

PANKH: An unsteady potential flow solver for hovering airfoils


Rohit Chowdhury¹, Nipun Arora¹, and Ashish Pathak¹

¹ Department of Mechanical Engineering, Indian Institute of Technology, Jodhpur, Rajasthan, India

DOI: [10.21105/joss.08865](https://doi.org/10.21105/joss.08865)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Pi-Yueh Chuang](#)  

Reviewers:

- [@andrewgiuliani](#)
- [@stebiondi](#)
- [@daavid00](#)

Submitted: 28 March 2025

Published: 28 October 2025

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

Analyzing the aerodynamics of unsteady airfoils is essential for understanding the performance of wind turbine blades, helicopter rotors, and the flight dynamics of birds, insects, fixed-wing aircraft, and micro air vehicles (MAVs). While computational fluid dynamics (CFD) offers detailed and high-fidelity results, potential flow solvers provide a faster and more accessible alternative that still captures key aerodynamic phenomena. These include time-resolved pressure distribution, unsteady lift generation, thrust prediction, power consumption analysis, and estimation of propulsive efficiency. The use of free-wake modeling further enables these solvers to capture critical unsteady effects such as vortex shedding and wake-induced flow interactions.

We introduce PANKH (**P**anel **A**nalysis of **u**nsteady **K**inematics of **H**overing **a**irfoils), an open-source C++ tool that employs the unsteady vortex panel method to evaluate aerodynamic forces on airfoils in arbitrary motion. Its flexible, modular design enables users to specify custom kinematic patterns (impulsive motion, pitching, plunging), ideal for exploring bio-inspired flapping flight and gust responses.

The solver's source code, validation examples, and comprehensive Doxygen-generated API documentation are hosted on GitHub, ensuring accessibility and reproducibility. Future enhancements may include expanded input capabilities, improved wake modeling techniques for more accurate unsteady flow prediction, and the integration of viscous effects along with two-way fluid–structure interaction (FSI) to simulate flexible airfoils and their coupling with the flow in a strongly coupled manner.

Statement of Need

For low-Mach, incompressible applications—where viscous phenomena are largely confined to thin boundary layers and wakes—potential-flow solvers provide a computationally efficient alternative to high-fidelity CFD. Representative studies and benchmarks report computational cost reductions varying from tens to several orders of magnitude depending on solver formulation and problem size ([Dimitriadis, 2023](#); [Katz & Plotkin, 2001](#); [Persson et al., 2012](#)). Narrowing down to the case of a pitching and plunging airfoil, PANKH demonstrated a runtime reduction on the order 10^3 compared to the commercial package ANSYS, when executed on identical single-core hardware. High-fidelity CFD simulations, executed on high-performance computing (HPC) clusters comprising multiple nodes and cores, often demand days of runtime due to their intensive computational requirements and parallel processing across distributed architectures. Experimental aerodynamic investigations, on the other hand, involve considerable capital investment and extended timelines. Designing wind/water tunnel setups, procuring flow visualization tools, and conducting controlled tests may take several months to years. Moreover, accommodating different flow conditions often requires substantial modifications to the experimental apparatus. To mitigate these limitations, scaling laws and dimensional

analysis are commonly used but come with their own assumptions and constraints. In contrast, PANKH is a modular and open-source aerodynamic solver written in C++. It is designed to offer a rapid and accessible alternative for potential flow analysis in low-speed aerodynamic applications. Although still in its early development stage, PANKH is capable of delivering accurate estimates of aerodynamic loads within minutes. It runs efficiently even on standard single-core desktop systems. Future enhancements of the software will include parallelization via OpenMP or MPI, significantly improving performance for large-scale unsteady simulations. Additionally, the planned development of a graphical user interface (GUI) will streamline usability across platforms, including Android-based systems, transforming PANKH into a portable and interactive educational tool for both undergraduate and graduate-level aerodynamics courses.

Low-to-medium fidelity solvers like PANKH solve Laplace's equation to model inviscid, incompressible, and irrotational flows, minimizing computational complexity while delivering precise lift estimations for aerodynamic analysis. By solving Laplace's equation with appropriate boundary conditions, PANKH enables rapid aerodynamic analysis, making it particularly useful for preliminary design, scaling studies, and parametric investigations. Despite their advantages, open-source tools for unsteady potential flow analysis remain scarce. Established tools such as XFOIL (Drela, 1989), NASA's FoilSIM III (Center, 2022), JavaFoil (Hepperle, 2020) are tailored for steady-state aerodynamics and lack the capability to address unsteady flow phenomena. Existing research codes for unsteady vortex panel methods are often proprietary, poorly documented, or no longer maintained. In contrast, PANKH is purpose-built for the unsteady aerodynamics of hovering airfoils, offering advanced features tailored for applications such as fixed-wing aircraft, flapping-wing micro-air vehicles (MAVs), ornithopters, and hovering rotorcraft. The software repository includes validation cases comparing PANKH's results with experimental studies by (Anderson et al., 1998) and (Floryan et al., 2017), as well as with numerical simulations reported by (Dimitriadis, 2023). For three-dimensional wing design, understanding two-dimensional cross-sectional properties through airfoil analysis is critical for selecting optimal shapes.

PANKH shows strong potential for real-world applications across research, education, and hobbyist domains. In the aviation industry, it can be integrated with structural solvers to perform coupled fluid–structure interaction analyses, aiding the design of flexible lifting surfaces. Its modular and procedural backend makes it suitable for classroom use, allowing students to simulate and visualize various unsteady flow scenarios with ease. PANKH can also support hobbyists and independent experimentalists working on flapping-wing vehicles, to estimate aerodynamic forces on key sections—supporting early design decisions and material selection.

Methodology

Governing Equation and Boundary Conditions

The numerical framework solves Laplace's equation, $\nabla^2 \Phi = 0$, where Φ denotes the velocity potential function. The solution is obtained by applying appropriate boundary conditions. These include the no-penetration condition on the surface of the body and the far-field condition, the latter being inherently satisfied by the elementary solutions of Laplace's equation. In two-dimensional potential flow, Laplace's equation admits a set of fundamental/elementary solutions, such as point singularities (sources, doublets, and vortices) and singularity distributions of constant or linearly varying strength. The linear nature of the governing PDE allows these solutions to be combined using the superposition principle to model complex flow fields. Thus, the problem reduces to determining an appropriate combination of singularity distributions such that, the geometry or the solid boundary itself becomes a streamline of the flow.

This solver adopts Neumann boundary conditions, formulating the problem in terms of velocity rather than directly solving for Φ . The velocity field is expressed through singularity distributions placed along the boundary of the immersed body. The computational efficiency of the panel

method stems from its surface-only discretization—the governing equations are enforced solely on the body boundary, eliminating the need to discretize and solve for the unknowns throughout the entire flow field, as is required in Navier–Stokes-based solvers. Once the surface velocity distribution is obtained, the unsteady Bernoulli equation is applied to evaluate the pressure along the airfoil contour. Integration of this pressure field over the surface then yields the aerodynamic coefficients, including lift and the pressure-induced (inviscid) drag.

Discretization of Geometry and Singularity Element Distribution

In unsteady potential flow modeling, accurate representation of the wake is as important as modeling the solid body itself. The overall flow field consists of a distribution of singularity elements along the solid boundary, referred to as the bound vorticity, and a wake region downstream of the body containing wake vorticity that is free to convect with the flow. In unsteady motion, the bound vorticity varies with time, and at each time step, a corresponding amount of vorticity equal to the change in bound circulation must be shed into the wake. This ensures compliance with Kelvin's circulation theorem (Eldredge, 2019; Katz & Plotkin, 2001). The following paragraphs describe how these principles are implemented within the present numerical framework.

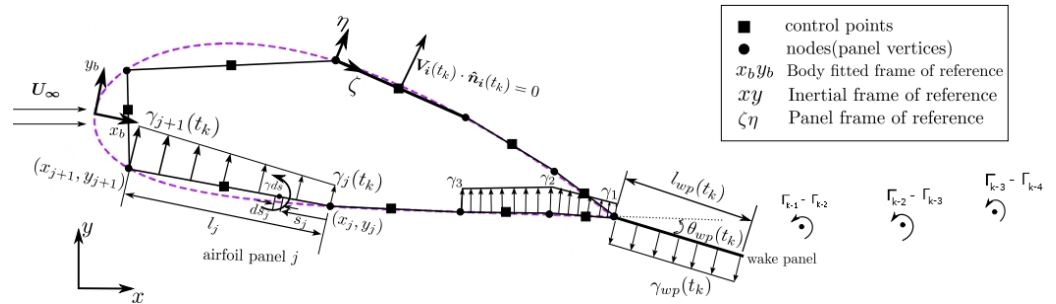


Figure 1: A schematic diagram of the panel discretization on an airfoil and wake modeling at time instant t_k in the inertial frame of reference xy .

The airfoil is discretized into n nodes (panel endpoints), forming $n - 1$ flat geometric panels. A *piecewise linearly varying vortex distribution* is placed along the airfoil surface, as illustrated in **Figure 1**. On each panel, the vortex strength varies linearly from γ_j to γ_{j+1} , and these strengths are treated as unknowns to be solved.

The wake vorticity sheet is discretized as follows:

- A *constant-strength vortex panel*, denoted γ_{wp} , is shed from the trailing edge at the current time step. This value is also treated as an unknown.
- A *series of point vortices* represents the wake vorticity shed during all previous time steps. Their strengths are known from past computations.

In the next time step, the current wake panel is convected downstream with the flow and becomes part of the known wake vorticity field, modeled as a point vortex of equivalent strength. Consequently, the system has a total of $n + 1$ unknowns, which are as follows: - **(a)** n bound vortex strengths γ_j ($1 \leq j \leq n$), and - **(b)** $(n + 1)^{\text{th}}$ unknown, γ_{wp} , representing the strength of the latest shed wake panel.

Accordingly, a system of $n + 1$ equations is required to determine the $n + 1$ unknowns. The following section describes the formulation of these $n + 1$ equations in detail. For additional methodological details, refer to (Veza & Galbraith, 1985) and (Basu & Hancock, 1978).

Each airfoil panel has a control point (or collocation point) at its midpoint, where the no-penetration boundary condition is enforced, yielding $n - 1$ equations:

$$[(\nabla\Phi - \mathbf{V}_{\text{body}})_i \cdot \hat{\mathbf{n}}_i]_{t_k} = 0, \quad (1 \leq i \leq n - 1)$$

Here, Φ is the total velocity potential, composed of the perturbation and freestream components, i.e., $\Phi = \Phi_{\text{perturbation}} + \Phi_{\infty}$. The perturbation potential $\Phi_{\text{perturbation}}$ accounts for the influence of both bound and wake vortices, while Φ_{∞} represents the freestream velocity potential. The same equation can be equivalently expressed explicitly in terms of velocity as follows:

$$\left[\left(\mathbf{V}_{\text{bound}} + \mathbf{V}_{\text{wake}} + \underbrace{\left(\mathbf{V}_{\infty} - \frac{\mathbf{V}_{\text{body}}}{\mathbf{V}_0 + \boldsymbol{\Omega} \times \mathbf{r}_i} \right)}_{\mathbf{V}_{\text{kinematics}}} \right) \cdot \hat{\mathbf{n}}_i \right]_{t_k} = 0, \quad (1 \leq i \leq n-1)$$

Here, $\mathbf{V}_{\text{kinematics}}$ is the relative velocity between the surface of the body and the surrounding flow. The contribution of bound vortices to the total velocity field is represented by $\mathbf{V}_{\text{bound}}$. $\mathbf{V}_{\text{bound}}$ at i^{th} control point is calculated by summing the contribution of only the vortex panels on the airfoil surface.

$$\begin{bmatrix} V_{x_i} \\ V_{y_i} \end{bmatrix}_{\text{bound}} = \sum_{j=1}^{n-1} [\mathbf{P}]_{ji} \begin{bmatrix} \gamma_j \\ \gamma_{j+1} \end{bmatrix}$$

where $[\mathbf{P}]_{ji}$ is the 2×2 *panel coefficient matrix*, describing the coefficients of the velocity induced by j^{th} panel on i^{th} control point. The expression for the $[\mathbf{P}]_{ji}$ is derived from (Phillips, 2004).

The velocity contribution due to the wake, \mathbf{V}_{wake} , comes from the most recently shed wake panel and all previously shed vortices. This is more complex for the current wake panel, since its strength γ_{wp} , length l_{wp} , and orientation θ_{wp} are initially unknown. To compute its velocity contribution, a guess for length and orientation is made. A 2×2 Jacobian matrix is formed along with a residual from two equations (one for length, one for orientation), which is solved using Newton's method. The corrected length and orientation are then used to compute \mathbf{V}_{wake} for the current time step.

Other Physical Considerations

Trailing Edge Condition (Kutta Condition)

Satisfying the boundary conditions alone does not yield a unique solution for the bound vortex strengths, γ_j ($1 \leq j \leq n$). To obtain a unique solution, the flow must leave the airfoil's sharp trailing edge smoothly along the bisector line. This requirement is known as the Kutta Condition (Eldredge, 2019). Mathematically, this condition is expressed as:

$$\gamma_{\text{TE}}(t_k) = 0, \quad \text{where} \quad \gamma_{\text{TE}}(t_k) = \gamma_1(t_k) + \gamma_n(t_k) + \gamma_{wp}(t_k)$$

Here, γ_{TE} represents the vorticity strength at the trailing edge (TE). In other words, Kutta Condition in this case implies that the strength of vorticity at the trailing edge must vanish.

This serves as the n^{th} equation of the system.

Kelvin's Circulation Theorem

For unsteady flows, the motion of the airfoil causes wake formation. The reason for wake formation can be explained by surrounding the airfoil by a sufficiently large contour, and noting that only conservative forces, such as pressure, act on the contour (Katz & Plotkin, 2001). Then Kelvin's theorem states that the total circulation in the contour remains constant, i.e.,

$$\frac{D\Gamma}{Dt} = \frac{D}{Dt} (\Gamma_{\text{airfoil}} + \Gamma_{\text{wake}}) = 0$$

Here, Γ represents the total circulation in the flow domain enclosed by the contour. It is the sum of: - **(a)** Γ_{airfoil} , the bound circulation associated with the airfoil, and - **(b)** Γ_{wake} , the circulation in the wake.

This equation provides the $(n + 1)^{\text{th}}$ constraint, required to solve for the $n+1$ unknowns in the system.

The resulting system, formulated as shown in **Figure 2**, is expressed as a single matrix equation and solved using the PartialPivLU decomposition solver available in the *Dense Linear Problems and Decompositions* module of the Eigen Library (Guennebaud et al., 2010). In-depth insights into post-processing steps, encompassing the solution of the unsteady Bernoulli equation and computations of aerodynamic loads, are detailed in (Chowdhury et al., 2024).

$$\begin{array}{c}
 \vec{V}_{cp_i} \cdot \hat{n}_i = 0 \\
 \Gamma_{TE} = 0 \\
 \frac{D\Gamma}{Dt} = 0
 \end{array}
 \begin{array}{c}
 \text{Influence of airfoil panels} \\
 \begin{array}{ccccc}
 a_{11} & a_{12} & \dots & a_{1n} & a_{1wp} \\
 a_{21} & a_{22} & \dots & a_{2n} & a_{2wp} \\
 \vdots & \vdots & \ddots & \vdots & \vdots \\
 a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n} & a_{n-1,wp} \\
 1 & 0 & \dots & 1 & a_{nwp} \\
 0.5l_1 & 0.5(l_1 + l_2) & \dots & 0.5(l_{n-1}) & l_{wp}
 \end{array} \\
 \text{Influence of latest wake panel}
 \end{array}
 \begin{bmatrix}
 \gamma_1(t_k) \\
 \gamma_2(t_k) \\
 \vdots \\
 \gamma_{n-1}(t_k) \\
 \gamma_n(t_k) \\
 \gamma_{wp}(t_k)
 \end{bmatrix}
 =
 \begin{bmatrix}
 RHS_1 \\
 RHS_2 \\
 \vdots \\
 RHS_{n-1} \\
 0 \\
 \Gamma_{bound}(t_k - 1)
 \end{bmatrix}$$

$$RHS_i = - \left[\underbrace{(\nabla \Phi_i)_{\Gamma_{wake}^{t-\Delta t}}(\mathbf{x}, t)}_{\text{Velocity induced due to previously shed vortices}} + \underbrace{\mathbf{V}_{\infty} - (\mathbf{V}_0 + \boldsymbol{\Omega} \times (\mathbf{x}_i(t) - \mathbf{x}_{cg}(t)))}_{\mathbf{V}_{kin} \text{ Relative velocity between fluid and solid surface}} \cdot \hat{n}_i(\mathbf{x}, t)$$

Figure 2: Equations formulated to determine the set of unknowns at a time instant t_k .

Wake Modeling

PANKH utilizes a robust time-stepping wake model to capture the intricate dynamics of vortex shedding, which is a critical aspect of unsteady aerodynamics. Shed vortices convect, interact, and induce velocities on both the airfoil and neighboring wake elements, significantly influencing the pressure distribution and resulting aerodynamic loads. While some existing methods, such as the one used in ULVPC (Prosser et al., 2011), model the most recently shed vortex as a point vortex for simplicity, this approximation may limit the physical fidelity of wake dynamics. In contrast, PANKH adopts the vortex panel-based approach of (Basu & Hancock, 1978). In this approach, the newly shed wake vortex is modeled as a constant-strength vortex panel whose strength, orientation, and length are not known a priori. This leads to a nonlinear system of equations, which is solved iteratively using Newton's method.

The entire numerical framework discussed above is implemented in the code. The implementation is structured into separate .cpp files, each containing functions dedicated to specific numerical tasks. The function and variable names are chosen to align with their respective roles in the numerical framework, ensuring clarity and ease of use for the user. A detailed guide on input parameters, motion specification, and output interpretation is provided in the project's [README](#). An extensive testing and validation suite has been used to ensure the accuracy of the results obtained from the source code.

Acknowledgements

This work was supported by the Defence Research and Development Organization (DRDO) under Grant No. DFTM/14/3958/IITJ/P/04/FOMS-02/1034/D(R&D)/2024.

References

- Anderson, J. M., Streitlien, K., Barrett, D., & Triantafyllou, M. S. (1998). Oscillating foils of high propulsive efficiency. *Journal of Fluid Mechanics*, 360, 41–72. <https://doi.org/10.1017/S0022112097008392>
- Basu, B. C., & Hancock, G. J. (1978). The unsteady motion of a two-dimensional aerofoil in incompressible inviscid flow. *Journal of Fluid Mechanics*, 87, 159–178. <https://doi.org/10.1017/S0022112078002980>
- Center, N. G. R. (2022). *FoilSim III – interactive airfoil simulation*.
- Chowdhury, R., Pathak, A., & Arora, N. (2024, June). *Development of an unsteady vortex panel method for a flapping airfoil*. https://www.researchgate.net/publication/393158131_Development_of_an_unsteady_vortex_panel_method_for_a_flapping_airfoil
- Dimitriadis, G. (2023). *Unsteady aerodynamics: Potential and vortex methods*. John Wiley & Sons, 2023. ISBN: 978-1-119-76247-8
- Drela, M. (1989). XFOIL: An analysis and design system for low Reynolds number airfoils. *Low Reynolds Number Aerodynamics: Proceedings of the Conference Notre Dame, Indiana, USA, 5–7 June 1989*, 1–12. https://doi.org/10.1007/978-3-642-84010-4_1
- Eldredge, J. D. (2019). *Mathematical modeling of unsteady inviscid flows* (Vol. 50). Springer. <https://doi.org/10.1007/978-3-030-18319-6>
- Floryan, D., Van Buren, T., Rowley, C. W., & Smits, A. J. (2017). Scaling the propulsive performance of heaving and pitching foils. *Journal of Fluid Mechanics*, 822, 386–397. <https://doi.org/10.1017/jfm.2017.302>
- Guennebaud, G., Jacob, B., & others. (2010). *Eigen v3: C++ template library for linear algebra*. <https://eigen.tuxfamily.org>.
- Hepperle, M. (2020). *JavaFoil - airfoil analysis program*.
- Katz, J., & Plotkin, A. (2001). *Low-speed aerodynamics* (Vol. 13). Cambridge University Press. <https://doi.org/10.1115/1.1669432>
- Persson, P.-O., Willis, D. J., & Peraire, J. (2012). Numerical simulation of flapping wings using a panel method and a high-order Navier–Stokes solver. *International Journal for Numerical Methods in Engineering*, 89(10), 1296–1316. <https://doi.org/10.1002/nme.3288>
- Phillips, W. F. (2004). *Mechanics of flight*. John Wiley & Sons. ISBN: 978-0-471-33458-3
- Prosser, D., Crassidis, A., Ghosh, A., & Roemer, M. (2011). The applicability of unsteady vortex panel methods to the design of hovering flapping-wing micro air vehicles. *AIAA Atmospheric Flight Mechanics Conference*, 6392. <https://doi.org/10.2514/6.2011-6392>
- Veza, M., & Galbraith, R. M. (1985). A method for predicting unsteady potential flow about an aerofoil. *International Journal for Numerical Methods in Fluids*, 5(4), 347–356. <https://doi.org/10.1002/flid.1650050405>