Titre: Title:	Development of a Full Potential Solver Using an Immersed Boundary Method for Icing Simulation
Auteur: Author:	Guillaume Auger
Date:	2026
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Auger, G. (2026). Development of a Full Potential Solver Using an Immersed Boundary Method for Icing Simulation [Master's thesis, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/76259/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Development of a Full Potential Solver Using an Immersed Boundary Method for Icing Simulation

GUILLAUME AUGER
Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie aérospatial

Avril 2026

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Development of a Full Potential Solver Using an Immersed Boundary Method for Icing Simulation

présenté par **Guillaume AUGER**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Roberto PAOLI, président
Éric LAURENDEAU, membre et directeur de recherche
Yannick HOARAU, membre

DEDICATION

À mes proches...

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all those who contributed, directly or indirectly, to the completion of this research project.

First and foremost, I wish to thank my research supervisor, Éric Laurendeau, who introduced me to the field of computational fluid dynamics and from whom I have learned immensely. His expertise, guidance, and enthusiasm were a constant source of inspiration.

I would also like to express my gratitude to Professor Yannick Hoarau for guiding and welcoming me during my research stay, in the ICube research laboratory at the University of Strasbourg. This experience provided the opportunity to implement the immersed boundary method, which constitutes a key component of this work.

I am also grateful to my colleagues for their support and insightful discussions throughout this project. I particularly thank Maxime Blanchet and Karim Zayni for pioneering the immersed boundary method within CHAMPS and developing key preprocessing tools, as well as Clément Gorand for the boundary layer model, essential to icing simulations. I would also like to thank Hieu Nhan Tran for the valuable discussions we shared and for his contributions to this research project. As he takes over this work for his own master's studies, I wish him the very best for the remainder of his research. Finally, I acknowledge all the colleagues I had the pleasure of working with: Anthony, Baptiste, Charles L.P., Charles R., Justin, Lauren, Pablo, Samuel A., Samuel F., Simon, Titouan, and Vincent.

I gratefully acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada and the *Fonds de recherche du Québec – Nature et technologies*. I also thank the Digital Research Alliance of Canada and Calculs Québec for providing computational resources on the Trillium cluster used for most simulations in this thesis.

I would also like to acknowledge the use of artificial intelligence tools (ChatGPT) for editing and improving the grammar of the text in this thesis.

On a more personal note, I would like to thank my parents, who have always believed in me and have invested a lot of time and effort in my education from an early age. I am also grateful to my brother, my friends, and my family for providing balance and support throughout this journey.

Finally, I would like to express my deepest gratitude to my partner for her unconditional support and understanding throughout this project. This work required a significant investment of time and energy, and her patience and encouragement meant everything to me.

RÉSUMÉ

Le processus de conception préliminaire d'un aéronef nécessite le développement d'outils aérodynamiques à la fois rapides et suffisamment précis afin de permettre une conception optimale. L'approche standard actuelle repose principalement sur des méthodes de haute fidélité basées sur les équations de Navier–Stokes moyennées de Reynolds (RANS), reconnues pour leur robustesse et leur précision, mais dont le coût de calcul demeure trop élevé pour les phases initiales de conception. À l'inverse, les méthodes de plus faible fidélité, telles que la méthode de la ligne portante non linéaire ou la méthode des vortex non linéaires (NLVLM), bien que peu coûteuses, ne permettent pas de capturer certaines caractéristiques physiques essentielles de l'écoulement.

Par ailleurs, il est souhaitable que les outils aérodynamiques soient couplés dès les premières étapes du processus de conception afin de permettre des simulations et des optimisations multidisciplinaires, notamment pour mieux prédire des phénomènes tels que l'aérogivrage. Ces simulations impliquent la prise en compte de géométries complexes et évolutives, dont le traitement reste difficile avec les approches classiques reposant sur des maillages conformes au corps.

L'objectif de ce mémoire est de répondre à ces deux enjeux en développant un solveur aérodynamique de fidélité intermédiaire, basé sur l'équation du potentiel compressible discrétisée en volumes finis sur des maillages non structurés. Cette approche vise à offrir un meilleur compromis entre coût de calcul et précision. Une méthode de frontières immergées à cellules fantômes est également intégrée afin de permettre la simulation d'écoulements autour de maillages non conformes au corps, simplifiant ainsi considérablement le processus de génération de maillage.

Le solveur non visqueux développé est ensuite couplé à un modèle de couche limite permettant d'approximer les effets visqueux ainsi que les coefficients thermiques nécessaires à la simulation des phénomènes de givrage.

Finalement, l'approche proposée est intégrée au cadre de simulation de givrage du logiciel CHApel Multi-Physics Simulation (CHAMPS), afin d'évaluer les performances de la méthode aérodynamique dans ce contexte.

En comparaison avec un solveur Euler, les résultats montrent que le solveur est capable de prédire avec précision l'écoulement autour de plusieurs configurations, notamment les profils NACA0012 à bord de fuite mince et épais, le profil supercritique RAE2822, ainsi que l'aile

ONERA M6, sur des maillages structurés et non structurés. Une légère perte de précision est toutefois observée en régime transsonique, où les ondes de choc sont prédites plus fortes et légèrement décalées vers l'aval, en raison de l'hypothèse isentropique du modèle.

Du point de vue du coût de calcul, le solveur proposé se montre plus efficace que le solveur Euler en régime subsonique. En revanche, en régime transsonique, ses performances sont actuellement limitées par la formulation de la Jacobienne analytique.

Concernant la méthode à frontières immergées, les résultats obtenus concordent avec ceux issus des simulations à maillage conforme. Toutefois, par rapport à la méthode conforme au corps, une perte d'ordre de précision est observée, la méthode étant maintenant limitée à une convergence du premier ordre.

Enfin, en 2D, les simulations de givrage concordent avec les observations expérimentales pour les cas de givre blanc (*rime*), notamment lorsque l'accrétion est modélisée par une approche multi-couches. L'utilisation de la méthode à frontières immergées facilite grandement ces simulations en éliminant le besoin de remaillage. En revanche, pour les cas de givre transparent (*glaze*), les résultats nécessitent encore des améliorations. Les coefficients de transfert thermique prédits sont sous-estimés, ce qui affecte la précision des simulations et nécessite des ajustements supplémentaires.

ABSTRACT

The preliminary design process of an aircraft requires the development of aerodynamic tools that are both computationally efficient and sufficiently accurate to enable effective optimization. The current standard approach primarily relies on high-fidelity methods based on the Reynolds-Averaged Navier–Stokes (RANS) equations, which are known for their robustness and accuracy but remain too computationally expensive for early design stages. Conversely, lower-fidelity methods, such as nonlinear lifting-line theory and the Non-Linear Vortex-Lattice Method (NLVLM), although inexpensive, are unable to capture several key physical features of the flow.

In addition, it is desirable to couple aerodynamic tools as early as possible in the design process to enable multidisciplinary simulations and optimizations, particularly for the prediction of phenomena such as aero-icing. These simulations involve complex and evolving geometries, which remain difficult to handle using conventional approaches based on body-fitted meshes.

The objective of this thesis is to address these challenges by developing a medium-fidelity aerodynamic solver based on the compressible potential equation, discretized using a finite-volume formulation on unstructured meshes. This approach aims to provide a suitable compromise between computational cost and accuracy. A ghost-cell immersed boundary method is also incorporated to enable simulations on non-body-fitted meshes, thereby significantly simplifying the mesh generation process.

The resulting inviscid solver is coupled with a boundary layer model in order to approximate viscous effects, as well as the thermal quantities required for aero-icing simulations.

The proposed methodology is then integrated into the CHAMPS icing simulation framework to assess the performance of the aerodynamic approach in this context.

In comparison with an Euler solver, the results demonstrate that the solver is capable of accurately predicting the flow around several configurations, including the NACA0012 airfoil with both sharp and blunt trailing edges, the RAE2822 supercritical airfoil and the ONERA M6 wing, using both structured and unstructured meshes. A slight loss of accuracy is observed in transonic conditions, where shock waves are predicted to be stronger and slightly shifted downstream due to the isentropic assumption of the model.

In terms of computational cost, the proposed solver is more efficient than the Euler solver for subsonic flows. However, in transonic regimes, its performance is currently limited by the formulation of the analytical Jacobian.

Regarding the immersed boundary method, the results show good agreement with those obtained using body-fitted meshes. However, compared to the body-fitted approach, a reduction in the order of accuracy is observed, with the method being limited to first-order convergence.

Finally, two dimensional icing simulations show good agreement with experimental observations for rime ice cases, particularly when a multi-layer accretion approach is used. The immersed boundary method greatly facilitates these simulations by eliminating the need for remeshing. In contrast, for glaze ice cases, the results remain unsatisfactory. The predicted heat transfer coefficients are underestimated, which affects the accuracy of the simulations and requires further investigation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

AOA Angle Of Attack

BF Body Fitted

BFB Boundary Flux Balancing

CFL Courant–Friedrichs–Lewy

CHAMPS CHApel Multi-Physics Simulation

DNS Direct Numerical Simulation

DPW Drag Prediction Workshop

FDM Finite Difference Method

FEM Finite Element Method

FP Full Potential

FVM Finite Volume Method

GCIBM Ghost-Cell Immersed Boundary Method

GMRES Generalized Minimal RESidual

HLPW High Lift Prediction Workshop

HTC Heat Transfer Coefficient

IB Immersed Boundary

IBM Immersed Boundary Method

IC Implicit Circulation

IP Image Point

IPW Ice Prediction Workshop

LES Large Eddy Simulation

MS Mesh Sequencing

NLVLM Non-Linear Vortex-Lattice Method

PDE Partial Differential Equation

RANS Reynolds-Averaged Navier–Stokes

RBF Radial Basis Function

RL Refinement Level

RMS Root Mean Square of Residuals

SGS Symmetric Gauss-Seidel

SLOR Successive Line Over-Relaxation

SWIM Shallow Water Icing Model

TE Trailing Edge

TSD Transonic Small Disturbance

VLM Vortex-Lattice Method

CHAPTER 1 INTRODUCTION

1.1 Context

The aircraft design process must integrate a wide range of requirements across multiple disciplines during the many workflows throughout the development cycle, up to certification. This process is commonly divided into three main phases: conceptual, preliminary and detailed design. During the conceptual and preliminary phases, a large number of design evaluations are required in order to explore the design space and identify optimal configurations. These early design stages are particularly critical, as it is estimated that up to 80% of an aircraft's life-cycle cost is determined during this period [2]. Therefore, it is beneficial to incorporate multidisciplinarity analyses and higher-fidelity models as early as possible in the design process (see Fig. 1.1), while maintaining a computational cost compatible with large-scale design space exploration.

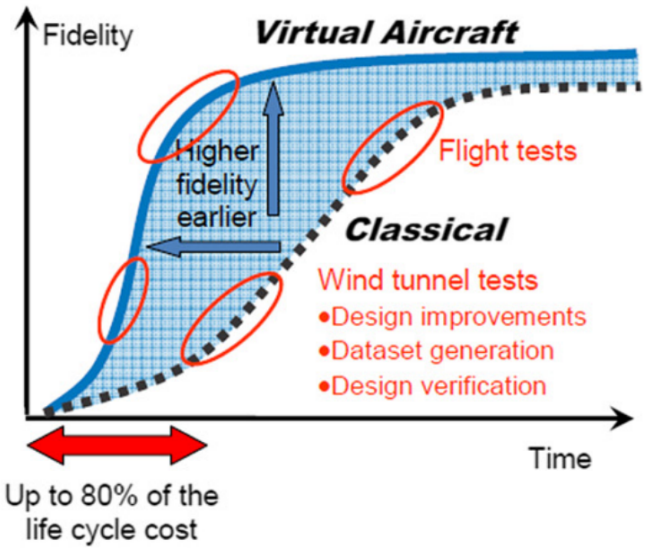


Figure 1.1 Classical product development compared against Virtual Aircraft approach - [2]

1.2 Definitions and Basic Concepts

1.2.1 Aerodynamic Numerical Modeling

There are different fidelity levels of aerodynamic solvers as shown in Fig. 1.2. These solvers uses spatial discretizations and assumptions to approximate the solution of the Navier-Stokes

Equations that are unsolvable analytically for real world applications. The Navier-Stokes Equations arises from the conservation of mass, the conservation of momentum (in all three directions) and the conservation of energy:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\ \frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla p - \nabla \cdot \boldsymbol{\tau} &= 0, \\ \frac{\partial (\rho E)}{\partial t} + \nabla \cdot (\rho \mathbf{u} H) - \nabla \cdot (\boldsymbol{\tau} \cdot \mathbf{u}) &= 0. \end{aligned} \tag{1.1}$$

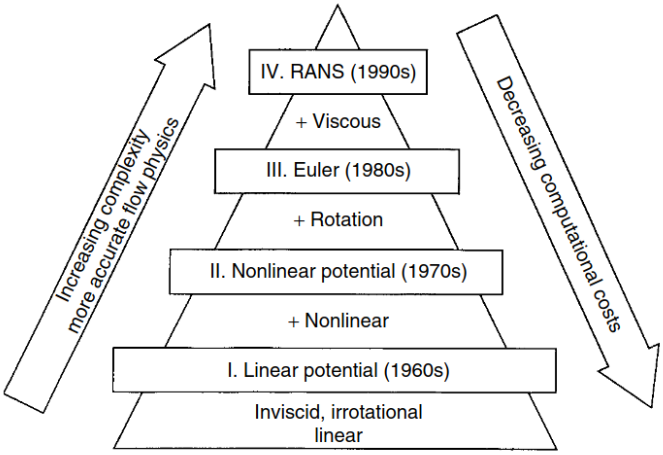


Figure 1.2 Hierarchy of aerodynamic flow solvers - [3]

Direct Numerical Simulation (DNS)

Direct Numerical Simulation (DNS) is the highest fidelity achievable in numerical modeling. In this model, the Navier-Stokes equation are solved numerically without any assumption and turbulence model. This requires the whole range of spatial and temporal scales of turbulence to be solved, which is particularly expensive. Indeed, the number of operations needed to undertake such a simulation is estimated to be a factor of Re^3 [8]. The Reynolds number of a large commercial aircraft lies generally between 10^7 and 10^8, making it far too expensive in such applications.

Large Eddy Simulation (LES)

The first layer of assumption is done by the Large Eddy Simulation (LES) model where the turbulent scales are filtered to directly solve the larger ones while the smaller scales are approximately solved using a turbulence model. This gives the advantage of not having to mesh up to the Kolmogorov scale making it less computationally expensive than DNS but still very demanding.

Reynolds-Averaged Navier–Stokes (RANS)

Reynolds-Averaged Navier–Stokes (RANS) approach represents the next level of approximation in modeling fluid flows. In this framework, the flow variables are decomposed into a mean and a fluctuating component. For example, the velocity in the x directions is expressed as $u = \bar{u} + u'$, following the Reynolds decomposition. The equations are then averaged, yielding a set of equations for the mean flow variables. These equations are similar in form to Eq. (2.30), but include an additional term known as the Reynolds stress tensor, which introduces new unknowns. To close the system, a turbulence model must be employed. This approach eliminates the need to resolve turbulent scales directly, reducing the computational cost compared to DNS and LES. This makes it a model of choice for cheaper high-fidelity simulations. However, the mesh still need to be adequately refined near the boundary layer to accurately capture its physics which makes it too expensive for early design.

Euler

The Euler equations describe the flow under the assumption of inviscid conditions ($\mu = 0$), as shown in Eq. (1.2). This model looses the capturing of the viscous and turbulent effects of the flow but offers the advantage of not requiring a turbulence model, which reduces computational cost. Additionally, since the Euler equations do not capture boundary layer effects, the mesh does not need to be refined in the wall-normal direction, further improving computational efficiency.

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0 \\ \frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla p &= 0 \\ \frac{\partial (\rho E)}{\partial t} + \nabla \cdot (\rho \mathbf{u} H) &= 0 \end{aligned} \tag{1.2}$$

Full Potential

The Full Potential (FP) model introduces further simplifications by assuming that the flow is both isentropic and irrotational. Under these assumptions, the velocity can be expressed as the gradient of a velocity potential, allowing the momentum and energy equations to be removed and only the continuity equation (Eq. 1.3) to be solved. This simplification makes the full potential model approximately 4 to 5 times more computationally efficient [9, 10] than the Euler model for 2D and 3D simulations, respectively.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1.3}$$

However, these additional assumptions come with some limitations. The model loses accuracy in the presence of shocks and requires the implementation of a Kutta condition to properly capture lifting flows. Despite these limitations, the error relative to traditional Euler solvers is estimated to remain below 2% for upstream shock local Mach numbers below 1.3 [9], which is sufficient for many commercial aircraft applications.

Linearized Potential Models

The final level of simplification assumes that the flow is incompressible (ρ = const) and linear. This results in a family of models where the solution can be computed directly on the surface of the geometry, providing maximum computational efficiency. However, this comes at the cost of lower fidelity, as these models cannot capture compressible or transonic effects. Examples of such models include Panel Methods where the actual geometry is represented and the Vortex-Lattice Method (VLM) where the geometry thickness is neglected. In these approaches, the governing equation becomes the Laplace equation, as shown in Eq. (1.4).

$$\nabla^2 \phi = 0 \tag{1.4}$$

1.2.2 Mesh Generation

Numerical aerodynamic solvers require the spatial discretization of the computational domain. In most cases, this involves discretizing the volumetric flow domain extending from the aircraft surface to a farfield boundary located sufficiently far from the body so it is not influenced by it. Fig. 1.3 is an example of a computational domain where the farfield boundary was brought closer to the geometry for illustrative purposes. An exception to this approach is found in linearized potential models, for which only the surface of the geometry is discretized.

In volumetric approaches, the computational domain is decomposed into a large number of small and simple geometric entities, commonly referred to as cells or elements. The governing equations described in Subsection 1.2.1 are then solved over each of these elements.

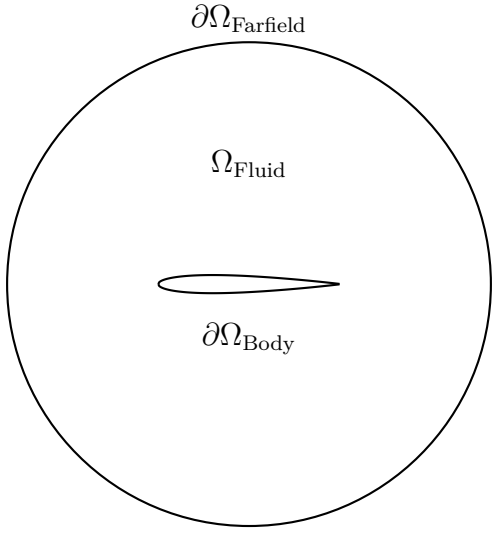


Figure 1.3 Illustration of a computational domain

This discretization process, commonly referred to as body-conforming mesh generation, represents the standard practice for aerodynamic simulations. This approach can become difficult to automate and often requires significant user intervention when dealing with complex or evolving geometries, such as those encountered in aero-icing simulations. In these cases, ice accretion leads to highly irregular surface features that significantly complicate mesh generation. An example of a body-conforming mesh generated around a glaze ice shape on an airfoil is shown in Fig. 1.4.

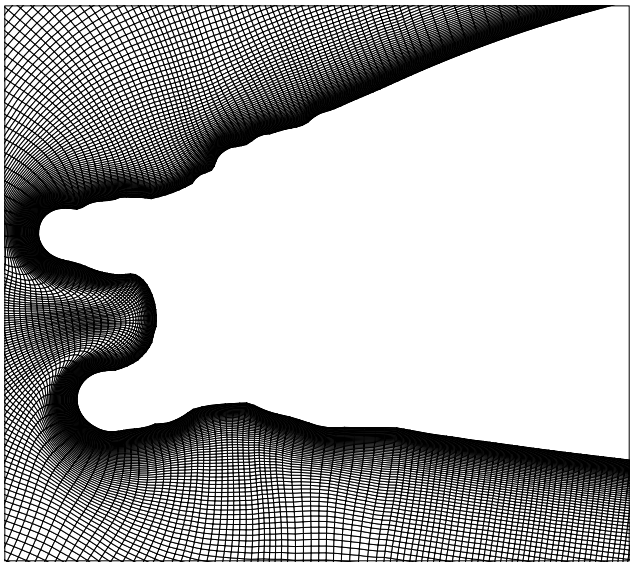


Figure 1.4 Body-conforming mesh of a glaze ice shape on an airfoil

1.2.3 Aero-Icing Numerical Modeling

Aero-icing is a multiphysics phenomenon governed by the complex interaction between aerodynamic flow, water droplet trajectories and thermodynamic heat-transfer processes. As a result, numerical icing simulations rely on a set of dedicated models, each responsible for resolving one of these physical aspects. These models are typically coupled within an iterative framework to predict ice accretion over time. A representative coupling strategy for an icing numerical model is illustrated in Fig. 1.5.

The process begins with the generation of a mesh around the clean geometry of interest, as described in Section 1.2.2. An aerodynamic simulation is then performed to compute the flow field, providing quantities such as density, velocity and pressure, which directly influence the droplet dynamics. In addition, the aerodynamic solver supplies surface-related quantities required by the thermodynamic model, including recovery temperature, wall pressure, friction coefficients and heat transfer coefficients.

Subsequently, a droplet simulation is carried out to determine the water impingement rate on the surface. This information is then used by a thermodynamic solver to compute the local ice accretion rate on each surface facet. Based on this accretion rate, a geometry solver modifies the surface to account for the newly formed ice. Since icing is an unsteady phenomenon, for improved accuracy, this procedure is typically performed in multiple successive icing layers and repeated iteratively until the prescribed total icing time is reached.

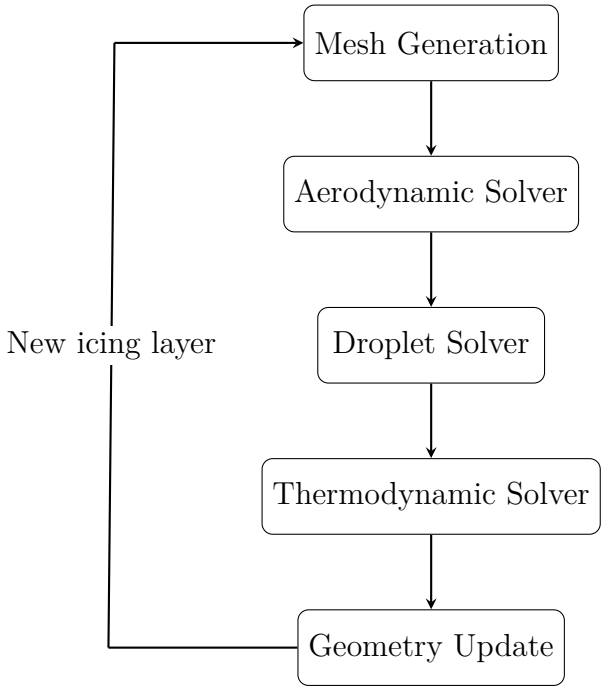


Figure 1.5 Icing numerical coupling framework

1.3 Key Aspects of the Problem

1.3.1 Balancing Fidelity and Computational Cost

As explained in Subsection 1.2.1, increasing model fidelity is associated with a rise in computational cost as shown in Fig. 1.2. Standard practice high-fidelity approaches based on RANS methods [11], while robust and accurate, are often too computationally expensive for early-stage design. On the other hand, low-fidelity models like the Non-Linear Lifting Line [12] and the Non-Linear Vortex-Lattice Method (NLVLM) [13, 14] can fall short in capturing critical flow phenomena, particularly in transonic regimes or around complex geometries. Therefore, a large range of fidelity is often desired to be able to handle all design phases more efficiently.

1.3.2 Body-Fitted Meshes Generation Challenges

Another major challenge encountered in numerical aerodynamic simulations is the generation of body-fitted meshes [15]. For design optimization and multidisciplinary analyses, such as aero-icing or aeroelastic simulations, mesh generation must be fully automated and capable of handling frequent geometric modifications as explained in Subsection 1.2.2. This requirement becomes especially challenging when the geometry evolves significantly or when it increases

in complexity. This feature further motivates the development of numerical approaches that eliminates the dependency on body-conforming meshes.

1.3.3 Aircraft Icing

Aircraft icing is a multidisciplinary phenomenon that occurs when supercooled water droplets come into contact with the aircraft surface. At very low temperatures, droplets freeze almost immediately upon impact, leading to the formation of rime ice. At higher temperatures, droplets may not freeze instantaneously and can run back along the surface before solidifying, resulting in glaze ice formations characterized by more complex geometries, such as ice horns. In some conditions, both mechanisms may occur simultaneously, leading to mixed ice accretion. Ice formation on aircraft wings can significantly alter the aerodynamic behavior of the flow, causing a reduction in lift and an increase in drag, as illustrated in Fig. 1.6. These effects can lead to stall at lower angles of attack and degrade the effectiveness of control surfaces, potentially resulting in loss of control.

The severity of these effects has been tragically illustrated in recent years. Indeed, during a domestic flight in Brazil, a regional turboprop aircraft operating in severe icing conditions experienced substantial aerodynamic performance degradation and subsequently lost control [16]. This highlights the necessity of integrating aero-icing simulations earlier in the aircraft design process in order to better predict, assess and mitigate the associated risks.

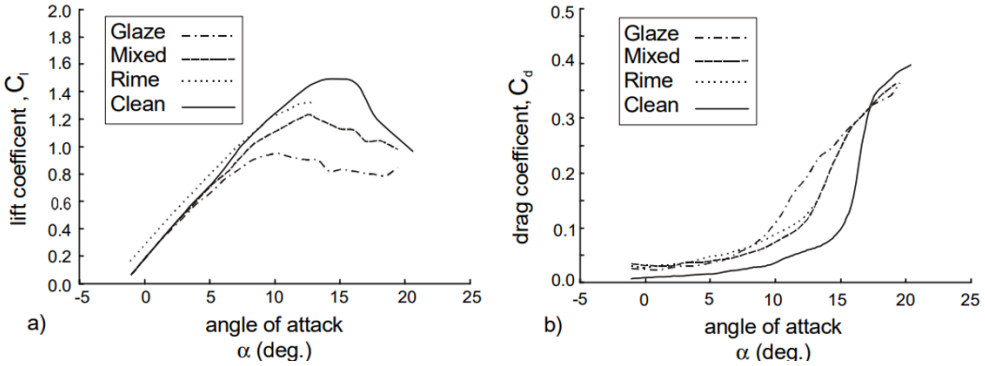


Figure 1.6 Effect of different airfoil ice formations on lift and drag coefficients - [4]

1.4 Research Objectives

The aim of the present research is to address the challenges in early aircraft design identified in Section 1.3. In order to mitigate these challenges, the research is structured around three specific sub-objectives:

1. To develop a medium-fidelity numerical aerodynamic solver based on the full potential model to accurately capture subsonic and transonic flow phenomena in a preliminary design context.

2. To facilitate mesh generation for complex and evolving geometries by integrating an immersed boundary ghost-cell method into the full potential solver, thereby eliminating the need for body-conforming meshes.

3. To conduct aero-icing simulations by coupling the developed aerodynamic solver with an icing simulation suite.

1.5 Plan of Thesis

To address the research objectives, this thesis is organized into three main chapters: literature review; development of a full potential solver for complex geometries using an immersed boundary method and boundary-layer coupling; and aero-icing simulations.

Chapter 2, the literature review, presents mesh generation techniques, numerical methods for solving partial differential equations and the development history of full-potential solvers. It also introduces immersed boundary methods, the fundamentals of boundary layer modeling and the main components of numerical icing simulation frameworks, including solver options for each module.

Chapter 3 describes the development of a finite-volume, unstructured full potential solver using body-fitted meshes, along with validation against Euler solutions. It then presents the integration of an immersed boundary method based on the ghost-cell approach and discusses the associated results. Finally, the boundary layer model used to couple with the inviscid solvers is introduced, along with comparisons to reference results from the literature.

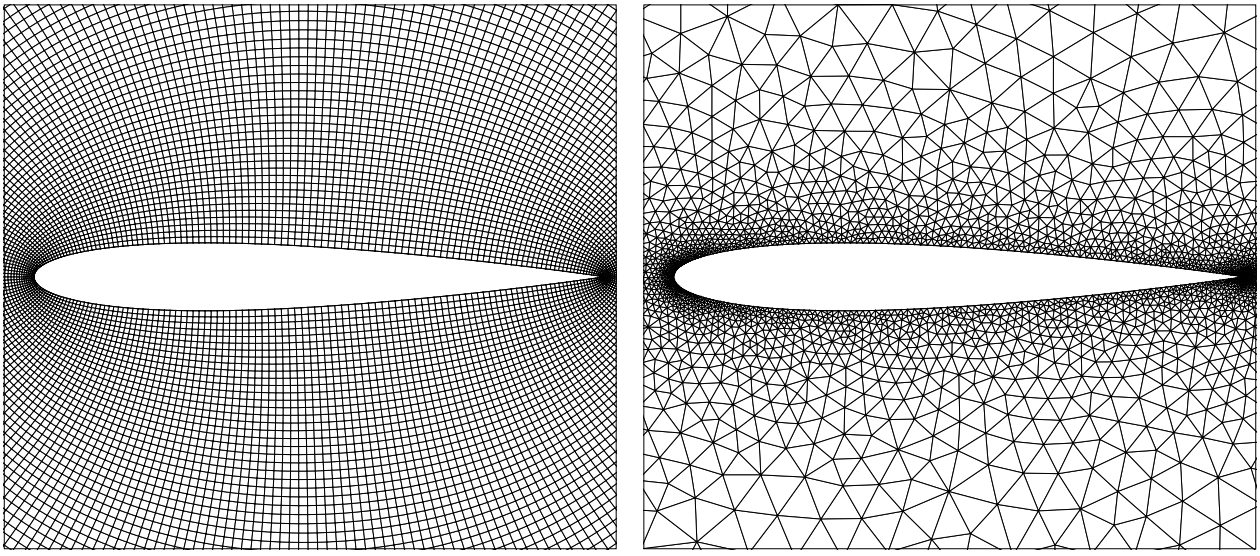
Chapter 4 presents icing simulations obtained by integrating the developed aerodynamic solvers into the CHAMPS icing framework. Cases 241 and 242 from the first Ice Prediction Workshop (IPW1) are considered. The results are compared with experimental data, other participants' results and RANS simulations from CHAMPS.

The thesis concludes with a summary of the research work, a discussion of the identified limitations and recommendations for future research.

CHAPTER 2 LITERATURE REVIEW

2.1 Mesh Generation

As stated in Subsection 1.2.2, the application of a numerical aerodynamic model requires the computational domain to be subdivided without overlapping into a finite number of smaller and simpler geometric entities, commonly referred to as cells or elements. This process results in what is called a mesh or a grid. The quality and structure of this mesh play a crucial role in the accuracy, robustness and computational efficiency of the numerical simulation. The meshes can be classified into two main families: structured and unstructured. Examples of both meshes types for a NACA0012 airfoil is illustrated at Fig.2.1.



(a) Structured mesh (b) Unstructured mesh

Figure 2.1 Structured and unstructured meshes for a NACA0012 airfoil geometry

2.1.1 Structured Meshes

Structured meshes are characterized by a regular and ordered connectivity. In two dimensions, they are composed of quadrilateral elements, while in three dimensions they consist of hexahedral elements. The connectivity between neighboring cells is implicitly defined through an index array arrangement, typically using index systems such as (i, j) or (i, j, k), as illustrated in Fig. 2.2. This implicit connectivity eliminates the need for storing explicit neighbors or nodes information, leading to improved computational efficiency.

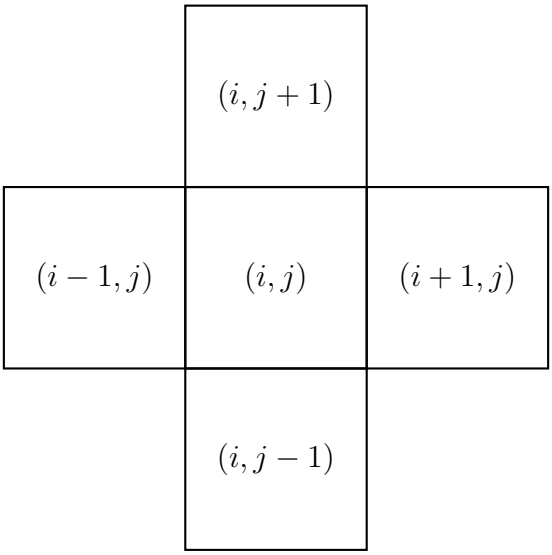


Figure 2.2 Structured quadrilateral cell (i, j) and its nearest neighbors

Another advantage of structured meshes lies in the grid alignment with the predominant flow direction and regions of strong gradients, such as boundary layers or shock waves. This alignment generally enhances numerical accuracy compared to unstructured meshes [17].

Despite these advantages, structured mesh generation can be challenging, particularly for complex geometries. The construction of high-quality structured grids often requires using an algebraic technique or solving Partial Differential Equations (PDEs) and may involve significant user intervention such as explained in [18,19]. As a result, generating structured meshes for configurations such as multi-element airfoils or iced geometries can be time-consuming and difficult.

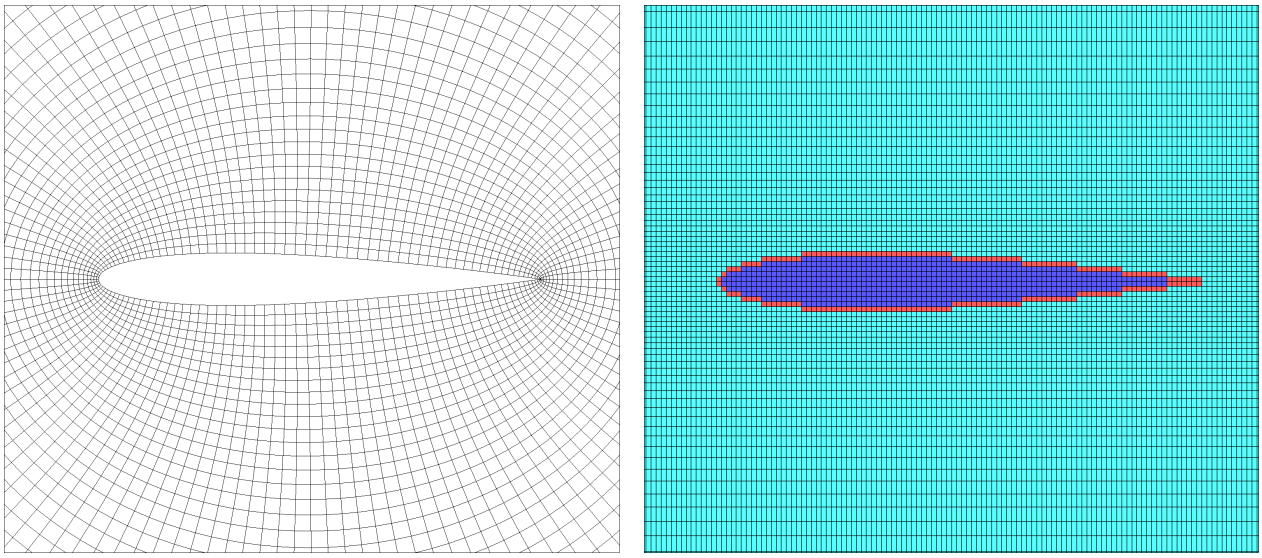
2.1.2 Unstructured Meshes

Conversely, unstructured meshes are defined by irregular and unordered connectivity, which requires explicitly storing the relationships between cells, nodes and faces in a connectivity table demanding more computational resources. This type of mesh offers greater flexibility, supporting a wide variety of cell shapes, such as triangles or quadrilaterals in 2D and tetrahedra, hexahedra, prisms or pyramids in 3D. Common generation techniques include Delaunay triangulation and advancing front methods [20]. Usually, generating unstructured meshes is easier than structured meshes. Nevertheless, complex geometries may still require user intervention during the mesh generation process [8].

2.1.3 Body-Fitted and Immersed Boundary Meshes

Meshes may also be classified into two additional categories: body-fitted meshes and immersed boundary meshes. In numerical aerodynamics, body-fitted meshes remain the standard approach. However, it introduces significant challenges in terms of automation, particularly when dealing with complex or evolving geometries.

These challenges can be mitigated by using immersed boundary grids which simplify greatly the mesh generation process and improve automation. Nevertheless, the use of immersed boundary meshes requires using immersed boundary methods, which increases the algorithmic complexity and demands additional care within the implementation of the aerodynamic solver. These methods are discussed in more detail in Section 2.4. A comparison between body-fitted and immersed boundary meshes for a NACA0012 airfoil is shown in Fig. 2.3.



(a) Body-fitted mesh

(b) Immersed boundary mesh

Figure 2.3 Body-fitted and immersed boundary meshes for a NACA0012 airfoil geometry

2.2 Numerical Discretization Methods

The numerical solution of PDE requires their discretization in space. Three classical approaches are commonly employed for this purpose: the finite difference method, the finite element method and the finite volume method [8, 21].

2.2.1 Finite Difference Method

The Finite Difference Method (FDM) is the oldest among the classical discretization techniques used to solve partial differential equations. It relies on the fact that a function can be locally approximated by a Taylor series expansion, as given in Eq. (2.1).

$$f(x) = \sum_{i=0}^{n} \frac{f^{(i)}(a)}{i!}(x-a)^i + \mathcal{O}\Big((x-a)^{n+1}\Big), \quad \text{as } x \to a. \tag{2.1}$$

The expansion point is generally chosen as $a = x_i$, while x is written as $x_i + c\,\Delta x_i$, where c is an integer corresponding to neighboring grid points in the vicinity of node i, as illustrated in Fig. 2.4. Taylor expansions about x_i are then written at several neighboring nodes and linearly combined to construct discrete approximations of the spatial derivatives at node i. The lowest power of Δx appearing in the resulting error $\mathcal{O}$ of the approximation expression determines the theoretical order of accuracy of the truncation error.

Once the derivative approximations are established, the governing PDE are enforced at each grid node, leading to a system of linear equations. The solution of this system yields the discrete approximation of the unknown function values at the grid nodes.

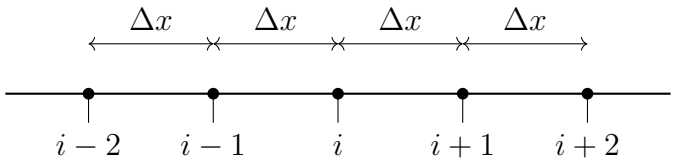


Figure 2.4 One-dimensional finite-difference stencil centered at node i.

The main advantage of the FDM lies in its conceptual simplicity and ease of implementation. Moreover, higher-order accuracy can be achieved using higher-order derivative approximations which generally involve wider stencils. However, the method also presents significant limitations. It is naturally restricted to cartesian grids and generally requires a structured body-fitted coordinate system to be transformed into a cartesian coordinate system, with the governing equations reformulated accordingly [22]. This requirement can become particularly restrictive when dealing with complex geometries.

2.2.2 Finite Element Method

The Finite Element Method (FEM) can be applied to arbitrary computational domains discretized using meshes generally composed of triangular or quadrilateral elements in two

dimensions and tetrahedral or hexahedral elements in three dimensions. This geometric flexibility constitutes a major advantage over FDM, particularly for problems involving complex geometries.

In FEM, the governing partial differential equations are first multiplied by a set of weight or test functions and the resulting expressions are integrated over the computational domain. Through integration by parts, a weak formulation of the governing equations is obtained. The solution is then approximated locally within each element using shape functions expressed in terms of the element's nodal values. These shape functions may be linear, quadratic, or of higher polynomial order, with the chosen interpolation order directly influencing the accuracy of the method.

Comprehensive descriptions of the finite element method or examples of implemantations can be found in [21, 23–29]. Like FDM, one of the main strengths of FEM lies in its natural adaptability to higher-order accuracy through the use of higher-order shape functions. However, since the method is formulated in a weak sense, it does not inherently guarantee strict conservation of the governing variables and generally involves greater implementation complexity.

2.2.3 Finite Volume Method

Similar to FEM, the Finite Volume Method (FVM) can be applied to computational domains of arbitrary complexity. It supports a wider range of element shapes, including triangles or quadrilaterals in two dimensions and polyhedral elements such as tetrahedra, hexahedra, prisms or pyramids in three dimensions.

In FVM, the governing partial differential equations are integrated over a set of control volumes. These control volumes may coincide with the mesh cells in a cell-centered scheme, or be constructed around mesh vertices by connecting the centroids of surrounding cells in a cell-vertex scheme. The control volumes associated with both approaches are illustrated in Fig. 2.5. The divergence theorem is then applied to convert volume integrals of divergence terms into surface integrals of fluxes across the control volume boundaries. These fluxes are typically evaluated at the centers of the control volume faces. Terms in the governing equations that are not expressed in divergence form are treated as source terms and are commonly approximated using the value at the control volume center multiplied by the control volume size. The sum of fluxes and source contributions defines the residual of each control volume, which is iteratively driven to zero to obtain the solution.

(a) Cell-centered scheme (b) Cell-vertex scheme

Figure 2.5 Cell-centered and cell-vertex control volume schemes

The main advantages of the finite volume method are its relative ease of implementation, its conservative nature by construction and the fact that discretization may be performed directly in physical space without requiring coordinate transformations. A notable limitation of FVM is that the overall order of accuracy often depends on the specific numerical schemes used to evaluate the fluxes. Consequently, achieving higher-order accuracy generally requires the development or adoption of more advanced flux reconstruction schemes. Comprehensive descriptions of the finite volume method and examples of practical implementations can be found in [8–10, 19, 21, 30–37].

2.2.4 Iterative Methods for Solving Linear Systems of Equations

Regardless of whether FDM, FEM or FVM is employed to discretize the governing PDEs, the resulting formulation ultimately leads to a linear system of equations that must be solved to obtain the solution at each control point. This system can be written in the general form

$$\mathbf{A}\,\mathbf{x} = \mathbf{b},$$

where $\mathbf{A}$ is a square matrix of size $n \times n$, with n corresponding to the number of control points multiplied by the number of unknowns in the governing equations.

In practical applications, the matrix $\mathbf{A}$ is typically very large and sparse. Moreover, for unstructured discretizations, it is generally unbanded. As a result, direct solution techniques, based on matrix factorization or inversion, quickly become computationally prohibitive for multidimensional problems as the number of control points increases. Consequently, iterative methods are preferred for solving such linear systems.

Among the most commonly used iterative solvers applicable to both structured and unstruc-

tured discretizations are the Jacobi method, the symmetric Gauss-Seidel method and the Generalized Minimal RESidual (GMRES) method.

Jacobi Method

The Jacobi method [38] is based on a decomposition of the system matrix $\mathbf{A}$ into the sum of a strictly lower triangular matrix $\mathbf{L}$, a diagonal matrix $\mathbf{D}$ and a strictly upper triangular matrix $\mathbf{U}$, such that

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}. \tag{2.2}$$

To initiate the iterative procedure, an initial approximation of the solution vector, denoted $\mathbf{x}^{(0)}$, must be provided. This initial guess is typically chosen as the zero vector or as the previous solution, depending on whether the algorithm is used to compute variable updates or the variables themselves. In many practical applications, the expected solution corrections are small, making such initializations reasonable.

The Jacobi iteration updates the solution by the following recurrence relation:

$$\mathbf{x}^{(k+1)} = \mathbf{D}^{-1}\left(\mathbf{b} - (\mathbf{L} + \mathbf{U})\mathbf{x}^{(k)}\right), \tag{2.3}$$

where k denotes the inner iteration index.

At each inner iteration, the non-diagonal contribution of the matrix, represented by $\mathbf{L}+\mathbf{U}$, is transferred to the right-hand side and evaluated using the solution from the previous iteration. This approach implicitly assumes that the current approximation $\mathbf{x}^{(k)}$ is sufficiently close to the updated solution $\mathbf{x}^{(k+1)}$. The inner iteration solution is then obtained by applying the inverse of the diagonal matrix $\mathbf{D}$, which is trivial and computationally inexpensive since it involves only scalar divisions.

One of the main advantages of the Jacobi method is its low computational cost per iteration. Additionally, the algorithm is naturally parallelizable, as each component of $\mathbf{x}^{(k+1)}$ depends only on values from the previous inner iteration. However, the method generally requires a large number of inner iterations to achieve convergence, which can lead to slow overall convergence compared to more advanced iterative solvers. A sufficient condition for convergence is that the matrix $\mathbf{A}$ be strictly diagonally dominant, although convergence may still occur even when this condition is not satisfied [39].

Symmetric Gauss–Seidel Method

The Symmetric Gauss-Seidel (SGS) method [38] is closely related to the Jacobi method, with the key difference that once a new value $\mathbf{x}_i^{(k+1)}$ is computed for a given cell i, it is immediately used in all subsequent calculations within the same inner iteration.

As in the Jacobi method, the matrix $\mathbf{A}$ is first decomposed according to Eq. (2.2).

After setting an initial guess $\mathbf{x}^{(0)}$, the SGS iteration proceeds in two sweeps per inner iteration: a forward sweep followed by a backward sweep. In matrix form, these steps are expressed as:

$$(\mathbf{L} + \mathbf{D})\,\mathbf{x}^{(k+1/2)} = \mathbf{b} - \mathbf{U}\mathbf{x}^{(k)}, \tag{2.4}$$

$$(\mathbf{D} + \mathbf{U})\,\mathbf{x}^{(k+1)} = \mathbf{b} - \mathbf{L}\mathbf{x}^{(k+1/2)}, \tag{2.5}$$

where $\mathbf{x}^{(k+1/2)}$ denotes the intermediate solution after the forward sweep of iteration k. Each of these linear systems can be solved using Gaussian elimination, proceeding from top to bottom for the lower-triangular system $\mathbf{L}+\mathbf{D}$ and from bottom to top for the upper-triangular system $\mathbf{D} + \mathbf{U}$.

In a cell-wise notation, the forward and backward sweeps can be written as:

$$\text{Forward sweep: } x_i^{(k+1/2)} = \frac{1}{a_{i,i}} \left(b_i - \sum_{j<i} a_{i,j} x_j^{(k+1/2)} - \sum_{j>i} a_{i,j} x_j^{(k)} \right), \quad i = 1, \ldots, N \tag{2.6}$$

$$\text{Backward sweep: } x_i^{(k+1)} = \frac{1}{a_{i,i}} \left(b_i - \sum_{j>i} a_{i,j} x_j^{(k+1)} - \sum_{j<i} a_{i,j} x_j^{(k+1/2)} \right), \quad i = N, \ldots, 1 \tag{2.7}$$

The main advantage of the SGS method is its faster convergence, since updated values are used immediately within the same iteration. However, this sequential dependency, requiring $x_i^{(k+1/2)}$ to be computed before $x_{i+1}^{(k+1/2)}$ in the forward sweep and before $x_{i-1}^{(k+1)}$ in the backward sweep, makes the algorithm inherently sequential and less prone to parallelization, unlike the Jacobi method. A sufficient condition for convergence is that $\mathbf{A}$ be either strictly diagonally dominant or symmetric positive-definite [39]. Nonetheless, the algorithm may converge even if these conditions are not strictly satisfied.

GMRES Method

The GMRES method, introduced by Yousef Saad and Martin H. Schultz in 1986 [40], is more complex to implement than Jacobi or SGS but is often significantly more efficient for solving large, sparse linear systems. The GMRES algorithm seeks an approximate solution to the

linear system (3.45) in the form:

$$\mathbf{x} = \mathbf{x_0} + \mathbf{z}, \tag{2.8}$$

where $\mathbf{x}_0$ is an initial guess and $\mathbf{z}$ lies in the Krylov subspace K_m defined by

$$\mathbf{z} \in K_m, \;\; K_m \equiv \text{span}\left\{\mathbf{r_0},\, A\mathbf{r_0},\, A^2\mathbf{r_0},\, ...,\, A^{m-1}\mathbf{r_0}\right\} \tag{2.9}$$

with $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$ the initial residual and m the number of search directions (Krylov vectors). The GMRES solution $\mathbf{z}$ is obtained by minimizing the residual in a least-squares approach:

$$\|\mathbf{b} - A\left(\mathbf{x_0} + \mathbf{z}\right)\|. \tag{2.10}$$

An efficient implementation of this procedure is detailed in [8].

While GMRES is more complex to implement, it is highly effective for a wider range of matrices $\mathbf{A}$. Its performance can be further enhanced using preconditioners that cluster the eigenvalues of $\mathbf{A}$ near unity. A widely used preconditioner is ILU(0) [38]. Additionally, matrix reordering techniques such as the Cuthill-McKee algorithm [41] can reduce the bandwidth of $\mathbf{A}$, improving computational efficiency.

2.3 Full Potential

2.3.1 Governing Equations and Mathematical Properties

The FP formulation is derived from the mass conservation equation of the Navier–Stokes system,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{2.11}$$

where ρ denotes the density and $\mathbf{u}$ the flow velocity. If the flow is assumed to be irrotational ($\nabla \times \mathbf{u} = \mathbf{0}$), the velocity field can be expressed as the gradient of a scalar velocity potential ϕ:

$$\nabla \phi = \mathbf{u}. \tag{2.12}$$

Under the additional assumption of steady flow, the temporal derivative in Eq. (2.11) vanishes, yielding the steady conservative FP equation,

$$\nabla \cdot (\rho \nabla \phi) = 0. \tag{2.13}$$

Equation (2.13) represents a single equation involving two unknowns, namely the density ρ and the velocity potential ϕ. Closure of the system is obtained by relating the density to the velocity field through the compressible Bernoulli equation combined with the isentropic flow assumption. This leads to an explicit expression of the density as a function of the velocity potential,

$$\rho = \left(1 + \frac{\gamma - 1}{2} M_\infty^2 \left(1 - \|\nabla\phi\|^2\right)\right)^{\frac{1}{\gamma - 1}}, \tag{2.14}$$

where γ is the specific heat ratio and M_∞ is the freestream Mach number.

Substituting Eq. (2.14) into the conservative form of the governing equation (2.13) and expanding the ∇ operator using the chain rule, yield the non-conservative form of the full potential equation in three dimensions:

$$\left(a^2 - u^2\right)\phi_{xx} + \left(a^2 - v^2\right)\phi_{yy} + \left(a^2 - w^2\right)\phi_{zz} - 2uv\phi_{xy} - 2uw\phi_{xz} - 2vw\phi_{yz} = 0, \tag{2.15}$$

where (u, v, w) are the velocity components and a is the local speed of sound.

Although Eqs. (2.13) and (2.15) are mathematically equivalent, their numerical solutions may differ. This discrepancy appears in transonic regimes, where the non-conservative formulation introduces a spurious mass-flow term in the vicinity of shock waves. Fortuitously, this numerical artifact can partially compensate for the modeling error associated with the isentropic shock assumption, often resulting in improved agreement with experimental data [42–44].

For simplicity, the following analysis is restricted to two-dimensional flow by neglecting variations in the z-direction. Equation (2.15) then reduces to

$$\left(a^2 - u^2\right)\phi_{xx} + \left(a^2 - v^2\right)\phi_{yy} + -2uv\phi_{xy} = 0. \tag{2.16}$$

Comparing Eq. (2.16) with the general second-order quasi-linear PDE,

$$A\phi_{xx} + B\phi_{xy} + C\phi_{yy} = F, \tag{2.17}$$

the coefficients are identified as $A = a^2 - u^2$, $B = 2uv$ and $C = a^2 - v^2$. The discriminant of the equation is therefore given by:

$$\Delta = B^2 - 4AC = 4(u^2v^2 - a^4 - u^2v^2 + u^2a^2 + v^2a^2) = 4a^2(\|\mathbf{u}\|^2 - a^2). \tag{2.18}$$

The type of the governing equation depends on the sign of Δ. When $\|\mathbf{u}\| < a$, corresponding

to subsonic flow, the equation is elliptic. At $\|\mathbf{u}\| = a$, the equation becomes parabolic. When $\|\mathbf{u}\| > a$, i.e. in supersonic regions, the equation is hyperbolic.

The characteristic directions associated with Eq. (2.16) are obtained from the general expression:

$$\left(\frac{dy}{dx}\right)_{1,2} = \frac{-B \pm \sqrt{\Delta}}{2A} = \frac{-uv \pm \sqrt{a^2(\|\mathbf{u}\|^2 - a^2)}}{a^2 - u^2}. \tag{2.19}$$

In subsonic regions, the characteristics are complex conjugates, while at sonic conditions a single repeated real characteristic exists. In supersonic regions, two distinct real characteristics are present. Because the nature of the characteristics varies with the local Mach number, the treatment of farfield boundary conditions in FP solvers differs fundamentally from Euler solvers, where boundary conditions are typically imposed using Riemann invariants. Similarly, the definition of a Courant–Friedrichs–Lewy (CFL) stability condition is not straightforward and cannot be directly adopted from Euler-based formulations.

2.3.2 Numerical Formulations

In this subsection, the various numerical methods used to solve the FP equation are enumerated. This overview is based on the comprehensive literature reviews provided by Holst [42] and Crovato [23].

Early efforts to numerically solve the FP equation date back to the early 1970s and initially focused on the steady Transonic Small Disturbance (TSD) equation,

$$\left(1 - M_\infty^2 - M_\infty^2(\gamma + 1)\,\frac{\varphi_x}{u_\infty}\right)\varphi_{xx} + \varphi_{yy} + \varphi_{zz} = 0, \tag{2.20}$$

where $\varphi = \phi - \phi_\infty$ denotes the perturbation potential. The TSD equation is derived as an approximation of the non-conservative FP formulation under the assumption that the velocity perturbations φ_x, φ_y and φ_z remain small compared to the freestream velocity u_∞. As a result, the TSD model is only valid for flows predominantly aligned with a single direction and loses accuracy near stagnation points or for thick geometries.

A major difficulty in solving transsonic flow with the TSD equation arises from its mixed-type nature: the equation is elliptic in subsonic regions and hyperbolic in supersonic regions, similarly to the FP equation. In 1971, Murman and Cole [45] introduced a type-dependent differencing scheme that applied central differencing in subsonic regions and upwind differencing in supersonic regions. This innovation enabled stable transonic simulations using a poten-

tial formulation. The approach was subsequently extended to axisymmetric bodies [46, 47], three-dimensional wings [48–50] and wing–body configurations [51–54]. Nevertheless, the restrictive assumptions inherent to the TSD model limited its applicability.

To overcome these limitations, researchers turned toward the numerical solution of the FP equation. The first successful implementations were reported by Steger and Lomax [55] and by Garabedian and Korn [56]. These early works employed the non-conservative FP formulation discretized using finite-difference methods on body-fitted structured grids, relying on conformal mappings of airfoil geometries onto a unit circle. The first direct coupling between a FP solver and a boundary-layer model was introduced in [57], using a semi-empirical turbulent boundary-layer correction [58]. During this period, most FP solvers relied on Successive Line Over-Relaxation (SLOR), a relaxed Gauss–Seidel method well suited for structured grids.

The extension of FP solvers to fully three-dimensional configurations was achieved by Jameson and Caughey with the FLO22 code [59, 60]. Their solver employed a shear–parabolic conformal mapping and a rotated differencing scheme to better align the numerical domain of dependence with the physical one. Immersed boundary approaches were also explored to alleviate mesh generation constraints. Early Cartesian embedded-grid methods using cut cells were proposed in [61], while ghost-cell immersed boundary methods were later introduced for FP solvers in [62].

The conservative formulation of the FP equation was first solved numerically in three dimensions by Jameson [63]. Stability in transonic regimes was achieved through the introduction of artificial viscosity, leading to the development of FLO27, FLO28 and FLO30 solvers [64–66]. These solvers progressively enabled simulations of realistic wing–fuselage configurations using increasingly complex conformal mappings. FLO30 was later enhanced through direct viscous–inviscid coupling with a boundary-layer model [67, 68]. Another notable finite-volume FP solver is the commercial code blwf [69], which employs a chimera mesh strategy [70] to discretize individual aircraft components and includes quasi-simultaneous boundary-layer coupling.

Unstructured-grid approaches for FP solvers emerged in the late 1990s. Neel [9] developed the first finite-volume FP solver on unstructured meshes and demonstrated its capabilities for transonic lifting wings. Building on this work, Lyu et al. [10] introduced a highly automated finite-volume FP solver compatible with quadtree and octree meshes, incorporating a cut-cell strategy and a solution-adaptive mesh refinement. Their solver was successfully applied to wing–body configurations. Further finite-volume developments on unstructured-grid include the work of Liegl [37], who solved the FP equation on unstructured tetrahedral grids and

used a panel method to initialize circulation more accurately. He also noticed convergence difficulties in transonic regimes. Parrinello and Mantegazza [33, 34] proposed a two-field FP formulation that simplifies unsteady simulations and demonstrated their approach on the ONERA M6 wing, while also noting reduced convergence rates in transonic cases.

The FEM has also been explored for solving the FP equation. Nishida [26] developed a simultaneous coupling of a FEM FP solver with a boundary-layer model and reported good agreement with experimental data for transonic wing flows. Galbraith et al. [24] adapted the two-field FP formulation of Parrinello and Mantegazza into a FEM framework and demonstrated its applicability to wing–body–tail configurations. Eller [27] developed a three-dimensional unstructured FEM FP solver, but required an infinitesimal wake gap in the mesh, complicating mesh generation. This limitation was addressed by cutting wake-crossing elements [71] or by approximating the wake within the mesh resolution [29]. Crovato [23] further extended FEM FP solvers to enable aeroelastic simulations in the transonic regime. Núñez et al. [28] proposed an embedded FEM FP solver using a level-set representation of both geometry and wake. A widely used commercial FEM FP solver is Tranair [72, 73], which relies on a variational formulation derived from the Bateman principle which is modified for cells crossing the wall boundary. Tranair is popular since it enables solution-driven refinement and mesh sequencing on quadtree and octree immersed boundary grids.

Finally, the FP equation can also be solved using field panel methods. In this approach, the FP equation is rewritten as a Poisson equation with a linear Laplacian operator on the left-hand side and a nonlinear source term on the right-hand side,

$$\begin{aligned} \nabla^2 \phi &= \sigma, \\ \sigma &= \frac{1}{\rho} \nabla \rho \cdot \nabla \phi. \end{aligned} \tag{2.21}$$

The linear part is solved using a panel method on the boundaries, while the nonlinear source term is evaluated in the field using finite differences [74–76]. A major advantage of this technique is its ability to operate on non-body-conforming grids, greatly simplifying mesh generation. This approach has been successfully implemented in the commercial software FlightStream, where it is employed within a dedicated transonic flow solver [77]. However, the method remains challenging to implement and is less commonly used [23].

2.3.3 Stabilizing Transonic Simulations

The principal difficulty in the numerical solution of the FP equations lies in the change of mathematical type of the governing equation, which is elliptic in subsonic regions and hyperbolic in supersonic regions. A first successful treatment of this issue was proposed by Murman and Cole [45] through a type-dependent differencing scheme. Later, Jameson [63] introduced numerical stabilization via artificial viscosity. Subsequently, Hafez et al. [78] developed a density upwinding technique in supersonic regions to stabilize transonic flows.

Due to its simplicity, density upwinding has become one of the most widely used stabilization strategies. In this approach, the density ρ appearing in the mass flux is replaced by an upwinded value $\tilde{\rho}$. Several formulations for upwinding the density have been proposed in the literature. For instance, in [9, 10], the upwinded density is defined as:

$$\tilde{\rho} = \rho - \frac{\mu}{\|\mathbf{u}\|} \left(u\rho_x \Delta x + v\rho_y \Delta y + w\rho_z \Delta z \right), \tag{2.22}$$

where μ is a switching function that activates artificial compressibility when the Mach number exceeds a prescribed threshold M_C.

To avoid the explicit computation of density gradients, a simpler formulation was later proposed by Crovato [23], in which the upwinded density is given by:

$$\tilde{\rho} = \rho - \mu \left(\rho - \rho_U \right), \tag{2.23}$$

where ρ_U denotes the density in the upwind direction.

Two main forms of the switching function μ are commonly found in the literature:

$$\mu = \mu_C \, max \left(0, M^2 - M_C^2 \right) \tag{2.24}$$

and

$$\mu = \mu_C \, max \left(0, \left(1 - \frac{M_C^2}{M^2} \right) \right) \tag{2.25}$$

where M is the Mach number of the upwind cell, M_C is the cut-off Mach number above which density upwinding is applied and μ_C is the artificial compressibility coefficient. Both formulations yield nearly identical results, as reported in [79, 80].

An alternative stabilization strategy, closely related to density upwinding, is flux upwinding.

In this method, the entire mass flux is upwinded rather than only the density [81]. The modified density used in the flux computation is expressed as

$$\tilde{\rho} = \rho - \frac{\|\mathbf{\Delta x}\|}{\|\mathbf{u}\|^2}\left(u(\overline{\rho\|\mathbf{u}\|})_x + v(\overline{\rho\|\mathbf{u}\|})_y + w(\overline{\rho\|\mathbf{u}\|})_z\right), \tag{2.26}$$

with

$$\overline{\rho\|\mathbf{u}\|} = \begin{cases} 0, & \text{if } M < 1, \\ \rho\|\mathbf{u}\| - \rho^*\|\mathbf{u}\|^* & \text{if } M \geq 1, \end{cases} \tag{2.27}$$

where ρ^* and $|\mathbf{u}|^*$ denote the density and velocity magnitude at sonic conditions ($M = 1$).

Flux upwinding has been shown to provide improved resolution of weak shocks, resulting in less numerical smearing [82]. However, despite this advantage, it is less frequently adopted in modern implementations than density upwinding.

Moreover, recent FEM FP formulations introduce density clamping through Padé approximations to enhance robustness. This approach mitigates numerical issues encountered at high velocities, where the density calculation (see Eq. (2.14)) would require negative arguments in an expression involving rational powers [23, 24].

2.3.4 Entropy Correction

One of the main advantages of solving the FP equation in conservative form is that it enables the use of an entropy correction to alleviate errors associated with entropy generation across shock waves, which violates the isentropic assumption underlying the FP model. Several implementations of this approach have been proposed in the literature [33, 83, 84]. In this framework, a non-isentropic density is introduced and computed as

$$\rho_{non-isen} = \rho_{isen} e^{\frac{\Delta s}{R}}, \tag{2.28}$$

where ρ_{isen} denotes the isentropic density obtained from the FP solver, R is the specific gas constant and Δs represents the entropy increase across a normal shock.

The entropy increase is evaluated using the Rankine-Hugoniot relations and is expressed as

$$\frac{\Delta s}{R} = \frac{1}{\gamma - 1}\ln\left[\frac{2\gamma M_1^2 - (\gamma - 1)}{\gamma + 1}\right] - \frac{\gamma}{\gamma - 1}\ln\left[\frac{(\gamma + 1)M_1^2}{(\gamma - 1)M_1^2 + 2}\right], \tag{2.29}$$

where M_1 is the local Mach number immediately upstream of the shock.

The application of this correction requires identifying the shock wave location, which is typically achieved by detecting two neighboring cells along the flow direction for which the local Mach number transitions from supersonic ($M > 1$) to subsonic ($M < 1$).

2.4 Immersed Boundary Methods

Following the classification works done by Lavoie [35] and Elices [32], the Immersed Boundary Methods (IBMs) are a family of approaches enabling simulations over immersed boundaries meshes and are declined in three specific implementations: cut-cell, continuous forcing and discrete forcing methods. Each of these methods will be addressed in the following subsections.

2.4.1 Cut-Cell Method

The cut-cell method can be regarded primarily as a geometric approach, in which most of the effort is concentrated in the preprocessing stage to adapt an immersed boundary grid into a wall-conforming grid. The method was originally introduced for two-dimensional inviscid flows in the context of the FP equations in [85] and later extended to the Euler equations for multi-element airfoils in [86]. Subsequent developments expanded the methodology to three-dimensional viscous flow applications at low Reynolds numbers, as reported in [87].

Several successful implementations of the cut-cell approach exist in the literature. For instance, Cart3D [88] applies this technique to solve inviscid Euler flows over complex three-dimensional geometries. More recently, an application of the cut-cell methodology within a finite-volume FP solver has been presented in [10]. In general, this method is better suited for inviscid flow models, since viscous and turbulent formulations, such as RANS, typically require strong mesh refinement in the wall-normal direction. Nevertheless, viscous cut-cell formulations have been demonstrated for two-dimensional geometries at both low and high Reynolds numbers in [89].

The cut-cell procedure begins with the identification of grid cells intersected by the solid boundary. These cells are then geometrically modified by removing the portion of their volume that lies inside the solid domain. Figure 2.6 illustrates this process for a two-dimensional configuration. While conceptually straightforward in two dimensions, the geometric complexity increases significantly for three-dimensional applications. As a result, the flow solver must be capable of handling polygonal control volumes in two dimensions and polyhedral control volumes in three dimensions, each with a variable number of edges or faces.

A major advantage of the cut-cell method is that it allows the numerical flow solver itself

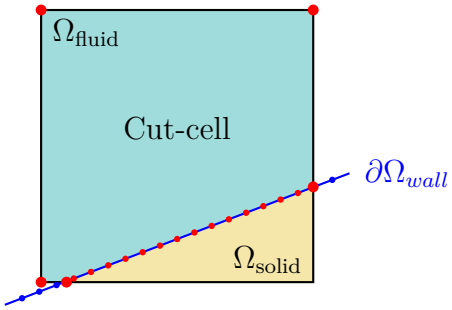


Figure 2.6 Illustration of a 2D cut-cell

to remain unchanged, while enforcing boundary conditions in a sharply manner comparable to body-fitted meshes. This sharp representation ensures both local and global conservation properties. However, the approach also presents notable challenges. In particular, the presence of very small cut cells adjacent to regular cells can lead to large variations in cell volumes. Such disparities can severely restrict the allowable pseudo-time step for a given CFL number and introduce stiffness into the iterative solution procedure. Several strategies have been proposed to alleviate these issues, including merging small cut cells with neighboring larger cells [10, 90] or linking them with a larger master cell [91].

2.4.2 Continuous Forcing Method

The continuous forcing method enforces boundary conditions directly within the continuous form of the governing equations. This approach was originally introduced by Peskin [92] to simulate blood flow inside the elastic boundaries of the heart. While effective for deformable boundaries, the method faced stability challenges when applied to rigid surfaces, such as those encountered in aerodynamics. To address these limitations, penalization techniques were developed. Arquis and Caltagirone [93] first applied penalization to numerically solve Darcy's equation for flow through porous media. Later, Angot et al. [94] adapted this concept to enforce no-slip wall boundary conditions for the Navier–Stokes equations by modeling solid walls as very low-permeability porous media.

For simplicity, the incompressible Navier–Stokes equations are considered:

$$\begin{aligned} \nabla \cdot \mathbf{u} &= 0 \\ \frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) + \frac{1}{\rho}\nabla p - \frac{1}{\rho}\nabla \cdot \boldsymbol{\tau} &= \boldsymbol{f}, \end{aligned} \tag{2.30}$$

where $\boldsymbol{f}$ represents the forcing (penalization) term:

$$\boldsymbol{f} = \lambda\chi\left(\boldsymbol{u_{bc}} - \boldsymbol{u}\right), \tag{2.31}$$

with λ as the penalization parameter, $\boldsymbol{u}_{bc}$ as the target velocity at the immersed boundary and χ a mask function defined as

$$\chi = \begin{cases} 0 & \text{if cell} \in \Omega_{fluid} \\ 1 & \text{if cell} \in \Omega_{solid} \end{cases} \tag{2.32}$$

The parameter λ is typically chosen large enough to dominate other terms in the governing equations, thereby enforcing the immersed boundary condition. However, excessively large values of λ can induce numerical instability and stiffness in the iterative solvers. Successful applications of this method include two-dimensional Euler flows [35] and Eulerian droplets with immersed boundaries [36], forming part of a broader framework for icing simulations.

The primary advantage of the continuous forcing method is its practical integration into existing solvers, which is generally simpler than for other immersed boundary approaches. Its main limitation, however, is that the sharp boundary is smeared over the local mesh size, leading to a classical first-order accuracy. Higher-order and more complex implementations exist but are more challenging to implement [95].

2.4.3 Discrete Forcing Method

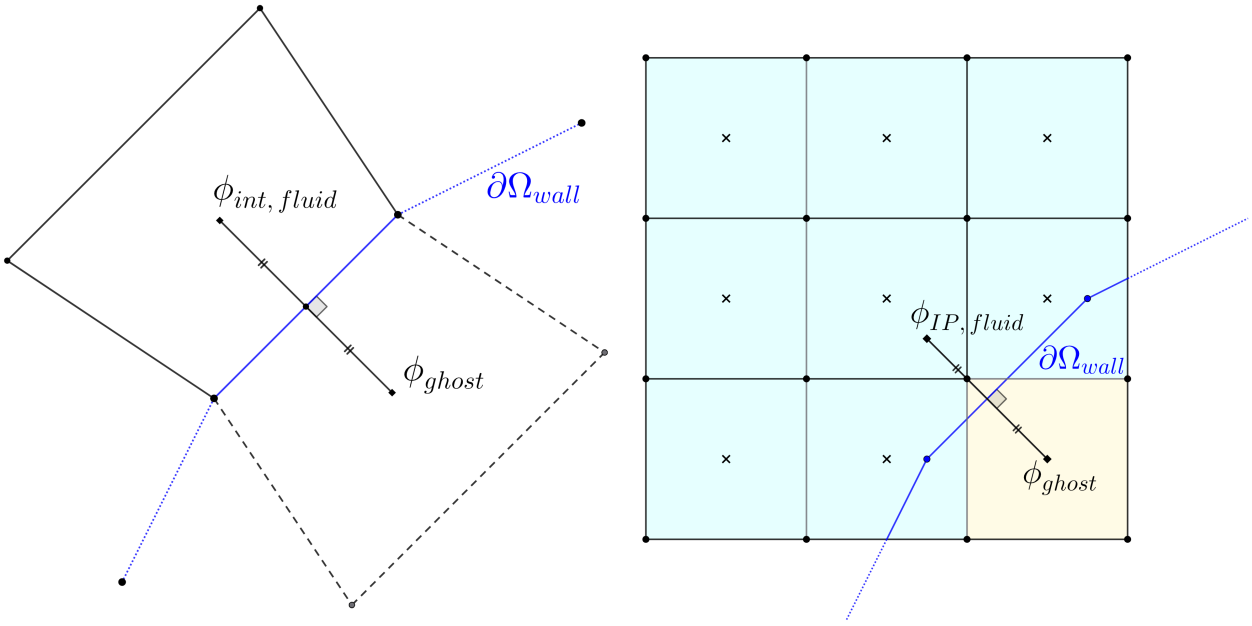
The discrete forcing method enforces immersed boundary conditions directly at the discrete level by modifying the local discrete equations arising from the spatial discretization. Two main variants of this methodology exist: cell-forcing and face-forcing.

In the cell-forcing approach, commonly referred to as the Ghost-Cell Immersed Boundary Method (GCIBM), mesh cells are first classified as either fluid or solid. Solid cells that are required by the discretization stencil of neighboring fluid cells are identified as ghost cells. In practice, this typically corresponds to the first one or two layers of solid cells adjacent to the fluid cells. The flow variables assigned to these ghost cells are then constructed to locally enforce the immersed boundary conditions, in a manner analogous to ghost cell used in body-fitted meshes.

A key distinction from body-fitted implementations lies in the location of the Image Point (IP). For body-fitted grids, the IP coincides with the center of an interior fluid cell. In contrast, for immersed boundary grids, the IP does not exactly match a fluid cell center.

Consequently, flow variables at the IP must be obtained through interpolation from surrounding fluid cells. The order of accuracy of this interpolation directly determines the overall accuracy of the method.

Figure 2.7 illustrates the ghost-cell concept for both boundary-fitted and immersed boundary grids. Successful second-order accurate implementations of GCIBM for three-dimensional Euler flows over arbitrary geometries are reported in [31, 96, 97]. The main advantage of the cell-forcing approach is its ability to achieve higher accuracy more readily than continuous forcing methods. However, it requires additional preprocessing steps, including ghost-cell identification, image-point construction and flow variables interpolations at the IP. Still, these requirements remain significantly less demanding than those associated with cut-cell methods. One disadvantage concerns the challenge in accurately representing sharp geometric features that may require careful placement of a sufficient number of ghost cells.



(a) Boundary conformed

(b) Immersed boundary

Figure 2.7 Ghost-cells representation for a boundary conformed and a immersed boundary grid

The face-forcing approach, originally developed by Capizzano, enforces immersed boundary conditions directly at mesh faces intersected by the embedded geometry. In this method, fluxes across intersected faces are modified to impose the desired boundary conditions, as illustrated in Fig. 2.8. A successful application of this technique to three-dimensional RANS

simulations over arbitrary geometries is presented in [5].

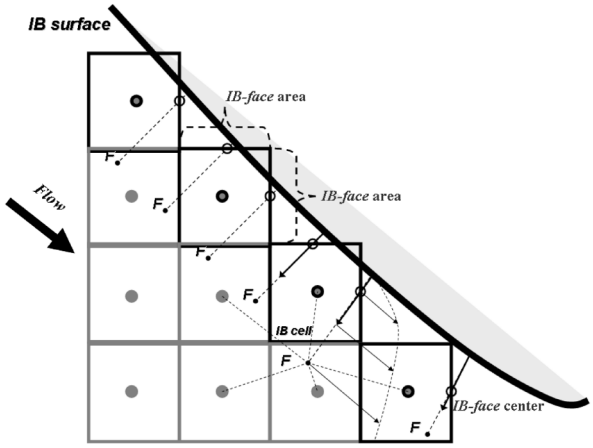


Figure 2.8 Illustration of the face-forcing approach - [5]

An important advantage of face-forcing over cell-forcing is that it does not require the introduction of ghost cells, thereby avoiding difficulties related to their placement in complex geometrical configurations. However, intersected faces require special treatment during flux evaluation, which introduces additional complexity into the numerical flux formulation.

2.5 Boundary Layer

In 1904, Prandtl introduced the boundary-layer concept [98], establishing a fundamental framework for the analysis of viscous flows at high Reynolds numbers. He demonstrated that, for such flows over solid bodies, viscous effects are confined to a thin region adjacent to the wall, known as the boundary layer, while the outer flow may be reasonably approximated as inviscid.

For viscous flows, the no-slip condition imposes a zero velocity at the wall. Consequently, the flow velocity increases in the direction normal to the surface until it asymptotically approaches the external inviscid velocity, denoted by u_e. Mathematically, this asymptotic value is reached only at an infinite distance from the wall. In practice, the boundary-layer thickness δ is commonly defined as the normal distance from the wall at which the velocity reaches 99% of u_e. A schematic illustration of a boundary layer developing over a flat plate is shown in Fig. 2.9.

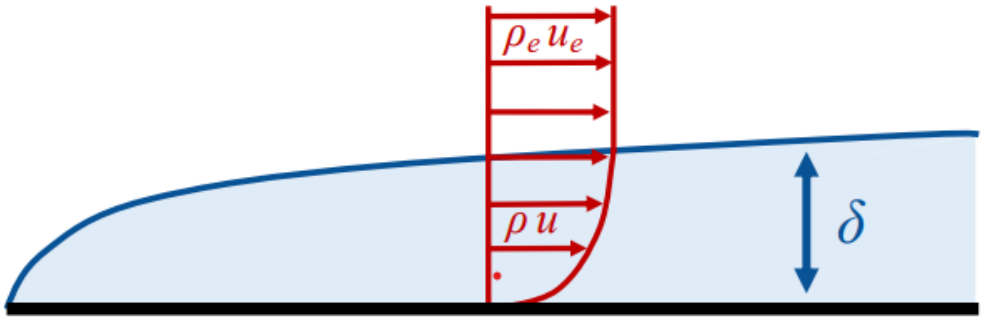


Figure 2.9 Boundary layer of a flat plate - From Clément Gorand

Because the boundary layer is thin compared to the characteristic length scale of the body, the incompressible Navier–Stokes equations can be nondimensionalized to perform an analysis of orders of magnitude. This analysis reveals that several terms are negligible within the boundary layer, leading to a significant simplification of the governing equations. Under these assumptions, the two-dimensional incompressible Navier–Stokes equations reduce to the Prandtl boundary-layer equations:

$$\begin{aligned} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0, \\ \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} &= -\frac{1}{\rho}\frac{\partial P}{\partial x} + \frac{1}{\rho}\frac{\partial \tau_{xy}}{\partial y}, \\ \frac{\partial P}{\partial y} &= 0, \end{aligned} \tag{2.33}$$

where x and y denote the tangential and normal directions relative to the wall, respectively. These equations may be solved numerically using methods such as finite differences or finite volumes [19]. However, for many aerodynamic applications, it is more practical to employ the von Kármán integral momentum equation, which is obtained by integrating the Prandtl equations (2.33) in the wall-normal direction from 0 to $+\infty$:

$$\frac{d\theta}{dx} + \theta(H+2)\frac{1}{u_e}\frac{du_e}{dx} = \frac{c_f}{2}, \tag{2.34}$$

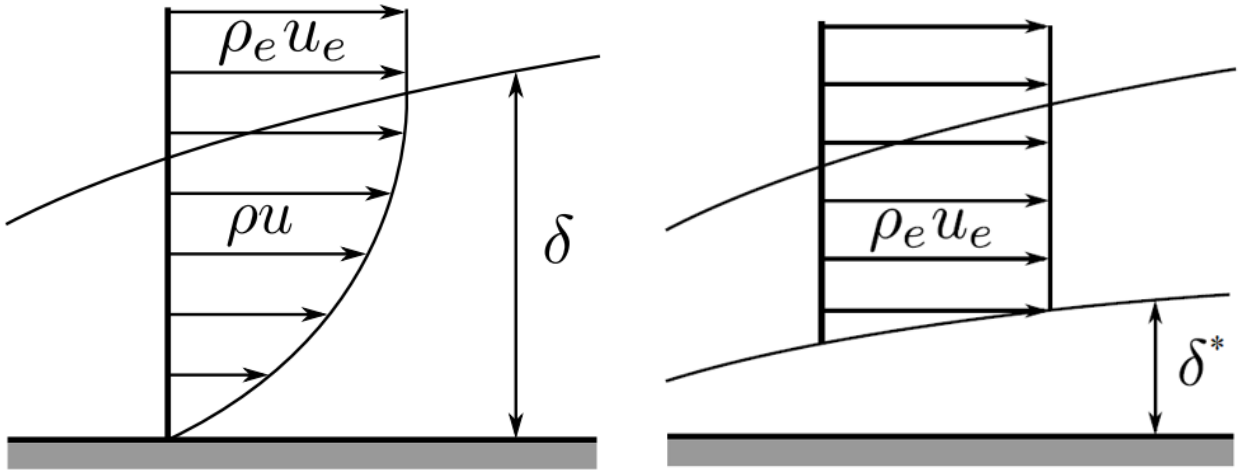
where θ is the momentum thickness, $H = \delta^*/\theta$ is the shape factor, c_f is the skin-friction coefficient and δ^* is the displacement thickness. The displacement thickness represents the fictitious thickness by which the wall would need to be displaced so that a uniform flow at velocity u_e would yield the same mass flux as the actual viscous flow. An illustration of this concept is provided in Fig. 2.10.

The displacement and momentum thicknesses are defined as

$$\delta^* = \int_0^{+\infty} \left(1 - \frac{u}{u_e}\right) dy, \tag{2.35}$$

$$\theta = \int_0^{+\infty} \frac{u}{u_e} \left(1 - \frac{u}{u_e}\right) dy. \tag{2.36}$$

.



(a) Real flow

(b) Fictitious equivalent flow

Figure 2.10 Displacement thickness δ^* representation - [6]

Equation (2.34) requires closure, as it contains three unknowns, θ, H and c_f. Two additional relations are therefore needed. These closure relations are typically derived from assumed velocity profiles and depend on whether the boundary layer is laminar or turbulent, allowing for an accurate approximation of the true boundary layer behavior.

Greater accuracy is achieved by accounting for the influence of the boundary layer on the inviscid flow through the displacement thickness δ^*. This can be accomplished either by modifying the geometry seen by the inviscid flow solver or by introducing an equivalent transpiration velocity in the slip-wall boundary condition. The inviscid and boundary-layer solvers are then sequentially iterated until convergence, a strategy known as direct coupling. While relatively simple to implement, this method is unable to handle boundary-layer separation, which limits its applicability.

An alternative strategy is the inverse method [99], in which the roles of the solvers are

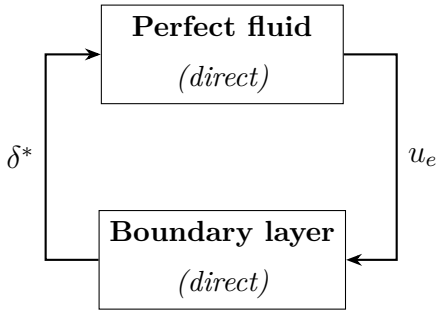
reversed: the boundary-layer solver takes δ^* as input and computes u_e, while the inviscid solver uses u_e to update δ^*. Although this approach can converge separated flows, it is unstable for attached flows and is therefore rarely used alone.

To overcome these limitations, the semi-inverse method combines direct and inverse approaches, using correction techniques to ensure convergence for both attached and separated flows [100].

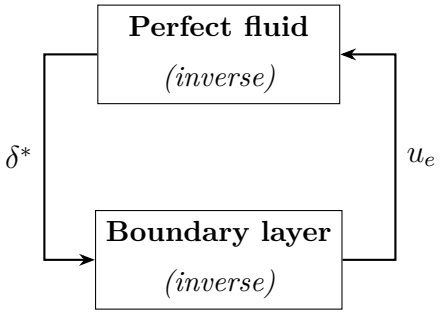
The most robust strategy is simultaneous coupling, in which the inviscid flow and boundary-layer equations are solved together as a single system [26]. This method offers superior stability and accuracy, but is also the most complex to implement, as it fundamentally alters the structure of the original inviscid flow solver.

Finally, quasi-simultaneous coupling attempts to approximate the benefits of simultaneous coupling while reducing implementation complexity. This is achieved by introducing an interaction law that models inviscid–viscous interaction within the boundary-layer solver [101]. Although this approach improves stability and convergence, selecting an appropriate interaction law can be challenging.

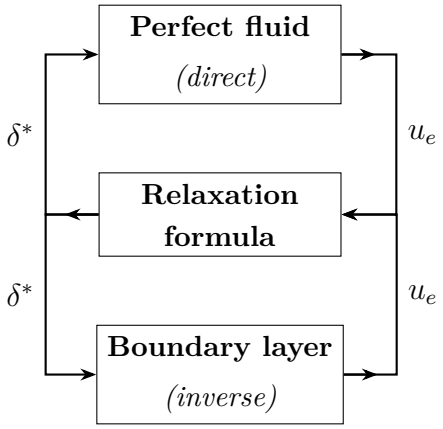
All inviscid–boundary-layer coupling strategies discussed above are summarized schematically in Fig. 2.11.



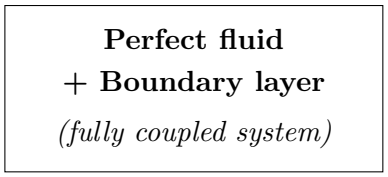
(a) Direct method



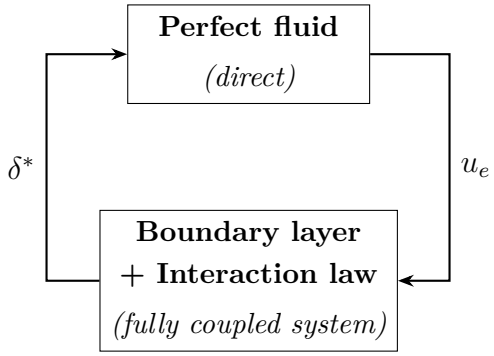
(b) Inverse method



(c) Semi-inverse method



(d) Simultaneous coupling



(e) Quasi-simultaneous method

Figure 2.11 Inviscid flow-boundary layer coupling strategies - Adapted from [6]

2.6 Aero-Icing Numerical Modeling

As introduced in Subsection 1.2.3, numerical simulations of ice accretion are commonly formulated within a coupling framework composed of five distinct modules (see Fig. 1.5). Each module addresses a specific physical aspect of the icing process.

The first component is the mesh generation module. A variety of strategies may be employed, including structured or unstructured grids, body-fitted meshes and immersed-boundary formulations, as discussed in Section 1.2.2.

The second component is the aerodynamic module, responsible for computing the airflow field. This module may rely on viscous flow solvers such as DNS, LES and RANS or on inviscid solvers, such as Euler, full potential and linearized potential formulations. In the latter case, viscous effects needs to be accounted through coupling with a boundary-layer solver.

The third component is the droplet module, which models the transport and impingement of supercooled water droplets onto the surface. Two main approaches are commonly used: Lagrangian and Eulerian formulations. In the Lagrangian approach, individual droplet trajectories are tracked by integrating Newton's equations of motion along their paths. Droplets may either impact the surface or bypass it entirely. Because this method does not require a volumetric mesh, it is particularly well suited for coupling with linearized potential flow solvers. However, its accuracy depends on tracking a sufficiently large number of droplets, which becomes computationally expensive for large geometries. In addition, this approach is difficult to parallelize efficiently [102].

In contrast, the Eulerian approach [103] treats the droplet phase as continuous, assuming a sufficiently high droplet density. This enables the droplet field to be governed by a set of partial differential equations discretized using standard techniques such as finite differences, finite volumes or finite elements. The droplet equations are typically solved on the same mesh as the aerodynamic solver, allowing straightforward parallelization. A limitation of the Eulerian approach is its inability to capture discrete phenomena such as droplet rebound or splashing, which can be represented in Lagrangian formulations.

The fourth component is the thermodynamic module, which determines the ice accretion rate based on mass and energy balances at the surface. The most widely used formulation is the classical Messinger model [104], along with several extensions and variants. In the original Messinger model, which is algebraic, the water film is assumed to originate at the stagnation point and split into upper and lower surface runback paths. A local mass and heat balance is then performed sequentially along the surface, with unfrozen water transported downstream

until the trailing edge is reached. A key limitation of this approach is its reliance on the stagnation point location to determine the runback direction, which becomes ambiguous for three-dimensional configurations.

To address this limitation, the Iterative Messinger model [105] was developed. In this formulation, the Messinger model is solved iteratively, allowing the runback direction to be determined automatically based on wall shear stress and pressure gradients.

Another extension is the Extended Messinger model [106], which is formulated as a Stefan problem and expressed as an ordinary differential equation. This model accounts for heat conduction within the ice layer, requiring the wall temperature to be prescribed. As a result, it is particularly suited for de-icing or anti-icing simulations but less applicable to purely passive aero-icing cases. Similar to the classical Messinger model, it originally relied on stagnation-point-based runback assumptions, but this limitation was removed through iterative enhancements in [107].

An alternative thermodynamic formulation is the Shallow Water Icing Model (SWIM) introduced in [108]. Based on lubrication theory, SWIM employs an unsteady PDE formulation to describe the evolution of the water film. Like the Iterative Messinger model, it accounts for wall shear stress and pressure gradients to predict runback direction. A comparative summary of the characteristics of the main thermodynamic models is presented in Fig. 2.12.

	Messinger	**Iterative Messinger**	**Extended Messinger**	**SWIM**
Conduction Through Ice	no	no	yes	no
Water Accumulation	no	yes	no	yes
Runback Water Direction	stagnation point	shear stress	stagnation point	shear stress
Formulation	Algebraic	Algebraic	ODE	PDE
State	steady	steady	unsteady	unsteady

Figure 2.12 Characteristics of the different thermodynamics modules - [7]

The final component of the framework is the geometric evolution module, which updates the surface geometry based on the local ice thickness predicted by the thermodynamic solver. The simplest and most widely used approach is the Lagrangian geometry method. In this method, ice thickness is added at each surface node along the bisector direction of the adjacent face normals. Although straightforward to implement, this technique does not conserve mass, may generate invalid geometries in concave regions and is not well suited to three-dimensional applications [102]. Relaxation techniques are often employed to alleviate these issues, but may introduce oscillations in the resulting ice shape.

More robust alternatives include the level-set method, which represents the geometry implicitly and can be modified to enforce mass conservation, at the expense of increased computational cost [102].

Another approach is the hyperbolic geometric evolution method, derived from structured mesh generation techniques and based on the solution of a hyperbolic partial differential equation. This method offers a favorable balance between robustness, mass conservation and computational efficiency [102].

CHAPTER 3 DEVELOPMENT OF A FULL POTENTIAL SOLVER FOR COMPLEX GEOMETRIES USING AN IMMERSED BOUNDARY METHOD AND BOUNDARY-LAYER COUPLING

As discussed in the introduction, FP solvers offer a computationally efficient alternative to higher-fidelity aerodynamic methods, capturing the majority of flow physics when shocks are weak or absent. When coupled with a boundary-layer solver, FP models can also approximate viscous effects, as outlined in Section 2.5.

Historically, most advances in FP solvers occurred between the 1970s and 1990s, during which structured grids and conformal mapping were standard practice. Consequently, many of the schemes developed in this period are not directly compatible with modern unstructured grid frameworks and are not well suited for arbitrary complex geometries. More recent efforts to solve the FP equations on unstructured grids have encountered challenges, particularly in simulating transonic flows with shocks, where convergence speed often significantly declines [9, 33, 34, 37].

Another critical challenge is the enforcement of the Kutta condition, which is necessary due to the irrotational nature of the FP model. However, detailed implementation strategies are scarce in the literature. Regarding IBM, very few implementations exist for FP solvers using a finite volume approach. Notably, Lyu et al. [10] employed a cut-cell method, but the implementation of this method is complex and not straightforward.

These gaps in the literature motivated the work presented in this thesis: the development of an FP solver for unstructured grids that integrates a ghost-cell immersed boundary method (GCIBM) and coupling with a boundary-layer solver to approximate viscous effects. Contributions from this work have been presented in [30] at the AIAA SciTech 2026 conference. As a result, several sections describing the FP solver and GCIBM integration closely follow or replicate the published material.

This chapter is organized into three main sections:

- Section 3.1 focuses on the development of the FP solver for unstructured, body-fitted grids. Solutions obtained with this solver are compared to Euler results, convergence rates between FP and Euler solvers are analyzed and a numerical verification of the order of convergence with respect to mesh refinement is performed to ensure the correctness of the implementation.
- Section 3.2 describes the modifications necessary to incorporate the GCIBM into the

FP solver. Comparisons of solutions and convergence rates are presented between body-fitted FP, GCIBM FP and Euler solvers, along with a numerical verification of error convergence order.

- Section 3.3 presents the boundary-layer equations used and the coupling with both body-fitted and IBM FP solvers. Results for simple test cases are compared with existing literature and with Euler-boundary-layer coupled solutions.

3.1 Development of an Unstructured Body-Fitted Full Potential Solver

3.1.1 CHAMPS Framework

The full potential development is carried out within the CHAMPS software framework. CHAMPS is an unstructured finite volume software developed at Polytechnique Montréal for aerodynamic [109], aero-elastic and aero-icing simulations [110] with scalable performance. It features classic compressible RANS-based algorithms but implemented in the new language Chapel [111]. The solver was first developed in 2019 and reached maturity to participate in several AIAA workshops including the Ice Prediction Workshop 2 (IPW2) [112], the Drag Prediction Workshop 7 (DPW7) [113] and the High Lift Prediction Workshop 5 (HLPW5) [114]. The size and capabilities of CHAMPS steadily increased since its inception. Today, it includes a broad suite of solvers designed to accommodate varying fidelity and computational cost requirements. These range, in order of increasing fidelity, from the VLM and NLVLM, to the full potential solver (presented in this thesis), the Euler and RANS solvers and up to a LES solver currently under development. As a result, this framework simplifies comparisons between models of varying fidelity and enables integration with existing multiphysics modules for aero-icing simulations.

3.1.2 Finite Volume Formulation

By integrating over the control volume to obtain its integral form, the governing equation (2.13) is transformed using the divergence theorem:

$$\iiint_{\Omega} \nabla \cdot (\rho \nabla \phi)\, d\Omega = \oint_{\partial\Omega} (\rho \nabla \phi) \cdot \mathbf{n}\, dS = 0, \tag{3.1}$$

where Ω represents the control volume and $\partial\Omega$ represents its enclosing surface. In this solver, the control volume corresponds directly to the grid cells, as is typical in cell-centered schemes. Hence, the potential ϕ is stored at cell-centers, whereas its gradient and the density ρ are computed at the face centers.

Numerically, the surface integral is approximated by adding up the mass fluxes throughout each face of a control volume as presented in Eq. (3.2):

$$\sum_{k=1}^{\#faces} \rho_k \left(\nabla\phi_k \cdot \mathbf{n_k}\right) S_k = R_i, \tag{3.2}$$

where the flow variables ρ_k, $\nabla\phi_k = [u, v, w]_k$, the face normal vector $\mathbf{n_k}$ and the face area S_k are located at the center of each face k. This sum represents the residual of cell i, which must be iteratively reduced to zero to obtain a solution consistent with the governing equation (2.13). Such iterative solvers were described in Subsection 2.2.4. Figure 3.1 provides a schematic illustration of a control volume, highlighting the locations of the associated flow variables.

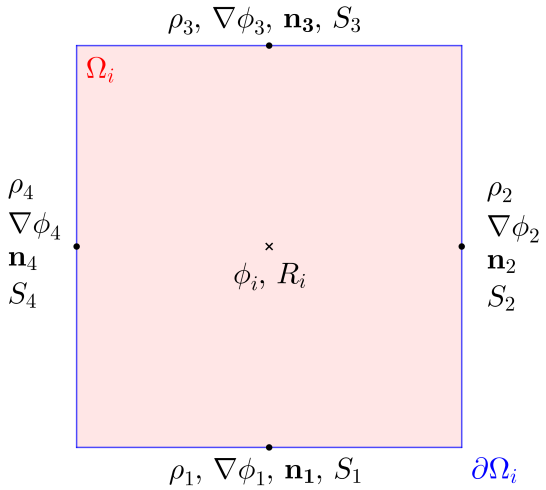


Figure 3.1 Illustration of a control volume

3.1.3 Initialization

In the finite-volume approach, prescribing an initial solution is a necessary prerequisite for starting the iterative process. In the absence of an initial guess, numerical fluxes at cell faces and the corresponding residuals within each control volume cannot be evaluated. The objective is therefore to provide an initial field that is as close as possible to the converged solution. Since in most configurations, the majority of the cells undergoes only small perturbations, a common and effective choice consists in assuming a uniform freestream flow, with the velocity equal to $\mathbf{u}_\infty$.

Because the conservative variable of the solver is the velocity potential ϕ, the initial condition

is defined as

$$\phi_{i,\ initial\ guess} = \mathbf{u}_\infty \cdot \mathbf{x_i}, \tag{3.3}$$

where $\mathbf{x}_i$ denotes the coordinates vector of the center of cell i.

When a solution corresponding to the same flow conditions (M_∞, AOA) has already been obtained on a coarser mesh, it can be interpolated onto the finer grid to provide a more accurate initial guess. This strategy, commonly referred to as mesh sequencing, can reduce the number of iterations required to achieve convergence on refined meshes.

3.1.4 Evaluation of Flow Quantities

As discussed previously, the velocity is obtained directly from the gradient of the potential, as shown in Eq. (2.12), with the undisturbed flow normalized such that $|\mathbf{u}_\infty| = 1$. The density is then computed from Eq. (2.14), with the undisturbed density naturally set to $\rho_\infty = 1$.

Once velocity and density are known, ther key flow quantities can be derived as follows:

the local speed of sound is computed as

$$a = \sqrt{\frac{\rho^{(\gamma-1)}}{M_\infty^2}}, \tag{3.4}$$

the normalized pressure is

$$p' = \frac{p}{p_\infty} = \frac{\frac{\rho^\gamma}{\gamma M_\infty^2}}{\frac{\rho_\infty^\gamma}{\gamma M_\infty^2}} = \rho^\gamma, \tag{3.5}$$

the local Mach number is

$$M = \frac{|\mathbf{u}|}{a}, \tag{3.6}$$

and the pressure coefficient is calculated as

$$C_p = \frac{p - p_\infty}{\frac{\gamma}{2} p_\infty M_\infty^2} = \frac{p' - 1}{\frac{\gamma}{2} M_\infty^2}. \tag{3.7}$$

It should be noted that these variables are not nondimensionalized in the conventional way used by standard Euler or RANS solvers. Therefore, care must be taken when coupling this

solver with other modules to ensure consistency.

3.1.5 Gradients

The FP solver requires two gradients. The first one is the potential gradient, which is directly involved in the flux calculation as shown in Eq. (3.2). To calculate it, an unconventional K-Exact Least Squares gradient method [9, 10] was chosen because of its unusual location at face centers. The second one is the density gradient. Although it does not appear explicitly in the flux expression in Eq. (3.2), it is essential to maintain numerical stability, as discussed in Subsection 3.1.7. Since this gradient is stored at cell centers, it can be computed using more conventional approaches like Weighted-Least Square or Green-Gauss [8] but can also be computed using the unconventional K-Exact Least Squares method.

K-exact Least Squares method

Both gradients can be computed using the K-exact Least Squares method. The only difference between the two implementations is the stencil used. The advantage of this method is that in can provide: i) a higher order approximation for structured and unstructured gradient; ii) a gradient value at the center of the faces, or at the center of cells.

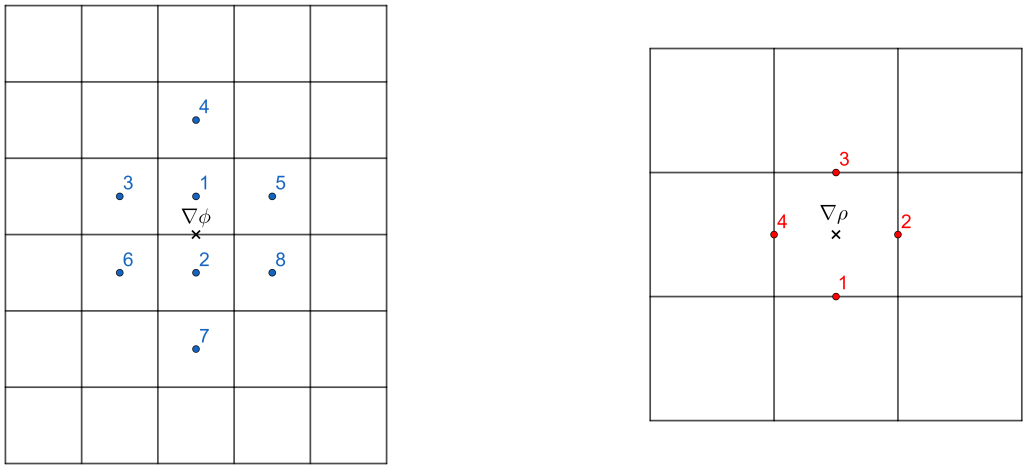
For the face-centered potential gradient $\nabla\phi$, the stencil consists of the two cells adjacent to the face of interest, combined with all their neighboring cells. Regarding the cell-centered density gradient $\nabla\rho$, it is computed using a stencil composed of all the faces surrounding the target cell. Both stencil configurations are illustrated in Fig. 3.2.

The K-exact Least Squares formulation was originally introduced for higher-order reconstructions in finite-volume Euler solvers on unstructured grids [115] and was later adapted for full potential gradient evaluation on unstructured meshes [9].

The objective of the method is to approximate the solution locally by a polynomial of degree k, constructed either about a face center (for the potential) or about a cell center (for the density). The general form of this polynomial is

$$P^k(x, y, z) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} \sum_{l=0}^{k-i-j} c_{i,\,j,\,l} x^i y^j z^l, \tag{3.8}$$

where the coefficients $c_{i,j,l}$ are unknowns. In two dimensions, the number of unknowns is $(k+1)(k+2)/2$, while in three dimensions it becomes $(k+1)(k+2)(k+3)/6$. The stencil must therefore contain at least as many points as unknown coefficients to ensure solvability.



(a) Face-centered gradient stencil ($\nabla\phi$) (b) Cell-centered gradient stencil ($\nabla\rho$)

Figure 3.2 Illustrations of the face-centered and cell-centered gradient stencils

To compute the gradients, $k = 1$ is set. This results in 3 unknowns in 2D and 4 unknowns in 3D :

$$P^1(x, y, z) = c_{0,0,0} + c_{1,0,0}x + c_{0,0,1}y + c_{0,0,1}z, \tag{3.9}$$

where the coefficients associated with x, y and z directly represent the components of the gradient.

For each element of the stencil, a linear relation is written as

$$\phi_i = c_{0,0,0} + c_{1,0,0}x + c_{0,0,1}y + c_{0,0,1}z, \tag{3.10}$$

where the subscript i denotes a stencil element. For density gradients, ϕ_i is simply replaced by ρ_i.

Since the stencil generally contains more points than unknown coefficients, the resulting system is overdetermined. A weighted least-squares approach is therefore employed to determine the coefficients that minimize the sum of the weighted squared error. The error associated with stencil element i is defined as

$$e_i = c_{0,0,0} + c_{1,0,0}x_i + c_{0,0,1}y_i + c_{0,0,1}z_i - \phi_i. \tag{3.11}$$

The objective function to be minimized is the weighted sum of squared errors,

$$F = \sum_{i=1}^{m} r_i e_i^2 = \sum_{i=1}^{m} r_i (c_{0,0,0} + c_{1,0,0} x_i + c_{0,0,1} y_i + c_{0,0,1} z_i - \phi_i)^2, \tag{3.12}$$

where m is the number of elements in the stencil. To minimize the error, F is differentiated with respect to each of the unknowns $c_{0,0,0}$, $c_{1,0,0}$, $c_{0,1,0}$ and $c_{0,0,1}$ and set to 0:

$$\begin{aligned}
\frac{\partial F}{\partial c_{0,0,0}} &= 2 \sum_{i=1}^{m} r_i (c_{0,0,0} + c_{1,0,0} x_i + c_{0,0,1} y_i + c_{0,0,1} z_i - \phi_i) = 0, \\
\frac{\partial F}{\partial c_{1,0,0}} &= 2 \sum_{i=1}^{m} r_i (c_{0,0,0} x_i + c_{1,0,0} x_i^2 + c_{0,0,1} x_i y_i + c_{0,0,1} x_i z_i - x_i \phi_i) = 0, \\
\frac{\partial F}{\partial c_{0,1,0}} &= 2 \sum_{i=1}^{m} r_i (c_{0,0,0} y_i + c_{1,0,0} x_i y_i + c_{0,0,1} y_i^2 + c_{0,0,1} y_i z_i - y_i \phi_i) = 0, \\
\frac{\partial F}{\partial c_{0,0,1}} &= 2 \sum_{i=1}^{m} r_i (c_{0,0,0} z_i + c_{1,0,0} x_i z_i + c_{0,0,1} y_i z_i + c_{0,0,1} z_i^2 - z_i \phi_i) = 0.
\end{aligned} \tag{3.13}$$

Using equations from (3.13), a linear system can be built for each face or cell where the gradient is computed as presented in Eq. (3.14).

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \sum_{i=1}^{m} r_i & \sum_{i=1}^{m} r_i x_i & \sum_{i=1}^{m} r_i y_i & \sum_{i=1}^{m} r_i z_i \\ \sum_{i=1}^{m} r_i x_i & \sum_{i=1}^{m} r_i x_i^2 & \sum_{i=1}^{m} r_i x_i y_i & \sum_{i=1}^{m} r_i x_i z_i \\ \sum_{i=1}^{m} r_i y_i & \sum_{i=1}^{m} r_i x_i y_i & \sum_{i=1}^{m} r_i y_i^2 & \sum_{i=1}^{m} r_i y_i z_i \\ \sum_{i=1}^{m} r_i z_i & \sum_{i=1}^{m} r_i x_i z_i & \sum_{i=1}^{m} r_i y_i z_i & \sum_{i=1}^{m} r_i z_i^2 \end{bmatrix} \begin{bmatrix} c_{0,0,0} \\ c_{1,0,0} \\ c_{0,1,0} \\ c_{0,0,1} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} r_i \phi_i \\ \sum_{i=1}^{m} r_i x_i \phi_i \\ \sum_{i=1}^{m} r_i y_i \phi_i \\ \sum_{i=1}^{m} r_i z_i \phi_i \end{bmatrix} = \mathbf{b} \tag{3.14}$$

In this formulation, r_i is the weighting factor associated with stencil element i, while x_i, y_i and z_i represent its relative coordinates with respect to the face or cell where the gradient is being evaluated. Since the matrix $\mathbf{A}$ only depends on the weights and on the grid metrics, its inverse $\mathbf{A^{-1}}$ can be computed only once and stored at the beginning of the simulation, increasing computational efficiency. This is not the case if the mesh changes during the simulation.

To be even more numerically efficient, Eq. (3.14) can be transformed to reduce the number of operations.

$$\mathbf{A^{-1}} = \begin{bmatrix} a'_{11} & a'_{12} & a'_{13} & a'_{14} \\ a'_{21} & a'_{22} & a'_{23} & a'_{24} \\ a'_{31} & a'_{32} & a'_{33} & a'_{34} \\ a'_{41} & a'_{42} & a'_{43} & a'_{44} \end{bmatrix} \tag{3.15}$$

$$c_{1,0,0} = a'_{21} \sum_{i=1}^{m} r_i \phi_i + a'_{22} \sum_{i=1}^{m} r_i \phi_i x_i + a'_{23} \sum_{i=1}^{m} r_i \phi_i y_i + a'_{24} \sum_{i=1}^{m} r_i \phi_i z_i \tag{3.16}$$

$$c_{0,1,0} = a'_{31} \sum_{i=1}^{m} r_i \phi_i + a'_{32} \sum_{i=1}^{m} r_i \phi_i x_i + a'_{33} \sum_{i=1}^{m} r_i \phi_i y_i + a'_{34} \sum_{i=1}^{m} r_i \phi_i z_i \tag{3.17}$$

$$c_{0,0,1} = a'_{41} \sum_{i=1}^{m} r_i \phi_i + a'_{42} \sum_{i=1}^{m} r_i \phi_i x_i + a'_{43} \sum_{i=1}^{m} r_i \phi_i y_i + a'_{44} \sum_{i=1}^{m} r_i \phi_i z_i \tag{3.18}$$

For each cell in the stencil, $c_{x,i}$, $c_{y,i}$ and $c_{z,i}$ can be stored at the beginning of the simulation since they are only functions of the mesh metrics and of the gradient stencil weights.

$$c_{x,i} = r_i(a'_{21} + a'_{22} x_i + a'_{23} y_i + a'_{24} z_i) \tag{3.19}$$

$$c_{y,i} = r_i(a'_{31} + a'_{32} x_i + a'_{33} y_i + a'_{34} z_i) \tag{3.20}$$

$$c_{z,i} = r_i(a'_{41} + a'_{42} x_i + a'_{43} y_i + a'_{44} z_i) \tag{3.21}$$

To compute $c_{1,0,0}$, $c_{0,1,0}$ and $c_{0,0,1}$, a much simpler approach with fewer steps is obtained:

$$c_{1,0,0} = \sum_{i=1}^{m} c_{x,i} \phi_i \tag{3.22}$$

$$c_{0,1,0} = \sum_{i=1}^{m} c_{y,i} \phi_i \tag{3.23}$$

$$c_{0,0,1} = \sum_{i=1}^{m} c_{z,i} \phi_i \tag{3.24}$$

Then, the gradient can be approximated with $\phi_x = c_{1,0,0}$, $\phi_y = c_{0,1,0}$ and $\phi_z = c_{0,0,1}$. Using this method, a second order gradient approximation is obtained.

For the face-centered gradient, the weighting factor r_i is set up around 1×10^4 for the two

cells adjacent and set up to 1 for all the other cells in the stencil as in Ref. [9]. For the cell-centered gradient, the weighting factors r_i are not necessary and can be set up to 1 for all the faces in the stencil.

3.1.6 Kutta Condition

Unlike Euler solvers, a Kutta condition must be imposed to simulate lifting flows for full potential solvers. This requires a special treatment. The computational domain is cut from the Trailing Edge (TE) of the airfoil to the farfield boundary creating a line (2D) or surface (3D), called the wake-cut. This cut allows the potential to be discontinuous through it while maintaining all other flow properties continuous.

The first step to apply the Kutta condition is to identify the discrete wake faces of the mesh that follow the wake-cut. In [9], a method is proposed to identify these faces. The procedure is the following: among all the faces aft of the trailing edge, identify those that have cells with both a top and a bottom neighbor relative to the wake reference line.

The second step is to identify the upper and lower body faces that are connected to the trailing-edge nodes. These faces are useful to compute the potential jumps. In 2D, the value of the potential jump is equal to the circulation of the flow Γ. A simple way to compute it is by interpolating the value from the last face before the trailing edge on the upper and lower surfaces ($\phi_{upper,TE}$ and $\phi_{lower,TE}$ respectively) and by computing the difference between the two:

$$\phi_{upper,TE} = \phi_{upper\,face,TE} + \nabla\phi_{upper\,face,TE} \cdot \boldsymbol{\Delta s_{upper,TE}}, \tag{3.25}$$

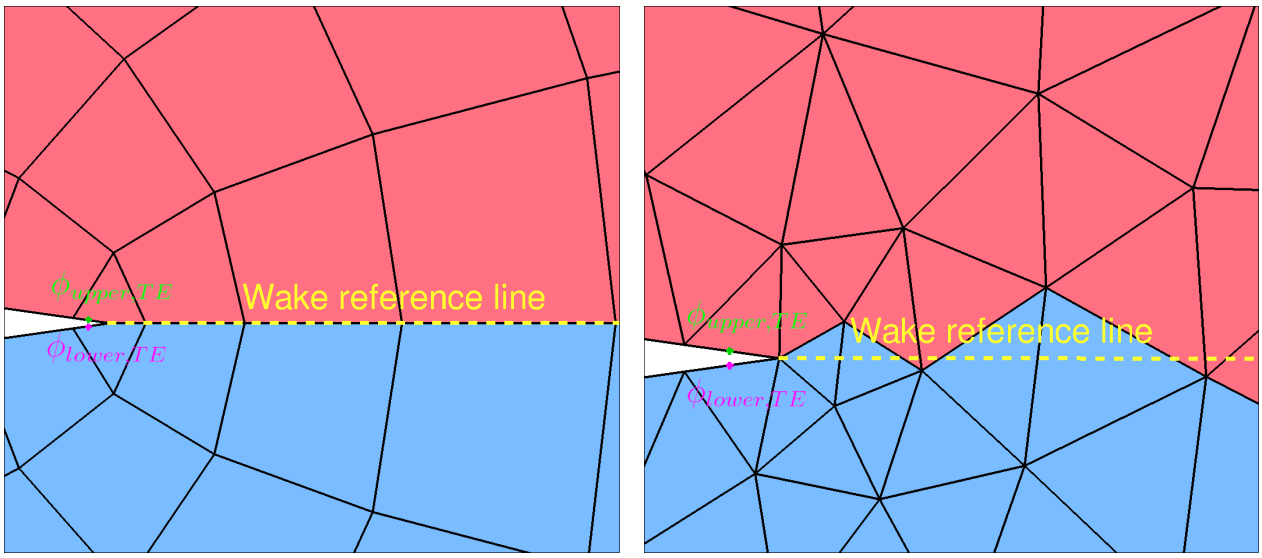
$$\phi_{lower,TE} = \phi_{lower\,face,TE} + \nabla\phi_{lower\,face,TE} \cdot \boldsymbol{\Delta s_{lower,TE}}, \tag{3.26}$$

$$\Gamma = \phi_{upper,TE} - \phi_{lower,TE}, \tag{3.27}$$

where $\phi_{upper\,face,TE}$ is the potential of the upper TE face, $\nabla\phi_{upper\,face,TE}$, the potential gradient of the upper TE face and $\boldsymbol{\Delta s_{upper,TE}}$, the coordinate vector from the center of the upper TE face to the TE node. $\phi_{lower\,face,TE}$, $\nabla\phi_{lower\,face,TE}$ and $\boldsymbol{\Delta s_{lower,TE}}$ are defined similarly for the lower TE face.

Examples of the discretization of the wake facets for a wake conforming mesh and for an embedded wake mesh are illustrated in Fig. 3.3. The cells above and below the wake facets

are respectively red and blue.



(a) Wake conforming (b) Embedded wake

Figure 3.3 Wake conforming and embedded wake meshes

In this work, the wake reference line was assumed to be straight and to follow $y = 0$ to simplify prepossessing efforts. Tests were executed to see the effect of the wake reference line shape (i.e. line following the freestream direction or line following the streamline originating from the trailing edge) but the impact seemed negligible for a two-dimensional single airfoil flow. However, it is expected that for flows with multiple lifting surfaces (multi-element airfoil or aircraft with wing and horizontal tail) the wake reference line/surface becomes important and will affect if the wake cut of the upstream surface goes below, on or above the downstream lifting surface. Therefore, further research is required on this particular aspect.

Once the potential jump is obtained, it is involved by modifying the potential gradient computation. Indeed, if a face is located below the wake faces, any cells in its $\nabla\phi$ stencil located above the wake-cut will have their potential value temporarily increased by the potential jump during the gradient calculation. For other cells below the wake-cut, the potential will remain unchanged. This would be the opposite for a face located above the wake faces. Any cells in its $\nabla\phi$ stencil located below the wake-cut will have their potential temporarily decreased by the potential jump value during the gradient calculation. For other cells above the wake-cut, the potential will remain unchanged. If a face is a wake face, any cells in its $\nabla\phi$ stencil located above the wake-cut will have their potential temporarily increased by half of the potential jump value. Any cells in its $\nabla\phi$ stencil located below

the wake-cut will have their potential temporarily decreased by half of the potential jump value during the gradient calculation, ensuring smooth gradients. Thus, the gradients Eqs. (3.22-3.24) become:

$$c_{1,0,0} = \sum_{i=1}^{m} c_{x,i} \left(\phi_i + b_i \, \Gamma\right), \tag{3.28}$$

$$c_{0,1,0} = \sum_{i=1}^{m} c_{y,i} \left(\phi_i + b_i \, \Gamma\right), \tag{3.29}$$

$$c_{0,0,1} = \sum_{i=1}^{m} c_{z,i} \left(\phi_i + b_i \, \Gamma\right), \tag{3.30}$$

where b_i is equal to -1, -0.5, 0, 0.5 or 1 depending on the face position and the stencil cell position relatively to the wake cut. All the cases of face and stencil cell position with resulting b_i value are presented in Table 3.1.

Table 3.1 b_i values as a function of face and stencil cell position relative to the wake cut

	Stencil cell above	**Stencil cell on**	**Stencil cell below**
Face above	0	-0.5	-1
Face on	0.5	0	-0.5
Face below	1	0.5	0

Blunt Trailing Edge

For simulating lifting flow over a blunt trailing edge geometry, a special treatment must be applied. To ensure smooth potential gradients, the slope of the upper TE surface and lower TE surface are extended until they intersect. After the intersection, the wake cut is determined as was the case for sharp TE. For the part of the mesh between the TE and the intersection, the faces and stencil cells are identified as above the wake cut if it is above the upper TE slope, on the wake cut if it is located between the upper and lower TE slope or below the wake cut if it is located below the lower TE slope. For the rest of the procedure, it is the same as sharp TE, only need to follow the Eqs. (3.28-3.30) and Table 3.1. An illustration of the Kutta condition for a blunt TE is provided in Fig. 3.4, where the upper cells, upon wake cells and below cells are colored respectively in red, dark grey and light blue. This procedure seems to be stable, but overshoots the lift coefficient giving similar results of equivalent sharp geometry. This is probably due to the upper and lower surface extension

that does not follow the physics accurately. A better way of doing this would be to follow a streamline at the upper and lower TE and see where they intersect. However, this would increase significantly the complexity of the procedure.

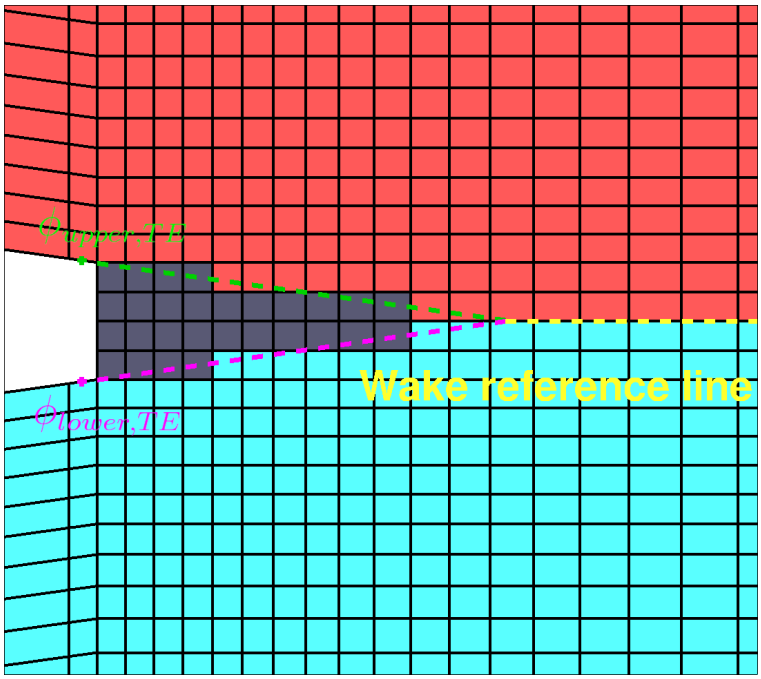


Figure 3.4 Illustration of the Kutta condition for a blunt trailing edge

Three-Dimensional Kutta Condition

In 3D, the potential jump varies because the value of Γ is not constant along the wing span. In this case, the circulation Γ_j is calculated at each trailing edge node j. If a TE node is shared by one or more upper and lower TE faces, the average value is used in cases where multiple ones exist. Then, the potential jump for a cell is determined through piecewise linear interpolation of the Γ_j values corresponding to the two trailing edge nodes that encompass the cell's centroid, using the wing span direction as the interpolation axis. An illustration of this procedure is provided in Fig. 3.5.

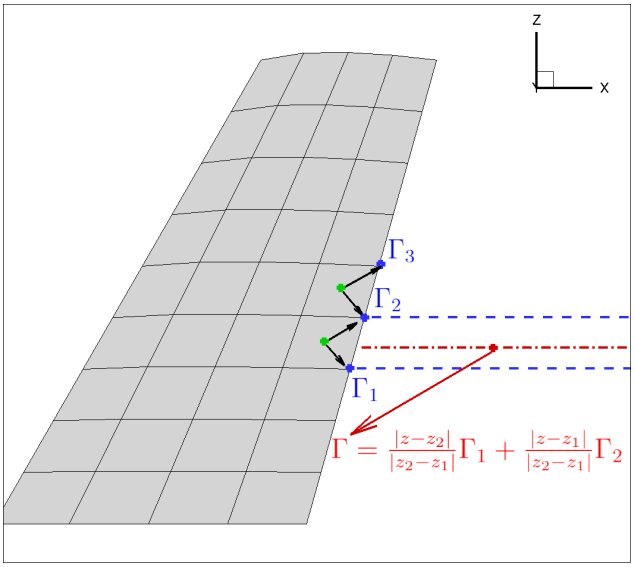


Figure 3.5 Illustration of the Kutta condition in 3D for a wing

3.1.7 Density Upwinding

In this work, the approach chosen is similar to Eq. (2.22) expressed in the literature review, but the distance components are brought outside the parentheses, so the density is really upwinded in the flow direction and not dependant on the different components of $\mathbf{\Delta x}$:

$$\tilde{\rho} = \rho - \frac{\mu \|\mathbf{\Delta x}\|}{\|\mathbf{u}\|} \left(u\rho_x + v\rho_y + w\rho_z\right), \tag{3.31}$$

where ρ, u, v and w are respectively the face's density and the velocity components in the x, y and z direction, μ is a switching function that activates upwinding when $M > M_c$, ρ_x, ρ_y and ρ_z are the density gradient components of the upwind cell computed and $\|\mathbf{\Delta x}\|$ is twice the distance from the center of the face where the flux is computed to the upwind cell center. An illustration where each of these parameters is identified is provided in Fig. 3.6.

There are also two different switching functions across the literature (see Eqs. (2.24, 2.25)). In this work, both switches were tested without noticing difference between both, but the second switch was chosen since it clamps the value of μ between 0 and μ_c.

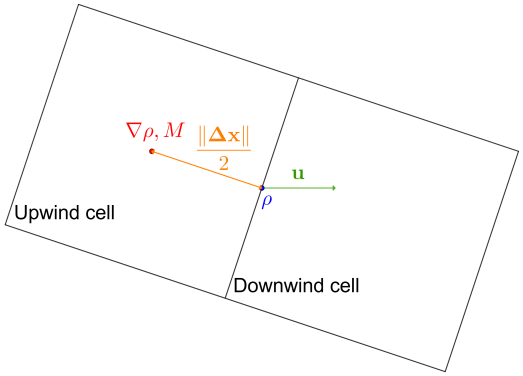


Figure 3.6 Illustration of the density upwinding procedure

3.1.8 Boundary Conditions

Three types of boundary conditions are implemented to solve the governing partial differential equation: slip-wall, symmetry and farfield conditions. The slip-wall and symmetry boundary conditions are treated identically in the present implementation.

An illustration of the face-centered gradient stencil used at a boundary face is provided in Fig. 3.7.

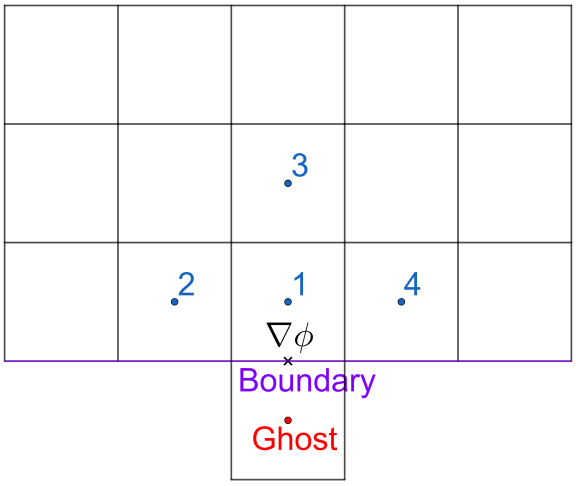


Figure 3.7 Boundary face-centered gradient stencil ($\nabla\phi$)

Slip Wall and Symmetry

The slip wall condition is applied on the surface of the body. This condition is essentially a non-penetrating condition:

$$\mathbf{u_{body}} \cdot \mathbf{n} = \nabla\phi_{body} \cdot \mathbf{n} = 0, \tag{3.32}$$

where $\mathbf{n}$ is the normal unit vector of the body surface. To enforce Eq. (3.32), ghost cells are introduced outside the computational domain. These ghost cells are constructed by reflecting interior cells across the boundary face, which therefore acts as a symmetry plane. A value of the potential ϕ_{ghost} must then be determined so that the discrete gradient approximation satisfies the non-penetration condition.

Using Eqs. (3.22–3.24) together with Eq. (3.32), the following constraint is obtained:

$$\sum_{i=1}^{m} c_{x,i}\phi_i n_x + \sum_{i=1}^{m} c_{y,i}\phi_i n_y + \sum_{i=1}^{m} c_{z,i}\phi_i n_z = 0, \tag{3.33}$$

from which the ghost-cell potential can be expressed as

$$\phi_{ghost} = -\frac{\sum_{i=1}^{m-1} c_{x,i}\phi_i n_x + \sum_{i=1}^{m-1} c_{y,i}\phi_i n_y + \sum_{i=1}^{m-1} c_{z,i}\phi_i n_z}{c_{x,ghost} n_x + c_{y,ghost} n_y + c_{z,ghost} n_z}. \tag{3.34}$$

Here, the index i ranging from 1 to $m-1$ corresponds to the cells in the stencil excluding the ghost cell.

Once ϕ_{ghost} has been evaluated, the potential gradients at the boundary faces can be computed. Since these gradients are stored at face centers, any remaining normal component can be removed explicitly:

$$\nabla\phi^*_{wall,i} = \nabla\phi_{wall,i} - (\nabla\phi_{wall,i} \cdot \mathbf{n})\,\mathbf{n}, \tag{3.35}$$

where $\nabla\phi^*_{wall,i}$ denotes the corrected gradient that satisfies the non-penetration condition, $\nabla\phi_{wall,i}$ is the initial gradient obtained using the K-Exact Least Squares method described in Subsection 3.1.5 and $\mathbf{n}$ is the unit normal vector of the boundary face.

Farfield

The farfield boundary condition is imposed on the outer surfaces of the mesh, positioned sufficiently far from the body of interest. It represents an undisturbed, freestream flow

condition.

$$\phi_{farfield} = \mathbf{u}_\infty \cdot \mathbf{x}. \tag{3.36}$$

This can be numerically enforced by directly imposing the farfield faces velocity to $\mathbf{u}_\infty$:

$$\nabla\phi^*_{farfield,i} = \mathbf{u}_\infty, \tag{3.37}$$

where $\nabla\phi^*_{farfield,i}$ is the new corrected potential gradient value.

The farfield condition is modified when the flow is lifting and the Kutta condition is enforced. A consistent potential jump is maintained by adding a vortex to the farfield potential:

$$\phi_{farfield} = \mathbf{u}_\infty \cdot \mathbf{x} - \frac{\Gamma\theta}{2\pi}, \tag{3.38}$$

where Γ represents the potential jump across the wake-cut, θ denotes the angle between the farfield face center and the wake-cut.

This condition is further enforced by prescribing the velocity at farfield faces as the gradient of the farfield potential expressed as Eq. (3.38):

$$u_{farfield,i} = u_\infty + \frac{\Gamma}{2\pi}\frac{y_i}{{x_i}^2 + {y_i}^2}, \tag{3.39}$$

$$v_{farfield,i} = v_\infty - \frac{\Gamma}{2\pi}\frac{x_i}{{x_i}^2 + {y_i}^2}. \tag{3.40}$$

Note that this condition is for a 2D lifting flow with the flow stream following the x axis and the y axis being vertical. In a 3D simulation, the z-component of the farfield velocity would simply be $w_{farfield,i} = w_\infty$.

3.1.9 Iterative Solvers

Different iterative strategies were examined using explicit and implicit discretizations for the iterative non-linear system which has the role of bringing the fluxes residuals to zero.

Explicit Solvers

Explicit solvers compute the update of a cell's conservative variable using only the local residual, without requiring residuals from neighboring cells or solving a coupled system. By

applying a relaxation method, this can be expressed as

$$\Delta\phi_i^n = \omega R_i^n, \tag{3.41}$$

where $\Delta\phi_i^n$ is the n$^{\text{th}}$ update of the potential of the cell i, ω is the relaxation factor and R_i^n is the residual of the cell i at the n$^{\text{th}}$ iteration.

The challenge with this formulation is that the optimal value of ω varies significantly across different cases and meshes, making it difficult to generalize. A more robust approach is to consider only the diagonal of the Jacobian matrix from the Newton linearization:

$$\mathbf{R^{n+1}} = \mathbf{R^n} + \left(\frac{\partial \mathbf{R}}{\partial \boldsymbol{\phi}}\right)^n \Delta\boldsymbol{\phi}^n = \mathbf{0}. \tag{3.42}$$

This leads to a linear system of equations, where $\left(\frac{\partial \mathbf{R}}{\partial \boldsymbol{\phi}}\right)^n$ is the Jacobian matrix at the n$^{\text{th}}$ iteration, $\Delta\boldsymbol{\phi}^n$ is the n$^{\text{th}}$ update for the potential array and $\mathbf{R^n}$ is the residual array at the n$^{\text{th}}$ iteration:

$$\left(\frac{\partial \mathbf{R}}{\partial \boldsymbol{\phi}}\right)^n \Delta\boldsymbol{\phi}^n = -\mathbf{R^n}. \tag{3.43}$$

By neglecting the off-diagonal elements of the Jacobian matrix, an explicit method can be derived from Eq. (3.43) and reformulated for each cell using a relaxation method:

$$\Delta\phi_i^n = -\omega \frac{R_i^n}{\left(\frac{\partial R_i}{\partial \phi_i}\right)^n}, \tag{3.44}$$

where $\left(\frac{\partial R_i}{\partial \phi_i}\right)^n$ is the (i, i) component of the Jacobian matrix. Its formulation is described in Subsection 3.1.10. Eq. (3.44) represents an improved version of Eq. (3.41). In this formulation, the optimal value of ω is less arbitrary and typically close to 1. However, for transonic flow simulations with strong shocks, ω may still need to be reduced to a value between 0 and 1 to ensure residual convergence stability.

Implicit Solvers

The concept of implicit solvers involves using the residuals of the cells that exert the most influence on the current cell i to compute its potential update. This often comes down to taking into account the residual of the cell itself and of the immediate neighboring cells. This approach involves solving the entire linear system described by Eq. (3.43) rather than

considering only the diagonal elements, as was done in the explicit solver.

The increased complexity of the implicit solvers comes from solving the large and sparse linear system of equations described by Eq. (3.43). The system is often too large to be solved directly. In other words, the problem comes back to solving the classic linear problem:

$$A\mathbf{x} = \mathbf{b} \tag{3.45}$$

where A is a large and sparse coefficient matrix since the solver works with unstructured meshes, it represents the Jacobian matrix $\left(\frac{\partial \mathbf{R}}{\partial \boldsymbol{\phi}}\right)^n$, $\mathbf{x}$ is the solution vector which represents the potential update array $\Delta\boldsymbol{\phi}^n$ and $\mathbf{b}$ is the right-hand side vector which represents the negative of the residual array $-\mathbf{R^n}$.

Consequently, an iterative method needs to be chosen to solve approximately the system. Namely, the Jacobi, the SGS and the GMRES algorithms are popular choices to do so. These methods were described in the literature review in Subsection 2.2.4. Each algorithm has its advantages. The Jacobi and SGS are simpler to implement and usually computationally less expensive per inner iteration, but they are usually less robust and efficient than the GMRES algorithm. In this work, the GMRES method has shown significant efficiency over the other methods.

3.1.10 Jacobian Matrix Formulation

The construction of the Jacobian matrix requires differentiating the residual of each control volume, R_i, defined in Eq. (3.2), with respect to the potential ϕ_j of all cells in the computational domain. Several strategies can be adopted for this purpose. In the present work, two approaches were investigated: an analytical approach and a finite-difference approach.

Analytical Method

In the analytical approach, the residual given by Eq. (3.2) is differentiated with respect to the potential ϕ_j:

$$\frac{\partial R_i}{\partial \phi_j} = \sum_{k=1}^{\#faces} \frac{\partial}{\partial \phi_j} \left(\tilde{\rho}_k \left(\nabla \phi_k \cdot \mathbf{n_k}\right) S_k\right) \approx \sum_{k=1}^{\#faces} \tilde{\rho}_k \left(\frac{\partial \nabla \phi_k}{\partial \phi_j} \cdot \mathbf{n_k}\right) S_k. \tag{3.46}$$

As commonly done in the literature [9, 10], the upwinded density $\tilde{\rho}_k$ is taken outside the derivative to simplify the Jacobian formulation, even though it depends on the potential. Since density variations remain limited in subsonic flows, this approximation is generally

acceptable in that regime. However, it becomes less accurate for transonic flows and it was observed that this assumption leads to significantly slower convergence in the presence of shocks compared to purely subsonic cases.

The term $\frac{\partial \nabla \phi_k}{\partial \phi_j}$ is obtained by developing the gradient expression of the K-Exact least squares method from Eqs. (3.28-3.30):

$$\frac{\partial \left(\phi_x\right)_k}{\partial \phi_j} = \frac{\partial c_{1,0,0}}{\partial \phi_j} = \sum_{l=1}^{m} \left(c_{x,l} \frac{\partial \phi_l}{\partial \phi_j} + b_l \frac{\partial \Gamma}{\partial \phi_j} \right) = \sum_{l=1}^{m} \left(c_{x,l} \delta_{l,j} + b_l \frac{\partial \Gamma}{\partial \phi_j} \right), \tag{3.47}$$

$$\frac{\partial \left(\phi_y\right)_k}{\partial \phi_j} = \frac{\partial c_{0,1,0}}{\partial \phi_j} = \sum_{l=1}^{m} \left(c_{y,l} \frac{\partial \phi_l}{\partial \phi_j} + b_l \frac{\partial \Gamma}{\partial \phi_j} \right) = \sum_{l=1}^{m} \left(c_{y,l} \delta_{l,j} + b_l \frac{\partial \Gamma}{\partial \phi_j} \right), \tag{3.48}$$

$$\frac{\partial \left(\phi_z\right)_k}{\partial \phi_j} = \frac{\partial c_{0,0,1}}{\partial \phi_j} = \sum_{l=1}^{m} c_{z,l} \left(c_{z,l} \frac{\partial \phi_l}{\partial \phi_j} + b_l \frac{\partial \Gamma}{\partial \phi_j} \right) = \sum_{l=1}^{m} \left(c_{z,l} \delta_{l,j} + b_l \frac{\partial \Gamma}{\partial \phi_j} \right). \tag{3.49}$$

Here, $\delta_{l,j}$ denotes the Kronecker delta, which is equal to 1 if $l = j$ and 0 otherwise. As a result, for a given face k, only the m cells belonging to its gradient stencil contribute to non-zero terms through $\partial \phi_l / \partial \phi_j$.

To limit the size and complexity of the Jacobian and to improve parallel efficiency, only the contributions associated with the cells directly adjacent to face k are retained. These entries dominate the remaining stencil contributions by a factor proportional to the weighting coefficient r_l, which is typically chosen on the order of 10^4.

The circulation jump at the wake is approximated as

$$\Gamma \approx \phi_{upper\,cell,TE} - \phi_{lower\,cell,TE}, \tag{3.50}$$

where $upper\,cell, TE$ and $lower\,cell, TE$ correspond to the interior cells connected respectively to the $upper\,face, TE$ and $lower\,face, TE$. Consequently, the derivative $\partial \Gamma / \partial \phi_j$ is equal to $+1$ when j is the upper trailing-edge cell and -1 when j is the lower trailing-edge cell. Therefore, cells located near the wake cut and associated with nonzero b_l coefficients have additional Jacobian contributions in the columns corresponding to these two trailing-edge cells.

As typically done in the literature, the $\partial \Gamma / \partial \phi_j$ term can be neglected, but doing so was found to significantly slow down convergence for lifting flows compared to non-lifting cases. For this reason, it is retained in the present formulation.

Finite Difference Method

In the finite-difference approach, the Jacobian is computed numerically by perturbing the potential of cell j by a small increment ϵ_j and evaluating the resulting change in the residuals:

$$\frac{\partial R_i}{\partial \phi_j} = \frac{R_i(\boldsymbol{\phi} + \boldsymbol{\epsilon_j}) - R_i(\boldsymbol{\phi})}{\epsilon_j}, \tag{3.51}$$

where $\boldsymbol{\phi}$ denotes the vector of cell potentials and $\boldsymbol{\epsilon}_j$ is a vector of identical size whose only non-zero entry is ϵ_j at index j.

In principle, this procedure should be applied to every (i, j) couple, but doing so would be computationally prohibitive. In practice, the index i is restricted to cell j and its first and second layers of neighbors. Additionally, when j corresponds to an upper or lower trailing-edge cell, i includes all cells affected by the wake-induced potential jump.

This approach was observed to converge in significantly fewer iterations than the analytical Jacobian for transonic flows with shocks. However, using this approach, the Jacobian is considerably more expensive to assemble and its assembling cost increases with the refinement of the mesh. To mitigate this cost in CHAMPS, a coloring strategy is employed such that cells sharing a stencil cannot have the same color. The potential is then perturbed color by color and the resulting residual variations are evaluated simultaneously. Even with this strategy, the Finite Difference is computationally expensive to assemble. Furthermore, at present, the coloring algorithm implemented in CHAMPS is limited to single zone partitioned meshes. For these reasons, the analytical Jacobian formulation is preferred for refined meshes in the current framework. Ideally, the robustness of the finite-difference Jacobian would be combined with an assembly cost comparable to the analytical approach. This should be feasible with further research on the analytical Jacobian so that it includes the upwinded density derivative.

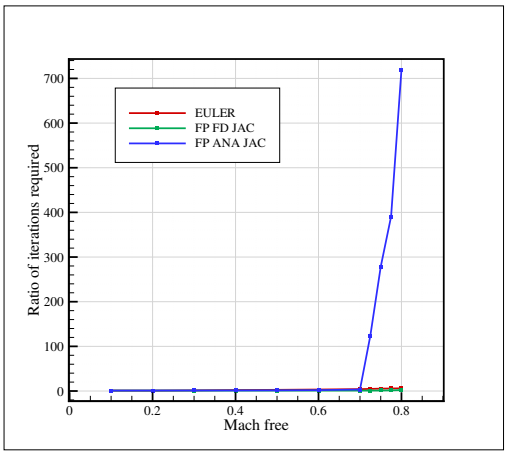
3.1.11 Results

Influence of M_∞ on Convergence

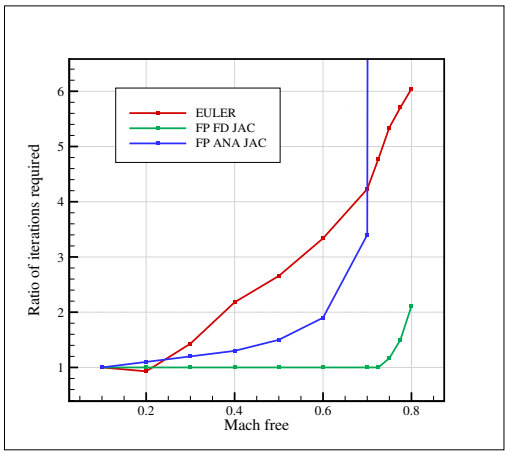
Since reduction in convergence rates was observed for transonic cases compared to subsonic ones, an analysis of this phenomenon was carried out. To investigate this behavior, a series of simulations was performed on the same NACA0012 128×128 mesh shown in Fig. 3.44b. The freestream Mach number was varied from $M_\infty = 0.1$ to $M_\infty = 0.8$, while maintaining an angle of attack of $AOA = 0.0°$.

Three different approaches were considered: the Euler solver, the full potential solver with an

analytical Jacobian and the full potential solver with a finite-difference Jacobian. The Euler solver employed the SGS linear solver, whereas both full potential solvers used GMRES. For each configuration and Mach number, the number of iterations and the computational time required to reach $\log(\text{RMS-}\rho) = -5$ were recorded. To facilitate comparison, the number of iterations and computational time were normalized by the corresponding values obtained for $M_\infty = 0.1$. The resulting trends are presented in Fig. 3.8 for the iteration ratio and in Fig. 3.9 for the time ratio.

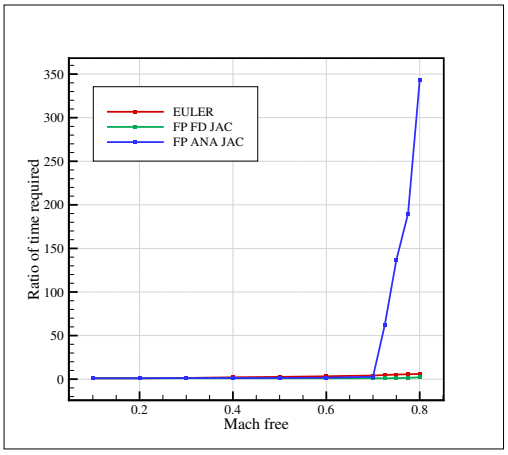


(a) Global view

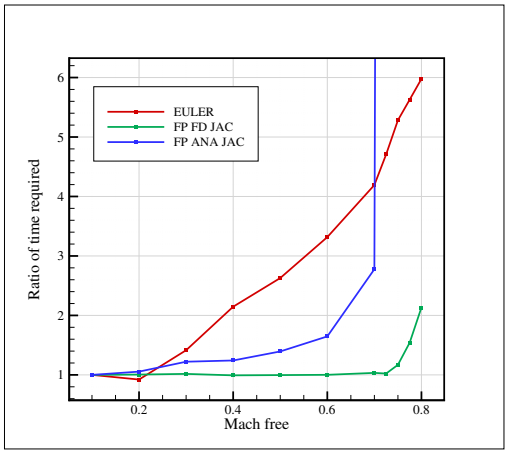


(b) Euler and FP FD Jacobian view

Figure 3.8 Ratio of the number of iterations required to reach $log(\text{RMS-}\rho) = -5$ as a function of M_∞ relatively to $M_\infty = 0.1$



(a) Global view



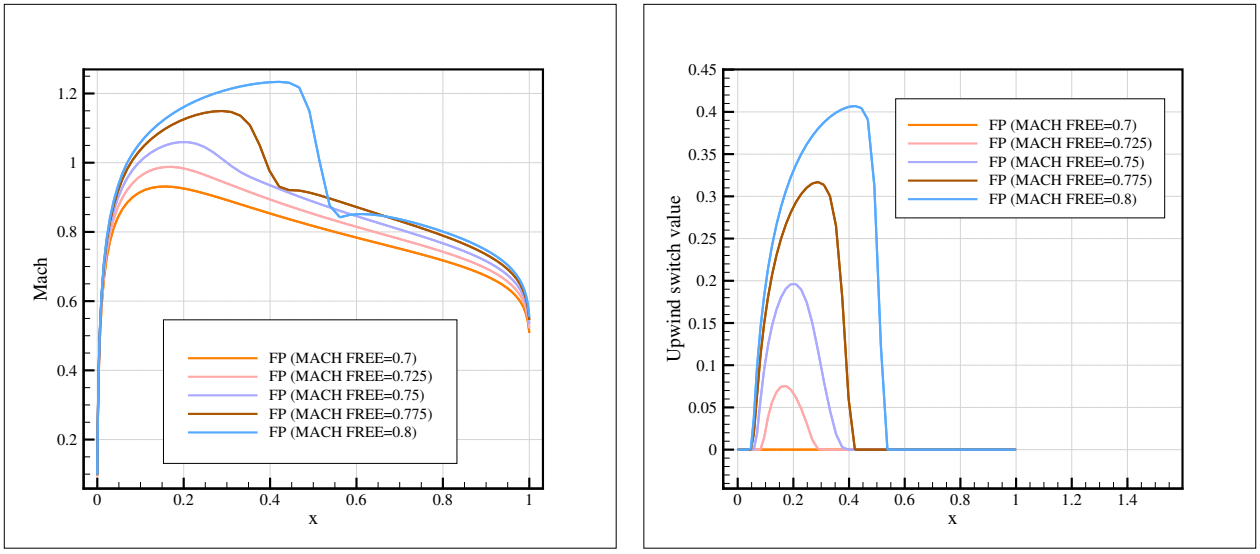
(b) Euler and FP FD Jacobian view

Figure 3.9 Ratio of the time required to reach $log(\text{RMS-}\rho) = -5$ as a function of M_∞ relatively to $M_\infty = 0.1$

As shown in these figures, the Euler solver exhibits an approximately linear increase in both iteration count and computational time as the Mach number increases.

In contrast, the full potential solver with the analytical Jacobian follows a behavior closer to an exponential trend. Both the iteration and time ratios increase rapidly once M_∞ reaches 0.725. This deterioration can be explained by examining the local Mach number and the distribution of the upwind switching parameter μ, shown in Fig. 3.10. Starting from $M_\infty = 0.725$, the local Mach number exceeds $M_c = 0.95$ and the μ switch becomes active. This modifies the fluxes and residuals computation, an effect that is not accounted for in the analytical Jacobian formulation.

The full potential solver using a finite-difference Jacobian shows a similar exponential trend, but the growth rate is significantly reduced. As a result, it remains more efficient than the Euler solver within the investigated Mach number range, unlike the analytical Jacobian approach. At higher Mach numbers, it is highly possible that the full potential finite-difference Jacobian curves would eventually surpass those of the Euler solver. However, such conditions would likely correspond to maximum local Mach numbers exceeding 1.3, beyond which the isentropic assumption introduces significant inaccuracies anyway.



(a) Local Mach number

(b) μ

Figure 3.10 Local Mach number and μ distribution as a function of x/c for all cases between $M_\infty = 0.7$ and $M_\infty = 0.8$

Influence of Implicit Circulation on Convergence

To demonstrate the importance of Implicit Circulation (IC) for the convergence rates of lifting flows over refined meshes, the subsonic lifting case ($M_\infty = 0.1$, $AOA = 1.5°$) was simulated on the NACA0012 1024×1024 mesh shown in Fig. 3.44e. Two approaches were considered. First, an Explicit Kutta formulation was used, which neglects the term $\partial\Gamma/\partial\phi_j$ in the Jacobian. Then, an Implicit Kutta formulation (also called Implicit Circulation) was applied, in which this term is included.

Table 3.2 shows that the aerodynamic coefficients remain unchanged when IC is introduced. Similarly, Fig. 3.11 confirms that the pressure coefficient distribution is unaffected.

Table 3.2 Comparison of aerodynamic coefficients for full potential with and without using Implicit Circulation (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Coefficient	Full Potential	Full Potential using IC
C_L	1.829e-01	1.829e-01
C_D	9.419e-04	9.419e-03
C_M	-4.825e-02	-4.825e-02

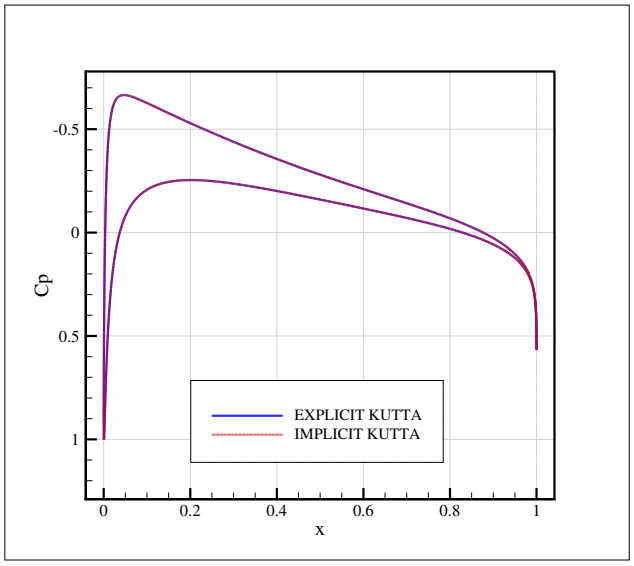
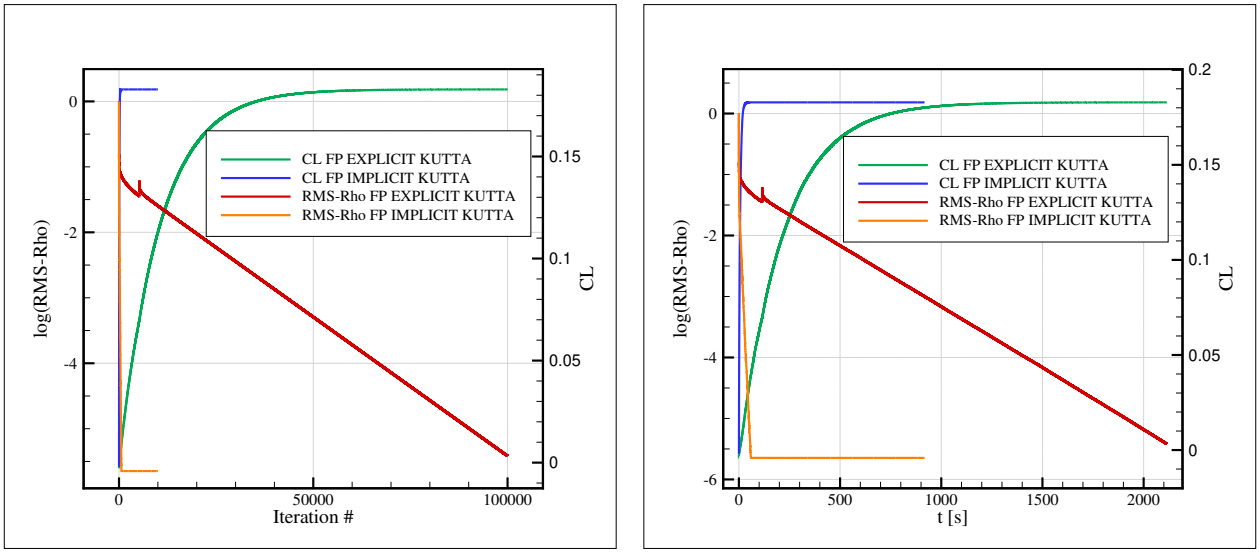


Figure 3.11 Full Potential pressure coefficient distributions with and without using Implicit Circulation (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

However, Fig. 3.12 highlights a significant improvement in convergence when the Implicit Circulation approach is employed. With IC, $\log(\text{RMS-}\rho)$ reaches -5 after approximately 500

iterations and 48 seconds. In contrast, without IC, the same convergence level is achieved only after about 90000 iterations and 1900 seconds.



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

Figure 3.12 Residuals of mass conservation equation for Full Potential with and without using IC as a function of iterations and time (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

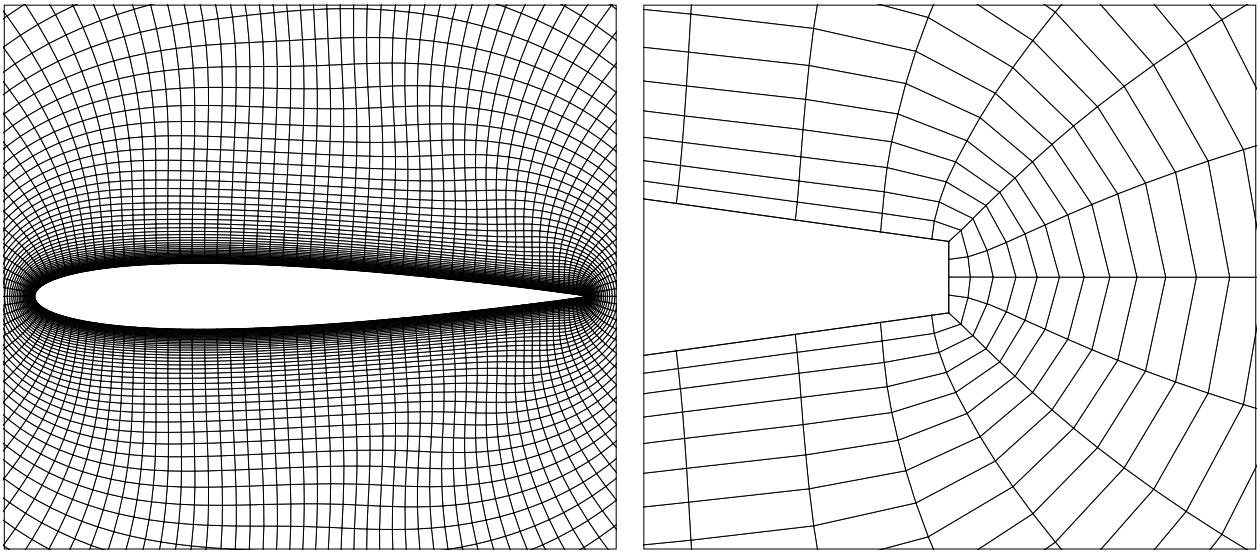
Blunt TE Capabilities

To demonstrate the ability of the FP solver to handle lifting flows around geometries with a blunt trailing edge, a subsonic lifting case ($M_\infty = 0.1$, $AOA = 1.5°$) was simulated on a blunt NACA0012 mesh shown in Fig. 3.13. The O-mesh contains $132 \times 106 = 13992$ quadrilateral cells and was generated using the normal extrusion tool of Pointwise. The initial cell height is $6 \times 10^{-4}\, c$, with a growth rate of 1.1, until the outer boundary reaches a distance of 150 chord lengths from the airfoil.

Table 3.3 compares the aerodynamic coefficients obtained with the Euler and FP solvers. The results show similar values for both approaches, although the lift coefficient predicted by the FP solver is slightly higher. This behavior can be attributed to the way blunt trailing edges are treated in the present Kutta condition, which tends to produce C_L values close to those obtained for sharp trailing edges. As a result, the predicted lift is slightly overestimated compared with the expected value for a blunt geometry, which should produce a smaller lift coefficient. This observation can be confirmed by comparing with the sharp trailing edge results reported in Table 3.2.

The pressure coefficient distributions obtained with both solvers are presented in Fig. 3.14. The results show good agreement between the Euler and FP solutions over the blunt geometry.

Finally, Fig. 3.15 illustrates the convergence behavior of both solvers. The efficiency advantage of the FP solver over the Euler solver is preserved for the blunt trailing-edge configuration. With the FP solver, log(RMS-ρ) reaches -7 after 44 iterations and approximately 2.2 seconds. In contrast, the Euler solver requires about 1700 iterations and 57.4 seconds to reach the same convergence level.



(a) Airfoil view (b) Trailing edge view

Figure 3.13 Close-up view of the blunt NACA0012 mesh

Table 3.3 Comparison of aerodynamic coefficients for Euler and Full Potential methods (blunt NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Coefficient	Euler	Full Potential
C_L	1.730e-01	1.822e-01
C_D	9.478e-03	-5.515e-04
C_M	-4.411e-02	-4.784e-02

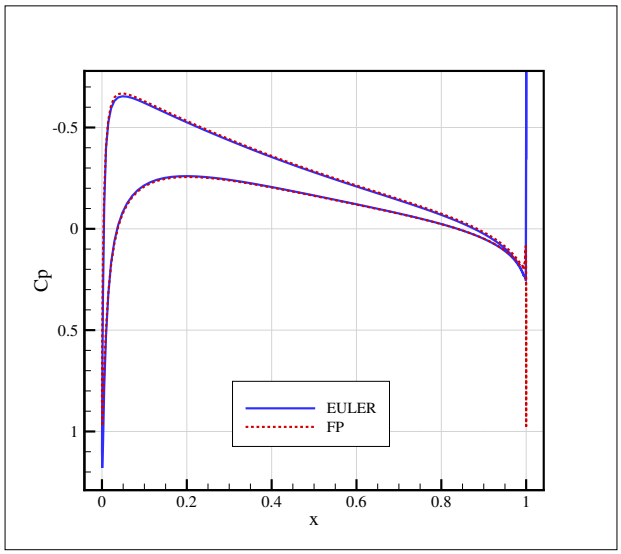
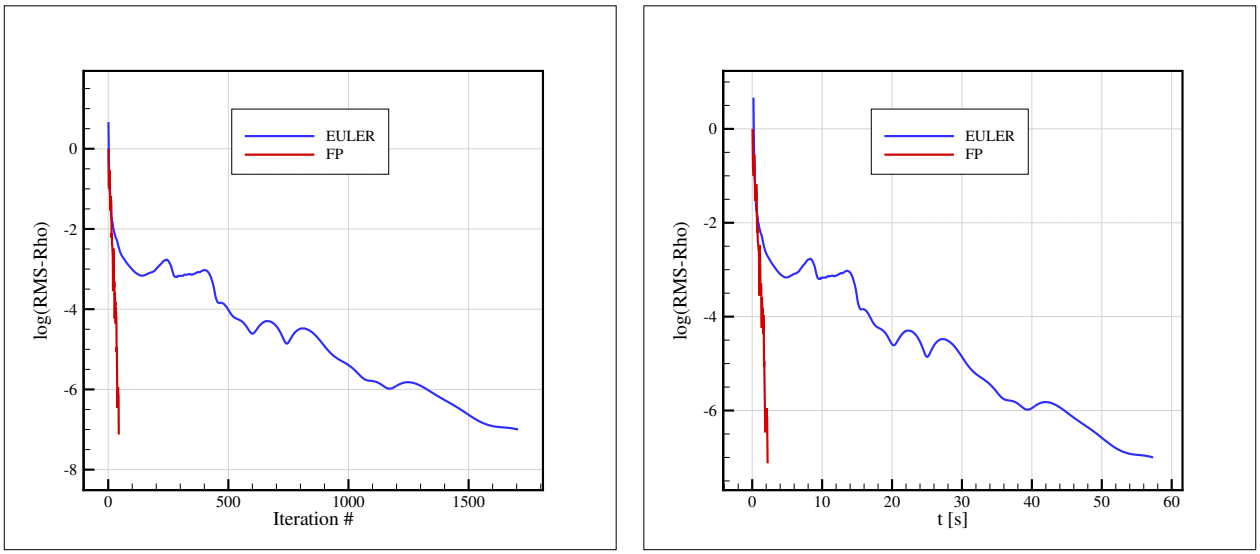


Figure 3.14 Euler and Full Potential pressure coefficient distributions (blunt NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

Figure 3.15 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (blunt NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Unstructured Mesh and Embedded Wake Capabilities

To demonstrate the ability of the present FP solver to handle simulations on unstructured meshes and lifting flows with an embedded wake, where the discrete wake faces do not

necessarily align with the reference wake line, four cases were considered: subsonic non-lifting, subsonic lifting, transonic non-lifting and transonic lifting. These simulations were performed on the unstructured NACA0012 mesh shown in Fig. 3.16 using both the FP and Euler solvers for comparison.

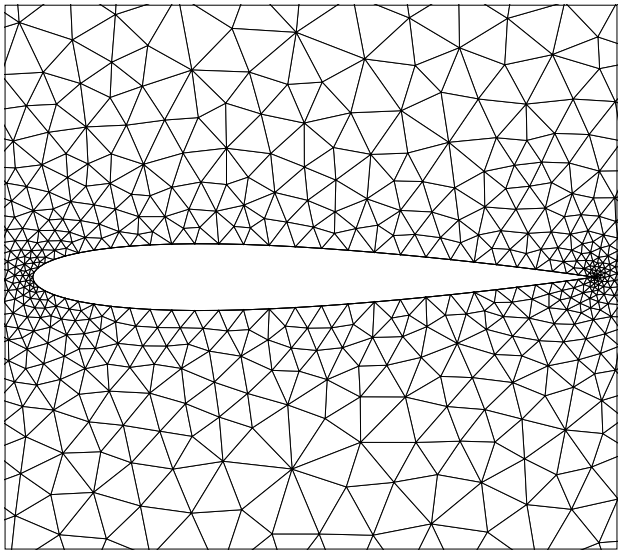


Figure 3.16 Close-up view of the NACA0012 unstructured mesh

The unstructured grid was generated with Pointwise using the unstructured assemble special tool. The inner boundary corresponds to a NACA0012 airfoil discretized with 65 points, while the outer boundary is a circular far-field boundary located at 150 chord lengths from the airfoil and also discretized with 65 points. This process produced a fully unstructured mesh composed of 3828 triangular cells.

For the subsonic non-lifting case ($M_\infty = 0.1$, $AOA = 0.0°$), the aerodynamic coefficients reported in Table 3.4 show good agreement between the two solvers. The pressure coefficient distributions presented in Fig. 3.17 are also in agreement, although the FP solver predicts a slightly lower C_p peak than the Euler solver. Regarding convergence, Fig. 3.18 shows that log(RMS-ρ) reaches -7 after 23 iterations and approximately 0.3 second for the FP solver using the analytical Jacobian, whereas the Euler solver requires 1279 iterations and about 11.7 seconds to reach the same convergence level.

For the subsonic lifting case ($M_\infty = 0.1$, $AOA = 1.5°$), the aerodynamic coefficients reported in Table 3.5 indicate a slightly lower lift coefficient for the FP solver. This difference can be explained by the pressure coefficient distributions shown in Fig. 3.19, where the FP C_p peaks are slightly reduced compared to those predicted by the Euler solver. As shown in Fig. 3.20,

log(RMS-ρ) reaches -7 after 25 iterations and approximately 4.5 seconds for the FP solver using the finite-difference Jacobian, whereas the Euler solver requires 2238 iterations and approximately 20.4 seconds.

For the transonic non-lifting case ($M_\infty = 0.8$, $AOA = 0.0°$), the aerodynamic coefficients presented in Table 3.6 again show good agreement between the two methods. The pressure coefficient distributions in Fig. 3.21 are also consistent, although the shock predicted by the FP solver is more smeared than the Euler solver results, which is a typical characteristic of full potential formulations. In terms of convergence, Fig. 3.22 shows that log(RMS-ρ) reaches -7 after 85 iterations and approximately 22 seconds for the FP solver using the finite-difference Jacobian, whereas the Euler solver reaches the same level after 2081 iterations and about 19.3 seconds.

Finally, for the transonic lifting case ($M_\infty = 0.75$, $AOA = 1.5°$), Table 3.7 indicates a slightly higher lift coefficient for the FP solver. This behavior can be explained by the pressure coefficient distributions shown in Fig. 3.23, where the FP solver predicts a stronger shock located further downstream than in the Euler solution, which is consistent with the isentropic assumption of the full potential formulation. As illustrated in Fig. 3.24, log(RMS-ρ) reaches -5.5 after 134 iterations and approximately 27 seconds for the FP solver using the finite-difference Jacobian, whereas the Euler solver reaches log(RMS-ρ) $= -7$ after 2411 iterations and approximately 22.4 seconds.

Overall, the FP results obtained on the unstructured mesh show good agreement with the Euler solutions. However, the discrepancies between the two approaches appear slightly larger than those observed on structured meshes. Indeed, the FP solver is able to solve lifting flows for embedded wake meshes, but it is still sensitive to the discrete wake faces orientation and seems to yield better results when it is aligned with the wake. In addition, while the FP solver retains a clear efficiency advantage for subsonic cases, this advantage is reduced or lost for transonic cases, similarly to what was observed on structured grids.

Table 3.4 Comparison of aerodynamic coefficients for Euler and Full Potential (NACA0012 unstructured grid, $M_\infty = 0.1$, $AOA = 0.0°$)

Coefficient	Euler	Full Potential
C_L	-3.288e-04	9.088e-04
C_D	1.039e-03	1.207e-02
C_M	-6.889e-05	-4.747e-04

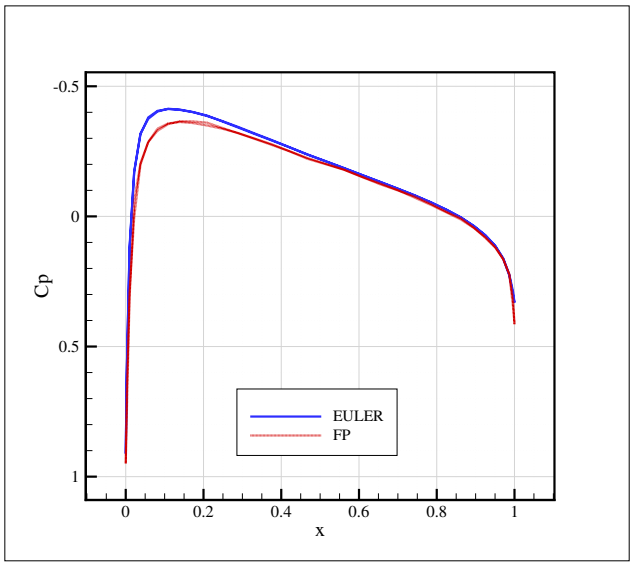
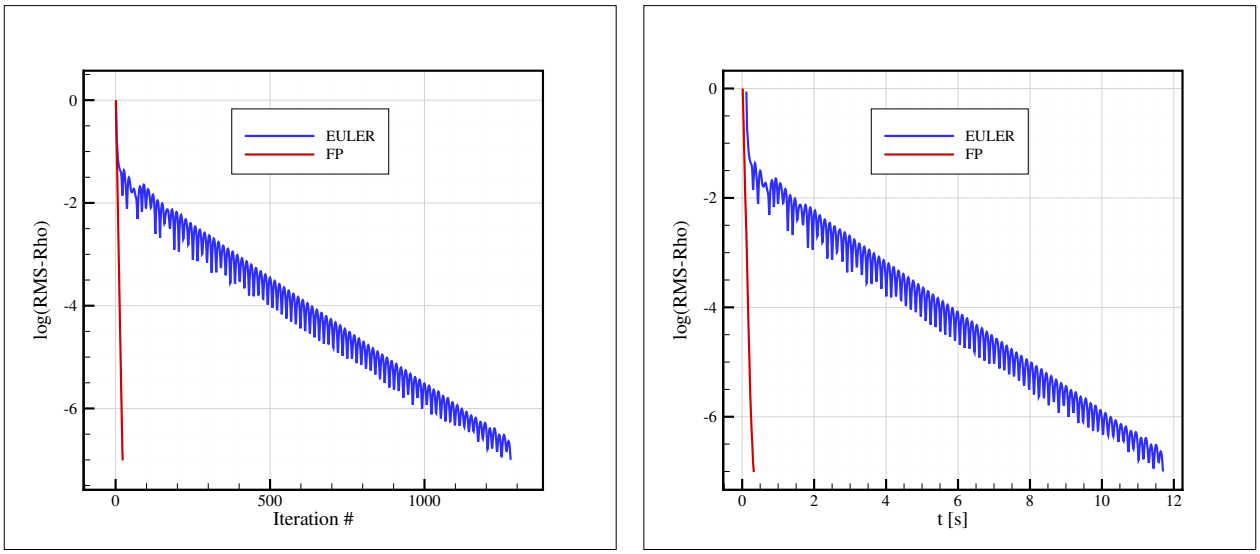


Figure 3.17 Euler compared to Full Potential pressure coefficient distributions (NACA0012 unstructured grid, $M_\infty = 0.1$, $AOA = 0.0°$).



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

Figure 3.18 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (NACA0012 unstructured grid, $M_\infty = 0.1$, $AOA = 0.0°$)

Table 3.5 Comparison of aerodynamic coefficients for Euler and Full Potential (NACA0012 unstructured grid, $M_\infty = 0.1$, $AOA = 1.5°$)

Coefficient	Euler	Full Potential
C_L	1.764e-01	1.684e-01
C_D	1.348e-03	1.305e-02
C_M	-4.639e-02	-4.455e-02

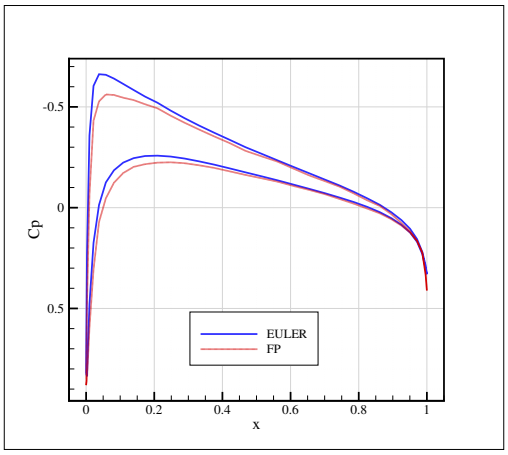
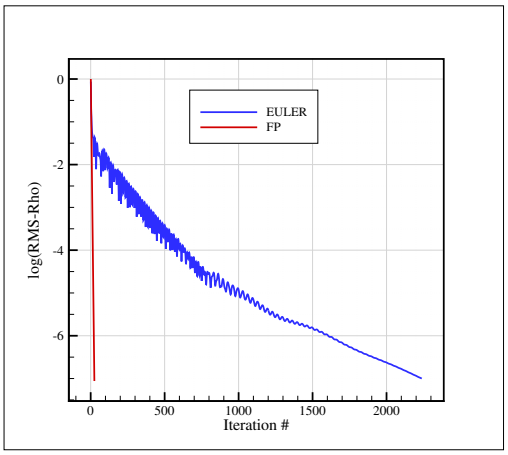
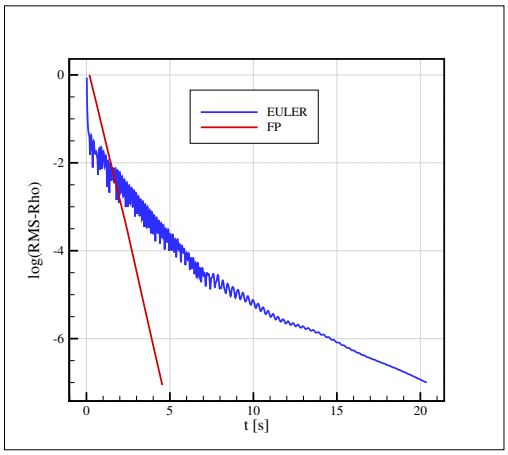


Figure 3.19 Euler compared to Full Potential pressure coefficient distributions (NACA0012 unstructured grid, $M_\infty = 0.1$, $AOA = 1.5°$).



(a) log(RMS-ρ) vs number of iterations



(b) log(RMS-ρ) vs time [s]

Figure 3.20 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (NACA0012 unstructured grid, $M_\infty = 0.1$, $AOA = 1.5°$)

Table 3.6 Comparison of aerodynamic coefficients for Euler and Full Potential (NACA0012 unstructured grid, $M_\infty = 0.8$, $AOA = 0.0°$)

Coefficient	Euler	Full Potential
C_L	-2.686e-03	-1.667e-03
C_D	1.018e-02	1.161e-02
C_M	7.510e-04	5.879e-04

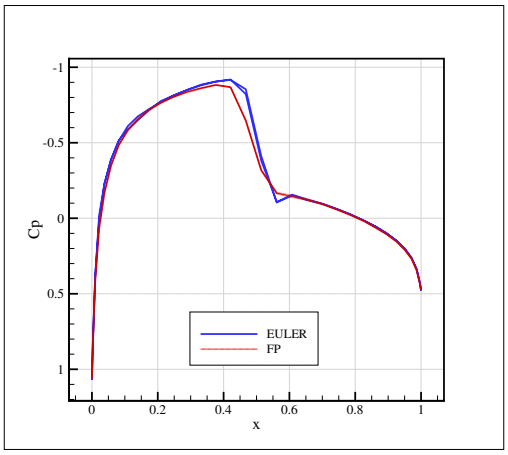
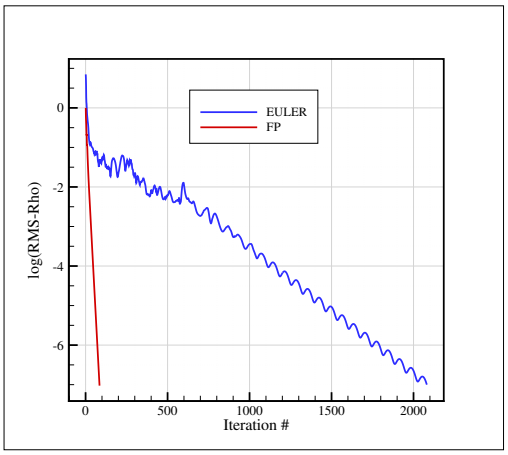
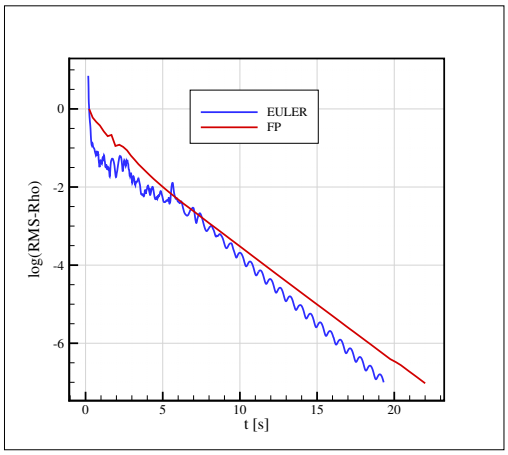


Figure 3.21 Euler compared to Full Potential pressure coefficient distributions (NACA0012 unstructured grid, $M_\infty = 0.8$, $AOA = 0.0°$).



(a) log(RMS-ρ) vs number of iterations



(b) log(RMS-ρ) vs time [s]

Figure 3.22 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (NACA0012 unstructured grid, $M_\infty = 0.8$, $AOA = 0.0°$)

Table 3.7 Comparison of aerodynamic coefficients for Euler and Full Potential (NACA0012 unstructured grid, $M_\infty = 0.75$, $AOA = 1.5°$)

Coefficient	Euler	Full Potential
C_L	3.086e-01	3.565e-01
C_D	7.784e-03	1.662e-02
C_M	-7.699e-02	-9.095e-02

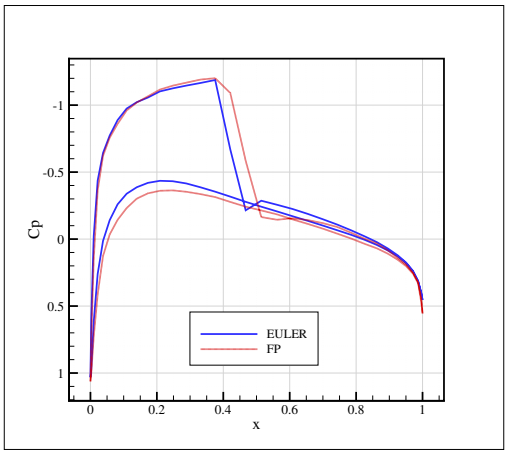
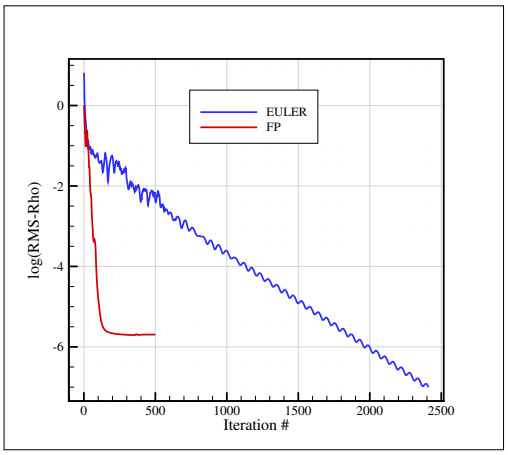
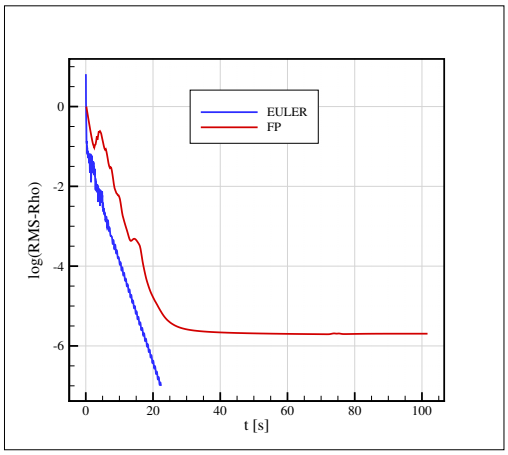


Figure 3.23 Euler compared to Full Potential pressure coefficient distributions (NACA0012 unstructured grid, $M_\infty = 0.75$, $AOA = 1.5°$).



(a) log(RMS-ρ) vs number of iterations



(b) log(RMS-ρ) vs time [s]

Figure 3.24 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (NACA0012 unstructured grid, $M_\infty = 0.75$, $AOA = 1.5°$)

RAE2822 Airfoil

To assess the capability of the present FP solver to simulate flows over supercritical airfoils, two test cases were considered on the RAE2822 airfoil: a subsonic lifting case and a transonic lifting one. The simulations were performed on the mesh shown in Fig. 3.25. The O-mesh contains $128 \times 106 = 13568$ quadrilateral cells and was generated using the normal extrusion tool of Pointwise. The initial cell height is $6 \times 10^{-4}\, c$, with a growth rate of 1.1 and the outer boundary is located at a distance of 150 chord lengths from the airfoil.

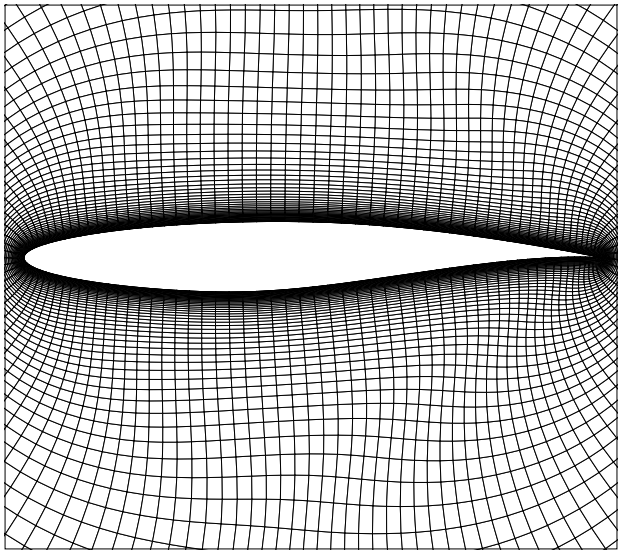


Figure 3.25 Close-up view of the RAE2822 mesh

For the subsonic lifting case ($M_\infty = 0.1$, $AOA = 2.0°$), Table 3.8 compares the aerodynamic coefficients obtained with the Euler and FP solvers. The results show good agreement between the two methods. Similarly, the pressure coefficient distributions presented in Fig. 3.26 are nearly identical for both solvers. The convergence behavior is shown in Fig. 3.27. In this case, the efficiency advantage of the FP solver over the Euler solver is preserved. Indeed, the FP solver log(RMS-ρ) reaches -7 after 27 iterations and approximately 14.5 seconds when using the finite-difference Jacobian, whereas the Euler solver requires about 939 iterations and 30.6 seconds to reach the same convergence level.

Table 3.8 Comparison of aerodynamic coefficients for Euler and Full Potential methods (RAE2822, $M_\infty = 0.1$, $AOA = 2.0°$)

Coefficient	Euler	Full Potential
C_L	4.916e-01	4.967e-01
C_D	6.872e-03	1.023e-03
C_M	-2.018e-01	-2.034e-01

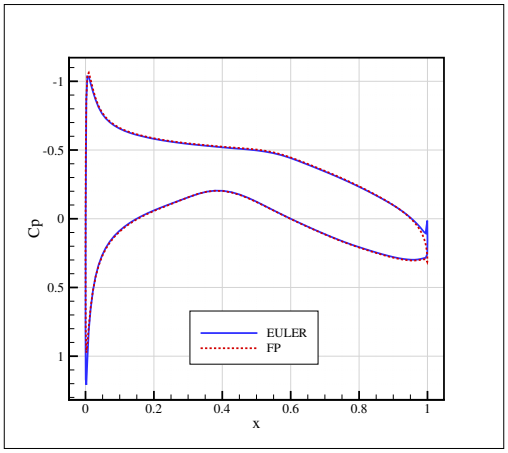
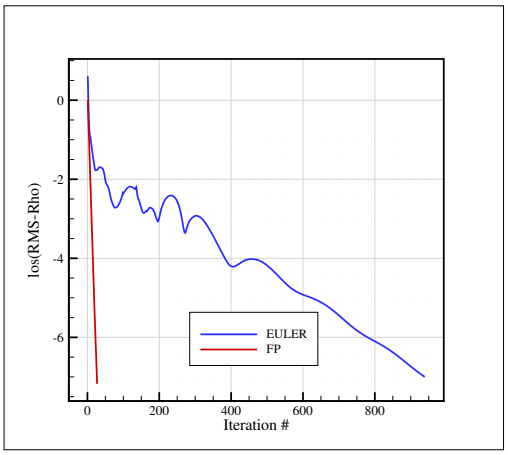
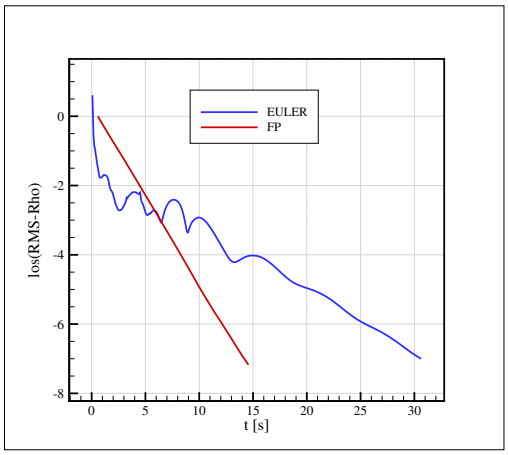


Figure 3.26 Euler and Full Potential pressure coefficient distributions (RAE2822, $M_\infty = 0.1$, $AOA = 2.0°$)



(a) log(RMS-ρ) vs number of iterations



(b) log(RMS-ρ) vs time [s]

Figure 3.27 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (RAE2822, $M_\infty = 0.1$, $AOA = 2.0°$)

For the transonic lifting case ($M_\infty = 0.715$, $AOA = 2.0°$), Table 3.9 compares the aerodynamic coefficients predicted by the two solvers. Overall agreement is observed, although the FP solver predicts a slightly higher lift coefficient. This difference can be explained by the pressure coefficient distributions shown in Fig. 3.28, where the shock predicted by the FP solver is located further downstream and is stronger than in the Euler solution. The convergence behavior for this case is illustrated in Fig. 3.29. In contrast to the subsonic case, the efficiency advantage of the FP solver is lost in the transonic regime, which is consistent with the behavior observed previously. Since the FP solver had difficulties to converge for this case, it was initially started using the analytical Jacobian and later restarted with the finite-difference Jacobian. Under these conditions, log(RMS-ρ) reaches -5.2 after approximately 10^5 iterations and about 2400 seconds. By comparison, the Euler solver reaches the same convergence level after about 2568 iterations and 84 seconds.

Table 3.9 Comparison of aerodynamic coefficients for Euler and Full Potential methods (RAE2822, $M_\infty = 0.715$, $AOA = 2.0°$)

Coefficient	Euler	Full Potential
C_L	8.205e-01	8.477e-01
C_D	2.820e-03	-2.865e-04
C_M	-3.210e-01	-3.312e-01

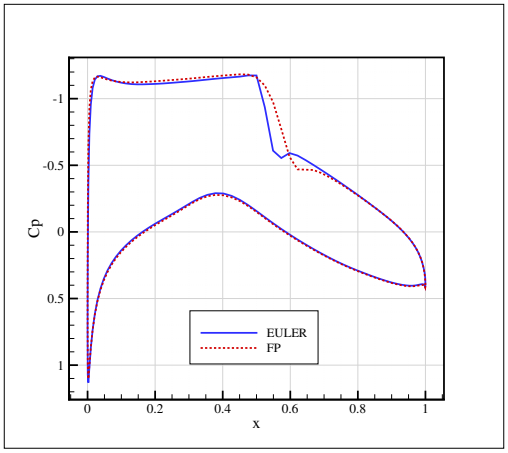
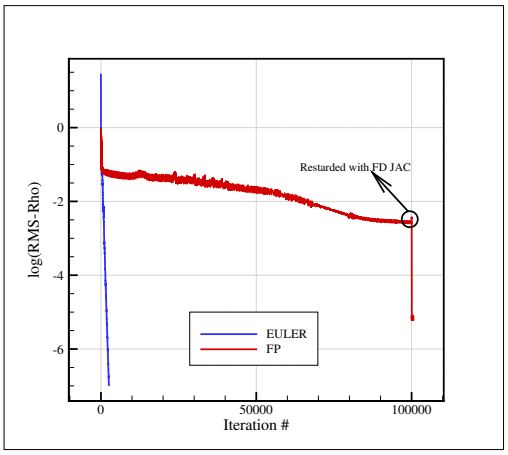
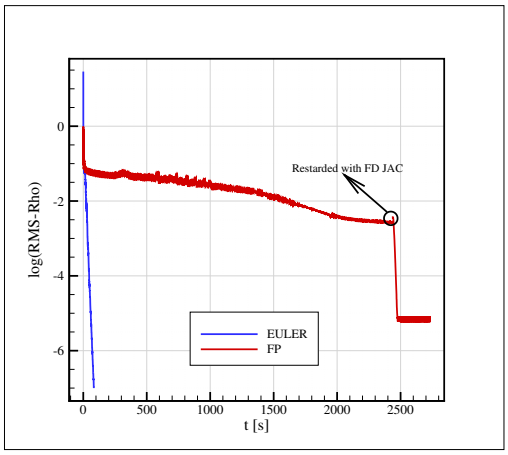


Figure 3.28 Euler and Full Potential pressure coefficient distributions (RAE2822, $M_\infty = 0.715$, $AOA = 2.0°$)



(a) log(RMS-ρ) vs number of iterations



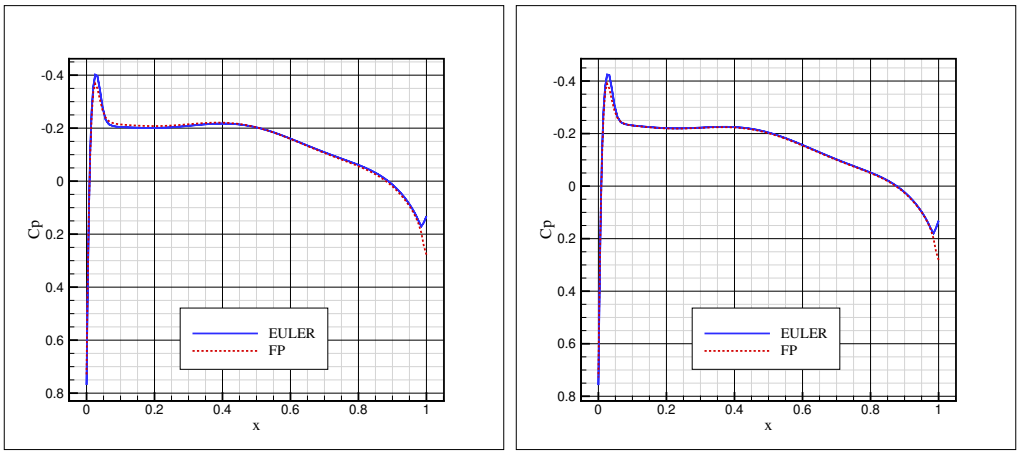
(b) log(RMS-ρ) vs time [s]

Figure 3.29 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (RAE2822, $M_\infty = 0.715$, $AOA = 2.0°$)

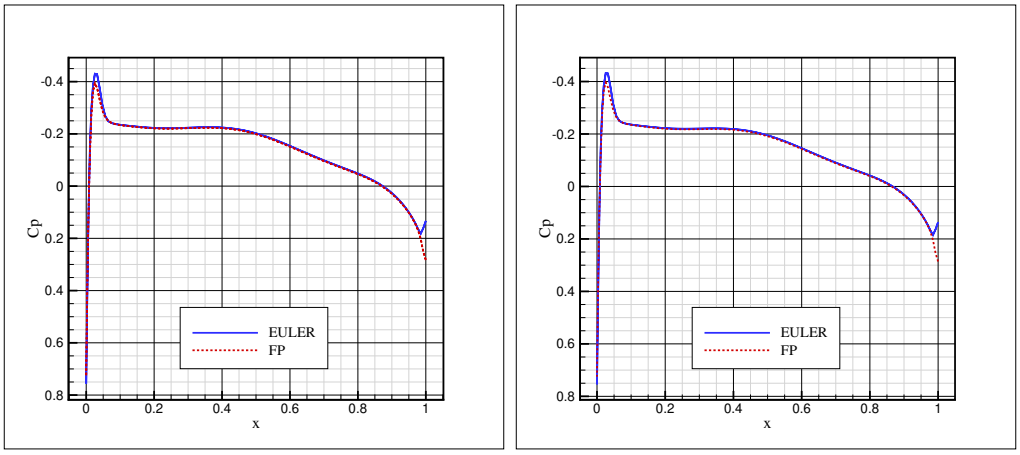
ONERA M6 Wing

To demonstrate the three-dimensional capabilities of the present FP solver, simulations of the flow over the ONERA M6 wing were performed for four flow conditions: subsonic non-lifting, subsonic lifting, transonic non-lifting and transonic lifting. The results obtained with the FP solver were compared with those of the Euler solver. The simulations were carried out on a grid composed of $33 \times 96 \times 64 = 202752$ hexahedral cells arranged in a C–C mesh topology. This mesh was obtained by coarsening the grid published in [116], which was originally designed for RANS simulations and therefore highly refined in the wall-normal direction to resolve the boundary layer. The coarsening was performed using Cassiopee [117] in order to obtain cells with reduced stretching and a more cube-like shape.

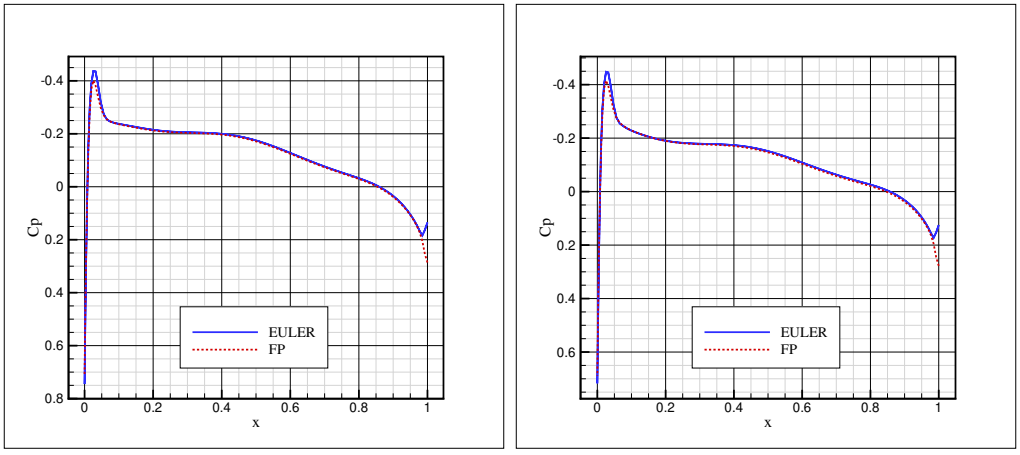
For the subsonic non-lifting case ($M_\infty = 0.1$, $AOA = 0.0°$), the pressure coefficient distributions at the six semi-span locations shown in Fig. 3.30 exhibit good agreement between the FP and Euler solvers. The convergence behavior is presented in Fig. 3.31. In this case, log(RMS-ρ) reaches -7 after 1385 iterations and approximately 23.2 seconds for the FP solver using the analytical Jacobian and the GMRES linear solver, whereas the Euler solver requires 8533 iterations and about 65.1 seconds to reach the same convergence level.



(a) 20% $\frac{y}{b}$ (b) 44% $\frac{y}{b}$

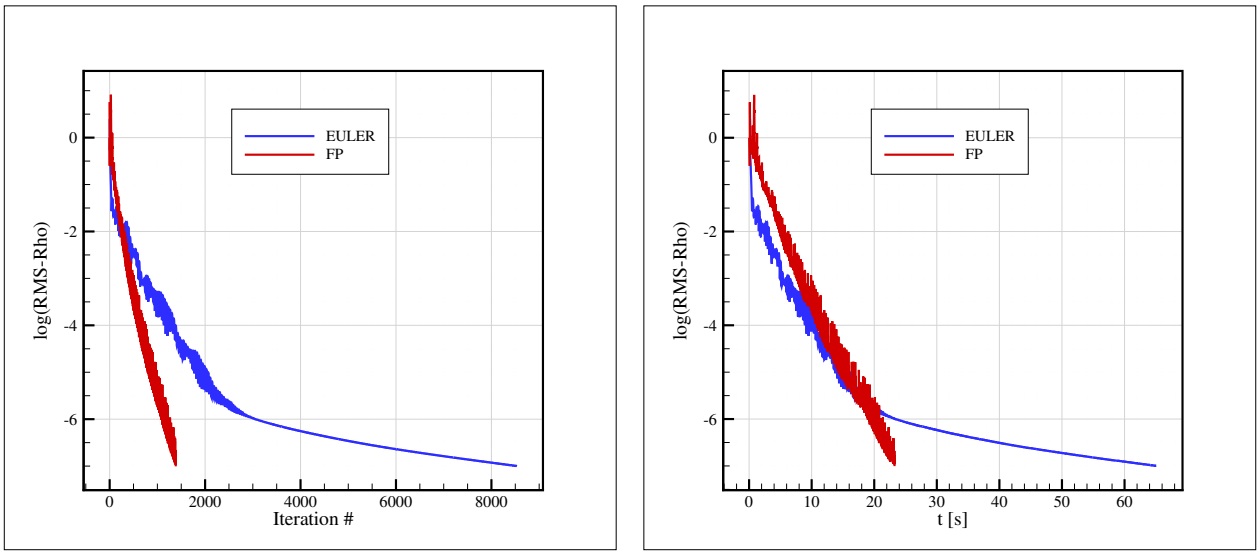


(c) 65% $\frac{y}{b}$ (d) 80% $\frac{y}{b}$



(e) 90% $\frac{y}{b}$ (f) 95% $\frac{y}{b}$

Figure 3.30 Euler and Full Potential pressure coefficient distributions (ONERA M6 wing, $M_\infty = 0.1$, $AOA = 0.0°$)

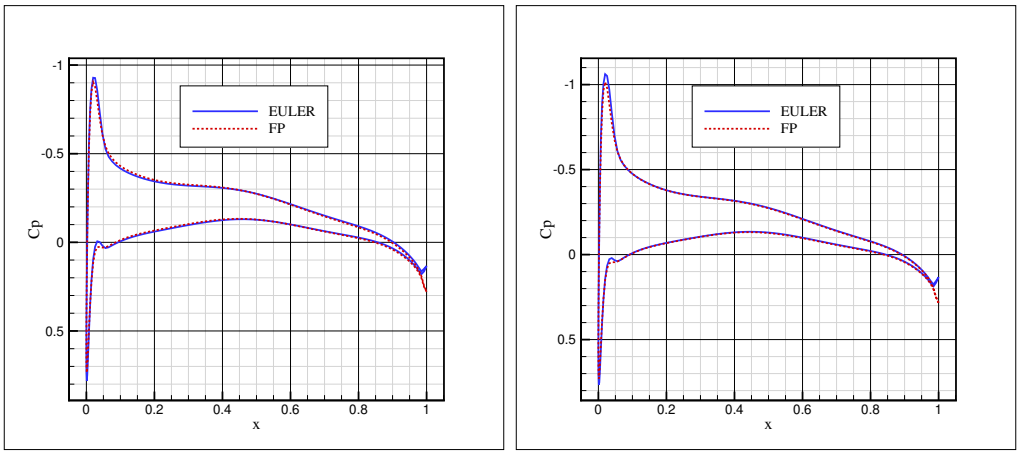


(a) log(RMS-ρ) vs number of iterations

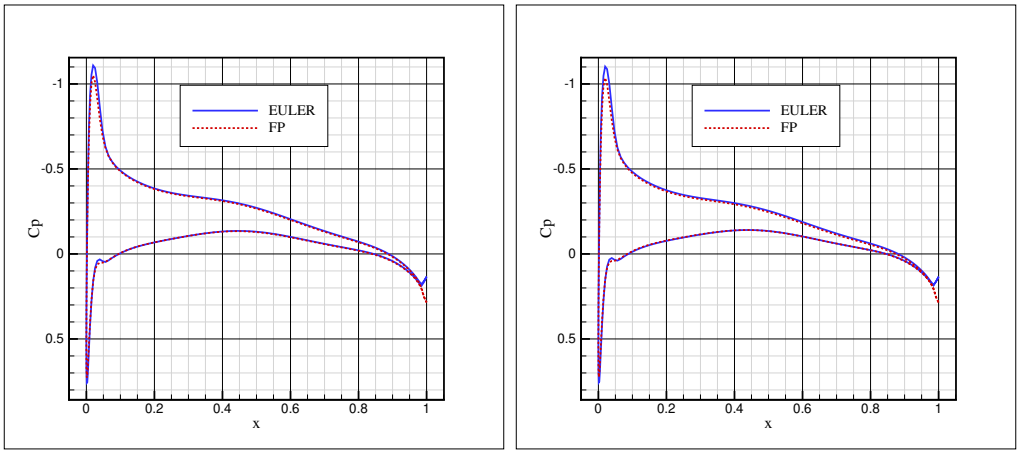
(b) log(RMS-ρ) vs time [s]

Figure 3.31 Evolution of mass residuals for Full Potential and Euler solvers as a function of iterations and time (ONERA M6 wing, $M_\infty = 0.1$, $AOA = 0.0°$)

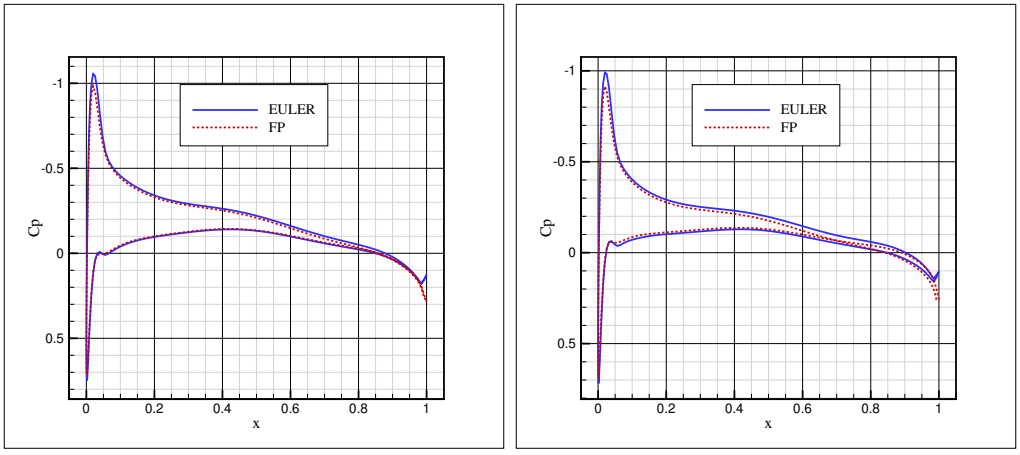
For the subsonic lifting case ($M_\infty = 0.1$, $AOA = 3.06°$), the pressure coefficient distributions shown in Fig. 3.32 remain in close agreement for most spanwise locations. However, discrepancies appear at the $y/b = 95\%$ section, where the pressure difference between the upper and lower surfaces is reduced and even reversed beyond $x/c = 0.7$. This behavior is likely related to the current implementation of the Kutta condition, which becomes ambiguous in the vicinity of the wingtip. The literature provides limited guidance on how to treat this specific region. It should also be noted that the implicit circulation formulation was not implemented for three-dimensional problems, which explains the reduced efficiency compared with the non-lifting case. As shown in Fig. 3.33, log(RMS-ρ) reaches approximately -7 after 4776 iterations and 34.7 seconds for the FP solver using the analytical Jacobian and the SGS linear solver, whereas the Euler solver requires 5417 iterations and about 50.5 seconds.



(a) 20% $\frac{y}{b}$ (b) 44% $\frac{y}{b}$

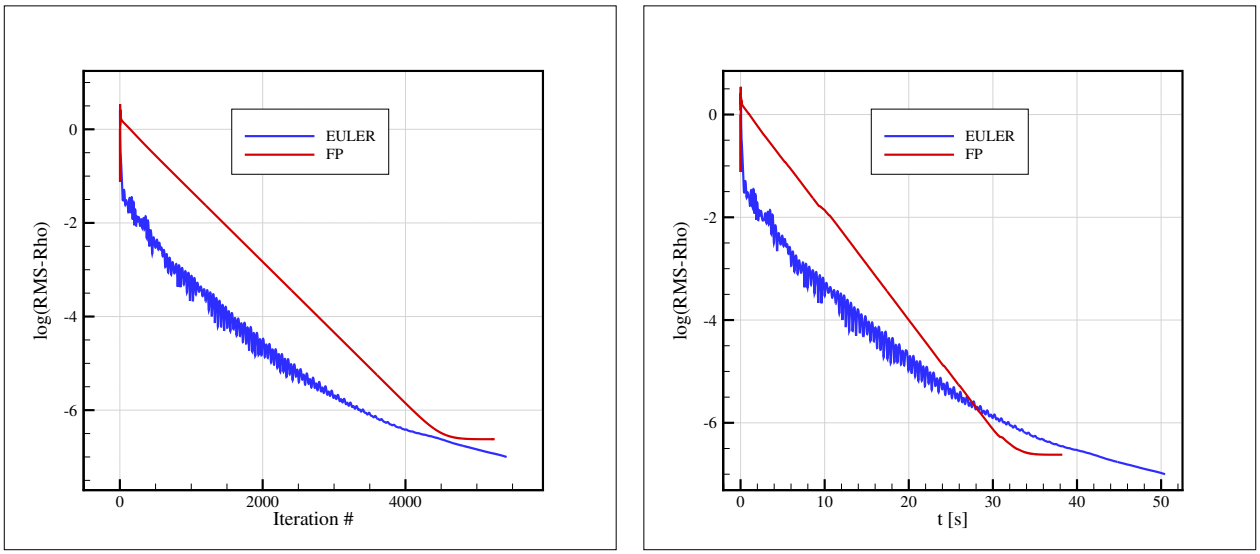


(c) 65% $\frac{y}{b}$ (d) 80% $\frac{y}{b}$



(e) 90% $\frac{y}{b}$ (f) 95% $\frac{y}{b}$

Figure 3.32 Euler and Full Potential pressure coefficient distributions (ONERA M6 wing, $M_\infty = 0.1$, $AOA = 3.06°$)

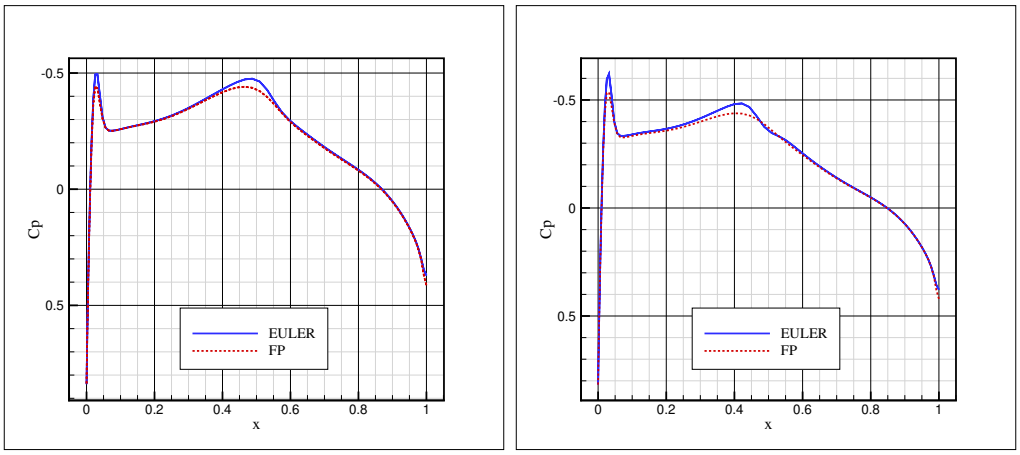


(a) log(RMS-ρ) vs number of iterations

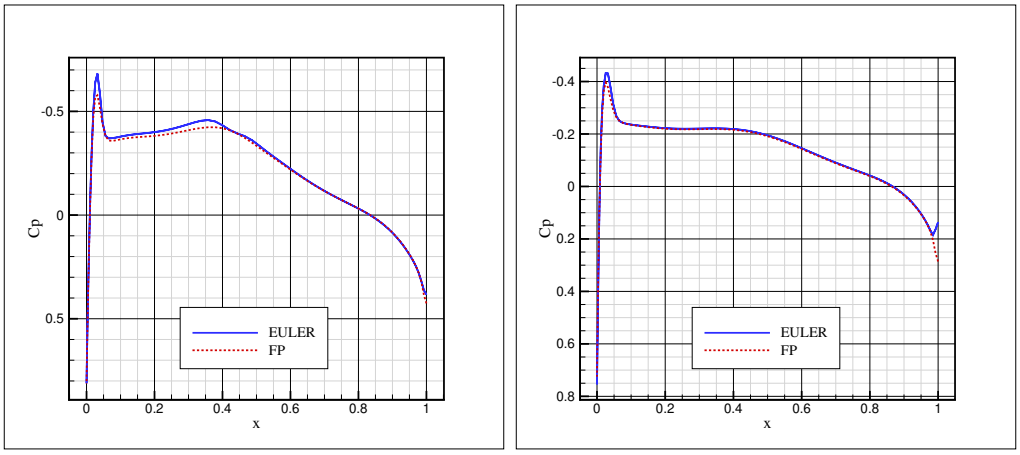
(b) log(RMS-ρ) vs time [s]

Figure 3.33 Evolution of mass residuals for Full Potential and Euler solvers as a function of iterations and time (ONERA M6 wing, $M_\infty = 0.1$, $AOA = 3.06°$)

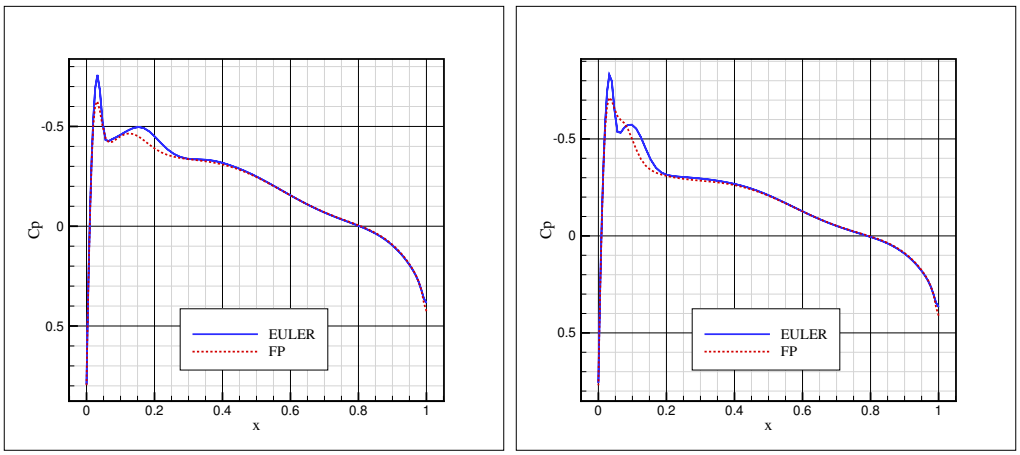
For the transonic non-lifting case ($M_\infty = 0.84$, $AOA = 0.0°$), the pressure coefficient distributions at the six semi-span locations (Fig. 3.34) again show good agreement between the two solvers. The convergence histories presented in Fig. 3.35 indicate that log(RMS-ρ) reaches -7 after approximately 244430 iterations and 800 seconds for the FP solver when using the explicit iterative approach, whereas the Euler solver reaches the same convergence level after 12077 iterations and about 95.7 seconds. In this case, the finite-difference Jacobian was not yet available for three-dimensional simulations. The explicit iterative method was therefore used, as it generally provides greater robustness than analytical Jacobian implicit methods for transonic flows.



(a) 20% $\frac{y}{b}$ (b) 44% $\frac{y}{b}$

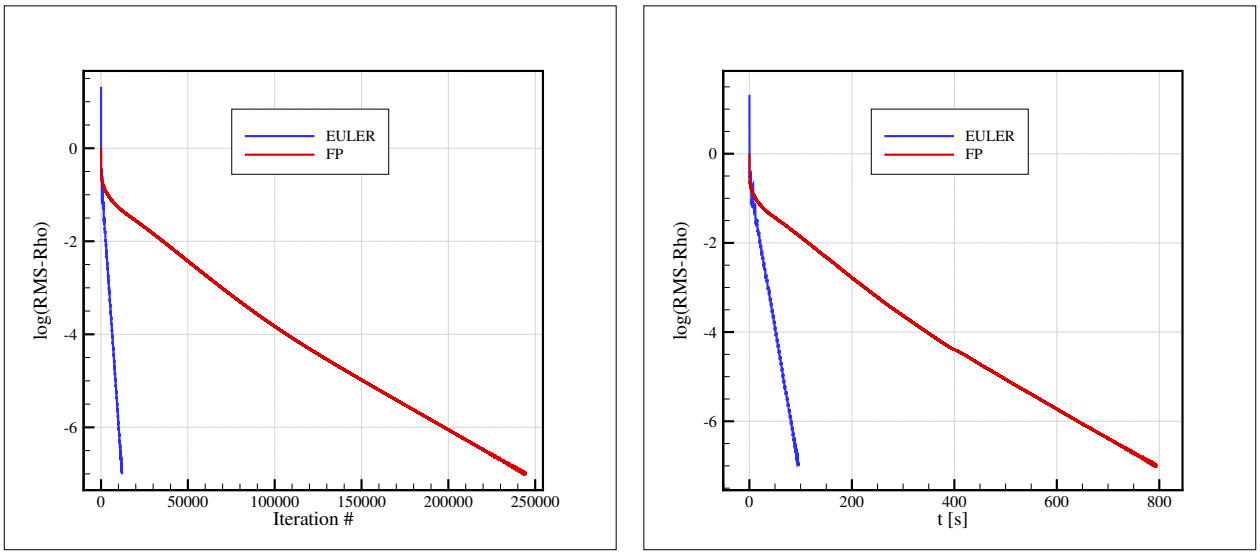


(c) 65% $\frac{y}{b}$ (d) 80% $\frac{y}{b}$



(e) 90% $\frac{y}{b}$ (f) 95% $\frac{y}{b}$

Figure 3.34 Euler and Full Potential pressure coefficient distributions (ONERA M6 wing, $M_\infty = 0.84$, $AOA = 0.0°$)

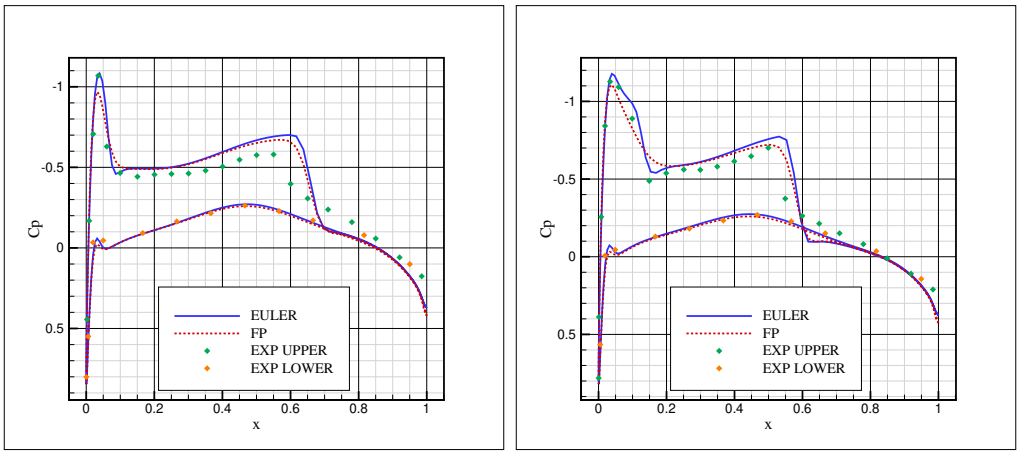


(a) log(RMS-ρ) vs number of iterations

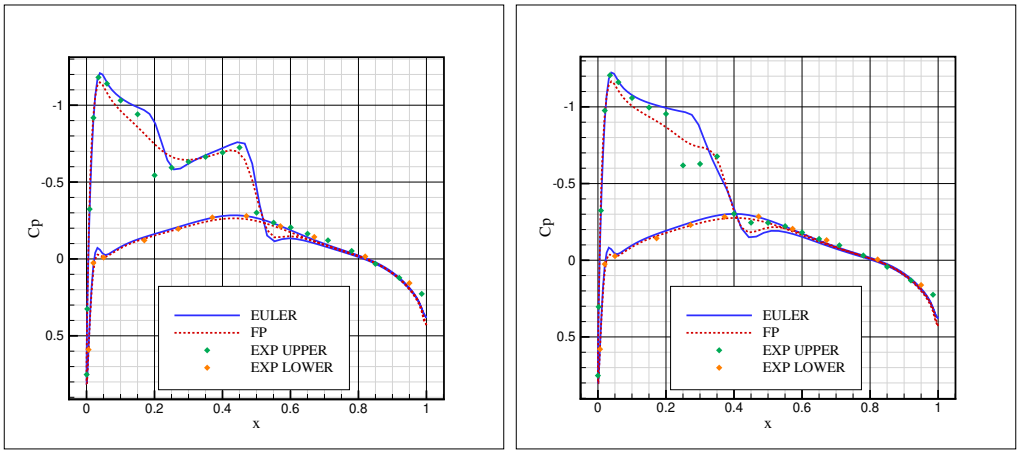
(b) log(RMS-ρ) vs time [s]

Figure 3.35 Evolution of mass residuals for Full Potential and Euler solvers as a function of iterations and time (ONERA M6 wing, $M_\infty = 0.84$, $AOA = 0.0°$)

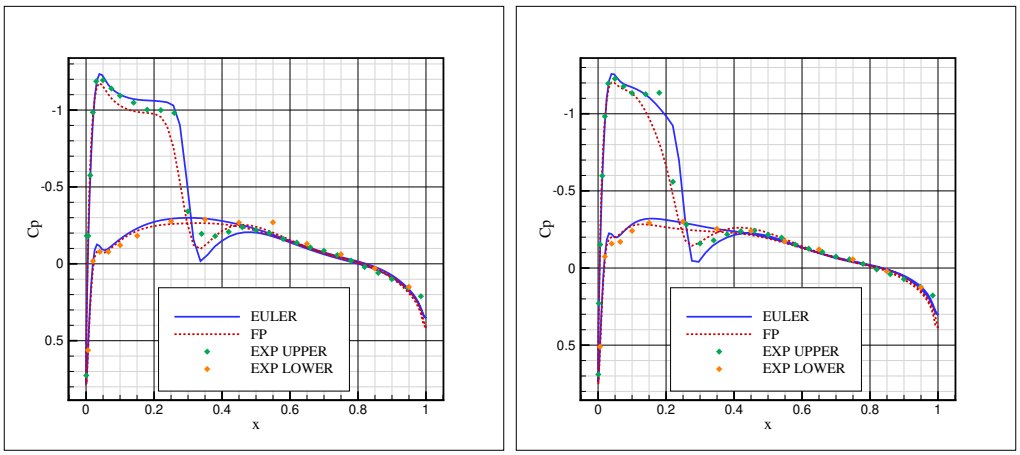
Finally, the transonic lifting case ($M_\infty = 0.84$, $AOA = 3.06°$), which is commonly used as a benchmark in the literature, was also investigated. The pressure coefficient distributions at the six semi-span locations shown in Fig. 3.36 are compared with both Euler results and experimental measurements from [118]. The FP and Euler solutions are in close agreement, although the shocks predicted by the FP solver appear slightly more smeared. Both numerical solutions also show good overall agreement with the experimental data, with minor differences in shock location likely caused by viscous effects associated with the boundary layer, which are neglected in both FP and Euler formulations. This case proved to be the most challenging for the FP solver and required the use of the more robust but slower explicit iterative approach. As shown in Fig. 3.37, log(RMS-ρ) reaches -4.8 after approximately 1.5×10^6 iterations and about 8900 seconds for the FP solver, whereas the Euler solver reaches log(RMS-ρ) $= -7$ after 8900 iterations and about 70 seconds.



(a) 20% $\frac{y}{b}$ (b) 44% $\frac{y}{b}$

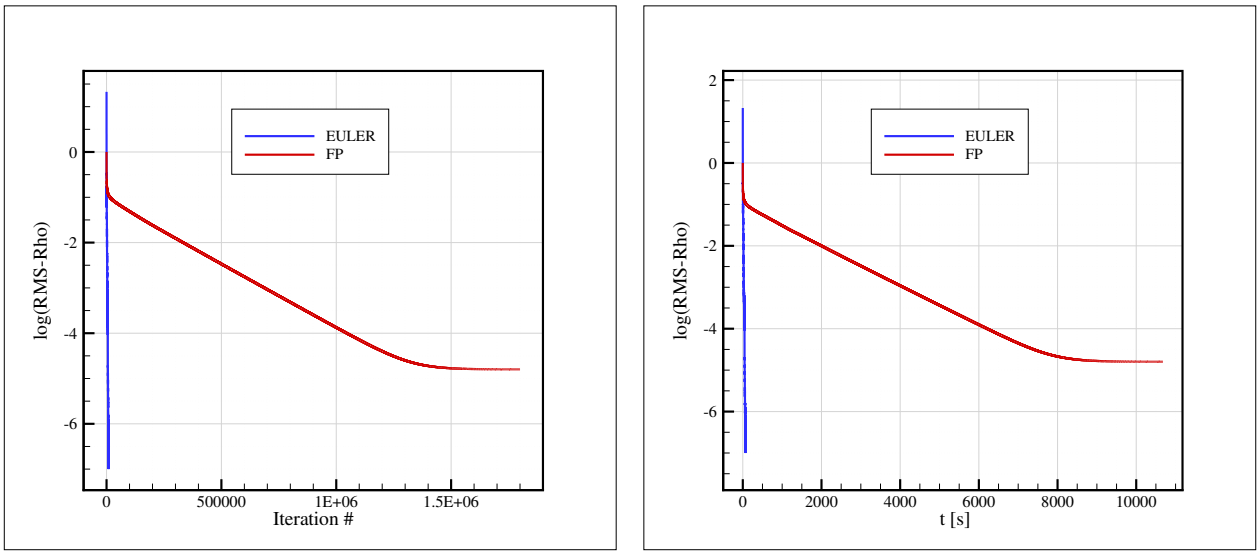


(c) 65% $\frac{y}{b}$ (d) 80% $\frac{y}{b}$



(e) 90% $\frac{y}{b}$ (f) 95% $\frac{y}{b}$

Figure 3.36 Euler and Full Potential pressure coefficient distributions (ONERA M6 wing, $M_\infty = 0.84$, $AOA = 3.06°$)



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

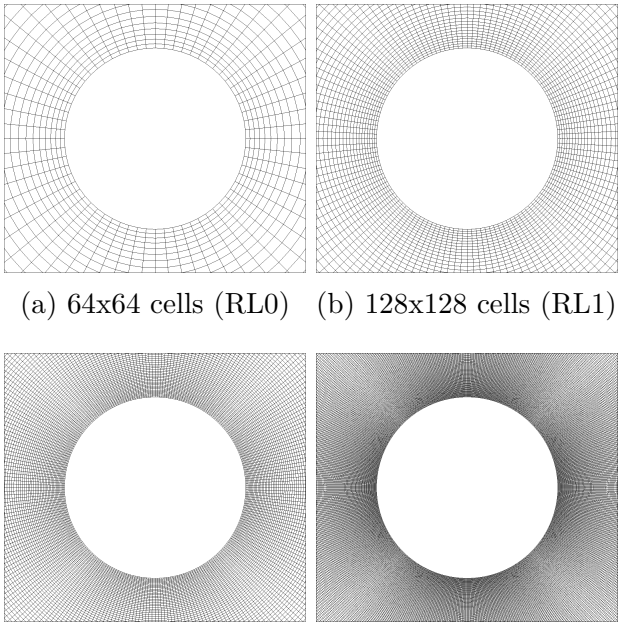
Figure 3.37 Evolution of mass residuals for Full Potential and Euler solvers as a function of iterations and time (ONERA M6 wing, $M_\infty = 0.84$, $AOA = 3.06°$)

Overall, the FP results obtained for the ONERA M6 wing show good agreement with the Euler solutions. As observed in previous test cases, the FP solver retains a clear efficiency advantage for subsonic conditions, whereas this advantage is significantly reduced or lost in transonic regimes. Additionally, the treatment of the Kutta condition in the vicinity of the wingtip requires further investigation. Specifically, it remains unclear how the potential jump should be handled when computing gradients for faces located further along the spanwise direction that have stencil elements on both sides of the wing. These faces can experience very large potential gradients, which often lead to numerical instability and premature termination of the simulation in the current implementation.

Discretization Error Verification

A practical way to obtain the order of convergence of the discretization error for this solver is to use the canonical case of an incompressible flow over a cylinder since the exact solution of this case is known.

The meshes were generated in Pointwise using the normal-extrude tool and follow an O-mesh topology. Each refinement level was obtained by doubling the resolution of the previous mesh, producing a sequence starting at 64x64 cells and ending at 512x512 cells. Close-up views of the four refinement levels are presented in Figure 3.38.



(a) 64x64 cells (RL0) (b) 128x128 cells (RL1)

(c) 256x256 cells (RL2) (d) 512x512 cells (RL3)

Figure 3.38 Body-fitted cylinder series of meshes

The exact solution for the potential of an incompressible flow over a cylinder is the following in cartesian coordinates:

$$\phi(x, y) = u_\infty \left(x + \frac{x}{x^2 + y^2} \right). \tag{3.52}$$

The velocity components can be obtained by deriving ϕ with respect to x and y respectively:

$$u(x, y) = u_\infty \left(1 - \frac{x^2 - y^2}{(x^2 + y^2)^2} \right), \tag{3.53}$$

$$v(x, y) = -\frac{2\,x\,y\,u_\infty}{(x^2 + y^2)^2}. \tag{3.54}$$

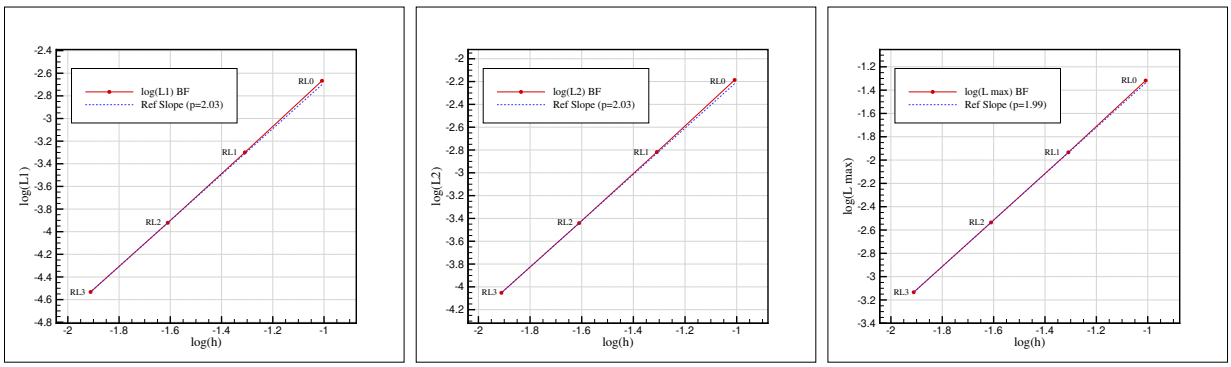
Then, the velocities obtained numerically by both approaches for the flow over a cylinder at $M_\infty = 1 \times 10^{-7}$ could be compared to the exact solution to build three different metrics (L_1, L_2, L_∞):

$$L_1 = \frac{1}{\#facets} \sum_{i=1}^{\#facets} \left(|u_i - u_{exact,i}| + |v_i - v_{exact,i}| \right), \tag{3.55}$$

$$L_2 = \sqrt{\frac{1}{\#facets} \sum_{i=1}^{\#facets} \left(|u_i - u_{exact,i}| + |v_i - v_{exact,i}|\right)^2}, \tag{3.56}$$

$$L_\infty = \max_{i=1}^{\#facets} \left(|u_i - u_{exact,i}| + |v_i - v_{exact,i}|\right). \tag{3.57}$$

Figure 3.39 presents the velocity error metrics as a function of the minimum cell size $log(h)$. The observed numerical order of convergence is close to 2, in agreement with the theoretical second-order accuracy expected from the K-Exact least-squares formulation (see Subsection 3.1.5). This agreement provides confidence in the correctness of the full potential body-fitted implementation. Achieving this result required two key modifications. First, ghost cells were incorporated into all $\nabla\phi$ stencils, rather than only those adjacent to the boundary faces. Second, the wall-face gradients had to be recomputed using a second-order method. A centered finite-difference derivative with the correct tangential direction was used because the combination of the $\nabla\phi$ direction forcing and the K-Exact least-squares gradient method tended to reduce the effective order of accuracy to 1.



(a) $log(L_1)$ vs $log(h)$ (b) $log(L_2)$ vs $log(h)$ (c) $log(L_\infty)$ vs $log(h)$

Figure 3.39 Error metrics as a function of the minimum cell size $log(h)$ using the body-fitted approach

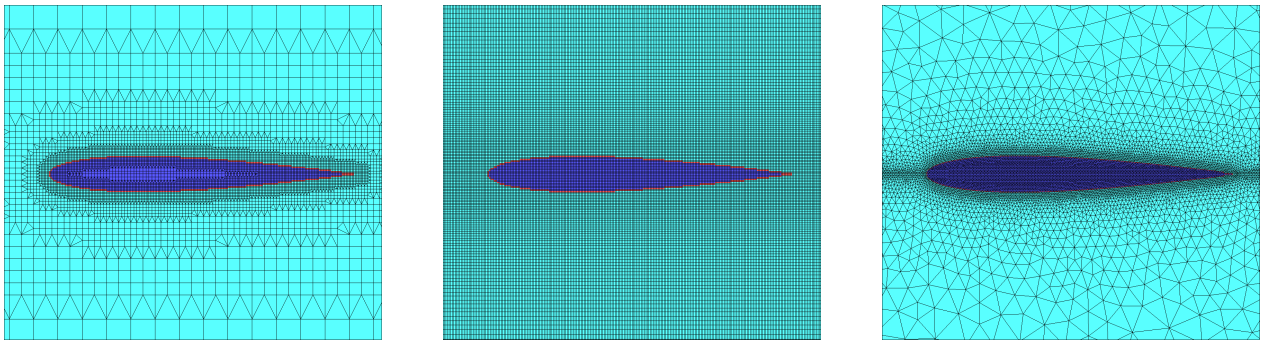
3.2 Integration of a Ghost-Cell Immersed Boundary Method

3.2.1 Mesh Configurations

The immersed boundary ghost-cell method facilitates the mesh generation process by allowing simulations over a wide range of mesh types without requiring wall-conforming grids. In this work, three mesh families were successfully tested: quadtree, cartesian and unstructured triangular meshes.

For a comparable level of refinement, corresponding to a minimum cell size of 0.005 chord, all three meshes were generated. The quadtree mesh consists of 11672 cells (9302 quadrilaterals and 2370 triangles). It was generated using the MixedQuadTree program [119], which avoids hanging nodes by inserting transitional triangles between refinement levels. The cartesian mesh consists of 580 x 368 = 213440 cells. It was generated in-house with CHAMPS as described in Subsection 3.2.5 and corresponds to the second level of refinement of the cartesian series of mesh presented at Figure 3.45. The unstructured triangular mesh, composed of 39662 cells, was generated using Pointwise and its unstructured assembly tool. All three meshes are illustrated in Figure 3.40.

Among the three, the quadtree approach is the most efficient, as it needs fewer cells to generate a mesh of equal quality near the wall, where accuracy is most critical.



(a) Quadtree mesh (b) Cartesian mesh (c) Unstructured mesh

Figure 3.40 Example of meshes generated for immersed boundary simulations

3.2.2 Cell and Face Classification

The first step in the Immersed Boundary (IB) approach lies in the fluid/solid cells identification. This comes down to determine whether the center of each cell is outside or inside the simulated surface. This task can be achieved by following the ray casting algorithm as

explained in [120] or using the winding number algorithm [121].

Next, ghost cells are identified as solid cells adjacent to at least one fluid cell. Then, the immersed boundary faces are identified by flagging the faces that have both a solid and a fluid cell neighbor. An example of the cell classification is shown in Figure 3.41. The light blue, red and dark blue cells correspond respectively to fluid, ghost and solid cells.

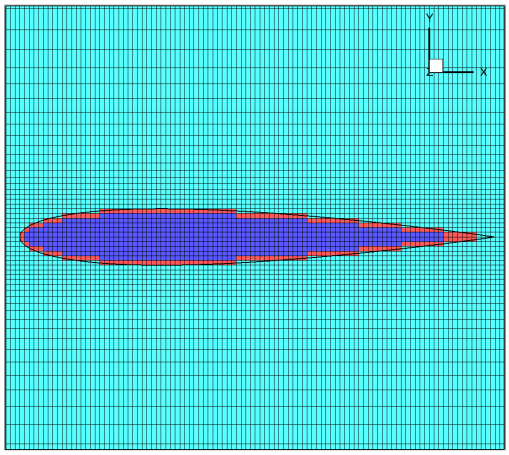


Figure 3.41 Cells classification for a NACA0012 airfoil with a cartesian mesh

3.2.3 Wall-Point Projection and Normal Vector Construction

For each immersed boundary face, a corresponding wall point must be determined. The wall point is obtained by projecting the IB face center onto each body surface face using its normal orientation. The wall point corresponds to the projection that minimizes the distance. The wall-normal vector $\mathbf{n_{IB_{Fi}}}$ is then obtained by normalizing the vector from the IB-face center to its associated wall point. This vector defines the direction in which the non-penetration condition must be enforced at the wall point. Figure 3.42 illustrates the wall-point identification process and the resulting $\mathbf{n_{IB_{Fi}}}$ vectors. Once again, the fluid cells are shown in light blue, the ghost cells in red and the solid cells in dark blue. The dashed red curve represents a section of the closed body surface. Ghost-cells center are represented by green circles, IB faces center by blue crosses, wall points by brown diamonds and $\mathbf{n_{IB_{Fi}}}$ by purple lines.

A key observation is that a single ghost cell may be associated with several IB faces, each with a different normal direction. Consequently, a single ghost-cell potential value is insufficient for enforcing all non-penetration constraints. This is known as the multiple values problem. Since CHAMPS uses an unstructured framework, this issue is resolved by storing specific face potential values for each IB face and using them within the appropriate gradient stencils.

To enforce the non-penetration condition, we first compute and store the potential associated with each IB face using Eq. (3.34) together with its corresponding $\mathbf{n_{IB_{Fi}}}$. Then, for each IB face, the component of velocity aligned with $\mathbf{n_{IB_{Fi}}}$ is removed according to Eq. (3.35), ensuring that the non-penetration direction is satisfied at the IB face center. However, the condition must be enforced at the wall point, not at the IB face center. To correct for this mismatch, a refined normal vector $\mathbf{n^*_{IB_{Fi}}}$ is introduced. This corrected normal ensures that the true non-penetration constraint is satisfied at the wall point.

To obtain $\mathbf{n^*_{IB_{Fi}}}$, the velocity gradients (∇u, ∇v) are computed at the fluid cell adjacent to the IB face the same way as the density gradient $\nabla \rho$. The velocity at the wall point is then approximated through linear extrapolation:

$$u_{WP} = u_{IBF} + \nabla u \cdot (\mathbf{x_{WP}} - \mathbf{x_{IBF}}), \tag{3.58}$$

$$v_{WP} = v_{IBF} + \nabla v \cdot (\mathbf{x_{WP}} - \mathbf{x_{IBF}}), \tag{3.59}$$

where (u_{WP}, v_{WP}) and (u_{IBF}, v_{IBF}) are the velocity components of the wall point and the IB face, respectively. Also, $\mathbf{x_{WP}}$ and $\mathbf{x_{IBF}}$ are the coordinate vector of the wall point and the IB face, respectively. The non-penetration condition is next enforced at the wall point by removing the normal component of the wall-point velocity:

$$\mathbf{u^*_{WP}} = \mathbf{u_{WP}} - \mathbf{u_{WP}} (\mathbf{u_{WP}} \cdot \mathbf{n_{IB_{Fi}}}). \tag{3.60}$$

Then, this corrected wall-point velocity is extrapolated back to the IB face:

$$u^*_{IBF} = u^*_{WP} + \nabla u \cdot (\mathbf{x_{IBF}} - \mathbf{x_{WP}}), \tag{3.61}$$

$$v^*_{IBF} = v^*_{WP} + \nabla v \cdot (\mathbf{x_{IBF}} - \mathbf{x_{WP}}). \tag{3.62}$$

For each IB face, the updated normal direction is then defined as the normal to the corrected IB-face velocity $\mathbf{u^*_{IBF}}$, providing the improved non-penetration direction $\mathbf{n^*_{IB_{Fi}}}$. This process is repeated at every iteration of the full potential IB solver. For improved stability, the corrected normal is relaxed by averaging half of its updated perpendicular direction with half of its value from the previous iteration.

Once the corrected normals are computed, the rest of the solver can act exactly the same way as explained in Section 3.1.

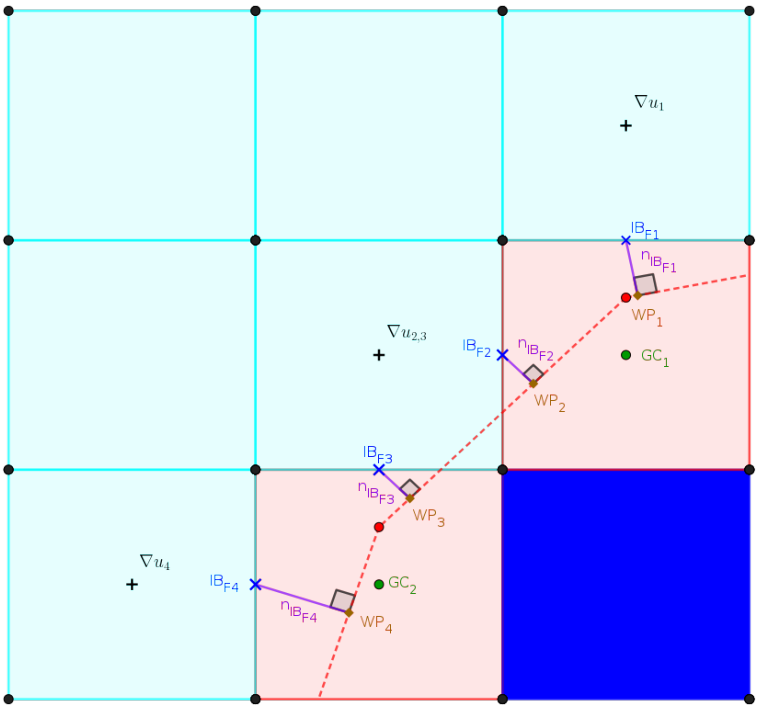


Figure 3.42 Illustration of the wall points identification procedure

3.2.4 Boundary Flux Balancing

For body-fitted meshes, the mass flux at the wall is always zero because the slip boundary condition forces the velocity to remain tangent to all wall faces (Eq. (3.32)). In contrast, for immersed boundary meshes, the fluid domain defined by the non-solid cells generally exhibits a non-zero mass flux throughout its inner boundary, as illustrated in Figure 3.43. This occurs because the face normals $\mathbf{n}_{F_i}$ of the mesh are not aligned with the corrected immersed boundary normals $\mathbf{n}^*_{\mathbf{IB_{Fi}}}$ used to enforce non-penetration.

Since the farfield boundary enforces zero net mass flux, any residual mass entering the domain through the immersed boundary face have no way to exit, violating the integral form of the governing equation (Eq. (3.1)). To prevent this, the total inward mass flux at the immersed boundary is redistributed uniformly over the farfield faces, allowing the solver to expel the same amount of mass.

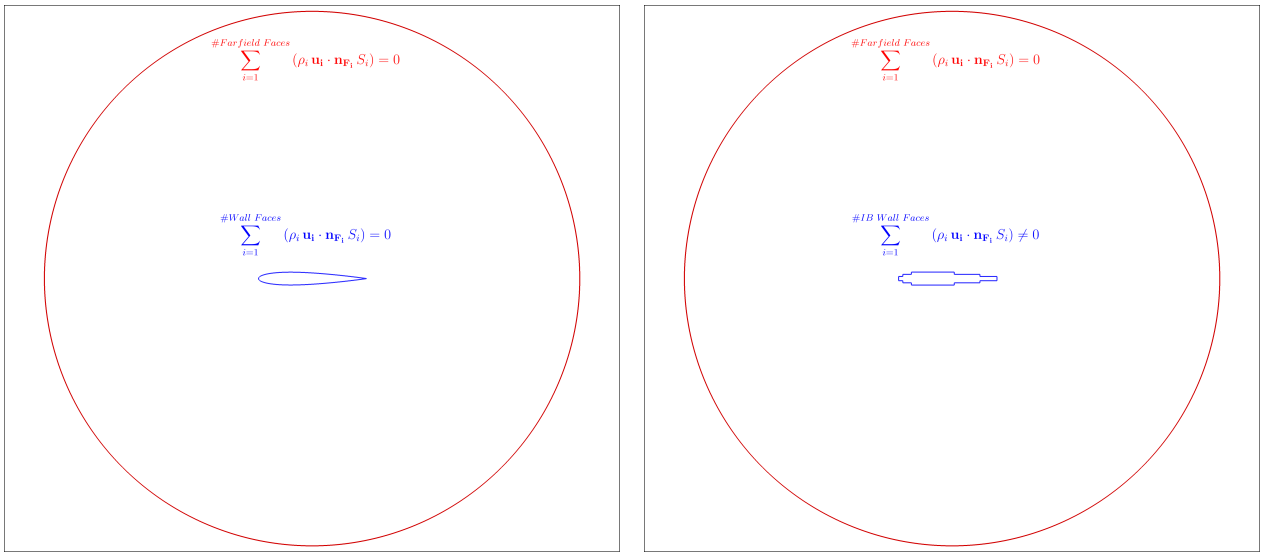
The inward mass flux at the immersed boundary is first computed by summing the normal mass flux across all immersed boundary faces:

$$mass\ flux\ IB\ Wall = \sum_{i=1}^{\#IB\ Wall\ Faces} \left(\rho_i\, \mathbf{u_i} \cdot \mathbf{n_{F_i}}\, \mathbf{S_i}\right). \tag{3.63}$$

This value is then used to define a corrective normal velocity increment $u_{\Delta,i}$ applied to each farfield face, without modifying their density:

$$u_{\Delta,i} = \frac{-mass\ flux\ IB\ Wall}{\rho_i \sum_{j=1}^{\#\,Farfield\,Faces} S_j}, \tag{3.64}$$

$$\mathbf{u_i^*} = \mathbf{u_i} + u_{\Delta,i}\,\mathbf{n_{F_i}}. \tag{3.65}$$



(a) Body-fitted mesh (b) Immersed boundary mesh

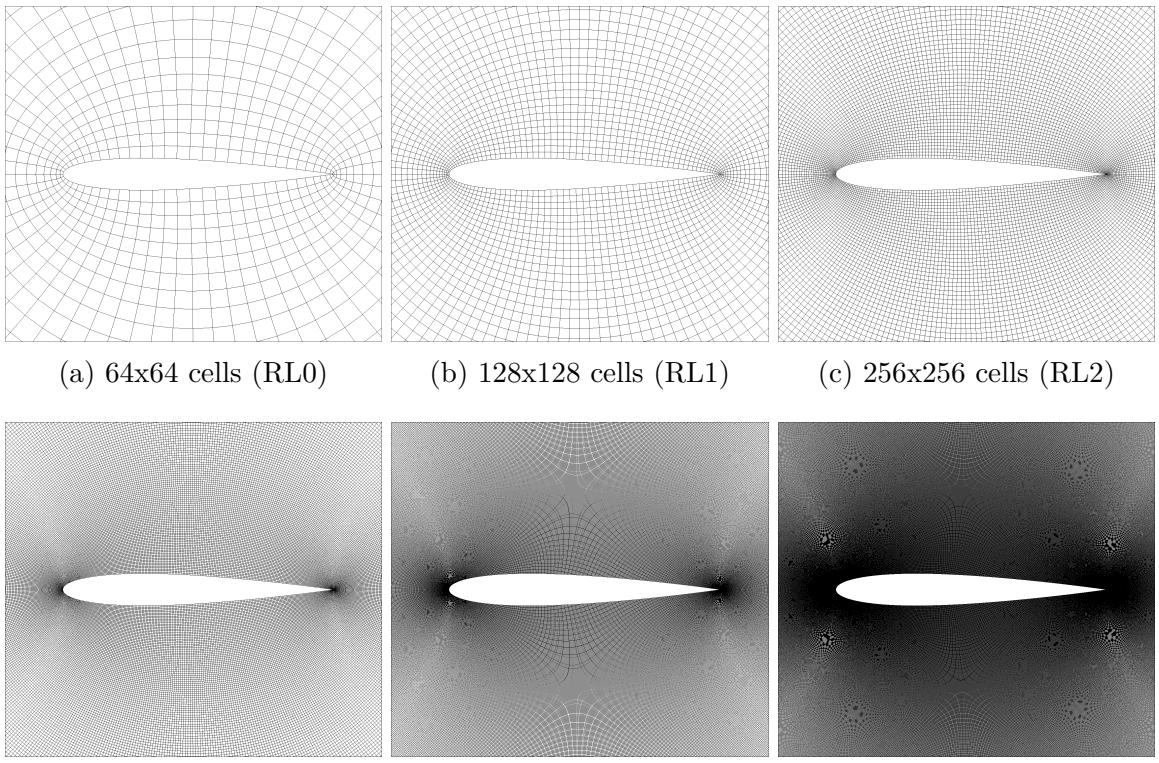
Figure 3.43 Boundary fluxes over the computational domain for the body-fitted and immersed boundary meshes

3.2.5 Results

Baseline Validation

The current Subsection compares both full potential approaches with the existing Body Fitted (BF) CHAMPS Euler solver.

For the Euler and Full-Potential BF flow simulations over the NACA0012 airfoil, the series of mesh generated by Vassberg and Jameson were used [122]. The base grid contains 64x64 cells. Successive Refinement Level (RL) were generated by uniformly subdividing each cell, doubling the resolution at each step up to a mesh size of 2048x2048. Figure 3.44 illustrates these meshes near the wall boundary.



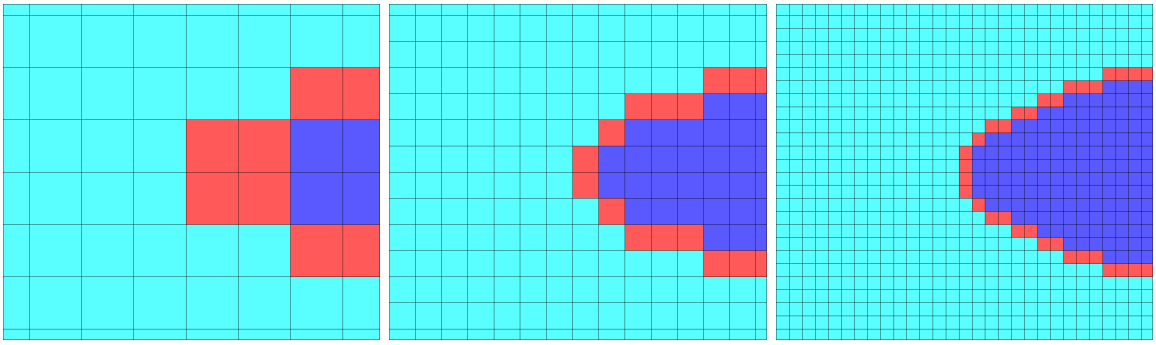
(a) 64x64 cells (RL0) (b) 128x128 cells (RL1) (c) 256x256 cells (RL2)

(d) 512x512 cells (RL3) (e) 1024x1024 cells (RL4) (f) 2048x2048 cells (RL5)

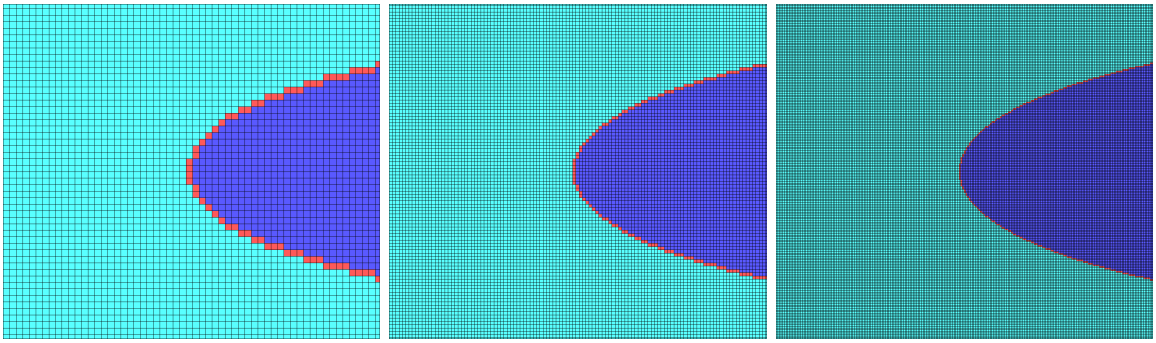
Figure 3.44 Vassberg and Jameson's series of mesh for NACA0012 airfoil

For the IB full potential solver, the base mesh (refinement level 0, containing 145x92 cells) consists of a cartesian grid with an initial cell size of 0.02 chord, a growth ratio of 1.1 and farfield boundaries positioned 10 chord lengths away from the airfoil. Higher refinement levels were generated by uniformly subdividing every cell of the previous mesh into four children. A close-up view of the leading edge portion for all refinement levels is presented in Figure 3.45.

Subsonic and transonic flows, with and without lift, were simulated for a NACA0012 airfoil. Each case was first run on the coarsest mesh and then on progressively refined meshes. For the subsonic cases, all meshes converged for both BF and IB full potential solvers, except for the IB approach with subsonic lifting configuration at refinement level 5. This loss of convergence is due to a memory issue in the mesh sequencing algorithm when extrapolating the RL4 solution to RL5, not to a deficiency of the solver itself. In contrast, for both BF and IB full potential solvers, the transonic cases showed increasing instability as the meshes were refined. As a result, IB transonic cases with and without lift converged up to refinement levels 3 and 2, respectively. For the BF full potential solver, transonic cases converged up to



(a) Refinement level 0 (RL0) (b) Refinement level 1 (RL1) (c) Refinement level 2 (RL2)



(d) Refinement level 3 (RL3) (e) Refinement level 4 (RL4) (f) Refinement level 5 (RL5)

Figure 3.45 Series of cartesian meshes for NACA0012 airfoil

the 512×512 mesh.

Transonic full potential simulations were converged on the finer meshes using the explicit iterative method described in Subsection 3.1.9. Although the implicit iterative algorithm is generally more efficient, particularly when solving with GMRES, it was found to be less stable for transonic flows since only the analytical Jacobian was implemented for the immersed boundary solver. This caused the full potential solver to become less efficient than the Euler solver for these conditions, despite expecting the opposite as explained in Subsection 3.1.11. Further work on the formulation of the analytical transonic Jacobian is necessary to mitigate this issue.

The results show good agreement with the CHAMPS BF Euler and full potential solvers when comparing the coefficient of pressure distribution presented at Figs. 3.46, 3.48, 3.50 and 3.51. The corresponding aerodynamic coefficients, summarized in Tables 3.10–3.13, also exhibit close agreement.

One difference lies in the shock position and in its strength for both full potential results for the transonic cases. This is caused by the isentropic assumption of the full potential

model and could be improved by a non-isentropic correction on the density as explained in the literature review in Subsection 2.3.4.

For lifting cases, the IB method displays a larger pressure coefficient difference between upper and lower surfaces near the trailing edge (see Fig. 3.48). This comes from the sharp airfoil trailing edge, which is difficult to represent precisely with IB facets. Because circulation is evaluated at this location, the slightly thicker effective trailing edge produced by the IB representation results in a greater circulation and a larger upper–lower surface pressure coefficient difference.

Moreover, for all cases, the IB pressure coefficient distributions tend to be less smooth than the BF approach since their solutions present small amplitude with high frequency oscillations.

For subsonic cases, Figures 3.47 and 3.49 compare residual convergence in terms of iterations and wall-clock time. When comparing meshes of similar size (2048×2048 for BF and RL4 for IB), the convergence rate of the BF and IB full potential solvers are similar, with the IB method being slightly slower. Both approaches converge faster than the Euler solver as expected. Indeed, the rule of thumb is that full potential methods converge four to five times faster than Euler solvers, since only a single equation is solved instead of 4 in 2D or 5 in 3D.

Table 3.10 Comparison of aerodynamic coefficients for BF Euler with BF and IB Full Potential methods (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$)

Coefficient	Euler BF (2048x2048)	Full Potential BF (2048x2048)	Full Potential IB RL5
C_L	-3.299e-07	-1.803e-12	-1.392e-13
C_D	1.982e-05	4.573e-04	4.468e-03
C_M	1.209e-07	4.839e-13	3.924e-14

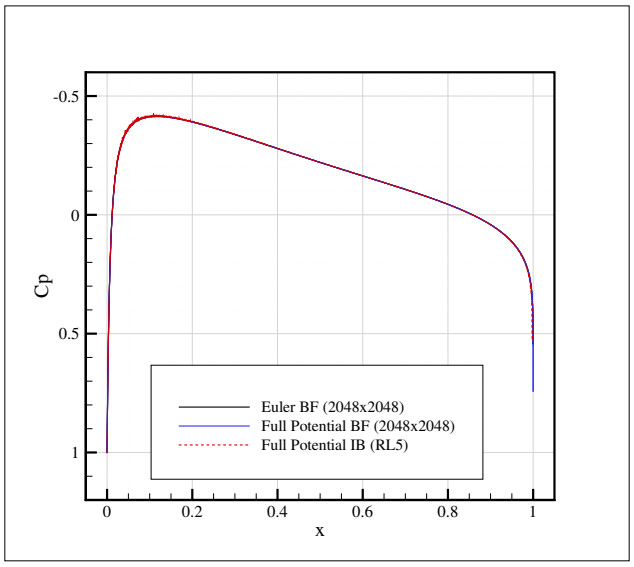
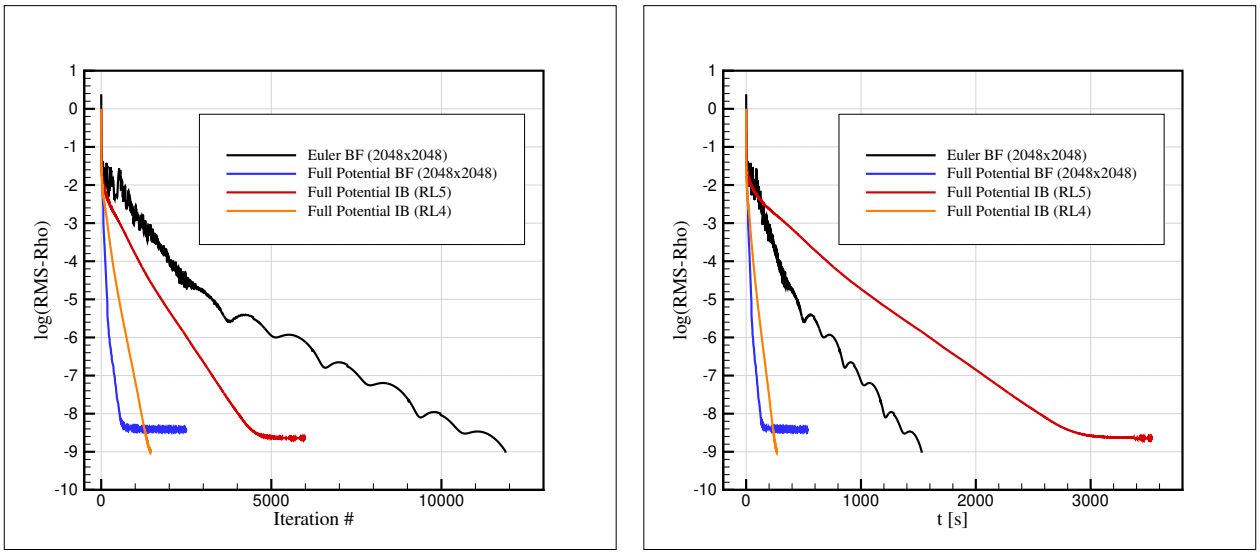


Figure 3.46 BF Euler compared to BF and IB Full Potential pressure coefficient distributions (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$).



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

Figure 3.47 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$)

Table 3.11 Comparison of aerodynamic coefficients for BF Euler with BF and IB Full Potential methods (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Coefficient	Euler BF (2048x2048)	Full Potential BF (2048x2048)	Full Potential IB RL4
C_L	1.828e-01	1.834e-01	1.817e-01
C_D	3.917e-05	4.765e-04	2.353e-04
C_M	-4.810e-02	-4.835e-02	-4.747e-02

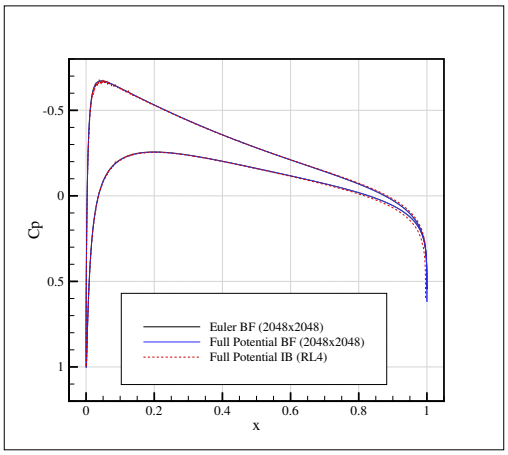
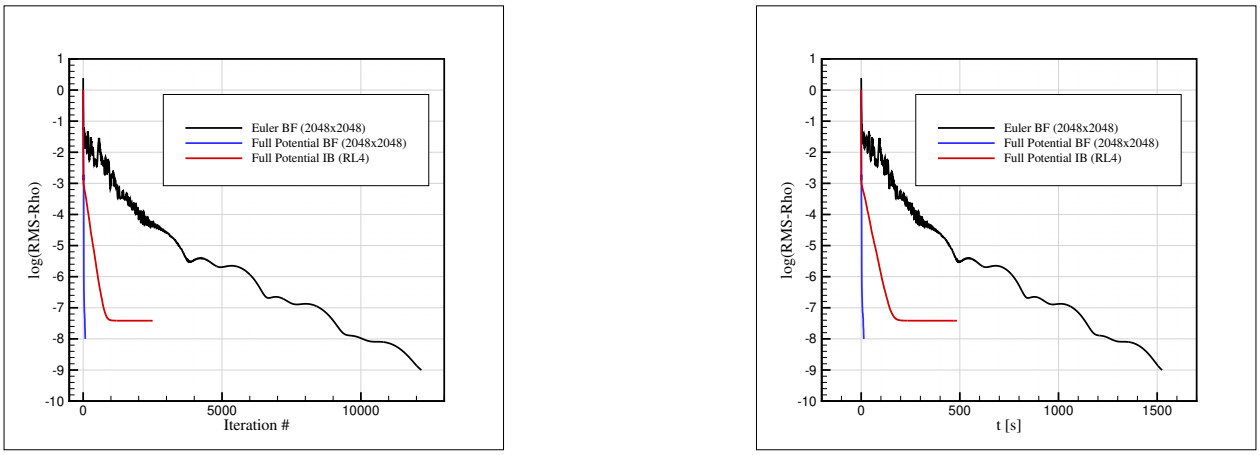


Figure 3.48 BF Euler compared to BF and IB Full Potential pressure coefficient distributions (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

Figure 3.49 Residuals of mass conservation equation for Full Potential and Euler solvers as a function of iterations and time (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Table 3.12 Comparison of aerodynamic coefficients for BF Euler with BF and IB Full Potential methods (NACA0012, $M_\infty = 0.8$, $AOA = 0.0°$)

Coefficient	Euler BF (2048x2048)	Full Potential BF (512x512)	Full Potential IB RL3
C_L	5.155e-06	1.640e-06	7.653e-12
C_D	8.352e-03	8.969e-03	7.689e-03
C_M	-4.134e-06	5.196e-06	-3.680e-12

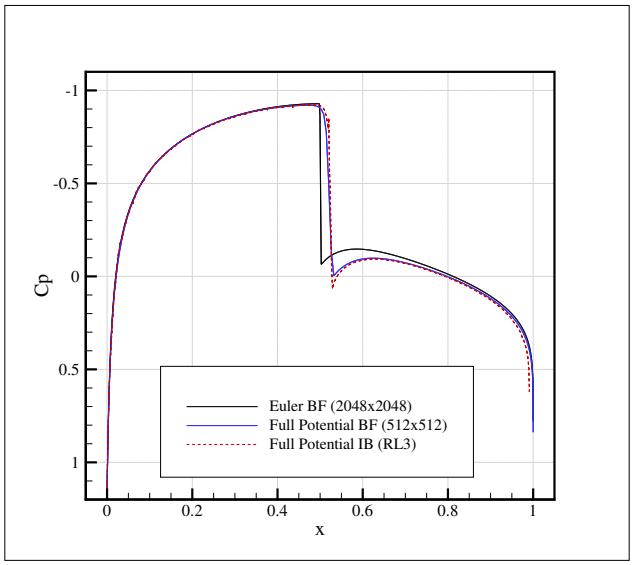


Figure 3.50 BF Euler compared to BF and IB Full Potential pressure coefficient distributions (NACA0012, $M_\infty = 0.8$, $AOA = 0.0°$).

Table 3.13 Comparison of aerodynamic coefficients for BF Euler with BF and IB Full Potential methods (NACA0012, $M_\infty = 0.75$, $AOA = 1.5°$)

Coefficient	Euler BF (2048x2048)	Full Potential BF (512x512)	Full Potential IB RL2
C_L	3.282e-01	3.873e-01	3.846e-01
C_D	6.230e-03	8.543e-03	7.951e-03
C_M	-8.270e-02	-1.029e-01	-1.020e-01

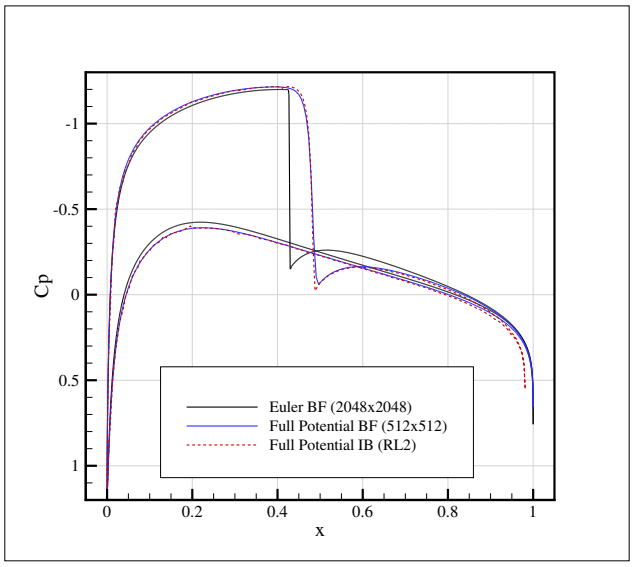


Figure 3.51 BF Euler compared to BF and IB Full Potential pressure coefficient distributions (NACA0012, $M_\infty = 0.75$, $AOA = 1.5°$)

Mesh Sequencing

For lifting cases, one difficulty encountered was a significantly slower convergence compared to non-lifting cases, particularly on fine meshes. This is caused by the fact that implicit circulation was not yet implemented at the time these cases were executed.

To mitigate this issue, a Mesh Sequencing (MS) strategy was employed. In this approach, the simulation is first run on a coarse mesh, where convergence is fast even in the presence of lift. Then, the coarse mesh solution (field of ϕ and Γ) is extrapolated to a finer mesh. By freezing the circulation during the early iterations on the refined grid, the solver gets the fast convergence behavior typical of non-lifting cases, without losing significant accuracy. Table 3.14 confirms that the aerodynamic coefficients are unchanged when MS is applied for the subsonic lifting case ($M_\infty = 0.1$, $AOA = 1.5°$) simulated on the NACA0012 1024×1024 mesh shown in Fig. 3.44e. Figure 3.52 shows that the pressure coefficient distribution is also preserved. Finally, Figure 3.53 illustrates the acceleration in convergence obtained with the Mesh Sequencing approach. With MS, log(RMS-ρ) is converged at -8 in about 250 iterations and 6 seconds. Without MS, log(RMS-ρ) is converged at about -5.4 in 100000 iterations and 2000 seconds. This strategy can be useful in the framework of steady simulations but it becomes difficult to apply for unsteady simulations, for which the implicit circulation approach should also work. This is why implicit circulation is considered superior.

Table 3.14 Comparison of aerodynamic coefficients for BF Full Potential with and without using Mesh Sequencing (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Coefficient	Full Potential	Full Potential using MS
C_L	1.829e-01	1.829e-01
C_D	9.419e-04	9.419e-03
C_M	-4.825e-02	-4.827e-02

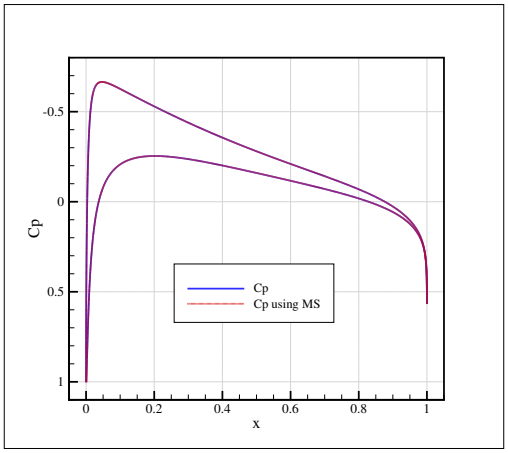
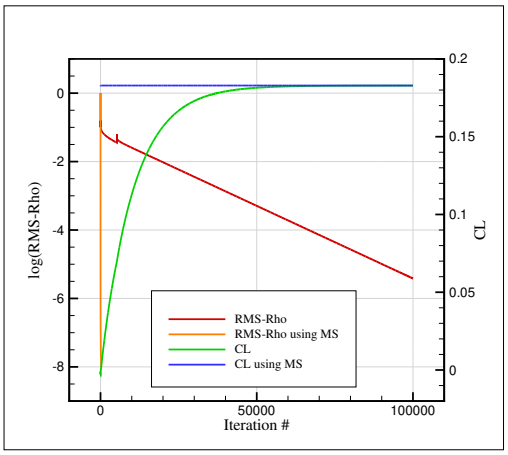
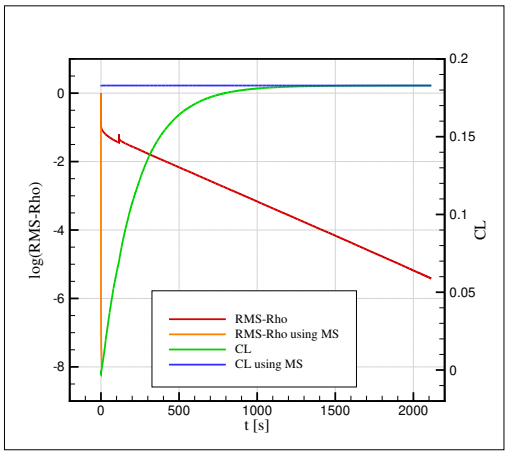


Figure 3.52 BF Full Potential pressure coefficient distributions with and without using Mesh Sequencing (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)



(a) log(RMS-ρ) vs number of iterations



(b) log(RMS-ρ) vs time [s]

Figure 3.53 Residuals of mass conservation equation for Full Potential with and without using Mesh Sequencing as a function of iterations and time (NACA0012, $M_\infty = 0.1$, $AOA = 1.5°$)

Boundary Flux Balancing

During the development of the immersed boundary method, it was observed that the residual often stagnated and failed to fully converge, even after a large number of iterations. This behavior was undesirable and did not occur in the body-fitted cases. By applying the Boundary Flux Balancing (BFB) technique described in Subsection 3.2.4, full convergence of the residuals is obtained, as demonstrated in Figure 3.55. Another important feature is that BFB does not alter the solution. The pressure coefficient distributions shown in Figure 3.54 are identical with and without applying the method.

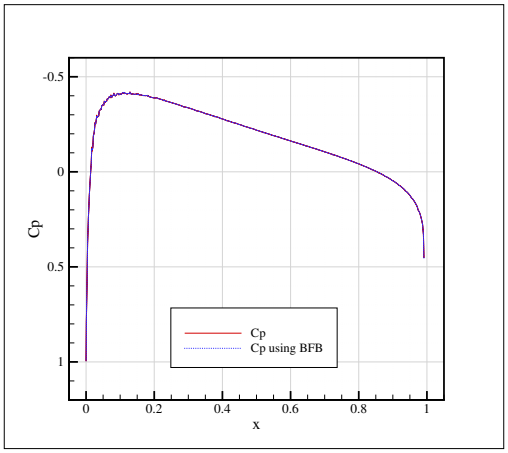


Figure 3.54 IB Full Potential pressure coefficient distributions with and without Boundary Flux Balancing (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$)

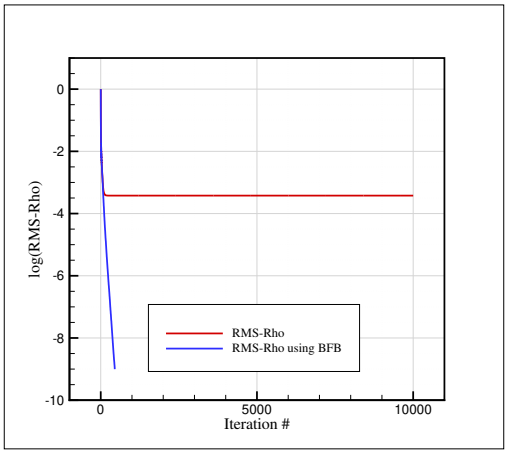
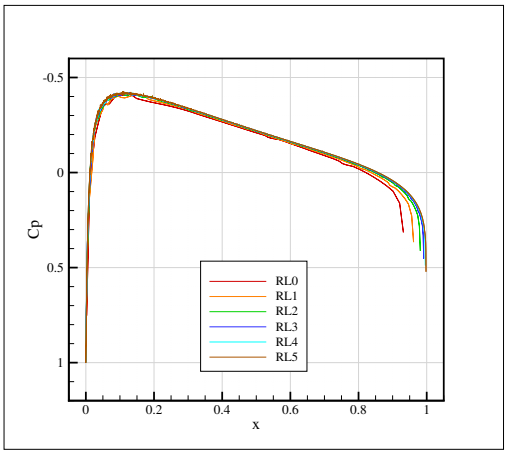


Figure 3.55 Residuals of mass conservation equation for Full Potential with and without Boundary Flux Balancing as a function of iterations (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$)

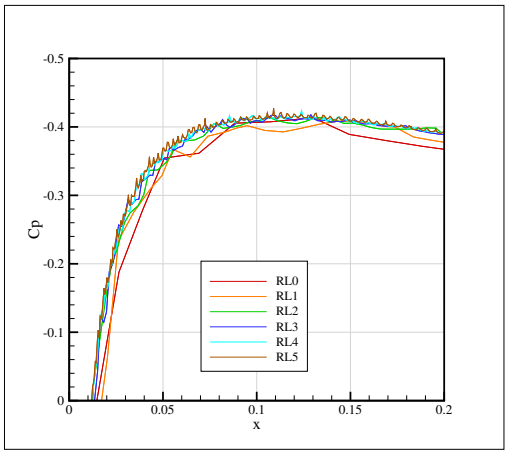
Mesh Refinement Study

The pressure coefficient distributions for all refinement levels of the IB NACA0012 airfoil are presented in Figure 3.56. Globally, the solutions appear to converge as the mesh is refined. However, closer inspection of specific regions such as the leading edge, the trailing edge and the pressure coefficient peak, reveals small amplitude and high-frequency oscillations in the solution, which persist even at the finest resolution.

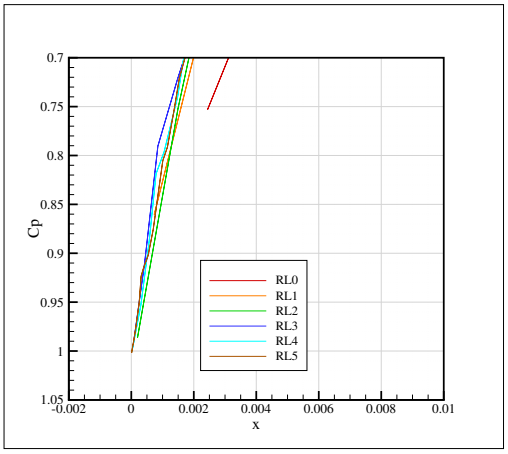
The residual convergences for all refinement levels are shown in Figure 3.57. As expected, finer meshes exhibit slower convergence.



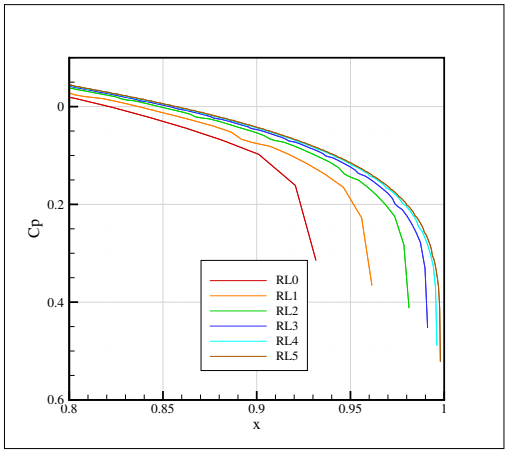
(a) Global distribution



(b) Peak distribution

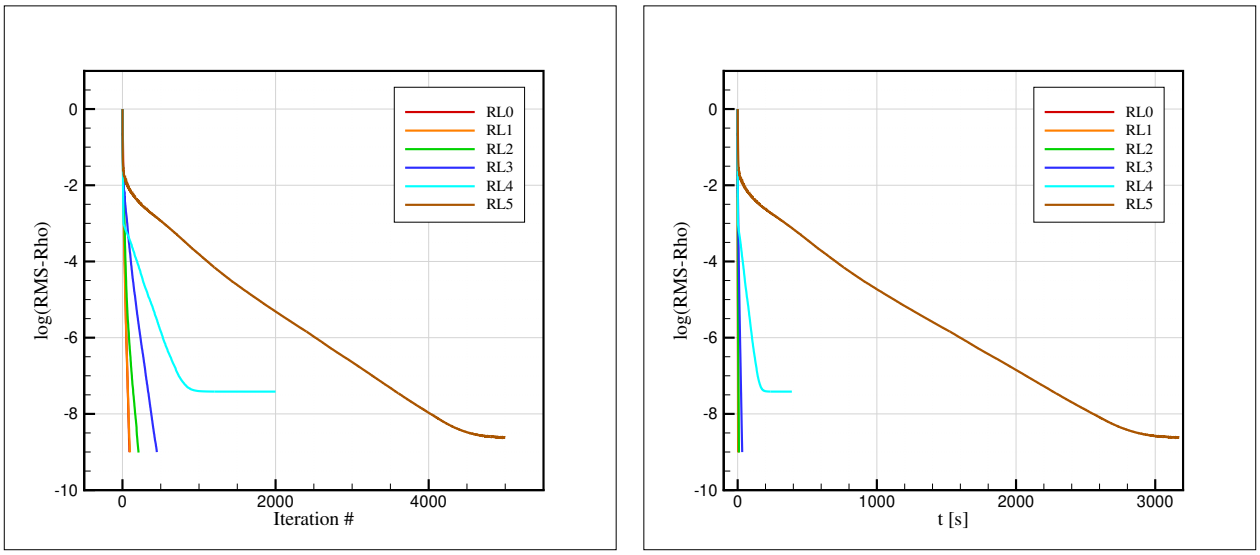


(c) LE distribution



(d) TE distribution

Figure 3.56 IB Full Potential pressure coefficient distributions for all refinement levels (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$)



(a) log(RMS-ρ) vs number of iterations

(b) log(RMS-ρ) vs time [s]

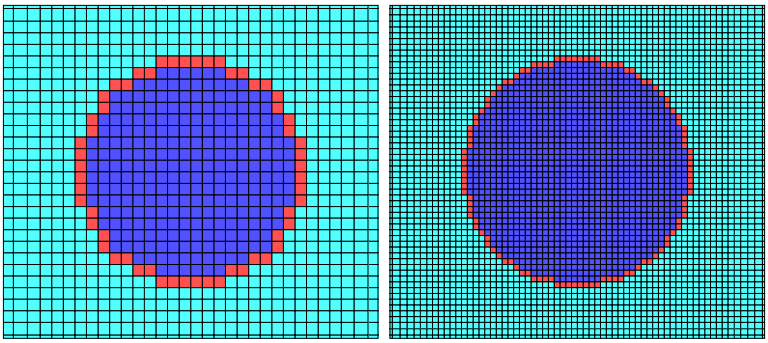
Figure 3.57 Residuals of mass conservation equation for IB Full Potential solver for all refinement levels as a function of iterations and time (NACA0012, $M_\infty = 0.1$, $AOA = 0.0°$)

Discretization Error Verification

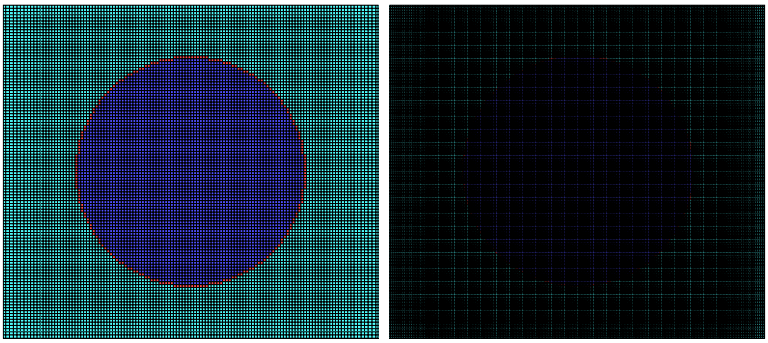
The discretization error verification assessment for the IB approach is done the same way as the BF approach as explained in Subsection 3.1.11. The canonical case of an incompressible flow over a cylinder is chosen and the numerical error is obtained through the exact solution.

The immersed boundary meshes for this case were generated in-house using CHAMPS. They consist of a cartesian base grid with a minimum cell size of 0.1 radius, a growth ratio of 1.1 and outer boundaries positioned at 10 radius. Each subsequent refinement level was produced by doubling the resolution of the previous mesh. Close-up views of the solid region of these meshes are presented in Figure 3.58.

For the immersed boundary approach, the observed convergence rate is close to 1. There are high-frequency oscillations in the solution that persist even under mesh refinement as shown in the previous Mesh Refinement Study Subsection 3.2.5. Several strategies were tested to mitigate these oscillations, such as extending the stencils used to compute $\nabla\phi$, ∇u and ∇v, but none were successful. Demonstrating correctness for the IB method is inherently more challenging, as additional sources of error that do not exist in the body-fitted formulation appear. Nevertheless, because the solutions converge toward the correct physical behavior across many test cases, it is likely that the IB implementation is fundamentally correct. Still, further development is required to achieve second-order convergence in practice. The wall



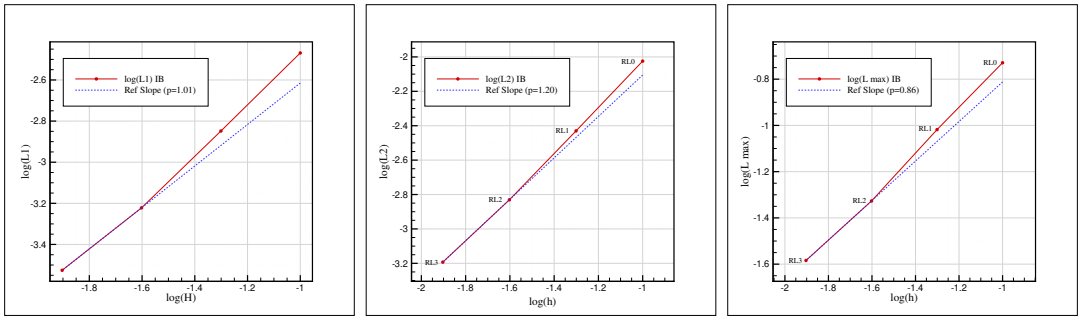
(a) Refinement Level 0 (RL0) (b) Refinement Level 1 (RL1)



(c) Refinement Level 2 (RL2) (d) Refinement Level 3 (RL3)

Figure 3.58 Immersed boundary cylinder series of meshes

point velocity reconstruction procedure is probably at cause. It includes the gradient of the velocities which come from the gradient of the potential, so this is possibly where accuracy is lost. An alternative would be to use $k = 2$ in the K-Exact-Least Square gradient method, but it would require a larger stencil which could lead to a parallelization problem with CHAMPS.



(a) $log(L_1)$ vs $log(h)$ (b) $log(L_2)$ vs $log(h)$ (c) $log(L_\infty)$ vs $log(h)$

Figure 3.59 Error metrics as a function of the minimum cell size $log(h)$ using the immersed boundary approach

3.3 Coupling to a Boundary Layer Solver

3.3.1 Laminar Boundary Layers

Laminar boundary layers are computed using the method proposed by Thwaites [123]. This approach introduces two additional parameters: λ, which represents a pressure gradient parameter and $l(\lambda)$, a length parameter expressed as a function of λ.

The parameter λ is first defined as a function of the momentum thickness θ:

$$\lambda(\theta) = \frac{\theta^2}{\nu}\frac{du_e}{dx}. \tag{3.66}$$

The skin-friction coefficient c_f can also be expressed in terms of θ and $l(\lambda)$:

$$\frac{c_f}{2} = \frac{\nu}{u_e\theta} l(\lambda). \tag{3.67}$$

Based on similarity solutions and experimental observations, Thwaites determined that the shape factor H and the function $l(\lambda)$ can be approximated by piecewise expressions of λ:

$$H(\lambda) = \begin{cases} \frac{0.0731}{0.14+\lambda} + 2.088 & \text{if } \lambda \in [-0.1, 0] \\ 2.61 - 3.75\lambda + 5.24\lambda^2 & \text{if } \lambda \in [0, 0.1] \end{cases} \tag{3.68}$$

and

$$l(\lambda) = \begin{cases} 0.22 + 1.402\lambda + \frac{0.018\lambda}{0.107+\lambda} & \text{if } \lambda \in [-0.1, 0] \\ 0.22 + 1.57\lambda - 1.8\lambda^2 & \text{if } \lambda \in [0, 0.1]. \end{cases} \tag{3.69}$$

With these relations, the von Kármán integral momentum equation (Eq. 2.34) can be solved. Substituting Eqs. (3.66) and (3.67) into Eq. (2.34) and rearranging the terms yields

$$\frac{u_e}{\nu}\frac{d\theta^2}{dx} = 2\left[l(\lambda) - (H(\lambda) + 2)\lambda\right] = F(\lambda). \tag{3.70}$$

Thwaites further simplified this relation by expanding the expressions for $l(\lambda)$ and $H(\lambda)$ in $F(\lambda)$ and by neglecting second or higher order terms, assuming that λ remains small and bounded within the interval $[-0.1, 0.1]$. This leads to the approximation

$$F(\lambda) \approx 0.45 - 6\lambda. \tag{3.71}$$

Using this approximation, Eq. 3.70 can be rewritten in the following simplified form:

$$\frac{1}{\nu}\frac{d(u_e^6\theta^2)}{dx} = 0.45u_e^5, \tag{3.72}$$

which can be easily solved numerically. The equation is integrated along the airfoil surface from the stagnation point toward the trailing edge on both the intrados and extrados. The integration takes place until the boundary layer either transitions to turbulence or separates. The initial condition is provided by the stagnation point, where the momentum thickness θ is equal to zero. Furthermore, at the stagnation point, the form factor H is assumed to be equal to 2.6 which is the result for a laminar boundary layer over a flat plate. Regarding separation, it is assumed to happen for a laminar boundary layer if $\lambda < -0.09$.

3.3.2 Turbulent Boundary Layers

In the case of turbulent boundary layers, the system of equations is closed using additional empirical relations. The first equation is derived from the entrainment formulation proposed by Head [124], which can be written as

$$H^*\frac{d\theta}{dx} + \frac{H^*\theta}{u_e}\frac{du_e}{dx} + \theta\frac{dH^*}{dx} = C_E, \tag{3.73}$$

where C_E is the lag-entrainment ratio coefficient and $H^* = \frac{\delta-\delta^*}{\theta}$ is a parameter involving the boundary layer thickness δ. Since the latter is not well defined in practice, the formulation used in this work relies instead on an empirical relation that removes this dependency. The resulting equation is given by

$$\theta\frac{dH}{dx} = -H(H-1)(3H-1)\frac{\theta}{u_e}\frac{du_e}{dx} + H(3H-1)\frac{c_f}{2} - (3H-1)^2\frac{0.0056}{2}\left(\frac{u_e\theta}{\nu}\right)^{-1/6}. \tag{3.74}$$

A second relation is required to close the system. This is provided by the correlation proposed by Ludwieg and Tillmann [125], which expresses the skin-friction coefficient c_f as a function of the shape factor H and the Reynolds number based on the momentum thickness Re_θ:

$$c_f = 0.246 \times 10^{-0.678H}\, Re_\theta^{-0.268}. \tag{3.75}$$

In the context of aero-icing simulations, where the surface is assumed to have a certain roughness, the formulations proposed by [6,31] are adopted. Thus, the skin-friction coefficient

is expressed as a function of the momentum thickness θ and the surface roughness equivalent sand-grain roughness k_s:

$$c_f = \frac{0.336}{\left[\ln\left(\frac{864\,\theta}{k_s} + 2.568\right)\right]^2}, \tag{3.76}$$

where k_s is a user-defined parameter, typically set to a value of $c/1000$, with c being the chord length.

Now, with Eqs. (2.34) and (3.74) combined with (3.75) or (3.76) depending on which skin friction formula is chosen, the turbulent boundary layer equations can be solved numerically in a manner similar to laminar boundary layers. The integration is performed from the location where the flow transitions from laminar to turbulent up to either the trailing edge or the point of boundary layer separation, whichever occurs first.

The transition point may either be specified by the user or predicted using dedicated criteria, which will be described in Subsection 3.3.4. The initial condition for θ is taken as the final value obtained in the laminar region, while the initial shape factor is set to $H = 1.4$, corresponding to the exact value for a turbulent boundary layer over a flat plate.

Finally, turbulent separation is assumed to occur when either $c_f \leq 0$ or when the shape factor exceeds $H \geq 3.5$.

3.3.3 Boundary Layer Transition

The transition from laminar to turbulent flow can either be specified by the user at a particular location on the airfoil or predicted using established criteria. For purely aerodynamic simulations, Michel's criterion [126] is employed, which assumes that transition occurs when

$$Re_\theta \geq 1.174\left(1 + \frac{22400}{Re_s}\right) Re_s^{0.46}, \tag{3.77}$$

where $Re_s = \frac{u_e s}{\nu}$ is the Reynolds number based on the curvilinear distance along the airfoil surface.

In aero-icing simulations, transition happens very close to the leading edge because of the roughness. Therefore, it is predicted using a criterion based on Mochizuki [127], which defines a threshold Reynolds number based on k_s:

$$Re_k = \frac{u_e k_s}{\nu} > 600. \tag{3.78}$$

3.3.4 Heat Transfer Calculation

In aero-icing simulations, the Heat Transfer Coefficient (HTC) is required to carry out the thermodynamic calculations. To compute it, the thermal boundary layer must be resolved or approximated. In this work, the formulations proposed by [6, 31] are adopted.

In the laminar region, the heat-transfer coefficient is evaluated as

$$HTC = \lambda_t \frac{u_e^{1.435}\nu}{3.4176} \left(\int_0^s u_e(s)^{1.87}\, ds \right)^{-1/2}, \tag{3.79}$$

where λ_t is the air thermal conductivity.

For the turbulent region, the heat transfer is computed using

$$HTC = \rho_e \tilde{C}_P u_e \frac{c_f/2}{Pr_t + \sqrt{c_f/2}(1.92 k_s^{+-0.45} Pr^{-0.8})^{-1}}, \tag{3.80}$$

where $\tilde{C}_P$ is the specific heat capacity of air at constant pressure, Pr is the Prandtl number which is set to 0.707, Pr_t is the turbulent Prandtl number which is set to 0.9 and k_s^+ is the non-dimensional sand-grain roughness defined as

$$k_s^+ = \frac{u_e \sqrt{c_f/2} k_s}{\nu}. \tag{3.81}$$

Regarding the heat transfer coefficient at the stagnation point, it is set to:

$$HTC_{stagn} = \lambda_t \frac{\sqrt{0.246\, Re \frac{c}{u_\infty} \frac{du_e}{ds}}}{c}. \tag{3.82}$$

3.3.5 Boundary Layer–Full Potential Coupling Validation

To evaluate the accuracy of the proposed boundary layer method in the context of two-dimensional aero-icing simulations, a test case was performed on a NACA0012 airfoil at low Mach number and zero angle of attack. Several inviscid–boundary layer coupling strategies were considered, including: weak boundary layer-BF FP coupling, direct boundary layer-BF FP coupling, weak boundary layer-BF Euler coupling, direct boundary layer-BF Euler coupling and a weak boundary layer-IB FP coupling. The results obtained with these different coupling strategies were compared both among themselves and with the boundary layer solver results reported by Bayeux [6].

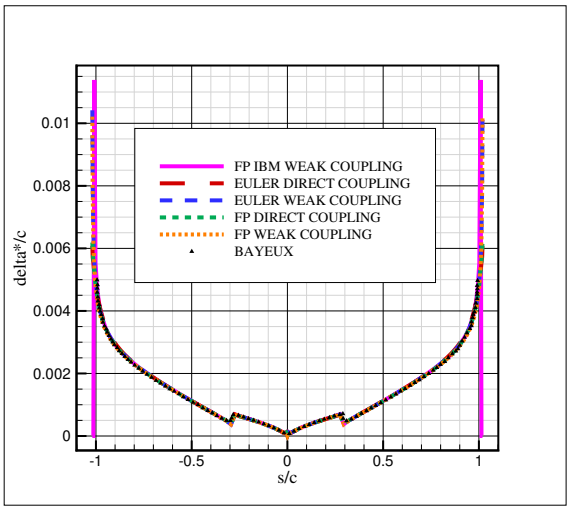
The simulations were carried out for a NACA0012 airfoil with a chord length of $c = 0.5\,\mathrm{m}$ under the following freestream conditions: $M_\infty = 0.15$, $AOA = 0.0°$, $P_\infty = 80000\,\mathrm{Pa}$ and $T_\infty = 263\,\mathrm{K}$. For comparison with the reference results, the transition location was forced at $s/c = 0.3$.

For this configuration, the body-fitted inviscid solvers used the NACA0012 128×128 mesh shown in Fig. 3.44b. Regarding the IB FP solver, the simulation was carried on a cartesian grid with an initial cell size of 0.003 chord and a growth rate of 1.1 up to the outer boundary located at 20 chords, away from the airfoil. This cartesian grid consists in total of a $537 \times 186 = 99882$ quadrilateral cells.

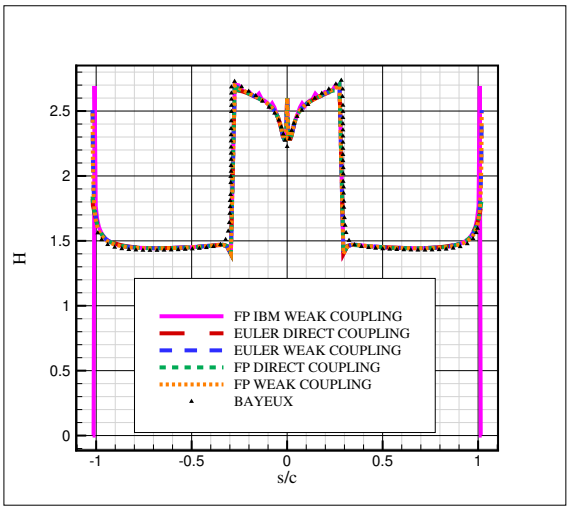
The results of the boundary layer method are presented in Fig. 3.60, which shows the distributions of δ^*/c, H, c_f and HTC along the curvilinear coordinate s/c. For the dynamic boundary layer quantities (δ^*/c, H and c_f), good agreement is observed between all coupling strategies, with the exception of the trailing-edge region for the weak IB FP coupling. In the latter case, significant oscillations appear in the projected surface solution near the trailing edge, leading the boundary layer solver to predict an artificial numerical separation. However, this issue does not affect aero-icing simulations, since the region of interest for icing phenomena is the leading edge, which is far from the trailing-edge region.

The distribution of the heat-transfer coefficient HTC along s/c also shows good agreement between all coupling strategies within the laminar portion of the boundary layer. In the turbulent region, the different approaches remain consistent with each other but appear slightly scaled down comparatively to the reference solution reported by Bayeux. Since the turbulent heat-transfer relation in Eq. (3.80) depends on three variables (c_f, u_e and ρ_e), which already show close agreement between the different simulations, the discrepancy in HTC is likely related to differences in the Prandtl numbers used in the model. The turbulent heat-transfer coefficient is particularly sensitive to the turbulent Prandtl number Pr_t. By adjusting this parameter, it was possible to approach the results reported by Bayeux, who does not specify the values used for Pr and Pr_t. Nevertheless, the simulations in this work were performed using the standard values $Pr = 0.707$ and $Pr_t = 0.9$, just like the other solutions coming from the boundary layer solver used in this thesis.

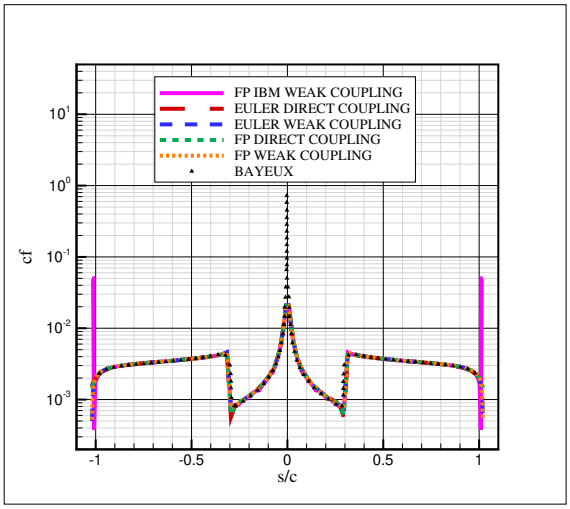
Since the results for all coupling strategies were deemed satisfactory, even for weak boundary layer coupling, it was assumed that these simpler strategies could be used to perform icing simulations.



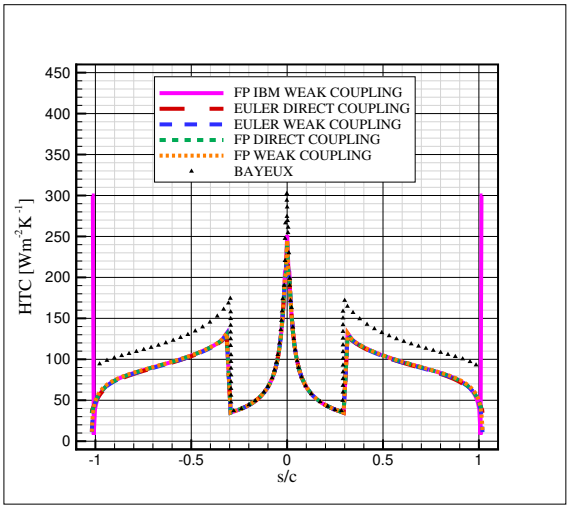
(a) δ^*/c vs s/c



(b) H vs s/c



(c) c_f vs s/c



(d) HTC vs s/c

Figure 3.60 Boundary layer results for the distributions of δ^*, H, c_f and HTC (NACA0012, $M_\infty = 0.15$, $AOA = 0.0°$)

CHAPTER 4 AERO-ICING SIMULATIONS

To evaluate the capability of the body-fitted and immersed boundary full potential aerodynamic solvers developed in Chapter 3 within the framework of numerical aero-icing simulations, two two-dimensional test cases 241 and 242 from the first Ice Prediction Workshop (IPW1) were considered. Case 241 corresponds to a rime ice accretion simulation, whereas case 242 represents a glaze ice accretion simulation. The conditions associated with these cases are summarized in Table 4.1 [1].

Table 4.1 Flow and icing conditions for Cases 241 and 242 - [1]

Case	241	242
Type of ice	Rime	Glaze
Airfoil	NACA23012	NACA23012
Chord c [m]	0.4572	0.4572
AOA [°]	2	2
Mach number M_∞ [-]	0.325	0.31
Reynolds number Re [-]	3.8×10^6	3.4×10^6
Static pressure P [Pa]	92528	92941
Static temperature T [°C]	-23.0	-7.1
Spray time [min]	5	5
Median volumetric diameter MVD [μm]	30	15
Liquid water content LWC [g/m^3]	0.42	0.81

These cases were simulated using four different aerodynamic solvers: the body-fitted full potential (FP BF) solver and the immersed boundary full potential (FP IB) solver developed in Chapter 3, as well as the body-fitted Euler and RANS solvers available in CHAMPS. Each aerodynamic approach has different requirements within the aero-icing simulation framework, as illustrated in Fig. 1.5.

Since the full potential and Euler solvers are inviscid, they must be coupled with a boundary layer solver in order to estimate the skin-friction coefficient c_f and the HTC. In contrast, the RANS solver inherently predicts these quantities through its turbulence model.

For the FP IB approach, the droplet solver must also operate within an immersed boundary framework. For this purpose, an Eulerian penalization method is employed, following the approach described in [35,36]. For the body-fitted aerodynamic solvers, a body-fitted Eulerian

droplet approach is used.

Regarding the thermodynamic calculations, all solvers are coupled with an iterative Messinger formulation. The geometry evolution is handled using the hyperbolic geometric solver available in CHAMPS.

Mesh handling differs between immersed boundary and body-fitted approaches. For the FP IB solver, the mesh is generated only once at the beginning of the simulation. In contrast, body-fitted approaches require mesh regeneration during multi-layer icing simulations. For the FP BF and Euler BF solvers, mesh deformation is performed using a Radial Basis Function (RBF) method, whereas the RANS solver employs a hyperbolic mesh generator.

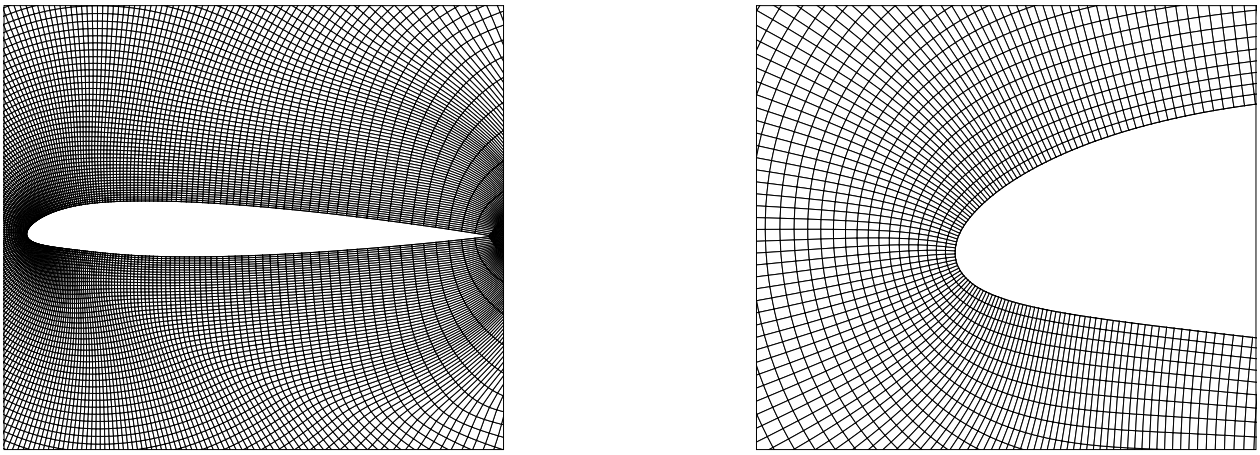
A summary of the numerical approaches used for each module of the aero-icing framework is provided in Table 4.2.

Table 4.2 Summary of the solvers used for each module of the aero-icing framework

Module	**FP IB**	**FP BF**	**Euler BF**	**RANS BF**
Aerodynamic solver	FP IB	FP BF	Euler BF	RANS BF
HTC **and** c_f **prediction**	Weak BL coupling			Intrinsic
Mesh generation	Quadtree [119]	RBF	RBF	Hyperbolic
Droplet solver	Penalized Eulerian	Eulerian		
Thermodynamic solver	Iterative Messinger			
Geometry update	Hyperbolic			

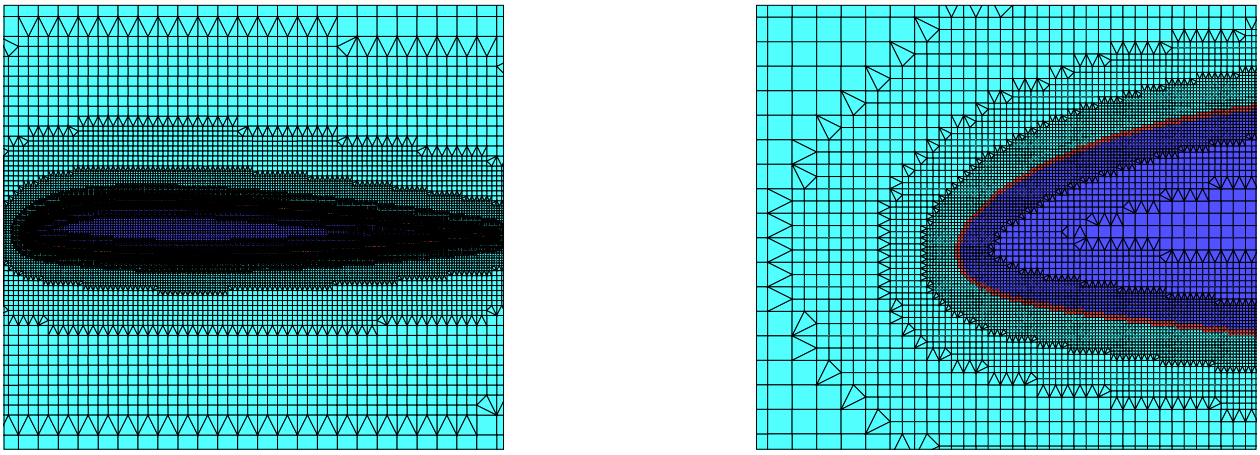
The initial mesh of the NACA23012 airfoil used to initialize the body-fitted aerodynamic solvers for cases 241 and 242 is shown in Fig. 4.1. The O-type mesh contains $258 \times 256 = 66048$ quadrilateral cells. Near the leading edge, the mesh resolution corresponds to approximately $\Delta x/c \approx 1.3 \times 10^{-3}$.

The mesh used to perform the single-layer icing simulations with the FP IB aerodynamic solver for cases 241 and 242 is shown in Fig. 4.2. The quadtree mesh was generated using the MixedQuadTree program [119] and consists of 63420 cells, including 54348 quadrilaterals and 9072 triangles. The finest level of refinement corresponds to a cell size of approximately $\Delta x/c \approx 1.3 \times 10^{-3}$.



(a) Airfoil view (b) Leading edge view

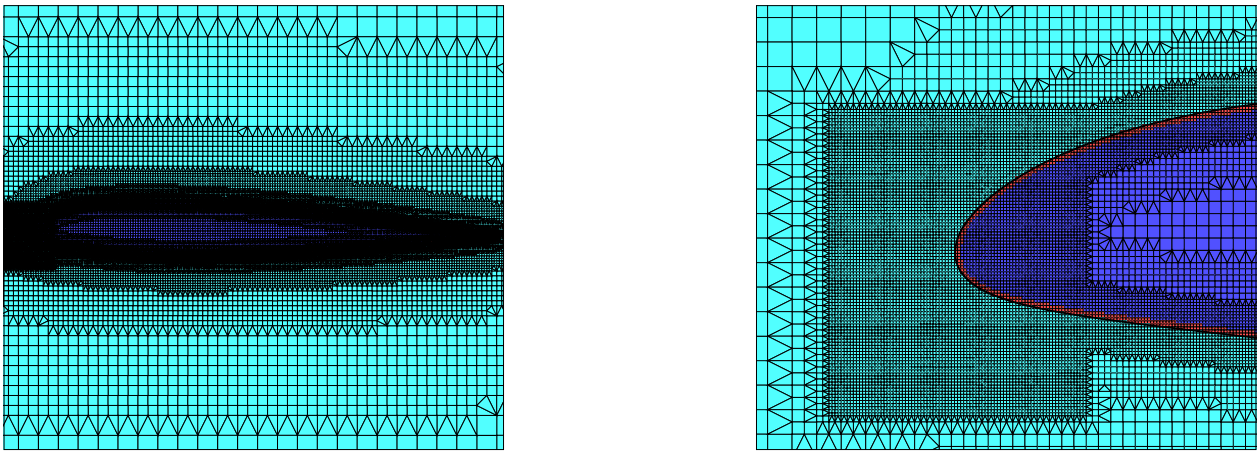
Figure 4.1 Close-up view of the NACA23012 body-fitted mesh



(a) Airfoil view (b) Leading edge view

Figure 4.2 Close-up view of the NACA23012 immersed boundary mesh for single-layer icing simulations

The mesh used for the immersed boundary multi-layer icing simulations is shown in Fig. 4.3. This mesh is similar to the single-layer IB mesh, but the region of maximum refinement has been extended into a rectangular zone around the leading edge to ensure that it contains the intermediate geometry evolutions during ice accretion. Therefore, the finest refinement level also corresponds to a cell size of approximately $\Delta x/c \approx 1.3 \times 10^{-3}$. The mesh contains a total of 69064 cells, including 59766 quadrilaterals and 9298 triangles.



(a) Airfoil view

(b) Leading edge view

Figure 4.3 Close-up view of the NACA23012 immersed boundary mesh for multi-layer icing simulations

4.1 Case 241 of the IPW1

Case 241 corresponds to a low-temperature icing condition, resulting in a rime ice accretion phenomena in which the impinging water droplets freeze immediately upon contact with the NACA23012 airfoil. Consequently, the ice accretion process is primarily governed by the distribution of the droplet collection efficiency over the airfoil surface, while the influence of the HTC is comparatively less significant.

The distributions of the HTC near the stagnation region over the clean airfoil obtained with the four aerodynamic solvers are compared with the results from other participants of the IPW1 in Fig. 4.4. Significant discrepancies can be observed among the different participants. The RANS solver produces an HTC distribution that lies within the average range of the reported results. In contrast, the three inviscid solvers (FP IB, FP BF and Euler BF) provide nearly identical predictions that appear systematically lower than those reported by most participants, including IGLOO2D, which is also based on an inviscid formulation.

In the turbulent region of the boundary layer, the predicted HTC values are approximately 100 $Wm^{-2}K^{-1}$ lower than those obtained by other participants. However, in the vicinity of the stagnation point and within the laminar region, the discrepancy increases to approximately 300 $Wm^{-2}K^{-1}$, which may significantly affect the thermodynamic predictions.

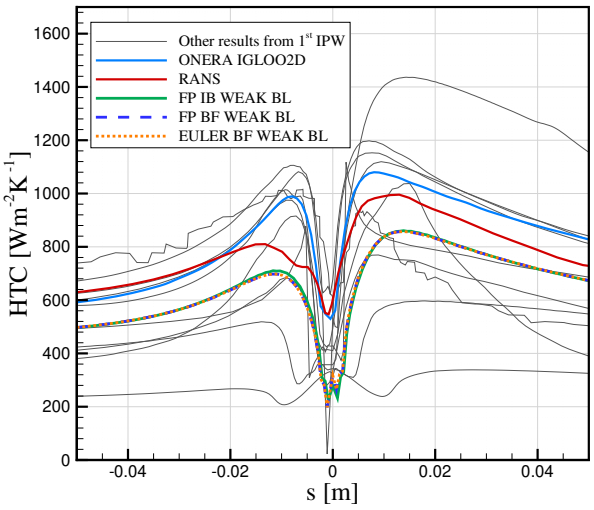


Figure 4.4 Comparisons of *HTC* distributions near the stagnation point for the FP IB, FP BF, Euler BF and RANS BF solvers with other IPW1 participants (Case 241)

The distributions of the droplet collection efficiency β along the clean airfoil surface are presented in Fig. 4.5 as a function of the vertical coordinate. In contrast with the *HTC* results, the β distributions predicted by all aerodynamic solvers show very good agreement, indicating that the droplet trajectory predictions are consistent across the different approaches.

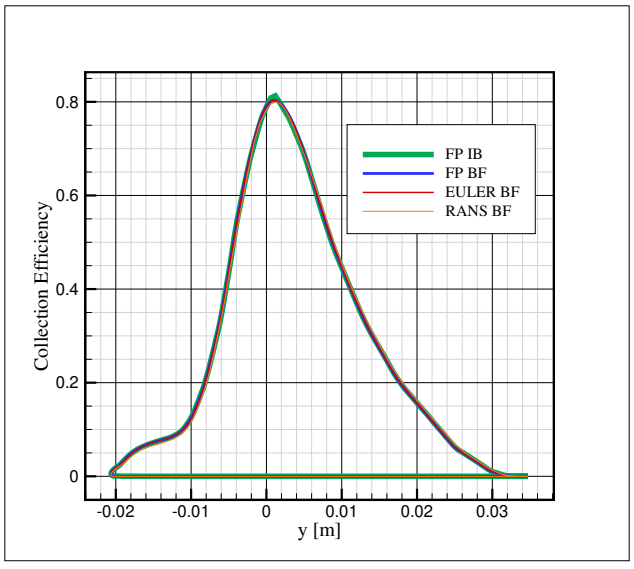
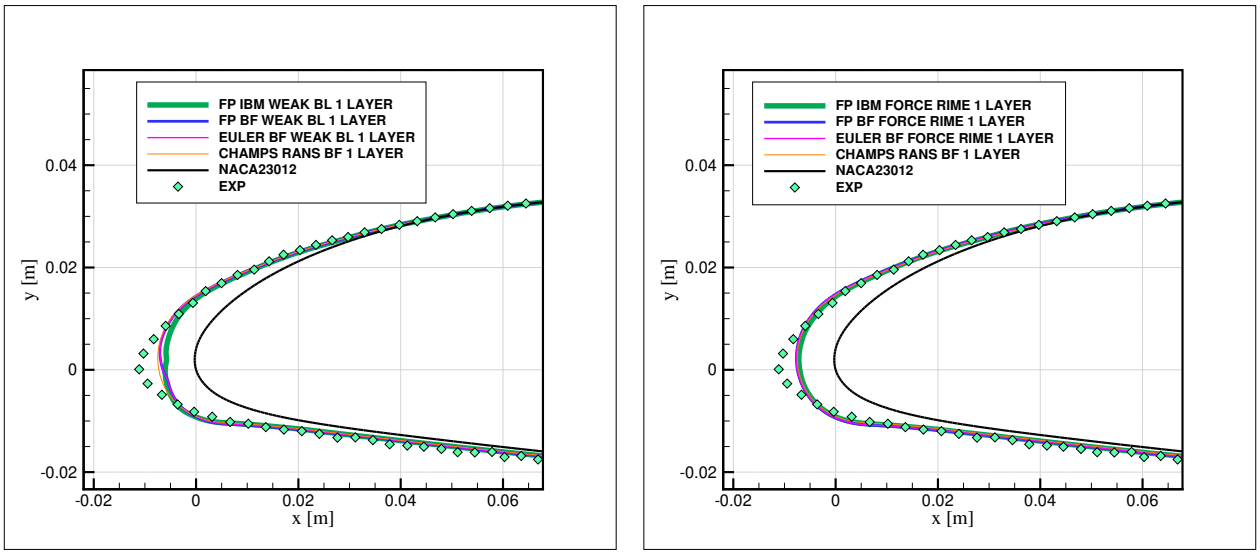


Figure 4.5 Collection efficiency (β) distributions according to the vertical coordinate for the FP IB, FP BF, Euler BF and RANS BF solvers (Case 241)

The ice shapes predicted from single-layer icing simulations are shown in Fig. 4.6. Two approaches are considered. In Fig. 4.6a, the standard numerical procedure of the aero-icing framework described in Fig. 1.5 and Table 4.2 is applied. In Fig. 4.6b, complete solidification of the impinging water droplets is artificially enforced by overriding the thermodynamic module for the inviscid aerodynamic solvers.

When the standard thermodynamic solver is used, the inviscid approaches predict smaller ice accretions than the RANS solver at the stagnation point, confirming that the underestimated HTC values lead to incomplete freezing of the droplets. When full solidification is enforced, the results obtained with the inviscid solvers become consistent with the RANS solution. Nevertheless, for all aerodynamic approaches, a single-layer simulation remains insufficient to reproduce the experimentally observed ice shape.

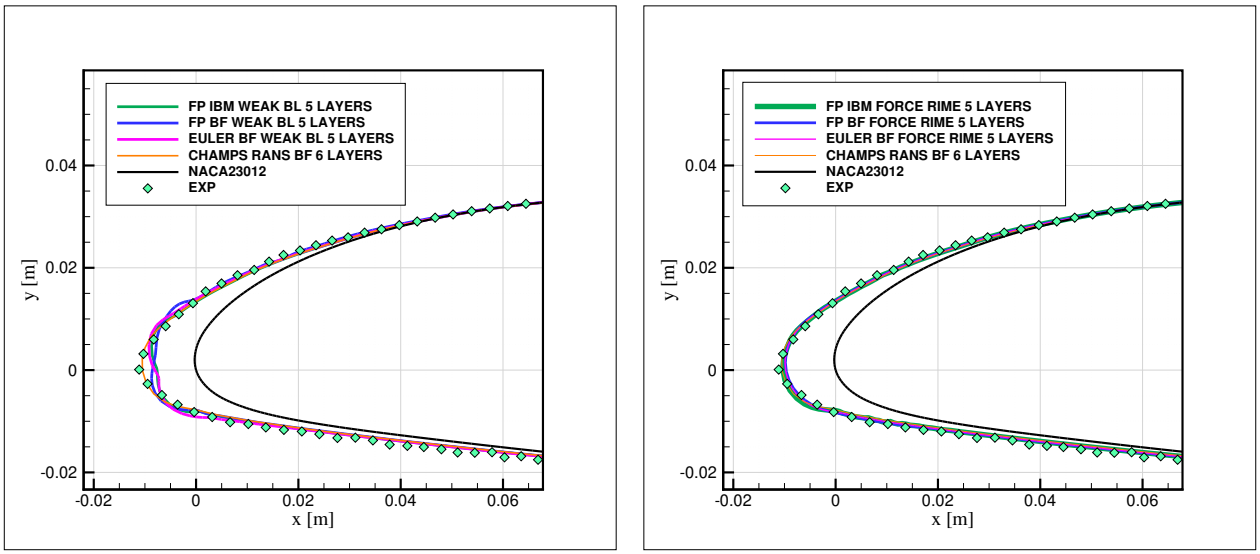


(a) With thermodynamic solver

(b) With complete solidification enforced

Figure 4.6 Ice shape predicted from a single-layer icing simulation using FP IB, FP BF and Euler BF solvers compared with the RANS solution and experimental data (Case 241)

Multi-layer icing simulations were therefore performed using the same two approaches. The results are presented in Fig. 4.7. When the thermodynamic module is used with the inviscid solvers, a fraction of the ice accretion is again missing due to the underestimated HTC values near the stagnation point. When the freezing fraction is artificially set to unity, the ice shapes predicted by the inviscid solvers become consistent with both the RANS results and the experimental data.



(a) With thermodynamic solver (b) With complete solidification enforced

Figure 4.7 Ice shape predicted from a multi-layer icing simulation using FP IB, FP BF and Euler BF solvers compared with the CHAMPS RANS solution and experimental data (Case 241)

The evolution of the ice shape during the multi-layer simulation is illustrated in Fig. 4.8 for both the FP BF and FP IB solvers when complete solidification is enforced. Good agreement between the two approaches is observed at each accretion step.

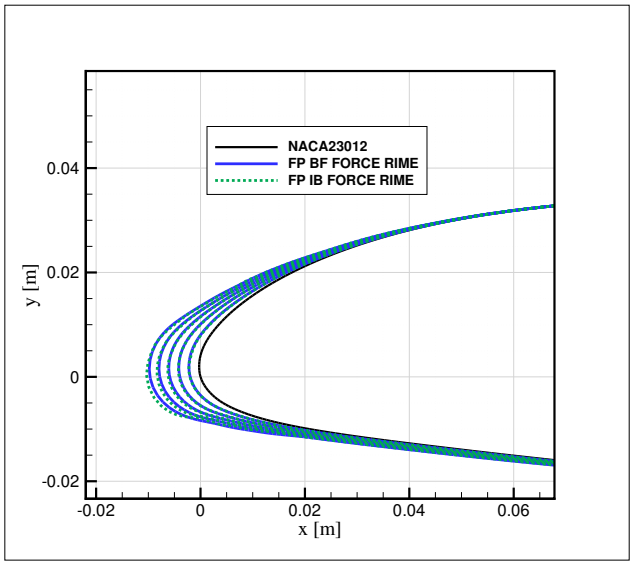
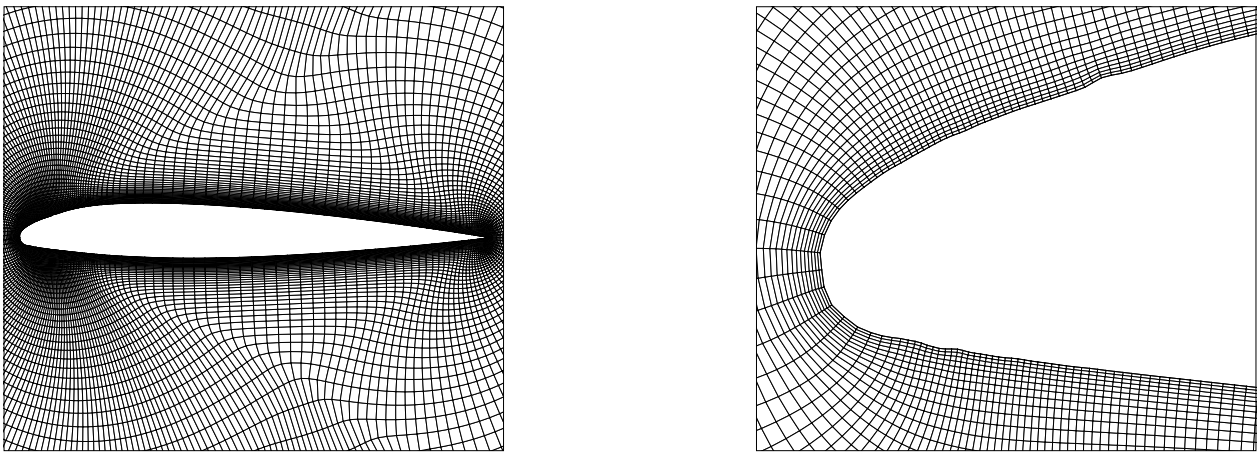


Figure 4.8 Ice shape evolution for the multi-layer simulation when complete solidification is enforced for the FP IB and FP BF solvers (Case 241)

To further assess the performance of the FP BF and FP IB solvers for aerodynamic simulations over rime-type ice geometries, additional flow simulations were performed over the final ice shape obtained from the RANS multi-layer icing simulation. These results were compared with those obtained using the Euler solver under the same flow conditions defined for case 241 in Table 4.1.

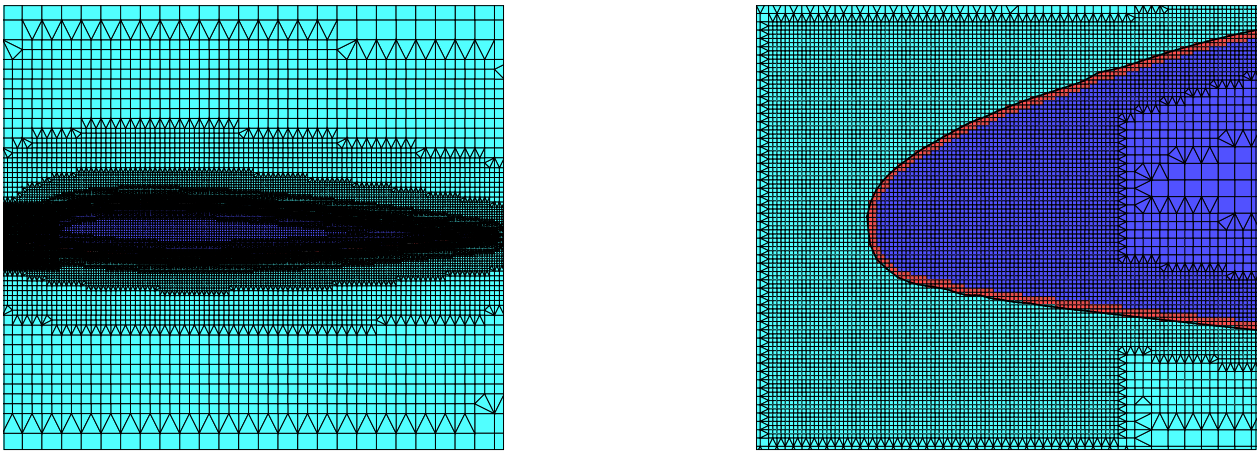
The body-fitted mesh used for these aerodynamic simulations is shown in Fig. 4.9. For the FP IB solver, the same initial mesh used for the multi-layer icing simulations was employed. The resulting immersed boundary mesh, including the corresponding cell reclassification around the iced geometry, is shown in Fig. 4.10.



(a) Airfoil view

(b) Leading edge view

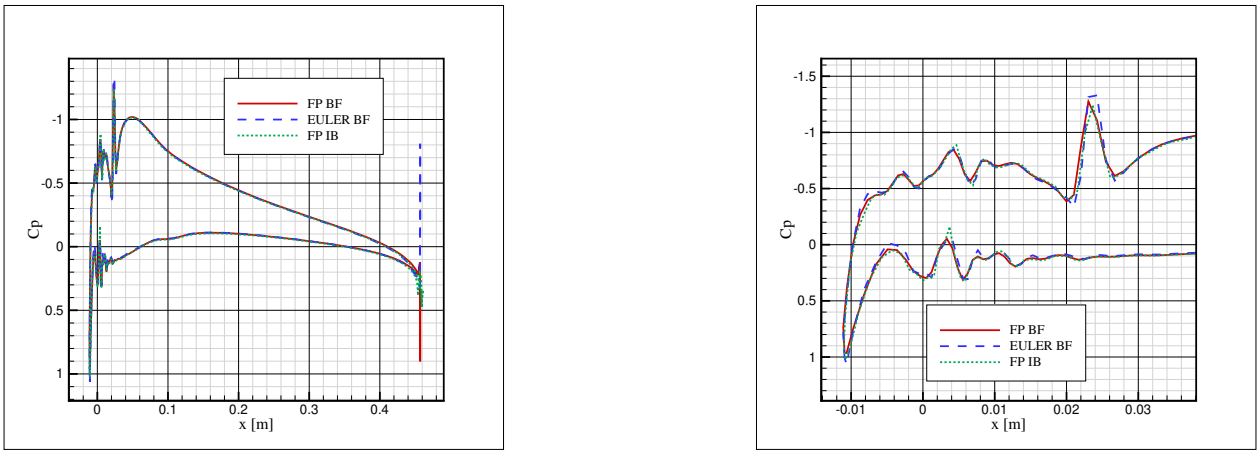
Figure 4.9 Close-up view of the body-fitted mesh for the final ice shape over NACA23012 (Case 241)



(a) Airfoil view (b) Leading edge view

Figure 4.10 Close-up view of the immersed boundary mesh for the final ice shape over NACA23012 (Case 241)

The resulting pressure coefficient distributions are presented in Fig. 4.11. Despite oscillations near the leading edge caused by small ice geometric irregularities, the results obtained with the FP BF and FP IB solvers show good agreement with those of the Euler solver. These results indicate that the full-potential solvers developed in Chapter 3 are capable of accurately predicting the inviscid aerodynamic characteristics of airfoils with rime-type ice accretions.



(a) Global distribution (b) LE distribution

Figure 4.11 Pressure coefficient distributions for the FP BF, Euler BF and FP IB solver for the final ice shape over NACA23012 (Case 241)

4.2 Case 242 of the IPW1

Case 242 corresponds to a higher-temperature icing condition, resulting in a glaze icing phenomena. In this case, water droplets do not freeze immediately upon impact but instead tend to run back along the surface before eventually freezing. Consequently, the aero-icing process is more complex than in rime icing conditions and depends not only on the droplet collection efficiency distribution but also on the thermodynamic model through the HTC and the skin-friction coefficient c_f distributions.

The HTC distributions over the clean airfoil near the stagnation point obtained with the four aerodynamic solvers are compared with results from other IPW1 participants in Fig. 4.12. As observed previously, noticeable discrepancies exist among the participants. The RANS solver produces HTC values that lie within the average range of the submitted results. In contrast, the three inviscid solvers (FP IB, FP BF and Euler BF) yield very similar HTC distributions but predict lower values compared with most other participants, particularly in the vicinity of the stagnation point and within the laminar region of the boundary layer.

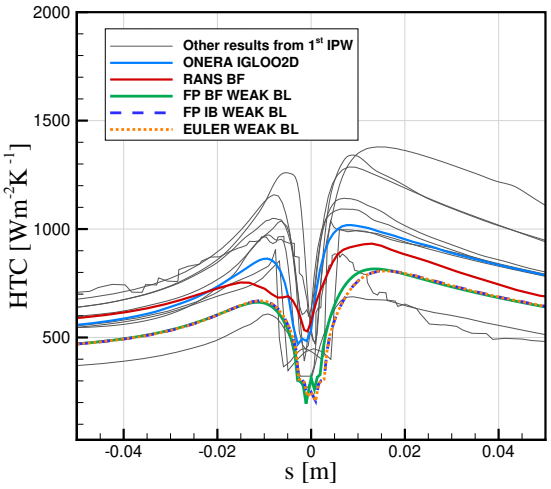


Figure 4.12 Comparisons of HTC distributions near the stagnation point for the FP IB, FP BF, Euler BF and RANS BF solvers with other IPW1 participants (Case 242)

The collection efficiency distributions β over the clean airfoil, expressed as a function of the vertical coordinate of the profile, are shown in Fig. 4.13 for all aerodynamic solvers. The results indicate a good agreement between the different approaches, suggesting that the droplet trajectory predictions are consistent among the solvers.

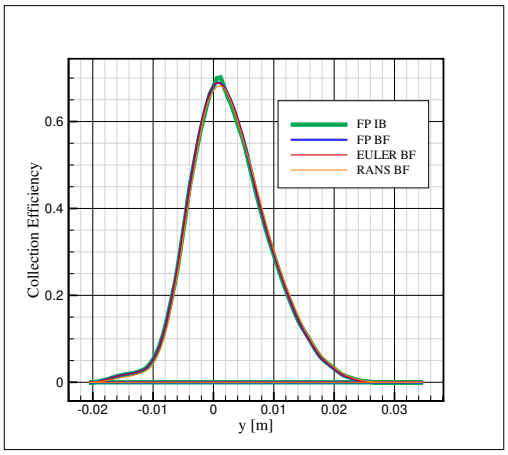


Figure 4.13 Collection efficiency (β) distributions according to the vertical coordinate for the FP IB, FP BF, Euler BF and RANS BF solvers (Case 242)

The ice shapes predicted by the single-layer simulations are presented in Fig. 4.14. The inviscid solvers produce nearly identical ice shapes and remain reasonably close to the RANS prediction, although the RANS ice shape terminates slightly closer to the stagnation point. Nevertheless, all predicted shapes differ significantly from the experimental geometry, which exhibits characteristic horn formations. Despite this discrepancy, all solvers predict the correct ice thickness at the stagnation point.

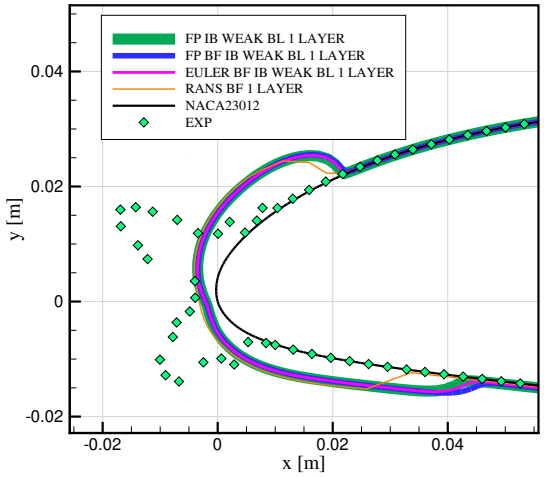


Figure 4.14 Ice shape predicted from a single-layer icing simulation using FP IB, FP BF and Euler BF solvers compared with the CHAMPS RANS solution and experimental data (Case 242)

The results obtained from the multi-layer icing simulations are shown in Fig. 4.15. The multi-layer accretion process allows the RANS solver to better reproduce the experimental ice shape. In contrast, the inviscid solvers show little improvement compared with their single-layer predictions. These results were obtained using a roughness coefficient of $k_s = c/1000$ for all solvers. Increasing this parameter can improve the agreement with the experimental geometry by enhancing the horn-like features of the ice shape. For instance, tuning the RANS simulation with $k_s = 4c/1000$ produces a closer match to the experimental profile. The inviscid solvers are also sensitive to this parameter because it directly influences the c_f and HTC values in the turbulent region of the boundary layer (see Eqs. (3.76, 3.80)). However, even after adjusting this parameter, the horn structures observed in the experimental ice shape are not recovered with the inviscid approaches.

Several factors may explain this discrepancy. First, the HTC values predicted by the inviscid methods appear to be too low, particularly in the laminar region of the boundary layer. This reduces heat transfer near the stagnation point, delaying freezing and allowing water to run further downstream before solidifying. Second, the horn-shaped geometry may generate recirculation zones associated with flow separation and reattachment, phenomena that are not captured with a weakly coupled boundary layer model. Finally, the u_e distributions provided to the boundary layer solver were smoothed to avoid separation and ensure numerical stability, which may also influence the boundary layer predictions.

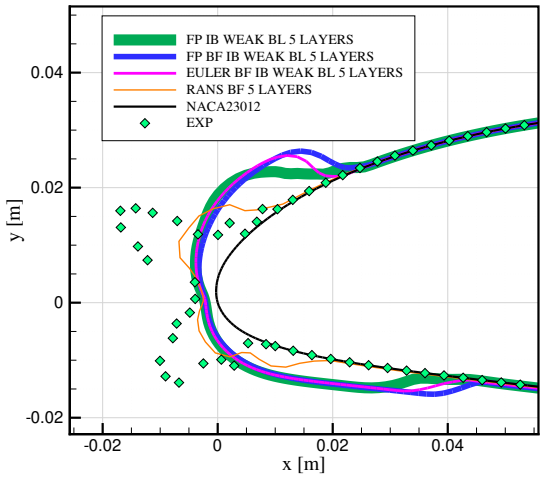
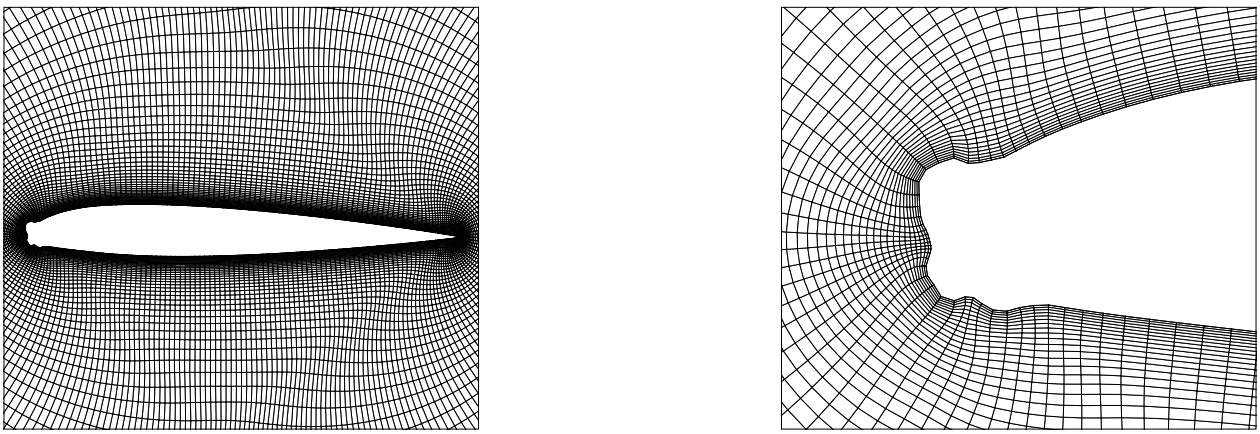


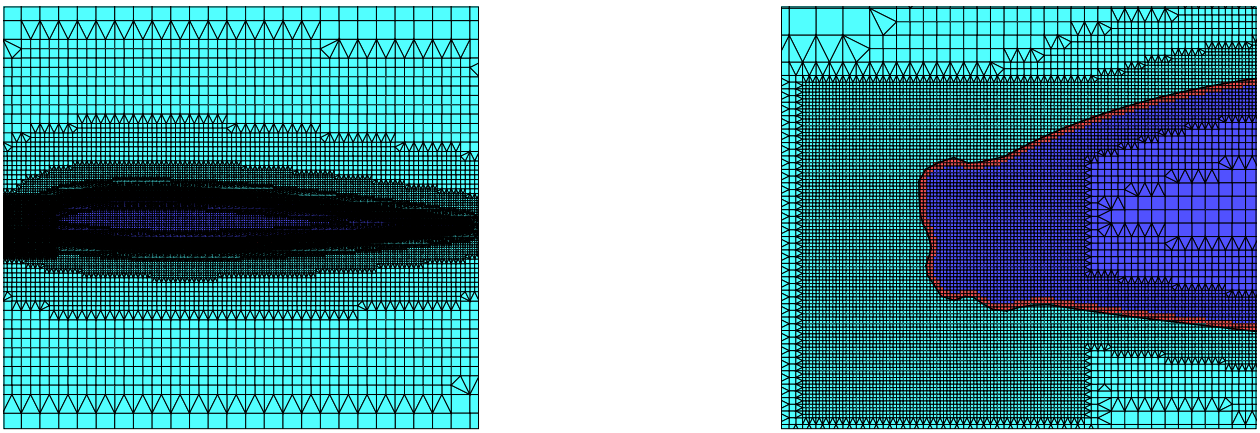
Figure 4.15 Ice shape predicted from a multi-layer icing simulation using FP IB, FP BF and Euler BF solvers compared with the CHAMPS RANS solution and experimental data (Case 242)

To further evaluate the FP BF and FP IB solvers for flows over glaze-type ice geometries, aerodynamic simulations were performed on the final NACA23012 geometry obtained from the RANS multi-layer icing simulation. These results were compared with those obtained using the Euler solver under the same conditions described for case 242 in Table 4.1. The body-fitted mesh used for these simulations is shown in Fig. 4.16. For the FP IB solver, the same base mesh used in the multi-layer icing simulations was employed, it is shown in Fig. 4.17, with the corresponding cell reclassification.



(a) Airfoil view (b) Leading edge view

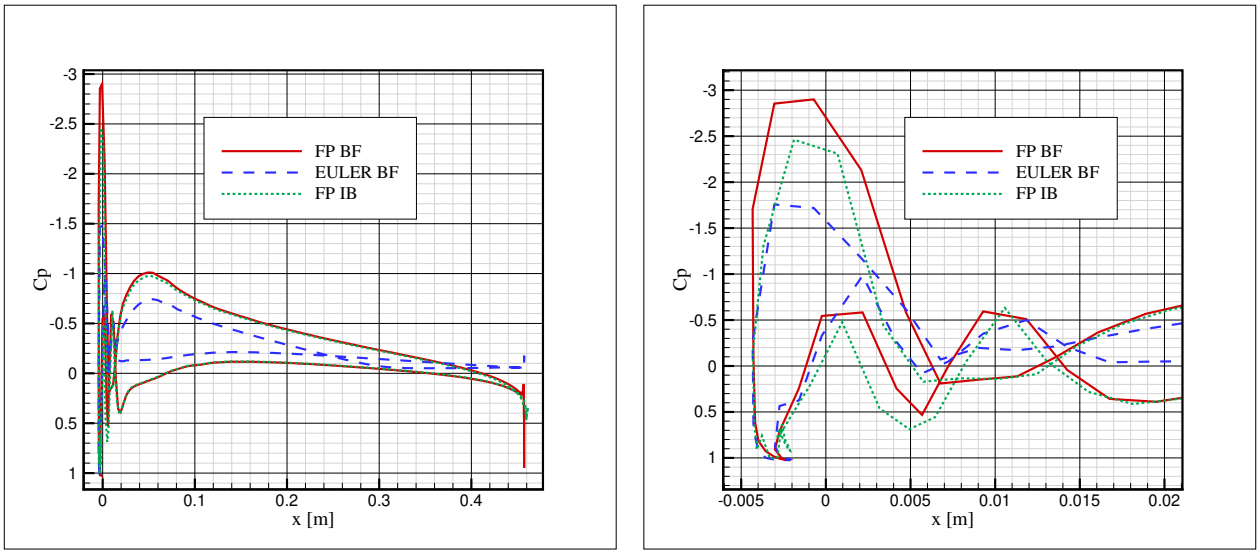
Figure 4.16 Close-up view of the body-fitted mesh for the final ice shape over NACA23012 (Case 242)



(a) Airfoil view (b) Leading edge view

Figure 4.17 Close-up view of the immersed boundary mesh for the final ice shape over NACA23012 (Case 242)

The pressure coefficient distributions are presented in Fig. 4.18. A good agreement is observed between the two full-potential solvers, whereas discrepancies appear in the results obtained with the Euler BF solver.



(a) Global distribution

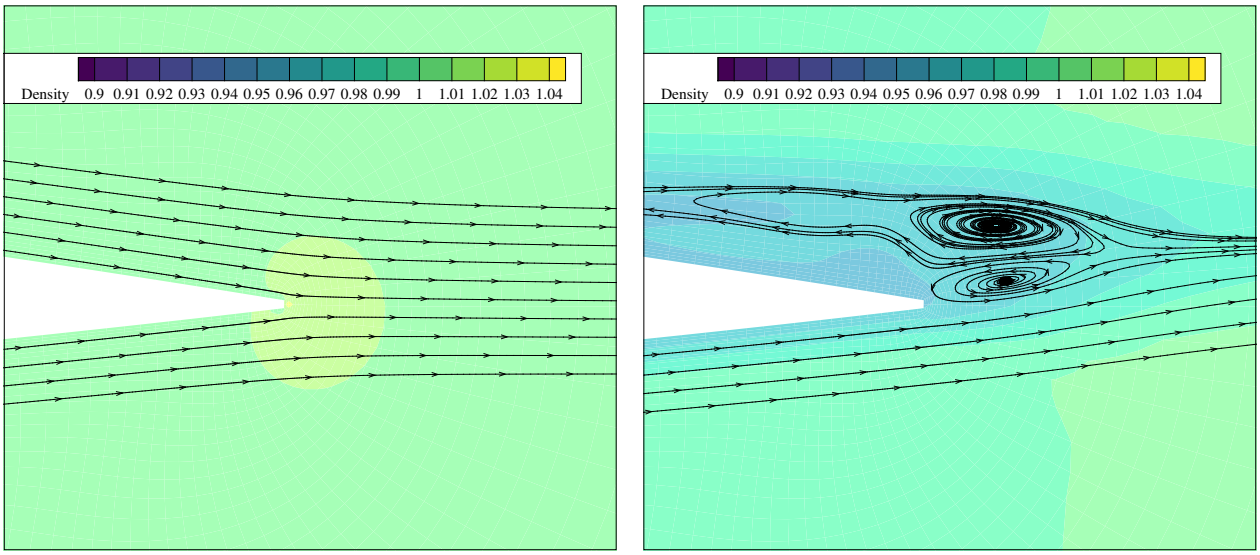
(b) LE distribution

Figure 4.18 Pressure coefficient distributions for the FP BF, Euler BF and FP IB solver for the final ice shape over NACA23012 (Case 242)

These discrepancies can be explained by the presence of a recirculation zone predicted in the Euler solution (see Fig. 4.19), which is an artifact of artificial dissipation. When the Euler simulation was repeated on a more refined mesh, the recirculation region disappeared and the resulting c_p distribution became consistent with the full-potential solutions.

Such recirculation was never observed in the full-potential simulations since the governing equations assume an irrotational flow field. While this limitation prevents the method from capturing physically occurring separation phenomena, it also avoids spurious recirculation regions caused by excessive artificial dissipation in Euler solvers.

Finally, the close agreement between the FP IB and FP BF solutions demonstrates that the immersed-boundary implementation provides reliable aerodynamic predictions even for complex iced geometries.



(a) FP BF solver

(b) Euler BF solver

Figure 4.19 Streamlines in the vicinity of the trailing edge for the FP-BF and Euler-BF solutions over the NACA23012 airfoil with the final ice shape (Case 242)

CHAPTER 5 CONCLUSION

5.1 Summary of Works

The objective of this research was to develop a medium-fidelity aerodynamic solver suitable for preliminary design applications, with a particular focus on simplifying mesh generation for complex and evolving geometries encountered in aero-icing simulations.

5.1.1 Development of an Unstructured Body-Fitted Full Potential Solver

To achieve this objective, a body-fitted full potential solver was first developed. A K-Exact Least Squares method, applicable to both structured and unstructured meshes, was used to approximate the gradients. The proposed implementation of the Kutta condition enables the simulation of blunt trailing edge geometries and wakes that do not conform to the mesh. The method was also tested on three-dimensional configurations, where limitations of the formulation were identified in the wingtip region. A modified formulation of the density upwinding was introduced to better account for the actual upstream flow direction. Several iterative strategies were investigated to drive the residuals to zero. When the Jacobian matrix is properly formulated, implicit solvers based on GMRES were found to be the most efficient. Two approaches were explored for assembling the Jacobian matrix: an analytical formulation and a numerical approximation using finite differences. Although more expensive, the numerical approach proved to be more robust in transonic regimes, whereas the analytical formulation neglects the upwinded density contribution. The importance of including the circulation contribution in the Jacobian was also demonstrated for lifting flows, as it significantly improves solver efficiency.

The solver was validated on several configurations, including the sharp and blunt NACA0012 airfoils, the RAE2822 supercritical airfoil, and the ONERA M6 wing, using both structured and unstructured meshes. Overall, the results obtained with the full potential solver showed good agreement with Euler simulations. In transonic conditions, the predicted shocks were slightly stronger and displaced downstream due to the isentropic assumption of the full potential formulation. The solver also demonstrated superior computational efficiency for subsonic flows.

Finally, a verification study confirmed the correctness of the implementation, as the expected second-order convergence rate of the discretization error was observed.

5.1.2 Integration of a Ghost-Cell Immersed Boundary Method

A GCIBM was integrated into the full potential solver, enabling simulations on non-body-conforming meshes. It includes a Boundary Flux Balancing method which addresses stagnant residual issues. Also, a mesh sequencing strategy was proposed to efficiently simulate lifting flows without relying on the implicit circulation formulation.

For all test cases considered on the NACA0012 airfoil, the immersed boundary solver demonstrated good agreement with the body-fitted full potential solver while maintaining comparable computational efficiency. A mesh refinement study revealed the presence of low-amplitude, high-frequency oscillations in the solution. The final verification study indicated an overall first-order convergence rate for the immersed boundary formulation.

5.1.3 Coupling to a Boundary Layer Solver

The inviscid full potential solvers were coupled with a boundary layer solver in order to approximate the skin-friction coefficient c_f and the heat transfer coefficient (HTC). For the laminar boundary layer, a formulation based on Thwaites' method was employed. Turbulent boundary layers were modeled using a formulation derived from Head's entrainment method. For the thermal boundary layer, an approach similar to that proposed by Bayeux and Elices [6, 31] was adopted.

The results obtained from this coupling were compared with those reported by Bayeux for a subsonic non-lifting flow around a NACA0012 airfoil. Good agreement was obtained for the dynamic boundary layer quantities with all inviscid solvers, while slight discrepancies were observed for the HTC values in the turbulent region.

5.1.4 Aero-Icing Simulations

The full potential solvers developed in this work were subsequently integrated into the CHAMPS icing framework in order to evaluate their suitability for aero-icing simulations. Two benchmark cases from the first Ice Prediction Workshop (IPW1), namely cases 241 and 242, were investigated.

For both cases, the predicted HTC distributions were significantly lower than those obtained with the CHAMPS RANS solver, particularly near the stagnation point. In contrast, the predicted collection efficiency distributions showed good agreement. For case 241, corresponding to rime icing conditions, the experimental ice shape was successfully reproduced in multi-layer simulations when complete freezing of the droplets was enforced. For case 242,

which corresponds to glaze icing conditions, the characteristic horn-shaped ice geometry observed experimentally was not reproduced by the inviscid solvers, even when using multiple accretion layers.

Finally, aerodynamic simulations were performed on the final ice geometries predicted by the RANS multi-layer simulations, for which good agreement was obtained between the inviscid solvers.

5.2 Limitations

The research presented in this thesis has limitations, which were already discussed throughout the previous sections.

At the level of the full potential solver, the analytical Jacobian fails to retain its efficiency advantage over the Euler solver for transonic flow simulations. In addition, the isentropic assumption inherent to the full potential formulation introduces inaccuracies in shock prediction. The current implementation of the Kutta condition is limited to configurations with a single lifting surface. Moreover, a loss of lift was observed near the wingtip for three-dimensional wing simulations, which is attributed to the present formulation of the Kutta condition. Finally, due to the irrotational assumption of the full potential model, recirculating flows cannot be captured.

For the immersed boundary method, the approach has only been implemented and tested in two dimensions. The surface solution obtained with the IBM formulation exhibits small-amplitude, high-frequency oscillations, and the current implementation is limited to first-order accuracy.

In the present coupling between the inviscid solver and the boundary layer model, the formulation is unable to handle separated flows and is currently restricted to two-dimensional configurations. In addition, the predicted heat transfer coefficient appears to be underestimated, which reduces the accuracy of the icing simulations. As a consequence, the methodology does not yet provide satisfactory predictions for glaze icing cases.

5.3 Future Research

Several avenues of research can be explored in order to address the limitations identified in this work.

For the full potential solver, a promising direction would be the inclusion of the upwinded density term in the analytical formulation of the Jacobian matrix. This modification could

potentially improve the efficiency of transonic simulations and make the solver more competitive than Euler solvers for these conditions.

For configurations involving multiple lifting surfaces, one possible approach would be to define a Kutta condition for each lifting surface. Each one of them could be associated with its own reference line following the path of its wake, allowing the different wakes to intersect and be paired consistently.

Regarding the three-dimensional Kutta condition, a potential approach to address the wingtip issue would be to compute the potential jump differently for stencil cells located upstream of the trailing edge. This could be achieved by evaluating the difference between the potentials obtained by interpolating the potential values on the upper and lower surfaces of the wing along both the flow direction and the spanwise direction.

The immersed boundary method could also be extended to three-dimensional simulations. Although the general extension is expected to be relatively straightforward, particular care will be required for the implementation of the corresponding Kutta condition. In addition, the development of a second-order accurate formulation would be desirable. This may be achievable by considering $k = 2$ in the K-Exact Least Squares reconstruction at the immersed boundary facets.

Further work is also required on the coupling between the inviscid solver and the boundary layer model. In particular, a formulation that generalizes more naturally to three-dimensional flows and that can account for boundary layer separation should be investigated. Moreover, the reasons behind the relatively low predicted values of the heat transfer coefficient (HTC) should be further analyzed.

Finally, the immersed boundary full potential solver developed in this thesis could be applied to additional research areas. For example, it could be integrated into aerodynamic shape optimization frameworks or coupled with aeroelastic simulation tools.

REFERENCES

[1] "1st aiaa ice prediction workshop," https://icepredictionworkshop.wordpress.com/workshop/, May 2021.

[2] A. Rizzi, "Modeling and simulating aircraft stability and control—The SimSAC project," *Progress in Aerospace Sciences*, vol. 47, no. 8, pp. 573–588, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0376042111000704

[3] A. Jameson, *Aerodynamics*. John Wiley Sons, Ltd, 2004, ch. 11. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/0470091355.ecm062

[4] Direction Générale de l'Aviation Civile, "Givrages aéronefs : Document de synthèse," Direction Générale de l'Aviation Civile, Tech. Rep., Oct. 2008.

[5] F. Capizzano, "Turbulent wall model for immersed boundary methods," *AIAA journal*, vol. 49, no. 11, pp. 2367–2381, 2011.

[6] C. Bayeux, "Méthode intégrale pour la couche limite tridimensionnelle - Applications au givrage," Theses, INSTITUT SUPERIEUR DE L'AERONAUTIQUE ET DE L'ESPACE (ISAE) ; UNIVERSITE DE TOULOUSE, Dec. 2017. [Online]. Available: https://hal.science/tel-01715950

[7] P. Lavoie, "Modeling of thin water films on swept wings in icing condition," Master's thesis, École Polytechnique de Montréal, avril 2017. [Online]. Available: https://publications.polymtl.ca/2558/

[8] J. Blazek, *Computational Fluid Dynamics*, 3rd ed. Elsevier, Mar. 2015. [Online]. Available: https://shop.elsevier.com/books/computational-fluid-dynamics/blazek/978-0-08-099995-1

[9] R. E. Neel, "Advances in computational fluid dynamics: Turbulent separated flows and transonic potential flows," Ph.D., Virginia Polytechnic Institute, 1997, iSBN: 9780599536869. [Online]. Available: https://www.proquest.com/docview/304371848/abstract/26078C96633B45BBPQ/1

[10] Fanxi Lyu, Tianhang Xiao, and Xiongqing Yu, "A fast and automatic full-potential finite volume solver on Cartesian grids for unconventional configurations," *Chinese Journal of Aeronautics*, vol. 30, no. 3, pp. 951–63, Jun. 2017, place: Netherlands Publisher: Elsevier B.V.

[11] J. Vassberg and A. Jameson, "Industrial applications of aerodynamic shape optimization," in *VKI Lecture-II*. von Karman Institute for Fluid Dynamics, 2022. [Online]. Available: https://store.vki.ac.be/vkils201804-article4.html

[12] S. Gallay and Laurendeau, "Nonlinear Generalized Lifting-Line Coupling Algorithms for Pre/Poststall Flows," *AIAA Journal*, vol. 53, no. 7, pp. 1784–1792, 2015, publisher: American Institute of Aeronautics and Astronautics. [Online]. Available: https://publications.polymtl.ca/34941/

[13] R. Mukherjee, A. Gopalarathnam, and S. Kim, "An Iterative Decambering Approach for Post-Stall Prediction of Wing Characteristics from Known Section Data," in *41st Aerospace Sciences Meeting and Exhibit*, ser. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, Jan. 2003. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2003-1097

[14] M. Parenteau, "A Non-Linear Frequency Domain Potential Flow Model for Compressible, Transonic and Viscous Aeroelastic Analyses," PhD Thesis, Polytechnique Montréal, 2021. [Online]. Available: https://publications.polymtl.ca/6629/

[15] J. Slotnick *et al.*, "CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences," NASA, Tech. Rep., Mar. 2014. [Online]. Available: https://www.semanticscholar.org/paper/CFD-Vision-2030-Study%3A-A-Path-to-Revolutionary-Slotnick-Khodadoust/c41748358b6c70c5cea0e15d7b7beef1149c1132

[16] Centro de Investigação e Prevenção de Acidentes Aeronáuticos (CENIPA), "Preliminary report of the ps-vpb aircraft accident," Center for Investigation and Prevention of Aeronautical Accidents (CENIPA), Tech. Rep., 2024. [Online]. Available: https://dedalo.sti.fab.mil.br/en/85259

[17] P. G. Tucker, "Mesh Generation," in *Advanced Computational Fluid and Aerodynamics*, ser. Cambridge Aerospace Series, P. G. Tucker, Ed. Cambridge: Cambridge University Press, 2016, pp. 67–147. [Online]. Available: https://www.cambridge.org/core/books/advanced-computational-fluid-and-aerodynamics/mesh-generation/556AD1FF1E8884D5CFADAC1C76AC5FC3

[18] J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin, *Numerical Grid Generation: Foundations and Applications*. North-Holland, 1985.

[19] T. Cebeci *et al.*, *Computational Fluid Dynamics for Engineers*. Springer, 2005.

[20] D. J. Mavriplis, "Unstructured mesh generation and adaptivity," NASA, Tech. Rep. NAS 1.26:195069, Apr. 1995, nTRS Author Affiliations: Institute for Computer Applications in Science and Engineering NTRS Document ID: 19950020607 NTRS Research Center: Legacy CDMS (CDMS). [Online]. Available: https://ntrs.nasa.gov/citations/19950020607

[21] J. H. Ferziger, M. Perić, and R. L. Street, *Computational methods for fluid dynamics*. Springer, 2002, vol. 3.

[22] Y. Abe *et al.*, "Geometric interpretations and spatial symmetry property of metrics in the conservative form for high-order finite-difference schemes on moving and deforming grids," *Journal of Computational Physics*, vol. 260, pp. 163–203, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999113008188

[23] A. Crovato, "Steady Transonic Aerodynamic and Aeroelastic Modeling for Preliminary Aircraft Design," Ph.D. dissertation, ULiège - Université de Liège, Dec. 2020, publisher: ULiège - Université de Liège. [Online]. Available: https://orbi.uliege.be/handle/2268/251906

[24] M. C. Galbraith, S. R. Allmaras, and R. Haimes, "Full Potential Revisited: A Medium Fidelty Aerodynamic Analysis Tool," in *55th AIAA Aerospace Sciences Meeting*, ser. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, Jan. 2017. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2017-0290

[25] O. C. Zienkiewicz *et al.*, *The finite element method*. Elsevier, 1977, vol. 3.

[26] B. A. Nishida, "Fully simultaneous coupling of the full potential equation and the integral boundary layer equations in three dimensions," Thesis, Massachusetts Institute of Technology, 1996, accepted: 2005-08-18T12:00:00Z. [Online]. Available: https://dspace.mit.edu/handle/1721.1/11188

[27] D. Eller, "Fast, Unstructured-Mesh Finite-Element Method for Nonlinear Subsonic Flow," *Journal of Aircraft*, vol. 49, no. 5, pp. 1471–1479, Sep. 2012. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.C031738

[28] M. Núñez *et al.*, "An embedded approach for the solution of the full potential equation with finite elements," *Computer Methods in Applied Mechanics and Engineering*, vol. 388, p. 114244, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S004578252100565X

[29] I. P. López Canalejo, "A finite-element transonic potential flow solver with an embedded wake approach for aircraft conceptual design," Ph.D. dissertation, Technische Universität München, Munich, Germany, 2022.

[30] G. Auger, E. Laurendeau, and Y. Hoarau, "Full potential calculations with immersed boundary ghost-cell method," in *AIAA SCITECH 2026 Forum*, 2026, p. 0890.

[31] P. Elices Paz, "Immersed boundary methods for the three-dimensional modeling of in-flight ice accretion," Ph.D. dissertation, Polytechnique Montréal, 2025.

[32] P. Elices Paz *et al.*, "Second-Order Ghost-Cell Immersed Boundary Methodology for Inviscid 3d Flows in Complex Geometries," Rochester, NY, Feb. 2025. [Online]. Available: https://papers.ssrn.com/abstract=5158155

[33] A. Parrinello and P. Mantegazza, "Independent Two-Fields Solution for Full-Potential Unsteady Transonic Flows," *AIAA Journal*, vol. 48, no. 7, pp. 1391–1402, Jul. 2010. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.J050013

[34] ——, "Improvements and Extensions to a Full-Potential Formulation Based on Independent Fields," *AIAA Journal*, vol. 50, no. 3, Mar. 2012. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.J051270

[35] P. Lavoie, "Immersed Boundary Methods for the Modeling of In-Flight Ice Accretion," PhD Thesis, Polytechnique Montréal, May 2021. [Online]. Available: https://publications.polymtl.ca/6577/

[36] P. Lavoie *et al.*, "An improved characteristic based volume penalization method for the Euler equations towards icing applications," *Computers & Fluids*, vol. 222, p. 104917, May 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045793021000839

[37] R. M. Liegl, "A full potential solver for lifting flows on unstructured tetrahedral meshes," Master's thesis, Concordia University, 2005. [Online]. Available: https://spectrum.library.concordia.ca/id/eprint/8788/

[38] Y. Saad, *Iterative methods for sparse linear systems*. SIAM, 2003.

[39] G. Meurant, *Computer solution of large linear systems*. Elsevier, 1999, vol. 28.

[40] Y. Saad and M. H. Schultz, "Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and*

Statistical Computing, vol. 7, no. 3, pp. 856–869, 1986. [Online]. Available: https://doi.org/10.1137/0907058

[41] E. Cuthill and J. McKee, "Reducing the bandwidth of sparse symmetric matrices," in *Proceedings of the 1969 24th National Conference*, ser. ACM '69. New York, NY, USA: Association for Computing Machinery, 1969, p. 157–172. [Online]. Available: https://doi.org/10.1145/800195.805928

[42] T. L. Holst, "Transonic flow computations using nonlinear potential methods," *Progress in Aerospace Sciences*, vol. 36, no. 1, pp. 1–61, Jan. 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037604219900010X

[43] P. A. Newman and J. C. South Jr, "Conservative versus nonconservative differencing: transonic streamline shape effects," NASA, Tech. Rep., 1976.

[44] ——, "Influence of nonconservative differencing on transonic streamline shapes," *AIAA Journal*, vol. 14, no. 8, pp. 1148–1149, 1976.

[45] E. M. Murman and J. D. Cole, "Calculation of plane steady transonic flows," *AIAA Journal*, vol. 9, no. 1, pp. 114–121, Jan. 1971. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.6131

[46] F. R. Bailey, "Numerical calculation of transonic flow about slender bodies of revolution," NASA, Tech. Rep. TN D-6582, 1971.

[47] J. Krupp and E. Murman, "The numerical calculation of steady transonic flows past thin liftingairfoils and slender bodies," in *4th Fluid and Plasma Dynamics Conference*, ser. Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, Jun. 1971. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.1971-566

[48] F. R. BAILEY and J. L. STEGER, "Relaxation Techniques for Three-Dimensional Transonic Flow about Wings," *AIAA Journal*, vol. 11, no. 3, pp. 318–325, 1973, _eprint: https://doi.org/10.2514/3.50473. [Online]. Available: https://doi.org/10.2514/3.50473

[49] W. Ballhaus and F. Bailey, "Numerical calculation of transonic flow about swept wings," in *Society of Naval Architects and Marine Engineers, and U.S. Navy, Advanced Marine Vehicles Meeting*, ser. Advanced Marine Vehicles Conferences. American Institute of Aeronautics and Astronautics, Jul. 1972. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.1972-677

[50] P. A. Newman and E. B. Klunker, "Computation of Transonic Flow about Finite Lifting Wings," *AIAA Journal*, vol. 10, no. 7, pp. 971–973, Jul. 1972. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.50274

[51] W. Schmidt and S. Hedman, "Recent explorations in relaxation methods for three-dimensional transonic potential flow," in *ICAS Paper*, no. 76-22, 1976.

[52] W. MASON *et al.*, "A numerical three-dimensional viscous transonic wing-body analysis and design tool," in *16th Aerospace Sciences Meeting*. American Institute of Aeronautics and Astronautics, 1978, _eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.1978-101. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.1978-101

[53] C. M. Albone, M. G. Hall, and G. Joyce, "Numerical Solutions for Transonic Flows Past Wing-Body Combinations," in *Symposium Transsonicum II*, K. Oswatitsch and D. Rues, Eds. Berlin, Heidelberg: Springer, 1976, pp. 541–548.

[54] J. van der Vooren *et al.*, "Remarks on the Suitability of Various Transonic Small Perturbation Equations to Describe Three-Dimensional Transonic Flow; Examples of Computations Using a Fully-Conservative Rotated Difference Scheme," in *Symposium Transsonicum II*, K. Oswatitsch and D. Rues, Eds. Berlin, Heidelberg: Springer, 1976, pp. 557–566.

[55] J. Steger and H. Lomax, "Numerical calculation of transonic flow about two-dimensional airfoils by relaxation procedures," in *4th Fluid and Plasma Dynamics Conference*. American Institute of Aeronautics and Astronautics, 1971, _eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.1971-569. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.1971-569

[56] P. R. Garabedian and D. G. Korn, "Analysis of transonic airfoils," *Communications on Pure and Applied Mathematics*, vol. 24, no. 6, pp. 841–851, 1971, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160240608. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160240608

[57] F. Bauer *et al.*, *Supercritical Wing Sections II*, ser. Lecture Notes in Economics and Mathematical Systems. New York: Springer, 1975, vol. 108.

[58] J. F. Nash and A. G. J. Macdonald, "The calculation of momentum thickness in a turbulent boundary layer at mach numbers up to unity," Aeronautical Research Council, London, Tech. Rep. C.P. No. 963, 1967.

[59] A. Jameson, "Iterative solution of transonic flows over airfoils and wings, including flows at mach 1," *Communications on Pure and Applied Mathematics*, vol. 27, no. 3, pp. 283–309, 1974, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160270302. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160270302

[60] A. Jameson *et al.*, "A brief description of the jameson–caughey nyu transonic swept-wing computer program flo-22," NASA, Technical Memorandum TM X-73996, 1976.

[61] T. A. Reyhner, "Cartesian Mesh Solution for Axisymmetric Transonic Potential Flow Around Inlets," *AIAA Journal*, vol. 15, no. 5, pp. 624–631, May 1977. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.60670

[62] B. Sanderse, "Cartesian grid methods for preliminary aircraft design," MSc Thesis, Delft University of Technology, 2008.

[63] A. Jameson, "Transonic potential flow calculation using conservative form," in *Proceedings of the Second AIAA Computational Fluid Dynamics Conference*, 1975, pp. 148–155.

[64] A. Jameson and D. A. Caughey, "A finite-volume method for transonic potential flow calculations," in *Proceedings of the Third AIAA Computational Fluid Dynamics Conference*, 1977, pp. 35–54.

[65] D. A. Caughey and A. Jameson, "Numerical calculation of transonic potential flow about wing-body combinations," *AIAA Journal*, vol. 17, pp. 175–181, 1979.

[66] ——, "Progress in finite-volume calculations for wing-fuselage combinations," *AIAA Journal*, vol. 18, pp. 1281–1288, 1980.

[67] C. L. Streett, "Viscous-Inviscid Interaction for Transonic Wing-Body Configurations Including Wake Effects," *AIAA Journal*, vol. 20, no. 7, pp. 915–923, Jul. 1982. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.51149

[68] S. H. Woodson, J. F. Campbell, and F. R. DeJarnette, "Interactive three-dimensional boundary-layer method for transonic flow over swept wings," *AIAA Journal*, vol. 29, pp. 678–679, 1991.

[69] O. V. Karas and V. E. Kovalev, "Blwf56 presentation," http://blwf-aero.ru/BLWF_code/index_en.html, accessed online.

[70] T. L. Holst, "Multizone Chimera Algorithm for Solving the Full-Potential Equation," *Journal of Aircraft*, vol. 35, no. 3, pp. 412–421, May 1998. [Online]. Available: https://arc.aiaa.org/doi/10.2514/2.2339

[71] M. Davari *et al.*, "A cut finite element method for the solution of the full-potential equation with an embedded wake," *Computational Mechanics*, vol. 63, no. 5, pp. 821–833, May 2019. [Online]. Available: https://doi.org/10.1007/s00466-018-1624-3

[72] F. T. Johnson *et al.*, "Tranair: A full-potential, solution-adaptive, rectangular grid code for predicting subsonic, transonic, and supersonic flows about arbitrary configurations," NASA, Tech. Rep., 1992.

[73] M. B. Bieterman *et al.*, "Boundary layer coupling in a general configuration full potential code," The Boeing Company, Seattle, WA, Tech. Rep., 1994.

[74] L. L. Erickson and S. M. Strande, "A theoretical basis for extending surface-paneling methods to transonic flow," *AIAA Journal*, vol. 23, no. 12, pp. 1860–1867, Dec. 1985. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.9188

[75] P. M. Sinclair, "An exact integral (field panel) method for the calculation of two-dimensional transonic potential flow around complex configurations," *The Aeronautical Journal*, 1986.

[76] ——, "A three-dimensional field-integral method for the calculation of transonic flow on complex configurations — theory and preliminary results," *The Aeronautical Journal*, 1988.

[77] Altair Engineering, "Flightstream theory manual," https://flightstream-theory.altair.com/, 2026.

[78] M. Hafez, E. Murman, and J. South, "Artificial compressibility methods for numerical solutions of transonic full potential equation," in *11th Fluid and PlasmaDynamics Conference*, ser. Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, Jul. 1978. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.1978-1148

[79] W. G. Habashi and M. M. Hafez, "Finite element solutions of transonic flow problems," *Aiaa Journal*, vol. 20, no. 10, pp. 1368–1376, 1982.

[80] G. S. Dulikravich, "Analysis of artificial dissipation models for the transonic full-potential equation," *AIAA journal*, vol. 26, no. 10, pp. 1238–1245, 1988.

[81] M. Hafez, W. Whitlow Jr, and S. Osher, "Improved finite-difference schemes for transonic potential flow calculations," *AIAA journal*, vol. 25, no. 11, pp. 1456–1462, 1987.

[82] G. Volpe and A. Jameson, "Transonic potential flow calculations by two artificial density methods," *AIAA journal*, vol. 26, no. 4, pp. 425–429, 1988.

[83] M. Hafez and D. Lovell, "Entropy and vorticity corrections for transonic flows," in *6th Computational Fluid Dynamics Conference Danvers*, 1983, p. 1926.

[84] G. H. Klopfer and D. Nixon, "Nonisentropic potential formulation for transonic flows," *AIAA journal*, vol. 22, no. 6, pp. 770–776, 1984.

[85] J. W. Purvis and J. E. Burkhalter, "Prediction of critical mach number for store configurations," *AIAA Journal*, vol. 17, no. 11, pp. 1170–1177, 1979.

[86] D. K. Clarke, M. Salas, and H. Hassan, "Euler calculations for multielement airfoils using cartesian grids," *AIAA journal*, vol. 24, no. 3, pp. 353–358, 1986.

[87] B. Muralidharan and S. Menon, "A high-order adaptive cartesian cut-cell method for simulation of compressible viscous flow over immersed bodies," *Journal of Computational Physics*, vol. 321, pp. 342–368, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999116301954

[88] M. Aftosmis, M. Berger, and G. Adomavicius, "A parallel multilevel method for adaptively refined cartesian grids with embedded boundaries," in *38th Aerospace Sciences Meeting and Exhibit*, 2000, p. 808.

[89] M. Berger and M. Aftosmis, "Progress towards a cartesian cut-cell method for viscous compressible flow," in *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2012, p. 1301.

[90] T. Ye *et al.*, "An accurate cartesian grid method for viscous incompressible flows with complex immersed boundaries," *Journal of Computational Physics*, vol. 156, no. 2, pp. 209–240, 1999. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999199963568

[91] M. Kirkpatrick, S. Armfield, and J. Kent, "A representation of curved boundaries for the solution of the navier–stokes equations on a staggered three-dimensional cartesian grid," *Journal of Computational Physics*, vol. 184, no. 1, pp. 1–36, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002199910200013X

[92] C. S. Peskin, "Flow patterns around heart valves: A numerical method," *Journal of Computational Physics*, vol. 10, no. 2, pp. 252–271, 1972. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999172900654

[93] E. Arquis and J.-P. Caltagirone, "Sur les conditions hydrodynamiques au voisinage d'une interface milieu fluide-milieu poreux: application à la convection naturelle," *CR Acad. Sci. Paris II*, vol. 299, no. 1, 1984.

[94] P. Angot, C.-H. Bruneau, and P. Fabrie, "A penalization method to take into account obstacles in incompressible viscous flows," *Numerische Mathematik*, vol. 81, no. 4, pp. 497–520, Feb. 1999. [Online]. Available: https://doi.org/10.1007/s002110050401

[95] A. Etcheverlepo, "Développement de méthodes de domaines fictifs au second ordre," Theses, Université Sciences et Technologies - Bordeaux I, Jan. 2013, issue: 2013BOR14758. [Online]. Available: https://theses.hal.science/tel-00821897

[96] A. Dadone and B. Grossman, "Ghost-cell method for analysis of inviscid three-dimensional flows on cartesian-grids," *Computers & fluids*, vol. 36, no. 10, pp. 1513–1528, 2007.

[97] P. Elices Paz *et al.*, "Second-order ghost-cell immersed boundary methodology for inviscid 3d flows in complex geometries," *Available at SSRN 5158155*, 2025.

[98] L. Prandtl, "Motion of fluids with very little viscosity," NASA, Tech. Rep., 1928.

[99] D. Catherall and K. W. Mangler, "An indirect method for the solution of the navier–stokes equations for laminar incompressible flow at large reynolds numbers," Royal Aircraft Establishment, Farnborough, United Kingdom, Tech. Rep. RAE Aero 2683, 1963, also issued as A.R.C. 25,599.

[100] J. Carter, "Viscous-inviscid interaction analysis of transonic turbulent separated flow," in *14th Fluid and Plasma Dynamics Conference*, 1981, p. 1241.

[101] A. E. P. Veldman, "New, quasi-simultaneous method to calculate interacting boundary layers," *AIAA journal*, vol. 19, no. 1, pp. 79–85, 1981.

[102] S. Bourgault-Côté, "Ice interface evolution modelling algorithms for aircraft icing," Ph.D. dissertation, Polytechnique Montréal, février 2019. [Online]. Available: https://publications.polymtl.ca/3805/

[103] Y. Bourgault *et al.*, "A finite element method study of eulerian droplets impingement models," *International Journal for Numerical methods in fluids*, vol. 29, no. 4, pp. 429–449, 1999.

[104] B. L. Messinger, "Equilibrium temperature of an unheated icing surface as a function of air speed," *Journal of the aeronautical sciences*, vol. 20, no. 1, pp. 29–42, 1953.

[105] C. ZHU *et al.*, "3d ice accretion simulation for complex configuration basing on improved messinger model," *International Journal of Modern Physics: Conference Series*, vol. 19, pp. 341–350, 2012. [Online]. Available: https://doi.org/10.1142/S2010194512008938

[106] T. G. Myers, "Extension to the messinger model for aircraft icing," *AIAA journal*, vol. 39, no. 2, pp. 211–218, 2001.

[107] J. Huang *et al.*, "Multistep simulation for three-dimensinal ice accretion on an aircraft wing," in *AIAA Modeling and Simulation Technologies Conference*, 2016, p. 1918.

[108] Y. Bourgault, H. Beaugendre, and W. G. Habashi, "Development of a shallow-water icing model in fensap-ice," *Journal of Aircraft*, vol. 37, no. 4, pp. 640–646, 2000.

[109] M. Parenteau *et al.*, "Development of Parallel CFD Applications with the Chapel Programming Language," in *AIAA Scitech 2021 Forum*, ser. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, Jan. 2021. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2021-0749

[110] H. P. Laroche *et al.*, "Development of an aircraft aero-icing suite using chapel programming language," in *8th annual Chapel Implementers and Users Workshop (CHIUW 2021)*, 2021. [Online]. Available: https://publications.polymtl.ca/49718/

[111] Chapel Team, "The Chapel Programming Language," 2025. [Online]. Available: https://chapel-lang.org/

[112] M. Blanchet, M. K. Zayni, and E. Laurendeau, "Contribution to IPW2 by Studying Single and Multi-Layer Icing in 2D, 2.5D and 3D," in *AIAA AVIATION FORUM AND ASCEND 2024*, ser. AIAA Aviation Forum and ASCEND co-located Conference Proceedings. American Institute of Aeronautics and Astronautics, Jul. 2024. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2024-3680

[113] F. Plante, "DPW7 Results from Champs, Including Global Stability Analyses," in *AIAA AVIATION 2023 Forum*. American Institute of Aeronautics and Astronautics,

2023, _eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2023-3399. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2023-3399

[114] B. Arnould and Laurendeau, "CHAMPS' Fixed Grid RANS Simulations at Fifth High Lift Prediction Workshop," in *AIAA SCITECH 2025 Forum*. American Institute of Aeronautics and Astronautics, Jan. 2025. [Online]. Available: https://publications.polymtl.ca/61900/

[115] T. Barth and P. Frederickson, "Higher order solution of the euler equations on unstructured grids using quadratic reconstruction," in *28th aerospace sciences meeting*, 1990, p. 13.

[116] J. W. Slater, "Onera m6 wing: Study #1," https://www.grc.nasa.gov/www/wind/valid/m6wing/m6wing01/m6wing01.html, 2002.

[117] C. Benoit, S. Peron, and S. Landier, "Cassiopee: a cfd pre- and post-processing tool," *Aerospace Science and Technology*, vol. 45, pp. 272–283, 2015.

[118] V. Schmitt and F. Charpin, "Pressure distributions on the onera-m6-wing at transonic mach numbers," AGARD, AGARD Advisory Report AR-138, May 1979, experimental Data Base for Computer Program Assessment, Fluid Dynamics Panel Working Group 04.

[119] F. Jaillet and C. Lobos, "Fast Quadtree/Octree adaptive meshing and re-meshing with linear mixed elements," *Engineering with Computers*, vol. 38, no. 4, pp. 3399–3416, Aug. 2022. [Online]. Available: https://doi.org/10.1007/s00366-021-01330-w

[120] M. Yousefzadeh and I. Battiato, "High order ghost-cell immersed boundary method for generalized boundary conditions," *International Journal of Heat and Mass Transfer*, vol. 137, pp. 585–598, Jul. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0017931018357636

[121] E. Haines, "Point in polygon strategies," in *Graphics gems IV*. USA: Academic Press Professional, Inc., 1994, pp. 24–46.

[122] J. C. Vassberg and A. Jameson, "In Pursuit of Grid Convergence for Two-Dimensional Euler Solutions," *Journal of Aircraft*, vol. 47, no. 4, pp. 1152–1166, Jul. 2010, publisher: American Institute of Aeronautics and Astronautics. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.46737

[123] B. Thwaites, "Approximate calculation of the laminar boundary layer," *Aeronautical Quarterly*, vol. 1, no. 3, pp. 245–280, 1949.

[124] M. R. Head, "Entrainment in the turbulent boundary layer," Aeronautical Research Council, Technical Report R&M 3152, 1958.

[125] H. Ludwieg and W. Tillmann, "Untersuchungen über die wandschubspannung in turbulenten reibungsschichten," *Ingenieur-Archiv*, vol. 17, no. 4, pp. 288–299, 1949.

[126] R. Michel, "Étude de la transition sur les profils d'aile, Établissement d'un critère de détermination du point de transition et calcul de la traînée de profil incompressible," ONERA, Technical Report 1/1578A, 1951.

[127] M. Mochizuki, "Smoke observation on boundary layer transition caused by a spherical roughness element," *Journal of the Physical Society of Japan*, vol. 16, no. 5, pp. 995–1008, May 1961.