Computational Grid Generation for a NACA Airfoil

1. Abstract

This report outlines the process and results of a computational project focused on generating a C-type grid around a NACA 63-412 airfoil. The primary objective was to create a structured mesh suitable for computational fluid dynamics (CFD) simulations. The methodology involves two key stages: an initial algebraic grid generation using **Trans-Finite Interpolation (TFI)**, followed by a grid refinement and smoothing process using an **Elliptic Grid Generation** solver. The entire process is implemented in Python, utilizing libraries such as NumPy, Pandas, Matplotlib, and SciPy for numerical operations, data handling, and visualization. The final output consists of two structured grids, demonstrating the effectiveness of both techniques.

2. Introduction

In the field of computational fluid dynamics, the quality of the computational mesh is paramount to achieving accurate simulation results. The process of discretizing a physical domain into a set of smaller, non-overlapping cells or elements is known as grid generation. For external aerodynamics, such as analyzing airflow over an airfoil, creating a high-quality grid that conforms to the body's geometry while extending to a far-field boundary is a critical first step.

This project focuses on generating a 2D structured C-grid around a NACA 63-412 airfoil. The C-grid topology is chosen as it is well-suited for airfoil simulations, allowing for high resolution near the leading edge and in the wake region. The project implements and contrasts two common grid generation techniques:

1. **Trans-Finite Interpolation (TFI):** An algebraic method that generates an initial grid by interpolating from the specified boundary curves. It is computationally inexpensive but offers limited control over interior grid properties like orthogonality.
2. **Elliptic Grid Generation:** A method based on solving a system of elliptic partial differential equations (PDEs). This technique produces smoother grids with better control over cell spacing and orthogonality, albeit at a higher computational cost.

3. Methodology

The grid generation process is implemented through a series of functions within the provided Python script.

3.1. Airfoil Geometry Definition

- **Data Loading:** The coordinates of the NACA 63-412 airfoil are loaded from a CSV file (NACA63412coordinates.csv). The script skips header rows and parses the x and y coordinates.

- **Data Cleaning & Scaling:** Non-numeric values are removed, and the coordinates are scaled (divided by 1000). The script ensures the airfoil geometry is closed by appending the first coordinate to the end if it is not already present.
- **Cubic Spline Fitting (CUBICSPLINE_FITTING):** To ensure a smooth and continuous airfoil surface, a cubic spline interpolation is applied to the raw coordinate data. This function resamples the airfoil into a specified number of points (NX), which corresponds to the number of grid points along the airfoil surface.

3.2. Initial Grid Generation: Trans-Finite Interpolation (TFI)

The TFI_GRID_GENERATION function creates the initial C-grid.

- **Computational Domain:** The physical domain is mapped to a rectangular computational domain defined by coordinates (ξ, η), where ξ varies along the airfoil and wake, and η varies from the airfoil surface to the far-field boundary.
- **Boundary Definition:**
  - **Inner Boundary (Bottom):** The smoothed NACA airfoil coordinates.
  - **Outer Boundary (Top):** A semi-circular far-field boundary.
  - **Side Boundaries (Left/Right):** Straight lines connecting the airfoil's trailing edge to the far-field boundary, forming the wake cut of the C-grid.
- **Interpolation:** The TFI formula is applied to interpolate the interior grid points (X, Y) from the four defined boundary curves. This results in a structured grid that conforms to all boundaries but may lack smoothness and orthogonality in the interior.

3.3. Grid Refinement: Elliptic Grid Generation

The ELLIPTIC_GRID_GENERATION function takes the TFI grid as an initial guess and iteratively refines it.

- Governing Equations: The grid is smoothed by solving a system of Poisson's equations:
  $\alpha X_{\xi\xi} - 2\beta X_{\xi\eta} + \gamma X_{\eta\eta} = 0$
  $\alpha Y_{\xi\xi} - 2\beta Y_{\xi\eta} + \gamma Y_{\eta\eta} = 0$
  where X and Y are the physical coordinates, and the coefficients α, β, and γ are functions of the grid metrics, calculated in the abc function. These coefficients control the spacing and orientation of the grid lines.
- **Numerical Scheme:** The PDEs are discretized using second-order central differences. The resulting system of algebraic equations is solved iteratively using a Gauss-Seidel-like relaxation method.
- **Boundary Conditions:** The grid points on the airfoil surface and the far-field boundary (Dirichlet boundary conditions) remain fixed throughout the iterative process.
- **Convergence:** The iterative process continues until the maximum change in grid point locations between successive iterations falls below a specified tolerance (tol=1e-3) or a

maximum number of iterations is reached.

4. Results & Discussion

The script successfully generates and visualizes two grids for a 60x60 mesh resolution.

4.1. TFI Grid

The first plot shows the grid generated using Trans-Finite Interpolation.

- **Characteristics:** The grid lines perfectly conform to the airfoil and the outer boundary. However, grid lines emanating from the surface can be observed to be skewed, and there are regions of rapid change in cell size and shape, particularly near the leading and trailing edges. This lack of smoothness and orthogonality can reduce the accuracy of CFD solvers.

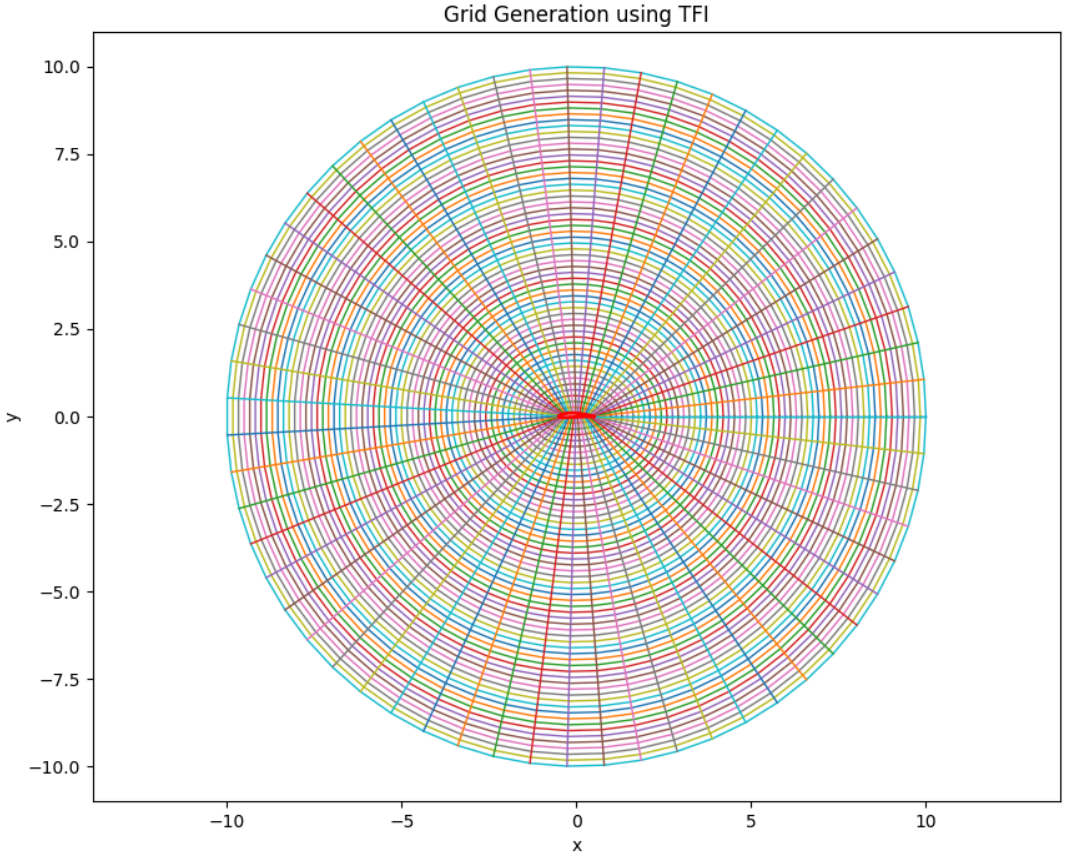
4.2. Elliptic Grid

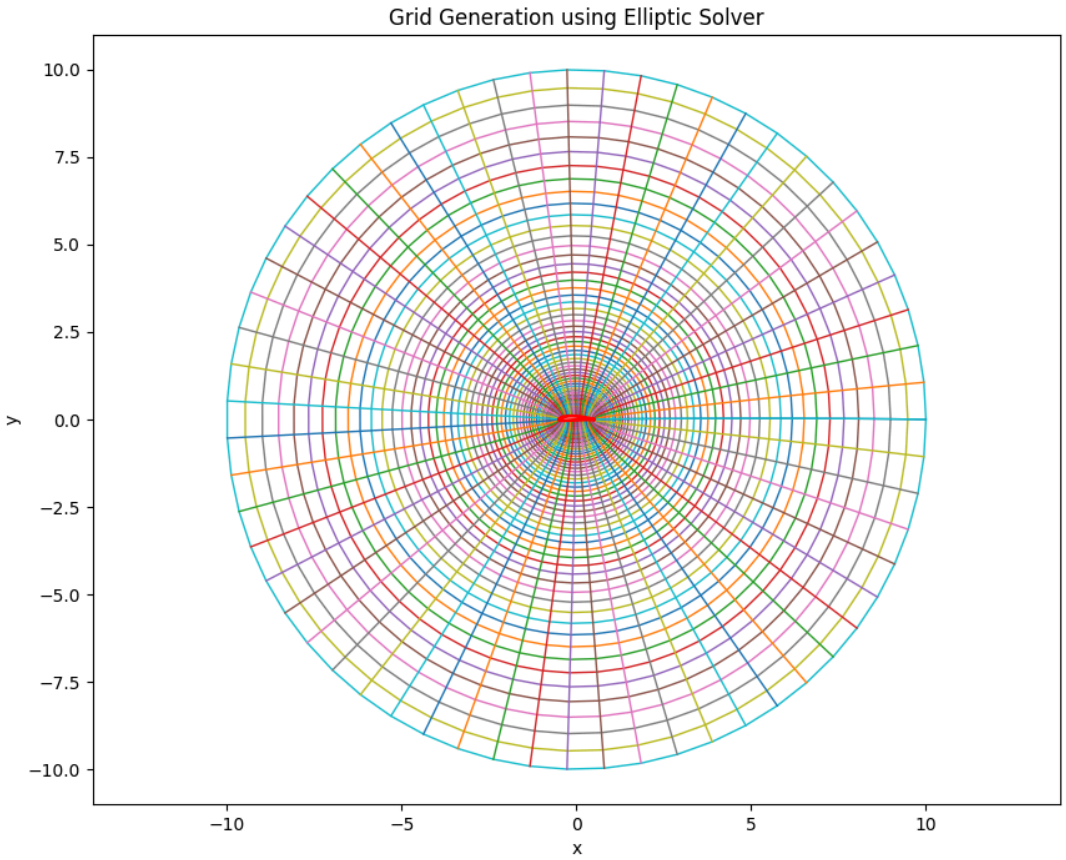
The second plot displays the grid after being processed by the elliptic solver.

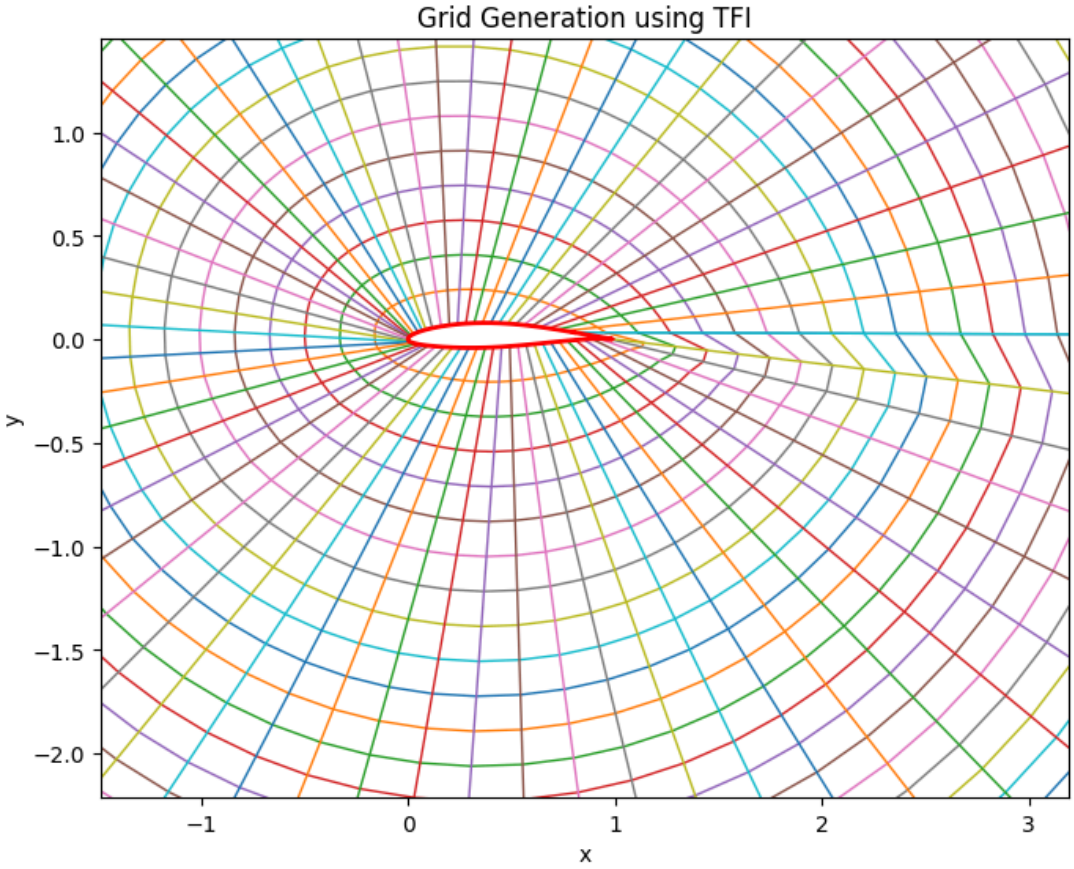
- **Characteristics:** The elliptic grid is visibly smoother than the TFI grid. The grid lines are more orthogonal to the airfoil surface, and the transition in cell size from the inner to the outer boundary is more gradual. This improved quality comes from the smoothing nature of the elliptic PDEs. The iterative solver successfully refines the initial algebraic grid into a higher-quality mesh suitable for numerical simulations.

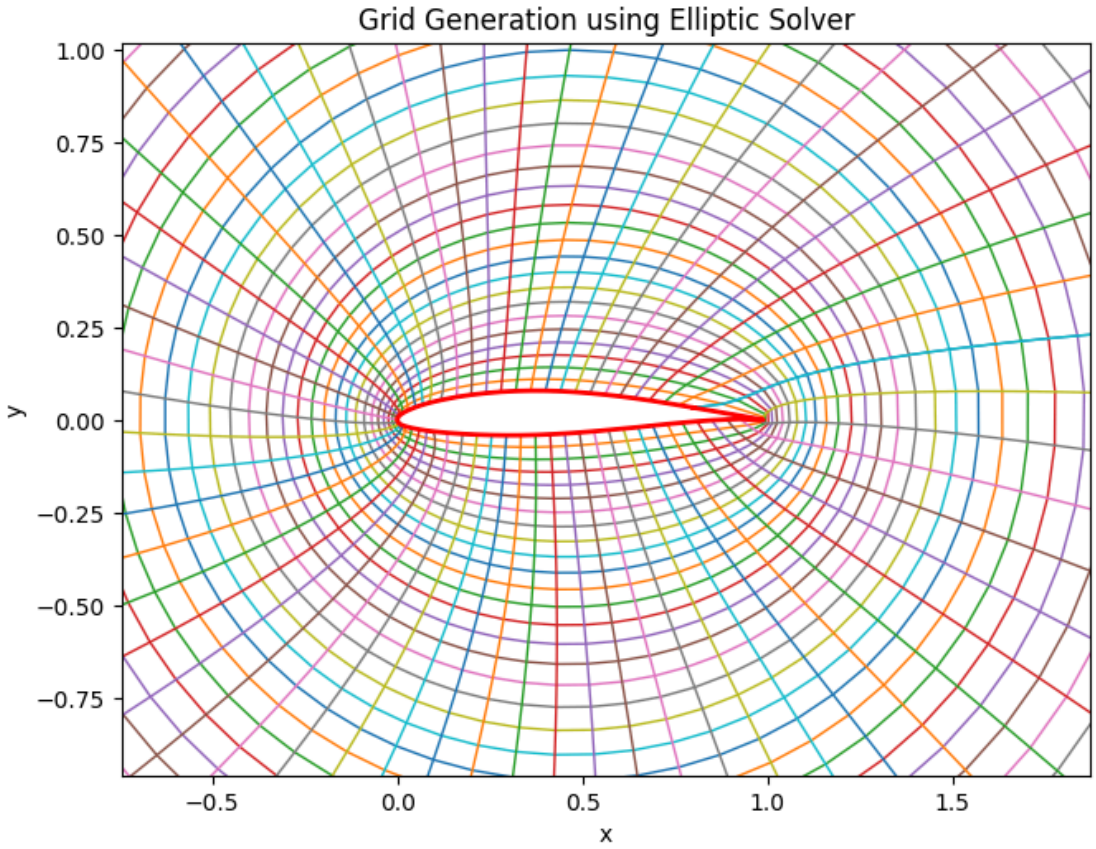
5. Conclusion

This project successfully demonstrates a two-step approach for generating a high-quality structured C-grid around a NACA airfoil. The algebraic TFI method provides a rapid and robust way to generate an initial body-fitted grid. The subsequent application of an elliptic PDE solver significantly improves the grid's key properties, namely smoothness and orthogonality. The resulting Python script serves as a practical tool and an educational example of fundamental grid generation techniques essential for computational analysis in fluid dynamics and other fields.


Grid Generation using TFI
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10
−5
0
5
10
x


Grid Generation using Elliptic Solver
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
y
−10
−5
0
5
10
x


Grid Generation using TFI
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
y
−1
0
1
2
3
x


Grid Generation using Elliptic Solver
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
y
−0.5
0.0
0.5
1.0
1.5
x