

UC Berkeley

Charlene Conrad Liebau Library Prize for Undergraduate Research

Title

Lift and Drag Calculations with a 2D Finite Difference Navier—Stokes Solver

Permalink

<https://escholarship.org/uc/item/93s7m5sr>

Authors

Patel, Parshv

Hosten-Mittermaier, Orhan

Publication Date

2026-04-01

Undergraduate

Lift and Drag Calculations with a 2D Finite Difference Navier–Stokes Solver

Parshv Patel^{1,*} and Orhan Hosten-Mittermaier^{2,†}

¹*College of Computing, Data Science, and Society, University of California, Berkeley*

²*Department of Physics, University of California, Berkeley*

In aerospace design, determining the airflow characteristics and force production around aircraft components is vital for evaluating performance, efficiency, and safety. For wing design in particular, accurate computational modeling of lift and drag can save the time and resources that would otherwise be spent on experimental testing of numerous candidate geometries. We present a finite difference numerical solver for the two-dimensional incompressible Navier–Stokes equations that models viscous fluid flow around solid airfoils. The computational approach employs explicit Euler time-stepping for the momentum equations, an ω -relaxed Jacobi iterative method for solving the pressure Poisson equation, and a projection method to enforce incompressibility. We introduce a flexible framework for generating smooth airfoil geometries by interpolating through control points using Piecewise Cubic Hermite Interpolating Polynomials (PCHIP), which preserves monotonicity and avoids the spurious oscillations inherent to cubic spline methods. Force calculations on the airfoil boundary utilize Voronoi-based influence sections to compute pressure-induced lift and drag. The primary challenge encountered was the numerical stability at higher airspeeds; computational constraints on the spatial and temporal step lengths prevented stable results above 0.1 m/s. The simulations were validated against canonical results, including increased drag from thicker airfoils and at higher airspeeds, as well as negligible lift from symmetric airfoils at 0° angle of attack. We further demonstrate that both custom-parameterized and four-digit NACA series airfoils produce significant lift within our framework. Of all configurations tested, NACA 5315 achieved the best lift-to-drag ratio of 1.479 at 0.1 m/s. This solver provides a foundation for computational aerodynamic analysis and demonstrates the feasibility of parameter-driven wing shape optimization.

I. INTRODUCTION AND PHYSICS MOTIVATION

Problem statement. Understanding the aerodynamic forces acting on aircraft components is fundamental to aerospace engineering and vehicle design. When air flows around an airfoil, the resulting pressure distribution generates two primary forces: *lift*, which acts perpendicular to the freestream velocity and enables flight, and *drag*, which acts parallel to the flow and opposes motion. The lift-to-drag ratio (L/D) serves as a critical metric for aerodynamic efficiency, directly influencing fuel consumption, range, and overall aircraft performance.

Relevance of computational methods. Analytically solving the Navier–Stokes equations that govern fluid dynamics in general cases remains one of the outstanding challenges in mathematical physics; indeed, the existence and smoothness of solutions in three dimensions constitutes one of the Millennium Prize Problems [1]. Consequently, the primary methods for analyzing airflow are experimental wind tunnel testing and computational fluid dynamics (CFD) simulations based on numerical approximations to the governing equations. While wind tunnel experiments provide ground-truth measurements, they are expensive, time-consuming, and impractical for exploring large design spaces. CFD simulations offer a complementary approach that enables rapid iteration through design parameters at a fraction of the cost, and

have therefore gained massive prominence in aerospace design [3].

Why it matters. Computational aerodynamics serves several critical functions in modern engineering and physics. First, CFD enables systematic exploration of wing geometries to maximize L/D ratios, reducing the need for expensive prototype fabrication and wind tunnel testing and thereby accelerating the design cycle [3]. Second, numerical simulations reveal detailed velocity and pressure fields that are difficult or impossible to measure experimentally, providing insight into phenomena such as boundary layer separation, vortex shedding, and turbulent transition [4]. Third, validated CFD tools enable engineers to predict aerodynamic performance of novel configurations before physical prototypes exist [5]. Beyond aerospace, the numerical methods developed for CFD find applications in weather prediction, pollutant transport modeling, blood flow simulation, and structural wind loading analysis [18].

Scope of this work. In this work, we present a two-dimensional finite difference Navier–Stokes solver for computing lift and drag forces on arbitrarily shaped airfoils. The solver models incompressible viscous airflow—a valid approximation for airspeeds below approximately Mach 0.3, or about 100 m/s [2, 18]. This regime encompasses general aviation aircraft such as Cessna single-engine planes, making the approach relevant for practical aerodynamic analysis.

Contributions. This project makes the following contributions:

- A complete 2D incompressible Navier–Stokes solver using finite difference discretization with explicit

* parshvpatel_0910@berkeley.edu

† orhanhm@berkeley.edu

time-stepping, ω -relaxed Jacobi iteration for pressure, and physically meaningful boundary conditions.

- A flexible airfoil geometry generation framework using PCHIP interpolation through parameterized control points, enabling systematic and continuous shape variation.
- A dynamic time stepping scheme based on the Courant–Friedrichs–Lewy (CFL) condition to enhance numerical stability.
- A Voronoi-based force calculation procedure that assigns pressures to airfoil boundary sections.
- Validation against canonical aerodynamic results: negligible lift for symmetric airfoils at zero angle of attack and monotonically increasing drag for thicker profiles.
- Demonstration that both custom-parameterized and NACA-standard airfoils generate physically meaningful lift-to-drag ratios.

Paper roadmap. Section II presents the governing equations, underlying theory, and prior work. Section III describes the computational domain, fluid properties, boundary conditions, and airfoil geometry generation. Section IV details the numerical algorithms for velocity advancement, pressure solving, velocity projection, and force calculation. Section V presents validation studies and lift-to-drag measurements for various airfoil configurations. Section VI analyzes limitations and compares results to expected physical behavior. Section VIII provides information for reproducing the results. Section VII summarizes findings and outlines future directions.

II. BACKGROUND AND RELATED WORK

Governing equations. The motion of a viscous, incompressible fluid is governed by the Navier–Stokes equations, expressing conservation of momentum and mass. In two dimensions these take the form

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla) \vec{u} = -\frac{1}{\rho} \nabla \mathbf{p} + \nu \nabla^2 \vec{u} + \vec{a}, \quad (1)$$

$$\nabla \cdot \vec{u} = 0, \quad (2)$$

where $\vec{u}(x, y, t) = (u, v)^T$ is the velocity field with horizontal component u and vertical component v , $\mathbf{p}(x, y, t)$ is the pressure field, ρ is the (constant) fluid density, ν is the kinematic viscosity, and $\vec{a} = \vec{F}_b/\rho$ represents body accelerations (e.g., gravity or externally applied forcing).

Equation (1) is the momentum equation, whose terms represent (from left to right) local acceleration, convective acceleration, the pressure gradient, viscous diffusion,

and body forces. Equation (2) is the incompressibility constraint, requiring the velocity field to be divergence-free.

The incompressibility assumption is valid when the Mach number $\text{Ma} = U/c_s \ll 1$, where U is the characteristic flow velocity and $c_s \approx 340 \text{ m s}^{-1}$ is the speed of sound in air. For $\text{Ma} < 0.3$ (approximately 100 m s^{-1}), density variations are negligible and Eqs. (1)–(2) accurately describe the flow [2].

The Reynolds number $\text{Re} = UL/\nu$ characterizes the ratio of inertial to viscous forces, where L is a characteristic length scale (typically the chord length). Low Reynolds number flows ($\text{Re} < 10^3$) are dominated by viscosity and remain laminar, while high Reynolds number flows ($\text{Re} > 10^6$) exhibit turbulence requiring specialized modeling [4].

The pressure force on a surface in two dimensions is given by

$$\vec{F}_{\text{net}} = \int_C \mathbf{p} \hat{n}_{\text{in}} ds, \quad (3)$$

where $\hat{n}_{\text{in}}$ is the inward unit normal to the surface. This integral is used to compute the net aerodynamic force acting on the airfoil.

Numerical methods for CFD. Three primary spatial discretization approaches exist for solving the Navier–Stokes equations numerically. Finite difference methods (FDM) approximate derivatives using Taylor series expansions on structured grids; they are simple to implement but limited to regular geometries [18]. Finite volume methods (FVM) integrate conservation laws over control volumes, ensuring global conservation, and dominate commercial CFD codes [6]. Finite element methods (FEM) expand solutions in basis functions over unstructured meshes, offering flexibility for complex geometries at greater computational cost [7].

We opted for finite differences owing to their conceptual simplicity and straightforward implementation on rectangular grids [16], without reliance on specialized CFD or PDE libraries. The choice of derivative stencils—specifically a mix of central and upwind-biased approximations—was informed by successful early CFD work [17, 20], and the treatment of pressure solving and boundary conditions follows standard CFD textbook practice [18, 19].

For time integration, the coupling between velocity and pressure in incompressible flows requires special treatment. The *projection method*, introduced by Chorin [8], addresses this by (1) advancing velocities without the pressure gradient to obtain provisional velocities, (2) solving a Poisson equation for pressure, and (3) projecting velocities onto the divergence-free subspace. This approach decouples the momentum and continuity equations and enables efficient iterative solution [9].

Prior work. The finite difference approach to CFD has a rich history dating to the marker-and-cell (MAC) method of Harlow and Welch [10]. Modern textbook

treatments by Ferziger and Perić [18] and Fletcher [11] provide comprehensive coverage of discretization schemes and iterative solvers.

For airfoil aerodynamics, the NACA four-digit series developed by the National Advisory Committee for Aeronautics in the 1930s remains widely used for standardized testing [12]. The four digits encode maximum camber, location of maximum camber, and maximum thickness as percentages of chord length.

Interpolation methods for smooth curve generation include cubic splines [13], which minimize curvature but may exhibit Runge-type oscillations, and monotonicity-preserving methods such as PCHIP [14] that sacrifice global smoothness for local shape preservation.

III. COMPUTATIONAL SETUP

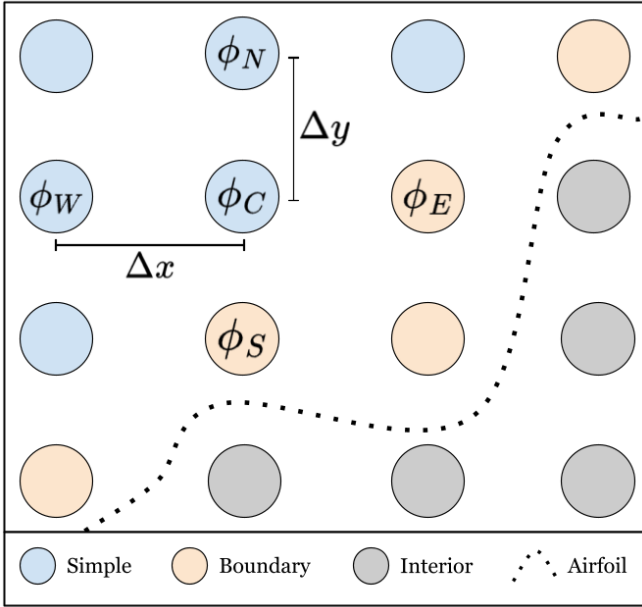


FIG. 1. **System setup and definitions.** Notational conventions for grid spacing and the five-point stencil are indicated. Classification of fluid points is determined by the airfoil boundary: boundary fluid points are those whose five-point stencil contains at least one interior (solid) point.

Domain and grid. The flow is simulated in a rectangular domain of size $L_x \times L_y = 2 \text{ m} \times 1 \text{ m}$, discretized on a uniform Cartesian grid with $n_x \times n_y$ points. After resolution studies (Sec. V), a grid of $n_x = 800$, $n_y = 400$ was selected, yielding spacings $\Delta x = \Delta y = 2.5 \text{ mm}$. The domain is centered vertically about $y = 0$, spanning $y \in [-L_y/2, L_y/2]$. One additional padding cell is included on the left and right boundaries to simplify the implementation of boundary conditions.

To implement the finite difference method, each field variable (u , v , $\mathbf{p}$) is stored at every grid point at each timestep. Much of the analysis relies on updating values

at each grid point based on its immediate neighbors in each coordinate direction, composing a five-point stencil (Fig. 1). For a field variable ϕ at timestep n , the values at each stencil point are denoted ϕ_C^n (center), ϕ_N^n (north), ϕ_S^n (south), ϕ_E^n (east), and ϕ_W^n (west).

Fluid properties. Air at 3000 m altitude is modeled using the U.S. Standard Atmosphere [15]: density $\rho = 0.9093 \text{ kg m}^{-3}$, dynamic viscosity $\mu = 1.694 \times 10^{-5} \text{ Pa s}$, and kinematic viscosity $\nu = \mu/\rho = 1.863 \times 10^{-5} \text{ m}^2 \text{ s}^{-1}$. A chord length of 0.7 m and maximum thickness between 0.1 m and 0.2 m were chosen to correspond with the typical scale of small aircraft wings.

Grid masking. A continuous airfoil is represented as a dense set of points forming a closed boundary. Field variables at points interior to the wing are neglected, as there is no physical interpretation for velocity or pressure within a solid body. Three Boolean masks are constructed over the computational grid:

- **wing_mask:** points strictly inside the airfoil, determined by path containment (matplotlib `Path` class).
- **boundary_mask:** fluid points adjacent to at least one interior point, identified via five-point stencil neighbor checking.
- **simple_points:** standard fluid points requiring no special boundary treatment.

These masks enable efficient vectorized computation while correctly handling the immersed boundary.

Boundary conditions. The following boundary conditions are employed:

- **Periodic boundaries in y :** The top and bottom edges are connected, eliminating wall effects and simulating an effectively infinite span. This is appropriate for two-dimensional simulations in which three-dimensional tip effects are neglected.
- **Zero-gradient (Neumann) conditions in x :** The leftmost and rightmost grid columns are set equal to their interior neighbors, approximating fully developed flow at the inlet and outlet.
- **No-slip condition on the airfoil:** At the wing surface and interior, $\vec{\mathbf{u}} = 0$. This models the physical adhesion of a viscous fluid to a solid surface [16, 18].
- **Neumann pressure condition at the airfoil:** The normal pressure gradient vanishes at the airfoil boundary ($\partial p / \partial n = 0$), ensuring zero mass flux through solid walls. In the Jacobi solver, this is implemented by replacing any neighbor that lies inside the wing with the local (center) pressure value.

Flow initialization and ramping. Simulations begin from rest ($\bar{\mathbf{u}} = 0$, $p = 0$) with a uniform body acceleration a_x applied to fluid cells outside the airfoil. This gradually accelerates the flow toward a target velocity $u_{\max}$ without the numerical instabilities that would result from an impulsive start at finite velocity. The acceleration is maintained until the mean horizontal velocity in the fluid region reaches $u_{\max}$, after which body forcing is removed.

Airfoil geometry generation. A central contribution of this work is a flexible framework for generating smooth airfoil shapes from parameterized control points. Rather than restricting attention to standard airfoil families, we enable continuous variation of wing shape through the following procedure:

1. **Control point specification.** Upper and lower surface shapes are defined by arrays of height parameters $\{h_i^{\text{upper}}\}$ and $\{h_i^{\text{lower}}\}$ at evenly spaced chordwise locations. Endpoints are fixed at zero to ensure sharp leading and trailing edges.
2. **Parameter scaling.** Raw parameters are mapped to the interval $(0, 1)$ via a sigmoid transformation,

$$h_{\text{scaled}} = \frac{1}{1 + e^{-4h}}, \quad (4)$$

bounding the control points to physically reasonable values while maintaining differentiability for potential gradient-based optimization.

3. **Interpolation.** Smooth curves through the control points are generated using PCHIP [14]. Unlike cubic splines, PCHIP preserves monotonicity between control points, eliminating spurious oscillations that would create non-physical surface undulations.
4. **Thickness normalization.** The interpolated surfaces are scaled so that the maximum cross-sectional width equals a specified thickness parameter, enabling fair comparison between different shapes.

For validation against standard results, NACA four-digit airfoils are also generated following the standard formulas [12]:

$$y_t = 5t(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4), \quad (5)$$

where t is the maximum thickness as a fraction of chord, and the camber line is defined piecewise based on maximum camber m and its chordwise location p . These profiles are passed as dense height arrays to the airfoil class, bypassing the sigmoid scaling.

IV. METHODS

The general procedure for analyzing an airfoil is as follows. At each time iteration, velocities are stepped forward and the pressure field is constructed so as to make the subsequent velocity field satisfy mass conservation (Eq. 2). A constant horizontal body acceleration ramps the airspeed until the spatially averaged horizontal velocity reaches $u_{\max}$, at which point time stepping is halted and forces on the airfoil are computed.

Computational pipeline. Each time iteration comprises the following steps:

1. Compute an adaptive time step satisfying stability constraints.
2. Advance velocities explicitly using the discretized momentum equation.
3. Apply boundary conditions to the provisional velocities.
4. Construct the right-hand side of the pressure Poisson equation.
5. Solve the pressure Poisson equation iteratively.
6. Project velocities to enforce the divergence-free condition.
7. (After reaching $u_{\max}$) Compute forces on the airfoil from the pressure field.

Dynamic time stepping. Explicit Euler time integration is employed with an adaptive time step determined by the Courant–Friedrichs–Lewy (CFL) condition:

$$\Delta t = C \times \min\left(\frac{\Delta x^2}{\nu}, \frac{\Delta x}{|\bar{\mathbf{u}}|_{\max}}\right), \quad (6)$$

where $C \in (0, 1)$ is a safety factor. The first constraint (viscous stability) prevents diffusive instabilities, while the second (advective stability) ensures that information does not propagate more than one cell per time step. A value of $C = 0.3$ was found to provide a good balance between stability and simulation speed. The time step is recomputed before each velocity advancement.

Velocity advancement. Inspecting the momentum equation (1), the time derivative is discretized with a forward difference and all spatial derivatives are evaluated at the current timestep, yielding an explicit update for the velocity:

$$\bar{\mathbf{u}}^{n+1} = \bar{\mathbf{u}}^n - \Delta t \left[(\bar{\mathbf{u}}^n \cdot \nabla) \bar{\mathbf{u}}^n + \frac{1}{\rho} \nabla p^n - \nu \nabla^2 \bar{\mathbf{u}}^n - \bar{\mathbf{a}} \right]. \quad (7)$$

Spatial derivatives are approximated as follows: an upwind (backward) difference is used for the horizontal advective term $u \partial \phi / \partial x$ to enhance stability by respecting the predominant flow direction; central differences are used for all remaining first-order derivatives; and the

standard second-order central stencil is used for the diffusion (Laplacian) terms.

Expanding the horizontal velocity update for an interior fluid point,

$$u_{i,j}^{n+1} = u_{i,j}^n - \Delta t \left[u_{i,j}^n \frac{u_{i,j}^n - u_{i-1,j}^n}{\Delta x} + v_{i,j}^n \frac{u_{i,j+1}^n - u_{i,j-1}^n}{2\Delta y} + \frac{1}{\rho} \frac{p_{i+1,j}^n - p_{i-1,j}^n}{2\Delta x} - \nu \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{\Delta x^2} - \nu \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{\Delta y^2} - a_x \right], \quad (8)$$

with an analogous expression for $v_{i,j}^{n+1}$ (see Appendix A). After computing provisional velocities, the no-slip condition is enforced by setting $\vec{u} = 0$ on the `boundary_mask` and `wing_mask` regions.

Listing 1. Implementation of the horizontal velocity update.

```
u_star = u_C - dt * (
    u_C * (u_C - u_W) / dx
    + v_C * (u_N - u_S) / (2*dy)
    + (p_E - p_W) / (rho*2*dx)
    - nu * (u_E - 2*u_C + u_W) / dx**2
    - nu * (u_N - 2*u_C + u_S) / dy**2
    - ax)
```

Pressure Poisson equation. The incompressible Navier–Stokes system in two dimensions has three unknowns (u, v, p) and three equations: two momentum equations and the continuity constraint. Since the momentum equations govern the velocity evolution, the pressure must be determined from the remaining constraint. Taking the divergence of the projection step $\vec{u}^{n+1} = \vec{u}^* - (\Delta t/\rho) \nabla p$ and requiring $\nabla \cdot \vec{u}^{n+1} = 0$ yields the pressure Poisson equation:

$$\nabla^2 p = \frac{\rho}{\Delta t} \nabla \cdot \vec{u}^*, \quad (9)$$

whose right-hand side (RHS) quantifies the divergence that must be eliminated by the pressure gradient. The divergence is computed using central differences:

$$(\nabla \cdot \vec{u}^*)_{i,j} = \frac{u_{i+1,j}^* - u_{i-1,j}^*}{2\Delta x} + \frac{v_{i,j+1}^* - v_{i,j-1}^*}{2\Delta y}, \quad (10)$$

with periodic wrapping for the j indices at the vertical domain boundaries.

Jacobi iteration for pressure. Equation (9) is solved using the ω -relaxed Jacobi method, initialized with the pressure field from the previous timestep. Discretizing the Laplacian with second-order central differences and

rearranging for the central value gives the iteration

$$p_{i,j}^{(k+1)} = (1 - \omega) p_{i,j}^{(k)} + \frac{\omega}{2(\Delta x^2 + \Delta y^2)} \times \left[\Delta y^2 (p_{i+1,j}^{(k)} + p_{i-1,j}^{(k)}) + \Delta x^2 (p_{i,j+1}^{(k)} + p_{i,j-1}^{(k)}) - \Delta x^2 \Delta y^2 \text{RHS}_{i,j} \right], \quad (11)$$

where $\omega \in (0, 2)$ is the relaxation parameter. A value of $\omega = 0.99$ (slight over-relaxation) was used to accelerate convergence.

For computational efficiency, a sparse matrix representation of the Jacobi update is precomputed, enabling vectorized updates via matrix–vector multiplication. The matrix encodes (i) the standard five-point stencil weights for fluid cells, (ii) periodic connectivity for the top and bottom boundaries, and (iii) the Neumann condition at the airfoil, implemented by transferring the coefficient of any neighbor lying inside the wing to the center point. Iteration continues until the maximum pressure change falls below an absolute tolerance of 10^{-7} Pa or a relative tolerance of 10^{-3} times the interquartile pressure range ($p_{95} - p_5$), or until a maximum of 1000 iterations is reached. After convergence, the spatial mean pressure is subtracted to prevent drift.

Listing 2. Implementation of the Jacobi pressure iteration.

```
it = 0
while it < max_iters:
    p_new = jacobi_matrix @ p \
        - (diff_prod / (2*diff_sum)) * RHS
    max_delta = np.max(np.abs(p_new - p))
    p = p_new
    it += 1
    pressure_scale = np.percentile(p, 95) \
        - np.percentile(p, 5)
    if (max_delta <= tol
        or max_delta <= rel_tol * pressure_scale)
        :
        break
```

Velocity projection. The final divergence-free velocity field is obtained by subtracting the pressure gradient from the provisional velocities:

$$\vec{u}^{n+1} = \vec{u}^* - \frac{\Delta t}{\rho} \nabla p, \quad (12)$$

using central differences for the gradient. Near the airfoil, if a neighbor cell lies inside the wing it is replaced with the center cell’s pressure, consistent with the Neumann boundary condition. The no-slip condition is re-applied after projection.

Force calculation. Aerodynamic forces arise from pressure acting normal to the airfoil surface. To assign a pressure to each section of the boundary, we proceed as follows:

1. For each point on the discretized wing surface, the nearest grid boundary point is identified.

2. Wing surface points sharing a nearest boundary neighbor are grouped into “influence sections,” forming a Voronoi-like partition of the boundary.
3. For each influence section, the pressure force contribution is

$$\vec{F}_i = p_i \Delta S_i \hat{n}_i, \quad (13)$$

where p_i is the pressure at the nearest boundary fluid point, ΔS_i is the arc length of the section, and $\hat{n}_i$ is the inward-pointing unit normal.

4. The total force is obtained by summation:

$$\vec{F}_{\text{total}} = \sum_i \vec{F}_i. \quad (14)$$

The horizontal component is the drag; the vertical component is the lift.

The inward normal is computed by rotating the tangent vector (connecting section endpoints) by 90° clockwise. Because a dense grid is used, the airfoil is split into approximately 100–1000 sections, justifying the approximation that each section is locally linear.

Hyperparameter summary. Table I collects the principal simulation parameters. The physical domain size, chord length, and altitude were chosen to correspond with a small general-aviation aircraft at cruising altitude (3000 m). The grid resolution of 800×400 was the densest that could feasibly be used for the large number of production runs required (Sec. V). Airfoil boundaries are resolved with 10 000 points, a conservative choice that does not substantially affect runtime.

TABLE I. Principal simulation parameters.

Parameter	Value
Domain size ($L_x \times L_y$)	2 m $\times$ 1 m
Grid resolution ($n_x \times n_y$)	800 $\times$ 400
Grid spacing ($\Delta x = \Delta y$)	2.5 mm
Chord length	0.7 m
Airfoil thickness	0.1 m–0.2 m
Air density (ρ)	0.9093 kg m $^{-3}$
Kinematic viscosity (ν)	1.863 $\times 10^{-5}$ m ^{2} s $^{-1}$
CFL safety factor (C)	0.3
Jacobi relaxation (ω)	0.99
Max. Jacobi iterations	1000
Absolute pressure tolerance	10 $^{-7}$ Pa
Relative pressure tolerance	10 $^{-3}$
Airfoil boundary points	10 000

V. RESULTS

A. Grid Resolution Study

To assess numerical accuracy, simulations were performed at grid resolutions ranging from 200×100 to

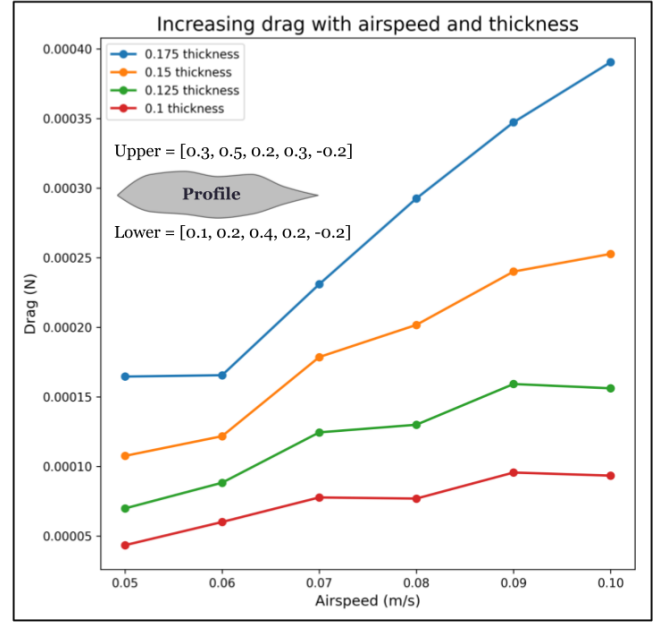


FIG. 2. **Drag validation.** Drag values for a representative airfoil at thicknesses ranging from 0.1 m to 0.175 m and airspeeds from 0.05 m/s to 0.1 m/s. Speed was ramped to 0.1 m/s using the standard procedure, and drag was recorded once the airspeed exceeded each threshold. The airfoil was generated using the displayed upper and lower parameters with PCHIP interpolation.

3200 $\times$ 1600. The lift-to-drag ratio converges as resolution increases, with diminishing returns beyond 800×400 . This resolution was therefore selected for all production runs as a balance between accuracy and computational cost; the L/D values at 800×400 lie within 3% of those obtained at 3200×1600 .

B. Reproducing Expected Results

As an initial validation step, the simulation is tested in regimes where the qualitatively correct answer is known, establishing internal consistency and physical fidelity before drawing broader aerodynamic conclusions.

1. Drag Validation

Classical aerodynamic theory predicts that drag increases monotonically with both airfoil thickness and airspeed, approximately linearly for small perturbations in either parameter [12]. Testing a representative profile at thicknesses from 0.1 m to 0.175 m and airspeeds from 0.05 m/s to 0.1 m/s, the simulation reproduces both trends: drag increases with thickness and with airspeed, and the dependence on airspeed at fixed thickness is approximately linear (Fig. 2).

To further isolate the effect of geometry, three profiles of the same planform are simulated under identical flow conditions ($u = 0.1$ m/s), varying only thickness. The resulting drag values are $D = 2.27 \times 10^{-5}$ N (thin), 1.50×10^{-4} N (medium), and 3.26×10^{-4} N (thick), spanning more than one order of magnitude (Figs. 3–5). The monotonic increase is consistent with the established relationship between chord thickness, frontal projected area, and pressure (form) drag: a thicker profile presents a larger cross-section to the oncoming flow, steepening the adverse pressure gradient near the trailing edge and substantially increasing form drag [12]. Viscous (skin-friction) drag also grows with wetted area, reinforcing the trend. These results confirm that the solver correctly captures the dominant drag mechanisms.

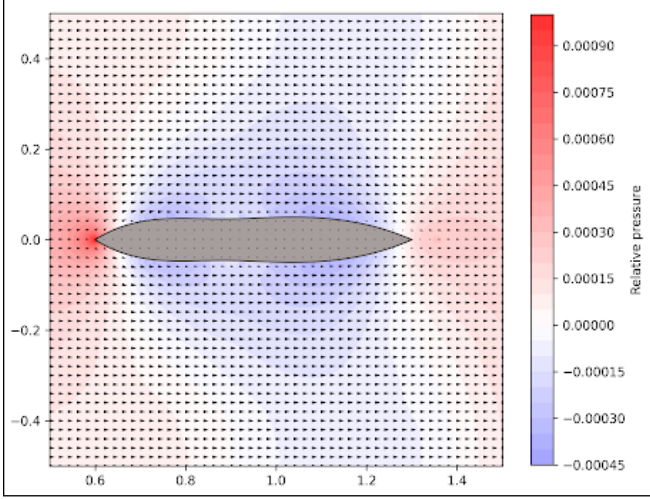


FIG. 3. **Thin profile.** Pressure and velocity fields at $u = 0.1$ m/s. Drag: $D = 2.27 \times 10^{-5}$ N.

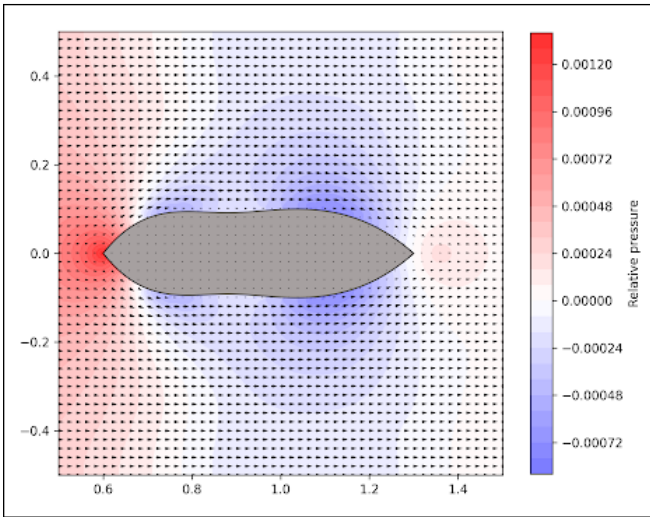


FIG. 4. **Medium profile.** Pressure and velocity fields under identical flow conditions. Drag: $D = 1.50 \times 10^{-4}$ N.

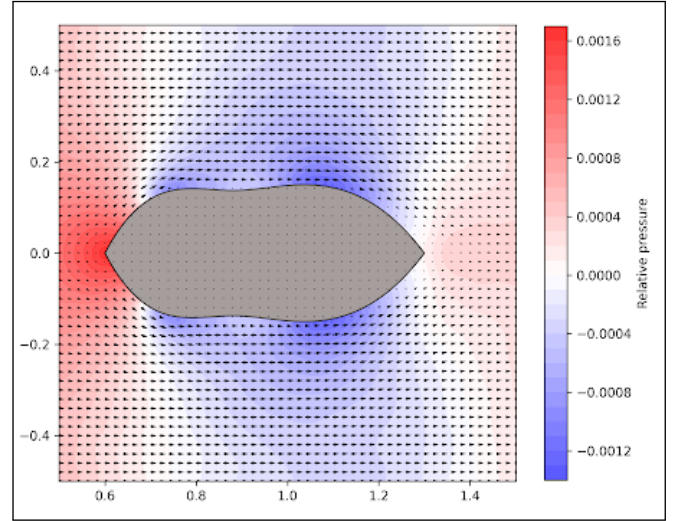


FIG. 5. **Thick profile.** Pressure and velocity fields under identical flow conditions. Drag: $D = 3.26 \times 10^{-4}$ N—over an order of magnitude larger than the thin case.

2. Lift Validation: Symmetric Airfoils at Zero Angle of Attack

A fundamental result of aerodynamic theory is that a symmetric airfoil at zero angle of attack generates no net lift: the upper and lower surfaces present identical geometry to the flow, yielding mirror-image pressure distributions that cancel exactly in the vertical direction. The simulation confirms this across multiple distinct symmetric profiles.

For the primary validation case (upper and lower parameters [0.3, 0.0, 0.1], thickness 0.175 m, PCHIP interpolation), the solver yields $L = -1.47 \times 10^{-19}$ N at 0.1 m/s (Fig. 6)—indistinguishable from zero to any physical precision. The velocity and pressure fields are visually symmetric about the chord line, as expected.

Two additional symmetric profiles of geometrically distinct shapes corroborate this result. A streamlined profile—featuring pointed leading and trailing edges with a single central bulge—yields $L = -4.18 \times 10^{-19}$ N (Fig. 7). A bimodal profile—exhibiting a prominent leading-edge bulge, a narrow midsection, and a secondary trailing-edge bulge—yields $L = -1.62 \times 10^{-18}$ N (Fig. 8). In both cases the velocity vectors confirm smooth, attached flow along the chord with no preferential deflection, and the pressure contours are symmetric about the midplane. Across all three cases the lift magnitude does not exceed 10^{-18} N—many orders of magnitude below any physically meaningful aerodynamic force—confirming that the residual asymmetry is of purely numerical origin.

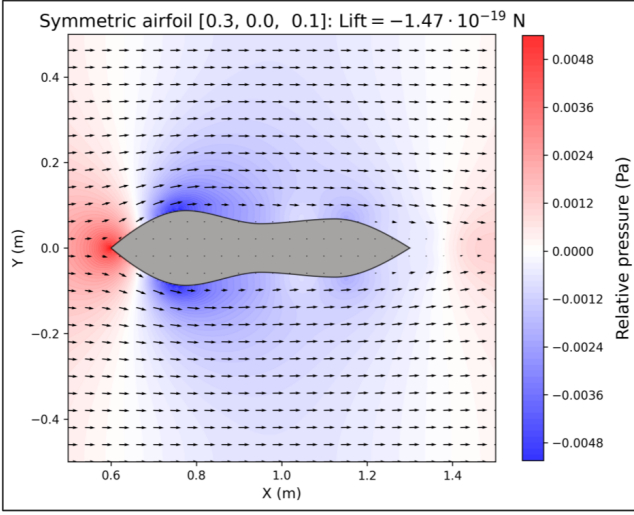


FIG. 6. **Lift validation—primary case.** Symmetric airfoil (parameters [0.3, 0.0, 0.1], thickness 0.175 m, PCHIP). Lift: $L = -1.47 \times 10^{-19}$ N.

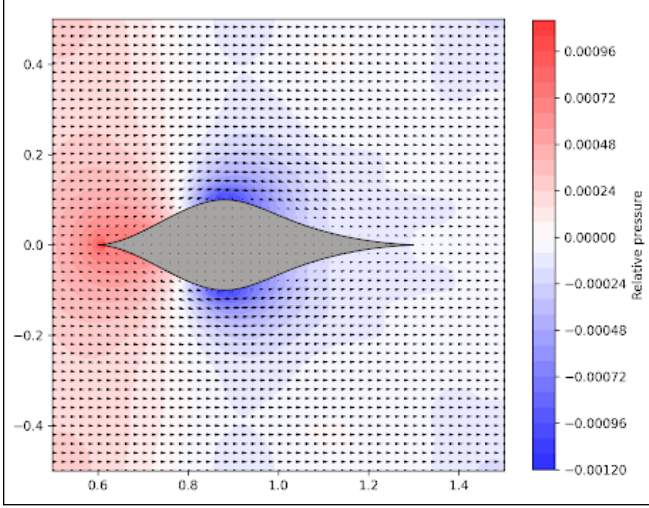


FIG. 7. **Lift validation—streamlined symmetric profile.** Pointed leading and trailing edges with a central bulge. Lift: $L = -4.18 \times 10^{-19}$ N.

C. Lift Generation from Cambered Airfoils

Having established the physical fidelity of both force calculations, we demonstrate that the solver correctly captures lift generation by asymmetric (cambered) airfoils. A cambered profile was first created using the custom parameterization (upper parameters [0.3, 0.5, 0.25, 0.05, -0.2]; lower parameters [-0.2, -0.1, -0.15, -0.2, -0.4]; cubic interpolation; thickness 0.2 m). At 0.1 m/s, this airfoil generates $L = 4.483 \times 10^{-4}$ N and $D = 3.660 \times 10^{-4}$ N, for an L/D ratio of 1.225 (Fig. 9). The pressure field reveals the expected low-pressure region above the upper surface and elevated pressure below, consistent with the asymmet-

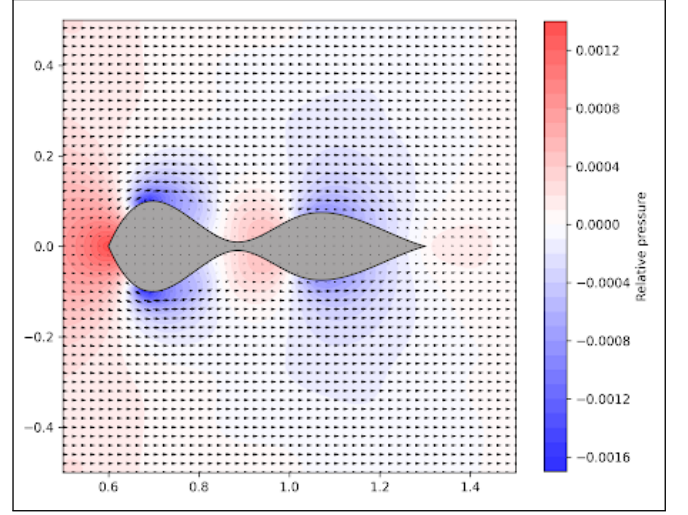


FIG. 8. **Lift validation—bimodal symmetric profile.** Leading-edge bulge, narrow midsection, and secondary trailing-edge bulge. Lift: $L = -1.62 \times 10^{-18}$ N.

ric path lengths imposed by the camber. These results confirm that the interpolation-based definition method is capable of producing designs with substantial aerodynamic performance.

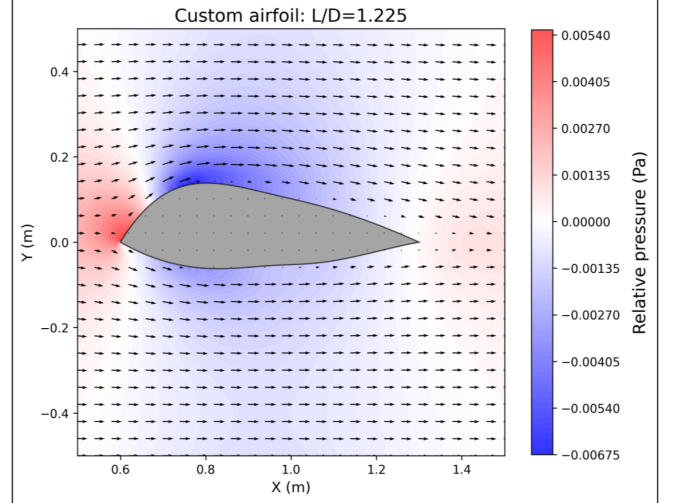


FIG. 9. **Lift from a custom cambered airfoil.** Pressure and velocity fields at 0.1 m/s. Velocity vectors displayed at stride 16. $L/D = 1.225$ ($L = 4.483 \times 10^{-4}$ N, $D = 3.660 \times 10^{-4}$ N).

A systematic search over 98 four-digit NACA airfoils was then conducted. Of these, NACA 5315 achieved the highest L/D of 1.479 (Fig. 10), followed by NACA 4310 ($L/D = 1.464$) and NACA 6415 ($L/D = 1.462$). The mean lift across all tested configurations was 3.200×10^{-4} N and the mean L/D was 0.720. These findings replicate the well-known broad lift-production capabilities of the NACA family [12] and further validate the

solver’s sensitivity to the camber parameters that govern lift generation.

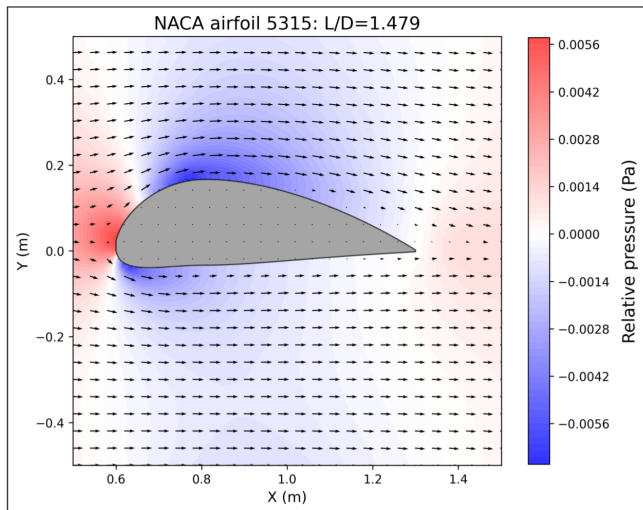


FIG. 10. **Best NACA result: NACA 5315.** Pressure and velocity fields at 0.1 m/s; velocity vectors at stride 16. $L/D = 1.479$ ($L = 7.836 \times 10^{-4}$ N, $D = 5.298 \times 10^{-4}$ N).

VI. DISCUSSION

Physical interpretation. The simulations successfully capture the fundamental physics of airfoil aerodynamics within the modeled regime. The pressure differential between upper and lower surfaces, arising from flow acceleration over the cambered upper surface, generates lift consistent with classical theory. The magnitude of the computed forces, while small due to the low flow velocity, scales appropriately with dynamic pressure $q = \frac{1}{2}\rho u^2$. The corresponding Reynolds numbers ($Re \sim 10^3$ – 10^4) place the flow in the laminar regime, where the solver’s assumptions are valid. Real aircraft operate at $Re > 10^6$, where turbulent boundary layers and flow separation introduce effects not captured by a laminar solver.

Validation metrics. Beyond the canonical tests presented in Sec. V, the mean absolute divergence of the velocity field was monitored as a diagnostic. In stable simulations, $\langle |\nabla \cdot \mathbf{u}| \rangle$ stabilized around 1.5×10^{-3} during the speed-ramping process, which is acceptable given the grid resolution. Lowering the Jacobi convergence tolerances provided only marginal reduction in divergence at substantially greater computational cost, indicating proximity to the accuracy limit achievable with this method and resolution.

Comparison to thin airfoil theory. Thin airfoil theory predicts a lift coefficient $C_L = 2\pi(\alpha + m/c)$, where α is the angle of attack and m/c is the camber-to-chord ratio. At $\alpha = 0$ with typical NACA camber ratios ($m/c \sim 0.04$ – 0.06), the theory predicts $C_L \approx 0.25$ – 0.38 . A quantitative comparison is complicated by the very low Reynolds

number of the present simulations, which lies outside the range where thin airfoil theory (an inviscid, high-Re result) is strictly applicable. Nevertheless, the qualitative trend—greater camber producing greater lift—is reproduced.

Uncertainty and reproducibility. Quantifying the uncertainty of the computed forces is challenging given the absence of ground-truth data for the non-standard, non-physical 2D airfoils at these Reynolds numbers. However, the procedure is internally consistent in the sense that it preserves the expected relative ordering of lift and drag across configurations (e.g., thicker profiles always produce more drag). Substantial differences in L/D between two airfoils are therefore likely to reflect genuine physical trends. Nevertheless, the evidence is insufficient to guarantee that the solver preserves the correct ordering of airfoils with only minor differences in geometry, such as distinguishing the top-ranked NACA profiles.

Repeated runs with identical initial conditions and ramping procedures yield consistent force calculations, velocity fields, and divergence diagnostics, indicating that accumulated rounding errors are not a significant concern. Varying the grid size, however, does alter the force predictions, likely because the effective region over which the no-slip condition is enforced depends on the grid spacing. In the absence of experimental data on the relevant length scales, the grid spacing was held constant across all comparative tests.

Limitations. Several factors limit the accuracy and applicability of the results:

1. **Low velocity constraint.** Numerical instability prevented stable simulations at airspeeds above 0.1 m/s. This is attributed to the spatial linearity assumption inherent to finite differences: as airspeed increases, the length scale over which field variables change decreases, requiring finer grids for stability. Simulation cost scales as roughly $O(n^3)$ (halving the grid spacing quadruples the number of points and halves the CFL-limited time step), making significantly finer grids computationally prohibitive. Implicit time integration or higher-order schemes would alleviate this constraint.
2. **Two-dimensional assumption.** Real wings exhibit three-dimensional effects including tip vortices and spanwise flow variations. The present 2D results represent an infinite-span idealization.
3. **Laminar flow only.** Turbulence is not modeled, limiting applicability to low Reynolds number flows. Incorporating Reynolds-averaged Navier–Stokes (RANS) equations with turbulence closure would extend the solver to realistic aircraft conditions.
4. **Quasi-steady analysis.** Forces are reported after the flow reaches a quasi-steady state; unsteady phenomena such as vortex shedding at higher Reynolds numbers are not analyzed.

5. **Pressure forces only.** Only pressure-induced forces are computed; viscous (skin-friction) drag is neglected. This underestimates total drag, particularly for streamlined bodies where skin friction is the dominant contribution.
6. **Finite difference conservation.** Unlike finite volume methods, the finite difference approach does not enforce strict local conservation of mass, energy, or momentum, making simulations more susceptible to numerical error at coarse resolution or high velocity.

VII. CONCLUSIONS AND FUTURE WORK

A two-dimensional finite difference solver for the incompressible Navier–Stokes equations has been developed and applied to compute lift and drag forces on airfoils. The implementation combines explicit Euler time-stepping, ω -relaxed Jacobi pressure solving with sparse matrix acceleration, and Voronoi-based force integration. A flexible airfoil parameterization using PCHIP interpolation enables systematic shape variation. Although stable simulations were limited to airspeeds up to approximately 0.1 m/s, the solver predicts physically meaningful lift production from both custom-parameterized and four-digit NACA series airfoils.

Key findings include:

- Symmetric airfoils produce negligible lift ($|L| < 10^{-18}$ N) at zero angle of attack, confirming internal consistency.
- Drag increases monotonically with airfoil thickness and airspeed, in agreement with classical theory.
- Cambered airfoils generate positive lift, with NACA 5315 achieving the highest L/D of 1.479 among the 98 NACA profiles tested.
- The parameterized airfoil framework successfully generates lift-producing shapes suitable for optimization studies.

Future work should address the primary limitations:

1. **Improved stability:** Implementing implicit time integration (e.g., Crank–Nicolson) or semi-Lagrangian advection to enable higher flow velocities approaching realistic flight conditions.
2. **Three-dimensional extension:** Capturing tip vortices, induced drag, and finite aspect ratio effects critical for realistic wing analysis.

3. **Turbulence modeling:** Incorporating RANS equations with turbulence closure models to extend applicability to high Reynolds number flows.
4. **Optimization:** Coupling the solver with gradient-free (genetic algorithm) or gradient-based (adjoint method) optimizers to discover high- L/D airfoil shapes automatically.
5. **Experimental validation:** Comparing computed forces against wind tunnel measurements to quantify absolute accuracy and guide model improvements.

This project demonstrates that physically meaningful aerodynamic simulations are achievable with moderate computational resources and provides a foundation for more advanced CFD studies.

VIII. REPRODUCIBILITY AND CODE AVAILABILITY

All code, data, and analysis notebooks are publicly available at https://github.com/ParshvCrafts/CFD_Navier-Stokes_Solver. The code runs under Python 3.13 with recent versions of NumPy, SciPy, Matplotlib, and Pandas. Saved simulation states in `.npz` format are provided for post-processing. The repository README contains detailed instructions for reproducing the results.

Computational cost. A typical simulation to steady state (~ 1000 time steps) at 800×400 resolution requires approximately 6 minutes on a single CPU core. The one-time Jacobi matrix construction takes approximately 30 seconds. Memory usage is dominated by the sparse Jacobi matrix and the velocity/pressure arrays.

Appendix A: Discretized Scalar Momentum Equations

The two scalar momentum conservation equations for 2D incompressible flow are

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + a_x, \quad (\text{A1})$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = -\frac{1}{\rho} \frac{\partial p}{\partial y} + \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) + a_y. \quad (\text{A2})$$

Using a forward difference for $\partial \phi / \partial t$, a backward difference for $u \partial \phi / \partial x$, and central differences for all remaining terms, the fully discretized equations become

$$\frac{u_C^{n+1} - u_C^n}{\Delta t} + u_C^n \frac{u_C^n - u_W^n}{\Delta x} + v_C^n \frac{u_N^n - u_S^n}{2\Delta y} = -\frac{1}{\rho} \frac{p_E - p_W}{2\Delta x} + \nu \left(\frac{u_E^n - 2u_C^n + u_W^n}{\Delta x^2} + \frac{u_N^n - 2u_C^n + u_S^n}{\Delta y^2} \right) + a_x \quad (\text{A3})$$

and

$$\frac{v_C^{n+1} - v_C^n}{\Delta t} + u_C^n \frac{v_C^n - v_W^n}{\Delta x} + v_C^n \frac{v_N^n - v_S^n}{2\Delta y} = -\frac{1}{\rho} \frac{p_N - p_S}{2\Delta y} + \nu \left(\frac{v_E^n - 2v_C^n + v_W^n}{\Delta x^2} + \frac{v_N^n - 2v_C^n + v_S^n}{\Delta y^2} \right) + a_y, \quad (\text{A4})$$

which are trivially rearranged for u_C^{n+1} and v_C^{n+1} . All other continuous derivatives appearing in this work are discretized using the same scheme.

AUTHOR CONTRIBUTIONS

- **Parshv Patel:** Implemented the core Navier–Stokes solver and explicit time-stepping routines; created the airfoil geometry generation framework (PCHIP interpolation, NACA implementation) and visualization.
- **Orhan Hosten-Mittermaier:** Developed the pressure Poisson solver and Jacobi iteration with sparse matrix optimization; conducted validation studies; implemented force calculation using Voronoi influence sections and boundary masking; performed the resolution analysis.
- Both authors contributed to writing and reviewing the manuscript.

USE OF AI TOOLS (DISCLOSURE)

AI tools (Claude, ChatGPT) were used in select instances to gain an overview of computational methods available for certain aspects of the project. The methods ultimately adopted were implemented only after confirming their validity with non-AI sources. Use of generative AI for code was limited to debugging Python code, suggesting improvements to numerical stability, and generating the NACA airfoil thickness function. All code was inspected and validated against independent references. Generative AI was not used for the report-writing process. All AI-generated suggestions were reviewed, tested, and modified by the authors; the core algorithms, physical interpretations, and conclusions are the original work of the authors.

ACKNOWLEDGMENTS

The authors thank the instructor Eric Ma and GSI Harper Sewalls of Physics 77 for guidance on research scope and on the presentation of results.

-
- [1] C. L. Fefferman, “Existence and smoothness of the Navier–Stokes equation,” in *The Millennium Prize Problems* (Clay Mathematics Institute, 2006), pp. 57–67.
 - [2] J. D. Anderson, *Fundamentals of Aerodynamics*, 5th ed. (McGraw-Hill, New York, 2010).
 - [3] A. Jameson, “Aerodynamic design via control theory,” *J. Sci. Comput.* **3**, 233–260 (1988).
 - [4] S. B. Pope, *Turbulent Flows* (Cambridge University Press, Cambridge, 2000).
 - [5] J. Slotnick *et al.*, “CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences,” NASA/CR-2014-218178 (2014).
 - [6] H. K. Versteeg and W. Malalasekera, *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*, 2nd ed. (Pearson, Harlow, 2007).
 - [7] O. C. Zienkiewicz, R. L. Taylor, and P. Nithiarasu, *The Finite Element Method for Fluid Dynamics*, 6th ed. (Elsevier, Oxford, 2005).
 - [8] A. J. Chorin, “Numerical solution of the Navier–Stokes equations,” *Math. Comput.* **22**, 745–762 (1968).
 - [9] J.-L. Guermond, P. Mineev, and J. Shen, “An overview of projection methods for incompressible flows,” *Comput. Methods Appl. Mech. Eng.* **195**, 6011–6045 (2006).
 - [10] F. H. Harlow and J. E. Welch, “Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface,” *Phys. Fluids* **8**, 2182–2189 (1965).
 - [11] C. A. J. Fletcher, *Computational Techniques for Fluid Dynamics*, 2nd ed. (Springer, Berlin, 1991).
 - [12] I. H. Abbott and A. E. Von Doenhoff, *Theory of Wing Sections* (Dover, New York, 1959).
 - [13] C. de Boor, *A Practical Guide to Splines* (Springer, New York, 1978).
 - [14] F. N. Fritsch and R. E. Carlson, “Monotone piecewise cubic interpolation,” *SIAM J. Numer. Anal.* **17**, 238–246 (1980).
 - [15] NOAA, NASA, and USAF, “U.S. Standard Atmosphere, 1976,” NOAA-S/T 76-1562 (1976).
 - [16] L. A. Barba and G. F. Forsyth, “CFD Python: the 12 steps to Navier–Stokes equations,” *J. Open Source Educ.* **2**(16), 21 (2019).
 - [17] S. E. Rogers and D. Kwak, “An upwind differencing scheme for the incompressible Navier–Stokes equations,” *Appl. Numer. Math.* **8**, 43–64 (1991).
 - [18] J. H. Ferziger, M. Perić, and R. L. Street, *Computational Methods for Fluid Dynamics*, 4th ed. (Springer, Cham, 2020).
 - [19] T. J. Chung, *Computational Fluid Dynamics* (Cambridge University Press, Cambridge, 2002).
 - [20] Q. Kong, T. Siau, and A. M. Bayen, “Numerical Differentiation,” in *Python Programming and Numerical Methods* (Academic Press, 2021), pp. 337–351.