INSTITUTO SUPERIOR TÉCNICO

UNIVERSIDADE DE LISBOA

1º Semester 2023/2024

Computational Fluid Mechanics

1^{st} Computational Exercise: Finite Differences and Finite Volume Methods

---

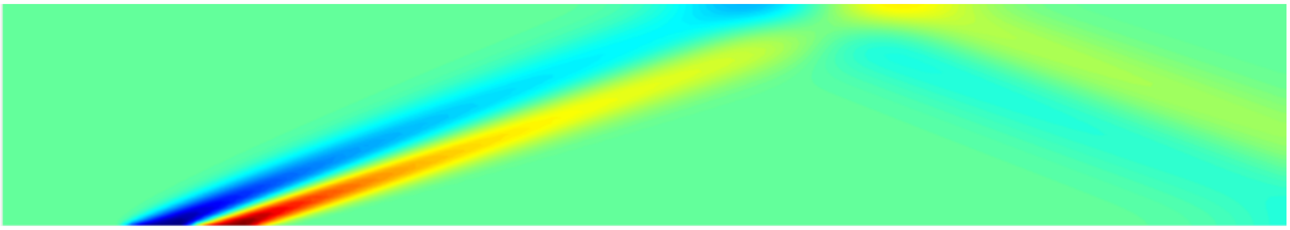


Figure 1: Mach field of reflecting shock

GROUP NUMBER 7

João Fernandes Marques, 93270
Miguel Dias Casalinho, 93307
João Pedro Sousa Gaspar, 96930
Pedro Afonso Simões Gonçalves, 105745

October 24, 2023

Contents

1 Introduction

Computational Fluid Dynamics (CFD) is an engineering tool applied to various fields in project design that allows the analysis of different flow variables in problems of heat transfer and general fluid flow. Despite its current accessibility through mainstream codes, CFD is still as much an art form as it is a strict science, which makes its results hard to reliably evaluate for new users and people without the necessary knowledge in fluid dynamics and numerical methods, that might not have the sensibility to properly create the right setup and give the right inputs. The purpose of this project is to respond to the suggested work of HW1 that aims to provide some of this sensibility when dealing with error estimation and general application of the Finite Volume Method.

The first part of the present work on the numerical estimation of first derivatives for two distinct functions, $\phi_1(x) = x(\cos(0.01x))^2$ and $\phi_2(x) = x(\cos(0.1x))^2$, at $x = 150$, when applying different finite differences methods. These methods were a centered 4th order finite difference scheme and a 1st order backward finite difference scheme. The investigation includes an analysis of error decay with varying grid sizes, the determination of the order of accuracy for each formula, and the plotting of truncation error estimates. This first part provides valuable insights into the performance and suitability of these numerical methods for derivative approximations.

The second part of the project deals with the linearized potential flow equations and the small perturbations approach with which it was possible to compute the 2D flow field around a diamond airfoil. This analysis involved the application of the Finite Volume Method, different discretization schemes and different solvers. Furthermore, different settings for the mesh used were also studied, from grid refinement, to the influence of the application of a non-uniform mesh.

2 Finite Differences

2.1 Methodology

In this section, the methods employed to address the problem of estimating the first derivatives of the functions $\phi_1(x) = x(cos(0.01x))^2$ and $\phi_2(x) = x(cos(0.1x))^2$ at $x = 150$ will be discussed, using two different finite differences formulas.

Firstly, the functions of interest, $\phi_1(x)$ and $\phi_2(x)$, were defined, and the point of interest, $x = 150$, using MATLAB. For a rigorous analysis, the analytical expressions for the first derivatives of $\phi_1(x)$ and $\phi_2(x)$, denoted as $\frac{d\phi_1}{dx}$ and $\frac{d\phi_2}{dx}$, were defined. These analytical derivatives serve as the reference values for assessing the accuracy of the numerical approximations.

For the finite difference approximations, two distinct finite difference formulas were applied to approximate the derivatives of ϕ_1 and ϕ_2. In the first case, the 4^{th} order central difference scheme, which combines forward and backward differences to provide a high level of accuracy, was utilized. The formula used is represented as 1, where h represents the step size used to discretize the domain.

$$\frac{\partial f}{\partial x} \approx \frac{-f(x+2h)+8f(x+h)-8f(x-h)+f(x-2h)}{12h} \tag{1}$$

The formula 2 was used to employ the simpler 1st order backward difference formula, which is suitable for basic approximations.

$$\frac{\partial f}{\partial x} \approx \frac{f(x)-f(x-h)}{h} \tag{2}$$

To assess the accuracy of numerical methods, the absolute error between the numerically estimated derivatives and their corresponding analytical values was calculated for various grid sizes. The error computation is expressed in 3.

$$Error = \left(\frac{\partial \phi}{\partial x}\right)_{exact} - \left(\frac{\partial \phi}{\partial x}\right)_{i} \tag{3}$$

This error quantification provides insights into the accuracy and its variation with the grid sizing

In parallel with the error computation, the truncation error for both finite differences formula is estimated by the first term neglected in the Taylor Series. To compute the truncation error from each finite differences the Taylor series expansion around x given by 4 is used.

$$\varphi(x+h)=\sum_{n=0}^{\infty}\frac{\varphi^{(n)}(x)}{n!}h^n=\varphi(x)+\varphi'(x)(h)+\frac{1}{2!}\varphi''(x)(h)^2+\frac{1}{3!}\varphi^{(3)}(x)(h)^3+\frac{1}{4!}\varphi^{(4)}(x)(h)^4+\frac{1}{5!}\varphi^{(5)}(x)(h)^5 \tag{4}$$

With some manipulation, the truncation error, $T(x)$, for centered finite differences is derived, as given by 5, and for backward finite differences, as given by 6.

$$T(x)=\frac{\varphi}{\partial x}-\frac{-\varphi(x+2h)+8\varphi(x+h)-8\varphi(x-h)+\varphi(x-2h)}{12h}=\frac{\varphi^{(5)}(x)h^4}{30}+O(h^6) \tag{5}$$

$$T(x)=\frac{\varphi}{\partial x}-\frac{\varphi(x)-\varphi(x-h)}{h}=\frac{\varphi^{(2)}(x)h}{2}+O(h^2) \tag{6}$$

Finally, to rigorously evaluate the accuracy of each formula, the order of accuracy for each is analytically determined. The order of accuracy can be observed through the first term of the truncation error as demonstrated in 5 and 6, which exhibits orders of $O(h^4)$ and $O(h)$ respectively. This observation confirms that the order of accuracy for the 4^{th} order finite difference method is 4, while for the 1^{st}-order finite difference method it is 1.

2.2 Results and Discussion

To address the first part of the prompt, one begins by plotting the error as a function of the grid size for both finite differences formulas, namely the 4^{th} order centered and 1st order backward methods, applied to the function ϕ_1, as shown in figure 2 and for ϕ_2, as displayed in figure 3.

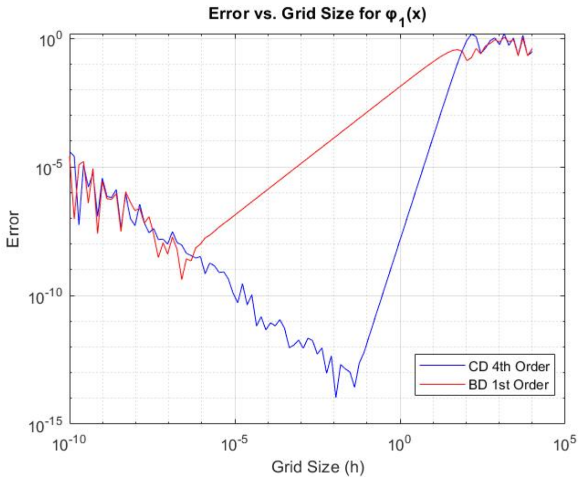


Figure 2: Variation of the error of φ_1 with grid size.

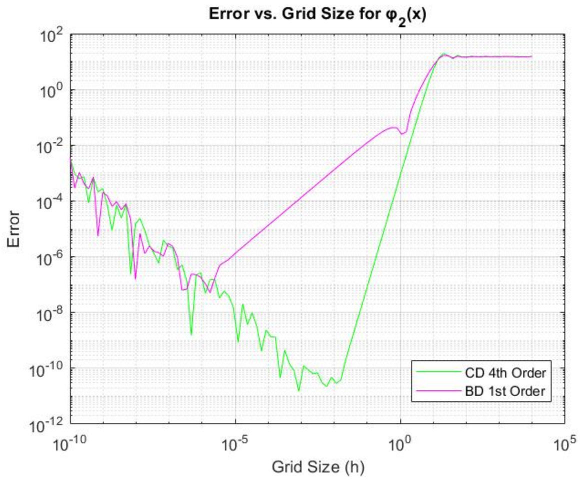


Figure 3: Variation of the error of φ_2 with grid size.

By analyzing the above plots, the order of accuracy for each finite difference formula can be determined graphically. This graphical analysis involves fitting a linear regression, like the one represented in 7, to the data within the range where the asymptotic behavior occurs on a log-log scale and observing the slope, p, of the resulting line.

$$log|error| = p\,log\,h + log\,C \tag{7}$$

For larger step sizes on the right side of figures 2 and 3, the error versus step size plot appears as a straight line on the log-log scale. This behaviour is consistent with the equation 7.

The 4th order centered formula exhibits a slope of approximately 4 in both plots (in 2 and 3), indicating an order of accuracy of approximately 4, this is, a decrease in mesh size of 1 order of magnitude leads to a decrease in the error of 4 orders. The same pattern is observed for the 1st order backward formula, with a slope of approximately 1, confirming the fourth and first order error for each respective finite difference expression used in this analysis as demonstrated in 2.1.

In the initial part of the graphs, for small values of h, where $h < 10^{-1}$ for ϕ_1 and $h < 10^{-2}$ for ϕ_2 for the 4th order finite difference, and for the 1st order where $h < 10^{-7}$ for ϕ_1 and $h < 10^{-6}$ for ϕ_2, it is noticeable that the finite difference analysis suffers from another source of error. This can be explained by the fact that in the numerator of the finite difference expression (equations 1 and 2), as his value becomes smaller, it involves the subtraction of nearly equal numbers to obtain the error. This type of error is referred to as round off error, stemming from the need for a computer to round real numbers to a finite precision.

The next step involves plotting the estimation of truncation error for both finite difference formulas (4^{th} order centered and 1st order backward) on a log-log scale for both function, as depicted in figures 4 and 5.

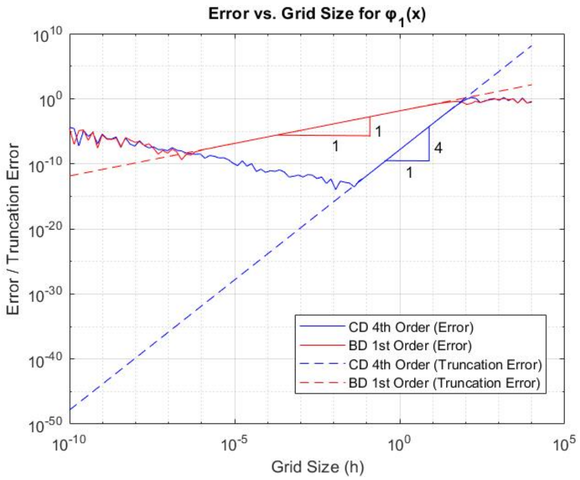


Figure 4: Variation of the truncation error of φ_1 with grid size.

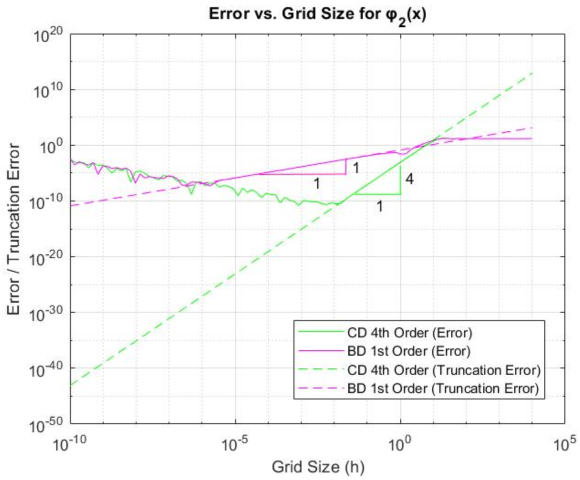


Figure 5: Variation of the truncation error of φ_2 with grid size.

In these graphs, it is evident that the truncation error is directly proportional to the step size, h. As the step size decreases, the truncation error decreases as well. Additionally, it is observed that the lines representing the truncation error and the straight error line for each function closely coincide, affirming the accuracy of the results.

3 Finite volume in fluid mechanics

3.1 Mesh and Equations Discretization

The equations that model the problem are given bellow, being the linearized potencial flow equations applied to small perturbations. Equation 8 is applied to supersonic and subsonic flow (linear equation) and equation 9 transonic flow (non-linear equation):

$$\frac{\partial^2 \phi}{\partial x^2}\left(1 - M_\infty^2\right) + \frac{\partial^2 \phi}{\partial y^2} = 0 \tag{8}$$

$$\frac{\partial^2 \phi}{\partial x^2}\left(1 - M_\infty^2 - (1+\gamma)\frac{M_\infty^2}{U_\infty}\frac{\partial \phi}{\partial x}\right) + \frac{\partial^2 \phi}{\partial y^2} = 0 \tag{9}$$

Assuming α as the coefficient of the first term in each equation, the general discretization applying finite volume method gives:

$$\Delta y_P \left(\alpha_e \left.\frac{\partial \phi}{\partial x}\right|_e - \alpha_w \left.\frac{\partial \phi}{\partial x}\right|_w \right) - \Delta x_P \left(\left.\frac{\partial \phi}{\partial y}\right|_n - \left.\frac{\partial \phi}{\partial y}\right|_s \right) \tag{10}$$

When utilising the linear equation, α is constant and thus does not change with iterations, being treated as a constant ($\alpha_w = \alpha_e = 1 - M_\infty^2$). This does not happen for the transonic equation were α depends on $\frac{\partial \phi}{\partial x}$, thus needing to be updated during the iteration process, as it will be explained later. The value of α at the faces was computed by upwind differences.

Depending on the flow regime entering and leaving the cell (represented by α), the gradients in the general discretized equation needed to be calculated differently, in order to have a more physically accurate model. The general approach was to approximate the gradient in x by central differences when $\alpha > 0$, to better capture the parabolic nature of the resultant equation, and if $\alpha < 0$, upwind differences were used so that the hyperbolic behaviour could be taken into consideration (shown in equation 11).For both cases, central differences were used for the gradients in the y direction.

$$\left.\frac{\partial \phi}{\partial x}\right|_e = \frac{\phi_E - \phi_P}{\Delta x_{P-E}}\; if\; \alpha_e > 0; \quad \left.\frac{\partial \phi}{\partial x}\right|_w = \frac{\phi_P - \phi_W}{\Delta x_{P-W}}\; if\; \alpha_e > 0; \quad \left.\frac{\partial \phi}{\partial x}\right|_e = \frac{\phi_P - \phi_W}{\Delta x_{P-W}}\; if\; \alpha_e < 0; \quad \left.\frac{\partial \phi}{\partial x}\right|_w = \frac{\phi_W - \phi_{WW}}{\Delta x_{W-WW}}\; if\; \alpha_e < 0 \tag{11}$$

Boundary conditions (BC) were given in all domain limits except for the outlet in supersonic flow, where this condition is not necessary due to the equation's hyperbolic behaviour.

For all cases, there is a homogeneous Dirichlet boundary condition applied at the inlet of the domain, that is needed in order to obtain an unique solution that can be interpreted. This BC could have been non-homogeneous, a different solution with the same physical meaning would have been obtained; the BC being homogeneous simplifies the problem. Special care was taken in the discretization of the inlet volumes where the distance used needed to be taken as the distance from the inlet face to the centroid of the 1^{st} volumes, which gives a stronger influence of that face of the volume on its value, compared with the other faces. Also, if upwind differences were used, the same needed to be considered when solving the equation in the volumes at the 2^{nd} column of the mesh grid.

Homogeneous Newman BC's were specified at the rest of the boundaries, except at the inlet of subsonic regime and at the outlet of supersonic regime. These represent open boundaries that do not add/take any momentum to/from the domain, i.e., the flux of the potential field through the boundaries is null. A special case occurs at the bottom boundary where the airfoil is to be modeled such that an upward flux is given as $\left.\frac{\partial \phi}{\partial y}\right|_s = U_\infty \frac{\partial y}{\partial x}$, simulating, in this small perturbation approach, the presence of the airfoil.

3.2 Code's General Description

This section describes the code created to simulate the behavior of fluid flow over the airfoil. This program's main goal is to offer insights into the dynamics of the flow around an airfoil under various flow regimes.

The key input parameters, which define the conditions under which the simulation will occur, are first defined by the program. The geometrical characteristics of the airfoil, the reference temperature and pressure, and the Mach number are some of these factors.

The creation of a computational mesh is one of the program's main components. The discrete grid cells created by this mesh allow for the approximation of fluid properties at specific locations within the simulation domain. Considerable consideration is given to variables including the domain size, the number of grid cells, and information about the airfoil's shape. Furthermore, the possibility of having an non unifom mesh can also be addressed by the code, providing the possibility of increasing accuracy in important parts of the domain with a relatively small penalty in computational time, depending on the mesh characteristics.

The solver, which is at the center of the program, is in charge of iteratively updating the flow variables (in this case the potential field) across the mesh. The Successive Over-Relaxation (SOR) approach is the base of the iterative process. This iterative approach generates a new solution until either a specified level of convergence is reached (for both residuals and error) or a maximum number of iterations is executed. The relaxation factor is one element that affects how well the method performs. In order to accelerate convergence, this factor can be changed throughout iterations, this is, increasing its value once the error drops to a small enough value when convergence is almost guarantied.

Another important aspect of the program is its capability of handling transonic flow conditions, and the non-linearities associated with them. The program modifies the value α after some iterations of the SOR solver to properly account for these conditions, returning with the new values of the coefficients after the $alpha$ update.

Finally, once the iterative process has converged, the code offers post processing capabilities for visualizing the results. Plots of the flow field properties like velocity, pressure, and temperature are included in these visualizations. Error residuals are also plotted in order to assess how accurate the solution is.

3.3 Iterative Solver

After the discretization of the equations governing the flow, the matrices A e b are constructed, which represent the coefficient matrix and the source vector of these equations in a matrix system of the form $A\phi = B$, where ϕ is the potential function of the flow. Once this system is created, it is necessary to solve it in order to obtain the resulting potential. For this purpose, the Successive Relaxation Method (SOR) has been implemented. To implement the iterative SOR method, two strategies were used. The first strategy was based on formula 12, solving the system in a matrix form - hereafter referred to as "SOR1."

$$\phi^{k+1} = \phi^{k} + \omega\left(\phi_{*}^{k+1} - \phi^{k}\right). \tag{12}$$

For obtaining ϕ_{*}^{k+1}, a first approximation by the Gauss-Seidel

$$x^{(k+1)} = (L+D)^{-1}b - (L+D)^{-1}Ux^{(k)}, \tag{13}$$

considering $A = L - D - U$, where L is the lower matrix, D is the diagonal matrix and U the upper matrix. The second strategy solved the system using the equation 14, element by element - hereafter referred to as "SOR2." These two methods for solving the system using SOR, although equivalent, will have different performances.

$$\phi_{\mathrm{P}}^{i,k+1} = (1-\omega)\phi_{\mathrm{P}}^{i,k} + \frac{\omega}{a_{\mathrm{P}}^{i}}\left(\sum_{c_1} a_{c_1}^{i}\phi_{c_1}^{i,k} + \sum_{c_2} a_{c_2}^{i}\phi_{c_2}^{i,k+1} + b_{\mathrm{P}}^{i}\right) \tag{14}$$

In order to compare the two described strategies, similar simulations were conducted, where the Mach number at the inlet of the domain and the type of SOR used were changed. Another parameter that was changed in the simulations was the number of successive SOR iterations, which helps to determine with more precision the iteration at which the method achieved convergence. This parameter is important because it allowed the user to make updates to the relaxation factor while the simulation was running (see section 3.6). The simulation parameters used for the flow, mesh, and convergence criteria are summarized in the table 1. In Section 5.1, we present the evolution of errors and residuals (figures 12, 13, 14) during the flow solution process for each regime. In all cases, we observe a consistent trend of decreasing both the residue and the error as iterations progress.

The table 2 displays the main results of using the two previously described strategies, specifying values for runtime, number of iterations, error, and residue resulting from the iterative method. The results are presented based on flow regimes: subsonic, transonic, and supersonic for the values shown in table 1. This table also provides runtime information solely for the purpose of temporal comparison. It should be noted that if a faster or slower computer were used, the runtime would vary. However, the number of iterations would remain constant as the iterative method would require the same number of iterations to achieve the desired tolerance.

In general, both strategies were able to reach the tolerance of 1e-3 when solving the system with the desired precision. Another important point to consider is the significant difference in convergence speed between the transonic regime and the subsonic and transonic regimes. This is due to the nonlinearity of the potencial flow equation and the consequent need of the update of the parameter α throughout the iterative process.

SIMULATION PARAMETERS	
FLOW	
Mach	{0.2; 0.91, 2.2}
MESH	
x domain	[0, 20]
y domain	[0, 15]
No. of cells x	40
No. of cells y	40
CONVERGENCE	
Function Tolerance	1e-3
Number max of iterations	100000
Successive SOR iterations	{10,10,2}

Table 1: Simulations parameters for SOR methods comparison

SOR1				
	Timer [s]	Iterations	Error	Residual
Subsonic	3.13	48	7.62e-04	9.51e-04
Transonic	45.11	920	4.09e-04	9.97e-04
Supersonic	1.93	24	3.48e-04	1.98e-04
SOR2				
	Timer [s]	Iterations	Error	Residual
Subsonic	3.72	364	6.73e-05	9.99e-04
Transonic	329.15	42350	1.05e-05	1.00e-03
Supersonic	0.40	26	9.74e-04	8.56e-04

Table 2: General results of SOR solvers for the three flow conditions

3.4 Uniform mesh analysis

Firstly, an uniform mesh was used to solve each of the 3 problems. As it was stated in section 3.2, for this type of mesh, the user can set the domain limits in the x and y directions, the airfoil position and the number of finite volumes per column/row of the mesh grid.

Thus, this section will touch on this aspects of the project. Section 3.4.1 will explain how to properly choose the domain size and airfoil position for each of the cases, and sections 3.4.2, 3.4.3 and 3.4.4 will compare the solutions of each for progressively more refined meshes.

In sections 5.2, 5.3 and 5.4 of the Appendixes are shown the most refined meshes that were obtained.

3.4.1 Domain size and airfoil position

Some attention needs to be given to this parameters in order to reduce the time and number of iterations necessary to solve the problem properly. In addition, the physical nature of each of the cases needs to be taken into account in order to not overlook important phenomena of the flow.

In the **subsonic case** the linearized potential equation is parabolic, which means that what is happening upstream of the flow propagates in all directions of the domain, including downstream. Due to this fact, the position of the airfoil needs to be in the middle of the domain in order to capture this behaviour. For this case, we found that a domain of 10 by 5 meters has the proper size to capture these propagation effects, without the computational cost of larger domains; a bigger domain can obviously be used, at the cost of a bigger computational effort, that is being "wasted" computing volumes that are very far from the airfoil and practically irrelevant to the solution.

For the **supersonic case** the nature of the equation is hyperbolic, i.e., information only propagates upstream. Due to that, the airfoil was positioned closer to the inlet border of the domain (the middle of the airfoil is at $x = 1.5$), without any effect to the solution. If the domain was to large in the x direction or too small in the y direction the shock waves would reflect at the boundaries, as represented in figure 1 in the cover of this project. This is not desirable for this study, so for this reason, the domain chosen in this case is smaller in the x direction than the previous one (5 by 5 meters).

Lastly, the **transonic case**, it is mostly subsonic except near the airfoil, where there are regions where the flow goes supersonic. Because of that "mostly subsonic" behaviour, for this case the airfoil was placed at the center of the domain, similar to the subsonic case. Due to the non-linear behaviour of the equation, a larger domain was chosen, namely 15x15 meters. This ensures that the gradients of the solution were almost null closer to the borders, specially at inlet and outlet. This change mitigates problems related with the update of the α constant between iterations.

In the following subsections the limits of the colorbar of the Potential, Mach and density field plots was maintained constant so that the difference in accuracy between each mesh was immediately perceptible. In subsonic and supersonic the tolerance after each the solution was considered converged was 10^{-5} while of transonic, due to the increased complexity of the equation, the limit was 10^{-3}. The methods explained in section 3.6 about the variable relaxation factor were used in these following sub sections in order to run the simulations faster.

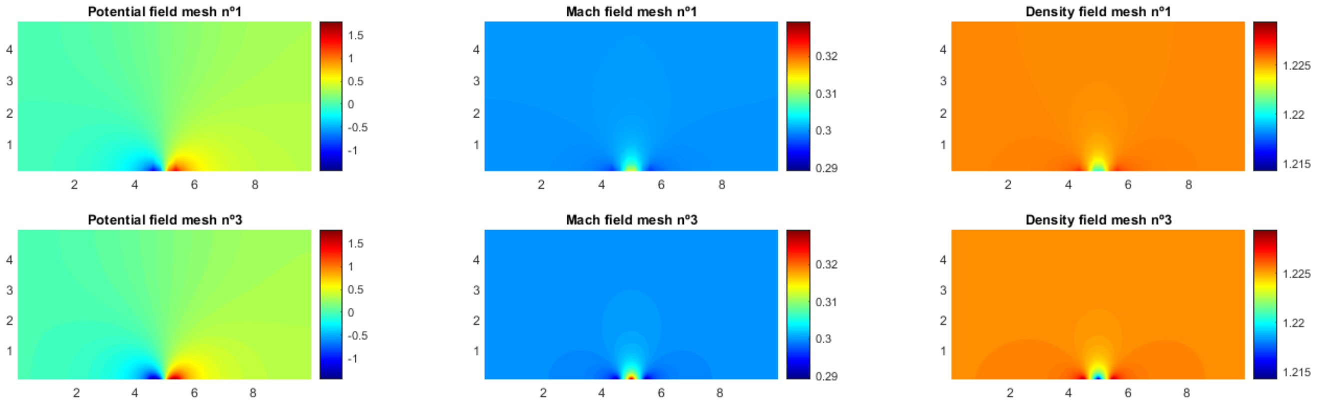
3.4.2 Subsonic uniform mesh refinement

Figure 6 shows the comparison between the first and last meshes of the refinement. The meshes nº1 to nº3 had 40x20, 80x40 and 160x80 volumes, respectively. In the appendix, figures 15 show mesh nº3 but with bigger images. As it can be seen, the general physics of the problem are captured : as expected there is a reduction of the velocity when approaching the airfoil and a increase on the top part of it.

As for the refinement, it is apparent that in mesh nº1 the strong gradients near the airfoil are not nearly as good compared to mesh nº3. In mesh 1 the highest Mach number is $M_{max} = 0.312$ while in mesh nº3 $M_{max} = 0.329$. Also, other less strong gradients, are not as sharply defined compared to mesh nº3.

Other aspect that is very noticeable between the meshes is how the disturbances near the airfoil don't seem to affect the inlet and outlet boundaries as much in the most refined mesh, while in the other ones, there's a clear influence of the airfoil. At the inlet, closer to the bottom boundary, Mach number is lower than at the inlet top boundary, this is not supposed to happen.

Time spent in the convergence with mesh nº3 was the highest (approximately 60 times more). The poorer results are caused by the lack of elements near the airfoil, so a non-uniform mesh might be helpful in order to reduce computation time without compromising the results as much.


Potential field mesh nº1

Mach field mesh nº1

Density field mesh nº1

Potential field mesh nº3

Mach field mesh nº3

Density field mesh nº3

Figure 6: Subsonic Results (mesh 1: 40x20 | mesh 2: 160x80

3.4.3 Supersonic uniform mesh refinement

Figure 8 shows the comparison between the first and last meshes of the refinement. The meshes nº1 to nº3 had 20x20, 40x40 and 80x80 volumes (same cell dimensions of the subsonic case).

In the supersonic case, the difference in detail between grids is more pronounced. In the Mach field plot of mesh nº1, the image is so blurry that the shocks are almost unintelligible. As in subsonic, the regions with larger gradients, for this case, closer to the shocks, are not discretized enough to correctly capture the phenomenon.

In the finer mesh, it can be clearly identified the 4 expected regions around the diamond airfoil in supersonic flow: the unperturbed flow before the shock, the first shock (around 1m) where the mach decreases considerably, the Prandtl-Meyer expansion fan with the increase of mach and finally the second oblique shock that returns the Mach to 2.2. The theoretical mach angle for It can even be seen that the angle of the 2^{st} shock is smaller than the 1^{st} one, as expected. Furthermore, applying oblique shock theory, the shock angle expected for a Mach of 2.2 and a dihedral of 3.43º is 29.8º which correlates well with the visual results obtained where the angle seems to be between 27 and 31 degrees. Mach distribution in the airfoil can be seen in Figure 9 when non-uniform grids are analysed. Opposite to the subsonic behaviour, the residuals present a initial increase on its value and only then start converging. This behaviour can be clearly seen in figures 10 and 11.

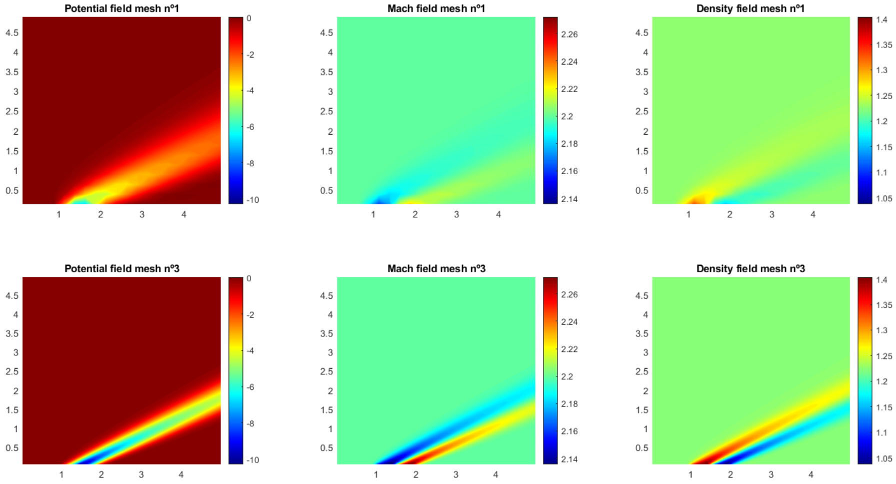


Figure 7: Supersonic mesh refinement

3.4.4 Transonic uniform mesh refinement

As previously mentioned, the transonic case required the update of the alpha parameter from the velocity gradient, which would change the coefficients related to the specific point. Due to this non linearty of the problem, the convergence presented to be difficult and more dependent on good user inputs than the other regimes. The following figures show the results for two transonic cases with meshes of 40x40 and 40x60 elements, for a domain of 10 meters in both directions. Furthermore a mesh with 60x60 with the domain of 15 meters was also performed, it is shown in section 5.4. It has the same grid spacing as the previous discretizations but with less perturbation from boundaries.

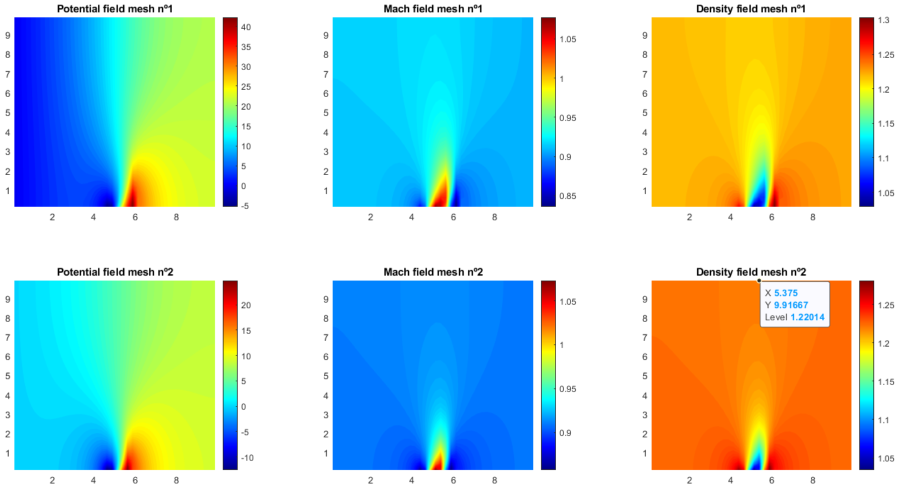


Figure 8: Transonic mesh refinement

As can be seen, it is possible to identify the shock on top of the airfoil after the flow turns supersonic.

3.5 Non-uniform mesh analysis

As stated in the code description section, the program has the capability of generating a non-uniform mesh. Despite the low robustness and high dependency in good user inputs (care as to be taken for the airfoil to be positioned with the same number the cells for each side), the meshes created can provide higher resolution with minimal increase in computational cost. As an example, three meshes are presented in figure 18 to 20 in the Appendix.

To better understand the inter dependencies of grid sizes, plots of Mach vs X coordinate in the centroid of the first cell, are shown on the plot of Figure 9. The results were obtained, utilizing the grids presented above.

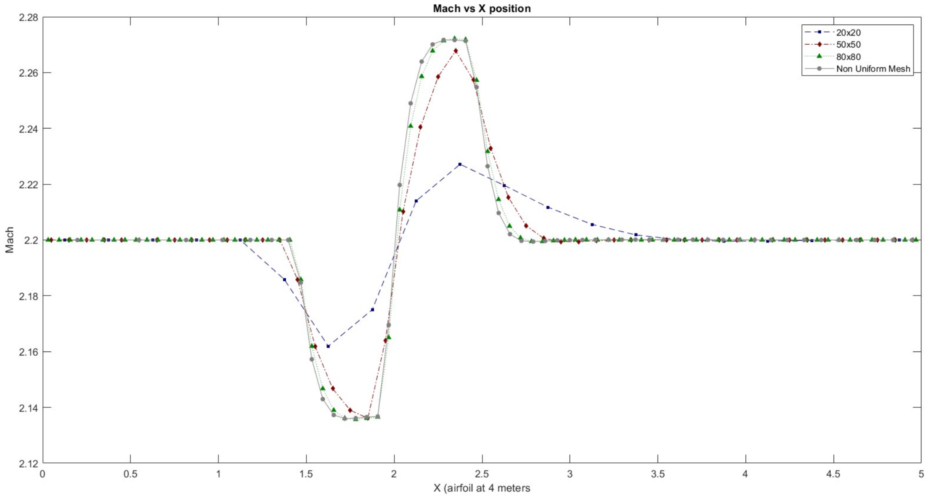


Figure 9: Comparison between Mach variation for different Mesh

The results shown are for an airfoil at Mach 2.2. Both shock waves can be identified by the sharp decrease in Mach number at 1.5 and 2.5, while near the top of the airfoil, the expansion fan is visualized by the rapid increase in Mach number from 2.135 to approximately 2.27. Mesh size plays a key part in obtaining accurate results, with the mesh 50x50 lacking in quality compared with the other meshes. To note that, despite the small increase in the number of volumes compared with the 50x50, the non uniform mesh shows similar results when compared to the 80x80 (37% increase in number of cells). This is the clear advantage of the non uniform meshes: the allocation of mesh volumes to the regions of most interest.

3.6 Test variable relaxation parameter for subsonic and supersonic

While running and reviewing the program, specially in transonic and subsonic cases, it was quickly understood that more refined meshes lead to the most accurate results (see sections 3.5), this came, however, with a disproportional amount of CPU cost. To mitigate this problem, a variable relaxation factor was implemented.

While studying the subsonic case, the group decided to run 4 different cases for the variation of relaxation factor, in order to see its influence on convergence rate. The initial relaxation factor was $\omega = 1.4$, and from that, 4 cases were simulated: no relaxation factor increment, an increment of 0.4, increment 0.5 and finally an increment of 0.58. The target for the modification was the residual value bellow $1x10^{-3}$. The initial ω was set at 1.4 because if higher, the simulations would diverge, and the maximum increment was 0.58 because ω needed to be smaller than 2.

The residuals evolution of each of these 4 cases are plotted in figure 10. Until the increment is applied, the residuals decrease as expected similarly between cases, however, when the increment is applied, a sudden jump (seen by the green line close to 500 iterations) is followed by a sharp decrease of these residuals at a faster rate than the initial one, being the slope proportional to the increment taken. This increment lead to a decrease in computation time from 1114s (13900 iterations) with no increment, to converge within 45s (580 iterations) when 0.58 was applied, 24 to 25 times faster.

Because of this successful experiment, a similar test was performed on mesh nº3 of the supersonic case. The results are shown in Figure 11. For the stopping criteria chosen before, $1x10^{-5}$, there was really no problem in reaching it for the supersonic case provided that ω wasn't to high, which made the simulation diverge. So to test the effect of a variable relaxation factor in this case, the tolerance for which the simulation was considered converged needed to be extremely low, $1x10^{-12}$. Five cases were tested, two with no increment ($\omega = 1$ and $\omega = 1.1$) and three with an initial $\omega = 1.1$ and increment/decrement of either $+0.1$, -0.1 and -0.3, when the errors reached $1x10^{-9}$.

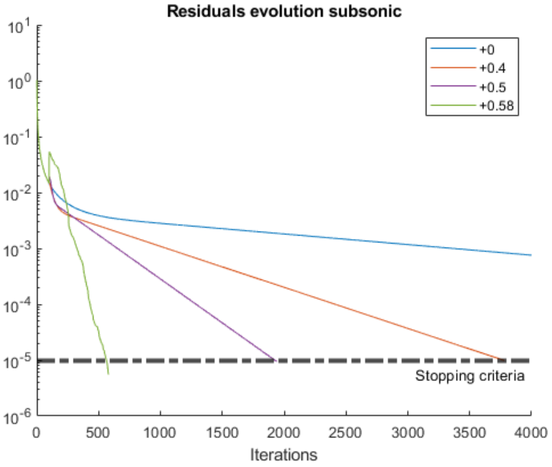


Figure 10: Subsonic convergence, $\omega_{initial} = 1.4$

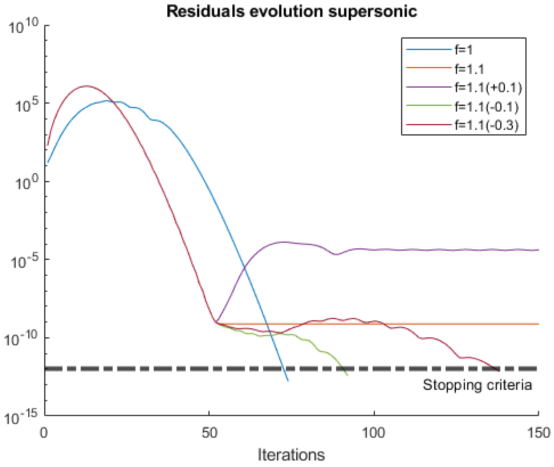


Figure 11: Supersonic convergence

For the cases where $\omega_{initial} = 1.1$, without increment, the residuals stagnate before the desired tolerance is reached. With a 0.1 increment, the residuals increase, which is obviously bad. A decrement makes the simulation converge, with the faster case being the one were ω decreases to 1 (decrement of 0.1), effectively turning the SOR method into the Gauss-Seidel method. The fixed $\omega = 1$ converges faster than any of the other cases for the chosen tolerance, but if for example, the tolerance was $1000x$ higher, it would be slower. Despite of that, the effect of a variable relaxation factor is not as powerfull as in the subsonic case, and, for supersonic, it is more beneficial to decrease ω than to increase it, if it is needed to reach very low tolerances.

4 Conclusion

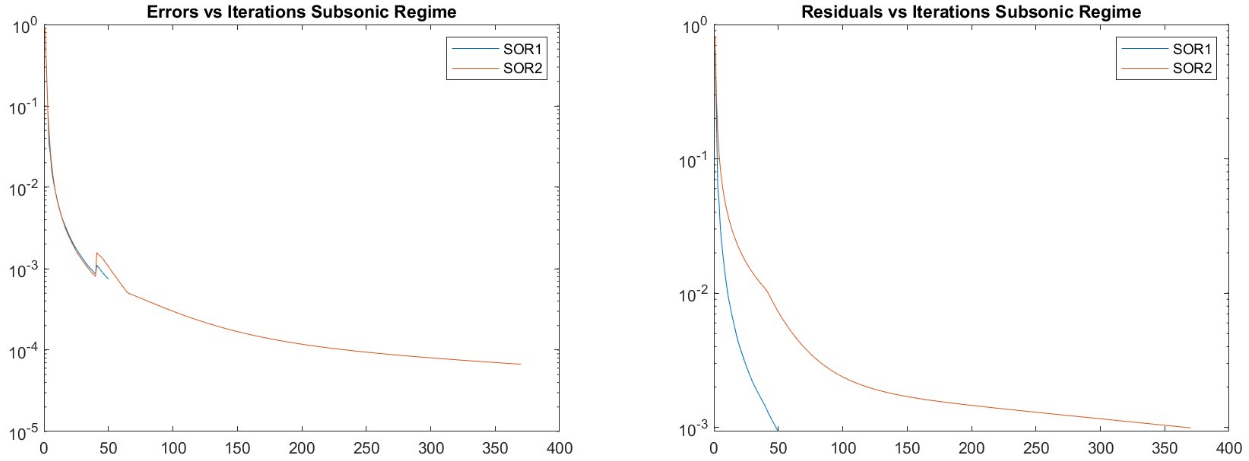
Computational Fluid Dynamics can be defined as the application of numerical methods on the resolution of fluid flow and heat transfer problems. In order to understand and apply this tool to its full potential, knowledge on what is behind the code is important and many times disregarded. This way, this project aimed to analyse the behaviour of errors and quality of solution when applying finite differences and finite volume method.

On the first part, the derivatives of two given functions were estimated utilizing a 4th order scheme and by another with a 1st order scheme. After the Taylor series expansion it was possible to estimate the error and its variation with grid size and verifying that effectively, the schemes used had the given order of approximation

On the second part of the project, a partial differential equation governing potential compressible flow was solved in a domain with boundary conditions simulating the flow around a diamond airfoil with small opening angle. Three regimes were simulated including transonic whose equation had a non-linear term which required special care in the discretization procedure. Furthermore, variation in relaxation factor and variable grid sizes, including non-uniform mesh, were evaluated. Supersonic presented the easiest convergence followed by subsonic and transonic. The application of non-uniform grids can be advantageous but its geometric inputs need to be carefully chosen for the simulation to converge. Finally, variation of the error and residuals showed that the application of a increment in the SOR process might be beneficial for faster convergence.

5 Appendix

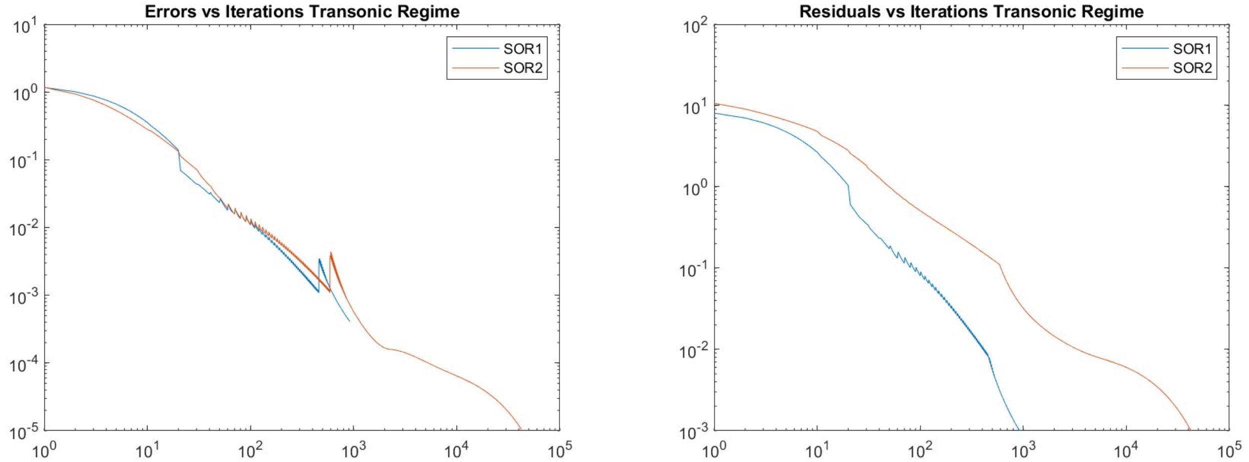
5.1 Errors and residuals evolutions over iterations for difference flow regimes



(a) Errors evolution

(b) Residuals evolution

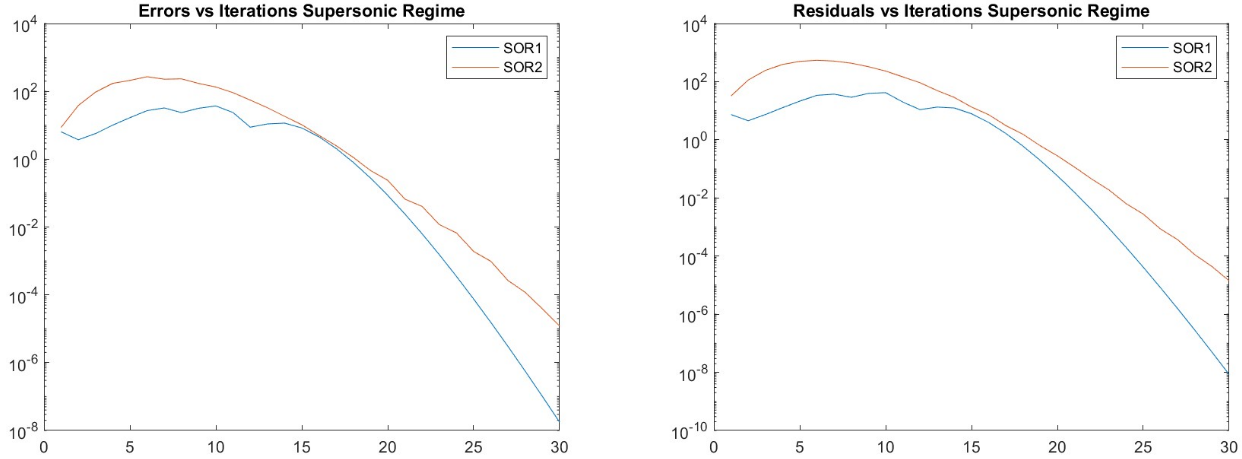
Figure 12: Errors and residuals evolution over iterations until convergence is reach for each solver for subsonic regime



(a) Errors evolution

(b) Residuals evolution

Figure 13: Errors and residuals evolution over iterations until convergence is reach for each solver for transonic regime

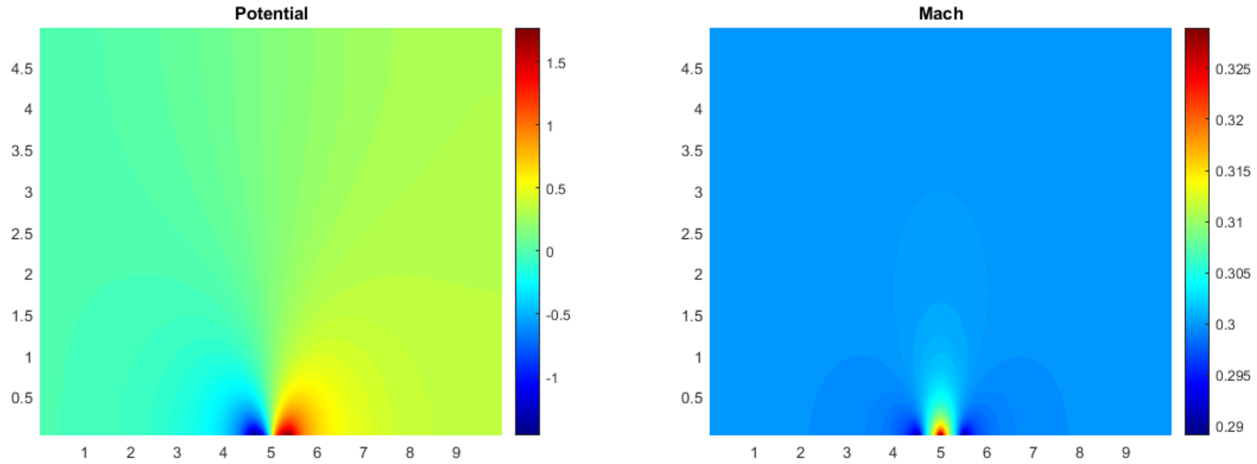


(a) Errors evolution

(b) Residuals evolution

Figure 14: Errors and residuals evolution over iterations until convergence is reach for each solver for supersonic regime

5.2 Close up of subsonic mesh nº3

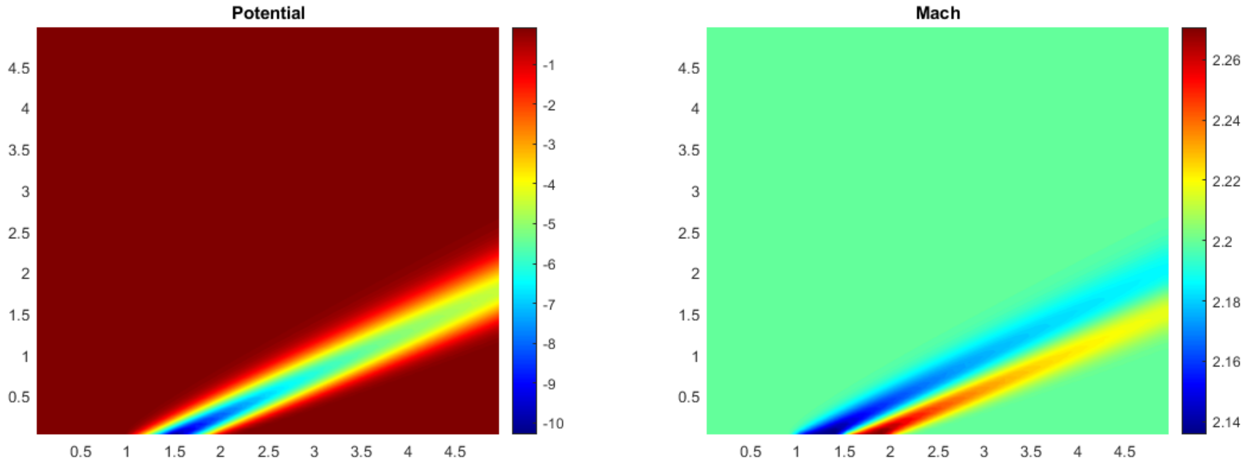


(a) Potential field of 160x80 mesh

(b) Mach field of 160x80 mes

Figure 15: Subsonic

5.3 Close up of supersonic mesh nº3

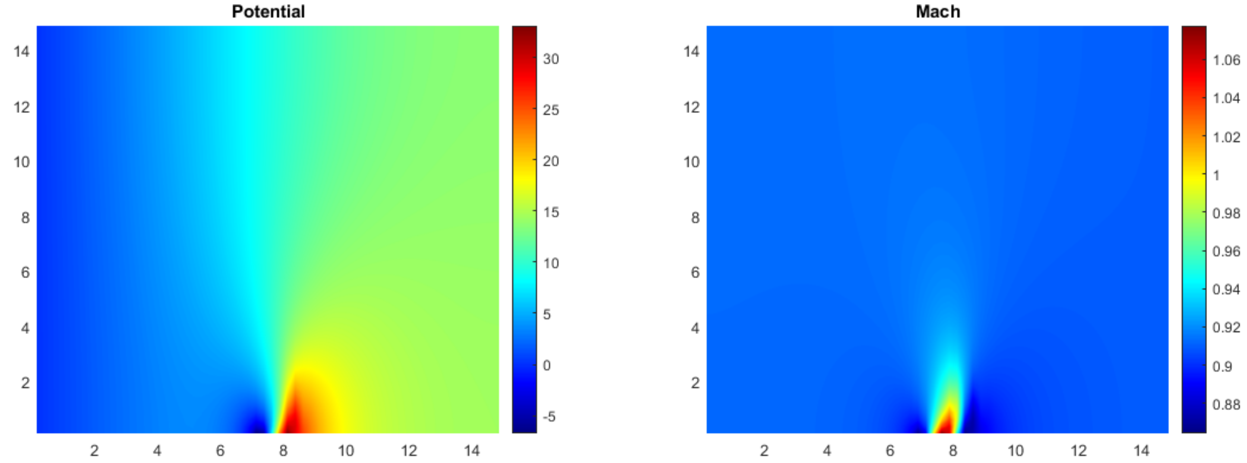


(a) Potential field of 80x80 mesh

(b) Mach field of 80x80 mesh

Figure 16: Supersonic

5.4 Close up of transonic mesh: 60x60 domain: 15 x 15 meters



(a) Potential field

(b) Mach field

Figure 17: Transonic

5.5 Close-up of meshes in section 3.5

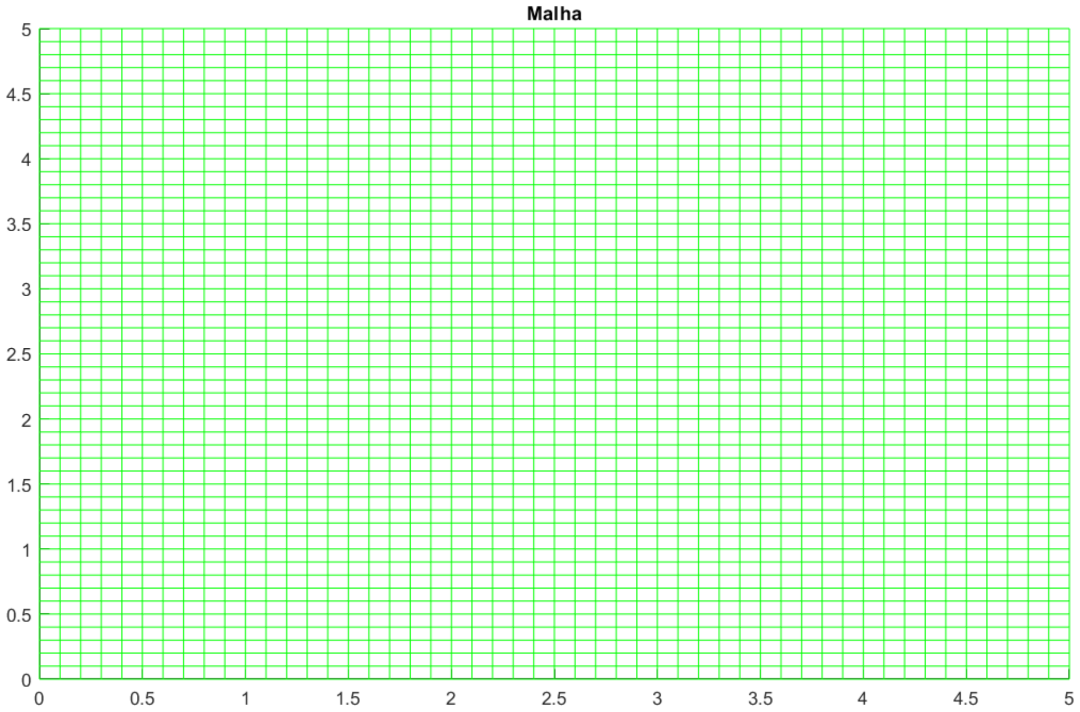


Figure 18: Uniform 50x50

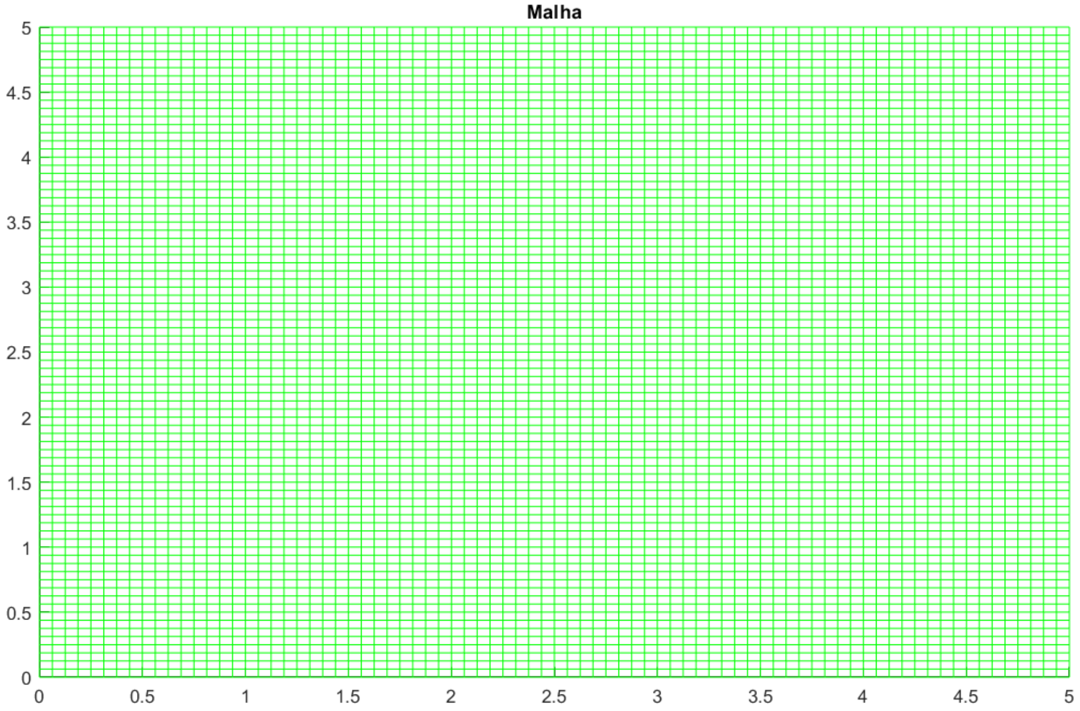


Figure 19: Uniform 80x80

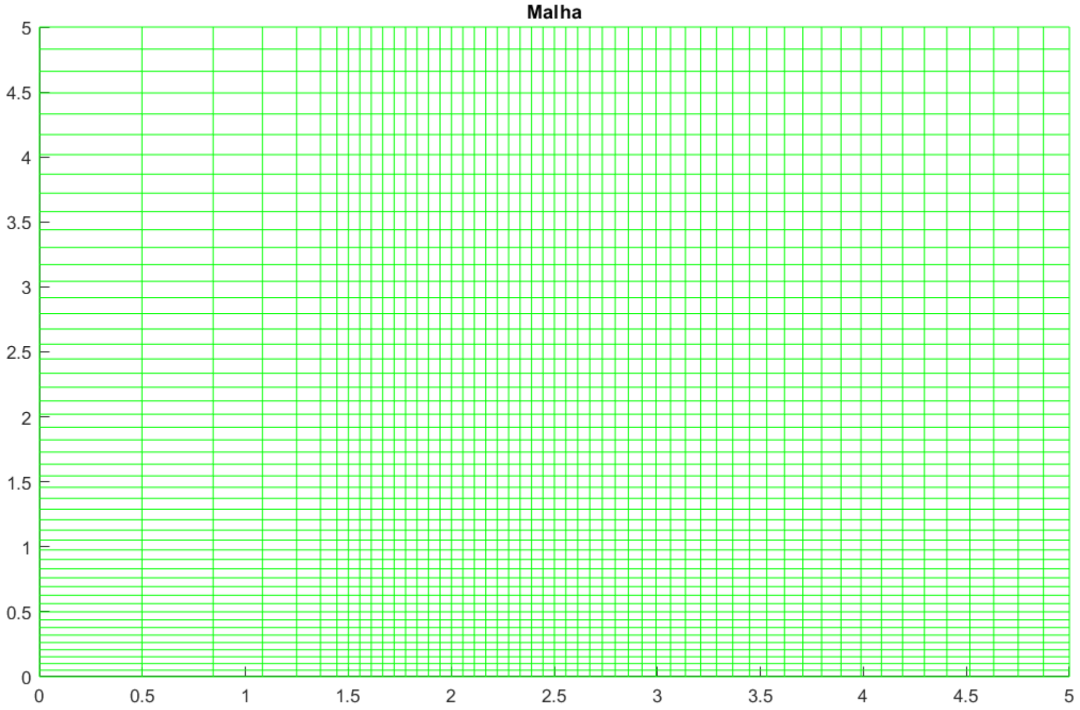


Figure 20: Non-uniform mesh 54x50