_{ij}) = \begin{cases} 0, & M_{ij} < 1 \\ 1, & M_{ij} \geq 1 \end{cases} \tag{21}$$

The combination of the two equations would then be:

$$u_{ij}(1 - M^2)(\varphi_{ij} - 2\varphi_{i-1,j} + \varphi_{i-2,j}) + (\varphi_{i,j+1} - 2\varphi_{i,j} + \varphi_{i,j-1})$$

$$+(1-u_{ij})(1-M^2)(\varphi_{i+1,j}-2\varphi_{i,j}+\varphi_{i-1,j})=0 \tag{22}$$

Following the previous steps, the dp of the transonic system can then be defined:

$$dp=-\varphi_{ij}+\frac{(\varphi_{i,j+1}+\varphi_{i,j-1})+(1-u_{ij})k_{ij}(\varphi_{i-1,j}+\varphi_{i+1,j})+u_{ij}k_{i-1,j}(-2\varphi_{i-1,j}+\varphi_{i-2,j})}{2(1-u_{ij})k_{ij}+2+u_{ij}k_{i-1,j}} \tag{23}$$

The above equation can easily be translated to the Python code with the previous state. The main idea was to use the 2D array to store the current solution of the flow and iterate it with dp to make the solution close to the exact solution. The more iteration the program did, the result it generated would be more accurate.

3. Result and discussion

The error of the result that the program generated can be defined as the sum of the value that resubmits the solution of the flow at each point in equations 5, 19, or 22, based on the type of the system (subsonic, supersonic, or transonic). The program can generally work well if the error converges with increasing iterations. Besides, the results can be stated as accurate if the error is small enough, for example, around 10^{-11}. In the following paragraph, the program will be tested by a slender tube with some small obstacles. The shape of the tube is shown below:

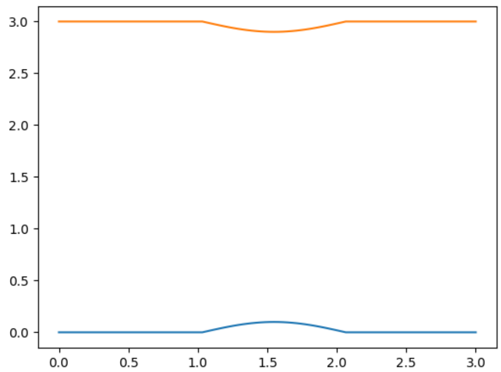

Figure 1. Shape of the tube (Photo/Picture credit: Original)

The testing tube has two small obstacles, each between the two boundaries. The shape of the tube also can be noticed as a small nozzle. Consider the mass conservation, the flow should follow the equation:

$$Av=constant \tag{24}$$

Where A was the cross-section area of the tube, and v was the speed of the flow. The changes between the area and the velocity also follow the equation [10]:

$$(M^2-1)\frac{dv}{v}=\frac{dA}{A} \tag{25}$$

According to Fig. 1, it is obvious that the cross-section area of the tube starts decreasing from 1.0 and reaches the smallest point at 1.5. In general, once the flow passes in the tube from the left-hand side, the velocity of the flow should increase between 1.0 and 1.5. In the program's prediction, the Mach number of the flow around that area should be larger compared with the inlet region. Following equation 25, with the Mach number smaller than 1, there was a negative correlation between the area and the velocity. The M_{inf} of the flow was set to 0.3 for testing, and the result of the prediction and the relative error was shown below:

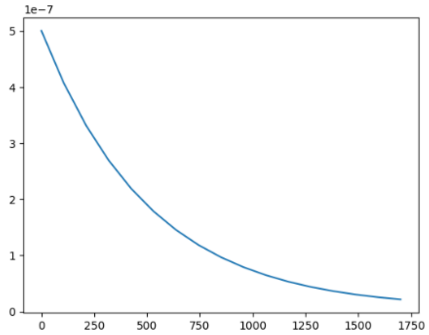

Figure 2. Error of the flow vs. iteration time (Photo/Picture credit: Original)

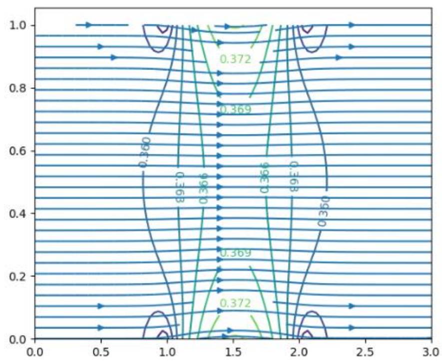

Figure 3. Mach line and flow shape with M_inf = 0.3 (Photo/Picture credit: Original)

According to Fig. 2, the error decreased with increasing iterations. Following the definition of the error, it can be said that the program worked well, and the result was accurate. According to Fig. 3, it can be found that the Mach number around the obstacle was greater than the surrounding obstacle, which also meets the requirements of mass conservation. Reset the Mach number of the inlet flow, where M_{inf} was set to be 0.1. The prediction is shown below:

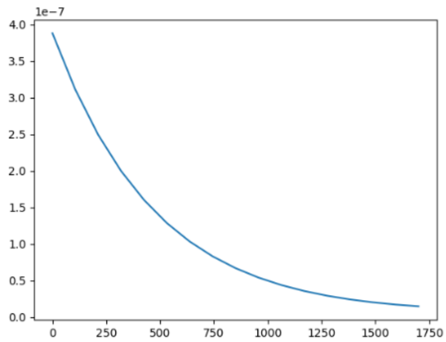

Figure 4. Error of the flow vs. iteration time (Photo/Picture credit: Original)

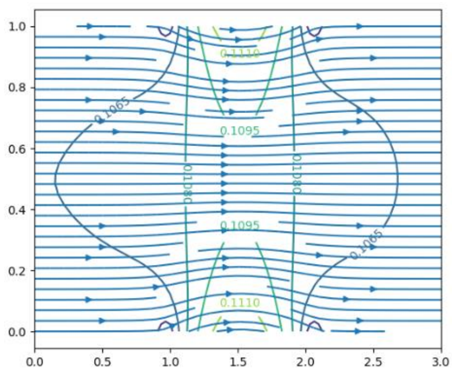

Figure 5. Mach line and flow with M_inf = 0.1 (Photo/Picture credit: Original)

Similar to the previous scenario, Fig. 4 proves the accuracy of the prediction and Fig. 4 shows the condition of the flow. The Mach line in Fig. 5 also meets the requirements of mass conservation. However, compared with Fig. 5 and Fig. 3, there was more empty space in Fig. 5, meaning the number of iterations was insufficient. For the supersonic case, it would be different from the subsonic case. According to the small-disturbance equation, the supersonic flow can be solved by applying equation 3. In this case, the M_{inf} was set to be 1.3, and the prediction was shown below:

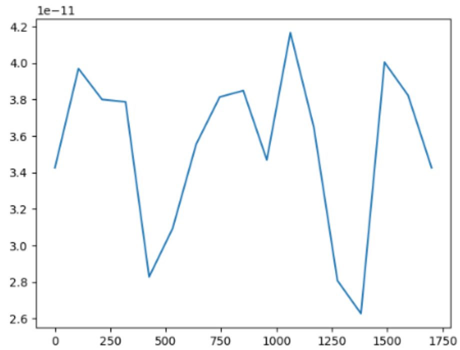

Figure 6. Error of the flow vs. iteration time (Photo/Picture credit: Original)

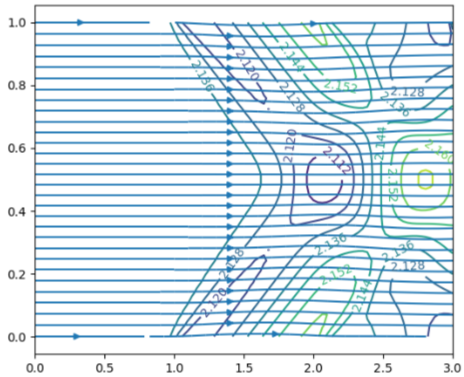

Figure 7. Mach line and flow with M_inf = 1.3 (Photo/Picture credit: Original)

According to Fig. 6, though the error did not decrease when the number of iterations was increasing, the error was already small enough which were around 10^{-11}. It could be stated that the prediction was accurate. Compared with subsonic flow, the relationship between the cross-section area and the Mach number was opposite for supersonic flow. Following equation 25, with the Mach number greater than 1, the area positively correlated with the velocity. Which means the velocity will increase if the area also increases. According to Fig. 7, it can be found that the Mach number decreased when the flow was passing through the throat. The Mach number increases again once it exits the throat.

For the transonic case, the result would be more complicated. Initially, the program would only focus on the bottom boundary, which means the upper boundary would be ignored. The result is shown below:

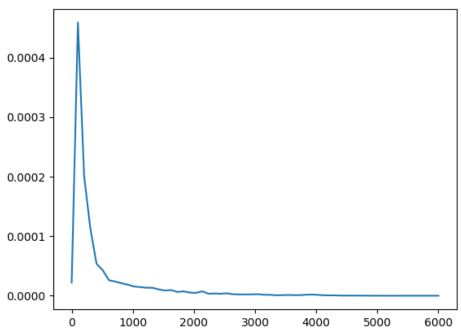

Figure 8. Error of the flow vs. iteration time (Photo/Picture credit: Original)

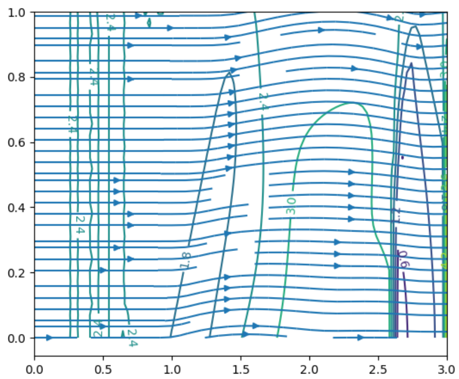

Figure 9. Mach line and flow with M_inf = 0.8 (Photo/Picture credit: Original)

According to Fig. 8, although there was a peak at the beginning, the rest of the errors were convergent. Compared with the subsonic and supersonic cases, it takes more iteration to reach acceptable results. From Fig. 9, it can be found that there were lots of Mach lines stuck together at the rightmost region. That was because, in the Python code, there was no definition of the $i-1$ term in the iteration loop. Because of the missing definition, the program cannot predict the condition of the transonic flow accurately with the upper boundary.

4. Conclusion

CFD has emerged as an indispensable tool in solving and analyzing complex fluid dynamics problems using numerical methods. Via CFD, engineers can solve the hypersonic flow system and easily predict the flow condition, which was the great challenge of traditional studies, for example, the experimental study. Engineers can solve the fluid dynamics problem with numerical analysis in this case. Numerical analysis has become more efficient with the rapid development of software and hardware. This research mainly illustrates how to predict the flow conditions in two-dimensional by applying the small-disturbance theory with Python. The equation was approximated and translated into computing language by applying the SOR and Euler methods. Considering the difference between subsonic and supersonic flow, the equation was separated into two parts and solved the two systems independently. Based on the numerical analysis, the research defined the prediction accuracy. According to the prediction of the subsonic and supersonic case, it can be stated that the program worked correctly because the error was small enough, and the condition generally met the requirement of mass conservation. The prediction would be more accurate with more iterations. However, the program did not work correctly for the transonic case, which was much more complicated. Though the error of the transonic flow with only considering one side boundary reached the acceptable region, it took more than 6000 times iteration, which was much greater than the first two cases. The error was unacceptable once the other boundary was added, and it took more time to generate the result. The program could be improved through the iteration algorithm, and the definition of each point should be confirmed.

References

[1] Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, et al. Machine learning–accelerated computational fluid dynamics. Proceedings of the National Academy of Sciences, 2021, 118 (21): e2101784118.

[2] Yan Yihuan and Tu Jiyuan. Computational Fluid Dynamics, In Bioaerosol Characterisation, Transportation and Transmission, 2023, 68-83.

[3] Shen Ruiqing, Jiao Zeren, Parker Trent, et al. Recent application of Computational Fluid Dynamics (CFD) in process safety and loss prevention: A review, Journal of Loss Prevention in the Process Industries, 2020, 67: 104252.

[4] Mingjun Wang, Yingjie Wang, Wenxi Tian, et al. Recent progress of CFD applications in PWR thermal hydraulics study and future directions, Annals of Nuclear Energy, 2021, 150: 107836.

[5] Van Dyke and Milton D. Applications of hypersonic small-disturbance theory. Journal of the Aeronautical Sciences, 1954, 21 (3): 179-186.

[6] Wuetcher Andrew and Wang Xiaowen. Small disturbance theory for hypersonic flow over slender bodies. In AIAA Scitech 2019 Forum, 2019, 0893.

[7] Angeli Letizia, Crisan Dan and Ottobre Michela. Uniform in time convergence of numerical schemes for stochastic differential equations via Strong Exponential stability: Euler methods, Split-Step and Tamed Schemes. arXiv preprint arXiv, 2023, 2303: 15463.

[8] Saad Yousef. Iterative methods for linear systems of equations: A brief historical journey. Contemporary Mathematics, 2020, 754: 197-215.

[9] Khalilia M. Ehsan, Larssonb Martin and Müller Bernhard. High order ghost-point immersed boundary method for viscous compressible flows based on summation-by-parts operators. International Journal for Numerical Methods in Fluids, 2019, 89 (7): 256-282.

[10] Shariatzadeh Omid Joneydi, Abrishamkar Afshin and Jafari Aliakbar Joneidi. Computational modeling of a typical supersonic converging-diverging nozzle and validation by real measured data. Journal of Clean Energy Technologies 2015, 3 (3): 220-225.