UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

Sottomessa al Dipartimento di
Ingegneria Meccanica e Idustriale

Dottorato di Ricerca in
Ingegneria Meccanica e Industriale
SSD ING-IND06

CICLO XXXVII

An adaptive high-order isogeometric discontinuous Galerkin method for the Reynolds-averaged Navier-Stokes equations with general equations of state

Dottorando
Daniel Bulgarini

Supervisore
Prof. Antonio Ghidoni

Abstract

Questa tesi presenta un metodo di Galerkin discontinuo per la discretizzazione delle equazioni di Navier-Stokes mediate alla Reynolds (RANS), basata sull'analisi isogeometrica con NURBS, applicata sia a regimi di gas ideale che reale. L'uso delle NURBS consente di risolvere le equazioni direttamente sulla geometria esatta, come definita in un ambiente di progettazione assistita al computer (CAD), senza essere influenzati dall'accuratezza della discretizzazione geometrica, ovvero dal numero di elementi della mesh. Questo approccio è integrato con un algoritmo di adattazione anisotropa in h, che permette di raffinare una mesh inizialmente grossolana in base all'accuratezza del campo di soluzione, valutata secondo diversi criteri di raffinamento. Grazie alle proprietà delle NURBS, la suddivisione della mesh può essere implementata direttamente, senza necessità di ulteriori modifiche al solutore, poiché la geometria è sempre preservata esattamente. Le equazioni RANS sono accoppiate con un modello di turbolenza Spalart-Allmaras modificato, e le capacità predittive del solutore sono migliorate grazie all'implementazione di equazioni di stato cubiche e multiparametro, per considerare le caratteristiche di gas in regime non ideale. Il solutore è stato validato con successo su una serie di casi test di riferimento che includono regimi non viscosi e turbolenti, subsonici e supersonici. L'algoritmo di adattazione in h viene descritto in dettaglio, confrontando diversi criteri di raffinamento e analizzando l'impatto del raffinamento anisotropo sull'efficienza complessiva. Questo lavoro esplora inoltre il potenziale dell'analisi isogeometrica nell'ottimizzazione di forma per flussi turbolenti, grazie alla possibilità di modificare direttamente la geometria agendo sulla definizione NURBS anziché sui nodi della mesh. Un approccio di programmazione quadratica sequenziale è utilizzato per risolvere problemi di ottimizzazione con vincoli di uguaglianza. Viene eseguita un'ottimizzazione basata sul gradiente, mostrando risultati promettenti che evidenziano i vantaggi di formulare il problema di ottimizzazione rispetto alla posizione dei punti di controllo NURBS.

Supervisore di tesi: Antonio Ghidoni
Titolo: Professore Ordinario di Meccanica dei Fluidi

Abstract

This thesis presents a high-order discontinuous Galerkin discretization for the Reynolds-averaged Navier-Stokes (RANS) equations within a NURBS-based isogeometric analysis framework for both the ideal and real gas regime. The use of NURBS allows to solve the equations on the exact geometry, as defined in a computer-aided design environment, which is not influenced by the accuracy of the geometry discretization, i.e., by the number of mesh elements. This approach is coupled with an anisotropic h-refinement algorithm, which allows to refine an initial coarse mesh according to the accuracy of the solution field, evaluated with different refinement criteria. NURBS properties enable a straightforward implementation of the mesh subdivision without the need of further manipulations since the geometry is always exactly preserved. The RANS equations are coupled with a modified Spalart-Allmaras turbulence model, and the prediction capabilities of the solver are enhanced by implementing a cubic and multiparameter equations of state to account for nonideal flow characteristics. The solver is successfully validated over a series of benchmark test cases ranging from inviscid to turbulent, from subsonic to supersonic regimes. The implemented h-refinement procedure is detailed, comparing different refinement criteria, and investigating the impact of the anisotropic refinement on the overall efficiency. This work also investigates the potential of isogeometric analysis in shape optimization for turbulent flows, since a direct morphing of the geometry is allowed by acting directly on the NURBS definition rather than on the mesh nodes. A sequential quadratic programming approach is used to solve the equality-constrained optimization problem. A gradient-based optimization is performed, showing promising results that put in evidence the advantages of solving an optimization problem with respect to the position of NURBS control points.

Thesis supervisor: Antonio Ghidoni
Title: Full Professor of Fluid Mechanics

Acknowledgments

I would like to express my deepest appreciation to my advisor, Professor Antonio Ghidoni, whose guidance and advice have been indispensable in the development of this work. Thanks to him, I had the freedom to experiment and pursue ideas and methods while receiving constant support, especially in solving the most complex problems.

A special thanks also goes to Professor Krzysztof J. Fidkowski, who mentored me during my time at the University of Michigan. He shared his knowledge with me, helping me grow as a researcher and bringing out the best in me. I recall with great fondness the lively exchange of ideas with him and other brilliant students at the U-M Aerospace Engineering department, especially Miles and Alex.

I am also grateful to Dr. David Pasquale, whose drive for innovation and technical expertise in turbomachinery inspired me to always strive for high-quality research, going beyond theoretical applications and thinking outside the box.

I would also like to thank Professors Stefano Rebay and Francesco Bassi, whose advice and personal insights played a crucial role in the implementation of numerical methods and the development of this work.

I acknowledge Dr. Gianmaria Noventa for the fruitful conversations and contributions. I also extend my thanks to Edoardo for his input and reflections, whose work also contributed to the development of this thesis.

In addition, I would like to thank my readers, Professors Alessandro Colombo and Nicoletta Franchina, for their valuable feedback on both the form and content of the thesis draft.

Finally I would like to thank my parents, Enza and John, my siblings Nicole and Fabio, my grandmothers Rita and Irene, my aunt Anna, my partner Lisa and all the people close to me for your love and the support shown in these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computational Fluid Dynamics (CFD) has become a fundamental tool in modern industrial applications, playing a key role in both the analysis and design of systems. Its ability to simulate complex flow phenomena complements physical experiments, reducing the need for costly prototypes. Continuous advances in computational power have enabled CFD to evolve beyond traditional methods, driving the development and adoption of more accurate and efficient algorithms. As industries such as aerospace, automotive, and naval engineering push the limits of performance and efficiency, there is increasing interest to achieve highly accurate flow predictions, which in turn has highlighted the limitations of conventional numerical methods.

1.1 Motivations

To evaluate the current capabilities and limitations of CFD in practical aerodynamics, the American Institute of Aeronautics and Astronautics (AIAA) began the High Lift Prediction Workshops (HLPW). These workshops, beginning in 2010, are designed to assess the effectiveness of CFD tools in predicting maximum lift characteristics in various high-lift configurations in industry-relevant applications. These workshops have the objective of evaluating the impact of spatial discretization and grid resolution on simulation results, focusing on flows with Reynolds numbers ranging from $\approx 4 \div 15$ millions [1, 2, 3, 4].

The results of the workshops consistently reveal that the uncertainties inherent in CFD methods present significant challenges. In terms of lift results, the workshops demonstrated that lift predictions, especially near maximum lift, often show considerable spread among CFD results, indicating variability between participants and grid configurations. For example, a common conjecture is that the CFD predictions of lift at high angles of attack show an oscillating behavior, that is, simulations overpredict and underpredict lift near stall with respect to experimental

value. Moreover, it is stated how the influence of the mesh is still quite dominant for high-lift problems.

Such findings highlight the limitations of current CFD methods in delivering reliable high-accuracy predictions. Innovative high-order and adaptive techniques offer a promising direction for future work. Addressing these challenges requires the development of accurate methods capable of resolving multiscale flow phenomena while efficiently employing computational resources, increasing the reliability of CFD for aerodynamic design.

The objective of this work is to develop a high-order, adaptive method for the simulation of two-dimensional turbulent flows, that uses the exact geometry information from the Computer Aided Design (CAD), allowing for more accurate results and direct geometry manipulations, such as shape morphing and local refinements.

1.2 Previous work

1.2.1 Discontinuous Galerkin method

In response to the growing demand for accuracy, high-order numerical methods, including discontinuous Galerkin (dG) methods, have gained significant importance [5, 6, 7, 8, 9, 10, 11]. Bassi et al. [5] discuss the use of dG methods to solve the Reynolds-averaged Navier-Stokes equations (RANS) with a $k-\omega$ turbulence model. The authors describe a dG approach for advection-diffusion problems and the handling of viscous terms. Numerical results validate the approach with simulations of turbulent boundary layers over flat plates and turbine blade wakes, demonstrating the method's stability, accuracy, and grid independence. Luo et al. [6] introduce a dG method using a Taylor basis to simulate compressible flows on arbitrary grids. The method represents the numerical solution using a Taylor series expansion in cell centroids, allowing compatibility with both cell-centered and vertex-centered finite volume (FV) schemes, since both can be viewed as special cases of this approach. Numerical experiments, such as shock tube and cylinder flows, are tested to establish the accuracy and efficiency of the dG method compared to the finite volume. Bassi et al. [7] study focuses on very high-order dG methods for turbulent transonic flows for aeronautical configurations. Key features include a directional shock-capturing technique to handle flow discontinuities and a description of solid-wall boundary conditions for the $k-\omega$ turbulence model. The results on different wings geometries demonstrate the ability of the dG method to capture shocks and provide accurate results even on coarse grids, showing the potential of dG to provide accurate solutions in complex aerodynamic contexts. Uranga et al. [8] investigate the use of a high-order dG method with Implicit Large-Eddy Simulation (ILES) to predict the laminar-to-turbulent transition in low Reynolds number flows. The study

demonstrates the ability of the dG method to resolve transitional structures such as laminar separation bubbles and capture transition phenomena. Garai et al. [9] examine scale-resolving simulations of bypass transition in a high-pressure turbine cascade using a high-order spectral element dG method. The work demonstrates the capability of dG methods to capture the features of transition and turbulence in turbomachinery applications. The work of de la Llave Plata et al. [10] explores the use of the dG method for Direct Numerical Simulation (DNS) and Large Eddy Simulation (LES) in wall-bounded turbulent flows. The authors highlight the benefits of the dG method in terms of spectral accuracy, flexibility for high-order convergence, and effective subgrid-scale modeling in LES, particularly with local *hp*-adaptation to capture complex flow structures and improve computational efficiency. Bassi et al. [11] present a *p*-adaptive, matrix-free dG method designed for ILES of incompressible transitional flows. Their results demonstrate significant reductions in computational cost without compromising accuracy.

These methods offer substantial advantages, such as flexibility in mesh handling and the ability to naturally incorporate *h/p* adaptive strategies [12, 13]. Furthermore, their compact stencils make them highly suitable for parallel computing environments, allowing efficient use of computational resources. However, this increased accuracy comes at a higher computational cost compared to traditional FV methods [14], which has slowed the widespread adoption of dG methods in industrial CFD.

1.2.2 Isogeometric analysis

One of the persistent challenges in CFD remains the mesh generation, particularly in scenarios that require complex geometries. Generating a high-quality mesh can be both time-consuming and computationally expensive, especially for aircraft, automotive, and ship design. The difficulty is further compounded by the fact that once a mesh is generated, its connection to the original Computer-Aided Design (CAD) model is often lost, limiting the effectiveness of adaptive mesh refinement, an essential technique for resolving intricate flow features in regions not known a priori.

IsoGeometric Analysis (IGA), introduced in the early 2000s, offers a promising approach to address these challenges. Using CAD basis functions, such as Non-Uniform Rational B-Splines (NURBS), IGA allows seamless integration of geometry representation and solution approximation [15]. Unlike traditional methods that rely on discrete geometric approximations, IGA maintains the exact geometry throughout the analysis, which is critical for applications involving shape optimization [16] and mesh refinement [17]. The ability of IGA to directly link the simulation to CAD models has great potential for industries that require high-fidelity simulations and efficient design cycles [18]. In the literature, IGA has been uti-

lized primarily in the field of structural mechanics to address elliptic or parabolic partial differential equations, with limited applications in CFD [19, 20]. Bazilevs et al. [19] introduce a NURBS-based isogeometric variational multiscale approach for wall-bounded turbulent flows, with weakly imposed Dirichlet boundary conditions to simplify implementation. The approach demonstrates robustness and accuracy on coarse meshes, showing potential for complex applications. Evans and Hughes [20] developed a divergence-conforming B-splines for the unsteady incompressible Navier–Stokes equations. This method provides pointwise divergence-free solutions, ensuring mass conservation and enabling balance laws for momentum, energy, vorticity, and helicity. The study demonstrates the ability of the method to simulate benchmark flows like the Taylor-Green vortex, indicating the potential for broader applications.

1.2.3 IGA-based discontinuous Galerkin method

The first attempts to couple dG and NURBS (IGA-dG) can be found in Sevilla et al. [21], Silveira et al. [22] and Zhang et al. [23]. Sevilla et al. [21] propose the NURBS-Enhanced Finite Element Method (NEFEM). NEFEM improves the accuracy of finite elements method (FEM) for problems with curved boundaries by using exact boundary representations. Silveria et al. [22] focus on the treatment of NURBS-based curved boundary representation. Zhang et al. [23] present an IDA-dG approach to solving elliptic equations on 3D surfaces with multiple NURBS patches. The method reduces geometric error and adapts to nonconforming patches, supporting both h- and p-refinement. Langer and Toulopoulos [24] present an IGA-dG method to solve linear diffusion with discontinuous diffusion coefficients. The report provides an a priori error analysis that shows optimal convergence rates of the discretization with respect to the error norm. Chan and Evans [25] develop a multipatch IGA-dG method for wave propagation where, as a key feature, a weight-adjusted approximation method is introduced to simplify the inversion of the mass matrix in curved domains. Yu et al. [26, 27] developed an adjoint-based adaptive IGA-dG solver for the Euler equations, where an error estimation procedure is used to improve the estimation of outputs of interest. As a step forward, Duvigneau et al. [28, 29] proposed a NURBS-based dG solver for compressible steady/unsteady Euler and Navier-Stokes equations. The approach is based on a fully isogeometric formulation to simulate laminar flows, demonstrating the effectiveness of the metohd beyond inviscid flow simulations.

1.2.4 IGA-dG shape optimization

The direct link to the CAD definition made available by IGA plays a crucial role in shape optimization procedures, where a manipulation of the geometry can be

applied directly to the original definition without any further approximation. Wang et al. [30] present an adjoint-based airfoil shape optimization algorithm using an IGA-dG method for compressible Euler equations. The airfoil is parameterized using B-spline curves, with specific control points serving as design variables for a gradient-based optimization, where the gradient is computed from the adjoint solution. Pezzano et al. [31] introduce a CAD-consistent aerodynamic optimization approach for airfoil design. A Bayesian optimization framework is employed to optimize NURBS control points under single- or multi-objective functions, demonstrating high parallelization efficiency.

1.2.5 Real gas modeling

The study of real gas effects in CFD is particularly important in high pressure and high temperature regimes near saturation, where deviations from classical gas dynamics arise, such as the suppression of shock waves. Real gas modeling often involves articulated equations of state (EoS) to account for complex molecular structures. The focus on nonideal compressible fluid dynamics (NICFD) has intensified because of its engineering applications in the conversion of renewable and waste energy sources. These include systems such as organic Rankine cycles (ORCs), power plants utilizing supercritical carbon dioxide cycles, combustors that operate with supercritical fluids, gas liquefaction processes, carbon capture and storage (CCS) systems, and heat pump technologies.

The coupling of an accurate CFD solver with sophisticated thermodynamic models has been investigated, mainly in the finite-volume framework [32, 33, 34, 35]. However, the increasing computational power and the higher accuracy expected by industry motivates the recent interest in higher-order accurate solvers for NICFD, especially within a dG framework [36, 37].

1.3 Thesis overview

This thesis presents a further improved isogeometric discontinuous Galerkin solver for the efficient solution of transonic turbulent flows. In particular, this work makes the following contributions:

- an isogeometric dG discretization of the RANS equations and Spalart-Allmaras turbulence model based on NURBS;
- a geometrically consistent anisotropic h-adaptation algorithm to improve the accuracy and computational cost of the solver;
- a gradient-based shape optimization procedure using sensitivities provided by the adjoint method;

- an implementation of a real-gas model for accurate turbomachinery applications.

This thesis begins with an introduction to NURBS in Chapter 2. Here, the mathematical framework and geometric representation capabilities of NURBS are detailed, setting up a basis for mesh generation and surface modeling crucial for isogeometric discontinuous Galerkin applications.

Chapter 3 introduces the discontinuous Galerkin framework, providing a comprehensive overview of the discretization of the RANS equations, the implementation of a modified Spalart-Allmaras turbulence model and a shock capturing technique. Information on the time integration scheme used here is also given.

Chapter 4 validates the accuracy and reliability of the solver by comparing the computational results with experimental data and literature results, ranging from inviscid to turbulent, from subsonic to transonic and supersonic flows.

Following validation, the focus shifts to the h-refinement techniques employed. Chapter 5 demonstrates how refining the grid locally in critical regions improves the accuracy of the results without excessively increasing computational costs, contributing significantly to the efficiency of the solver.

Chapter 6 presents a gradient-based shape optimization procedure, employing the discrete adjoint method for gradient calculation and the sequential quadratic programming approach to minimize an objective function. The approach is tested in aerodynamic optimization cases, illustrating its ability to act on quantities of interest such as lift and drag.

Chapter 7 is dedicated to real gas modeling using the Peng-Robinson equation of state in the context of a turbine cascade. This model addresses deviations from ideal gas behavior, particularly under high-pressure, high-temperature conditions typical for turbomachinery applications in ORCs and large heat pumps, providing insight into the impact of real gas effects on flow features.

Finally, the thesis concludes in Chapter 8 with a summary of the research contributions, discussing the strengths and limitations of the developed methods, and suggesting future developments for NURBS-based dG solvers.

Chapter 2

NURBS representation and mesh generation

CAD software representations are commonly based on B-Splines and NURBS [18]. In this chapter, the mathematical foundations of NURBS are introduced, detailing how parametric curves and surfaces are defined through control points, weights, and knot vectors. These concepts are used to define the basis functions used in the solver for both geometric representation and solution approximation. Special focus is placed on the basis manipulations necessary to make them suitable for a discontinuous framework and on how all these concepts translate in the generation of the computational domain.

2.1 Bézier curves

When dealing with parametric CAD geometries, having a clear understanding of what Bézier curves are is first necessary. These notions become particularly important since they play a key role within discontinuous frameworks, as will be outlined in Sec. 2.3. Given $\mathbf{P}_0, \mathbf{P}_i, \ldots, \mathbf{P}_n$ control points in the space, a Bézier curve of degree p is defined as [38]:

$$\mathbf{C}(\xi) = \sum_{i=0}^{n} B_i^p(\xi)\mathbf{P}_i, \tag{2.1}$$

where B_i^p are referred to as Bernstein polynomials, and are expressed in the form:

$$B_i^p(\xi) = \binom{p}{i-1}\xi^i(1-\xi)^{p-i}, \tag{2.2}$$

with $\xi \in [0, 1]$. Equation (2.1) acts as a transformation from the parametric domain (ξ) to the physical domain (x, y), as shown in Fig. 2.1.

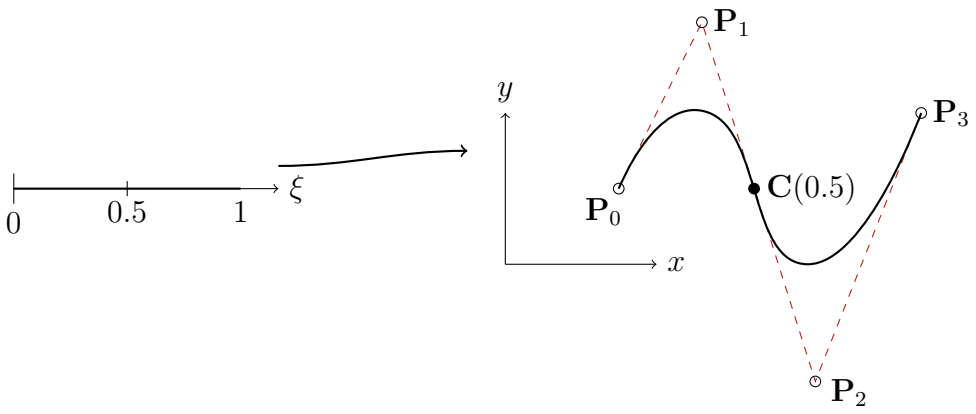


Figure 2.1: Cubic Bézier curve mapped from the parametric domain to the physical domain

The ease of definition and implementation of Bézier curves comes at the cost of a series of limitations, like the *i*) global control issue, where moving any control point affects the entire curve, making local adjustments difficult and unintuitive, *ii*) curve degree directly related to the number of control points, meaning that a high number of control points results in high-degree polynomials, that lead to numerical instability, *iii*) lack of local refinement, as adding control points always alters the entire curve, making it hard to refine just a specific section without affecting the rest, *iv*) inefficient representation for complex geometries, where many control points are needed to define intricate shapes, which in turn increases the curve degree and computational complexity, *v*) absence of a knot vector, which limits control over the curve's parameterization and makes it hard to adjust the curve's flexibility in specific regions.

2.2 NURBS representation

To overcome the limits of the Bézier representation, the use of B-splines is necessary. This class of curves can be used to represent complex geometries while maintaining a low degree and a low number of control points. Still, B-splines cannot be used to define conic sections, such as circles and ellipses. This is the reason why a further generalization is needed, which can be found in NURBS.

2.2.1 NURBS basis function

NURBS functions are a rational extension of B-Spline functions [39]. One-dimensional NURBS basis functions are defined over a parametric domain that

is subdivided by knots and are not nonzero on the entire interval. Let $\boldsymbol{\Xi} = [\xi_1, \ldots, \xi_m] \in \mathbb{R}^m$ represent a knot vector, a nondecreasing sequence of real numbers (knots). The half-open interval $[\xi_i, \xi_{i+1})$ defines the i-th knot span. In this work, only open knot vectors are considered, meaning the first and last knots have a multiplicity of $p+1$ (i.e., $\xi_1 = \ldots = \xi_{p+1}$ and $\xi_n = \ldots = \xi_{n+p+1}$), where p is the degree of the curve. In the physical space, this property ensures that the curve is both interpolating and tangent at its endpoints.

NURBS can be considered as a generalisation of B-Splines, where the i^{th} B-Spline function is multiplied by a weight $w_i \in \mathbb{R}^m$, thus obtaining the NURBS function of degree p, written as

$$R_i^p(\xi) = \frac{w_i N_i^p(\xi)}{\sum_{j=0}^{n} w_j N_j^p(\xi)}, \tag{2.3}$$

where

$$N_i^0(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{2.4}$$

$$N_i^p(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_i^{p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1}^{p-1}(\xi). \tag{2.5}$$

It is worth noting that a zero value is given where $0/0$ occurs. This is referred to as the Cox-de Boor recursion formula [40, 41].

2.2.2 NURBS curves

NURBS curves in $\mathbb{R}^d$ are constructed through a linear combination of rational basis functions. The coefficients of the combination are the so-called control points $\mathbf{P}_i$, which are in some way analogous to the nodal coordinates in finite element analysis. In general, control points are not interpolated by the NURBS curve, which can be defined as

$$\mathbf{C}(\xi) = \frac{\sum_{i=0}^{n} w_i N_i^p(\xi) \boldsymbol{P}_i}{\sum_{j=0}^{n} w_j N_j^p(\xi)} = \sum_{i=0}^{n} R_i^p(\xi) \mathbf{P}_i, \tag{2.6}$$

where $\mathbf{C}(\xi)$ are the coordinates of the curve points in the physical space. For the Bézier curves, Eq. (2.6) acts as a transformation from the parametric domain (ξ) to the physical domain (x, y). For example, each control point is associated with a unique basis function, whose trend defines the influence of the respective control point on the shape of the curve at a specific value of the parameter ξ. As a consequence, in a NURBS framework, each control point has a weight; by increasing the weight value, the curve will pass closer to that point, as visible in Fig. 2.2.

Rational geometries represented with basis expressed like in Eq. (2.6), hence with a common denominator, exhibit a graceful geometric explanation that produces efficient processing and compact data storage [39]. The basic idea resides in the

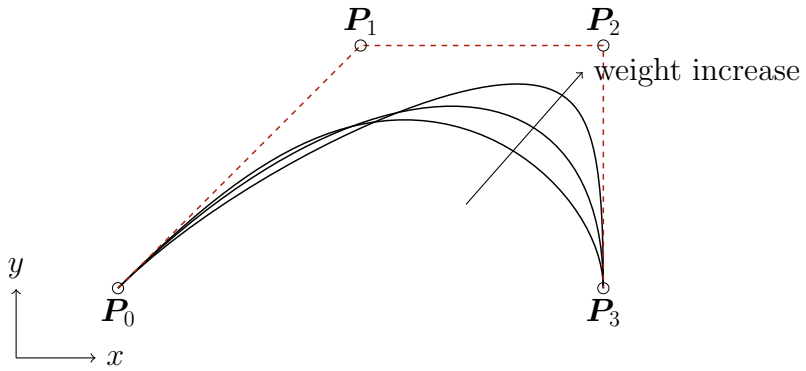


Figure 2.2: Comparison of NURBS curves by varying the weight of control point $\boldsymbol{P}_2$

use of homogeneous coordinates to define a rational curve in n dimensional space as a polynomial curve in $(n+1)$ dimensional space [42]. Starting with a control point in a three-dimensional Euclidean space, $\mathbf{P} = (x, y, z)$, one can rewrite $\mathbf{P}$ as $\mathbf{P}^w = (wx, wy, wz, w)$ in a four-dimensional space, with $w \neq 0$ being the weight assigned to it. Then $\mathbf{P}$ is obtained from $\mathbf{P}^w$ by dividing all coordinates by the weight, obtaining $\mathbf{P} = \left(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}, 1\right)$. This procedure acts as a mapping of $\mathbf{P}^w$ from the origin to the hyperplane $w = 1$, as shown in Fig. 2.3.

2.2.3 Derivatives of NURBS Curves

To compute the first derivative $\boldsymbol{C}'(\xi)$ of a NURBS curve as defined in Eq. (2.6), the quotient rule is applied [39]:

$$\boldsymbol{C}'(\xi) = \frac{\sum_{i=0}^{n} w_i N_i^{p\prime}(\xi)\boldsymbol{P}_i \cdot \sum_{j=0}^{n} w_j N_j^p(\xi) - \sum_{i=0}^{n} w_i N_i^p(\xi)\boldsymbol{P}_i \cdot \sum_{j=0}^{n} w_j N_j^{p\prime}(\xi)}{\left(\sum_{j=0}^{n} w_j N_j^p(\xi)\right)^2}, \tag{2.7}$$

where the derivative of the i-th B-Spline is defined as:

$$N_i^{p\prime}(\xi) = \frac{p}{\xi_{i+p} - \xi_i} N_i^{p-1}(\xi) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1}^{p-1}(\xi). \tag{2.8}$$

The expression in Eq.(2.7) can be simplified by defining the following quantities:

$$\boldsymbol{A}(\xi) = \sum_{i=0}^{n} w_i N_i^p(\xi)\boldsymbol{P}_i, \tag{2.9}$$

$$D(\xi) = \sum_{j=0}^{n} w_j N_j^p(\xi), \tag{2.10}$$

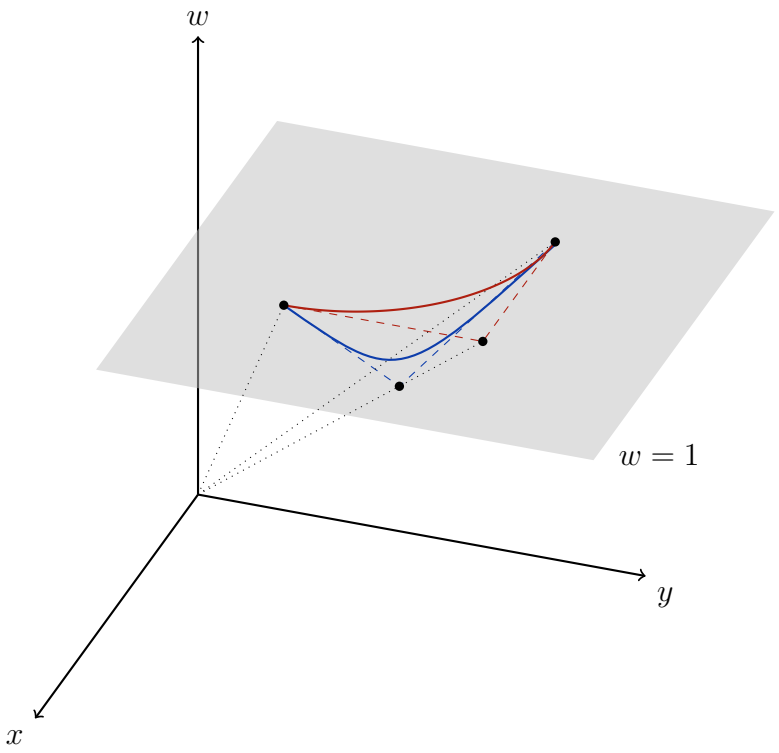


Figure 2.3: NURBS circular arc defined in homogeneous coordinates (blue) and its projection on the plane $w = 1$ (red)

where $\boldsymbol{A}(\xi)$ is the weighted sum of control points and $D(\xi)$ is the weighted sum of the basis functions. Then, the first derivative of the NURBS curve becomes:

$$\boldsymbol{C}'(\xi) = \frac{\boldsymbol{A}'(\xi)D(\xi) - \boldsymbol{A}(\xi)D'(\xi)}{(D(\xi))^2}, \tag{2.11}$$

where

$$\boldsymbol{A}'(\xi) = \sum_{i=0}^{n} w_i N_i^{p\prime}(\xi)\boldsymbol{P}_i, \tag{2.12}$$

$$D'(\xi) = \sum_{j=0}^{n} w_j N_j^{p\prime}(\xi). \tag{2.13}$$

For higher-order derivatives, a recursive application of the quotient rule is used. The second derivative $\boldsymbol{C}''(\xi)$ is given by:

$$\boldsymbol{C}''(\xi) = \frac{\boldsymbol{A}''(\xi)D(\xi) - 2\boldsymbol{A}'(\xi)D'(\xi) + \boldsymbol{A}(\xi)D''(\xi)}{(D(\xi))^2} - 2\frac{(\boldsymbol{A}'(\xi)D(\xi) - \boldsymbol{A}(\xi)D'(\xi))D'(\xi)}{(D(\xi))^3}, \tag{2.14}$$

where

$$\boldsymbol{A}''(\xi) = \sum_{i=0}^{n} w_i N_i^{p''}(\xi) \boldsymbol{P}_i, \tag{2.15}$$

$$D''(\xi) = \sum_{j=0}^{n} w_j N_j^{p''}(\xi). \tag{2.16}$$

For a general k-th derivative $\boldsymbol{C}^{(k)}(\xi)$, the expression is recursively differentiated, with each derivative involving both $\boldsymbol{A}$ and D terms up to order k, using the Leibniz rule. This results in:

$$\boldsymbol{C}^{(k)}(\xi) = \frac{\boldsymbol{A}^{(k)}(\xi) D(\xi) - \sum_{i=1}^{k} \binom{k}{i} \boldsymbol{C}^{(k-i)}(\xi) D^{(i)}(\xi)}{(D(\xi))^{k+1}}, \tag{2.17}$$

where

$$\boldsymbol{A}^{(k)}(\xi) = \sum_{i=0}^{n} w_i N_i^{p(k)}(\xi) \boldsymbol{P}_i, \tag{2.18}$$

$$D^{(k)}(\xi) = \sum_{j=0}^{n} w_j N_j^{p(k)}(\xi). \tag{2.19}$$

2.2.4 NURBS patches

NURBS surfaces generated by a tensor product of the one-dimensional basis functions are used to represent bidimensional physical domains. The NURBS surface of degree p in both parametric directions is defined as

$$\mathbf{S}(\xi, \eta) = \sum_{i=0}^{n} \sum_{j=0}^{m} R_{i,j}^{p}(\xi, \eta) \mathbf{P}_{i,j}, \tag{2.20}$$

where $\mathbf{S}$ are the coordinate of the surface points in the physical space, while $\mathbf{P}_{i,j}$ compose the so-called control grid. $R_{i,j}^{p}(\xi, \eta)$ is the two-parameter basis function obtained by the product of the one-dimensional i-th and j-th basis functions. An example of a set of quadratic rational basis functions is shown in Fig. 2.4. As for NURBS curves, Eq. (2.20) acts as a transformation from the parametric domain (ξ, η) to the physical domain (x, y), as shown in Fig. 2.5. It should be noted that the patch depicted in Fig. 2.5 also coincides with a Bézier patch of the same degree and the same control points.

In this work, the application is limited to two-dimensional cases, but the extension to NURBS volumes is trivial, due to the tensor nature of the basis functions.

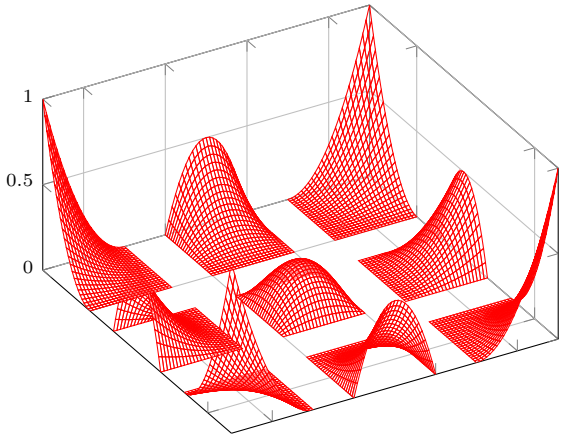


Figure 2.4: A set of quadratic rational NURBS basis functions

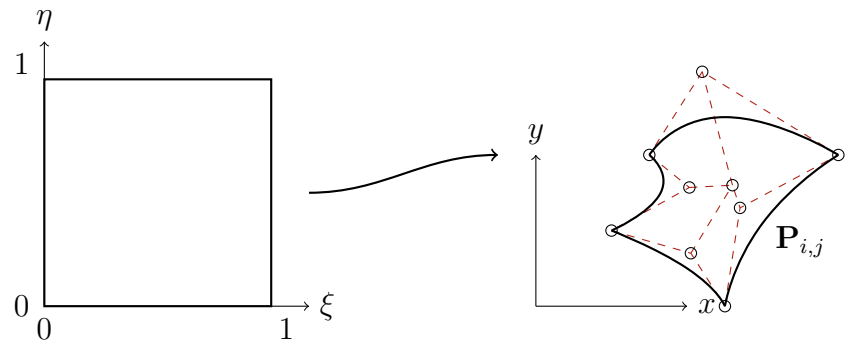


Figure 2.5: NURBS patch mapped from the parametric domain to the physical domain

2.3 NURBS implementation in a dG framework

NURBS basis functions are used to define a NURBS patch that represents exactly the physical domain. Generally, these basis are not suitable for DG methods, since they are not discontinuous across the knot spans, i.e. the elements. Therefore, the NURBS patch must be subdivided into a set of subpatches which are able to enforce discontinuities across the interfaces between elements.

2.3.1 Knot insertion

The subdivision of a curve or surface is achieved by applying a knot insertion procedure that does not change the original geometry. The possibility of inserting new knots into the definition of a NURBS curve or surface is useful for both the Bézier extraction (Sec. 2.3.2) and h-refinement (Sec. 5.1). For example, a NURBS curve

defined by the knot vector $\boldsymbol{\Xi} = [\xi_1, \ldots, \xi_m]$, with $n+1$ basis functions and control points, can be represented without altering its geometry using the knot vector $\overline{\boldsymbol{\Xi}} = \left[\xi_1, \ldots, \xi_k, \overline{\xi}, \xi_k + 1, \ldots, \xi_m\right]$. This newly defined knot vector is modified by introducing the additional knot $\overline{\xi}$. Given for simplicity a B-spline curve, the knot insertion procedure is defined as

$$x(\xi) = \sum_{i=0}^{n} N_i^p(\xi)\mathbf{P}_i = \sum_{i=0}^{n+1} \overline{N}_i^p(\xi)\overline{\mathbf{P}}_i, \tag{2.21}$$

where $\overline{N}_i^p$ are the p-th degree basis functions on $\overline{\boldsymbol{\Xi}}$, and $\overline{\mathbf{P}}_i$ the control points computed as

$$\overline{\mathbf{P}}_i = (1-\alpha_i)\mathbf{P}_{i-1} + \alpha_i \mathbf{P}_i, \;\; \alpha_i = \begin{cases} 1 & \text{if } i \leq k-p \\ \frac{\overline{\xi}-\xi_i}{\xi_{i+p}-\xi_i} & \text{if } k-p+1 \leq i \leq k \\ 0 & \text{if } i \geq k+1 \end{cases}. \tag{2.22}$$

Notice that inserting an already existing node in the knot vector decreases the regularity of the curve. Consider a curve of degree p at the knot ξ_k, which is characterized by a regularity C^{p-r}, where r represents the multiplicity of the knot ξ_k. If the same knot is inserted one more time, the curve does not change its shape, but its regularity decreases to C^{p-r-1}, since the knot multiplicity is increased by 1.

2.3.2 Bézier extraction

The knot insertion procedure is used to subdivide curves (and surfaces) into subpatches that exhibit discontinuities in the underlying basis functions. In order to obtain basis functions suitable for the discontinuous Galerking method, the multiplicity of each internal knot is increased to p, transforming each patch into a rational Bézier patch of degree p, which are the elements of the computational mesh. The transformation from continuous NURBS basis functions to discontinuous rational Bézier basis functions for a curve is shown in Fig. 2.6.

2.4 NURBS mesh generation

In this section, the mesh generator tools developed in this work are described. These tools can generate a series of different high-order meshes for a number of geometries such as channels, airfoils, and blades. The focus will be on the three main techniques utilized, namely *i*) lofting, *ii*) generation of Coons patches and *iii*) interpolation of multiblock linear meshes.

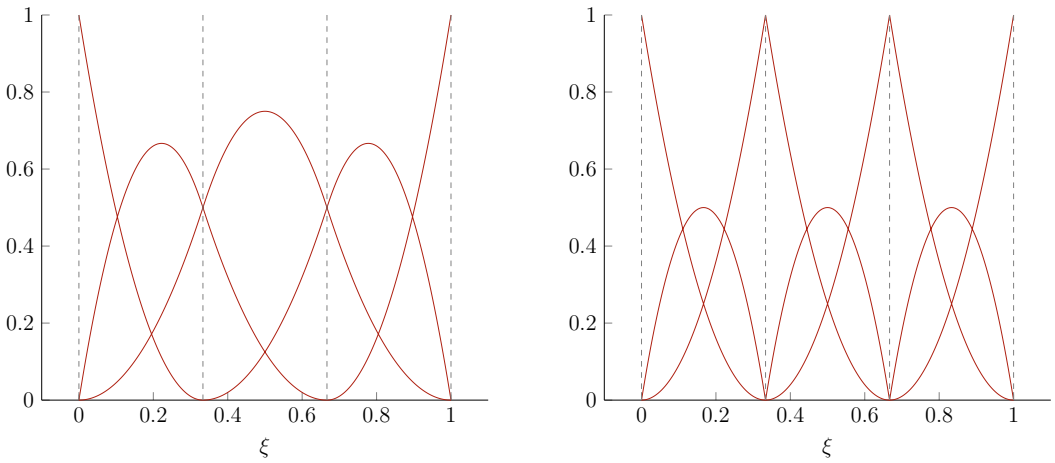


Figure 2.6: A single NURBS patch (left) defined by 5 basis functions is changed in 9 Bézier basis functions (right) with the extraction procedure

2.4.1 Airfoil geometry definition

The mesh generation process for isolated airfoils begins by defining the geometry based on a given distribution obtained considering a continuous generalization of the 4-digit series from the National Advisory Committee for Aeronautics (NACA) or by using a different parameterization, which will be detailed later. The NACA distributions taken into account here are modified for zero trailing-edge gap, where the camber and thickness functions are

$$z_c(x) = \begin{cases} \frac{z_{c,\max}}{1-z_{c,\mathrm{loc}}} \left[(1-2z_{c,\mathrm{loc}}) + 2z_{c,\mathrm{loc}}x - x^2\right] & \text{for } x < z_{c,\mathrm{loc}} \\ \frac{z_{c,\max}}{2z_{c,\mathrm{loc}}} \left[2z_{c,\mathrm{loc}}x - x^2\right] & \text{for } x \geq z_{c,\mathrm{loc}} \end{cases}, \tag{2.23}$$

and

$$t(x) = t_{\max}\left(0.2969\sqrt{x} - 0.126x - 0.3516x^2 + 0.2843x^3 - 0.1036x^4\right), \tag{2.24}$$

where $z_{c,\max}$ is the maximum camber, that is, the first digit of the NACA series divided by 100, $z_{c,\mathrm{loc}}$ is the location of the maximum camber, that is, the second digit divided by 10, and $t_{\max}$ is the maximum thickness, that is, the last two digits divided by 100. These expressions assume a unit chord airfoil with $0 \leq x \leq 1$.

The second parametrization allows for more flexibility in the camber and thickness profiles. It consists of polynomial expressions for both profiles, with bounding multiplicative envelopes:

$$z_c(x) = x(1-x)\left[c_0 + c_1 x + \cdots + c_{p_c} x^{p_c}\right], \tag{2.25}$$

$$t(x) = a\,\mathrm{abs}_{\mathrm{smooth}}\left[\sqrt{x}(1-x)\left(1 + t_1 x + \cdots + t_{p_t} x^{p_t}\right), 0.2x(1-x)\right], \tag{2.26}$$

where c_i are the coefficients of the order p_c camber polynomial, t_i are the coefficients of the order p_t thickness polynomial, and the smooth absolute value function is defined for positive b as

$$\mathrm{abs}_{\mathrm{smooth}}(a,b) = \begin{cases} |a| & \text{if } |a| \geq b, \\ \frac{a^2+b^2}{2b} & \text{otherwise} \end{cases}. \tag{2.27}$$

The coefficient a enforces a constant airfoil area, A:

$$\int_0^1 t(x)\,dx = A. \tag{2.28}$$

The mesh generator developed here handles both sharp and round trailing edges.

Turbine blade geometries are defined by a set of given points, without involving any parameterization. For other simpler cases, such as flat plates and ramps, the geometry definition is trivial.

2.4.2 Airfoil NURBS mesh generation

After defining the airfoil geometry, its points are approximated using a least-squares NURBS fitting procedure that ensures that the curve fits the unconstrained data points in a least-squares sense, while exactly interpolating the constrained points. A more in-depth study of this approach can be found in Appendix A. Figure 2.7 shows the result of this technique, where an airfoil NACA4412 is fitted using a $\mathbb{P}^3$ NURBS curve. Generally, the leading and trailing edges are treated as constrained points; a vertical direction is also imposed at the leading edge and at the trailing edge in the case of rounded trailing edge airfoils.

The next step is to define the farfield boundaries using NURBS curves of the same degree as the airfoil; then the computational domain is generated by lofting the airfoil and the farfield curves, resulting in one or more NURBS patches. To obtain a lofted surface of degree p, it is also necessary to create at least $p-1$ intermediate NURBS curves. Lofting, also known as skinning [39], is a method to construct a surface by blending a series of sectional curves together (a detailed description can be found in Appendix B). The obtained NURBS surface undergoes a Bézier extraction procedure, as detailed in Sec. 2.3, producing a high-order mesh suitable for dG methods. If necessary, multiple knot insertion procedures are used to generate a thicker boundary layer. An example of a $\mathbb{P}^3$ O-grid airfoil mesh is shown in Fig. 2.8, while a $\mathbb{P}^3$ C-grid is shown in Fig. 2.9.

2.4.3 NURBS geometry quality

Working with high-order schemes like the dG method requires the use of high-order geometries to ensure the solver accuracy [43] and to achieve faster convergence and

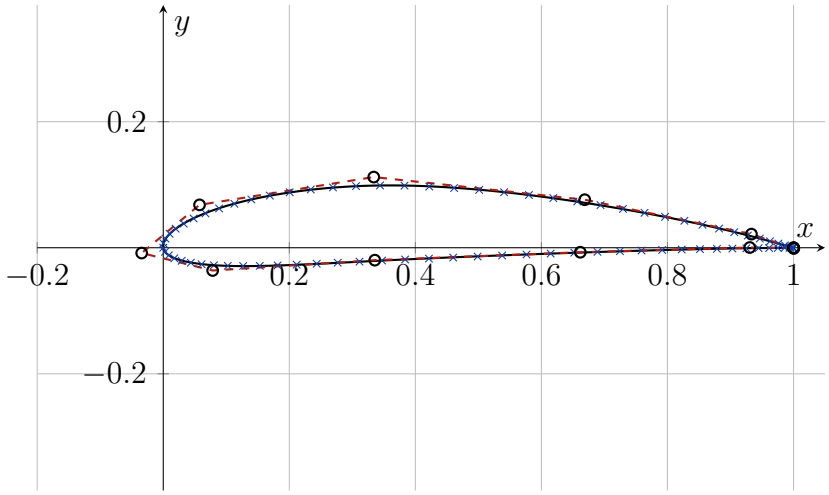


Figure 2.7: NACA 4412 airfoil $\mathbb{P}^3$ NURBS fitting, × sample points, —— NURBS curve, ∘ control points

faster decay of the integration error [22]. Silveira et al. [22] state that a faster error convergence in integrations can be achieved by using higher order NURBS. High-order NURBS also enhance the geometric approximation, resulting in the need of a lower number of control points. Figures 2.10 to 2.13 show the comparison of the curvature κ and its derivative κ' along the airfoil suction side for different numbers of control points. The curvature of a NURBS curve is computed with:

$$\kappa(\xi) = \frac{\|\boldsymbol{C}'(\xi) \times \boldsymbol{C}''(\xi)\|}{\|\boldsymbol{C}'(\xi)\|^3}, \tag{2.29}$$

while its curvature first derivative is evaluated as:

$$\kappa'(\xi) = \frac{\|\boldsymbol{C}'(\xi)\|^2 \left(\boldsymbol{C}'(\xi) \times \boldsymbol{C}'''(\xi)\right) - 3\left(\boldsymbol{C}'(\xi) \times \boldsymbol{C}''(\xi)\right)\left(\boldsymbol{C}'(\xi) \cdot \boldsymbol{C}''(\xi)\right)}{\|\boldsymbol{C}'(\xi)\|^5}. \tag{2.30}$$

The curvature and curvature derivative distributions show that, as expected, the $\mathbb{P}^2$ geometry presents discontinuities in both κ and κ', while the $\mathbb{P}^3$ geometry has a continuous curvature but discontinuous κ'. The distributions for higher-order geometries are continuous along the suction and pressure sides. In addition, Fig. 2.14 shows the approximation errors between the NURBS geometry and an exact NACA 0012 distribution for different degrees and an increasing number of control points. As expected, higher-order NURBS converge faster toward the machine error. Following the results presented here and the work of Silveira et al. [22], a $\mathbb{P}^3$ approximation is considered optimal to represent isolated airfoils.

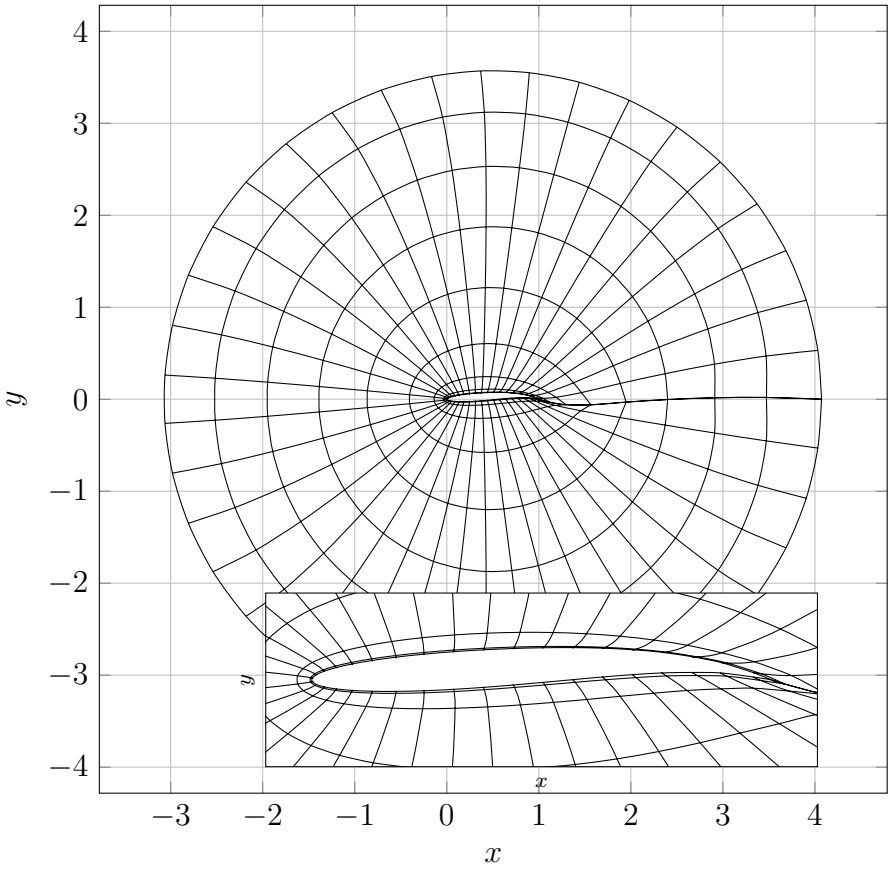


Figure 2.8: Airfoil O-grid mesh composed of a single $\mathbb{P}^3$ NURBS surface and close-up of the near-body region

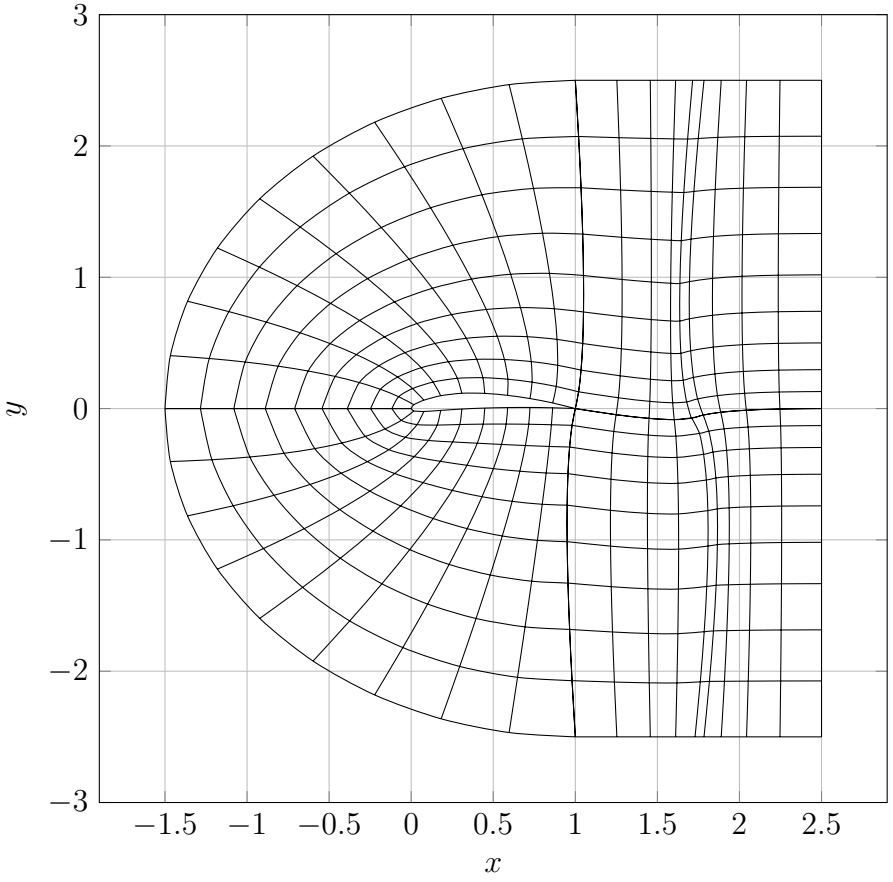


Figure 2.9: Airfoil C-grid mesh composed of four $\mathbb{P}^3$ NURBS surfaces

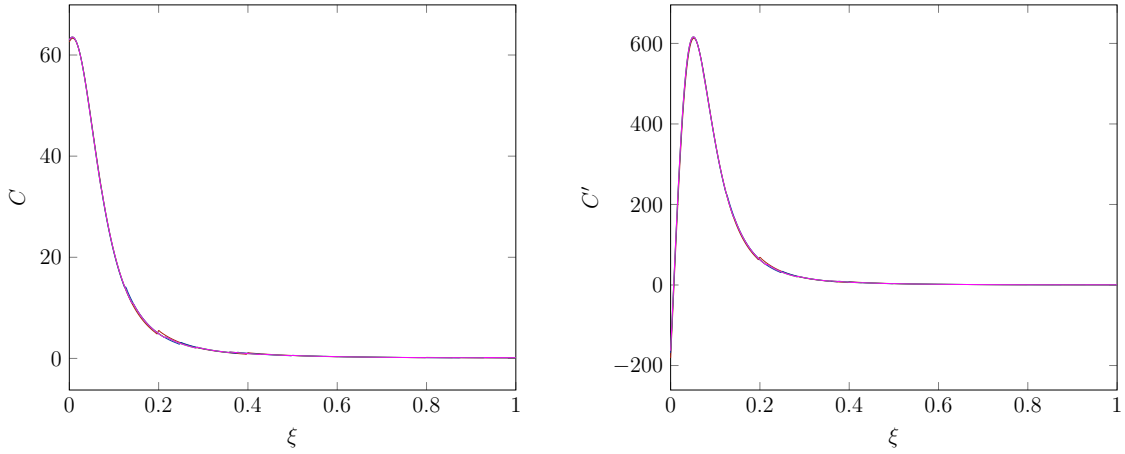


Figure 2.10: NACA0012. Comparison of the curvature C (left) and curvature derivative C' (right) along the suction side of a $\mathbb{P}^2$ NURBS representation for different numbers of control points (CPs); — 6 CPs, — 9 CPs, — 12 CPs, — 15 CPs

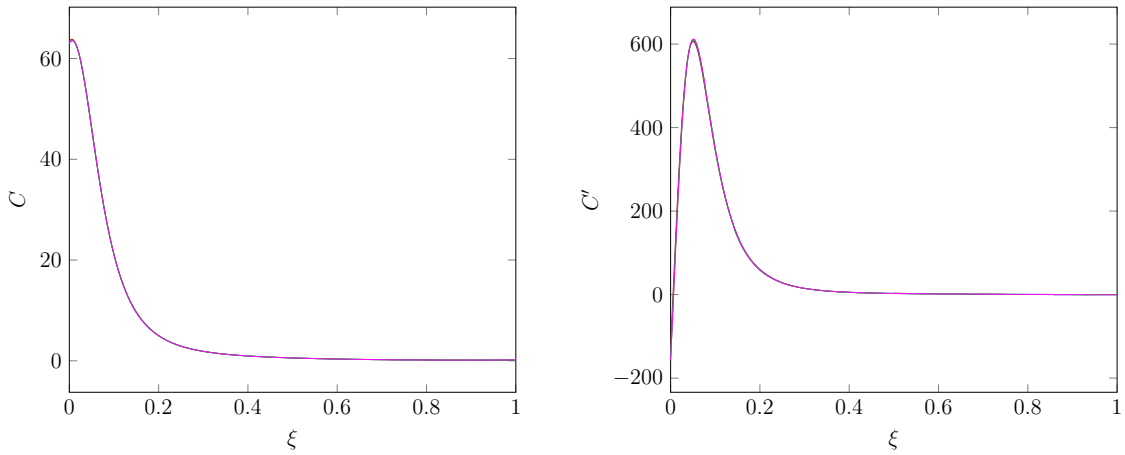


Figure 2.11: NACA0012. Comparison of the curvature C (left) and curvature derivative C' (right) along the suction side of a $\mathbb{P}^3$ NURBS representation for different numbers of control points (CPs); — 6 CPs, — 9 CPs, — 12 CPs, — 15 CPs

2.4.4 Channel-like mesh generation

The mesh generator tool described in Sec. 2.4.2 can also be used to create meshes easily defined by 4 boundaries of rectangular-shaped domains, such as those for the Ringleb flow problem or ramp geometries. The technique used to create these meshes is the discrete Coons patch approach (a detailed description can be found in Appendix C). An example of a $\mathbb{P}^3$ NURBS mesh obtained with this approach for the Ringleb flow is shown in Fig. 2.15.

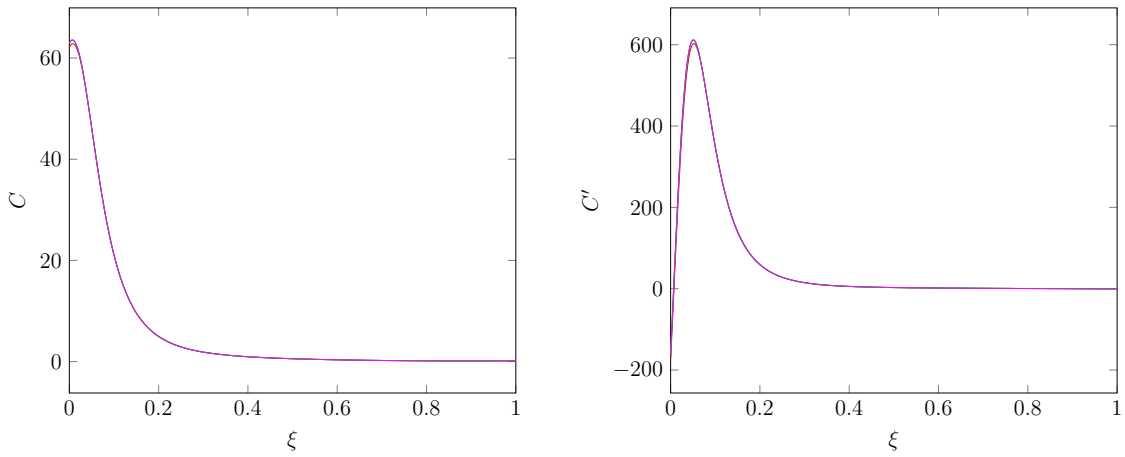


Figure 2.12: NACA0012. Comparison of the curvature C (left) and curvature derivative C' (right) along the suction side of a $\mathbb{P}^4$ NURBS representation for different numbers of control points (CPs); — 6 CPs, — 9 CPs, — 12 CPs, — 15 CPs

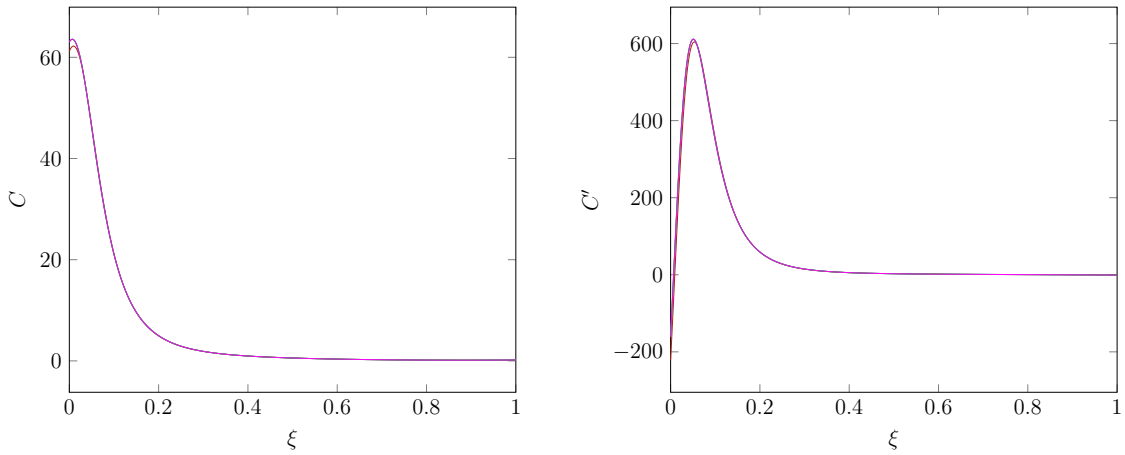


Figure 2.13: NACA0012. Comparison of the curvature C (left) and curvature derivative C' (right) along the suction side of a $\mathbb{P}^5$ NURBS representation for different numbers of control points (CPs); — 6 CPs, — 9 CPs, — 12 CPs, — 15 CPs

2.4.5 Turbine blade cascade mesh generation

A different approach is needed for turbine blade meshes. A multiblock linear mesh generator is first employed to create a suitable computational domain capable of handling periodic boundaries, since turbine blades are arranged in cascades. The resulting 9 blocks obtained by the linear generation are fitted with high-order NURBS of the desired degree. Each NURBS surface undergoes a Bézier extraction procedure as detailed in Sec. 2.3. An example of a mesh obtained with this procedure is shown

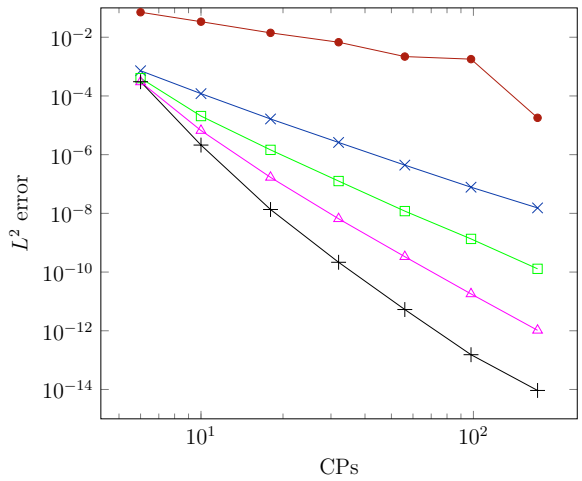


Figure 2.14: Discretization errors between the NURBS geometry and the analytical NACA 0012 distribution; $\mathbb{P}^1$, $\mathbb{P}^2$, $\mathbb{P}^3$, $\mathbb{P}^4$, $\mathbb{P}^5$

in Fig. 2.16.

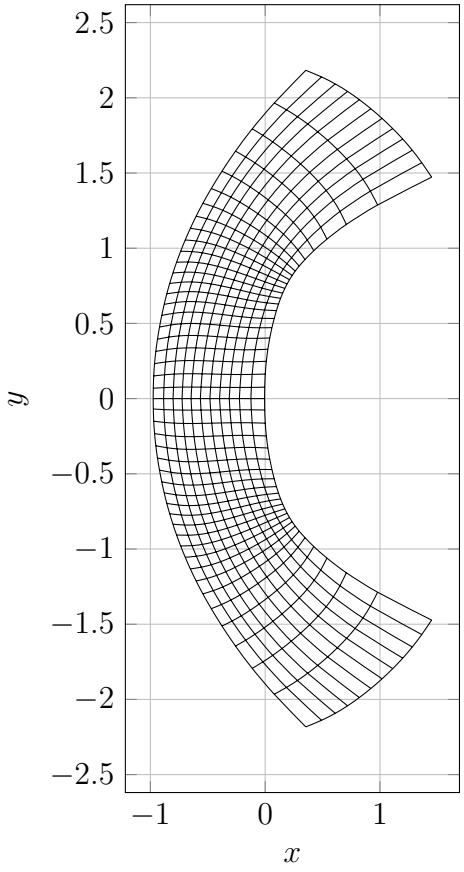


Figure 2.15: Ringleb flow mesh composed of a single $\mathbb{P}^3$ NURBS surface

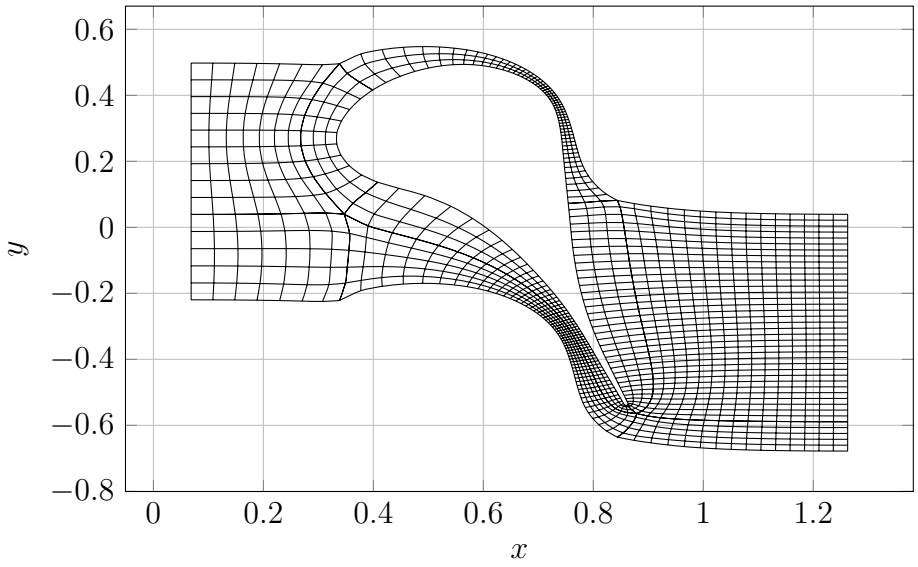


Figure 2.16: ORC turbine nozzle mesh composed of 9 $\mathbb{P}^2$ NURBS surfaces

Chapter 3

Numerical Framework

3.1 Isogeometric discontinuous Galerkin framework

The governing equations considered here are the Reynolds Averaged Navier-Stokes (RANS) equations for compressible flows, coupled with a one-equation Spalart-Allmaras turbulence model. In this chapter, these equations will be described, with particular attention to the implementation of the turbulence model (see Secs. 3.1.1 and 3.1.2). Section 3.1.3 details a standard dG discretization, while Sec. 3.1.4 describes the shock capture technique used. Section 3.1.5 is dedicated to the description of boundary treatment. Finally, Sec. 3.1.6 illustrates the time integration.

3.1.1 Governing equations

The flow model used in this work comprises the RANS equations coupled with the Spalart-Allmaras turbulence model for compressible flows [44]. The governing conservation laws are expressed as follows:

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\
\frac{\partial (\rho E)}{\partial t} + \nabla \cdot \left[(\rho E + P)\, \mathbf{u} \right] - \nabla \cdot \left(\rho \left(\nu + \nu_t \right) \left(\nabla \mathbf{u} - \nabla \mathbf{u}^{\intercal} \right) \mathbf{u} \right) - k \nabla T &= 0, \\
\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla P - \nabla \cdot \left(\rho \left(\nu + \nu_t \right) \left(\nabla \mathbf{u} + \nabla \mathbf{u}^{\intercal} \right) \right) &= \mathbf{0}, \\
\frac{\partial \tilde{\nu}}{\partial t} + \nabla \cdot (\mathbf{u} \tilde{\nu}) - \frac{1}{\sigma} \left[\nabla \cdot \left((\nu + \tilde{\nu}) \nabla \tilde{\nu} \right) \right] &= s,
\end{aligned} \tag{3.1}$$

where ρ represents the density, $\mathbf{u}$ denotes the velocity with components u and v, P is the pressure, ρE refers to the total energy, k is the thermal conductivity, T the

absolute temperature, ν the kinematic viscosity, ν_t the turbulent eddy viscosity, and $\tilde{\nu}$ is the viscosity-like variable from the Spalart-Allmaras model. The source term s is defined as

$$s = c_{b1}\tilde{S}\tilde{\nu} - c_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2 + \frac{c_{b2}}{\sigma}\left(\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}\right), \tag{3.2}$$

where d represents the distance from the wall, and $\tilde{S}$ is a production term defined in terms of the magnitude of the vorticity Ω and $\tilde{\nu}$ as follows:

$$\tilde{S} = \Omega + \frac{\tilde{\nu}}{\kappa^2 d^2}f_{v2}, \quad \Omega = \sqrt{2\boldsymbol{\omega}:\boldsymbol{\omega}}, \quad \boldsymbol{\omega} = \frac{\nabla\mathbf{u} - \nabla\mathbf{u}^\intercal}{2}. \tag{3.3}$$

The turbulent eddy viscosity is computed as $\nu_t = \tilde{\nu}f_{v1}$, while the closure functions and constants of the model are

$$\begin{gathered} \chi = \frac{\tilde{\nu}}{\nu}, \qquad g = r + c_{w2}\left(r^6 - r\right), \qquad r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2}, \\ f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \qquad f_{v2} = 1 - \frac{\chi}{1+\chi f_{v1}}, \qquad f_w = g\left[\frac{1+c_{w3}^6}{g^6 + c_{w3}^6}\right]^{\frac{1}{6}}, \end{gathered} \tag{3.4}$$

$$\begin{gathered} c_{b1} = 0.1355, \qquad c_{b2} = 0.622, \qquad c_{v1} = 7.1, \qquad \sigma = 2/3, \\ c_{w1} = \frac{c_{b1}}{k^2} + \frac{1+c_{b2}}{\sigma}, \qquad c_{w2} = 0.3, \qquad c_{w3} = 2, \qquad \kappa = 0.41. \end{gathered} \tag{3.5}$$

3.1.2 Spalart-Allmaras modifications

Evidences from literature [45, 46], show that the eddy viscosity decreases from a positive value to nearly zero within a layer that becomes thinner as the Reynolds number increases. For $Re \to \infty$, the equation permits weak solutions that lead to a discontinuous $\nabla\tilde{\nu}$ [44].

This phenomenon is closely tied to the numerical stability of high-order dG solvers. As observed by Persson et al. [47], and further acknowledged by Oliver and Darmofal [48], it can be attributed to the emergence of highly oscillatory or potentially unstable solutions, often characterized by negative values of $\tilde{\nu}$. Negative values for turbulent eddy viscosity lack of physical interpretation. Moreover, the closure functions (see Eq (3.4)) and closure constants (see Eq (3.5)) are valid only for positive values of $\tilde{\nu}$.

In this work, corrections for negative values of $\tilde{\nu}$ and for the closure function r are adopted, as described by Crivellini et al. [49]. Following this work, the source term of Eq (3.2) is rewritten as

$$s = \begin{cases} c_{b1}\tilde{S}\tilde{\nu} - c_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2 + \frac{c_{b2}}{\sigma}\left(\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}\right) & \chi \geq 0 \\ c_{b1}S\tilde{\nu}g_n + c_{w1}\left(\frac{\tilde{\nu}}{d}\right)^2 + \frac{c_{b2}}{\sigma}\left(\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}\right) & \chi < 0 \end{cases}, \tag{3.6}$$

where

$$g_n = 1 - \frac{10^3 \chi^2}{1+\chi^2}. \tag{3.7}$$

Moreover, Crivellini et al. [49] introduced, for convenience, the diffusion coefficient $\xi = \nu + \tilde{\nu}$, that appears in the form

$$\xi = \begin{cases} \nu\,(1+\chi) & \chi \geq 0 \\ \nu\left(1+\chi+\frac{1}{2}\chi^2\right) & \chi \leq 0 \end{cases}, \tag{3.8}$$

while the closure function r is rewritten as

$$r = \begin{cases} r_{max} & r^* < 0 \\ r^* & 0 \leq r^* < r_{max} \\ r_{max} & r^* \geq r_{max} \end{cases}, \quad r_{max} = 10^3, \quad r^* = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2}. \tag{3.9}$$

For all the turbulent computations presented here, the freestream turbulent viscosity is set to $\tilde{\nu}_\infty = 10^{-3}\nu_\infty$, as suggested by Crivellini et al. [49].

3.1.3 Space discretization

RANS and turbulence model equations can be written in conservation form as

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{F}_c(\mathbf{q}) + \nabla \cdot \mathbf{F}_v(\mathbf{q}, \nabla\mathbf{q}) + \mathbf{s}(\mathbf{q}, \nabla\mathbf{q}) = \mathbf{0}, \tag{3.10}$$

where $\mathbf{q} \in \mathbb{R}^m$ is the unknown solution vector of the m variables, $\mathbf{F}_c \in \mathbb{R}^m \otimes \mathbb{R}^d$ and $\mathbf{F}_v \in \mathbb{R}^m \otimes \mathbb{R}^d$ are the tensors that describe the convective and viscous flux functions, $\mathbf{s} \in \mathbb{R}^m$ is the source term vector, while d is the number of dimensions. As $d = 2$, the set of the $m = 5$ conservative variables employed is $\mathbf{q} = [\rho, \rho E, \rho u, \rho v, \tilde{\nu}]^\intercal$, while the components of the convective flux function $\mathbf{F}_c$ are

$$\mathbf{f}_c(\mathbf{q}) = \begin{bmatrix} \rho u \\ \rho u H \\ \rho u^2 + P \\ \rho u v \\ u\tilde{\nu} \end{bmatrix}, \quad \mathbf{g}_c(\mathbf{q}) = \begin{bmatrix} \rho v \\ \rho v H \\ \rho v u \\ \rho v^2 + P \\ v\tilde{\nu} \end{bmatrix}, \tag{3.11}$$

where $H = E + P/\rho$ is the total enthalpy. For ideal flows, $P = (\gamma - 1)\rho\left[E - (u^2+v^2)/2\right]$, where γ is the ratio between the specific heats of the fluid.

The components of the viscous flux function $\mathbf{F}_v$ are

$$\begin{aligned}
\mathbf{f}_v(\mathbf{q}, \nabla\mathbf{q}) &= \begin{bmatrix} 0 \\ (\mu+\mu_t)\left[u\left[2u_x+\lambda(u_x+v_y)\right]+v(v_x+u_y)+\gamma/PrE_x\right] \\ (\mu+\mu_t)\left[2u_x+\lambda(u_x+v_y)\right] \\ (\mu+\mu_t)\left[v_x+u_y\right] \\ \xi/\sigma\ \tilde{\nu}_x \end{bmatrix}, \\
\mathbf{g}_v(\mathbf{q}, \nabla\mathbf{q}) &= \begin{bmatrix} 0 \\ (\mu+\mu_t)\left[u(v_x+u_y)+v\left[2v_y+\lambda(u_x+v_y)\right]+\gamma/PrE_y\right] \\ (\mu+\mu_t)\left[v_x+u_y\right] \\ (\mu+\mu_t)\left[2v_x+\lambda(u_x+v_y)\right] \\ \xi/\sigma\ \tilde{\nu}_y \end{bmatrix},
\end{aligned} \tag{3.12}$$

where μ is the dynamic viscosity, μ_t is the turbulent dynamic viscosity, Pr is the Prandtl number and, under the Stokes hypothesis, $\lambda = -2/3$. The subscripts indicate the directional derivatives in either x or y direction.

The DG method is based on a weak formulation of the problem, which is obtained by multiplying Eq (3.10) by an arbitrary "test function" $\mathbf{v}$ and integrating over the domain Ω. The integrals are expanded into a summation over a collection of non-overlapping Bézier patches Ω_j, yielding

$$\sum_{\Omega_j}\left[\int_{\Omega_j}\mathbf{v}\frac{d\mathbf{q}}{dt}\,d\Omega+\int_{\Omega_j}\mathbf{v}\nabla\cdot\mathbf{F}(\mathbf{q},\nabla\mathbf{q})\,d\Omega+\int_{\Omega_j}\mathbf{v}\,\mathbf{s}(\mathbf{q},\nabla\mathbf{q})\,d\Omega\right]=\mathbf{0}\quad\forall\mathbf{v}, \tag{3.13}$$

where $\mathbf{F}(\mathbf{q},\nabla\mathbf{q}) = \mathbf{F}_c(\mathbf{q})+\mathbf{F}_v(\mathbf{q},\nabla\mathbf{q})$. By integrating by parts the flux contribution, Eq (3.13) can be rewritten as

$$\begin{aligned}
\sum_{\Omega_j}\Bigg[&\int_{\Omega_j}\mathbf{v}\frac{d\mathbf{q}}{dt}\,d\Omega+\oint_{\partial\Omega_j}\mathbf{v}\,\mathbf{F}(\mathbf{q},\nabla\mathbf{q})\cdot\mathbf{n}\,d\Gamma \\
&-\int_{\Omega_j}\nabla\mathbf{v}\cdot\mathbf{F}(\mathbf{q},\nabla\mathbf{q})\,d\Omega+\int_{\Omega_j}\mathbf{v}\,\mathbf{s}(\mathbf{q},\nabla\mathbf{q})\,d\Omega\Bigg]=\mathbf{0}\quad\forall\mathbf{v},
\end{aligned} \tag{3.14}$$

where $\partial\Omega_j$ is the boundary of the patch Ω_j, whose normal is $\mathbf{n}$. The continuous functions $\mathbf{q}$ and $\mathbf{v}$ are substituted by their discrete counterpart, $\mathbf{q}_h$ and $\mathbf{v}_h$, and Eq (3.14) can be rewritten as

$$\begin{aligned}
\sum_{\Omega_j}\Bigg[&\frac{d}{dt}\int_{\Omega_j}\mathbf{v}_h\mathbf{q}_h\,d\Omega+\oint_{\partial\Omega_j}\mathbf{v}_h\,\mathbf{F}(\mathbf{q}_h,\nabla\mathbf{q}_h)\cdot\mathbf{n}\,d\Gamma \\
&-\int_{\Omega_j}\nabla\mathbf{v}_h\cdot\mathbf{F}(\mathbf{q}_h,\nabla\mathbf{q}_h)\,d\Omega+\int_{\Omega_j}\mathbf{v}_h\,\mathbf{s}(\mathbf{q}_h,\nabla\mathbf{q}_h)\,d\Omega\Bigg]=\mathbf{0}\quad\forall\mathbf{v}_h,
\end{aligned} \tag{3.15}$$

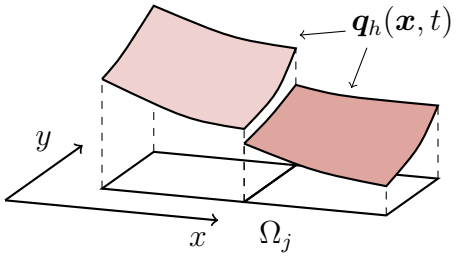


Figure 3.1: Solution $\boldsymbol{q}_h$ in the space of rational Bézier basis functions over two elements

In an isogeometric framework, the test function and the numerical approximation of the solution are based on rational Bézier basis functions of degree p as

$$\begin{aligned} \mathbf{q}_h = \mathbf{q}(\mathbf{x},t)_{h|\Omega_j} &= \sum_{i=0}^{n} \mathbf{Q}_i(t) R_i^p(\mathbf{x}), \\ \mathbf{v}_h = \mathbf{v}(\mathbf{x})_{h|\Omega_j} &= \sum_{i=0}^{n} \mathbf{V}_i R_i^p(\mathbf{x}) \;\; \forall \mathbf{x} \in \Omega_j, \end{aligned} \tag{3.16}$$

where the coefficients $\mathbf{Q}_i$ and $\mathbf{V}_i$ represent the degrees of freedom of the solution and of the test function in the element Ω_j. Figure 3.1 illustrates an example of a discontinuous solution over two elements. When computing the interface integrals of Eq (3.15), the flux terms lack of a unique definition due to the discontinuous function approximation. Therefore, it becomes necessary to replace the Navier-Stokes flux function $\mathbf{F}$ with a numerical flux function $\hat{\mathbf{F}}$. This numerical flux function depends on the interface states $\mathbf{u}^-$ and $\mathbf{u}^+$, which introduces a coupling between the unknowns of neighboring elements. In this work the Roe "approximate" flux [50] is used for the inviscid numerical flux function, while the numerical viscous flux is discretized with the BR2 scheme [43]. As described in Chapter 7, a generalized version of the Roe flux is employed (a detailed description can be found in Appendix D). Finally, taking into account Eq (3.16) and introducing the numerical flux function $\hat{\mathbf{F}}$, Equation (3.15) can be rewritten as

$$\begin{aligned} \sum_{\Omega_j} \frac{dQ_{j,i}}{dt} \int_{\Omega_j} R_i R_k \, d\mathbf{x} - \sum_{\Omega_j} \int_{\Omega_j} \frac{\partial R_k}{\partial x_n} \mathbf{F}\left(\mathbf{q}_h, \nabla_h \mathbf{q}_h + \mathbf{r}\left(\llbracket \mathbf{q}_h \rrbracket\right)\right) d\mathbf{x} \\ + \sum_{\partial\Omega_j} \int_{\partial\Omega_j} \llbracket R_k \rrbracket \; \hat{\mathbf{F}}\left(\mathbf{q}_h^{\pm}, \left(\nabla_h \mathbf{q}_h + \eta_{\partial\Omega_j} \mathbf{r}_{\partial\Omega_j}\left(\llbracket \mathbf{q}_h \rrbracket\right)\right)^{\pm}\right) d\Gamma \\ + \sum_{\Omega_j} \int_{\Omega_j} R_k \, \mathbf{s}\left(\mathbf{q}_h, \nabla_h \mathbf{q}_h + \mathbf{r}\left(\llbracket \mathbf{q}_h \rrbracket\right)\right) d\mathbf{x} = \mathbf{0}, \end{aligned} \tag{3.17}$$

where the operator $[\![\cdot]\!]$ is defined as the jump over the face $\partial\Omega_j$

$$[\![\cdot]\!] \stackrel{\text{def}}{=} (\cdot)\mathbf{n}^+_{\partial\Omega_j} + (\cdot)\mathbf{n}^-_{\partial\Omega_j}, \tag{3.18}$$

while $\mathbf{r}$ and $\mathbf{r}_{\partial\Omega_j}$ are the global and local lifting operators, respectively, $\eta_{\partial\Omega_j}$ is a user-defined coefficient, usually set to the number of neighboring elements. Figure 3.5 describes how normals and local frames are taken into account for a generic interface. It should be noted that integrals are solved in the parametric space by multiplying all terms inside these integrals by the determinant of the Jacobian matrix $\mathbf{J}_\Omega$ of the geometric transformation from the parametric to the physical space. The Jacobian matrix for a two-dimensional transformation is

$$\mathbf{J}_\Omega = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \sum_{i=0}^{n} \begin{bmatrix} \frac{\partial R_i}{\partial \xi}(\xi,\eta)x_i & \frac{\partial R_i}{\partial \eta}(\xi,\eta)x_i \\ \frac{\partial R_i}{\partial \xi}(\xi,\eta)y_i & \frac{\partial R_i}{\partial \eta}(\xi,\eta)y_i \end{bmatrix}. \tag{3.19}$$

Surface integrals are treated in the same manner, using the linear counterpart of the Jacobian matrix. Standard Gauss-Legendre quadrature rules are used. For a solution approximation of degree p and an integral of dimension d, a minimum of $(p+1)^d$ quadrature points are used. The impact of the number of quadrature points on the accuracy of the integrals computation was investigated. Initially, the integral of the non polynomial function $f(x,y) = \sin(x)\cos(y)$ over a square domain $[0,5]\times[0,5]$ represented with four $\mathbb{P}^5$ NURBS curves (see Fig. 3.2) is calculated with an increasing number of quadrature points. The error is calculated as $\frac{|\mathcal{F}-\hat{\mathcal{F}}|}{|\hat{\mathcal{F}}|}$, where $\mathcal{F}$ and $\hat{\mathcal{F}}$ are the computed and analytical integral value, respectively. Afterward, the integrals of the same non polynomial function $f(x,y) = \sin(x)\cos(y)$ and of the polynomial function $f(x,y) = x^2y^2$ are computed over a highly curved, distorted element Ω. This element is shown in Fig. 3.3 and is characterized by an average aspect ratio $AR \approx 6\times10^5$, a non constant thickness ranging between 10^{-6} and 10^{-7} mesh units. A curvature $\kappa = 6000$ mesh units is prescribed at $y = 0$ for the $\mathbb{P}^5$ NURBS curves used to represent the upper and lower element boundaries, yielding a high curvature value in that region. The analyzed element can be considered representative of the boundary layer elements needed for a turbulent simulation. Figure 3.4 shows the error convergence with respect to the number of quadrature points for both square and distorted domain. These tests demonstrate how the proposed method is able to correctly compute integrals with the Gauss-Legendre rules over high-order, distorted NURBS domain of non polynomial functions, which is a critical feature when solving turbulent flows on complex geometries.

3.1.4 Shock capturing technique

When dealing with flow discontinuities, dG methods require some form of stabilization to control numerical oscillations. This problem is solved following the approach

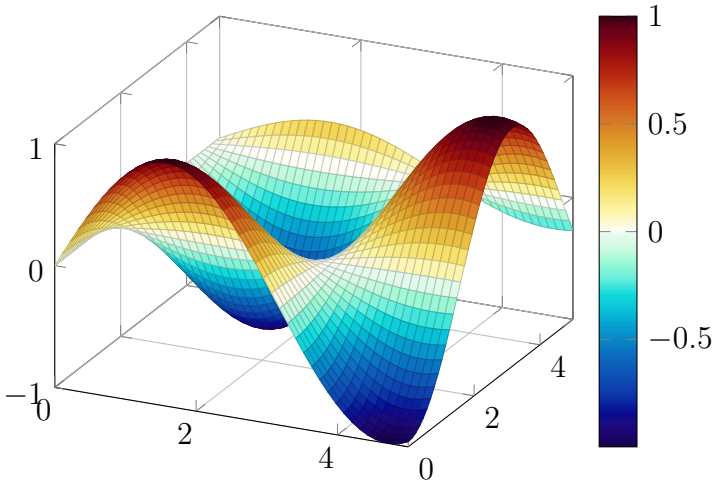


Figure 3.2: Function $f(x, y) = \sin(x)\cos(y)$ over the domain $[0, 5] \times [0, 5]$

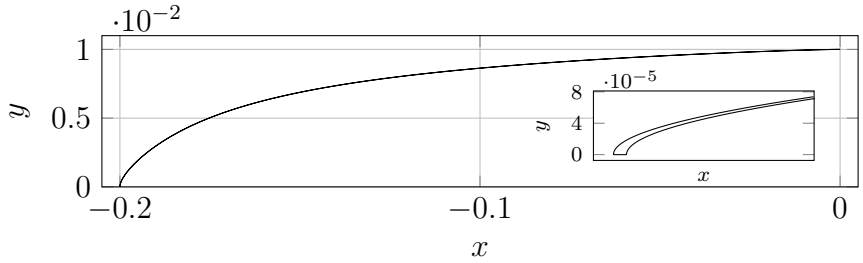


Figure 3.3: Details of the highly curved and distorted Ω element used for quadrature evaluation

proposed by Bassi et al. [7]. In this work, the divergence term d_p defined by Bassi et al. [7] is considered null. The shock-capturing term is added to Eq (3.17). In particular, an artificial diffusion contribution is explicitly introduced within each element $\Omega_j \in \Omega$, which is always and everywhere active, but introduces a numerical viscosity only where nonphysical oscillations occur. The contribution can be written as

$$\sum_{\Omega_j} \int_{\Omega_j} \epsilon_p \left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right) \left[(\mathbf{b} \cdot \nabla)\, \mathbf{v}_h\, (\mathbf{b} \cdot \nabla)\, \mathbf{q}_h\right] d\mathbf{x}, \tag{3.20}$$

where ϵ_p is the artificial viscosity defined as

$$\epsilon_p \left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right) = C h_{\Omega_j}^2 \frac{\left|s_p \left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right)\right|}{|p(\mathbf{q}_h)|}, \quad s_p \left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right) = \frac{\partial p(\mathbf{q}_h)}{\partial \mathbf{q}_h} \cdot \mathbf{s}_k \left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right), \tag{3.21}$$

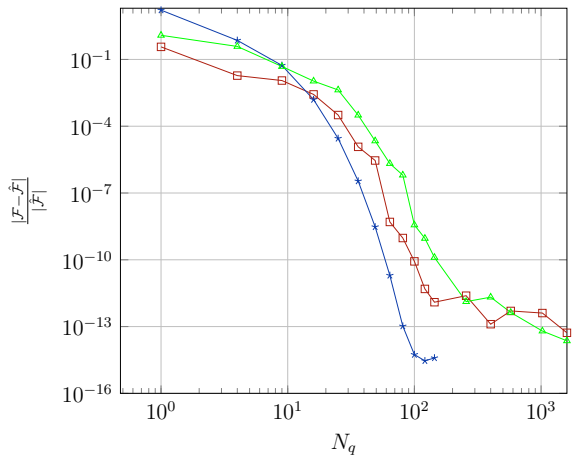


Figure 3.4: Area integral error convergence with respect to the number of quadrature points; $\int_0^5 \int_0^5 \sin(x)\cos(y)$, $\int_\Omega \sin(x)\cos(y)$, $\int_\Omega x^2 y^2$

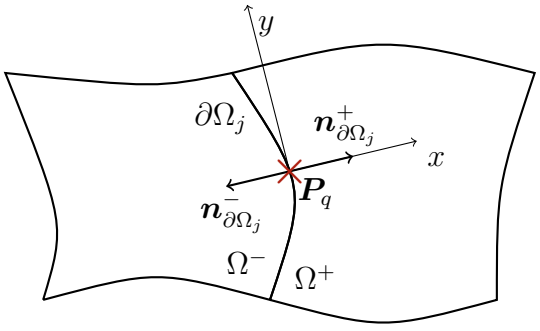


Figure 3.5: Normals and local frame at quadrature point $\boldsymbol{P}_q$ on internal interface $\partial\Omega_j$

while the direction along the pressure gradient $\mathbf{b}$ is computed as

$$\mathbf{b} = \frac{\nabla p(\mathbf{q}_h)}{|\nabla p(\mathbf{q}_h)| + \varepsilon}, \tag{3.22}$$

with ε small value of the machine precision order.

The function $\mathbf{s}_k\left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right)$ is obtained by solving the problem

$$\int_{\Omega_j} \mathbf{v}_h \mathbf{s}_k\left(\mathbf{q}_h^{\pm}, \mathbf{q}_h\right) \, d\mathbf{x} = \oint_{\partial\Omega_j} \mathbf{v}_h \left[\hat{\mathbf{F}}_{c,\mathbf{n}}\left(\mathbf{q}_h, \mathbf{q}_h^{+}, \mathbf{n}\right) - \mathbf{F}(\mathbf{q}_h) \cdot \mathbf{n}\right] \, d\Gamma, \tag{3.23}$$

where C is a user defined value, usually set to $C = 0.2$. Bassi et al. [7] define the element characteristic dimension h_{Ω_j} as

$$h_{\Omega_j} = \frac{1}{\sqrt{\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}}}, \tag{3.24}$$

where Δx and Δy are the maximum distance between element Ω_j nodes along the x and y direction, respectively. However, this is not a suitable option for a Bézier patch, since the control points usually do not interpolate the borders of the element. In this work, h_{Ω_j} is computed as the minimum distance between element corners, as suggested by Duvigneau [51].

3.1.5 Boundary treatment

The treatment of boundary conditions can be a complex task, as the required prescribed quantities often differ from the vector of discrete conservative variables $\boldsymbol{q}_h$ or from some of its components. In addition, a generalized set of boundary conditions must be determined to account for the effects of the real gas (see Chapter 7 for more details). At the boundary, the information provided involves nonlinear functions of conservative variables, denoted $\mathcal{B}_k(\boldsymbol{q}_h)$, for $k = 1, \ldots, p^{bc}$, with $p^{bc} \leq m$. These quantities could include, for instance, metrics such as pressure or the flow angle. Importantly, both the number of quantities p^{bc} that need to be specified and the specific characteristics of these quantities, exemplified by the explicit form of the functions $\mathcal{B}_k(\boldsymbol{q}_h)$, depend on the local flow conditions.

The determination of both the number of conditions and quantities that can be set involves examining the projection of the convective terms in a direction perpendicular to the boundary. This analysis reveals that the solution can be conceptualized as a superposition of m traveling waves, moving inward and outward, which are represented by nonlinear functions $\Gamma_k(\boldsymbol{q}_h)$, known as Riemann invariants for $k = 1, \ldots, m$. Since the boundary treatment for two-dimensional domains is

discussed, $m = 4$. These functions propagate at speeds dictated by the eigenvalues $\lambda_k(\boldsymbol{q}_h)$ (see Appendix D). At the boundary, the solution integrates information that travels outwards from and into the domain. Specifically, Riemann invariants associated with characteristic lines possess a space-time slope where $\lambda_k > 0$ if n denotes the outward unit normal vector. In contrast, the inward travel information requires an explicit prescription. Consequently, the number of elements that should be prescribed corresponds to the count of negative eigenvalues, where $\lambda_k < 0$.

The boundary state $\boldsymbol{q}_h^{bc}$ is, therefore, computed by combining the p^{bc} prescribed data, where p^{bc} is the number of negative eigenvalues with the $m - p^{bc}$ Riemann invariants associated to the positive eigenvalues, i.e. as the solution of the system:

$$\begin{cases} \mathcal{B}_k(\boldsymbol{q}_h^{bc}) = \mathcal{B}_k^{bc}, & k = 1, \dots, p \\ \Gamma_k(\boldsymbol{q}_h^{bc}) = \Gamma_k(\boldsymbol{q}), & k = p+1, \dots, m \end{cases}, \tag{3.25}$$

where $\mathcal{B}_k^{bc}$ denotes the prescribed value of the quantity $\mathcal{B}_k$, and $\boldsymbol{q}_h$ is the solution at the boundary (which is different from $\boldsymbol{q}^{bc}$ because of the weak enforcement of the boundary conditions). To ensure the solution of the system, the functions $\mathcal{B}_k(\boldsymbol{q}_h)$ must not be a combination of outward-traveling Riemann invariants. Since an extension of the Riemann invariants to real gas models is complex, a linearized version of the system in Eq (3.25) with respect to the set of discrete primitive variables $\boldsymbol{v}_h = [\rho, p, u, v]^\intercal$, as suggested by Colonna et al. [32]. The linearized $\mathcal{B}_k(\boldsymbol{v}_h^{bc})$ are defined as

$$\begin{aligned} \mathcal{B}_k(\boldsymbol{v}_h^{bc}) &\approx \left(\frac{\partial \mathcal{B}_k}{\partial \boldsymbol{v}_h}\right)_{\boldsymbol{v}_h} (\boldsymbol{v}_h^{bc} - \boldsymbol{v}_h) + \mathcal{B}^k(\boldsymbol{v}_h), \\ \Gamma_k(\boldsymbol{v}_h^{bc}) &\approx \left(\frac{\partial \Gamma_k}{\partial \boldsymbol{v}_h}\right)_{\boldsymbol{v}_h} (\boldsymbol{v}_h^{bc} - \boldsymbol{v}_h) + \Gamma_k(\boldsymbol{v}_h), \end{aligned} \tag{3.26}$$

so that the system to be solved in order to evaluate the boundary state can be written as

$$\begin{cases} \left(\frac{\partial \mathcal{B}_k}{\partial \boldsymbol{v}_h}\right)_{\boldsymbol{v}_h} \delta \boldsymbol{v}_h = \mathcal{B}_k^{bc} - \mathcal{B}_k(\boldsymbol{v}_h) \\ \left(\frac{\partial \Gamma_k}{\partial \boldsymbol{v}_h}\right)_{\boldsymbol{v}_h} \delta \boldsymbol{v}_h = 0 \end{cases}, \tag{3.27}$$

where the unknown is $\delta \boldsymbol{v}_h = \boldsymbol{v}_h^{bc} - \boldsymbol{v}_h$. Equations for the computation of $\delta \boldsymbol{v}_h$ in the inflow and outflow boundary conditions can be found in Appendix E.

3.1.6 Time integration

After solving the integrals in Eq (3.17) using suitable Gauss quadrature rules, the following system of nonlinear Ordinary Differential Equations (ODEs) is obtained as

$$\boldsymbol{M} \frac{d\mathbf{Q}}{dt} + \boldsymbol{R}(\mathbf{Q}) = 0, \tag{3.28}$$

where $\boldsymbol{M}$ is the mass matrix and $\boldsymbol{R}$ the vector of residuals. For steady solution on highly stretched meshes, Eq (3.28) is integrated by means of the implicit backward Euler method as

$$\left(\frac{\boldsymbol{M}}{dt} + \frac{\partial \boldsymbol{R}}{\partial \mathbf{Q}}\right)\left(\mathbf{Q}^{n+1} - \mathbf{Q}^{n}\right) = -\boldsymbol{R}(\mathbf{Q}^{n}), \tag{3.29}$$

where $\partial \boldsymbol{R}/\partial \mathbf{Q}$ is the Jacobian matrix associated to the dG space discretization, n and $n+1$ are the current and next time steps. Equation (3.29) is solved with the FGMRES algorithm at each time step, enhanced by a system preconditioning that relies on the block Jacobi method, where each block is solved with BILU(0). As described by Bassi et al. [52], a pseudo-transient continuation strategy is employed to integrate Equation (3.29). The CFL number is dynamically adjusted according to a factor s_{CFL}, defined as

$$s_{\mathrm{CFL}} = \sqrt{\frac{1}{N}\sum_{i=1}^{m}\left(\frac{\|\boldsymbol{R}_i\|_2}{\max(\|\boldsymbol{R}_i\|_2, \|\boldsymbol{R}_0\|_2)}\right)^2}, \tag{3.30}$$

where $\|\boldsymbol{R}_i\|_2$ is the L^2 norm of the residuals at the i-th time step, and $\|\boldsymbol{R}_0\|_2$ the L^2 norm of the residuals at the first time step. The CFL number is then calculated as

$$\mathrm{CFL} = \min\left(\max\left(\frac{\mathrm{CFL}_{min}}{s_{\mathrm{CFL}}^{\mathrm{CFL}_{exp}}}, \mathrm{CFL}_{min}\right), \mathrm{CFL}_{max}\right), \tag{3.31}$$

where CFL_{min} and CFL_{max} are the minimum and maximum allowable CFL numbers, respectively, while CFL_{exp} is an exponent applied to the factor s_{CFL} to control the sensitivity of the CFL number to changes in the ratio of current and initial residuals.

Chapter 4

Solver validation

The solver validation is carried out on different test cases where analytical or experimental data are available: *i*) the Ringleb flow (see Sec. 4.1.1), *ii*) the laminar flat plate (see Sec. 4.1.2), and *iii*) the turbulent flat plate (see Sec. 4.1.3). This first part serves the purpose of validating the numerical implementation. The remaining sections are instead focused on advanced simulations, where different flow regimes around a NACA 0012 airfoil are investigated. For every simulation, the results are given in terms of field solution and distribution of quantities of interest. A convergence history study is performed, which is done once for each regime, since more studies on the same regime would be redundant.

4.1 Numerical and model validation

4.1.1 Ringleb flow

The Ringleb flow provides an exact solution for the Euler equations with $\gamma = 1.4$ [53], and it is often used to check the formal order of accuracy of a solver. As shown by Masatsuka [54], each point (x, y) of the domain can be computed as a function of the physical variables (V, θ), i.e., the velocity and angle, as

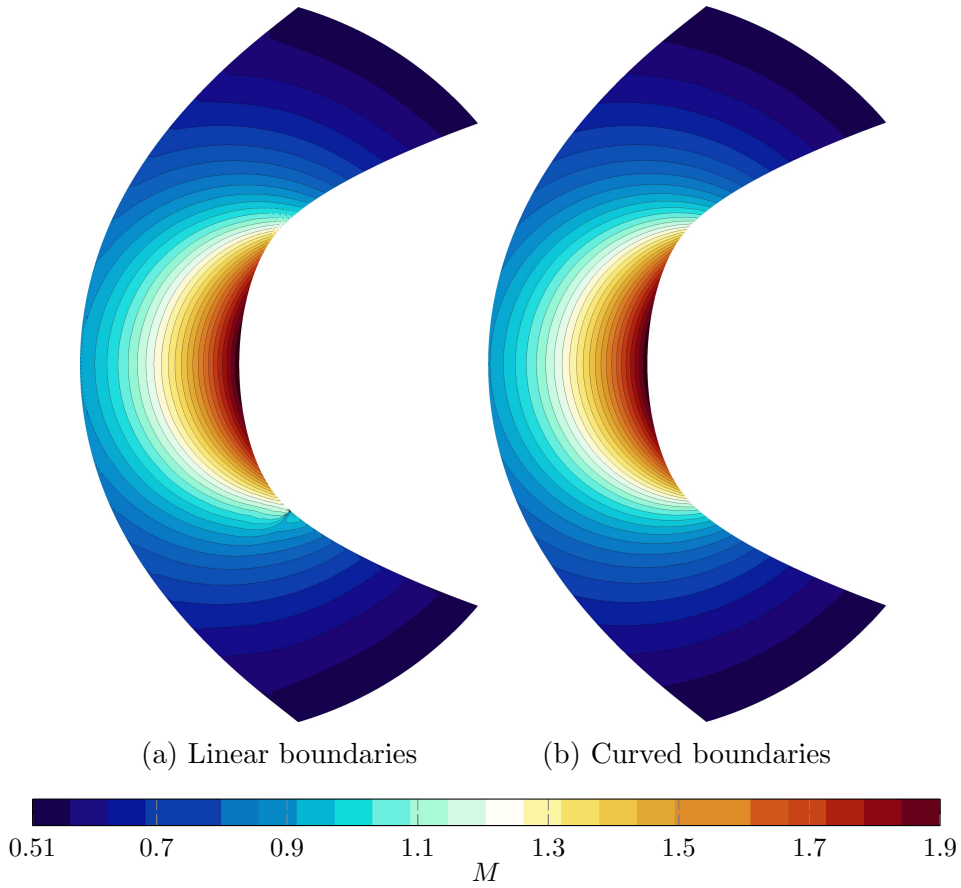
$$x(\psi, V) = \frac{1}{\rho}\left[\frac{1}{2V^2} - \psi^2\right] + \frac{L}{2}, \qquad y(\psi, V) = \pm\frac{\psi}{\rho V}\sqrt{1 - V^2\psi^2}, \tag{4.1}$$

where $\psi = \sin\theta/V$ is a stream function, while ρ and L are

$$\rho = b^5, \qquad b = \sqrt{1 - 0.2V^2}, \qquad L = \frac{1}{b} + \frac{1}{3b^3} + \frac{1}{5b^5} - \frac{1}{2}\ln\left(\frac{1+b}{1-b}\right), \qquad p = (1/\gamma)b^7.$$

The mesh is generated by computing the boundary points with Eqs. (4.1). The inflow velocity is set at $V_{min} = 0.5$. The flow is isoentropic and irrotational, reaching a

Mach number $M = 1.9$ at $y = 0$ (see Fig. 4.1). Figure 4.1 also shows the $\mathbb{P}^1$ Mach number contours obtained using linear boundaries with an imposed slip-wall boundary condition. As expected, spurious phenomena such as artificial shocks arise, preventing the correct representation of the solution, especially in the re-compression area of the domain. This result proves the higher accuracy and stability of using high-order curvilinear boundaries. The exact solution is used for both field initialization and boundary conditions. Figure 4.2 shows the convergence history of the L^2 norm of the energy error with respect to the mesh spacing to verify the convergence order of each solution approximation between $\mathbb{P}^1$ and $\mathbb{P}^5$. The mesh spacing considered for the study is the width of the boundary elements at $y = 0$.



(a) Linear boundaries (b) Curved boundaries

Figure 4.1: Ringleb. Mach number contours on the geometries with linear and curved boundaries; $\mathbb{P}^1$ and $\mathbb{P}^{\not 1}$ solution approximation, respectively

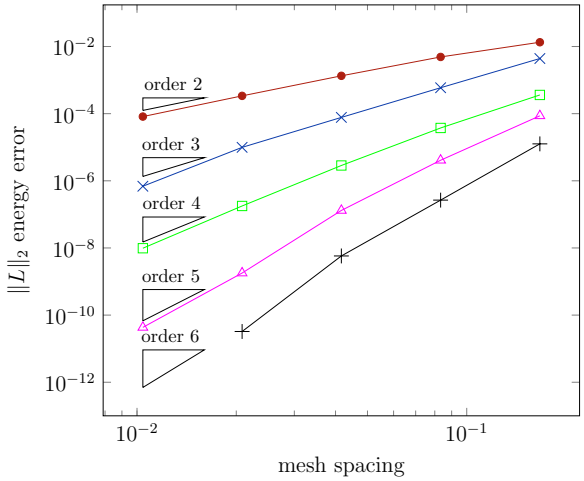


Figure 4.2: Ringleb. Convergence history of the L^2 norm of the energy error with respect to the mesh spacing with different meshes and solution approximation; $\mathbb{P}^1$, $\mathbb{P}^2$, $\mathbb{P}^3$, $\mathbb{P}^4$, $\mathbb{P}^5$

4.1.2 Laminar flat plate

The second validation test case is the laminar flow over a flat plate for freestream Mach number $M = 0.3$ and Reynolds number $Re = 10^6$, based on the freestream velocity u_∞ and the plate length. The calculation is carried out on a linear mesh of 1290 elements, which is designed to be sufficiently refined near the surface of the flat plate to accurately capture details of the boundary layer. An adiabatic no-slip wall boundary condition is imposed along the plate, while a symmetry condition is used for the boundary located before the leading edge of the plate, as illustrated in Fig. 4.5. The results are compared with the Blasius solution, both in terms of the skin friction coefficient C_f distribution along the plate, and the velocity profile u/u_∞, as shown in Fig. 4.3, where $\eta = y\sqrt{\frac{u_\infty}{\nu x}}$ is the nondimensional y coordinate. A perfect agreement can be assessed for both quantities.

4.1.3 Turbulent flat plate

In the second test case, the flow over an adiabatic flat plate with a Mach number $M = 0.2$ and a Reynolds number $Re = 1.1 \times 10^7$ based on the freestream condition and the plate length is computed. The computation is carried out on a linear mesh of 4902 elements, characterized by a different non dimensional height of the first cell adjacent to the wall $y^+ = \frac{u_\infty y}{\nu} = 1$, with growth ratios of 1.2 in the normal direction and 1.15 in the streamwise direction. The streamwise growth rate is changed to 1.00 at approximately 70% of the plate length. The aspect ratio of the first element at the leading edge is set to 40. The solution approximation is $\mathbb{P}^4$.

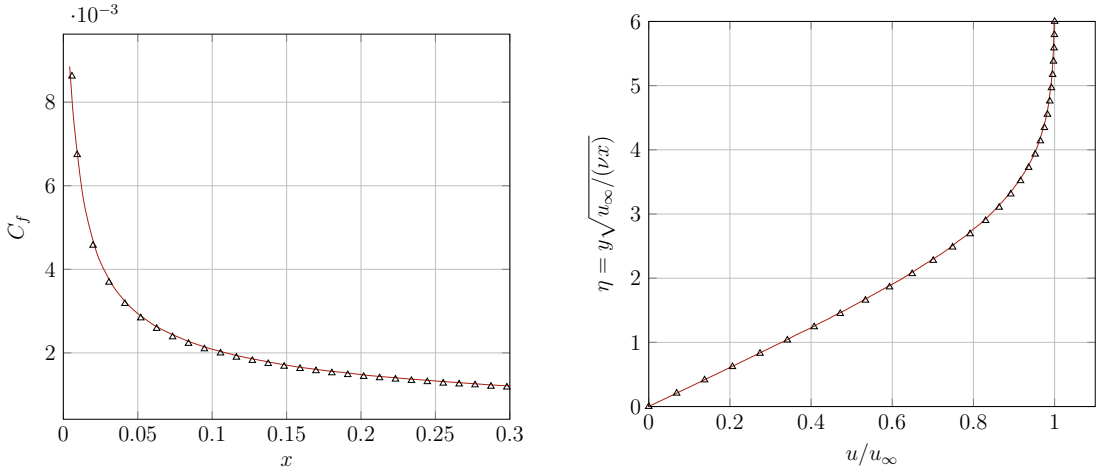


Figure 4.3: Laminar flat plate. Comparison of the C_f distribution along the plate (left) and the velocity profile u/u_∞ (right) with respect to the Blasius solution; — $\mathbb{P}^4$ solution approximation, ▵ Blasius solution

An adiabatic no-slip wall boundary condition is imposed along the plate, while a symmetry condition is used for the boundary located before the leading edge of the plate, as detailed in Fig. 4.5.

The results are compared with the law of the wall and the experimental data by Wieghardt [55], both in terms of the skin friction coefficient C_f distribution along the plate, and the velocity profile $u^+ = u/u_\tau$, as shown in Fig. 4.4. The law of the wall can be written in the viscous sublayer as

$$u^+ = y^+,$$

while in the log-layer as

$$u^+ = \frac{1}{\kappa} \log y^+ + B,$$

where $\kappa = 0.41$ is the Von Karman constant and $B = 5$. A perfect agreement can be appreciated for both quantities. The law of the wall in the viscous sublayer is perfectly followed, while the trend in the log-layer reaches the freestream plateau as described by Wieghardt [55].

Further investigations can be conducted by studying the effect of the degree $\mathbb{P}$ of the solution approximation and the value y^+ on the distribution of the skin friction coefficient and on the velocity profile. Figure 4.6 compares the results obtained on the mesh with $y^+ = 1$ for different approximations of the solution $\mathbb{P}^{0\to4}$ with the experimental data of Wieghardt and Tillmann [55] and the law of the wall. The results of $\mathbb{P}^0$ deviate significantly from the expected values, likely due to the absence of gradients, while the distribution of C_f progressively approaches the reference

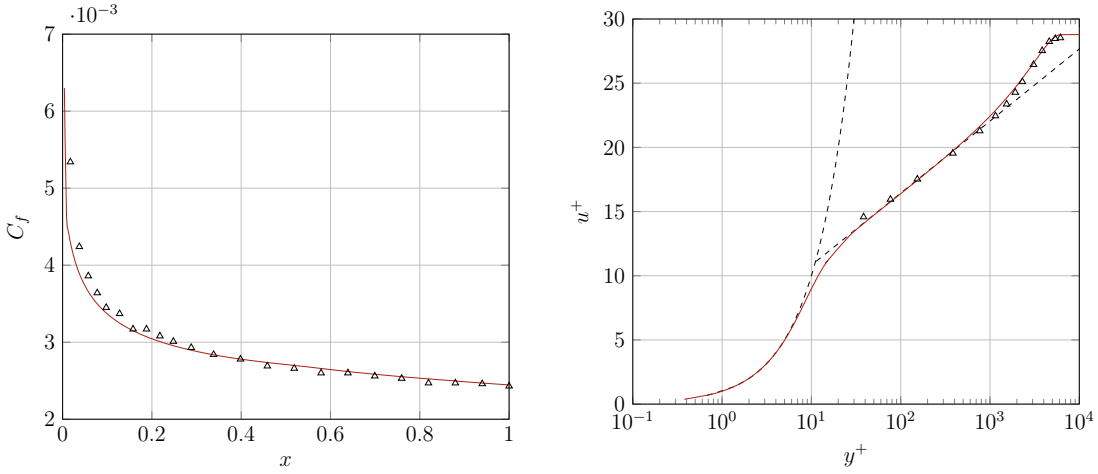


Figure 4.4: Turbulent flat plate. Comparison of the C_f distribution along the plate (left) and the velocity profile u^+ (right) with respect to the experimental data by Wieghardt [55] and the law of the wall; —— $\mathbb{P}^4$ solution approximation, - - - law of the wall, ▵ Wieghardt exp. [55]

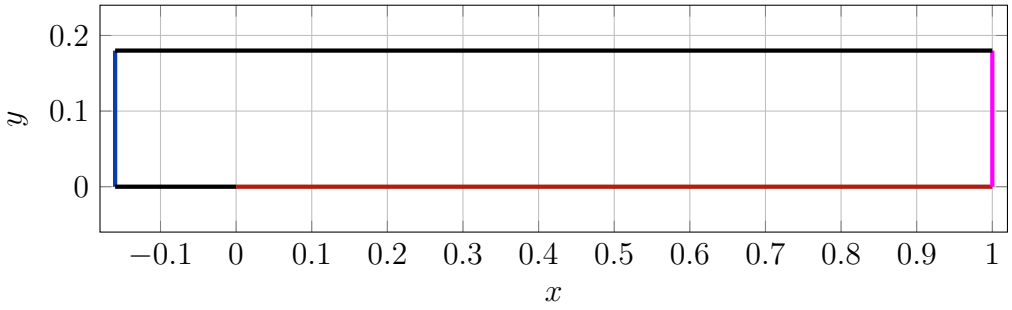


Figure 4.5: Laminar and turbulent flat plate. Details of the boundary conditions used for the simulations; —— inflow, —— symmetry, —— adiabatic wall, —— outflow

data from $\mathbb{P}^{0\to4}$. The velocity profiles are almost superimposed, with minor variations observed in the plateau region, suggesting a refined accuracy with increasing polynomial degrees.

Four different computations were carried out on four distinct linear grids, characterized by a different nondimensional height, $y^+ = \frac{u_\infty y}{\nu}$, of the first cell adjacent to the wall ($y^+ = 1$, $y^+ = 10$, $y^+ = 20$ and $y^+ = 40$). All meshes share the same growth ratio along the normal (1.2) and streamwise (1.15) direction. The streamwise growth ratio is changed to 1.00 at approximately 70% of the plate length. The aspect ratio of the first element at the leading edge is set to 40. The number of elements for each mesh is listed in Tab. 4.1. The maximum solution approximation is $\mathbb{P}^4$. Figure 4.7 shows the skin friction, c_f, distribution (left), and the nondimen-

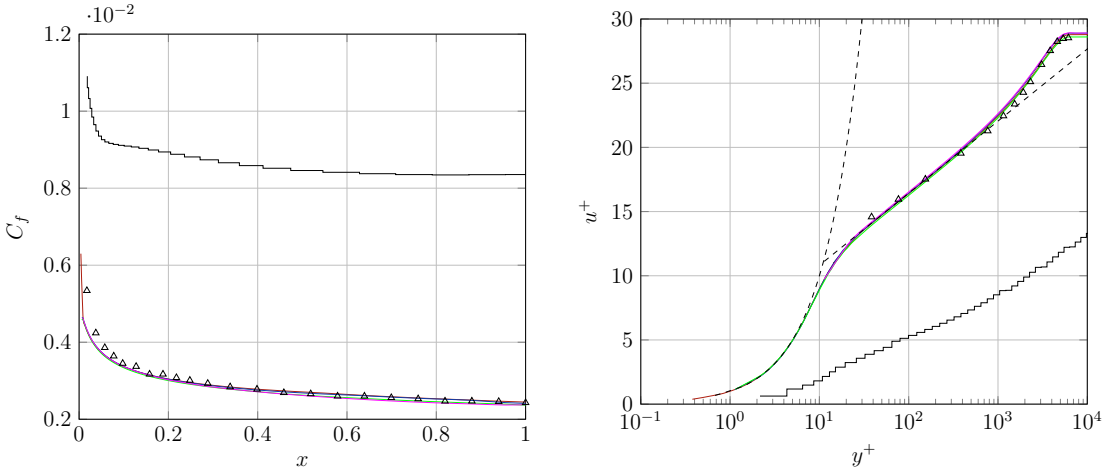


Figure 4.6: Turbulent flat plate. c_f distribution along the plate (left) and streamwise velocity profile u^+ (right) comparison for $\mathbb{P}^{0\to4}$ solutions and with theoretical and experimental data; — $\mathbb{P}^4$, — $\mathbb{P}^3$, — $\mathbb{P}^2$, — $\mathbb{P}^1$, — $\mathbb{P}^0$, - - - law of the wall, ▵ Wieghardt exp. [55]

Table 4.1: Number of elements for each mesh at different y^+ values

y^+	Elements
1	4902
10	3510
20	2296
40	1702

sional streamwise, u^+, velocity (right), obtained on meshes with different y^+ for a $\mathbb{P}^4$ solution approximation. The c_f curves are almost superimposed up to $y^+ = 40$, while some discrepancies are evident for the velocity profiles. In particular, as shown in Fig. 4.8, in the viscous sublayer (left), the velocity profile for the $y^+ = 40$ mesh is under estimated, while the log- and defect-layers (right) are correctly captured up to $y^+ = 40$.

4.2 Inviscid flow test cases

This section is dedicated to the validation of the solver on inviscid flows, i.e. governed by the Euler equations, around a NACA0012 airfoil, a benchmark geometry that will also be employed for the laminar and turbulent validation cases presented in Secs 4.3 and 4.4. For inviscid simulations, a C-grid mesh composed of 1392 $\mathbb{P}^3$ Bézier patches is used, as shown in Fig. 4.9.

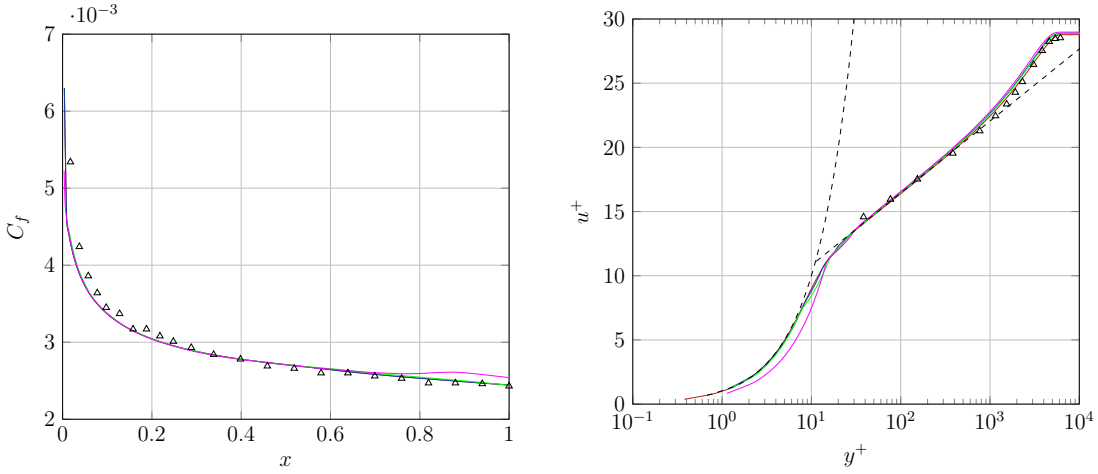


Figure 4.7: Turbulent flat plate. c_f distribution along the plate (left) and streamwise velocity profile u^+ (right) comparison for different values of y^+ and with theoretical and experimental data, $\mathbb{P}^4$ solution approximation; — $y^+ = 1$, — $y^+ = 10$, — $y^+ = 20$, — $y^+ = 40$, - - - law of the wall, ▵ Wieghardt exp. [55]

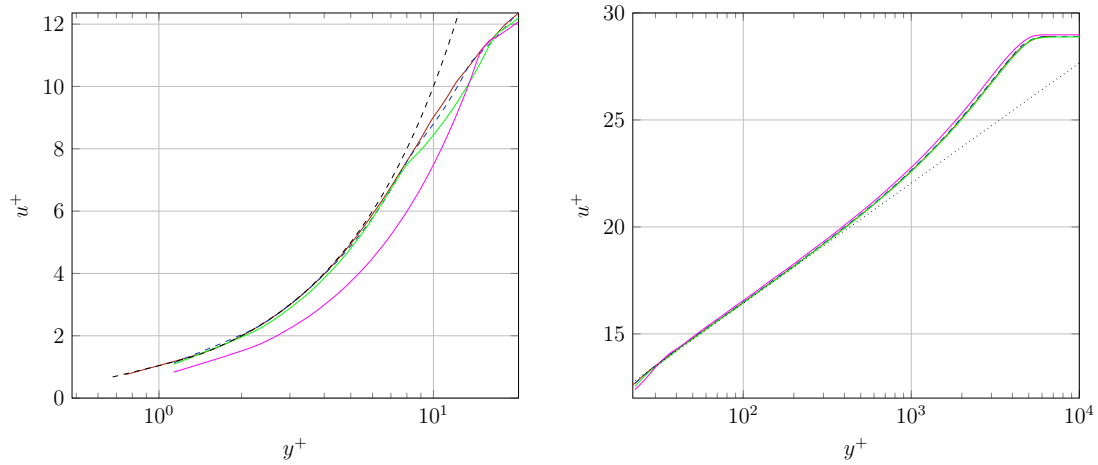


Figure 4.8: Turbulent flat plate. Close-up of the streamwise velocity profile in the viscous sublayer (left), and in the defect layer (right) on meshes for different values of y^+, $\mathbb{P}^4$ solution approximation; — $y^+ = 1$, - - - $y^+ = 10$, — $y^+ = 20$, — $y^+ = 40$, - - - $u^+ = y^+$, ····· law of the wall

4.2.1 Subsonic flow over a NACA0012 airfoil

The first case considered here is the subsonic flow characterized by an angle of attack $\alpha = 3.00°$ and a farfield Mach number $M_\infty = 0.50$. Figure 4.10 shows the $\mathbb{P}^4$ Mach number on the profile. The distribution of the pressure coefficient is compared with

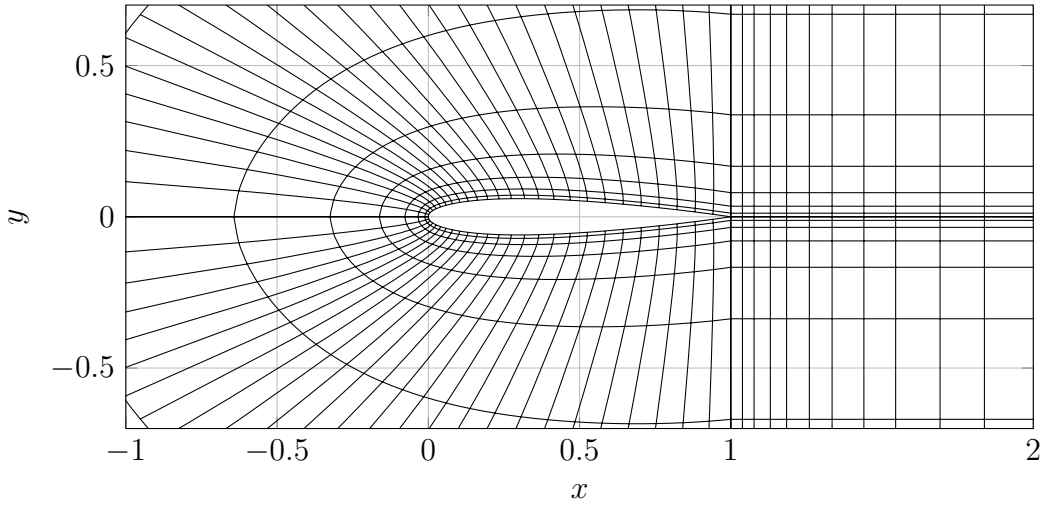


Figure 4.9: NACA0012. C-grid mesh used for inviscid simulations; 1392 $\mathbb{P}^3$ Bézier patches

the results obtained by Jameson [56], as illustrated in Fig. 4.11, showing a correct distribution along the airfoil surface. For this test case, a study of the convergence of the residuals of degrees $\mathbb{P}^0 \rightarrow \mathbb{P}^4$ is performed, resulting in the expected quadratic behavior, as shown in Fig. 4.12.

4.2.2 Transonic flow over a NACA0012 airfoil

The second test case detailed here is the transonic inviscid flow over a NACA0012. This case increases the complexity of the calculation since it deals with a shock formation. The flow in the freestream is defined by an angle of attack $\alpha = 0.00°$ and a farfield Mach number $M_\infty = 0.80$. The $\mathbb{P}^4$ Mach number contours of the solution are shown in Fig. 4.13. The distribution of the pressure coefficient, see Fig. 4.14, highlights the presence of a shock on both the pressure and suction side at approximately 50% of the chord length. The results are compared with the data presented by Jameson [56]. The solver also demonstrates the ability to correctly simulate the fluid behavior for transonic inviscid regimes. The mesh used for this simulation is the same as adopted for the subsonic flow, which is too coarse to correctly represent the shock discontinuity. For this reason, an isotropic h-refinement procedure was used to locally refine the mesh along the discontinuity. A more detailed explanation of this procedure can be found in Chapter 5.

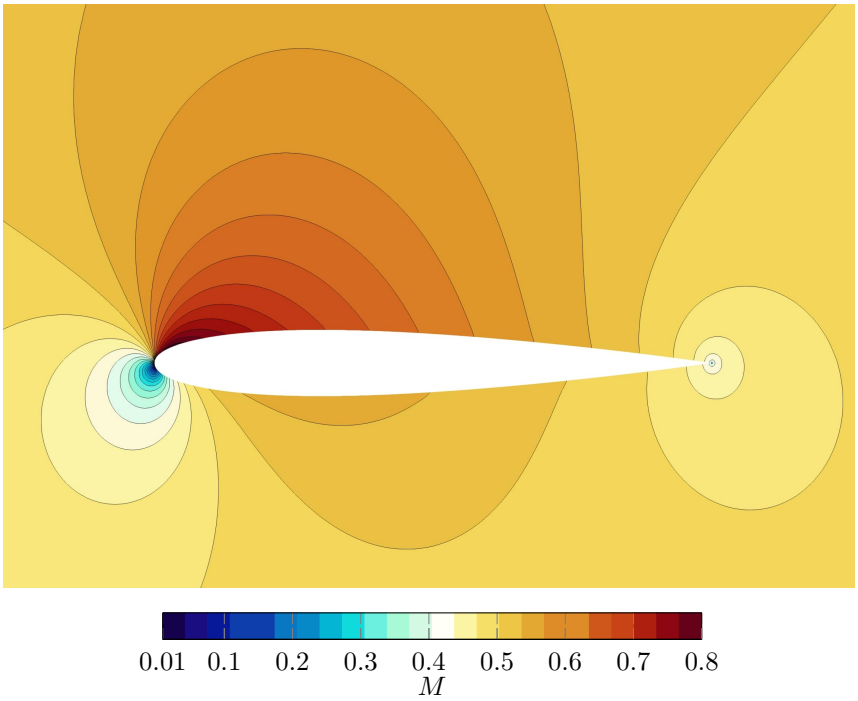


Figure 4.10: NACA0012. Mach number contours, $\mathbb{P}^4$ solution approximation ($M_\infty = 0.50$, $\alpha = 3°$)

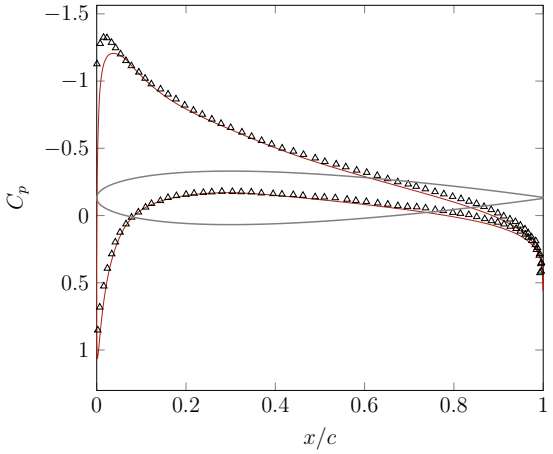


Figure 4.11: NACA0012. Comparison of the pressure coefficient distribution C_p on the airfoil surface with respect to the results of Jameson [56], $\mathbb{P}^4$ solution approximation; — NURBS-DG, ▵ Jameson [56] ($M_\infty = 0.50$, $\alpha = 3°$)

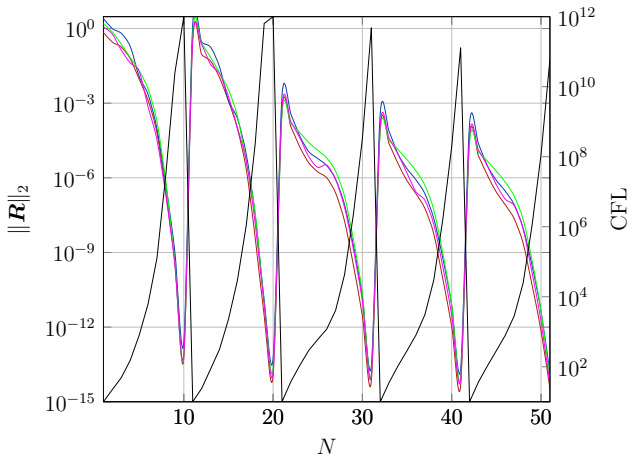


Figure 4.12: NACA0012. Residuals convergence history; — ρ , — ρE, — ρu, — ρv, — CFL number ($M_\infty = 0.50$, $\alpha = 3°$)

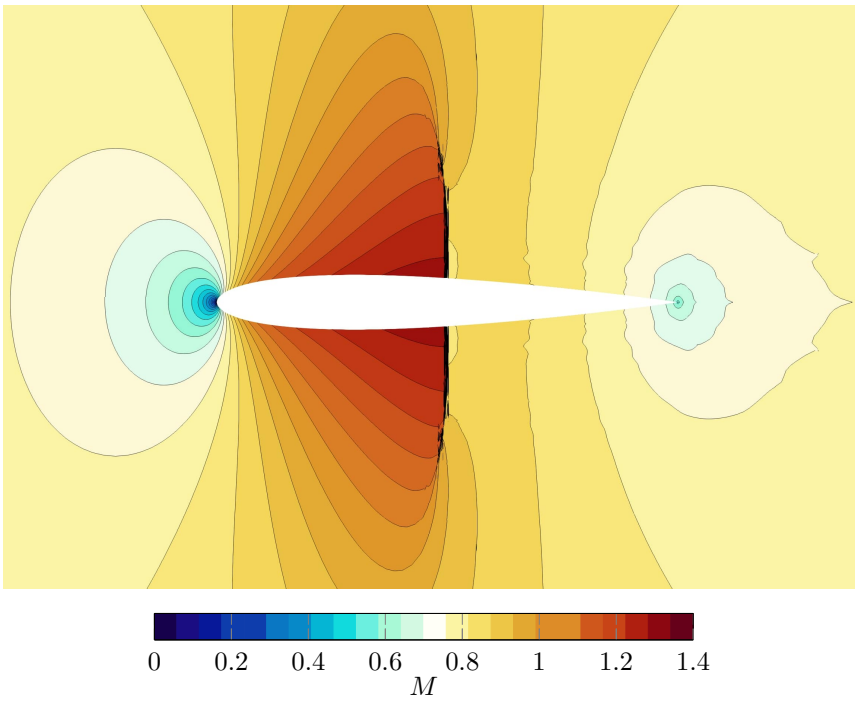


Figure 4.13: NACA0012. Mach number contours, $\mathbb{P}^4$ solution approximation ($M_\infty = 0.80$, $\alpha = 0°$)

4.3 Laminar flow test cases

This section is dedicated to the validation of the solver capability to correctly simulate laminar flows around airfoils for different Reynolds numbers. A mesh composed

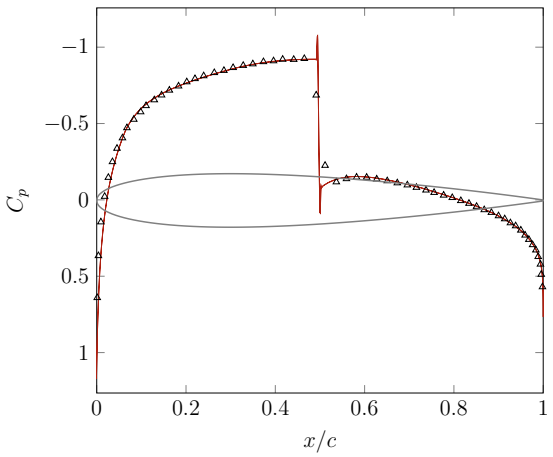


Figure 4.14: NACA0012. Comparison of the pressure coefficient distribution C_p on the airfoil surface with respect to the results of Jameson [56], $\mathbb{P}^4$ solution approximation; — NURBS-DG, ▵ Jameson [56] ($M_\infty = 0.80$, $\alpha = 0°$)

of 1020 $\mathbb{P}^3$ Bézier patches is used for the simulations at Reynodolds number $Re = 73$ and $Re = 106$ (Fig. 4.15), while a mesh composed of 2280 $\mathbb{P}^3$ Bézier patches is used for the simulation at $Re = 5000$ (Fig. 4.20). The Reynolds number used is computed on the basis of the freestream velocity and with respect to a unitary chord length. An adiabatic no-slip boundary condition is imposed on the airfoil surface.

4.3.1 Laminar flow over a NACA0012 airfoil at $Re = 73$

In the first test case the laminar transonic flow at an angle of attack $\alpha = 10°$, freestream Mach number of $M_\infty = 0.80$ and a Reynolds number $Re = 73$ is computed. The $\mathbb{P}^4$ Mach isolines are shown in Fig. 4.16. Figure 4.17 shows the distributions of the pressure coefficient and skin friction coefficient along the airfoil compared to the results obtained by Bassi and Rebay [43]. The distributions closely follow the expected trends.

4.3.2 Supersonic laminar flow over a NACA0012 airfoil at $Re = 106$

In the second test case the laminar supersonic flow at an angle of attack $\alpha = 10°$, a freestream Mach number of $M_\infty = 2.00$, and a Reynolds number $Re = 106$ is computed. The $\mathbb{P}^4$ Mach isolines are shown in Fig. 4.18, where a detached bow shock is clearly visible. Figure 4.19 shows the distributions of the pressure coefficient and the skin friction coefficient along the airfoil compared to the results obtained by

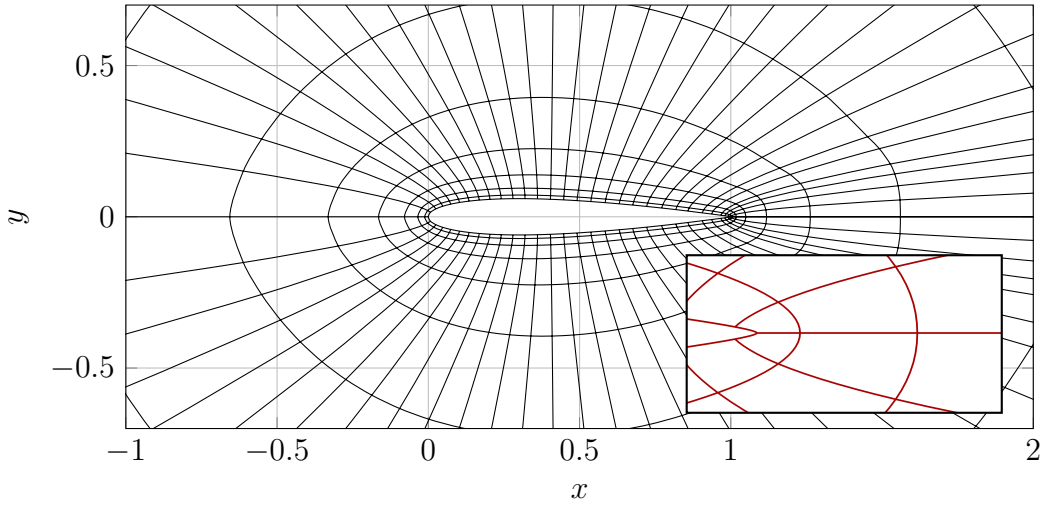


Figure 4.15: NACA0012. O-grid mesh used for the laminar simulations at $Re = 73$ and $Re = 106$ with a detail of the rounded trailing edge region; 1020 $\mathbb{P}^3$ Bézier patches

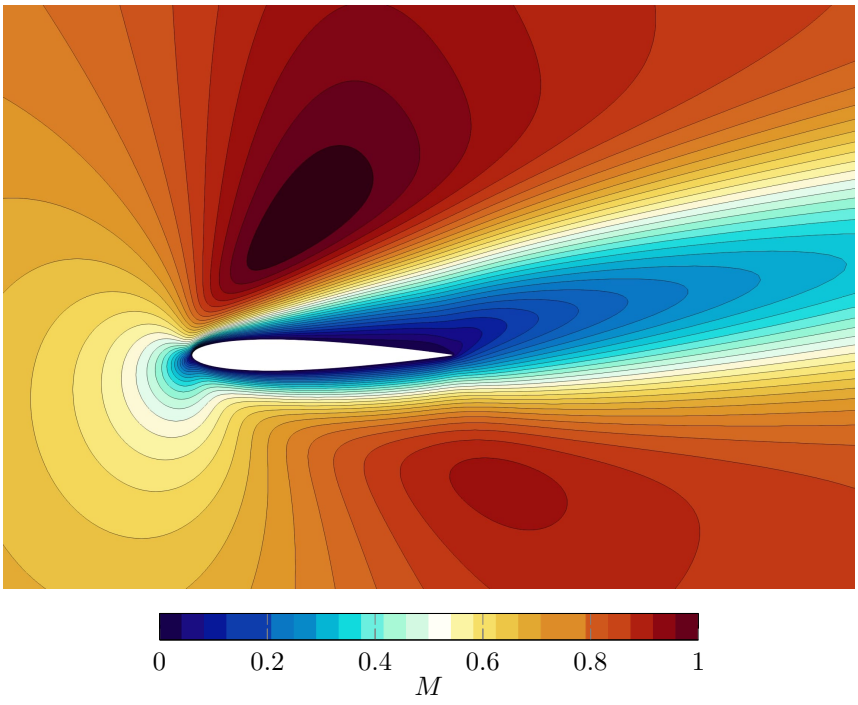


Figure 4.16: NACA0012. Mach number contours, $\mathbb{P}^4$ solution approximation ($Re = 73$, $M_\infty = 0.80$, $\alpha = 10°$)

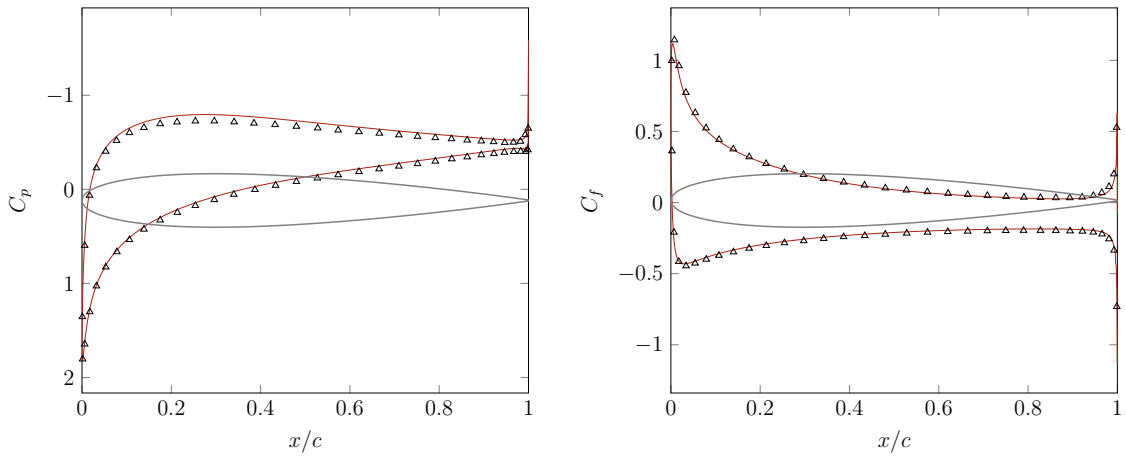


Figure 4.17: NACA0012. Comparison of the pressure C_p (left) and skin friction C_f (right) coefficient distributions on the airfoil with the results from Bassi and Rebay [43], $\mathbb{P}^4$ solution approximation; — NURBS-DG, ▵ Bassi and Rebay [43] ($Re = 73$, $M_\infty = 0.80$, $\alpha = 10°$))

Bassi and Rebay [43]. This test case successfully validates the capability to predict supersonic laminar flows in the presence of a shock.

4.3.3 Laminar flow over a NACA0012 airfoil at $Re = 5000$

In the third test case the laminar flow for angle of attack $\alpha = 0°$, a freestream Mach number $M_\infty = 0.5$ and a Reynolds number $Re = 5000$ is computed. The Reynolds number is close to the upper limit for a steady laminar flow. This test case is characterized by the detachment of the flow at the trailing edge, resulting in the creation of a small recirculation bubble that extends into the near-wake region of the airfoil, as can be seen in Fig. 4.21, where the Mach number $\mathbb{P}^4$ isolines are shown. As in previous test cases, the distribution of the pressure and skin friction coefficients is shown and compared to the available literature results [43] in Fig. 4.22. A study of the history of convergence for the residuals of degrees $\mathbb{P}^0 \rightarrow \mathbb{P}^4$ is shown in Fig. 4.23.

4.4 Turbulent flow test cases

Finally, the turbulent flow around a NACA0012 at $Re = 3 \times 10^6$ is investigated. The mesh used consists of 3950 $\mathbb{P}^3$ Bézier patches characterized by an average $y^+ \approx 1$. The patches are distributed in the same way as the mesh in Fig. 4.20, with a finer boundary layer. A $\mathbb{P}^3$ solution approximation is adopted.

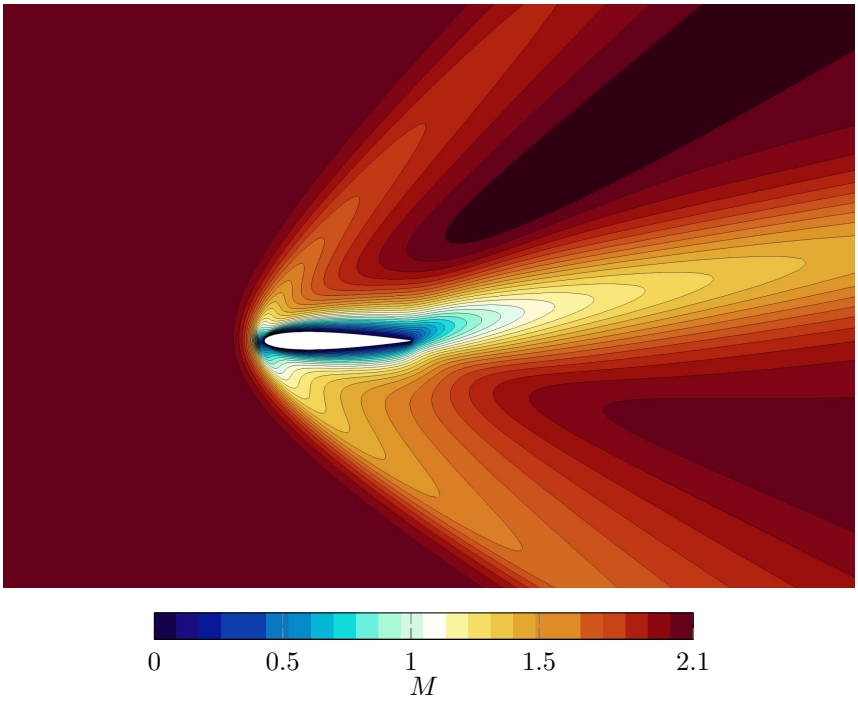


Figure 4.18: NACA0012. Mach number contours, $\mathbb{P}^4$ solution approximation ($Re = 106$, $M_\infty = 2.00$, $\alpha = 10°$)

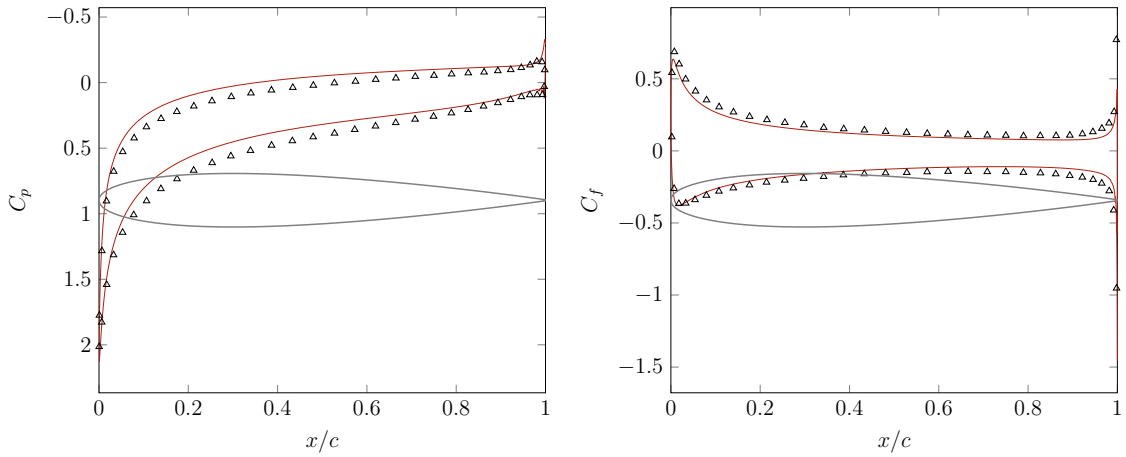


Figure 4.19: NACA0012. Comparison of the pressure C_p (left) and skin friction C_f (right) coefficient distributions on the airfoil with the results from Bassi and Rebay [43], $\mathbb{P}^4$ solution approximation; —— NURBS-DG, ▵ Bassi and Rebay [43] ($Re = 106$, $M_\infty = 2.00$, $\alpha = 10°$))

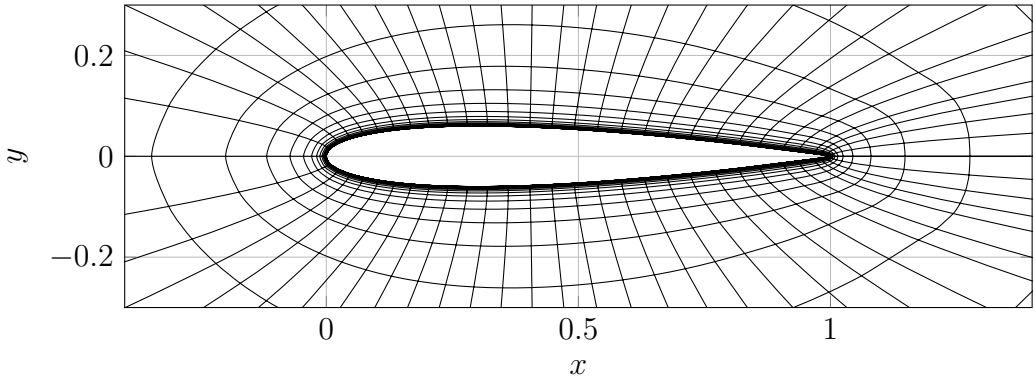


Figure 4.20: NACA0012. O-grid mesh used for the laminar simulation at $Re = 5000$; 2280 $\mathbb{P}^3$ Bézier patches

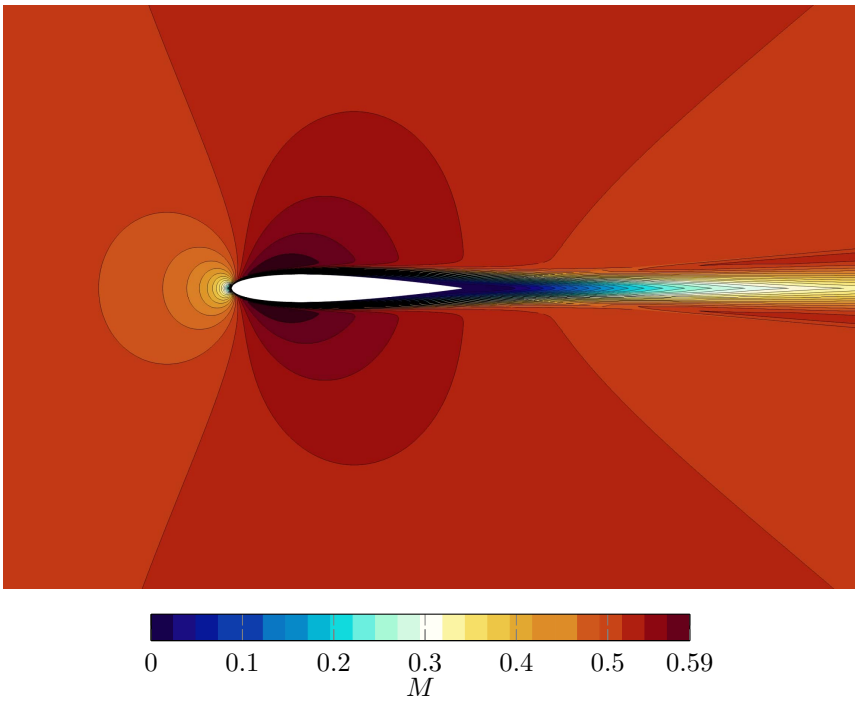


Figure 4.21: NACA0012. Mach number contours, $\mathbb{P}^4$ solution approximation ($Re = 5000$, $M_\infty = 0.50$, $\alpha = 0°$)

4.4.1 Subsonic turbulent flow over a NACA0012 airfoil $Re = 3 \times 10^6$

The turbulent flow over a NACA0012 airfoil is considered, with a freestream Mach number $M = 0.15$, a Reynolds number 3.0×10^6 and an angle of attack $\alpha = 10°$. The

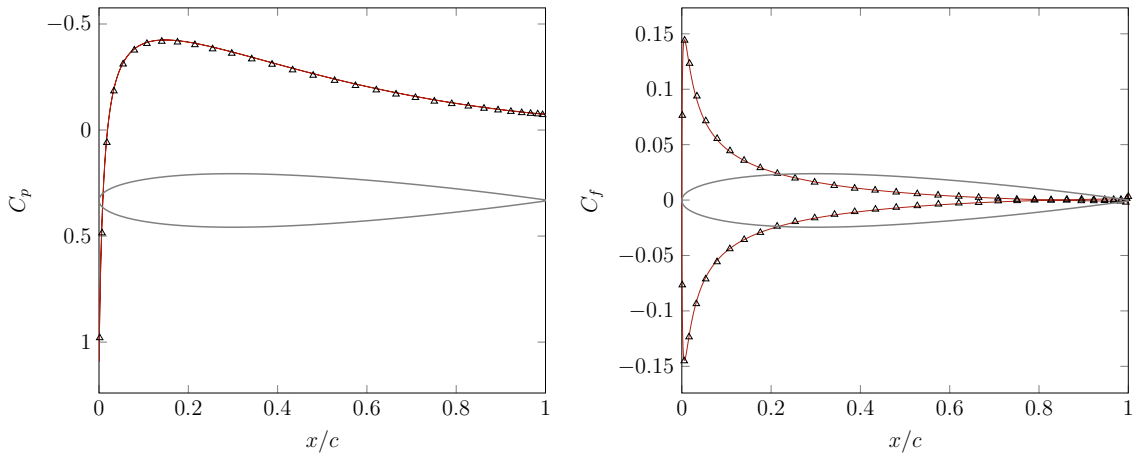


Figure 4.22: NACA0012. Comparison of the pressure C_p (left) and skin friction C_f (right) coefficient distributions on the airfoil with the results from Bassi and Rebay [43], $\mathbb{P}^4$ solution approximation; — NURBS-DG, ▵ Bassi and Rebay [43] ($Re = 5000$, $M_\infty = 0.50$, $\alpha = 0°$))

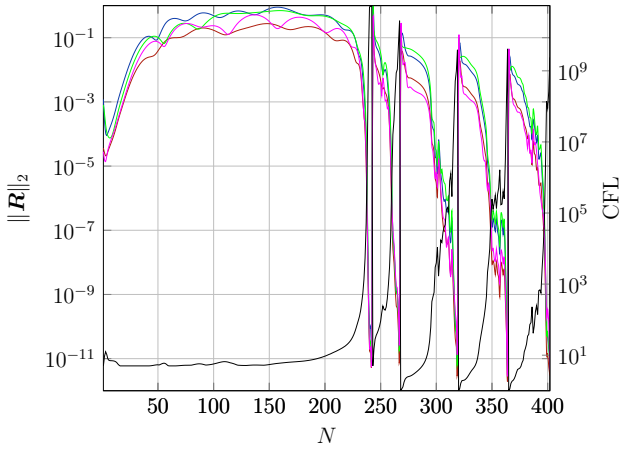


Figure 4.23: NACA0012. Residuals convergence history; — ρ , — ρE, — ρu, — ρv, — CFL number ($Re = 5000$, $M_\infty = 0.50$, $\alpha = 0°$)

flow is characterized by a fully turbulent boundary layer over the majority of the profile. Figure 4.24 shows the Mach number and $\tilde{\nu}$ contours. The predicted pressure coefficient distribution (see Fig. 4.25) agrees well with the experimental data by Ladson et al. [57] and Gregory et al. [58]. The skin friction coefficient cannot be validated since no data is available for this test case. Figure 4.26 shows the history of convergence of the residual norms for solution approximations $\mathbb{P}^0 \rightarrow \mathbb{P}^3$. The residuals show an initial oscillating behavior at $\mathbb{P}^1$, but recover in ≈ 200 iterations. The quadratic convergence is achieved for all degrees.

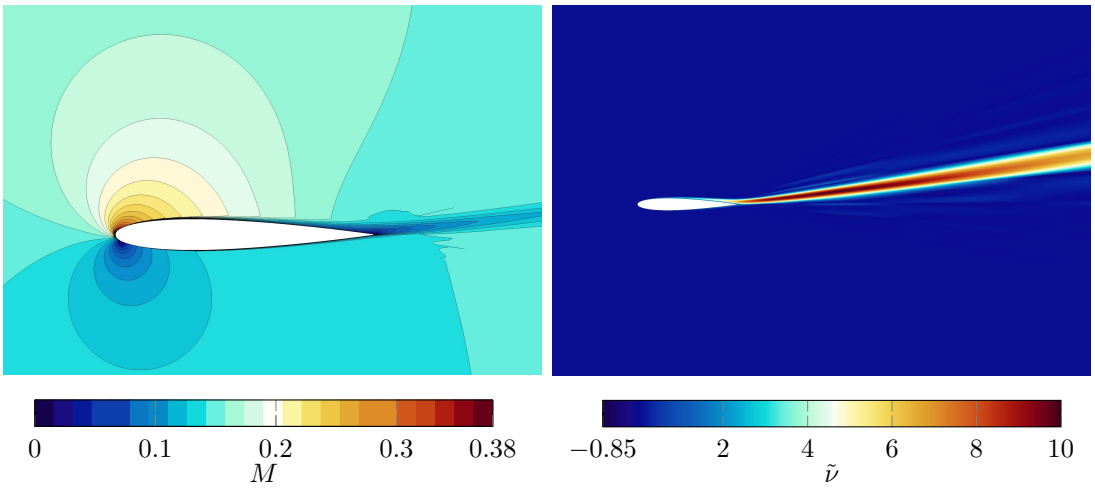


Figure 4.24: NACA0012. Mach number (left) and $\tilde{\nu}$ (right) contours, $\mathbb{P}^3$ solution approximation ($Re = 3 \times 10^6$, $M_\infty = 0.15, \alpha = 10°$)

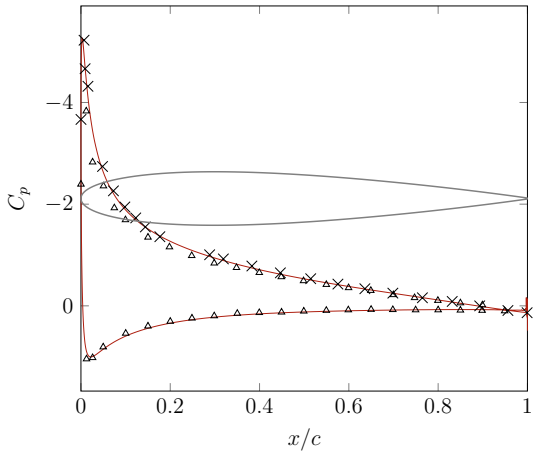


Figure 4.25: NACA0012. Comparison of the pressure C_p (left) coefficient with the experimental results, $\mathbb{P}^3$ solution approximation; — NURBS-DG, ▵ Ladson et al. exp [57], × Gregory et al. exp [58] ($Re = 3 \times 10^6$, $M_\infty = 0.15$, $\alpha = 10°$)

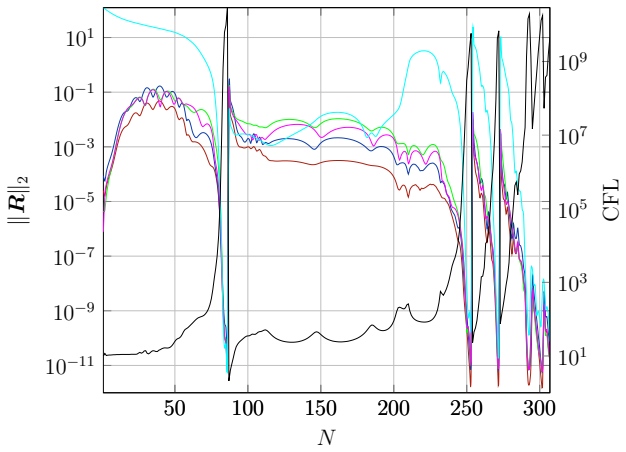


Figure 4.26: NACA0012. Residuals convergence history; — ρ , — ρE, — ρu, — ρv, — $\tilde{\nu}$, — CFL number ($Re = 3 \times 10^6$, $M_\infty = 0.15$, $\alpha = 10°$)

Chapter 5

Mesh Adaptation

5.1 h-adaptation

Because finer elements are clustered only where higher resolution is required for the solution, a NURBS-based isogeometric DG framework can greatly benefit from a h-adaptation strategy i) to increase computational efficiency and accuracy, and ii) to solve flow problems with complex features that are unknown in advance, such as wakes, shocks, or separations. A NURBS framework can simplify the creation of new elements with the standard knot insertion algorithm while maintaining the correctness of the geometry, as seen in Sec. 2.3. In a standard CFD framework, the implementation of a h-adaptation technique may not be simple, and the geometrical discretization error may increase with each adaptation step if there is no direct coupling with a CAD system. Notice that a conventional method with h-adaptation may introduce significant geometrical discretization errors early in the process if the initial mesh is coarse.. The error indicators utilized in a h-adaptation technique are essential for identifying elements that require refinement. Various criteria can be found in the literature and are here compared and evaluated in the adaptation of the mesh used for simulating incompressible laminar flow around a circular cylinder. Furthermore, as the mesh is composed only of quadrilateral Bézier patches, an anisotropic adaptation procedure is employed when the flow exhibits preferred directions. This approach aims to minimize the number of new elements generated during each adaptation iteration, particularly for various flow characteristics, including boundary layers, wakes, and shocks. Figure 5.1 illustrates an example of isotropic and anisotropic h-refinement achieved through knot insertion.

5.1.1 Refinement structure

The mesh refinement process is managed through a quad-tree structure [59], as done by Rose et al. [60], where each element of the initial mesh is marked as a

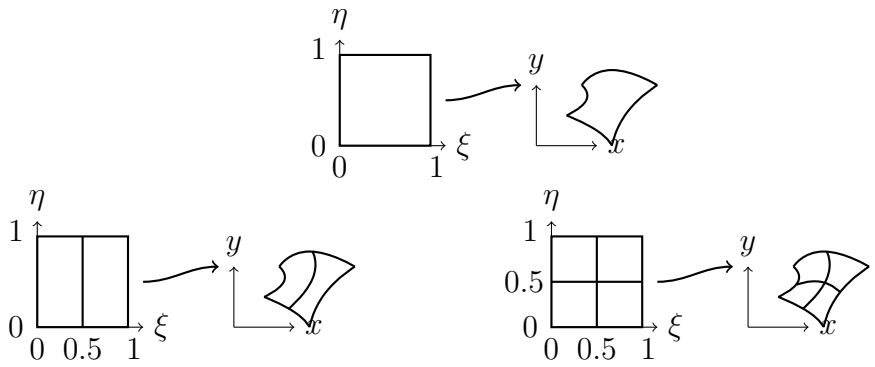


Figure 5.1: Isotropic and anisotropic refinement for a single Bézier patch

root at level 0. In an isotropic adaptation procedure, each refined element at level l, referred to as the "father", is divided into four smaller elements at level $l+1$, known as the "sons". Sons inherit properties from their fathers and communicate information with their siblings, which are neighboring elements that share the same parent. The nodes located at the lowest level of the hierarchy are referred to as "leaves", which indicate the smallest and most refined elements of the mesh. A restriction on the level difference of adjacent elements is added to prevent irregularity of the mesh. A maximum number of adaptation cycles, denoted $n^{adp}_{cyc,max}$, is selected in advance, along with the maximum level of refinement, defined as lvl^{adp}_{max}. The user defines the polynomial solution approximation utilized for adaptation, denoted as $\mathbb{P}^{adp}$. Each adaptation cycle is executed after reaching solution convergence to avoid the adaptation of spurious or transient phenomena. The integration of flux at the interface between elements at different levels in the quad-tree is carried out using the integration points of the smallest element, see Fig. 5.2, as proposed by Duvigneau [51]. During refinement and coarsening, the solution needs to be exactly conserved, when possible. This is achieved by using the same knot insertion applied to the geometry, allowing for exact conservation of the solution. More problems arise when coarsening occurs, as loss of information becomes inevitable. First, a L^2-minimization problem is employed, specifically:

$$\min_{\tilde{\boldsymbol{Q}}} \int_{\Omega_f} \left(\sum_{i=1}^{n} \tilde{Q}_i R_i - \boldsymbol{q}_{s,h} \right)^2 d\Omega. \tag{5.1}$$

The subscripts f and s refer to the father element and the son elements, respectively. $\tilde{\boldsymbol{Q}}$ are the solution coefficients in the father element in a least-squares sense, n is the number of the element basis function minus one, R_i is the $i-th$ basis function as defined in Eq. (2.3), while $\boldsymbol{q}_{s,h}$ is the solution defined on the child elements. This

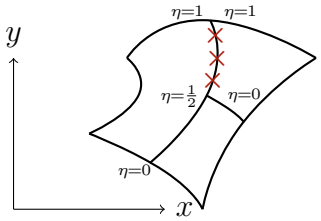


Figure 5.2: Example of quadrature points positioning for flux integration at the interface between patches at different refinement level

involves solving the linear system

$$\boldsymbol{M}\tilde{\boldsymbol{Q}} = \boldsymbol{b}, \tag{5.2}$$

where $\boldsymbol{b} = \sum_{s=1}^{n_{\text{sons}}} \int_{\Omega_s} R_i \boldsymbol{Q}_s \, d\Omega$ and $\boldsymbol{M}$ is the mass matrix. Then, as suggested by Duvigneau [51], the solution conservation is enforced in the following way:

$$\boldsymbol{Q}_f = \tilde{\boldsymbol{Q}} + \frac{1}{A_f}\left(\sum_{s=1}^{n_{\text{sons}}} \int_{\Omega_s} \boldsymbol{Q}_s \, d\Omega - \int_{\Omega_f} \tilde{\boldsymbol{Q}} \, d\Omega\right). \tag{5.3}$$

It is worth recalling that these integrals are solved in the reference space of the element by multiplying the integral arguments by the determinant of the Jacobian $|\boldsymbol{J}_\Omega|$. Unlike refinement, coarsening leads to a loss of information during the least-squares fitting procedure. This approach guarantees only the conservation of the average solution.

5.1.2 Marking procedure for refinement

The elements within the computational domain are ranked, based on some error indicator, while a user-defined percentage threshold, ne^{adp}, is applied to selectively identify how many elements are refined. If elements already refined appear at the bottom of the ranking below a certain threshold ne^{crs}, they are flagged for coarsening. An element can be refined following an isotropic or anisotropic strategy, depending on the solution gradients that can reveal a preferred flow direction. For mesh regularity reasons, each element cannot share an interface with more than two other elements.

The error and anisotropy indicators are:

i) Small-Scale Energy indicator (SSED) [61], defined as

$$\eta_j^{SSED} = \|(\rho \mathbf{u}_h)_p - (\rho \mathbf{u}_h)_{p-1}\|_{\Omega_j}, \tag{5.4}$$

where $(\rho\mathbf{u})_{h,p-1}$ is the projection of the discrete momentum density on the reduced-order functional space, and $\mathbf{u}$ is the vector velocity. The above criterion can be considered as a discretization error for $\mathbf{u}_{h,p}$ or a feature-based refinement indicator that measures the "kinetic energy" associated with the highest-order modes. This indicator shows poor results if the mesh has a large variation in element size. To prevent this behavior, the indicator is normalized with the element size as

$$\eta_j^{SSED} = \left(\frac{\|(\rho\mathbf{u}_h)_p - (\rho\mathbf{u}_h)_{p-1}\|_{\Omega_j}}{|\Omega_j|} \right)^{\frac{1}{2}}, \tag{5.5}$$

where $|\Omega_j|$ is the element size.

ii) Spectral Decay indicator (SD) [62], normalized by the element total "energy" and defined as

$$\eta_j^{SD} = \frac{\|(\rho\mathbf{u}_h)_p - (\rho\mathbf{u}_h)_{p-1}\|_{\Omega_j}}{\|(\rho\mathbf{u}_h)_p\|_{\Omega_j}}. \tag{5.6}$$

iii) Local Non-conformity indicator (LNC), evaluates the maximum jump in the numerical solution across element faces. This indicator is based on the assumption that the exact solution should be continuous across the elements, and the presence of a discontinuity can be used as a measure of the error [63]. Naddei et al. [64] define this indicator as

$$\eta_j^{LNC} = \max_{\partial\Omega_j} \max_q \frac{\|(\rho\mathbf{u}_h)^+(x_q) - (\rho\mathbf{u}_h)^-(x_q)\|}{\|(\rho\mathbf{u}_h)^+(x_q) + (\rho\mathbf{u}_h)^-(x_q)\|}, \tag{5.7}$$

where $\rho\mathbf{u}_h(x_q)$ is the discrete momentum density at the x_q quadrature point.

iv) Global Non-conformity indicator (GNC), evaluates the L^2 norm of all the jumps in the numerical solution across element faces. This indicator is inspired by the work of Duvigneau [51] and is defined as

$$\eta_j^{GNC} = \sum_{\partial\Omega_j} \sum_q \frac{\|(\rho\mathbf{u}_h)^+(x_q) - (\rho\mathbf{u}_h)^-(x_q)\|}{\|(\rho\mathbf{u}_h)^+(x_q) + (\rho\mathbf{u}_h)^-(x_q)\|}. \tag{5.8}$$

After an element is marked for refinement, an anisotropy indicator is evaluated to choose between an isotropic refinement (the element is split into four subelements) or an anisotropic refinement (the element is split into two subelements). The proposed anisotropy indicator is based on the solution gradient, which is mapped onto the parametric space through the inverse of the Jacobian matrix of the NURBS mapping. Taking as reference Fig. 5.3, the gradient $\nabla\boldsymbol{u}_q$ is known at the Gauss point $\boldsymbol{P}_q =$

$\boldsymbol{S}_\Omega(\xi_q, \eta_q)$, where $\boldsymbol{S}_\Omega$ is the Bézier patch representing a generic element Ω. The inverse of the local Jacobian matrix, $\mathbf{J}_{\Omega_q}^{-1}$, is used to project $\nabla \boldsymbol{u}_q$ components onto the space $\hat{x} - \hat{y}$, which is the mapping of the parametric coordinate system $\xi - \eta$ in the physical space centered in $\boldsymbol{P}_q$. Then, the projections of the gradient can be compared in the parametric space $\xi - \eta$. In this work, both the average of the gradient and the inverse of the Jacobian is used, thus obtaining an averaged quantity for the whole element. As the solution gradient is known at each Gauss point, a weighted averaged solution gradient is defined as

$$\overline{\nabla \mathbf{u}}_\Omega = \frac{\sum\limits_q \left(\nabla \mathbf{u}_q \cdot w_q \left| \mathbf{J}_{\Omega_q} \right| \right)}{\sum\limits_q w_q \left| \mathbf{J}_{\Omega_q} \right|}, \tag{5.9}$$

where the subscript q means the evaluation over the respective Gauss point, and w_q is the Gauss point weight. The weighted average Jacobian is defined as

$$\overline{\mathbf{J}_\Omega}^{-1} = \frac{\sum\limits_q \left(\mathbf{J}_{\Omega_q}^{-1} \cdot w_q \left| \mathbf{J}_{\Omega_q} \right| \right)}{\sum\limits_q w_q \left| \mathbf{J}_{\Omega_q} \right|}. \tag{5.10}$$

The components of the mapped gradient are compared and isotropic/anisotropic refinement can be initiated, as shown in Alg. 1.

Algorithm 1 Isotropic/anisotropic refinement

```
if ∇u_{iΩ,ξ} > 3∇u_{iΩ,η} then
    knot insertion at ξ = 0.5                    ▷ anisotropic along ξ
else if ∇u_{iΩ,η} > 3∇u_{iΩ,ξ} then
    knot insertion at η = 0.5                    ▷ anisotropic along η
else
    knot insertion at ξ = 0.5 and η = 0.5        ▷ isotropic
end if
```

5.1.3 Comparison of refinement criteria

The simulation of an incompressible laminar flow past a circular cylinder allows to compare the behavior of the different refinement criteria, as proposed by Naddei et al. [64] and Ims et al. [65]. The flow is characterized by a Mach number $M = 0.1$ and Reynolds number $Re = 40$. A $\mathbb{P}^2$ O-type mesh with 16 and 18 Bézier patches in the parametric directions is employed, where the freestream radius is set to $R = 20D$.

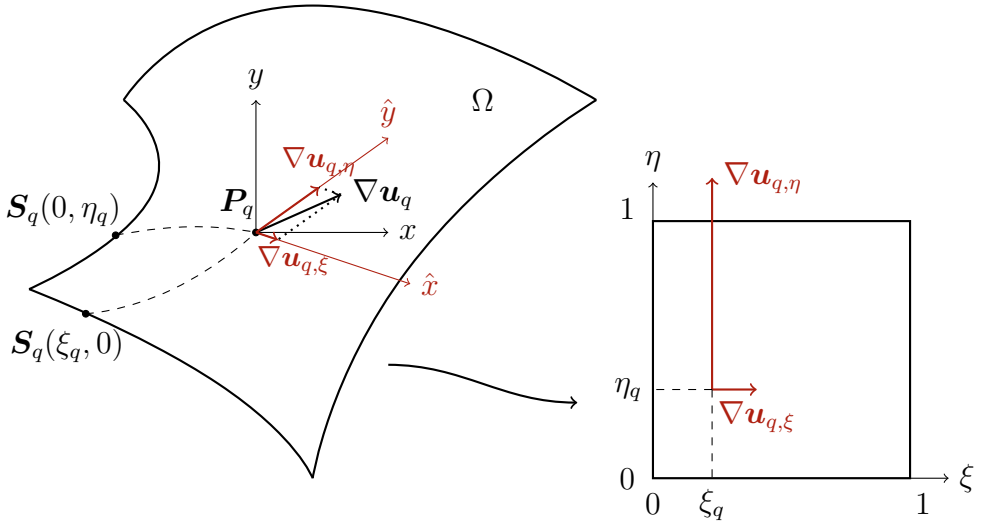


Figure 5.3: Projection of $\nabla \boldsymbol{u}_q$ onto the parametric space $\xi - \eta$ mapped in the physical space as $\hat{x} - \hat{y}$

In the radial direction the size of the patches changes with a ratio of 1.2. The h-adaptive algorithm is applied starting from a converged $\mathbb{P}^3$ solution.

The key objective is to investigate the convergence of the mesh with the different refinement criteria toward a reference value, that is, the drag coefficient C_d, previously computed through a simulation on a fine mesh, with 4608 elements, and with $\mathbb{P}^3$ solution approximation, corresponding to 73728 DoFs (see Fig. 5.4). The fine mesh is generated to reproduce the smallest element size resulting from the refinement. The parameters governing the refinement are listed in Tab. 5.7.

All indicators mark for refinement the area where the flow detaches from the cylinder and on the edges of the recirculation region (see Fig. 5.5). The SSED indicator appears to excessively refine the detachment area, resulting in an increased number of DoFs. The GNC indicator and the SSED indicator seem to adapt excessively flow areas outside the main regions of interest. The SD indicator seems to be the most effective overall, as it successfully marks the essential flow characteristics with a moderate number of DoFs. The local non-conformity error indicator shows similar performance and should be used for lower solution approximations. Figure 5.5 also demonstrates the capability of the indicators to adapt symmetrically a symmetric flow field; for example, the cylinder is symmetric with respect to the x-axis.

According to the evolution of the C_d convergence history of Fig. 5.6, all indicators guarantee a comparable computational improvement, reducing the number of DoFs with respect to the reference simulation, leading to a computational speed-up of ≈ 2 times. The SD indicator appears slightly more efficient in reaching the ref-

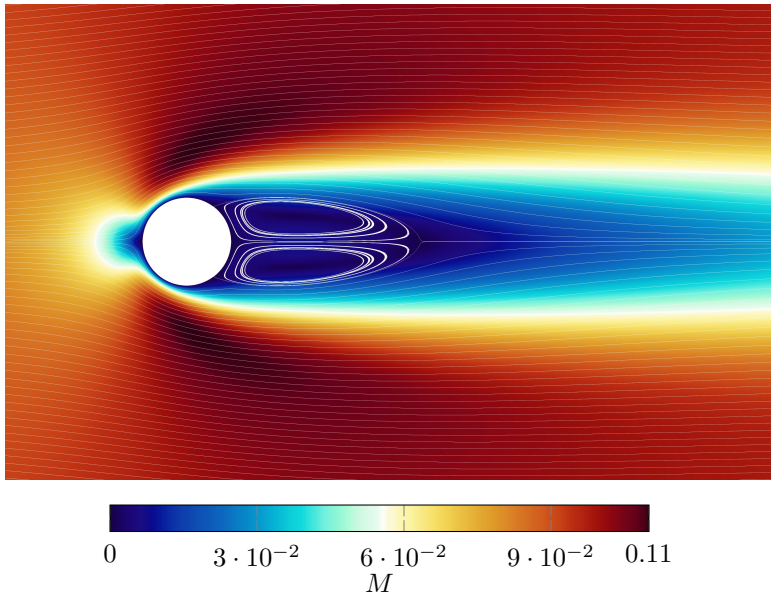


Figure 5.4: Cylinder. Streamlines of the x-component velocity and contour of the Mach number with the fine mesh, $\mathbb{P}^3$ solution approximation

erence value. Moreover, the anisotropic refinement shows a higher computational efficiency. Based on these considerations, the SD indicator is used for the following computations, as it showed the best trade-off between accuracy and DoFs reduction.

5.2 Numerical results

In this section, the assessment of the local refinement procedure is performed through a series of different test cases involving different key features that have to be described in a correct way, i.e. the interaction of different shock waves in the supersonic ramp case of Sec. 5.2.1, the transonic turbulent flow around a NACA0012 airfoil of Sec. 5.2.2, and, finally, the turbulent flow around the low pressure turbine blade T106A of Sec. 5.2.3.

5.2.1 Supersonic ramp

The inviscid supersonic flow ramp test case is characterized by an inlet Mach number $M = 2.0$. The geometry of the ramp, shown in Fig. 5.7 consists of a compression ramp of 10° followed by an expansion corner of 10° along the lower channel wall. The initial mesh, composed of 22×4 $\mathbb{P}^1$ Bézier patches, is obtained by building a Coons patch from the boundaries of the four sides. Information about the applied

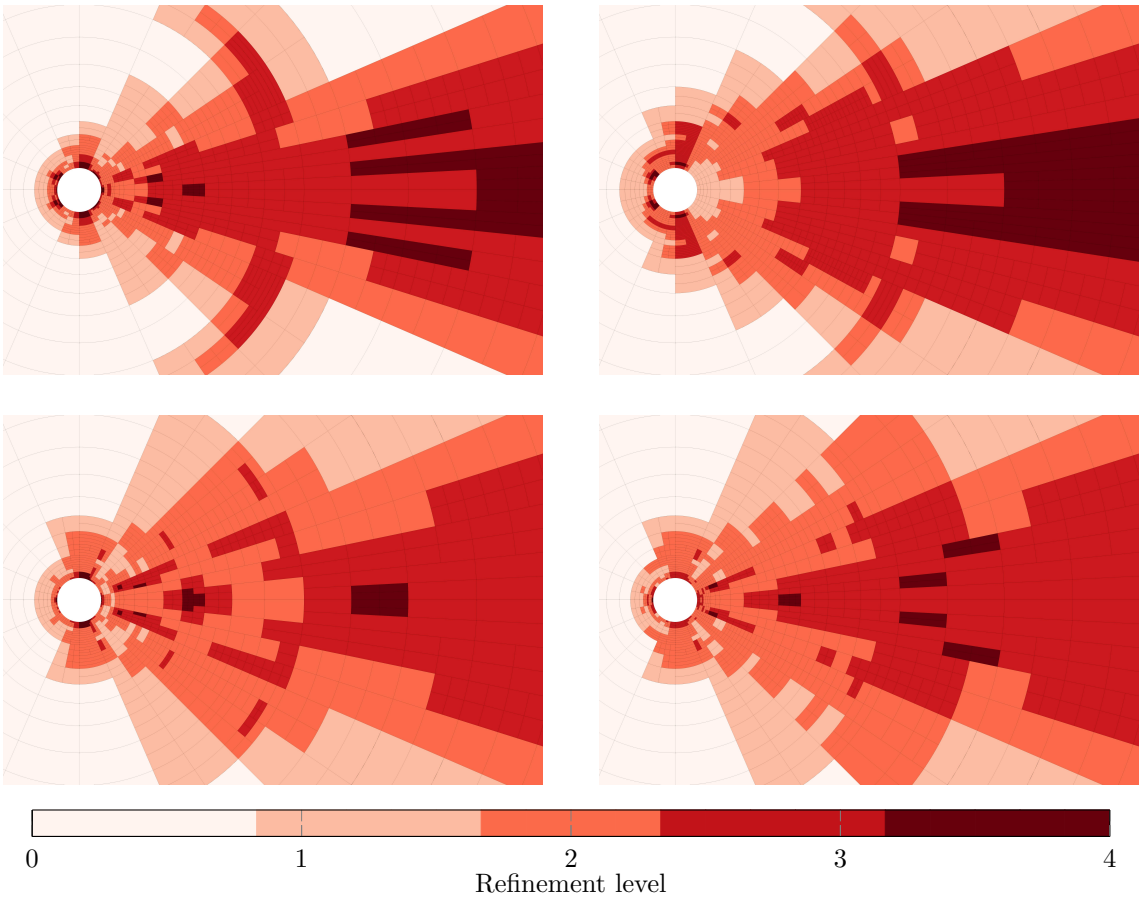


Figure 5.5: Cylinder. Refinement levels on the adapted mesh for the different indicators: SD (top, left), SSED (top, right), LNC (bottom, left), GNC (bottom, right), $\mathbb{P}^3$ solution approximation

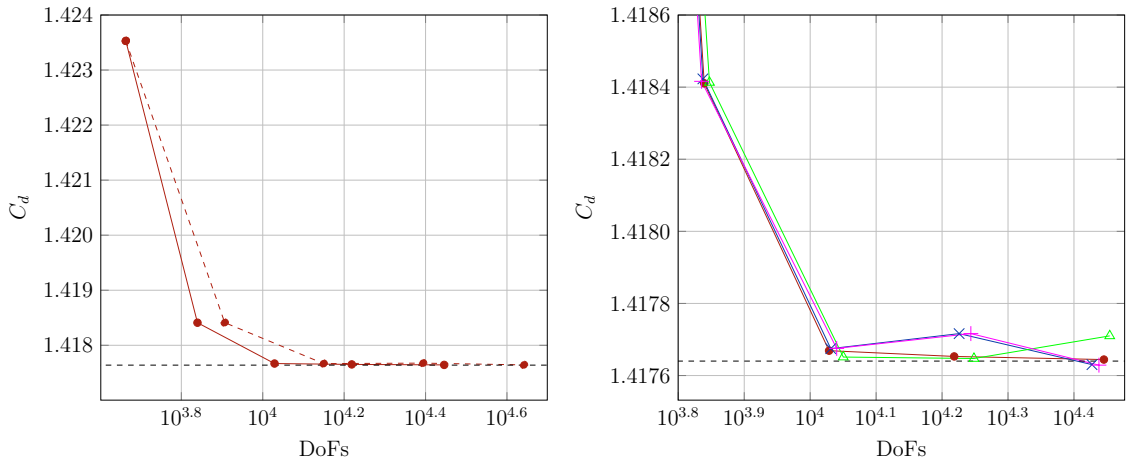


Figure 5.6: Cylinder. Evolution of the C_d convergence history for the different indicators and an anisotropic (solid lines) and isotropic (dashed lines) refinement, $\mathbb{P}^3$ solution approximation; SD, LNC, SSED, GNC, $C_{D_{\text{ref}}} = 1.41764$. The first point in the comparison of the different indicators is avoided to spotlight the differences in the results

boundary conditions can be found in Fig. 5.7. This simulation, characterized by a $\mathbb{P}^3$ solution approximation, involves oblique shock waves, shock reflections, and Prandtl - Meyer expansion fans, which analytical solutions can be found in many gas dynamics textbooks [66, 67].

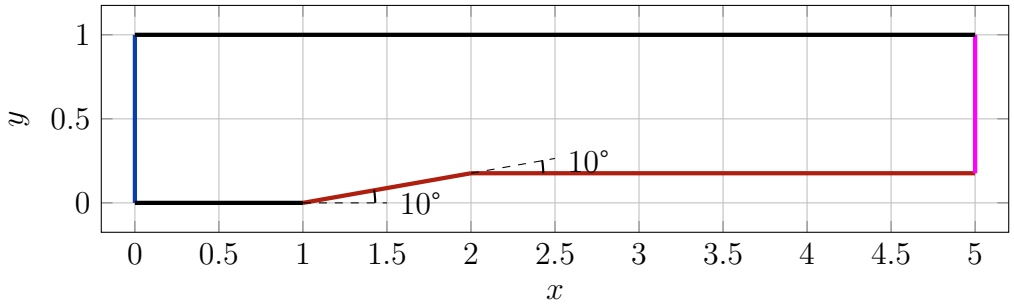


Figure 5.7: Supersonic ramp. Details of the geometry used for the simulation; supersonic inflow, symmetry, slip wall, supersonic outflow

An isotropic adaptation is performed using the parameters listed in Tab. 5.7. The variation in the number of elements and degrees of freedom for each cycle is listed in Tab. 5.1. The plot of the pressure coefficient in Fig. 5.8 shows the comparison between the distributions computed on the coarse and the fine mesh. The h-refinement procedure correctly targets the quantities of interest, allowing the

Table 5.1: Supersonic ramp. Number of elements, degrees of freedom, DoFs, and root mean square error with respect to fully adapted solution, RMSE, at each refinement cycle, *cyc*

cyc	DoFs	Elements	RMSE
0	1408	88	0.0555
1	3328	208	0.0337
2	8464	529	0.0241
3	21952	1372	0.0189
4	55360	3460	0.0137
5	136000	8500	-

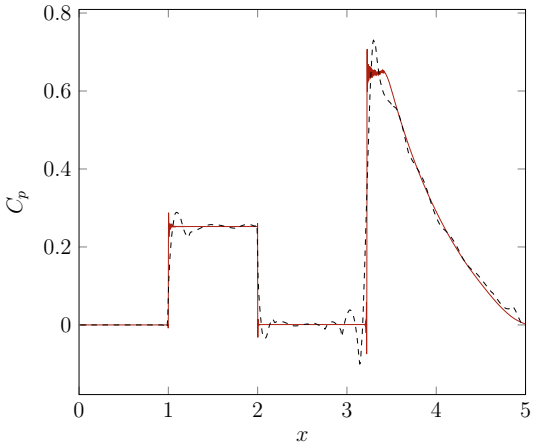


Figure 5.8: Supersonic ramp. Comparison of the pressure coefficient distribution C_p between coarse and fully adapted mesh; - - - coarse, —— adapted

solver to focus the computational effort on the key features of the flow, as shown in Fig. 5.9, where both the fully adapted mesh with isobar contours and the Mach number field are plotted. In Tab. 5.1, the root mean square error (RMSE) with respect to the fully adapted mesh solution is also reported and demonstrates how the error value falls in an acceptable range already after 3 refinement cycles.

The states computed in the test case are validated against analytical results up to the point where the expansion fan interacts with the reflected shock, after which the analytical solutions are no longer valid. Referring to Fig. 5.10, the oblique shock relations are applied between the states 1 and 2, while the Prandtl-Mayer expansion relations are employed between the states 2 and 3. Table 5.2 summarizes this comparison, showing the Mach numbers of the three states and the corresponding analytical values. In addition, the angle of the oblique shock for a $M = 2$ flow past a 10° incline is 39.3139°, while the predicted value is 39.3196°. The solver was

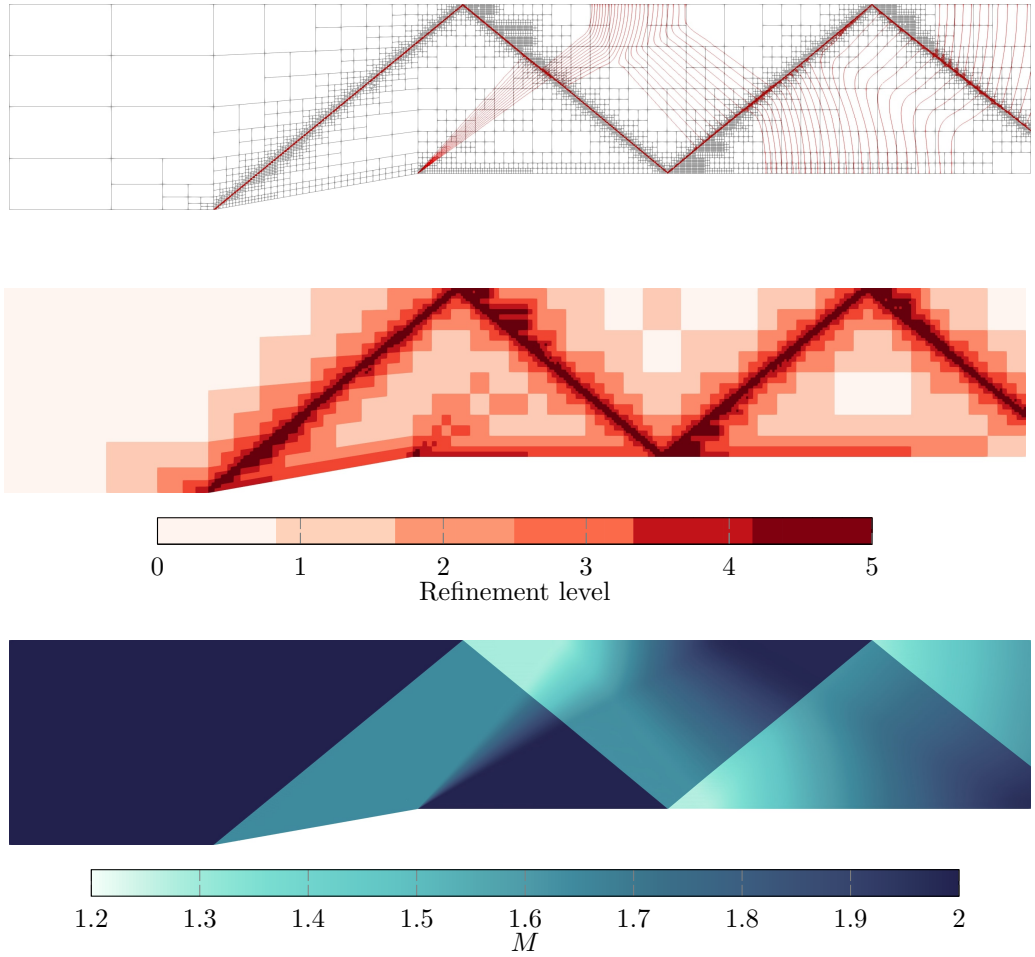


Figure 5.9: Supersonic ramp. Fully adapted mesh with isobar contours (top), refinement levels over the adapted mesh (middle) and Mach number field (bottom), $\mathbb{P}^3$ solution approximation

able to predict all relevant flow features, as an average error in the Mach number below 0.004% and an error in the shock angle below 0.015% are observed. This test case is also used to study the residuals convergence behavior when h-refinement is involved. Figure 5.11 shows the history of the residual norms for solution approximations $\mathbb{P}^0 \rightarrow \mathbb{P}^3$ and after each refinement cycle. The h-refinement has no impact on the convergence overall, and the quadratic behavior is still verified at every cycle.

5.2.2 NACA0012

The transonic turbulent flow around a NACA0012 airfoil is computed, where a single NURBS surface of degree $\mathbb{P}^2$ is used to represent the geometry of the airfoil. The flow

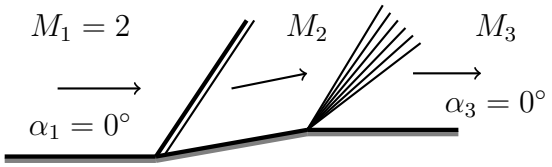


Figure 5.10: Supersonic ramp. Schematic representation of the test case, with an oblique shock followed by a Prandtl - Meyer expansion

Table 5.2: Supersonic ramp. Comparison between analytical and predicted Mach number at different states along the ramp

State	Analytical M	Computed M	%
1	2.00000	2.00000	$\pm 0.0\%$
2	1.64052	$1.64048 \div 1.64056$	$\pm 0.0024\%$
3	1.98835	$1.98827 \div 1.98837$	$\pm 0.0040\%$

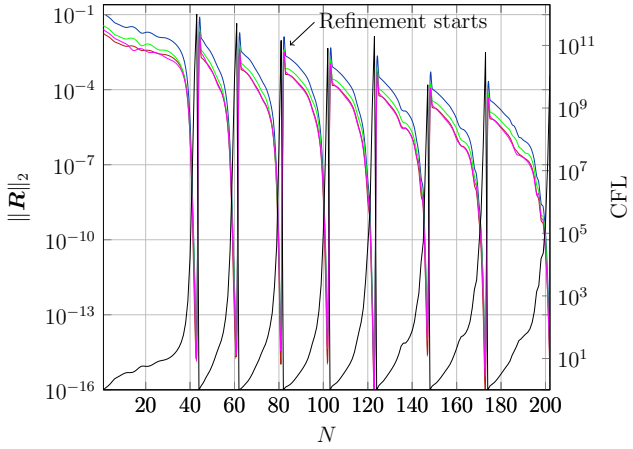


Figure 5.11: Supersonic ramp. Residuals convergence history; — ρ , — ρE, — ρu, — ρv, — CFL number

is characterized by a freestream Mach number $M_\infty = 0.751$ and Reynolds number $Re_\infty = 3.91 \cdot 10^6$, based on freestream velocity u_∞ and the airfoil chord and angle of attack $\alpha_\infty = 1.99°$. This test case is studied to asses the capability of the solver to correctly describe a transonic turbulent flow starting from a coarse mesh.

After a Bézier extraction procedure, a coarse mesh composed of $48 \times 35 = 1\,680$ Bézier patches is created. This mesh is then refined through a series of h-adaptation cycles. The coarse mesh presents high y^+ values along the airfoil that are reduced by the adaptation algorithm only where needed. The adaptation is performed using the parameters listed in Tab. 5.7. At the end of the adaptation cycles the mesh consists of 8 308 Bézier patches.

Figure 5.12 shows the different refinement levels around the airfoil, while Fig. 5.14 shows the x-component velocity and entropy contours and the distribution. The SD indicator targets the under-resolved areas of the domain, which lie along the shock, the boundary layer and the wake region. In particular, the h-adaptation procedure refines the separation bubble area, leading to a better location of the shock, and a more correct distribution of both the pressure and skin friction coefficient. Moreover, the anisotropic refinement behaves correctly in the boundary layer, subdividing the elements in the direction normal to the airfoil surface to decrease the local value of the y^+.

The results of the coarse and adapted meshes are compared with the experimental data of McDevitt et al. [68], and the reference results obtained with a fine mesh. The fine mesh presents the minimum size of the elements and y^+ comparable to the smallest values resulting from the h-adaptation procedure. The coarse, fine and adapted mesh are compared in Tab. 5.3, where the maximum, minimum and average values of the y^+, and the number of elements are reported. The refinement in the shock region allows to predict correctly the distribution of the pressure coefficient C_p along the airfoil, as demonstrated in Fig. 5.13, where the distributions of the pressure and skin friction coefficients, C_p and C_f, for the coarse, fine, and adapted mesh are compared. The skin friction coefficient C_f computed on the coarse mesh is completely wrong, while the adapted mesh solution follows the reference solution.

Finally, the effect of the parameter ne^{adp}, which sets the percentage of flagged elements, on the accuracy and computational efficiency is investigated, comparing the predicted results with the reference values obtained on the fine mesh. The results of this study are summarized in Tab. 5.4. The accuracy of the results is assessed by computing the deviation of the predicted drag coefficient C_d with respect to the reference value. All values of ne^{adp} guarantee a small deviation of the drag coefficient, but lower values of ne^{adp} correspond to a lower number of DoFs, and, as a consequence, to a greater computational efficiency (up to $\approx 6\times$ speed-up).

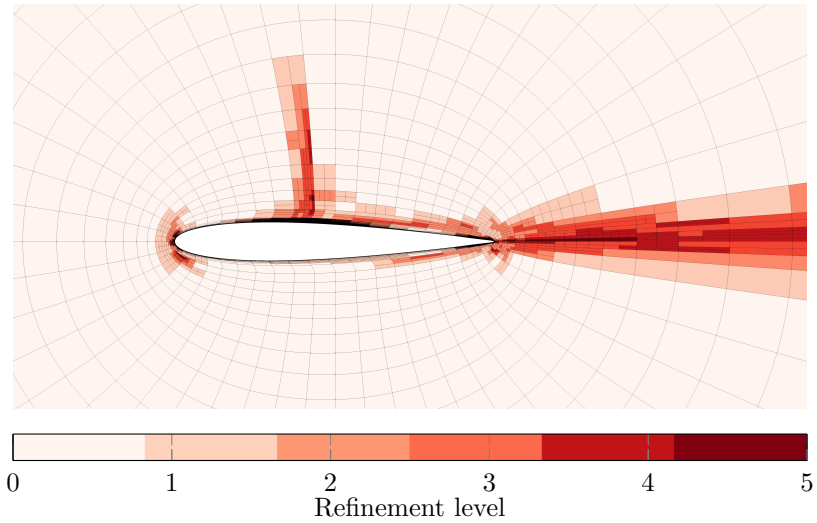


Figure 5.12: NACA0012. Refinement levels on the adapted mesh, $\mathbb{P}^3$ solution approximation

Table 5.3: NACA0012. Comparison of the y^+ distributions on the airfoil and number of elements for the coarse, fine, and adapted mesh

Mesh	$\overline{y^+}$	$y^+_{1,min}$	$y^+_{1,max}$	Elements
Coarse	177.399	0.553	339.476	1 680
Fine	9.182	0.079	16.052	15 360
Adapted	5.126	0.054	38.592	8 308

5.2.3 T106A

Finally, the subsonic turbulent flow through the T106A turbine blades is considered, where the geometry is defined by 9 NURBS surfaces of degree $\mathbb{P}^2$, which are arranged in a typical multi-block fashion for turbomachinery applications. The blade wall is considered adiabatic. At the inflow, the total temperature, the total pressure, the flow angle $\alpha_1 = 37.7°$ are prescribed, while at the outflow the static pressure is set, resulting in a downstream isentropic Mach number $M_2 = 0.59$. The Reynolds number is $Re = 1\,100\,000$, based on the downstream isentropic conditions and on the blade chord. The pitch to blase chord ration is $s/c = 0.799$. The objective of this test case is to asses the capability of the solver to correctly describe the flow in a turbomachinery application, starting from a coarse mesh with a y^+ value not generally suitable for turbulent flows computations.

After a Bézier extraction procedure, a coarse mesh composed of 933 Bézier patches is created. This mesh is then refined through a series of h-adaptation cycles. The coarse mesh presents high y^+ values along the blade that are reduced by the

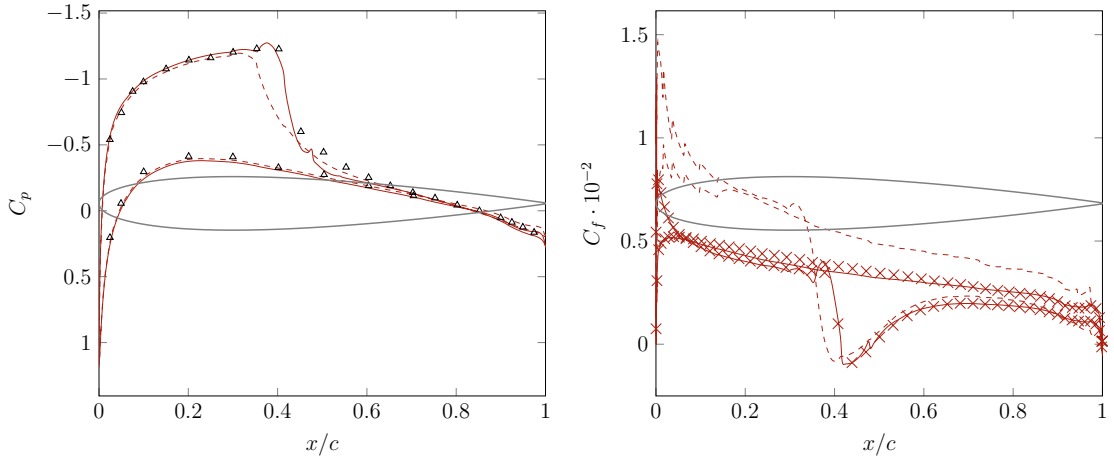


Figure 5.13: NACA0012. Comparison of the pressure C_p (left) and skin friction C_f (right) coefficient distributions on the airfoil with the different meshes with respect to the experimental data of McDevitt et al. [68], $\mathbb{P}^3$ solution approximation; —— adapted mesh, - - - coarse mesh, × fine mesh, ▵ McDevitt et al. exp. [68]

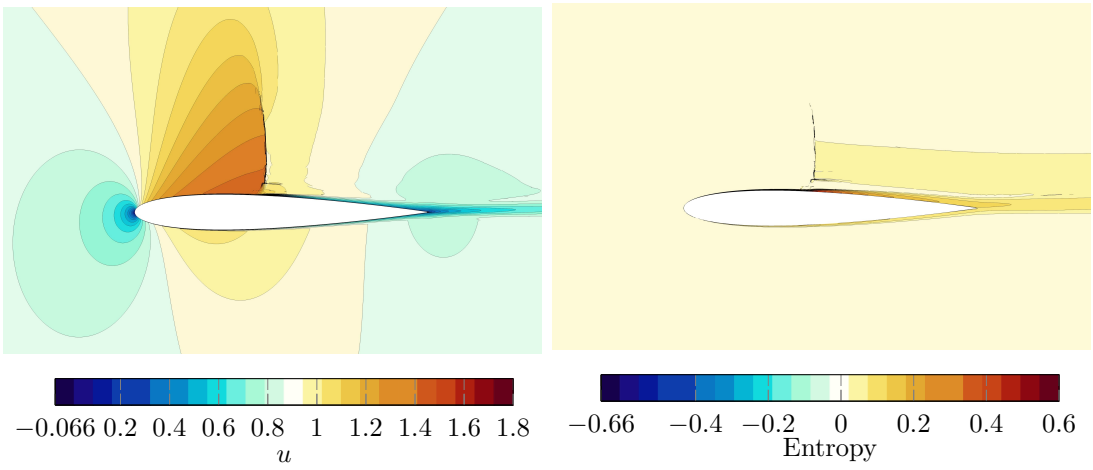


Figure 5.14: NACA0012. x-component velocity (left) and entropy (right) contours on the adapted mesh, $\mathbb{P}^3$ solution approximation

Table 5.4: NACA0012. Speed-up comparison for different fractions of refined elements and drag coefficient C_d deviation with respect to the reference value computed on the fine mesh ($C_{D_{\text{ref}}} = 1.80200 \cdot 10^{-2}$)

ne^{adp}	DoFs	Speed-up	C_d	$\frac{\lvert C_d - C_{D_{\text{ref}}} \rvert}{C_{D_{\text{ref}}}}$ [%]
6%	46528	$\approx 5.9\times$	$1.80805 \cdot 10^{-2}$	0.34%
12%	81232	$\approx 2.9\times$	$1.76780 \cdot 10^{-2}$	1.90%
20%	132928	$\approx 2.7\times$	$1.76816 \cdot 10^{-2}$	1.88%

adaptation algorithm only where needed. The adaptation is performed using the parameters listed in Tab. 5.7. At the end of the adaptation cycles the mesh consists of 5 106 Bézier patches.

Figure 5.15 shows the distribution of the different refinement levels around the blade, while Fig. 5.16 shows the Mach number and the entropy contours. The SD indicator targets the under-resolved areas of the domain, which lie in the boundary layer and wake region, with a particular focus on the leading and trailing edges, as well as along the first half of the pressure side. Moreover, the anisotropic refinement behaves correctly in the boundary layer, subdividing the elements in the direction normal to the airfoil surface to decrease the local value of the y^+.

The results of the coarse and adapted meshes are compared with experimental data of Hoheisel [69] and the reference results obtained with a fine mesh. The fine mesh presents the minimum size of the elements and y^+ comparable to the smallest values resulting from the h-adaptation procedure. The coarse, fine and adapted mesh are compared in Tab. 5.5, where the maximum, minimum and average values of the y^+, and the number of elements are reported. The pressure coefficient distribution does not change much between the coarse and the adapted mesh, while the predicted skin friction coefficient distribution along the suction side on the coarse mesh is slightly different than the reference distribution on the fine mesh, as demonstrated in Fig. 5.17. The predicted skin friction coefficient distribution along the suction side on the adapted mesh is in perfect agreement with the reference distribution.

One of the main advantages of the h-refinement procedure is the computational speed-up gained by increasing the DoFs only where it is necessary. Also for this test case the effect of the parameter ne^{adp}, which sets the percentage of flagged elements, on accuracy and computational efficiency is investigated, comparing the predicted results with the reference values obtained on the fine mesh. The results of this study are summarized in Tab. 5.6. The accuracy of the computed flow fields is assessed by computing the deviation of the predicted loss coefficient with respect to the reference value, which is defined as

$$\zeta = \frac{p_{0,1} - p_{0,2}}{p_{0,1} - p_2}, \tag{5.11}$$

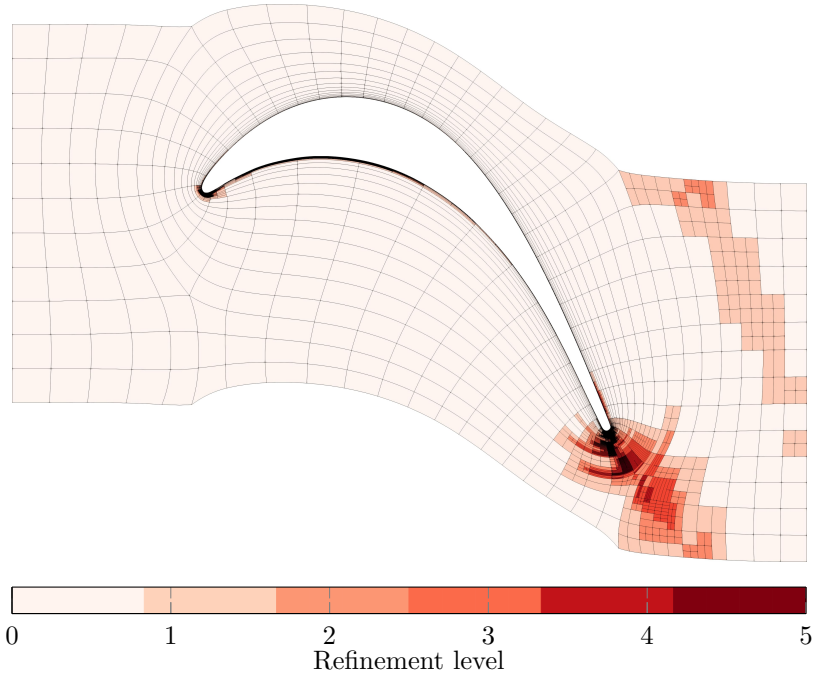


Figure 5.15: T106A. Refinement levels on the adapted mesh, $\mathbb{P}^3$ solution approximation

Table 5.5: T106A. Comparison of the y^+ distributions on the blade and number of elements for the coarse, fine, and adapted mesh

Mesh	$\overline{y^+}$	$y^+_{1,min}$	$y^+_{1,max}$	Elements
Coarse	31.019	5.763	76.869	933
Fine	0.821	0.0946	1.694	6847
Adapted	9.369	0.144	19.675	5106

where $p_{0,1}$ and $p_{0,2}$ are the mass flow-weighted averaged total pressures at the inlet and outlet section, respectively, and p_2 is the mass flow-weighted averaged static pressure at the outlet section. All values of ne^{adp} guarantee a small deviation of the loss coefficient ($< 1.5\%$), but lower values of ne^{adp} correspond to a lower number of DoFs, and, as a consequence, to a greater computational efficiency (up to $\approx 3\times$ speed-up).

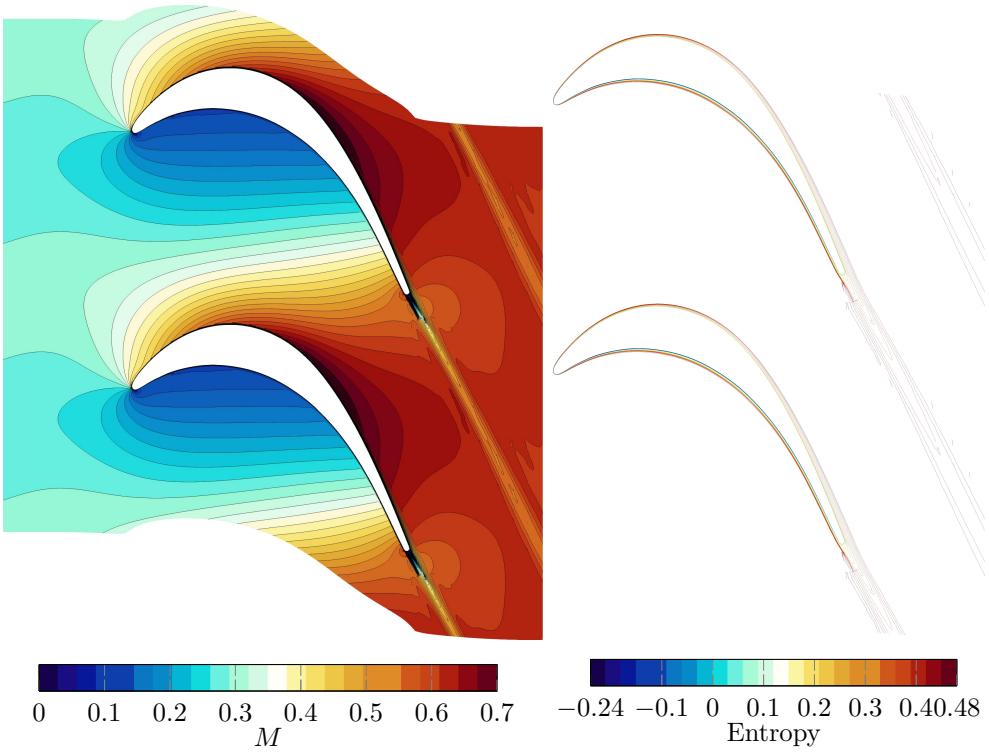


Figure 5.16: T106A. Mach number (left) and entropy (right) contours on the adapted mesh, $\mathbb{P}^3$ solution approximation

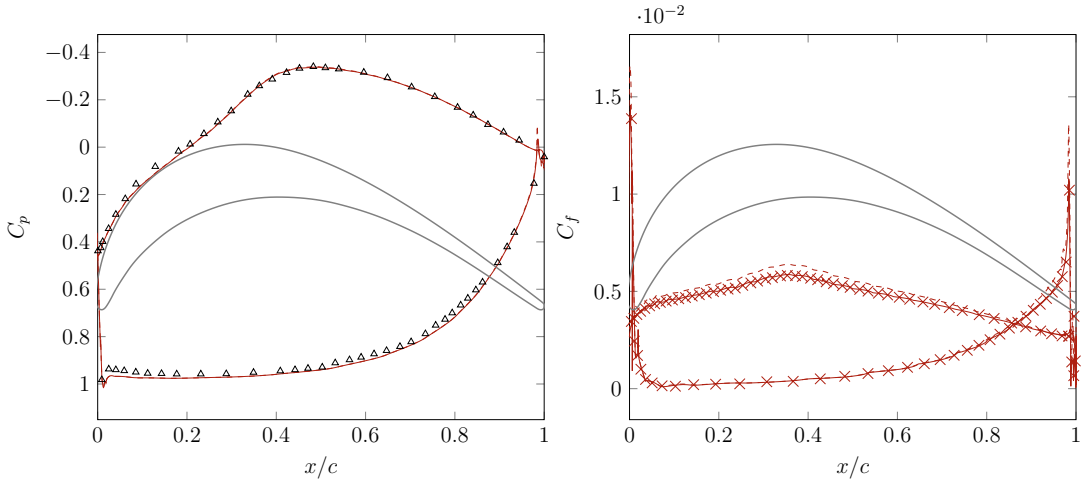


Figure 5.17: T106A. Comparison of the pressure C_p (left) and skin friction C_f (right) coefficient distributions on the blade with the different meshes with respect to the experimental data of Hoheisel [69], $\mathbb{P}^3$ solution approximation; —— adapted mesh, - - - coarse mesh, × fine mesh, ▵ Hoheisel exp. [69]

Table 5.6: T106A. Speed-up comparison for different fractions of refined elements and loss coefficient ζ deviation with respect to reference value computed on the fine mesh ($\zeta_{\text{ref}} = 2.417 \times 10^{-2}$)

ne^{adp}	DoFs	Speed-up	ζ	$\frac{\lvert\zeta-\zeta_{\text{ref}}\rvert}{\zeta_{\text{ref}}}$ [%]
5%	33392	$\approx 2.6\times$	$2.421 \cdot 10^{-2}$	0.17%
10%	56800	$\approx 1.7\times$	$2.381 \cdot 10^{-2}$	1.49%
15%	81696	$\approx 1.5\times$	$2.411 \cdot 10^{-2}$	0.24%

Table 5.7: Refinement parameters for the different test cases

Case	Indicator	$n^{adp}_{cyc,max}$	solution app.	ne^{adp}	lvl^{adp}_{max}
Cylinder	SD,SSED,LNC,GNC	4	$\mathbb{P}^3$	25%	4
Flat plate	SD	7	$\mathbb{P}^1$	7%	7
Ramp	SD	5	$\mathbb{P}^3$	45%	5
NACA0012	SD	4	$\mathbb{P}^2$	15%	4
T106A	SD	5	$\mathbb{P}^3$	15%	5

Chapter 6

Gradient-based shape optimization

This chapter presents the gradient-based shape optimization procedure used in this work. The discrete adjoint method is used for efficient gradient computation. The optimization process is carried out by applying the Sequential Quadratic Programming (SQP) approach, with a damped BFGS algorithm used to determine the search direction and merit function-based line search for the step calculation. The method's effectiveness is demonstrated through test cases focused on aerodynamic performance improvements.

6.1 The adjoint problem

The adjoint method is a robust technique widely used in sensitivity analysis [70], aerodynamic shape optimization [71, 72, 73], and output-based error estimation [74, 75, 76]. Its primary advantage lies in its ability to compute gradients efficiently, with a computational cost that is independent of the number of design variables. This efficiency makes the adjoint method exceptionally well suited for large-scale optimization problems, which often involve a high-dimensional design space. Optimization problems for systems governed by PDEs typically require expensive derivative computations. However, the adjoint method avoid these costs by constructing and solving a linear system, providing the required derivative information with significantly reduced computational overhead.

There are two main approaches to the adjoint method: the continuous one and the discrete one [77]. The continuous adjoint method involves differentiating the governing PDEs prior to their discretization. In contrast, the discrete adjoint method is applied after the discretization of the equations. Each approach has its advantages and is suited for different applications, but the discrete adjoint method is particularly favored in cases where strict consistency between the adjoint and primal solutions is critical, as in high-order methods.

In this work, the discrete adjoint method is employed. Consider a discrete state vector $\boldsymbol{Q} \in \mathbb{R}^N$ and a set of design variables $\boldsymbol{\mu} \in \mathbb{R}^{N_\mu}$. A scalar output of interest, such as lift, drag, or aerodynamic efficiency, can be expressed as a functional of the state vector and the design variables:

$$\mathcal{F} = \mathcal{F}(\boldsymbol{Q}, \boldsymbol{\mu}). \tag{6.1}$$

The goal of the adjoint method is to compute the sensitivity of this output with respect to a change in the residual vector. To achieve this, the discrete adjoint vector, denoted as $\boldsymbol{\Psi} \in \mathbb{R}^N$, is introduced. This vector represents the sensitivity of the output $\mathcal{F}$ to each component of the residual vector $\boldsymbol{R}(\boldsymbol{Q}, \boldsymbol{\mu})$. These changes in the residual vector are usually caused by modifications to the input parameters, $\boldsymbol{\mu}$. By solving the adjoint problem, which is done after solving the primal solution problem, one can compute the gradients of $\mathcal{F}$ with respect to the design variables $\boldsymbol{\mu}$, allowing gradient-based optimization at much lower computational cost than approaches that scale with the number of input parameters.

6.1.1 Adjoint for gradient calculation

To perform gradient-based optimization, it is necessary to compute the gradient of all objectives and constraints with respect to all design variables. For the scalar output $\mathcal{F}$, this gradient is determined by applying the chain rule:

$$\frac{d\mathcal{F}}{d\boldsymbol{\mu}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} + \frac{\partial \mathcal{F}}{\partial \boldsymbol{Q}} \frac{d\boldsymbol{Q}}{d\boldsymbol{\mu}}. \tag{6.2}$$

The term $\frac{d\boldsymbol{Q}}{d\boldsymbol{\mu}}$ can be evaluated by applying the chain rule to the residual $\boldsymbol{R}$, and taking the total derivative with respect to the design variables. Since the residuals are driven to zero at each design point, their total derivative satisfies:

$$\frac{d\boldsymbol{R}}{d\boldsymbol{\mu}} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\mu}} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}} \frac{d\boldsymbol{Q}}{d\boldsymbol{\mu}} = 0. \tag{6.3}$$

Rearranging this equation yields:

$$\frac{d\boldsymbol{Q}}{d\boldsymbol{\mu}} = -\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right)^{-1} \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\mu}}. \tag{6.4}$$

Substituting this expression for $\frac{d\boldsymbol{Q}}{d\boldsymbol{\mu}}$ back into Eq. (6.2), the gradient of $\mathcal{F}$ becomes:

$$\frac{d\mathcal{F}}{d\boldsymbol{\mu}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} - \frac{\partial \mathcal{F}}{\partial \boldsymbol{Q}} \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right)^{-1} \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\mu}}. \tag{6.5}$$

This formulation can be evaluated directly by solving the linear system $\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right)^{-1} \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\mu}}$, an approach known as the direct method. However, this method is computationally expensive because it requires solving a linear system for each design variable. The adjoint method introduces the adjoint vector $\boldsymbol{\Psi}$, obtained by solving the adjoint problem:

$$\boldsymbol{\Psi} = -\left(\frac{\partial \boldsymbol{R}^\intercal}{\partial \boldsymbol{Q}}\right)^{-1} \frac{\partial \mathcal{F}^\intercal}{\partial \boldsymbol{Q}}. \tag{6.6}$$

This adjoint system depends only on the number of output functions of interest, rather than the number of design variables, significantly reducing computational cost. Once the adjoint vector $\boldsymbol{\Psi}$ is computed, the gradient of $\mathcal{F}$ can be expressed as:

$$\frac{d\mathcal{F}}{d\boldsymbol{\mu}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} + \boldsymbol{\Psi}^\intercal \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\mu}}, \quad \frac{d\mathcal{F}}{d\boldsymbol{\mu}} \in \mathbb{R}^{1 \times N_\mu} = N_\mu \text{ sensitivities.} \tag{6.7}$$

The remaining partial derivatives on the right-hand side of Eq. (6.7) can be computed efficiently, as they do not require solving the residual equations again. In this work, these derivatives are calculated using finite differences. For each design variable, a small perturbation is introduced and the residual vector and the output in the perturbed state are recomputed. Then a forward difference approximation is applied to estimate the gradients.

Fig. 6.1 shows the x-momentum contours for an inviscid supersonic flow around a diamond airfoil with a maximum thickness equal to 15% of the chord length. The imposed Mach number at the farfield is $M = 1.5$, while the angle of attack is $\alpha = 0°$. Fig. 6.1 also shows the associated adjoint for a scalar output $\mathcal{F}$, which in this case is a pressure line integral. The adjoint field shows how the pressure integral is not affected by residuals in areas from which any kind of information cannot propagate toward the interested line. This result is in agreement with the properties of supersonic flows, where the information propagates downstream within a cone of angle $\frac{1}{M}$, called Mach angle. There is no upstream influence in a supersonic flow.

6.1.2 Local sensitivity analysis

In the context of fluid dynamics, the parameters used to define the geometry or to set the boundary conditions can be chosen as design variables. These variables, defined as $\boldsymbol{\mu} \in \mathbb{R}^{N_\mu}$, affect the residual vector $\boldsymbol{R}$, which in turn impacts the state vector used to compute the output of interest. This leads to the formulation of the following sequence of dependencies:

$$\underbrace{\boldsymbol{\mu}}_{\text{inputs} \in \mathbb{R}^{N_\mu}} \longrightarrow \underbrace{\boldsymbol{R}(\boldsymbol{Q}, \boldsymbol{\mu}) = 0}_{N \text{ equations}} \longrightarrow \underbrace{\boldsymbol{Q}}_{\text{state} \in \mathbb{R}^{N}} \longrightarrow \underbrace{\mathcal{F}(\boldsymbol{Q})}_{\text{scalar output}}. \tag{6.8}$$

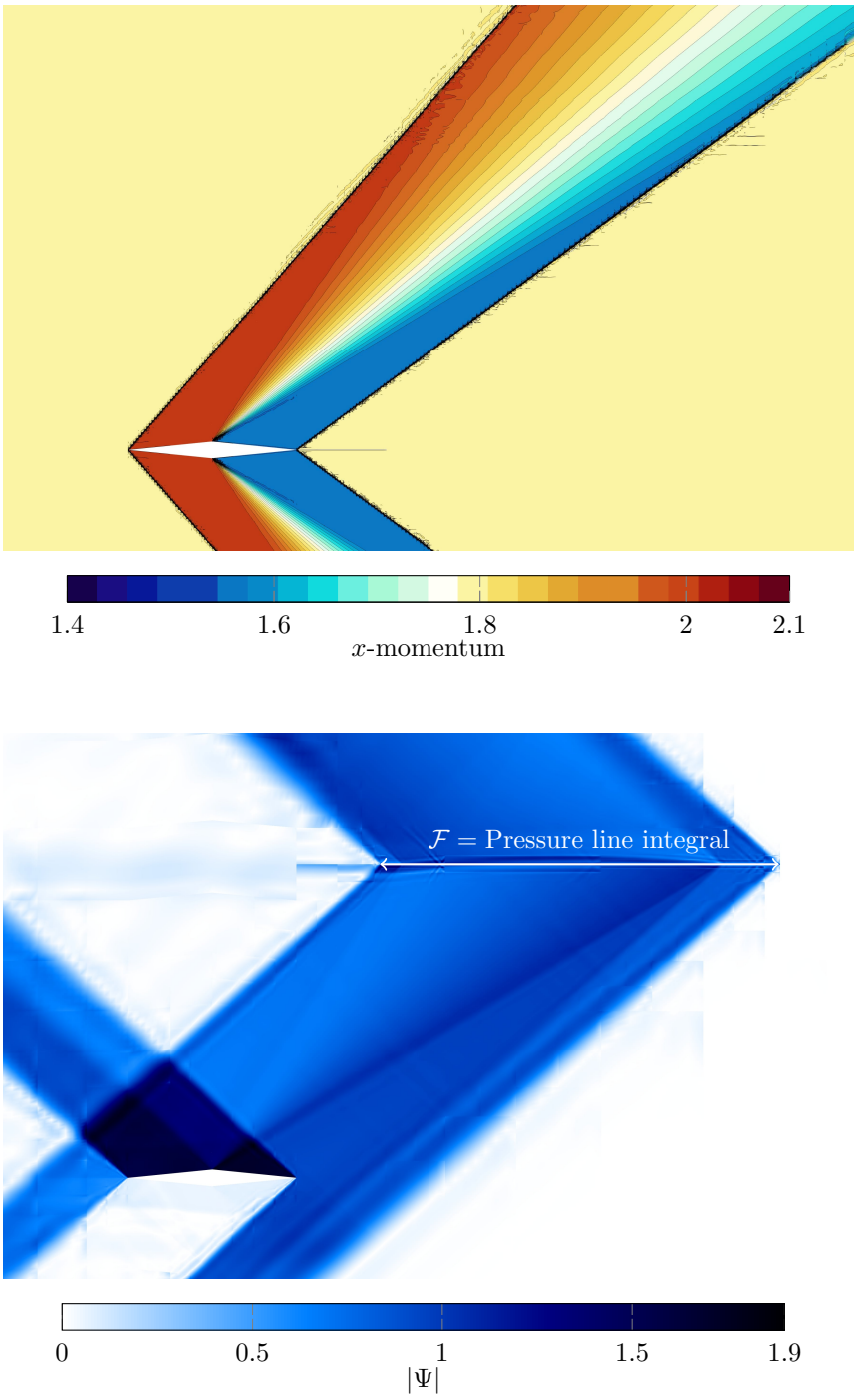


Figure 6.1: Diamond airfoil. x-momentum contours (top) and its associated adjoint for a pressure line integral $\mathcal{F}$ (bottom), $\mathbb{P}^3$ solution approximation

One way to test a discrete adjoint is to compare the output sensitivities computed using the adjoint with the results obtained by perturbing an input variable and solving again the primal problem. In this section, the comparison for the supersonic laminar flow over a NACA 0012 airfoil at $M = 2.0$, $Re = 106$, $\alpha = 10°$, presented in Sec. 4.3.2. The focus is on the sensitivities of the lift, drag, and moment coefficients with respect to changes in the angle of attack. The moment coefficient is calculated with respect to the leading edge. Since these coefficients depend on the freestream direction, they are directly dependent on the input parameters. Therefore, it is necessary to extend Eq. (6.7) by incorporating the partial derivative of $\mathcal{F}$ with respect to the input angle of attack, α, as follows:

$$\frac{d\mathcal{F}}{d\alpha} = \boldsymbol{\Psi}^\intercal \frac{\partial \boldsymbol{R}}{\partial \alpha} + \frac{\partial \mathcal{F}}{\partial \alpha}. \tag{6.9}$$

Fig. 6.2 illustrates the x-momentum field and the respective adjoint sensitivities for the lift, drag, and moment coefficients with respect to the angle of attack. Fig. 6.3 presents a comparison between the locally linearized sensitivity calculations using Eq. (6.9) and the results derived from the direct varying of the angle of attack, showing perfect agreement.

6.2 Aerodynamic shape optimization

Aerodynamic shape optimization can be formulated as a constrained optimization problem where the design parameters, $\boldsymbol{\mu} \in \mathbb{R}^{n_\mu}$, are adjusted to minimize an objective function $\mathcal{F}(\boldsymbol{\mu}) \in \mathbb{R}^1$, subject to n_h equality constraints $\boldsymbol{h}(\boldsymbol{\mu}) = 0$ and n_g inequality constraints $\boldsymbol{g}(\boldsymbol{\mu}) \leq 0$, where $\boldsymbol{h} \in \mathbb{R}^{n_h}$ and $\boldsymbol{g} \in \mathbb{R}^{n_g}$. The optimization problem can be expressed as

$$\begin{aligned} &\text{minimize } \mathcal{F}(\boldsymbol{\mu}) \\ &\text{by varying } \boldsymbol{\mu} \\ &\text{subject to } g_j(\boldsymbol{\mu}) \leq 0, \\ &h_l(\boldsymbol{\mu}) = 0, \end{aligned} \tag{6.10}$$

which can be solved by combining the objective function and the constraints into a Lagrangian function:

$$\mathcal{F} = \mathcal{F}(\boldsymbol{\mu}) + \boldsymbol{\lambda}^\intercal \boldsymbol{h}(\boldsymbol{\mu}) + \boldsymbol{\sigma}^\intercal (\boldsymbol{g}(\boldsymbol{\mu}) + \boldsymbol{s} \odot \boldsymbol{s}), \tag{6.11}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{n_h}$ and $\boldsymbol{\sigma} \in \mathbb{R}^{n_g}$ are the Lagrange multipliers for equality and inequality constraints, respectively. The slack variable $\boldsymbol{s} \in \mathbb{R}^{n_g}$ transforms inequality constraints into equality constraints and $\odot$ denotes element-wise multiplication.

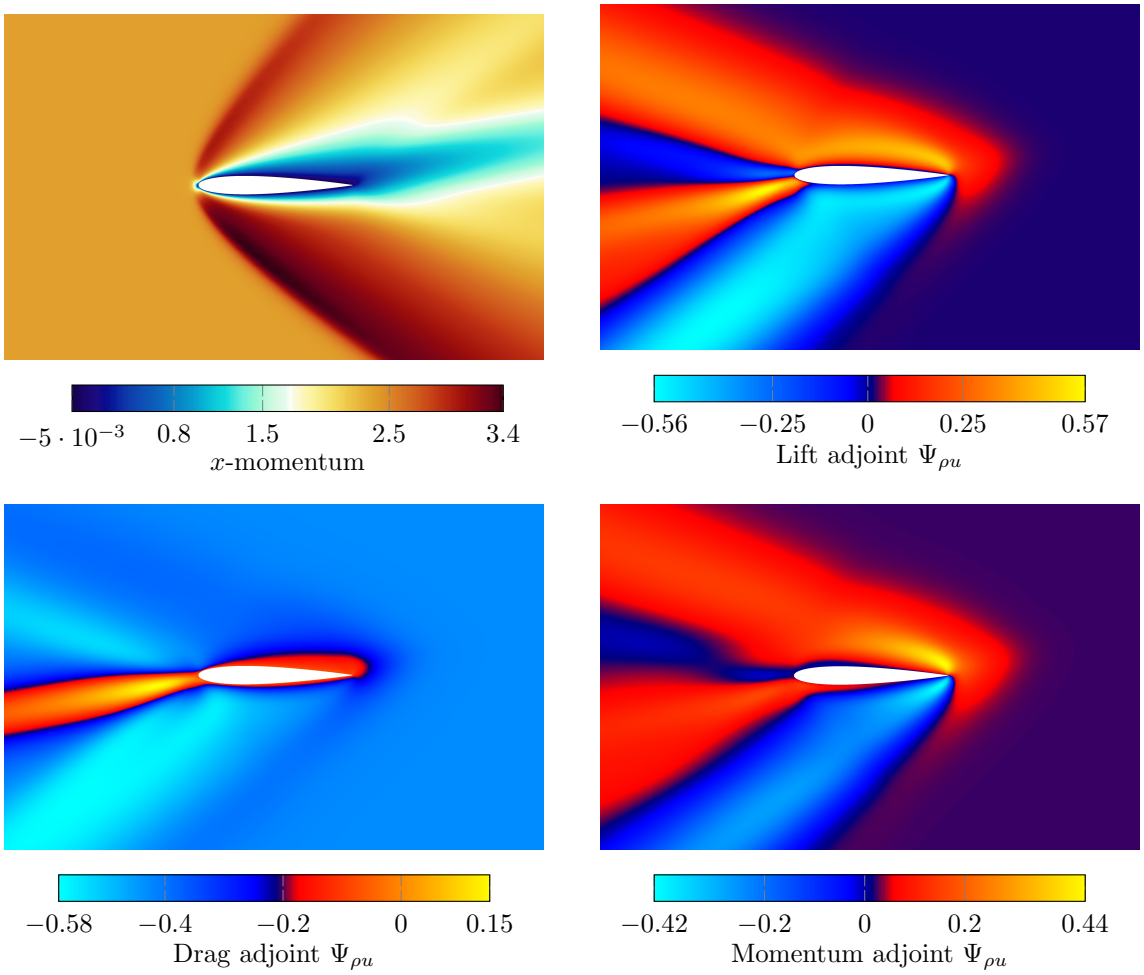
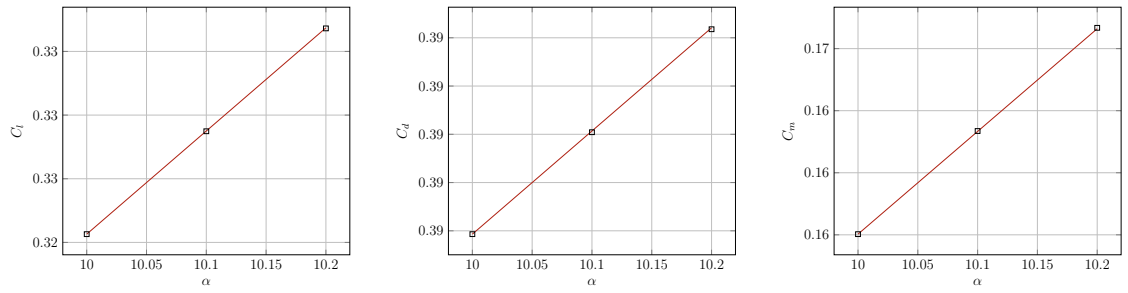


Figure 6.2: NACA0012. x-momentum and respective adjoints (fields $Re = 106$, $M_\infty = 2.00, \alpha = 10°$)



(a) Lift coefficient (b) Drag coefficient (c) Momentum coefficient

Figure 6.3: NACA0012. Lift, drag, and momentum coefficients adjoint-based sensitivities comparison with solver results; — adjoint linearized sensitivity, ▫ solver results ($Re = 106$, $M_\infty = 2.00, \alpha = 10°$)

Taking the derivatives of the Lagrangian with respect to the design variables $\boldsymbol{\mu}$, Lagrange multipliers $\boldsymbol{\lambda}$ and $\boldsymbol{\sigma}$, and the slack variable $\boldsymbol{s}$, and setting them to zero, yield the following system of equations:

$$\nabla_\mu \mathcal{F} = \nabla_\mu \mathcal{F} + \boldsymbol{J}_h^\intercal \boldsymbol{\lambda} + \boldsymbol{J}_g^\intercal \boldsymbol{\sigma} = \boldsymbol{0}, \tag{6.12}$$

$$\nabla_\lambda \mathcal{F} = \boldsymbol{h} = \boldsymbol{0}, \tag{6.13}$$

$$\nabla_\sigma \mathcal{F} = \boldsymbol{g} + \boldsymbol{s} \odot \boldsymbol{s} = \boldsymbol{0}, \tag{6.14}$$

$$\nabla_s \mathcal{F} = \boldsymbol{\sigma} \odot \boldsymbol{s} = \boldsymbol{0}, \tag{6.15}$$

$$\boldsymbol{\sigma} \geq \boldsymbol{0}. \tag{6.16}$$

Here, $\boldsymbol{J}_h$ and $\boldsymbol{J}_g$ are the Jacobians of the equality and inequality constraints, defined as:

$$\boldsymbol{J}_h = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{\mu}} = \begin{bmatrix} \frac{\partial h_1}{\partial \mu_1} & \cdots & \frac{\partial h_1}{\partial \mu_{n_\mu}} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_{n_h}}{\partial \mu_1} & \cdots & \frac{\partial h_{n_h}}{\partial \mu_{n_\mu}} \end{bmatrix} = \begin{bmatrix} \nabla h_1^\intercal \\ \vdots \\ \nabla h_{n_h}^\intercal \end{bmatrix}, \quad (n_h \times n_\mu), \tag{6.17}$$

$$\boldsymbol{J}_g = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\mu}} = \begin{bmatrix} \frac{\partial g_1}{\partial \mu_1} & \cdots & \frac{\partial g_1}{\partial \mu_{n_\mu}} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_{n_g}}{\partial \mu_1} & \cdots & \frac{\partial g_{n_g}}{\partial \mu_{n_\mu}} \end{bmatrix} = \begin{bmatrix} \nabla g_1^\intercal \\ \vdots \\ \nabla g_{n_g}^\intercal \end{bmatrix}, \quad (n_g \times n_\mu). \tag{6.18}$$

The conditions given in Equations (6.12) to (6.16) are known as the Karush-Kuhn-Tucker (KKT) conditions [78]. These conditions must be satisfied at the optimal point, corresponding to a stationary point of the Lagrangian. Solving this system requires knowing the values of the objective function and constraints, as well as their gradients with respect to the design variables. At the optimal point, the first KKT condition simplifies to:

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} = -\boldsymbol{\sigma}^\intercal \frac{\partial \boldsymbol{g}_a}{\partial \boldsymbol{\mu}} - \boldsymbol{\lambda}^\intercal \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{\mu}}, \tag{6.19}$$

where $\boldsymbol{g}_a$ denotes the set of active inequality constraints. By expressing the differential of the objective as $d\mathcal{F} = \left(\frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}}\right) d\boldsymbol{\mu}$ and combining active inequality constraints with equality constraints, this equation can be rewritten as:

$$d\mathcal{F} = -\boldsymbol{\lambda}^\intercal \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{\mu}} d\boldsymbol{\mu}. \tag{6.20}$$

This leads to the definition of the Lagrange multiplier λ_i for a given constraint i as:

$$\lambda_i = -\frac{d\mathcal{F}}{dh_i}. \tag{6.21}$$

The Lagrange multipliers thus represent the sensitivity of the objective function to changes in the constraints. In other words, they quantify how much the objective would vary if the constraints were relaxed.

6.3 Sequential quadratic programming

The first Sequential Quadratic Programming (SQP) technique to solve constrained nonlinear optimization problems was introduced by Wilson, R.B. in his Ph.D. thesis [79]. In the following years, SQP techniques have developed into a powerful and successful class of techniques for a variety of optimization issues [80, 81]. Conventional SQP methods find an approximate solution of a sequence of quadratic programming (QP) sub-problems in which a quadratic model of the objective function is minimized subject to the linearized constraints.

6.3.1 Equality constrained SQP

A QP problem is an optimization problem with a quadratic objective and linear constraints. In a general form, any equality constrained QP can be expressed as

$$\underset{x}{\text{minimize}} \quad \frac{1}{2}\boldsymbol{x}^\intercal \boldsymbol{H}\boldsymbol{x} + \boldsymbol{q}^\intercal \boldsymbol{x} \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} = \boldsymbol{0}. \tag{6.22}$$

The constraint is a matrix equation that represents multiple linear equality constraints, one for every row in $\boldsymbol{A}$. This optimization problem can be solved analytically using the optimality conditions. First, the Lagrangian is formed as:

$$\mathcal{F}(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{\mu}^\intercal \boldsymbol{H}\boldsymbol{\mu} + \boldsymbol{q}^\intercal \boldsymbol{\mu} + \boldsymbol{\lambda}^\intercal(\boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b}). \tag{6.23}$$

The partial derivatives are then set to zero:

$$\nabla_\mu \mathcal{F} = \boldsymbol{H}\boldsymbol{\mu} + \boldsymbol{q} + \boldsymbol{A}^\intercal \boldsymbol{\lambda} = \boldsymbol{0}, \tag{6.24}$$

$$\nabla_\lambda \mathcal{F} = \boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{b} = \boldsymbol{0}. \tag{6.25}$$

These equations can be written in matrix form as

$$\begin{bmatrix} \boldsymbol{H} & \boldsymbol{A}^\intercal \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{q} \\ -\boldsymbol{b} \end{bmatrix}. \tag{6.26}$$

These are linear equations, so they can be solved directly without any iteration.

As long as $\boldsymbol{H}$ is positive definite, the linear system always has a solution corresponding to the global minimum of the QP problem. The relative ease of solving a QP motivates the use of this technique. For a general constrained optimization problem, the nonlinear problem is locally approximated by a QP, which is then solved. This process is repeated iteratively until convergence is reached, requiring the solution of a sequence of QP problems, hence the name sequential quadratic programming.

To formulate the QP, the Lagrangian is approximated quadratically, while the constraints are approximated linearly for a step $\boldsymbol{p}$ near the current point in the optimization process. This leads to the following locally approximated QP:

$$\underset{\boldsymbol{p}}{\text{minimize}} \quad \frac{1}{2}\boldsymbol{p}^\intercal \boldsymbol{H}_{\mathcal{L}} \boldsymbol{p} + \nabla_\mu \mathcal{F}^\intercal \boldsymbol{p} \quad \text{subject to} \quad \boldsymbol{J}_h \boldsymbol{p} + \boldsymbol{h} = \boldsymbol{0}, \tag{6.27}$$

where $\boldsymbol{H}_{\mathcal{L}}$ is the Hessian of the Lagrangian with respect to the design variables. Plugging the gradient of the Lagrangian into the objective yields:

$$\frac{1}{2}\boldsymbol{p}^\intercal \boldsymbol{H}_{\mathcal{L}} \boldsymbol{p} + \nabla \mathcal{F}^\intercal \boldsymbol{p} + \boldsymbol{\lambda}^\intercal \boldsymbol{J}_h \boldsymbol{p}. \tag{6.28}$$

The constraint $\boldsymbol{J}_h \boldsymbol{p} = -\boldsymbol{h}$ can then be substituted into the objective, resulting in:

$$\frac{1}{2}\boldsymbol{p}^\intercal \boldsymbol{H}_{\mathcal{L}} \boldsymbol{p} + \nabla \mathcal{F}^\intercal \boldsymbol{p} - \boldsymbol{\lambda}^\intercal \boldsymbol{h}. \tag{6.29}$$

The last term in Eq. (6.29) can be omitted, as it does not depend on the step $\boldsymbol{p}$. This simplifies the problem to the following equivalent form:

$$\underset{\boldsymbol{p}}{\text{minimize}} \quad \frac{1}{2}\boldsymbol{p}^\intercal \boldsymbol{H}_{\mathcal{L}} \boldsymbol{p} + \nabla \mathcal{F}^\intercal \boldsymbol{p} \quad \text{subject to} \quad \boldsymbol{J}_h \boldsymbol{p} + \boldsymbol{h} = \boldsymbol{0}. \tag{6.30}$$

Using the QP solution method, the problem can be reduced to solving the following system of linear equations:

$$\begin{bmatrix} \boldsymbol{H}_{\mathcal{L}} & \boldsymbol{J}_h^\intercal \\ \boldsymbol{J}_h & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_\mu \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla \mathcal{F} \\ -\boldsymbol{h} \end{bmatrix}. \tag{6.31}$$

Replacing $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda} + \boldsymbol{p}_\lambda$ and multiplying through yields:

$$\begin{bmatrix} \boldsymbol{H}_{\mathcal{L}} & \boldsymbol{J}_h^\intercal \\ \boldsymbol{J}_h & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_\mu \\ \boldsymbol{p}_\lambda \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_h^\intercal \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{\lambda}_k = \begin{bmatrix} -\nabla \mathcal{F} \\ -\boldsymbol{h} \end{bmatrix}. \tag{6.32}$$

Finally, subtracting the second term from both sides, and using the definition in Eq. (6.12) simplifies the system to:

$$\begin{bmatrix} \boldsymbol{H}_{\mathcal{L}} & \boldsymbol{J}_h^\intercal \\ \boldsymbol{J}_h & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_\mu \\ \boldsymbol{p}_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla_\mu \mathcal{F} \\ -\boldsymbol{h} \end{bmatrix}. \tag{6.33}$$

This system consists of $n_\mu + n_h$ linear equations, where the Jacobian matrix $\boldsymbol{J}_h$ is square. The shape of the matrix and its blocks is shown in Fig. 6.4. A sequence of these QP problems is solved iteratively to converge to the optimal design variables and the corresponding optimal Lagrange multipliers. At each iteration, the design variables and Lagrange multipliers are updated as follows:

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k + \alpha_k \boldsymbol{p}_\mu, \tag{6.34}$$

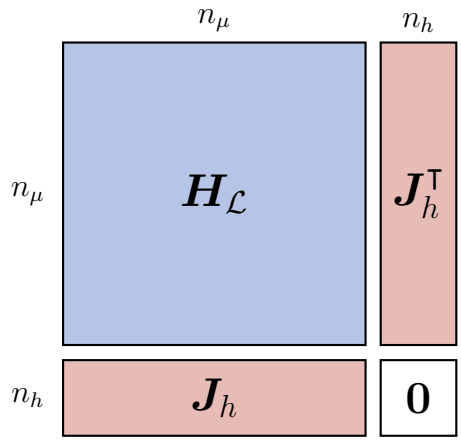


Figure 6.4: SQP system dimensions

$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \boldsymbol{p}_\lambda. \tag{6.35}$$

The parameter α_k indicates that the full Newton step (corresponding to $\alpha_k = 1$) is not automatically accepted. Instead, a line search is performed to determine the optimal step size. The step $\boldsymbol{p}$, calculated by solving Eq. (6.33), is used as the initial step in the line search procedure.

The reduction of objectives and the satisfaction of constraints often conflict, making it necessary to modify the objective function during the line search for comparison purposes. In this work, the following merit function is employed to compare the trial points and determine the progression of the algorithm:

$$\hat{\mathcal{F}}(\boldsymbol{\mu}, \pi) = \mathcal{F}(\boldsymbol{\mu}) + \pi \|\boldsymbol{h}(\boldsymbol{\mu})\|_1, \tag{6.36}$$

where π is a arbitrary penalty parameter which is kept constant throughout the optimization.

6.3.2 Quasi-Newton SQP

One problem that might arise while trying to solve a SQP problem is the necessity to have the Hessian matrix of the Lagrangian $\boldsymbol{H}_\mathcal{L}$, which might not be available or be too expensive to compute. To avoid having to deal with the exact Hessian, a quasi-Newton approach is needed to approximate it. This approximation at iteration k of the optimization procedure is denoted as $\tilde{\boldsymbol{H}}_{\mathcal{L}_k}$. The technique used to calculate the approximate Hessian in this work is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update [82, 83, 84, 85].

The version of this method used for constrained problems is the damped BFGS

update [81], where the Hessian update is computed as:

$$\tilde{\boldsymbol{H}}_{\mathcal{L},k+1} = \tilde{\boldsymbol{H}}_{\mathcal{L},k} - \frac{\tilde{\boldsymbol{H}}_{\mathcal{L},k}\boldsymbol{s}_k\boldsymbol{s}_k^\intercal\tilde{\boldsymbol{H}}_{\mathcal{L},k}}{\boldsymbol{s}_k^\intercal\tilde{\boldsymbol{H}}_{\mathcal{L},k}\boldsymbol{s}_k} + \frac{\boldsymbol{r}_k\boldsymbol{r}_k^\intercal}{\boldsymbol{r}_k^\intercal\boldsymbol{s}_k}, \tag{6.37}$$

where

$$\boldsymbol{s}_k = \boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k \quad , \quad \boldsymbol{r}_k = \theta_k\boldsymbol{y}_k + (1-\theta_k)\tilde{\boldsymbol{H}}_{\mathcal{L}_k}\boldsymbol{s}_k, \tag{6.38}$$

with

$$\boldsymbol{y}_k = \nabla_\mu\mathcal{F}(\boldsymbol{\mu}_{k+1},\boldsymbol{\lambda}_{k+1}) - \nabla_\mu\mathcal{F}(\boldsymbol{\mu}_k,\boldsymbol{\lambda}_{k+1}), \tag{6.39}$$

$$\theta_k = \begin{cases} 1 & \text{if } \boldsymbol{s}_k^\intercal\boldsymbol{y}_k \geq 0.2\boldsymbol{s}_k^\intercal\tilde{H}_{\mathcal{L}_k}\boldsymbol{s}_k, \\ \frac{0.8\boldsymbol{s}_k^\intercal\tilde{H}_{\mathcal{L}_k}\boldsymbol{s}_k}{\boldsymbol{s}_k^\intercal\tilde{H}_{\mathcal{L}_k}\boldsymbol{s}_k - \boldsymbol{s}_k^\intercal\boldsymbol{y}_k} & \text{if } \boldsymbol{s}_k^\intercal\boldsymbol{y}_k < 0.2\boldsymbol{s}_k^\intercal\tilde{\boldsymbol{H}}_{\mathcal{L}_k}\boldsymbol{s}_k. \end{cases} \tag{6.40}$$

The parameter $\boldsymbol{r}_k$ in Eq. (6.38) is needed to ensure a positive definite $\tilde{\boldsymbol{H}}_{\mathcal{L},k}$, condition that is necessary for Eq. (6.33) to have a solution. It is important to point out that the Lagrange multiplier is fixed to the latest value when approximating the curvature of the Lagrangian, since only the curvature in the space of design variables must be evaluated. Algorithm 2 details the equality constrained optimization procedure employed in this work. During the approximate Hessian update, an occasional reset to the identity matrix might be required since, as the optimization iterates in the design space, curvature information gathered far from the current point might become irrelevant or even counterproductive. In the proposed method, the number of optimization iterations before the reset is generally set between 5 and 10.

6.3.3 Shape Optimization with concurrent trimming

In shape optimization, concurrent trimming consists of adjusting a set of trim variables $\boldsymbol{\mu}^{\text{trim}}$, such as the angle of attack, throughout the optimization problem to weakly enforce target values on a set of trim functions $\boldsymbol{\mathcal{F}}^{\text{trim}}$, such as the lift coefficient C_l. This method is useful in aerodynamic applications since generally the satisfaction of target coefficients is mandatory. In addition, changes in the trim variables can also affect the main objective function, such as in the case of drag minimization with α as the trim variable. The following method takes into account this codependency while not interfering with the main optimization problem. Starting from an initial design, $\boldsymbol{\mu}_0$, the trim parameters are first optimized using multiple solver iterations. At each iteration, the trim parameters are updated according to:

$$\boldsymbol{\mu}_{k+1}^{\text{trim}} = \boldsymbol{\mu}_k^{\text{trim}} + \Delta\boldsymbol{\mu}_k^{\text{trim}}, \quad \Delta\boldsymbol{\mu}_k^{\text{trim}} = \left[\frac{d\boldsymbol{\mathcal{F}}^{\text{trim}}}{d\boldsymbol{\mu}^{\text{trim}}}\right]_k^{-1}\left(\overline{\boldsymbol{\mathcal{F}}}^{\text{trim}} - \boldsymbol{\mathcal{F}}^{\text{trim}}(\boldsymbol{\mu}_k)\right), \tag{6.41}$$

Algorithm 2 Equality constrained SQP with quasi-Newton approximation

Require: $\boldsymbol{\mu}_0$: starting point, τ_{opt}: optimality tolerance, τ_{feas}: feasibility tolerance
Ensure: $\boldsymbol{\mu}^*$: optimal point, $\mathcal{F}(\boldsymbol{\mu}^*)$: Corresponding function value

$\boldsymbol{\lambda}_0 \leftarrow 0$ ▷ Initial Lagrange multipliers
$\alpha_{\text{init}} \leftarrow 1$ ▷ For line search
Evaluate functions $(\mathcal{F}, \boldsymbol{h})$ and derivatives $(\nabla\mathcal{F}, \boldsymbol{J}_h)$
$\nabla_\mu \mathcal{F} = \nabla\mathcal{F} + \boldsymbol{J}_h^\intercal \lambda$
$k \leftarrow 0$
while $\|\nabla_x \mathcal{F}\|_\infty > \tau_{\text{opt}}$ **or** $\|\boldsymbol{h}\|_\infty > \tau_{\text{feas}}$ **do**
    if $k = 0$ **or** reset = **true then**
        $\tilde{\boldsymbol{H}}_{\mathcal{L}_k} \leftarrow \boldsymbol{I}$ ▷ Initialize to identity matrix
    else
        Update $\tilde{\boldsymbol{H}}_{\mathcal{L}_{k+1}}$ ▷ Compute damped BFGS, Eq. (6.37)
    end if
    Solve QP subproblem, Eq. (6.33), for $\boldsymbol{p}_\mu, \boldsymbol{p}_\lambda$
      minimize $\frac{1}{2}\boldsymbol{p}_\mu^\intercal \tilde{\boldsymbol{H}}_{\mathcal{L}} \boldsymbol{p}_\mu + \nabla_\mu \mathcal{F}^\intercal \boldsymbol{p}_\mu$
      subject to $\boldsymbol{J}_h \boldsymbol{p}_\mu + \boldsymbol{h} = 0$
    $\boldsymbol{\lambda}_{k+1} \leftarrow \boldsymbol{\lambda}_k + \boldsymbol{p}_\lambda$
    $\alpha \leftarrow \texttt{linesearch}(\boldsymbol{p}_\mu, \alpha_{\text{init}})$ ▷ Use merit function $\hat{\mathcal{F}}$
    $\boldsymbol{\mu}_{k+1} \leftarrow \boldsymbol{\mu}_k + \alpha \boldsymbol{p}_\mu$
    Evaluate functions $(\mathcal{F}, \boldsymbol{h})$ and derivatives $(\nabla\mathcal{F}, \boldsymbol{J}_h)$
    $\nabla_\mu \mathcal{F} = \nabla\mathcal{F} + \boldsymbol{J}_h^\intercal \lambda$
    $k \leftarrow k + 1$
end while

where $\frac{d\boldsymbol{\mathcal{F}}^{\text{trim}}}{d\boldsymbol{\mu}^{\text{trim}}}$, is computed using the discrete adjoint method. A user-specified trim tolerance, τ_{trim}, is used as the termination criterion.

Once the trimming is complete, the shape optimization begins. Each design evaluation consists of solving forward solutions for the state $\boldsymbol{Q}_k$, adjoint solutions for the objective $\mathcal{F}$, and trim functions $\boldsymbol{\mathcal{F}}^{\text{trim}}$. The adjoints computed this way are the sensitivities of all functions with respect to both the design and the trim variables. The trimmed gradient of the objective with respect to the design parameters is then computed as:

$$\left.\frac{d\mathcal{F}}{d\boldsymbol{\mu}}\right|_{\text{trim}} = \frac{d\mathcal{F}}{d\boldsymbol{\mu}} + \frac{d\mathcal{F}}{d\boldsymbol{\mu}^{\text{trim}}}\frac{d\boldsymbol{\mu}^{\text{trim}}}{d\boldsymbol{\mu}}, \quad \frac{d\boldsymbol{\mu}^{\text{trim}}}{d\boldsymbol{\mu}} = -\left[\frac{d\boldsymbol{\mathcal{F}}^{\text{trim}}}{d\boldsymbol{\mu}^{\text{trim}}}\right]^{-1}\frac{d\boldsymbol{\mathcal{F}}^{\text{trim}}}{d\boldsymbol{\mu}}, \tag{6.42}$$

that ensures that changes in the design and trim parameters, $\delta\boldsymbol{\mu}$ and $\delta\boldsymbol{\mu}^{\text{trim}}$, satisfy $\frac{d\boldsymbol{\mathcal{F}}^{\text{trim}}}{d\boldsymbol{\mu}}\delta\boldsymbol{\mu} = 0$ and $\frac{d\boldsymbol{\mathcal{F}}^{\text{trim}}}{d\boldsymbol{\mu}^{\text{trim}}}\delta\boldsymbol{\mu}^{\text{trim}} = 0$. The gradient in Eq. 6.42 is then used to compose the Lagrangian gradient $\nabla_\mu\mathcal{L}$.

These modifications incorporate in the objective function a linearized trim parameter update and an off-trim correction. Before solving the problem within the line search, the design and trim parameters are updated using the current trial step size α as:

$$\boldsymbol{\mu}(\alpha) = \boldsymbol{\mu}_k + \alpha\boldsymbol{p}_\mu, \quad \boldsymbol{\mu}^{\text{trim}}(\alpha) = \boldsymbol{\mu}_k^{\text{trim}} + \frac{d\boldsymbol{\mu}^{\text{trim}}}{d\boldsymbol{\mu}}\alpha\boldsymbol{p}_\mu, \tag{6.43}$$

where the purpose of the trim modification is to maintain linearized trim conditions. The corresponding flow solution $\boldsymbol{Q}(\alpha)$ may not strictly satisfy the trim conditions as the states and outputs are nonlinear. However, the trim modification accounts for linearizations about the zero step size state and design, resulting in a gradual convergence toward the target value throughout the whole optimization problem. The objective function is then modified using an off-trim correction:

$$\mathcal{F}(\alpha) = F(\boldsymbol{\mu}(\alpha)) + \left.\frac{d\boldsymbol{\mathcal{F}}}{d\boldsymbol{\mu}^{\text{trim}}}\right|_{\boldsymbol{Q}(\alpha),\boldsymbol{\mu}(\alpha)} \Delta\boldsymbol{\mu}^{\text{trim}}(\alpha), \tag{6.44}$$

where the trim parameter change, $\Delta\boldsymbol{\mu}^{\text{trim}}(\alpha)$, is computed using Eq. (6.41) and depends on $\boldsymbol{Q}(\alpha)$ and $\boldsymbol{\mu}(\alpha)$. The updated objective value, $\mathcal{F}(\alpha)$, is then used to compose the merit functions. Once the line search converges, the obtained step, combined with the search direction $\boldsymbol{p}_\lambda$, updates the design parameters (the Lagrange multipliers $\boldsymbol{\lambda}$ are updated prior to the search). Finally, the trim parameters are modified using both Eq. (6.43), for the linearized correction, and Eq. (6.41), for the nonlinear correction.

Table 6.1: Parameters for the aerodynamic optimization

Category	Variable	Quantity	Value
Objective	C_d	1	-
Variable	$\boldsymbol{c}$	5	-
	$\boldsymbol{t}$	4	-
	$\boldsymbol{y}_{cp}$	12	-
Constraint	A	1	0.06
Trim variable	α	1	-
Trim objective	C_l	1	0.5
Tolerance	τ_{opt}	1	10^{-5}
	τ_{feas}	1	10^{-6}
	τ_{trim}	1	10^{-5}

6.4 Drag minimization over an airfoil

This section is dedicated to the application of the described method for the aerodynamic optimization of an airfoil in both inviscid and turbulent regimes. The geometry chosen as the object of optimization is obtained from Eqs. (2.25)–(2.26). This choice lies in the fact that, thanks to this definition, both the equation parameters and the NURBS control points can be used as design variables, thus allowing the comparison between a standard parametric optimization and a CAD consistent optimization that acts directly on the geometry definition rather than on some external quantity. Specifically, the position of the control points is adjusted by displacing only the y coordinate, allowing each control point to move along a vertical line. The leading edge and trailing edge are kept fixed. For notation purposes, the subscript $(\cdot)_{ct}$ is used to refer to quantities relative to the optimization of coefficients, while the subscript $(\cdot)_{cp}$ refers to the optimization of control points.

The flow analyzed is characterized by a freestream Mach number $M_\infty = 0.80$ for both the inviscid and turbulent regimes, with a Reynolds number $Re = 10^6$ for the latter. The optimization parameters are the same for both cases and are listed in Tab 6.1. It is worth underlying that the camber-thickness coefficients, $\boldsymbol{c}$ and $\boldsymbol{t}$, and control points are not optimized at the same time but are chosen as distinct input variables in distinct computations, while in both cases α is adjusted so as to satisfy a target lift coefficient $\overline{C_l}$ through trimming. The only constraint is on the area A, with an equality constraint defined as $\frac{A-\overline{A}}{\overline{A}} = 0$, where $\overline{A}$ is the target area. A normalization of both the constraint and the design variables resulted in better scaled problems resulting in enhanced optimization convergence. Optimization is performed over a $\mathbb{P}^2$ solution approximation.

6.4.1 Transonic inviscid flow

The first aerodynamic shape optimization case presented here investigates the results for the case of an isolated airfoil in a transonic inviscid flow. Using the parameters listed on Tab. 6.1, the optimization is performed for both the camber-thickness coefficients and the control points coordinate y. Figures 6.5–6.6 show the history of the drag coefficient function C_d, the constraint violation, the lift error, and the angle of attack throughout the two optimizations. Although the optimized C_d is approximately the same, a faster convergence to that value is observed for the case with the coefficients as design variables, while the control points achieved a stronger satisfaction of the equality constraint. Moreover, it is interesting how the final angle of attack has taken a very different path for the two cases during trimming, yielding the same drag coefficient for very different values.

Figure 6.7 shows, on the left, the geometries obtained with shape optimization compared to the starting airfoil on the left, while on the right, the pressure coefficient distributions for the $\mathbb{P}^2$ solution approximation are compared. The geometry obtained with the optimization of the control points presents a less curved pressure side in the section near the trailing edge and a less blunt leading edge compared to the shape obtained by optimizing for the camber and thickness coefficients. The pressure coefficient distribution, for the optimization of the control points, shows a higher pressure differential between the suction side and the pressure side in the first section, while it becomes narrower toward the last airfoil section, while a opposite behavior is visible for the optimization of the camber and thickness coefficients. For both cases, the initial shock is almost completely dissipated, resulting in an overall reduced drag coefficient. Figure 6.8 shows the Mach number contours and the x-momentum adjoint fields for the two optimization results. Overall, for the inviscid aerodynamic optimization, both approaches yield similar results, but follow very different optimization paths. This motivates an increased interest toward the control point position optimization, since it may lead to an objective minimum not achievable with other more standard approaches.

6.4.2 Transonic turbulent flow

The second aerodynamic shape optimization case presented here investigates the results for the case of an isolated airfoil in a turbulent flow at $Re = 10^6$. Using the parameters listed on Tab. 6.1, the optimization is performed for both the camber-thickness coefficients and the control points coordinate y. Figs. 6.9–6.10 show the history of the drag coefficient function C_d, the constraint violation, the lift error, and the angle of attack throughout the two optimizations. In this case, the optimization of the control points reaches a slightly lower minimum drag coefficient than that of the camber and thickness coefficients. The vertical displacement of the control

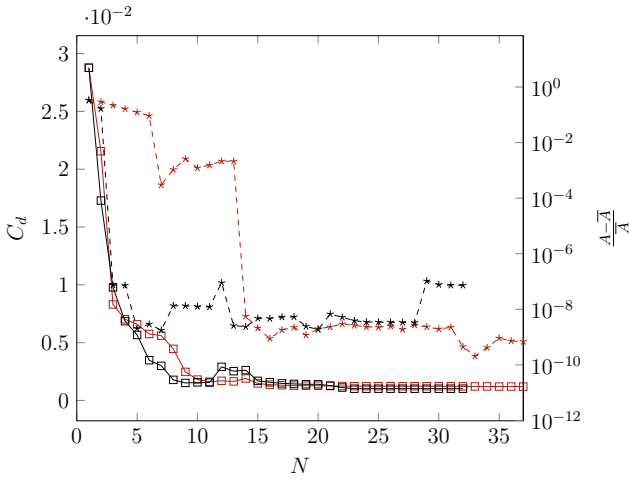


Figure 6.5: Inviscid shape optimization. History of drag coefficient C_d and constraint violation; $C_{d,cp}$, $C_{d,ct}$, $\frac{A_{cp}-\overline{A}}{\overline{A}}$, $\frac{A_{ct}-\overline{A}}{\overline{A}}$

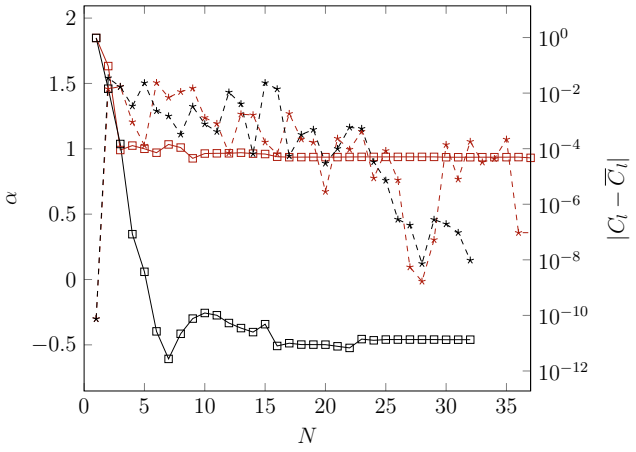


Figure 6.6: Inviscid flow shape optimization. History of angle of attack α and lift coefficient error; α_{cp}, α_{ct}, $|C_l - \overline{C}_l|_{cp}$, $|C_l - \overline{C}_l|_{ct}$

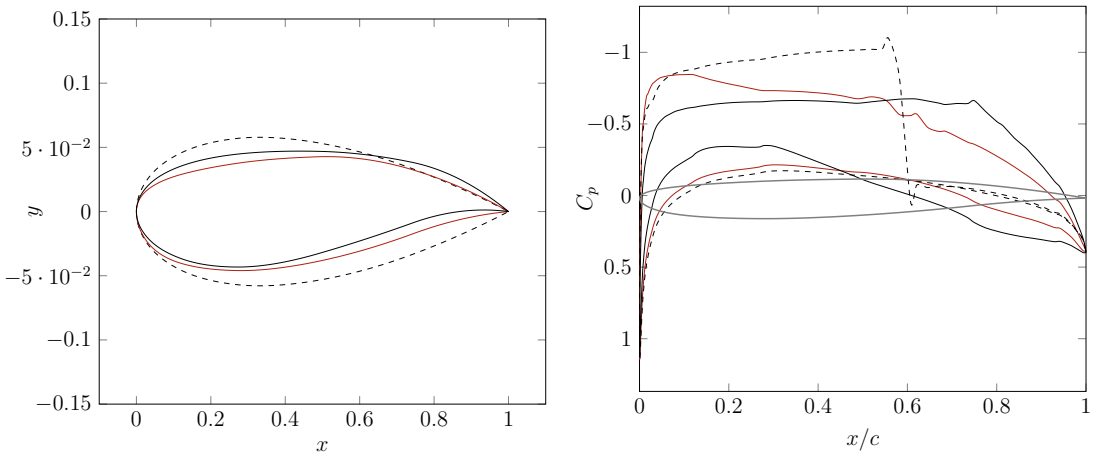


Figure 6.7: Inviscid flow shape optimization. Geometry and pressure coefficients comparison, $\mathbb{P}^2$ solution approximation; - - - initial, —— control points, —— coefficients

points allow a more efficient exploration of the design space, allowing the geometry to be morphed in a way that minimizes the drag coefficient more. Also, as for the inviscid case, the final angle of attack is different.

Figure 6.11 shows, on the left, the geometries obtained with shape optimization compared to the starting airfoil on the left, while on the right, the pressure coefficient distributions for the $\mathbb{P}^2$ solution approximation are compared. Interestingly, the shape obtained with the control point position optimization presents a rather narrower leading edge compared to the one obtained when the airfoil parameters are optimized. For both cases, the shock spreads and dissipates, resulting in a lower value in the drag coefficient. Figure 6.12 shows the Mach number contours and the x-momentum adjoint fields for the two optimization results. As in the previous case, similar results are achieved even when very different optimization paths are taken, increasing the interest toward non-conventional optimization techniques such as the displacement of control points. Acting directly on the NURBS definition in this way allows an interesting way of eplore the design space.

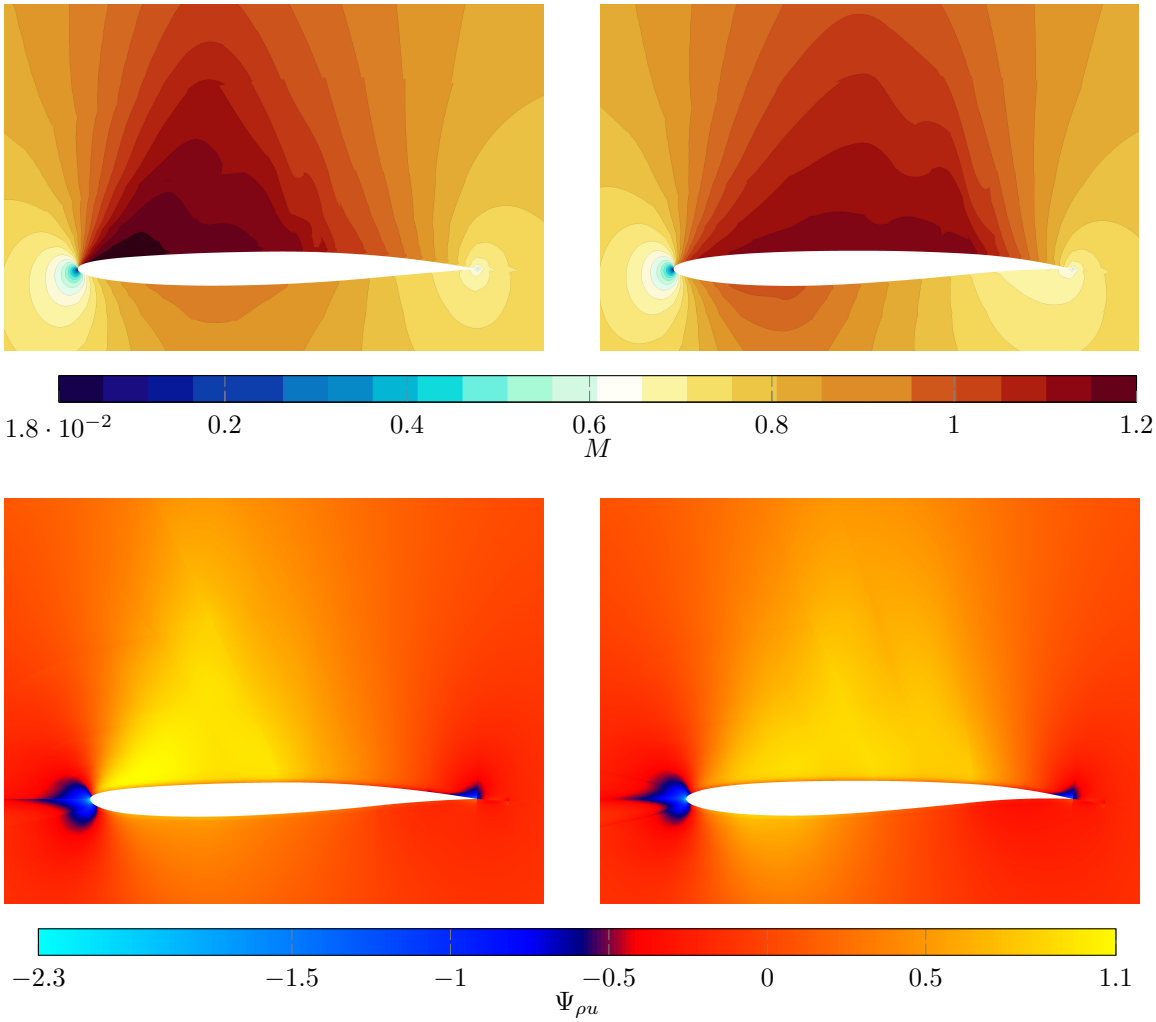


Figure 6.8: Inviscid flow shape optimization. Mach number contours (top) and drag x-momentum adjoint field (bottom) for the optimized control points (left) and optimized coefficients (right) geometries, $\mathbb{P}^2$ solution approximation

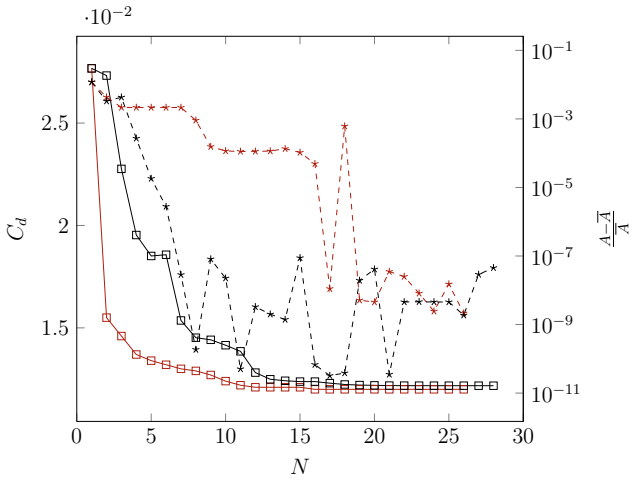


Figure 6.9: Turbulent shape optimization. History of drag coefficient C_d and constraint violation; $C_{d,cp}$, $C_{d,ct}$, $\frac{A_{cp}-\overline{A}}{\overline{A}}$, $\frac{A_{ct}-\overline{A}}{\overline{A}}$

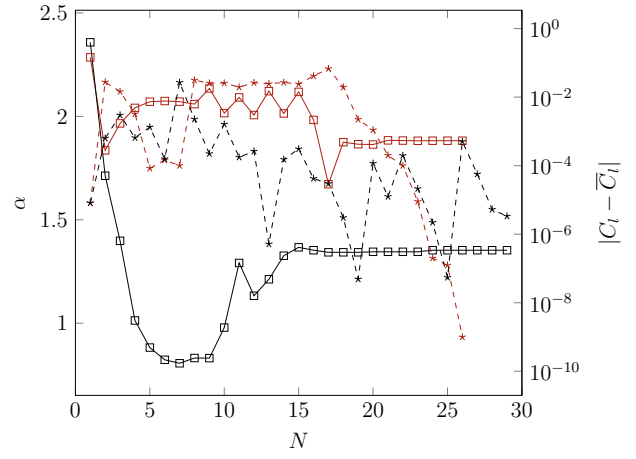


Figure 6.10: Turbulent flow shape optimization. History of angle of attack α and lift coefficient error; α_{cp}, α_{ct}, $|C_l - \overline{C}_l|_{cp}$, $|C_l - \overline{C}_l|_{ct}$

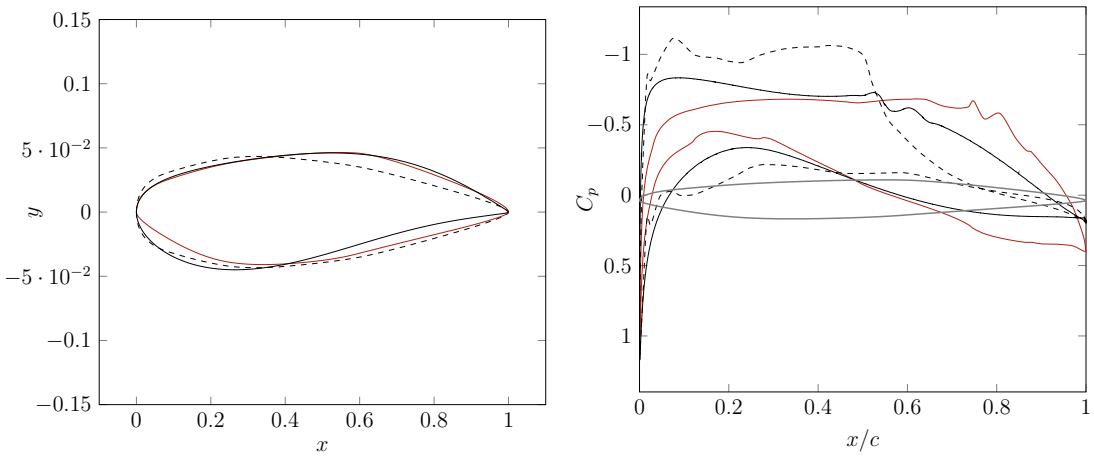


Figure 6.11: Turbulent flow shape optimization. Geometry and pressure coefficients comparison, $\mathbb{P}$ solution approximation; - - - initial, —— control points, —— coefficients

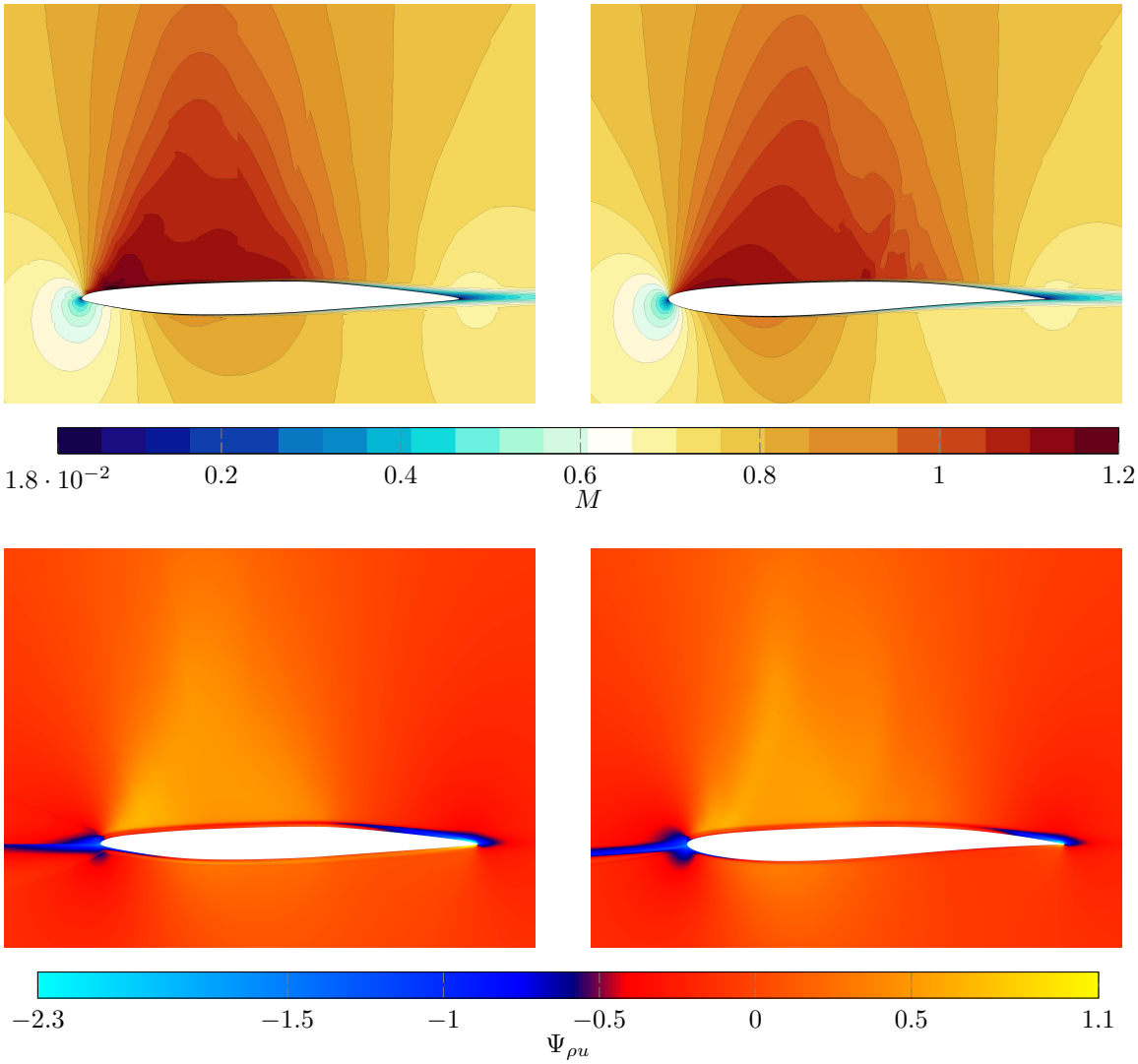


Figure 6.12: Turbulent flow shape optimization. Mach number contours (top) and drag x-momentum adjoint field (bottom) for the optimized control points (left) and optimized coefficients (right) geometries, $\mathbb{P}^2$ solution approximation

Chapter 7

Real gas simulation

This chapter is dedicated to assess the capability of the solver to simulate turbulent flows in nonideal flow conditions. The use of generalized equations of state is necessary when the flow conditions are far from being ideal, such as in turbomachinery for ORCs and carbon capture and storage. Nonideal compressible fluid dynamics (NICFD) implies a correct description of the thermophysical properties of the fluid, which is a key aspect for solver robustness. In this work, the thermodynamic models implemented are the pressure-explicit cubic equations of state (CEoS) of Peng-Robinson [86] and van der Waals [87], and the multiparameter Helmholtz energy equation of state (MEoSs) of Span-Wagner [88], implemented following the work of Mantecca et al. [36]. MEoSs typically require a greater number of coefficients compared to CEoSs, resulting in considerably higher computational costs. Nevertheless, they offer higher accuracy for thermodynamic quantities that are of key importance in fluid dynamic simulations, like the speed of sound.

7.1 Peng-Robinson and van der Waals model

The Peng–Robinson and van der Waals CEoS can be obtained from the general formulation [89]

$$p(\rho, T) = \frac{\rho R^* T}{(1 - \rho B)} - \frac{A(T)\rho^2}{(1 - \rho B + \rho C)(1 - \rho B + \rho D)} \tag{7.1}$$

where p is the fluid pressure, ρ the density, T the temperature, $R^* = R/m_M$ the mass-specific gas constant, $R = 8314.463\,\mathrm{J/(kmol \cdot K)}$ the universal gas constant, and m_M the molecular weight. The term $A(T)$ in Equation (7.1) takes into account the intermolecular attractions, while the terms B, C, and D take into account the molecular volume. The term A is usually written as $A = a\alpha^2(T)$, where the function $\alpha(T)$ (if not null) contains the dependence of $A(T)$ on the molecular shape,

Table 7.1: Parameters for the Peng–Robinson and van der Waals Equations of State

Parameter	Peng–Robinson	van der Waals
a	$0.45724\,(R^*T_{cr})^2/p_{cr}$	$0.421875\,(R^*T_{cr})^2/p_{cr}$
b	$0.0778\,R^*T_{cr}/p_{cr}$	$0.125\,R^*T_{cr}/p_{cr}$
h	$0.37464 + 1.54226\,\omega - 0.26992\,\omega^2$	0
α	$1 + h(1 - \sqrt{T/T_{cr}})$	1
A	$a\alpha^2$	a
B	b	b
C	$(2+\sqrt{2})b$	b
D	$(2-\sqrt{2})b$	b

while a is a constant. A, B, C, and D depend on some input parameters: critical pressure p_{cr} and temperature T_{cr}, molecular weight and acentric factor ω, which is an estimate of the non-sphericity of the molecules defined as $\omega = (-\log_{10}(p_r^{\text{sat}}) - 1)|_{T_r=0.7}$, where $p_r^{\text{sat}} = p_{\text{sat}}/p_{cr}$, $T_r = T/T_{cr}$, and $p^{\text{sat}}(T)$ is the saturation pressure. Table 7.1 summarizes the expressions for A, B, C, and D, which finally yield the pressure equations of the Peng-Robinson and van der Waals models:

$$p(\rho,T) = \frac{\rho R^* T}{1-\rho b} - a\rho^2, \quad p(\rho,T) = \frac{\rho R^* T}{1-\rho b} - \frac{a\alpha^2(T)\rho^2}{1+2\rho b - \rho^2 b^2}. \tag{7.2}$$

Note that when $A = B = C = D = 0$ the ideal gas law $p(\rho,T) = \rho R^* T$ is recovered. From the expressions in Equation (7.2), a complete characterization of a pure single-phase substance can be derived from the determination of at least one caloric EoS for each model [90], that is, determining an expression for the specific internal energy. A detailed procedure on how to obtain other flow quantities can be found in Appendix F.

7.2 Span-Wagner model

The Helmholtz-explicit MEsOS of Span–Wagner is developed through an optimized fitting of experimental data, applicable to any fluid with an accurate and adequate comprehensive data set [91]. This resulting EoS is expressed for the free Helmholtz energy state function, $a(\rho,T) = e(\rho,T) - Ts(\rho,T)$, presented in a dimensionless form with the sum of contributions from both an ideal gas and a real gas residual as follows:

$$\frac{a(\rho,T)}{R^*T} = \psi(\delta,\tau) = \psi^0(\delta,\tau) + \psi'(\delta,\tau), \tag{7.3}$$

where $\delta = \rho/\rho_{cr}$ is the reduced density and $\tau = T_{cr}/T$ is the inverse of the reduced temperature. In Equation (7.3), the dimensional ideal gas part is defined as

$$\begin{aligned} a^0(\rho,T) = h_0 + \int_{T_0}^{T} c_p^0(\eta)d\eta - R^*T \\ - T\left[s_0 + \int_{T_0}^{T} \frac{c_p^0(\eta)-R^*}{\eta}d\eta - R^*\log\left(\frac{\rho}{\rho_0}\right)\right], \end{aligned} \tag{7.4}$$

since for the ideal gas $p/\rho = R^*T$. Thus, when a proper approximation of $c_p^0(T)$ is given, $a^0(\rho,T)$ can be fully established by evaluating two integrals. Within this study, four distinct functional forms can be chosen to be activated, as the implementation of $c_p^0(T)$ is set up by

$$\begin{aligned} c_p^0(T) = &\sum_{i=1}^{n_{pol}} \left(c_{1,i}T^{c_{2,i}}\right) + \sum_{i=n_{pol}+1}^{n_{exp}} \left[c_{1,i}\left(\frac{c_{2,i}/T}{1-e^{-c_{2,i}/T}}\right)^2 e^{-c_{2,i}/T}\right] \\ &+ \sum_{i=n_{exp}+1}^{n_{hyc}} \left[\frac{c_{1,i}/T^2}{\cosh(c_{2,i}/T^2)^2}\right] + \sum_{i=n_{hyc}+1}^{n_{hys}} \left[\frac{c_{1,i}/T^2}{\sinh(c_{2,i}/T^2)^2}\right]. \end{aligned} \tag{7.5}$$

where each term stands for an approximation related to the statistical mechanical behavior of the heat capacity of an ideal gas, as proposed by Aly and Lee [92]. In Equation (7.5), coefficients $(c_{1,i}, c_{2,i})$ are arbitrary parameters, as many functional fittings can be found in the literature. The dimensionless residual part of Equation (7.3) is provided similarly as a summation of various activable terms. The nondimensional residual part of the Helmholtz energy is given as:

$$\begin{aligned} \psi'(\delta,\tau) = &\sum_{i=1}^{n_{pol}} c_{1,i}\delta^{c_{2,i}}\tau^{c_{3,i}} + \sum_{i=n_{pol}+1}^{n_{exp}} \left(c_{1,i}\delta^{c_{2,i}}\tau^{c_{3,i}}e^{-\delta^{c_{4,i}}}\right) \\ &+ \sum_{i=n_{exp}+1}^{n_{ga1}} \left\{c_{1,i}\delta^{c_{2,i}}\tau^{c_{3,i}}\exp\left[-c_{4,i}(\delta-c_{5,i})^2 - c_{6,i}(\tau-c_{7,i})^2\right]\right\} \\ &+ \sum_{i=n_{ga1}+1}^{n_{ga2}} \left\{c_{1,i}\Delta_i^{c_{2,i}}\exp\left[-c_{7,i}(\delta-1)^2 - c_{8,i}(\tau-1)^2\right]\right\}, \end{aligned} \tag{7.6}$$

with

$$\Delta_i = \left\{(1-\tau) + c_{3,i}\left[(\delta-1)^2\right]^{\frac{1}{(2c_{4,i})}}\right\}^2 + c_{5,i}[(\delta-1)^2]^{c_{6,i}}. \tag{7.7}$$

The last Gaussian bell-shaped sums are generally used to improve fluid description near the critical point.

The definition of Helmholtz energy allows for the calculation of all other pertinent thermodynamic properties through the application of Maxwell's relations, for instance:

$$p(\rho,T)=\rho^2\left(\frac{\partial a}{\partial \rho}\right)_T, \quad s(p,T)=-\left(\frac{\partial a}{\partial T}\right)_\rho, \quad e(\rho,T)=a(\rho,T)+Ts(\rho,T), \tag{7.8}$$

whereas the calculation of the enthalpy, specific heats, speed of sound, and fundamental derivative is the same as for the cubic models and is detailed in Appendix F, exception made for the density and the temperature.

7.3 Generalization of fluxes and boundary conditions

The implementation of the Peng-Robinson CEos requires a generalization to the algorithms that compute the convective numerical flux and to the boundary conditions. Concerning the numerical flux, the generalization of the Roe solver proposed by Vinokuer and Montagné [93] is adopted. This generalized Roe solver differs from the original version, since in the real gas regime the description of the Roe averaged state must be enriched with the definition of averaged values of the pressure derivatives $\bar{\chi}=\left(\frac{\partial p}{\partial \rho}\right)_e$ and $\bar{k}=\left(\frac{\partial p}{\partial e}\right)_\rho$ between the two sides of every mesh interface. These values are obtained here following the procedure proposed by Glaister [94] and then used to generalize the Roe averaged speed of sound for the determination of convective eigenvalues. The Harten entropy fix [95] is also employed to prevent null eigenvalues under sonic conditions.

Moreover, a generalized set of boundary conditions has to be determined, especially for inflow/outflow boundaries, which are normally based on the theory of the Riemann invariants. The extension of Riemann invariants to real gas models is quite complex, and, for this reason, the linearization proposed by Colonna and Rebay [32] is used in this work to solve the boundary problem in a consistent and generalized way. For the viscous part, the generalized multiparameter correlation of Chung et al. [96] is applied to determine transport properties in the real gas regime. In particular, the procedure allows for the estimation of reliable values of the molecular dynamic viscosity and thermal conductivity of polar and nonpolar fluids as functions of ρ and T. The additional input data required are the critical density ρ_{cr}, the dipole moment of the fluid molecules, and the equilibrium dissociation constant of the substance.

7.4 Model validation on a ORC turbine nozzle

To validate the implemented real gas model, the turbulent flow through an ORC turbine nozzle (geometry provided by Turboden S.p.A. [97]) with Siloxane MDM as the working fluid is computed. The turbine operating conditions, which also happen to be the prescribed boundary condition (except for the compressibility factor), are listed in Table 7.2. The expansion is shown in the $T - s$ diagram in Figure 7.1. In Figure 7.2, the throat area A_t and the exit area A_2 of the nozzle are illustrated. For this context, the exit-to-throat area ratio is calculated as $\frac{A_2}{A_t} = 2.04$. Since the passage is considered rectangular, this area ratio is equivalent to the length ratio. The grid is composed of 3551 $\mathbb{P}^2$ Bézier patches with an average $y^+ \approx 0.8$, obtained from 9 NURBS surfaces arranged as described in Sec. 2.4.5, and shown in Figure 7.3, while a $\mathbb{P}^3$ solution approximation is adopted. The results are compared with those obtained by Mantecca et al. [36], Colonna et al. [98] (who solved for the Euler equations) and those obtained by using the polytropic ideal-gas (PIG) law. The PIG model predicts erroneous caused by not correctly modeling the real gas behaviors. Specifically, the PIG model is given by $P = \rho RT$, where $R = \mathcal{R}/M$ is the specific gas constant, with M being the molecular weight and $\mathcal{R} = 8.314/\text{mol}\cdot\text{K}$. In the PIG model, a fixed ratio of specific heats, $\gamma = 1.01733$, is used. This value is determined from the ideal-gas specific heats at constant pressure and volume, measured at the critical temperature. This approach is chosen because this temperature is close to the turbine inlet temperature, similar to the method used by Colonna et al. [98].

Table 7.2: ORC turbine nozzle operating conditions

		Value
Pressure ratio	P_{01}/P_2, -	6
Inlet total pressure	P_{01}, bar	8
Inlet total temperature	T_{01}, °C	270.5
Inlet compressibility factor	Z_{01}, -	0.72
Outlet static pressure	P_2, bar	1.33
Outlet compressibility factor	Z_2, -	0.96

Figure 7.4 shows the contours of the Mach number and the compressibility factor, defined as $Z = p/\rho R^* T$. The absence of non-reflective boundary conditions generates reflections that disturb the downstream solution. The Mach number contours show that, from inflow to outflow, the fluid accelerates to supersonic velocities. The trailing edge generates a series of weak shocks, while further downstream the flow bends due to an overexpansion, generating a weak oblique shock wave. In addition, the contours of Z suggest that the expansion of which occurs only for a portion under nonideal fluid conditions, while downstream $Z \simeq 1$ occurs almost everywhere,

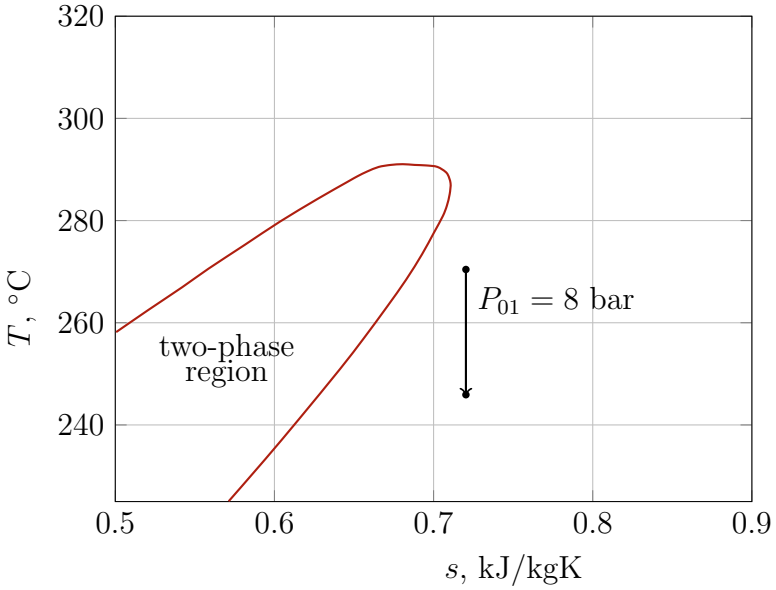


Figure 7.1: ORC nozzle expansion in the temperature-entropy plot for the MDM working fluid

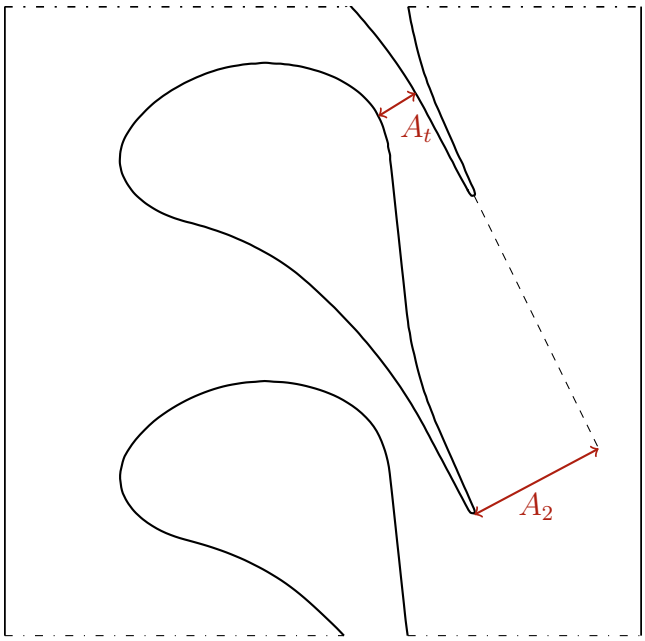


Figure 7.2: Definition of throat area A_t and exit area A_2 in the turbine nozzle passage; the geometry aspect ratio is deformed due to confidential property

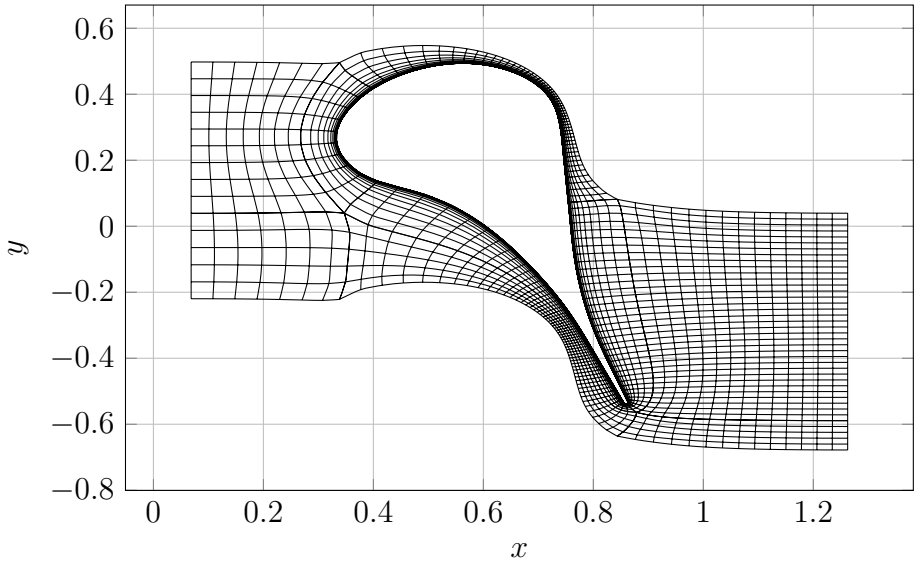


Figure 7.3: ORC turbine nozzle mesh composed of 9 $\mathbb{P}^2$ NURBS surfaces with average $y^+ \approx 4$

where $Z = 1$ represents the perfect gas conditions.

Figure 7.5 compares the pressure coefficients obtained along the blade, with the PIG law and the Peng-Robinson model, with the numerical data obtained by Colonna et al. [98] and Mantecca et al. [36], while for the τ_w only the Mantecca et al. data are available. The distributions are plotted on the $\overline{s}/\overline{s}_0$, where the curvilinear abscissa $\overline{s}$ along the suction and pressure sides is normalized with the curvilinear length $\overline{s}_0$ of the suction and pressure sides, respectively. The distribution of the pressure coefficient for the real gas model shows a behavior similar to the literature results, except for the last section of the suction side, where, at $\overline{s}/\overline{s}_0 \approx 0.67$, the oblique shock coming from the trailing edge of the upper blade in the cascade causes the separation of the boundary layer. The separation causes a pressure drop, while the plateau is ascribed to the recirculation bubble created after the shock. The flow field does not reattach before the trailing edge. This different behavior can be motivated by the different turbulence models employed. Figure 7.6 shows the density distribution along the blade surface obtained using the Peng-Robinson model and the PIG law. The density is notably high at the leading edge of the blade; however, it diminishes as the fluid expands into the region defined by a dilute ideal gas. Initially, the density calculated using the PIG model is observed to be $\approx 28\%$ lower than the one obtained with the CEoS of Peng-Robinson. However, as the fluid continues to expand towards lower pressure states - in accordance with ideal gas behavior - this density discrepancy progressively reduces, reaching only a $\approx 3\%$ difference along the rear section of the pressure side, while the values are similar along the end of

the suction side. Additional differences between the ideal gas model and the Peng-Robinson modeel can be analyzed along the pitchwise distribution in the outflow section for the flow angle, the Mach number, the total pressure loss coefficient ζ and the Spalart-Allmaras turbulence model variable $\tilde{\nu}$ (see Figs. 7.7 and 7.8. The PIG law solution tends to predict a reflected shock location slightly lower, causing a visible displacement in all the distributions at the outflow. The outflow angle is similar for both cases, except for the oscillatory region near the shock, The mach number shows higher values for the PIG law, while the loss coefficient is higher for the CEoS. Higher $\tilde{\nu}$ values are predicted for the PIG law.

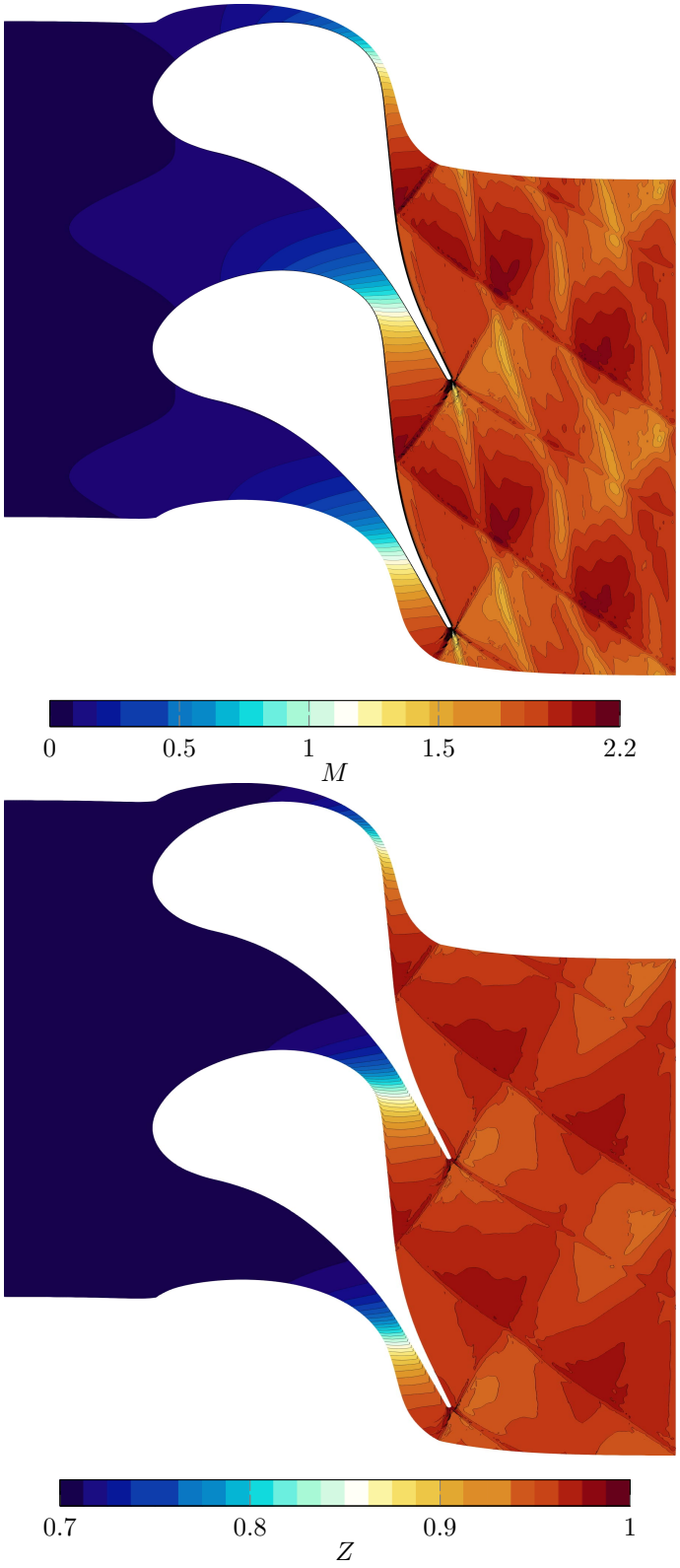


Figure 7.4: ORC nozzle. Mach number (top) and compressibility factor (bottom) contours on the adapted mesh, $\mathbb{P}^3$ solution approximation

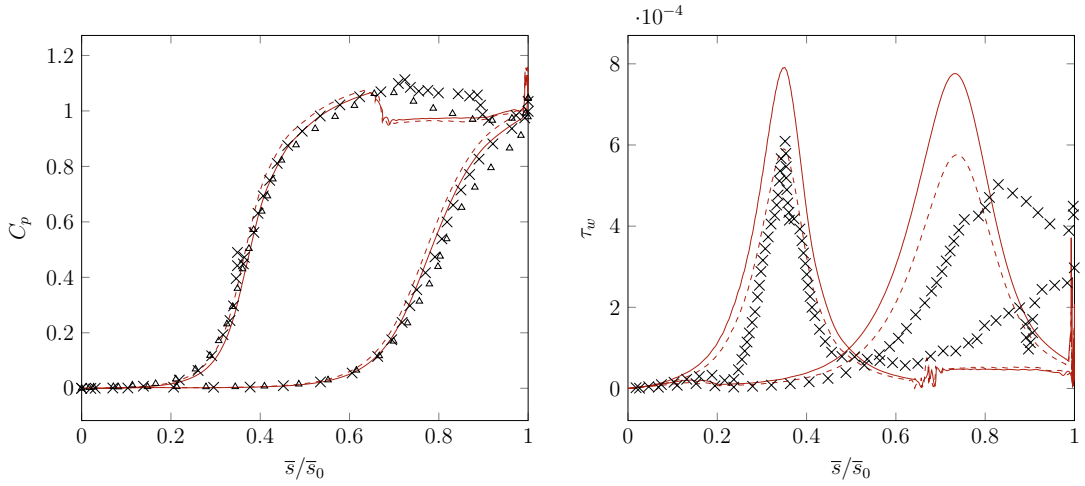


Figure 7.5: ORC Nozzle. Comparison of the pressure coefficient C_p (left) and τ_w (right) distributions on the blade surface with respect to the literature results, $\mathbb{P}^3$ solution approximation; —— NURBS-dG Peng-Robinson, - - - NUBRS-dG ideal gas, ▵ Colonna et al. [98], × Manetecca et al. [36]

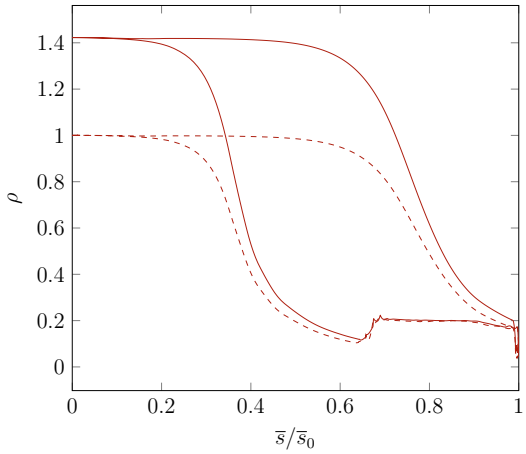


Figure 7.6: ORC Nozzle. Comparison of the density distributions on the blade surface obtained with the Peng-Robinson model and PIG law, $\mathbb{P}^3$ solution approximation; —— NURBS-dG Peng-Robinson, - - - NUBRS-dG ideal gas

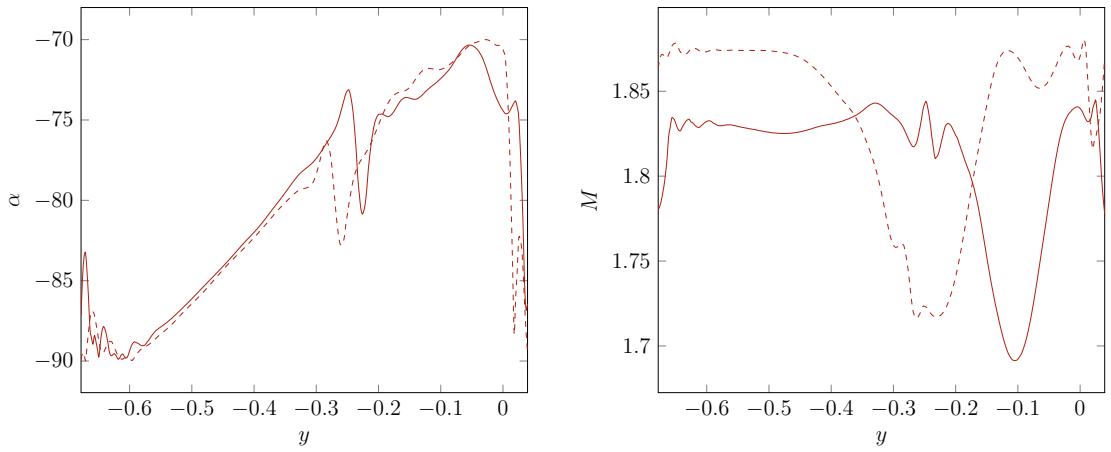


Figure 7.7: ORC Nozzle. Comparison of the flow angle α (left) and Mach number M (right) at the outflow between Peng-Robinson model and PIG law; $\mathbb{P}^3$ solution approximation; —— NURBS-dG Peng-Robinson, - - - NUBRS-dG ideal gas

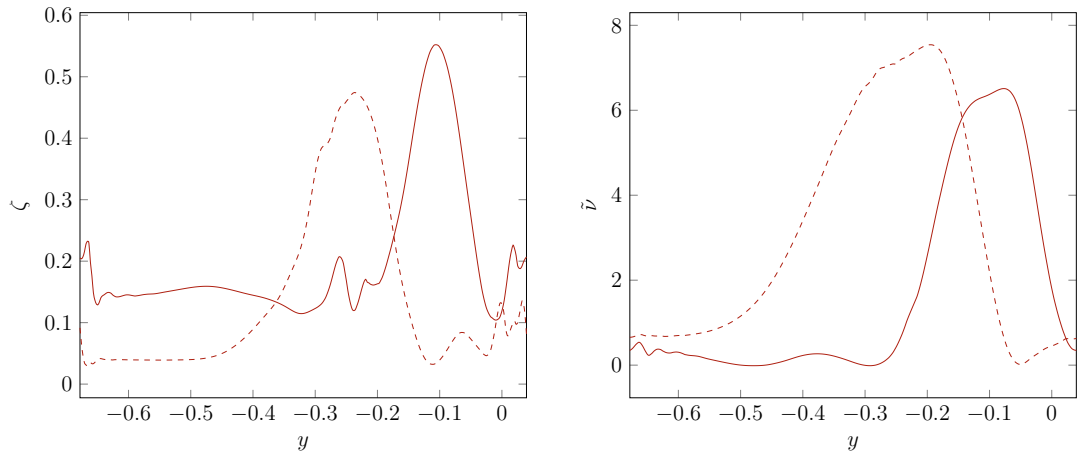


Figure 7.8: ORC Nozzle. Comparison of the loss coefficient ζ (left) and $\tilde{\nu}$ (right) at the outflow between Peng-Robinson model and PIG law; $\mathbb{P}^3$ solution approximation; —— NURBS-dG Peng-Robinson, - - - NUBRS-dG ideal gas

Chapter 8

Conclusions

8.1 Summary and conclusions

This thesis introduces a new isogeometric dG method for the RANS equations. This approach integrates high-order CAD-consistent meshes with an adaptive anisotropic mesh refinement, allowing simulations to be performed on exactly preserved CAD geometries, following the isogeometric analysis paradigma.

Tools for generating high-order NURBS meshes are developed and show a wide range of representation capabilities, from single-block isolated airfoil meshes to multi-block turbomachinery cascade meshes. The curved boundaries described by NURBS of order p allow one to achieve overall smoothness up to the derivative $p-1$, improving both accuracy and stability with increasing orders. Moreover, high-order NURBS with a low number of control points present satisfactory capability of fitting a set of points, when the original CAD geometry is not available.

A NURBS-based discontinuous Galerkin discretization is employed, where the same basis functions used for the geometry are also used for the solution approximation. The method solves the RANS equations coupled with the Spalart-Allmaras turbulence model, modified so as to prevent the appearance of spurious phenomena due to nonphysical values of the turbulent variables. Particular attention is given to spatial integration, where a study assesses the goodness of the Gauss-Legendre quadrature rules to solve area integrals of nonpolynomial functions over a highly distorted domain represented with rational basis functions, demonstrating capabilities of accuracy and stability when a sufficient number of quadrature points is used.

The solver accuracy and performance are evaluated on well-documented test cases, such as the Ringleb problem, the laminar, and turbulent flow over a flat plate, and different flow regimes over a NACA 0012 airfoil. The errors converge toward an analytical solution and the residual convergence show the expected trend, while the solution accuracy for more complex cases is assessed through a comparison

of nondimensional quantities with the experimental and literature results. A key aspect for the efficient implementation of the proposed solver is the mesh refinement strategy, which covers both isotropic and anisotropic approaches.

The h-adaptation process is based on a quad-tree structure to store and use the data needed by the refinement algorithm in an efficient manner. Different error indicators are investigated, and the spectral decay emerges as the best, as it guarantees the best accuracy in representing the flow field by minimizing the number of DoFs. Anisotropic and isotropic refinements are compared, with the former using a lower number of DoFs for the same level of accuracy. The use of anisotropic refinements is particularly advantageous when there are directional flow features, such as boundary layers, shocks, and wakes. A speed-up of $\approx 6\times$ with respect to the standard nonrefined meshes is achieved.

The proposed method is also used to perform a gradient-based constrained shape optimization, where an objective function is minimized while subjected to equality constraints, and where gradients are computed using the discrete adjoint method. The procedure is carried out with the SQP approach, where, at each step, a damped BFGS method is used to approximate the Hessian matrix. This method is coupled with a merit function-based line search technique to estimate the step size used to explore the design space in the direction obtained by solving the current QP. This approach is tested on airfoils in turbulent flows with the aim of minimizing the drag coefficient while satisfying area and lift coefficient constraints. Optimizations with respect to airfoil parameters and control points are compared, showing better and promising results when vertical displacements of the control points are chosen as design variables.

Finally, the implementation of real gas models is described and assessed on an ORC turbine nozzle, where strong nonideal gas behaviors arise. Two cubic equations of state and a multiparameter model are implemented and described to obtain a full flow characterization. The turbulent flow in an ORC turbine nozzle is simulated, obtaining satisfying results as compared to the data in the literature.

In summary, this work provides a substantial contribution in the IGA field, as the proposed isogeometric DG solver can represent a viable solution for the efficient simulation of turbulent real gas flows around complex geometries while performing shape optimization for the minimization of a desired objective.

8.2 Future Work

Many areas for future work remain. Some ideas for further research are described here to expand the applicability and efficiency of the method.

An hp-refinement strategy could further enhance the method accuracy and efficiency. hp-refinement in a NURBS-based framework is straightforward to imple-

ment, since the well-known knot insertion and degree elevation algorithms can be used. This approach would allow efficient and dynamic handling of regions that require fine detail through a dynamic refinement strategy based on error indicators to determine whether h, p, or a combination is most effective.

The extension of the refinement strategy to k-refinement, where the regularity of the NURBS basis functions is adaptively modified, represents a promising way for improving efficiency. By employing output-based error estimation techniques using the discrete adjoint method, the sensitivity of the solution error to changes in refinement could be quantified. This quantification could be used to drive the refinement to optimally modify the smoothness k of the basis functions. This method would be particularly beneficial in regions with sharp gradients or discontinuities, where controlling the smoothness could mitigate spurious oscillations.

The current solver is restricted to two-dimensional domains, which limits its application to more complex problems in engineering and physics. Extending the solver to three-dimensional domains would involve the development of tools able to generate 3D NURBS meshes, while the extension of the rational representation is helped by the tensor product nature of the basis functions.

Appendix A

Least squares curve fitting

This appendix is dedicated to the weighted and constrained least squares curve fitting method with curvatures presented by Piegl [39], adapted to NURBS from an algorithm by Smith et al. [99]. Let $\{Q_i\}, i = 0, \ldots, r$, represent the data points to be approximated. Optionally, the first derivative D_i, direction Dir_i, and curvature $Curv_i$ can be specified at any Q_i. Let $\{D_i^{(j)}, Dir_i^{(j)}, Curv_i^{(j)}\}, j = 0, \ldots, s$, be the set of derivatives, directions, and curvatures; $s \leq r$, where $s = -1$ means that the derivatives, directions, or curvatures are not specified. Furthermore, any Q_i, $D_i^{(j)}$, $Dir_i^{(j)}$, or $Curv_i^{(j)}$ can be constrained (precisely interpolated).

$\{Q_i\}$, $\{D_i^{(j)}\}$, $\{Dir_i^{(j)}\}$, and $\{Curv_i^{(j)}\}$ are divided into unconstrained and constrained sets. Denote by $Q_i^{(u)}$, $Q_i^{(c)}$, $D_i^{(u)}$, $D_i^{(c)}$, $Dir_i^{(u)}$, $Dir_i^{(c)}$, $Curv_i^{(u)}$ and $Curv_i^{(c)}$ the unconstrained and constrained items. The total number of data points is:

$$r = r_u + r_c + 1, \tag{A.1}$$

and the total number of derivatives, directions, and curvatures is:

$$B = B_u + B_c + 1. \tag{A.2}$$

The aim is to approximate the unconstrained data in the least squares sense while interpolating the constrained data exactly. The algorithm requires:

$$m_c < n \cdot \text{coordinates} \quad \text{and} \quad m_c + n \cdot \text{coordinates} < m_u + 1. \tag{A.3}$$

Now, let $\boldsymbol{S}_k, k = 0, \ldots, m_u$, represent the unconstrained data items, and $\boldsymbol{T}_k, k = 0, \ldots, m_c$, represent the constrained data items. Define the vectors/matrices:

$$\boldsymbol{S} = [\boldsymbol{S}_k], \quad \boldsymbol{T} = [\boldsymbol{T}_k], \quad \boldsymbol{W} = \text{diag}(W_k), \quad \boldsymbol{P} = [\boldsymbol{P}_k], \tag{A.4}$$

where $\boldsymbol{N} = [N_{i,p}(u_k)]$ and $\boldsymbol{M} = [M_{i,p}(u_k)]$ are the basis function matrices for the unconstrained and constrained data points, respectively, including their derivatives,

directions, and curvatures. The system of equations becomes:

$$\boldsymbol{NP} = \boldsymbol{S}, \tag{A.5}$$

and the constrained equations are:

$$\boldsymbol{MP} = \boldsymbol{T}. \tag{A.6}$$

The aim is to minimize the sum of the weighted squares of the error, subject to $\boldsymbol{MP} = \boldsymbol{T}$. The error in the unconstrained data is $\boldsymbol{S} - \boldsymbol{NP}$, thus minimizing:

$$(\boldsymbol{S}^T - \boldsymbol{P}^T\boldsymbol{N}^T)\boldsymbol{W}(\boldsymbol{S} - \boldsymbol{NP}) + \boldsymbol{A}^T(\boldsymbol{MP} - \boldsymbol{T}). \tag{A.7}$$

To minimize with respect to the unknowns $\boldsymbol{A}$ (Lagrange multipliers) and $\boldsymbol{P}$, differentiate and set to zero:

$$-2(\boldsymbol{S}^T\boldsymbol{WN} - \boldsymbol{P}^T\boldsymbol{N}^T\boldsymbol{WN}) + \boldsymbol{A}^T\boldsymbol{M} = 0, \tag{A.8}$$

$$\boldsymbol{MP} - \boldsymbol{T} = 0. \tag{A.9}$$

By transposing and factoring out constants, the system can be written as:

$$\boldsymbol{N}^T\boldsymbol{WNP} + \boldsymbol{M}^T\boldsymbol{A} = \boldsymbol{N}^T\boldsymbol{WS}, \tag{A.10}$$

$$\boldsymbol{MP} = \boldsymbol{T}. \tag{A.11}$$

In matrix form, the full system is:

$$\begin{pmatrix} \boldsymbol{N}^T\boldsymbol{WN} & \boldsymbol{M}^T \\ \boldsymbol{M} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{P} \\ \boldsymbol{A} \end{pmatrix} = \begin{pmatrix} \boldsymbol{N}^T\boldsymbol{WS} \\ \boldsymbol{T} \end{pmatrix}. \tag{A.12}$$

This system is solved for $\boldsymbol{P}$ and $\boldsymbol{A}$. To find $\boldsymbol{P}$, first solve for $\boldsymbol{A}$:

$$\boldsymbol{A} = \left[\boldsymbol{M}(\boldsymbol{N}^T\boldsymbol{WN})^{-1}\boldsymbol{M}^T\right]^{-1}\left[\boldsymbol{M}(\boldsymbol{N}^T\boldsymbol{WN})^{-1}\boldsymbol{N}^T\boldsymbol{WS} - \boldsymbol{T}\right]. \tag{A.13}$$

Then, substituting $\boldsymbol{A}$ back, we obtain $\boldsymbol{P}$, the control points:

$$\boldsymbol{P} = (\boldsymbol{N}^T\boldsymbol{WN})^{-1}\boldsymbol{N}^T\boldsymbol{WS} - (\boldsymbol{N}^T\boldsymbol{WN})^{-1}\boldsymbol{M}^T\boldsymbol{A}. \tag{A.14}$$

Figure A.1 (top) shows the result obtained by fitting a set of 30 points with a $\mathbb{P}^3$ NURBS curve of 13 control points. The only constrained points are the first and the last, whereas no directions, derivative, or curvature are imposed. Figure A.1 (bottom) shows the fit of the same data set with two additional constrained points, two constrained derivatives at the first and last points, respectively, of $(10, 0)$ and $(5, 5)$, and a constrained curvature of -1 at the first point; these numerical values are defined with respect to the unit length in space $x - y$.

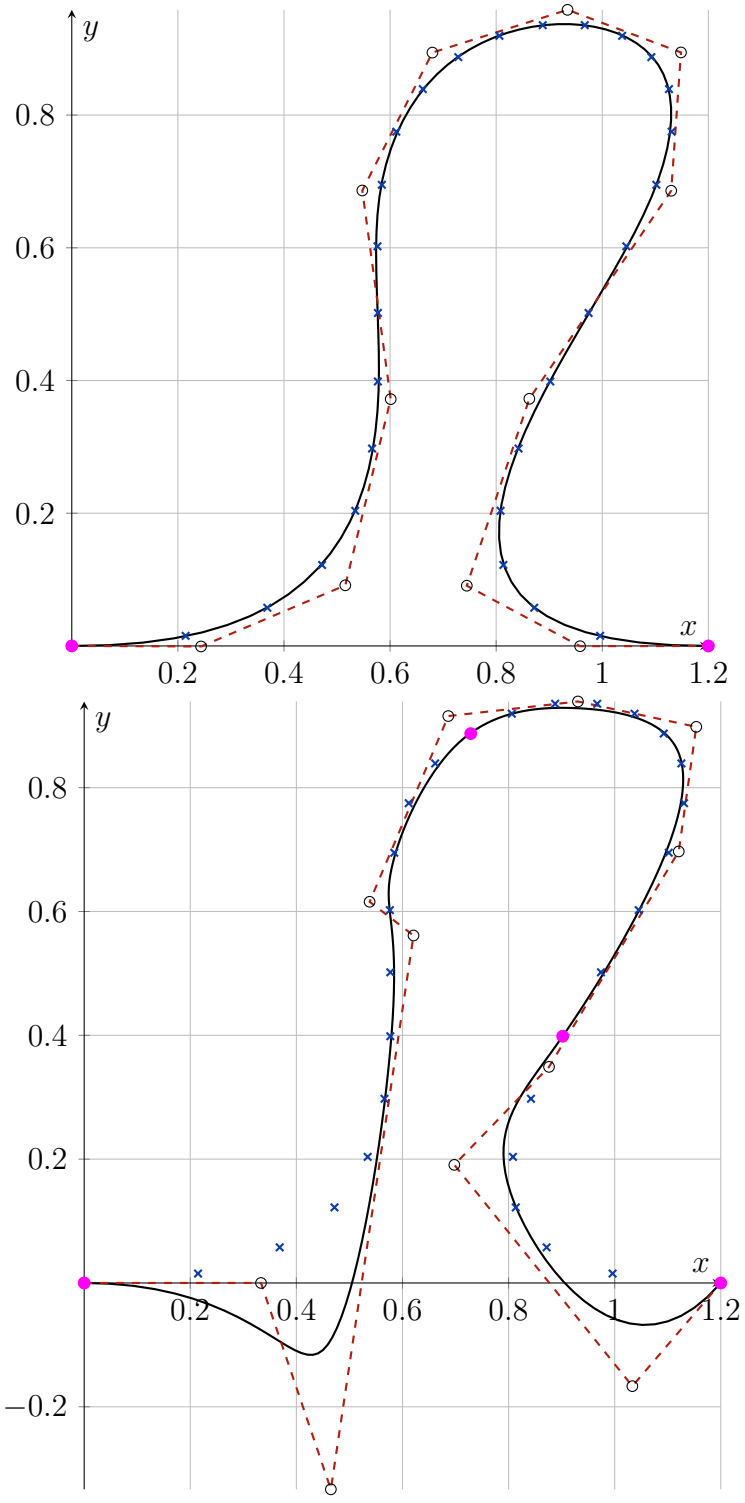


Figure A.1: Fitting of a data set with a $\mathbb{P}^3$ NURBS under different constraints; × unconstrained points, ● constrained points, ○ NURBS control points

Appendix B

Planar lofted surfaces

Given a set of B-Spline section curves, the action of blending them into a single surface is called lofting or, more precisely, skinning [39]. In this work, this technique is used to generate two-dimensional NURBS domains; therefore, only coplanar curves are taken into consideration, even if a more general discussion would expect a set of curves in the three-dimensional space.

The set of cross-sectional curves $\{\boldsymbol{C}_k(\xi)\}$, $k = 0, \ldots, K$, is planar in the ξ direction. The lofting process blends these curves in the η direction, also called the longitudinal direction, to form a smooth surface. Each curve is defined in terms of B-Spline basis functions, Eqs. (2.4) and (2.5), and its control points as:

$$\boldsymbol{C}_k(\xi) = \sum_{i=0}^{n} N_i^p(\xi)\boldsymbol{P}_{i,k}, \quad k = 0, \ldots, K. \tag{B.1}$$

These curves are defined on the same knot vector Ξ and have the same degree p. If necessary, the section curves can be brought to a common knot vector $\overline{\Xi}$ through knot insertion (see Section 2.3.1).

The lofting procedure constructs a surface by fitting between the control points of the section curves in the η direction with B-splines of degree q. It is important to note that, while setting all control points as constrained data, i.e. interpolating all of them, would be preferred for more general applications, in this work only the boundary control points are set as constrained data, while the internal control points are set as unconstrained. This approach demonstrates better results in the smoothness of the final mesh while preserving exactly the geometry of the boundaries. During this fitting process, directions and derivatives can be imposed along the boundaries; this is of particular interest when element edges normal to the body surfaces are desired.

The surface generated by the lofting procedure is expressed as:

$$\boldsymbol{S}(\xi,\eta) = \sum_{j=0}^{K} M_j^q(\eta)\boldsymbol{C}_j(\xi), \tag{B.2}$$

where $M_j^q(\eta)$ are the B-spline basis functions of degree q in the η direction, and the section curves $\boldsymbol{C}_j(\xi)$ act as the control curves along the u direction. For each control point $\boldsymbol{P}_{i,k}$ from the section curves, the lofting process approximates in the η direction to generate the control points $\boldsymbol{Q}_{i,j}$ of the surface. These control points are computed as:

$$\boldsymbol{Q}_{i,j} = \sum_{k=0}^{K} M_j^q(\eta)\boldsymbol{P}_{i,k}, \quad i = 0,\ldots,n, \quad j = 0,\ldots,K. \tag{B.3}$$

Thus, the control points $\boldsymbol{Q}_{i,j}$ define the skinned surface, ensuring a smooth blend between the section curves, as can be seen in the example of Figure B.1.

If any of the section curves $\{\boldsymbol{C}_k(\xi)\}$ is rational, the fitting in the direction η is carried out in the homogeneous space. This ensures that the rational properties of the curves are preserved throughout the lofting process. For 2D curves, control points $\boldsymbol{P}_{i,k}$ are treated as three-dimensional points $\boldsymbol{P}_{i,k} = [x_i, y_i, w_i]$, where w_i are the weights of the rational curves. After the interpolation is complete, the resulting control points $\boldsymbol{Q}_{i,j}$ are projected back into the two-dimensional space by dividing by the homogeneous weight:

$$\boldsymbol{Q}_{i,j} = \left(\frac{x_{i,j}}{w_{i,j}}, \frac{y_{i,j}}{w_{i,j}}\right). \tag{B.4}$$

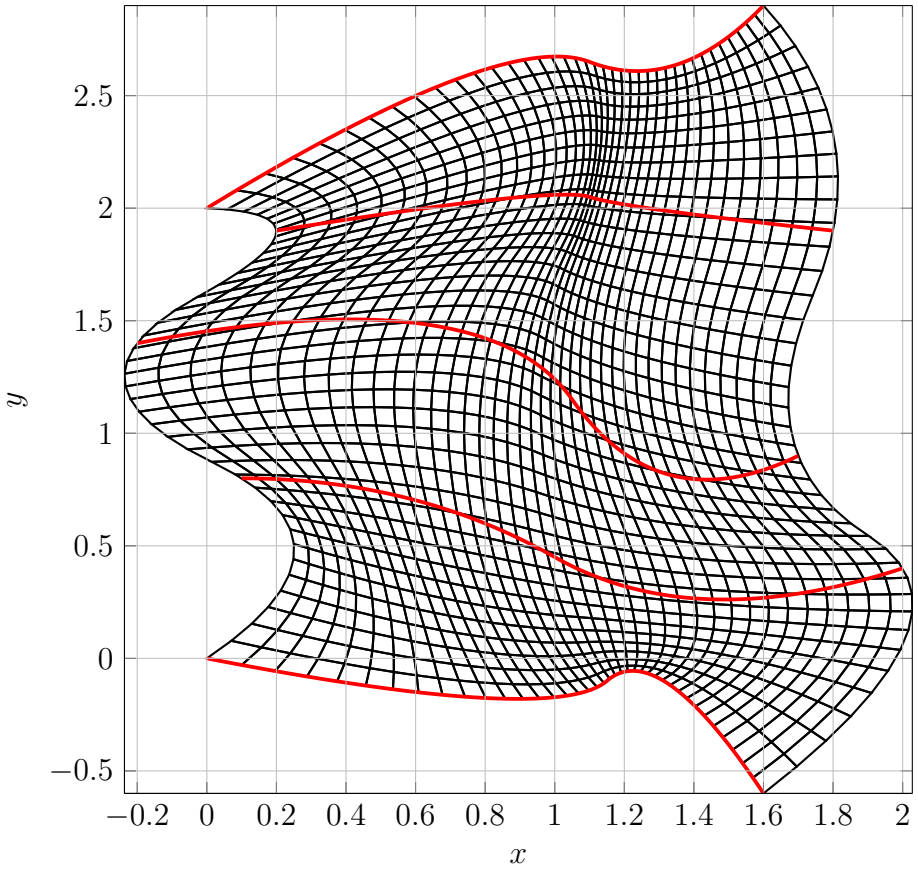


Figure B.1: $\mathbb{P}^2$ NURBS surface obtained by lofting 5 sectional curves; ◈ lofted surface, — sectional curves

Appendix C

Discrete Coons patch method

In this appendix, the discrete Coons patch method [100] is described in its application to construct NURBS surfaces given the control points of four boundary curves. The Coons patch is a bilinearly interpolated surface, and this method extends it to discrete points, specifically control points of the four boundary curves, to generate a surface that smoothly interpolates between them.

Let the four boundary curves be represented by $\boldsymbol{C}_1(\xi), \boldsymbol{C}_2(\eta), \boldsymbol{C}_3(\xi), \boldsymbol{C}_4(\eta)$, where $\boldsymbol{C}_1$ and $\boldsymbol{C}_3$ are NURBS curves in the ξ direction and $\boldsymbol{C}_2$ and $\boldsymbol{C}_4$ are NURBS curves in the η direction. The traditional Coons patch surface $\boldsymbol{S}(\xi, \eta)$ is obtained by blending the four boundaries in a bilinear way as follows:

$$\begin{aligned}\boldsymbol{S}(\xi,\eta) =&(1-\xi)\boldsymbol{C}_4(\eta) + \xi\boldsymbol{C}_2(\eta) + (1-\eta)\boldsymbol{C}_1(\xi) + \eta\boldsymbol{C}_3(\xi) \\ &- ((1-\xi)(1-\eta)\boldsymbol{P}_{11} + \xi(1-\eta)\boldsymbol{P}_{12} + (1-\xi)\eta\boldsymbol{P}_{21} + \xi\eta\boldsymbol{P}_{22}),\end{aligned} \tag{C.1}$$

where $\boldsymbol{P}_{11}, \boldsymbol{P}_{12}, \boldsymbol{P}_{21}, \boldsymbol{P}_{22}$ are the corner control points defined by the boundary curves. The terms on the right-hand side consist of interpolation along the ξ axis (first two terms), interpolation along the η axis (next two terms), and a bilinear correction term to remove the double counting of the corner points (last four terms).

The discrete Coons patch method instead involves the control points of the boundary curves. Let $\boldsymbol{C}_1$ be defined by n_1 control points $\boldsymbol{P}_1^i$, $i = 0, 1, \ldots, n_1 - 1$, $\boldsymbol{C}_2$ be defined by n_2 control points $\boldsymbol{P}_2^j$, $j = 0, 1, \ldots, n_2 - 1$, $\boldsymbol{C}_3$ be defined by n_1 control points $\boldsymbol{P}_3^i$ and $\boldsymbol{C}_4$ be defined by n_2 control points $\boldsymbol{P}_4^j$.

Let $\boldsymbol{X}[i_1, i_2]$ represent the control point of the surface at position (i_1, i_2), where $i_1 \in [0, n_1 - 1]$ and $i_2 \in [0, n_2 - 1]$. The boundary control points are set by the boundary curves as follows:

$$\boldsymbol{X}[i_1, 0] = \boldsymbol{P}_1^{i_1}, \quad \boldsymbol{X}[n_1{-}1, i_2] = \boldsymbol{P}_2^{i_2}, \quad \boldsymbol{X}[i_1, n_2{-}1] = \boldsymbol{P}_3^{i_1}, \quad \boldsymbol{X}[0, i_2] = \boldsymbol{P}_4^{i_2}. \tag{C.2}$$

For the interior points, the bilinear interpolation is used to compute the surface

control points. For each interior point $\boldsymbol{X}[i_1, i_2]$ with $1 \leq i_1 \leq n_1 - 2$ and $1 \leq i_2 \leq n_2 - 2$, the control point is given by:

$$\begin{aligned}\boldsymbol{X}[i_1, i_2] = &\left(1 - \frac{i_1}{n_1 - 1}\right) \boldsymbol{X}[0, i_2] + \frac{i_1}{n_1 - 1}\boldsymbol{X}[n_1 - 1, i_2] + \\ &\left(1 - \frac{i_2}{n_2 - 1}\right) \boldsymbol{X}[i_1, 0] + \frac{i_2}{n_2 - 1}\boldsymbol{X}[i_1, n_2 - 1] - B(i_1, i_2),\end{aligned} \tag{C.3}$$

where the bilinear correction $B(i_1, i_2)$ is computed as:

$$\begin{aligned}B(i_1, i_2) = &\left(1 - \frac{i_1}{n_1 - 1}\right)\left(1 - \frac{i_2}{n_2 - 1}\right) \boldsymbol{X}[0, 0] + \frac{i_1}{n_1 - 1}\left(1 - \frac{i_2}{n_2 - 1}\right) \boldsymbol{X}[n_1 - 1, 0] \\ &+ \left(1 - \frac{i_1}{n_1 - 1}\right) \frac{i_2}{n_2 - 1}\boldsymbol{X}[0, n_2 - 1] + \frac{i_1}{n_1 - 1}\frac{i_2}{n_2 - 1}\boldsymbol{X}[n_1 - 1, n_2 - 1].\end{aligned} \tag{C.4}$$

This procedure ensures that the control points are interpolated smoothly between the boundary curves and that the surface accurately conforms to the shape prescribed by the boundary control points. An example of a $\mathbb{P}^3$ NURBS surface obtained by applying the discrete Coons patch method is shown in Figure C.1.

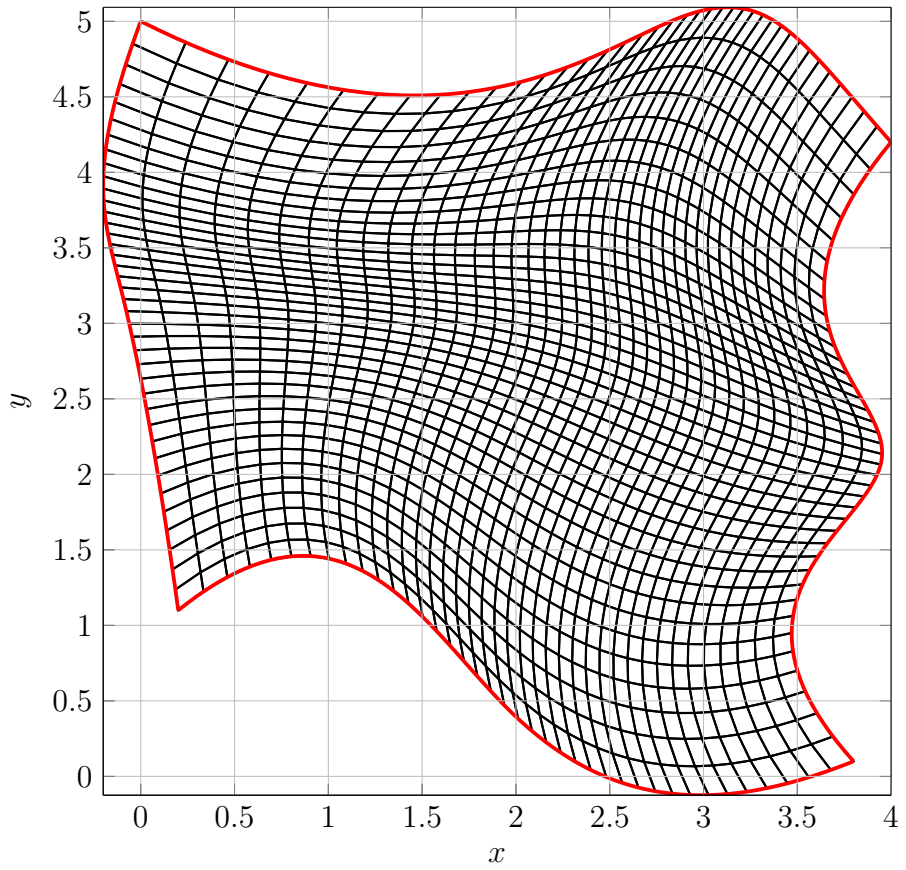


Figure C.1: $\mathbb{P}^3$ NURBS surface obtained by applying the discrete Coons patch approach; NURBS surface, boundary curves

Appendix D

Generalized Roe approximate flux

In this work, the Glaister generalized Roe numerical flux [94] is employed to handle the ideal and the real gas regimes. This appendix is dedicated to a description of this approach. The Roe-averaged state with the following relations:

$$\tilde{\rho} = \sqrt{\rho^- \cdot \rho^+}, \tag{D.1}$$

$$\tilde{e} = \frac{\sqrt{\rho^-} \cdot e^- + \sqrt{\rho^+} \cdot e^+}{\sqrt{\rho^-} + \sqrt{\rho^+}}, \tag{D.2}$$

$$\tilde{u} = \frac{\sqrt{\rho^-} \cdot u^- + \sqrt{\rho^+} \cdot u^+}{\sqrt{\rho^-} + \sqrt{\rho^+}}, \tag{D.3}$$

$$\tilde{v} = \frac{\sqrt{\rho^-} \cdot v^- + \sqrt{\rho^+} \cdot v^+}{\sqrt{\rho^-} + \sqrt{\rho^+}}, \tag{D.4}$$

$$\tilde{\nu} = \frac{\sqrt{\rho^-} \cdot \hat{\nu}^- + \sqrt{\rho^+} \cdot \hat{\nu}^+}{\sqrt{\rho^-} + \sqrt{\rho^+}}, \tag{D.5}$$

where the superscripts $(\cdot)^-$ and $(\cdot)^+$ represent the left and right sides, respectively. In this appendix, the notation $\tilde{\cdot}$ refers to the Roe averaged value of the quantity $(\cdot)$, while the Spalart-Allmaras turbulent variable is here referred to as $\hat{\nu}$, to distinguish it from its Roe averaged value $\tilde{\nu}$. The Roe averaged total enthalpy $\tilde{h}$ is given by:

$$\tilde{h} = \frac{\sqrt{\rho^-} \cdot h_0^- + \sqrt{\rho^+} \cdot h_0^+}{\sqrt{\rho^-} + \sqrt{\rho^+}}, \tag{D.6}$$

where h_t^- and h_t^+ are the total enthalpies in the left and right states, calculated as

$$h_0^- = e^- + \frac{p^-}{\rho^-} + \frac{(u^-)^2 + (v^-)^2}{2}, \tag{D.7}$$

$$h_0^+ = e^+ + \frac{p^+}{\rho^+} + \frac{(u^+)^2 + (v^+)^2}{2}. \tag{D.8}$$

The Roe-averaged speed of sound $\tilde{c}$ is calculated by using Glaister's method:

$$\tilde{c} = \sqrt{\left(\frac{\partial p}{\partial \rho}\right)_{\tilde{e}} + \left(\frac{\partial p}{\partial e}\right)_{\tilde{\rho}} \cdot \frac{\tilde{p}}{\tilde{\rho}^2}} \tag{D.9}$$

where $\left(\frac{\partial p}{\partial \rho}\right)_{\tilde{e}}$ and $\left(\frac{\partial p}{\partial e}\right)_{\tilde{\rho}}$ are the partial derivatives evaluated at the Roe-averaged state. These derivatives are calculated using the Peng-Robinson CEoS [86].

Using the Roe-averaged values computed above, the eigenvalues of the flux Jacobian matrix, representing the wave speeds, are determined as follows:

$$\lambda_1 = \|\tilde{\boldsymbol{u}}\|_n - \tilde{c}, \quad \lambda_2 = \|\tilde{\boldsymbol{u}}\|_n, \quad \lambda_3 = \tilde{q}_n + \tilde{c} \tag{D.10}$$

where $\|\tilde{\boldsymbol{u}}\|_n = \tilde{u}\, n_x + \tilde{v}\, n_y$ is the Roe-averaged normal velocity, with $\boldsymbol{n} = [n_x, n_y]^\intercal$ being the interface normal.

To prevent numerical instabilities near discontinuities, Harten's entropy fix is applied [95]:

$$\lambda_i = \begin{cases} \frac{\lambda_i^2 + \delta^2}{2\delta}, & \text{if } |\lambda_i| < \delta \\ \lambda_i, & \text{otherwise} \end{cases} \tag{D.11}$$

where $\delta = 0.1\,\tilde{c}$ is a small threshold value. These first and last eigenvalues must be modified to account for real gas effects in the following way:

$$\lambda^- = \frac{1}{2}\left(|\lambda_3| - |\lambda_1|\right) \tag{D.12}$$

$$\lambda^+ = -|\lambda_2| + \frac{1}{2}\left(|\lambda_3| + |\lambda_1|\right). \tag{D.13}$$

Then, the following auxiliary quantities d_1 and d_2 are introduced:

$$d_1 = \frac{\lambda^+ \Delta p}{\tilde{c}^2} + \frac{\lambda^- \Delta\rho\, \Delta q_n}{\tilde{c}} \tag{D.14}$$

$$d_2 = \frac{\lambda^- \Delta p}{\tilde{c}} + \lambda^+ \Delta\rho\, \Delta q_n \tag{D.15}$$

where $\Delta p = p^+ - p^-$ is the pressure difference, $\Delta\rho = \rho^+ - \rho^-$ is the density difference, and $\Delta\|\boldsymbol{u}\|_n = \|\boldsymbol{u}\|_n^+ - \|\boldsymbol{u}\|_n^-$ is the difference in normal velocities between the left and right states defined as $\|\boldsymbol{u}\|_n = un_x + vn_y$. The difference in convective fluxes between the left and right states is defined as $\Delta\boldsymbol{F}_c$, while an average quantity vector $\tilde{\boldsymbol{c}}_a$ and the vector $\boldsymbol{F}_n$ are introduced as:

$$\tilde{\boldsymbol{c}}_a = \begin{bmatrix} 1 \\ \tilde{h} \\ \tilde{u} \\ \tilde{v} \\ \tilde{\nu} \end{bmatrix}, \ \boldsymbol{F}_n = \begin{bmatrix} 0 \\ \tilde{q}_n \\ n_x \\ n_y \\ 0 \end{bmatrix}, \tag{D.16}$$

The flux $\hat{\boldsymbol{F}}$ is finally calculated as:

$$\digamma = |\lambda_2| \Delta \boldsymbol{F}_c + d_1 \tilde{\boldsymbol{c}}_a + d_2 \boldsymbol{F}_n, \tag{D.17}$$

$$\hat{\boldsymbol{F}} = \frac{\boldsymbol{F}_c^- + \boldsymbol{F}_c^+ - \digamma}{2}. \tag{D.18}$$

Appendix E

Boundary conditions

This appendix details the specific boundary conditions used in this work. Convective and viscous fluxes, which depend on both the boundary state and its gradient, are used to weakly enforce the prescribed quantities. Here, a detailed description of the calculation of the boundary state for the boundary conditions used in this work is given. These boundary conditions are described for two-dimensional domains, therefore $m = 4$.

E.1 Subsonic inflow

At the inflow boundary, with subsonic flow $M_n = \|\boldsymbol{u}\|_n / c < 1$, there is only one positive eigenvalue $\|\boldsymbol{u}\|_n + c$. This means that $m - 1$ quantities are prescribed, including static pressure p^{bc}, density ρ^{bc}, and flow direction. The subscript $(\cdot)_n$ represents the quantity $(\cdot)$ in the normal direction, so $\|\boldsymbol{u}\|_n = un_x + vn_y$, where $\boldsymbol{n} = [n_x, n_y]^\intercal$ is the outward normal boundary. The missing information to complete the boundary state is the Riemann invariant associated with $\|\boldsymbol{u}\|_n + c$ evaluated from inside the domain. For turbulent simulations, the value of $\tilde{\nu}$ is also prescribed as explained in Section 3.1.2. From the EoS used, the total specific enthalpy h_0 and the specific entropy s can be computed. The equations for the definition of $\boldsymbol{v}^{bc}$ are:

$$\delta h_0 = \delta h_0^{bc}, \tag{E.1}$$

$$\delta s = \delta s^{bc}, \tag{E.2}$$

$$\delta p + \rho c \, \delta \|\boldsymbol{u}\|_n = 0, \tag{E.3}$$

here the superscript $(\cdot)^{bc}$ indicates the prescribed boundary condition, c is the speed of sound. The direction of the boundary velocity $\boldsymbol{u}^{bc}$ is prescribed, while its magnitude $\|\boldsymbol{u}\|^{bc}$ is to be determined. If $\boldsymbol{\beta}$ is the unit normal vector along $\boldsymbol{u}^{bc}$, then

$\boldsymbol{u}^{bc} = \boldsymbol{\beta}\|\boldsymbol{u}\|^{bc}$, yielding:

$$\delta\|\boldsymbol{u}\|_n = \beta_n\,\delta\|\boldsymbol{u}\| - \delta\xi, \quad \delta\xi = \|\boldsymbol{u}\|_n - \beta_n\,\|\boldsymbol{u}\| \tag{E.4}$$

where $\beta_n = \boldsymbol{\beta}\cdot\boldsymbol{n}$. This leads to the modified equation for δP:

$$\delta p + \rho c\,\beta_n\,\delta\|\boldsymbol{u}\| = \rho c\,\delta\xi \tag{E.5}$$

The missing relation is provided by the Gibbs-Duhem equation with no phase changing:

$$\delta h = T_0\delta s + \frac{\delta p}{\rho} \tag{E.6}$$

This allows expressing δp as a function of δh_0 and δs:

$$\delta h_0 = T_0\delta s + \frac{\delta p}{\rho} + \frac{1}{2}\delta\|\boldsymbol{u}\|^2 \tag{E.7}$$

Solving for $\delta\|\boldsymbol{u}\|$:

$$(\delta\|\boldsymbol{u}\|)^{\pm} = (c\beta_n - \|\boldsymbol{u}\|) \pm \sqrt{(c\beta_n - \|\boldsymbol{u}\|)^2 - 2(c\delta\xi + T_0\delta s - \delta h_0^{bc})} \tag{E.8}$$

For a subsonic inflow boundary where $\beta_n < 0$ and $c\beta_n - \|\boldsymbol{u}\| < 0$, the solution $(\delta q)^+$ is selected to obtain $\delta\|\boldsymbol{u}\| = 0$, completing the boundary state evaluation. For the viscous part, the gradient employed in the calculation of the numerical flux is extracted from inside the domain, that is $\hat{\boldsymbol{F}}_v = \hat{\boldsymbol{F}}_v\left(\boldsymbol{q}, \nabla\boldsymbol{q}, \boldsymbol{q}^{bc}, \nabla\boldsymbol{q}; \boldsymbol{n}\right)$. This applies for all the inflow and outflow boundary conditions.

E.2 Supersonic inflow

The calculation of the boundary state for an inflow boundary that is supersonic in the normal direction is trivial since all the eigenvalues are negative and m the boundary data must be prescribed, therefore the entire vector $\boldsymbol{q}^{bc}$ is the prescribed data in this case.

E.3 Subsonic outflow

For an outflow boundary, defined as a subset of $\partial\Omega$ such that $\boldsymbol{u}\cdot\boldsymbol{n} > 0$. If the flow in the normal direction is subsonic, there is only one eigenvalue $\lambda = \|\boldsymbol{u}\|_n - c < 0$. The equations that allow for the computation of the boundary state are

$$\delta p + \rho c\,\delta\|\boldsymbol{u}\|_n = 0 \tag{E.9}$$

$$c^2\delta\rho + \delta p = 0 \tag{E.10}$$

$$\delta p = \delta p^{bc} \tag{E.11}$$

The solution of the system in this case is trivial:

$$\delta\rho = -\frac{\delta p^{bc}}{c^2} \tag{E.12}$$

$$\delta\|\boldsymbol{u}\|_n = \frac{\delta p^{bc}}{\rho c^2}\boldsymbol{n} \tag{E.13}$$

$$\delta p = \delta p^{bc} \tag{E.14}$$

E.4 Supersonic outflow

In the supersonic case, since all the eigenvalues are positive, no condition needs to be imposed. The boundary state is simply equal to the internal state, that is, $\boldsymbol{q}^{bc} = \boldsymbol{q}$.

E.5 No-slip adiabatic wall

At a no-slip, adiabatic wall, the desired conditions are zero velocity, heat flux, and eddy viscosity at the wall. Using this information, the boundary state is given by:

$$\boldsymbol{q}^{bc} = \left[\rho, \rho E - \frac{1}{2}\rho\|\boldsymbol{u}\|^2, 0, 0, 0\right]^\intercal \tag{E.15}$$

The inviscid flux is computed using the nonnumerical flux function $\boldsymbol{F}(\boldsymbol{q}^{bc})$. In addition, an adiabatic condition requires that the viscous flux in the energy equation satisfies $\nabla E \cdot \boldsymbol{n} = 0$. For this reason, the viscous flux is computed using the interior gradient, except for the energy-equation component, which is set to:

$$f_u = u - \|\boldsymbol{u}\|_n \cdot n_x \tag{E.16}$$

$$f_v = v - \|\boldsymbol{u}\|_n \cdot n_y \tag{E.17}$$

$$f_e = e - \frac{1}{2}\left(u^2 + v^2 + f_u^2 + f_v^2\right) \tag{E.18}$$

$$\nabla(\rho E)^{bc} = f_e \cdot \nabla\rho + f_u \cdot \nabla(\rho u) + f_v \cdot \nabla(\rho v). \tag{E.19}$$

E.6 Symmetry plane

The boundary condition applied to the plane of symmetry requires that the state exhibit symmetry with respect to the plane and ensures continuity in its derivatives, which are perpendicular to this plane. This necessitates that the flow remain tangential to the symmetry plane. Consequently, the boundary state is adjusted to match the interior state with the subtraction of the normal component of velocity:

$$\rho^{bc} = \rho \tag{E.20}$$

$$(\rho E)^{bc} = \rho E \tag{E.21}$$

$$(\rho u)^{bc} = \rho \left(u - 2\|\boldsymbol{u}\|_n n_x\right) \tag{E.22}$$

$$(\rho v)^{bc} = \rho \left(v - 2\|\boldsymbol{u}\|_n n_y\right) \tag{E.23}$$

$$\tilde{\nu}^{bc} = \tilde{\nu} \tag{E.24}$$

Regarding the state derivatives, the presence of the symmetry plane requires that the normal derivatives of the scalar quantities ρ, ρE, and $\tilde{\nu}$ are zero. Similarly, the normal derivative of the tangential velocity is also zero. However, this condition does not provide any data about the normal derivative of the normal velocity or the tangential derivative of any variable. The boundary state gradient is determined by integrating the information available on the normal derivatives with the derivatives from within the interior. Be $\boldsymbol{R}$ the reflection matrix with respect to the boundary normal $\boldsymbol{n} = [n_x, n_y]^\intercal$ defined as:

$$\boldsymbol{R} = \boldsymbol{I} - 2\boldsymbol{n}\boldsymbol{n}^T = \begin{bmatrix} 1 - 2n_x^2 & -2n_x n_y \\ -2n_x n_y & 1 - 2n_y^2 \end{bmatrix}, \tag{E.25}$$

where $\boldsymbol{I}$ is the identity matrix. The reflected gradients to be imposed are obtained by applying the reflection matrix $\boldsymbol{R}$ to each gradient vector:

$$\nabla\rho^{bc} = \boldsymbol{R}\nabla\rho = \begin{bmatrix} 1 - 2n_x^2 & -2n_x n_y \\ -2n_x n_y & 1 - 2n_y^2 \end{bmatrix} \begin{bmatrix} \rho_x \\ \rho_y \end{bmatrix}, \tag{E.26}$$

$$\nabla(\rho E)^{bc} = \boldsymbol{R}\nabla(\rho E) = \begin{bmatrix} 1 - 2n_x^2 & -2n_x n_y \\ -2n_x n_y & 1 - 2n_y^2 \end{bmatrix} \begin{bmatrix} (\rho E)_x \\ (\rho E)_y \end{bmatrix}, \tag{E.27}$$

$$\nabla\tilde{\nu}^{bc} = \boldsymbol{R}\nabla\tilde{\nu} = \begin{bmatrix} 1 - 2n_x^2 & -2n_x n_y \\ -2n_x n_y & 1 - 2n_y^2 \end{bmatrix} \begin{bmatrix} \tilde{\nu}_x \\ \tilde{\nu}_y \end{bmatrix}. \tag{E.28}$$

For the momentum component, the reflection matrix is updated as:

$$\boldsymbol{R'} = \boldsymbol{R} - 2\boldsymbol{n}\boldsymbol{n}^T = \begin{bmatrix} 1 - 4n_x^2 & -4n_x n_y \\ -4n_x n_y & 1 - 4n_y^2 \end{bmatrix}, \tag{E.29}$$

which is then used to define the momentum boundary gradients:

$$\nabla(\rho u)^{bc} = \boldsymbol{R'}\nabla(\rho u) = \begin{bmatrix} 1-4n_x^2 & -4n_x n_y \\ -4n_x n_y & 1-4n_y^2 \end{bmatrix} \begin{bmatrix} (\rho u)_x \\ (\rho u)_y \end{bmatrix}, \tag{E.30}$$

$$\nabla(\rho v)^{bc} = \boldsymbol{R'}\nabla(\rho v) = \begin{bmatrix} 1-4n_x^2 & -4n_x n_y \\ -4n_x n_y & 1-4n_y^2 \end{bmatrix} \begin{bmatrix} (\rho v)_x \\ (\rho v)_y \end{bmatrix}. \tag{E.31}$$

Appendix F

Generalized EoS flow characterization

Following the expressions in Equation (7.2), the mass-specific internal energy can be defined as:

$$e(\rho, T) = e_0 + \int_{T_0}^{T} c_v^0(\eta)\, d\eta + \int_0^{\rho} \frac{1}{\xi^2} \left[p - T \left(\frac{\partial p}{\partial T} \right)_{\xi} \right] d\xi, \tag{F.1}$$

while the mass-specific entropy can be written in the form:

$$s(\rho, T) = s_0 + \int_{T_0}^{T} \frac{c_v^0(\eta)}{\eta}\, d\eta - R^* \log\left(\frac{\rho}{\rho_0}\right) + \int_0^{\rho} \frac{1}{\xi^2} \left[R^* - \left(\frac{\partial p}{\partial T} \right)_{\xi} \right] d\xi, \tag{F.2}$$

where ξ and η are used as symbolic substitutes of ρ and T in the integral functions, while (ρ_0, T_0) identifies the chosen reference state. The last terms in both Eqs. (F.1) and (F.2) represent departure functions from the nonpolytropic ideal gas behavior since they vanish for sufficiently rarefied thermodynamic states, i.e. $\rho \to 0$. The remaining two integrals instead require an expression for the contribution of the ideal gas to isochoric specific heat $c_0^v(T)$, which is by definition the limit of $c_v(\rho, T)$ as $\rho \to 0$. In this work, a polynomial function of absolute temperature in the form $c_0^v(T) = c_p^0(T) - R^* = c_0 + c_1 T + c_2 T^2 + c_3 T^3 - R^*$ is used for each fluid considered, where $c_p^0(T)$ is the contribution of the ideal gas to the specific isobaric heat. The coefficients c_i for $i = 0, \ldots, 3$ can be theoretically determined from chemical group contribution methods [101], or from given polynomial fittings of available experimental data.

The values of $p(\rho, T)$ and $e(\rho, T)$ are sufficient to determine all other relevant thermodynamic properties. For example, the mass-specific enthalpy and real gas

isochoric specific heat are obtained as

$$h(\rho,T)=e(\rho,T)+\frac{p}{\rho},\quad c_v(\rho,T)=\left(\frac{\partial e}{\partial T}\right)_\rho. \tag{F.3}$$

The real gas isobaric specific heat and the speed of sound are obtained from

$$c_p(\rho,T)=c_v(T)+\frac{T}{\rho^2}\frac{\left(\frac{\partial p}{\partial T}\right)_\rho^2}{\left(\frac{\partial p}{\partial \rho}\right)_T},\quad c_p(\rho,T)=\sqrt{\frac{c_p(\rho,T)}{c_v(\rho,T)}\left(\frac{\partial p}{\partial \rho}\right)_T}. \tag{F.4}$$

The fundamental derivative of gas dynamics can be expressed following the work of Cramer [102]:

$$\Gamma(\rho,T)=\frac{1}{2\rho^3c^2}\left\{\rho^4\frac{\partial^2 p}{\partial\rho^2}\bigg|_T+2\rho^3\left(\frac{\partial p}{\partial\rho}\right)\bigg|_T+\frac{3\rho^2T}{c_v}\left(\frac{\partial p}{\partial T}\right)_\rho\left(\frac{\partial^2 p}{\partial\rho\partial T}\right)\right.$$
$$\left.+\left[\frac{T}{c_v}\left(\frac{\partial p}{\partial T}\right)_\rho\right]^2\left[3\left(\frac{\partial^2 p}{\partial T^2}\right)_\rho+\frac{1}{T}\left(\frac{\partial p}{\partial T}\right)_\rho\left(1-\frac{T}{c_v}\left(\frac{\partial c_v}{\partial T}\right)_\rho\right)\right]\right\}. \tag{F.5}$$

The density is obtained from the pressure equation. When the models are based on Equation (7.2), the thermal EoS can be reformulated as a third-degree polynomial in the density, whose coefficients are a function of temperature and pressure, that is, $d_0+d_1\rho+d_2\rho^2+d_3\rho^3=0$, with $d_i=d_i(p,T)$ for $i=0,\ldots,3$. For temperature, Newton's iterations are used in functions $p(\rho,T)$ and $e(\rho,T)$, since their derivatives are known and the resulting formulation is more complicated. Initial guesses are calculated using the ideal polytropic gas model with $\gamma=c_p^0(T_0)/c_v^0(T_0)$.

When the model is instead based on Equation (7.3), Newton's iterations are also used for density, but since the number of roots may be higher than in the CEoS case, some initial guesses are chosen as suggested by Span [91]. Specifically, the initial estimate for the density is provided by the Peng–Robinson model, whose coefficients are calculated and stored once. For the temperature, a simplified version of the van der Waals model with a power law ideal gas-specific heat is analytically inverted. The adopted expression is $c_v^0(T)=c_v^0(T_0)(T/T_0)^n$, where $n=\log(c_v^0(T_2)/c_v^0(T_1))/\log(T_2/T_1)$ and $T_1<T_0<T_2$, as suggested by Aly and Lee [92]. Furthermore, the van der Waals coefficients are calculated and stored before computations. For the first and second derivatives of the thermodynamic properties the reader is to the work by Mantecca et al. [36].

Bibliography

[1] Christopher. L. Rumsey et al. "Summary of the First AIAA CFD High-Lift Prediction Workshop". In: *Journal of Aircraft* 48.6 (2011), pp. 2068–2079. DOI: 10.2514/1.C031447.

[2] Christopher. L. Rumsey and Jeffrey. P. Slotnick. "Overview and Summary of the Second AIAA High-Lift Prediction Workshop". In: *Journal of Aircraft* 52.4 (2015), pp. 1006–1025. DOI: 10.2514/1.C032864.

[3] Christopher L. Rumsey, Jeffrey P. Slotnick, and Anthony J. Sclafani. "Overview and Summary of the Third AIAA High Lift Prediction Workshop". In: *Journal of Aircraft* 56.2 (2019), pp. 621–644. DOI: 10.2514/1.C034940.

[4] Christopher L. Rumsey, Jeffrey P. Slotnick, and Carolyn D. Woeber. "Fourth High-Lift Prediction/Third Geometry and Mesh Generation Workshops: Overview and Summary". In: *Journal of Aircraft* 60.4 (2023), pp. 1160–1177. DOI: 10.2514/1.C037168.

[5] Francesco Bassi et al. "Discontinuous Galerkin solution of the Reynolds-averaged Navier–Stokes and k–ω turbulence model equations". In: *Computers & Fluids* 34.4 (2005), pp. 507–540. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2003.08.004.

[6] Hong Luo, Joseph D. Baum, and Rainald Löhner. "A discontinuous Galerkin method based on a Taylor basis for the compressible flows on arbitrary grids". In: *Journal of Computational Physics* 227.20 (2008), pp. 8875–8893. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2008.06.035.

[7] Francesco Bassi et al. "Very High-Order Accurate Discontinuous Galerkin Computation of Transonic Turbulent Flows on Aeronautical Configurations". In: *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*. Ed. by Norbert Kroll et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 25–38. ISBN: 978-3-642-03707-8.

[8] Alejandra Uranga et al. "Implicit Large Eddy Simulation of transition to turbulence at low Reynolds numbers using a Discontinuous Galerkin method". In: *International Journal for Numerical Methods in Engineering* 87.1-5 (2011), pp. 232–261. DOI: https://doi.org/10.1002/nme.3036.

[9] Anirban Garai et al. "Scale-Resolving Simulations of Bypass Transition in a High-Pressure Turbine Cascade Using a Spectral Element Discontinuous Galerkin Method". In: *Journal of Turbomachinery* 140.3 (2017), p. 031004. ISSN: 0889-504X. DOI: 10.1115/1.4038403.

[10] Marta de la Llave Plata, Vincent Couaillier, and Marie-Claire le Pape. "On the use of a high-order discontinuous Galerkin method for DNS and LES of wall-bounded turbulence". In: *Computers & Fluids* 176 (2018), pp. 320–337. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2017.05.013.

[11] Francesco Bassi et al. "A p-adaptive Matrix-Free Discontinuous Galerkin Method for the Implicit LES of Incompressible Transitional Flows". In: *Flow, Turbulence and Combustion* 105 (2020). DOI: 10.1007/s10494-020-00178-2.

[12] Ralf Hartmann et al. "Discontinuous Galerkin methods for computational aerodynamics — 3D adaptive flow simulation with the DLR PADGE code". In: *Aerospace Science and Technology* 14.7 (2010), pp. 512–519. ISSN: 1270-9638. DOI: https://doi.org/10.1016/j.ast.2010.04.002.

[13] Li Wang and Dimitri J. Mavriplis. "Adjoint-based h–p adaptive discontinuous Galerkin methods for the 2D compressible Euler equations". In: *Journal of Computational Physics* 228.20 (2009), pp. 7643–7661. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2009.07.012.

[14] Norbert Kroll et al. *IDIHOM: Industrialization of High-Order Methods - A Top Down Approach. Results of a Collaborative Research Project Funded by the European Union, 2010-2014*. Vol. 128. 2015. ISBN: 978-3-319-12885-6. DOI: 10.1007/978-3-319-12886-3.

[15] John A. Cottrell, Thomas J. R. Hughes, and Yuri Bazilevs. "Isogeometric Analysis: Toward integration of CAD and FEA". In: 2009. ISBN: 0470748737. DOI: 10.1002/9780470749081.ch7.

[16] Wolfgang A. Wall, Moritz A. Frenzel, and Christian Cyron. "Isogeometric structural shape optimization". In: *Computer Methods in Applied Mechanics and Engineering* 197.33 (2008), pp. 2976–2988. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2008.01.025.

[17] John A. Cottrell, Thomas J.R. Hughes, and Alessandro Reali. "Studies of refinement and continuity in isogeometric structural analysis". In: *Computer Methods in Applied Mechanics and Engineering* 196.41 (2007), pp. 4160–4183. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2007.04.007.

[18] Thomas J. R. Hughes, John A. Cottrell, and Yuri Bazilevs. "Isogeometric analysis : CAD, finite elements, NURBS, exact geometry and mesh refinement". In: *Computer Methods in Applied Mechanics and Engineering* 194 (2005), pp. 4135–4195.

[19] Yury Bazilevs et al. "Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes". In: *Computer Methods in Applied Mechanics and Engineering* 199.13 (2010). Turbulence Modeling for Large Eddy Simulations, pp. 780–790. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2008.11.020.

[20] John A. Evans and Thomas J.R. Hughes. "Isogeometric divergence-conforming B-splines for the unsteady Navier–Stokes equations". In: *Journal of Computational Physics* 241 (2013), pp. 141–167. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2013.01.006.

[21] Ruben Sevilla, Sonia Fernández-Méndez, and Antonio Huerta. "NURBS-Enhanced Finite Element Method (NEFEM)". In: *Archives of Computational Methods in Engineering* 18.4 (2011), pp. 441–484. ISSN: 1886-1784. DOI: 10.1007/s11831-011-9066-5.

[22] Antonio S. Silveira et al. "Higher-order surface treatment for discontinuous Galerkin methods with applications to aerodynamics". In: *International Journal for Numerical Methods in Fluids* 79.7 (2015), pp. 323–342. DOI: https://doi.org/10.1002/fld.4050.

[23] Futao Zhang, Yan Xu, and Falai Chen. "Discontinuous Galerkin Methods for Isogeometric Analysis for Elliptic Equations on Surfaces". In: *Communications in Mathematics and Statistics* 2.3 (2014), pp. 431–461. ISSN: 2194-671X. DOI: 10.1007/s40304-015-0049-y.

[24] Ulrich Langer and Ioannis Toulopoulos. *Analysis of Multipatch Discontinuous Galerkin IgA Approximations to Elliptic Boundary Value Problems*. 2014.

[25] Jesse Chan and John A. Evans. "Multi-patch discontinuous Galerkin isogeometric analysis for wave propagation: Explicit time-stepping and efficient mass matrix inversion". In: *Computer Methods in Applied Mechanics and Engineering* 333 (2018), pp. 22–54. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2018.01.022.

[26] Shengjiao Yu, Renzhong Feng, and Tiegang Liu. "An isogeometric discontinuous Galerkin method for Euler equations". In: *Mathematical Methods in the Applied Sciences* 40.8 (2017), pp. 3129–3139. DOI: https://doi.org/10.1002/mma.4227.

[27] Shengjiao Yu, Renzhong Feng, Huiqiang Yue, Zheng Wang and Tiegang Liu. "Adjoint-Based Adaptive Isogeometric Discontinuous Galerkin Method for Euler Equations". In: *Advances in Applied Mathematics and Mechanics* 10.3 (2018), pp. 652–672. ISSN: 2075-1354. DOI: https://doi.org/10.4208/aamm.OA-2017-0046.

[28] Régis Duvigneau. "Isogeometric analysis for compressible flows using a Discontinuous Galerkin method". In: *Computer Methods in Applied Mechanics and Engineering* 333 (2018), pp. 443–461. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2018.01.039.

[29] Stefano Pezzano and Régis Duvigneau. "A NURBS-based Discontinuous Galerkin method for conservation laws with high-order moving meshes". In: *Journal of Computational Physics* 434.1 (2021). DOI: 10.1016/j.jcp.2020.110093.

[30] Kun Wang et al. "Adjoint-based airfoil optimization with adaptive isogeometric discontinuous Galerkin method". In: *Computer Methods in Applied Mechanics and Engineering* 344 (2019), pp. 602–625. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2018.10.033.

[31] Lisandro Dalcin et al. "PetIGA: A framework for high-performance isogeometric analysis". In: *Computer Methods in Applied Mechanics and Engineering* 308 (2016), pp. 151–181. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2016.05.011.

[32] Piero Colonna and Stefano Rebay. "Numerical simulation of dense gas flows on unstructured grids with an implicit high resolution upwind Euler solver". In: *International Journal for Numerical Methods in Fluids* 46.7 (2004), pp. 735–765. DOI: https://doi.org/10.1002/fld.762.

[33] Matteo Pini et al. "SU2: the Open-Source Software for Non-ideal Compressible Flows". In: *Journal of Physics: Conference Series* 821.1 (2017), p. 012013. DOI: 10.1088/1742-6596/821/1/012013.

[34] Jianhui Qi et al. "Development and validation of a Riemann solver in OpenFOAM® for non-ideal compressible fluid dynamics". In: *Engineering Applications of Computational Fluid Mechanics* 16.1 (2022), pp. 116–140.

[35] Salvatore Vitale et al. "Extension of the su2 open source cfd code to the simulation of turbulent flows of fuids modelled with complex thermophysical laws". In: *22nd AIAA computational fluid dynamics conference*. 2015, p. 2760.

[36] Edoardo Mantecca et al. "On the Development of an Implicit Discontinuous Galerkin Solver for Turbulent Real Gas Flows". In: *Fluids* 8.4 (2023). ISSN: 2311-5521. DOI: 10.3390/fluids8040117.

[37] Alessandro Colombo et al. "Development of a discontinuous Galerkin solver for the simulation of turbine stages". In: 2022. DOI: 10.23967/eccomas.2022.087.

[38] Gerald Farin. *Curves and Surfaces for CAGD*. Fifth Edition. Morgan Kaufmann, 2002. ISBN: 978-1-55860-737-8. DOI: https://doi.org/10.1016/B978-1-55860-737-8.50031-4.

[39] Les Piegl and Wayne Tiller. *The NURBS Book*. second. New York, NY, USA: Springer-Verlag, 1996.

[40] Maurice G. Cox. "The Numerical Evaluation of B-Splines*". In: *IMA Journal of Applied Mathematics* 10.2 (1972), pp. 134–149. ISSN: 0272-4960. DOI: 10.1093/imamat/10.2.134.

[41] Carl de Boor. "On calculating with B-splines". In: *Journal of Approximation Theory* 6.1 (1972), pp. 50–62. ISSN: 0021-9045. DOI: https://doi.org/10.1016/0021-9045(72)90080-9.

[42] Richard R. Patterson. "Projective transformations of the parameter of a Bernstein-Bézier curve". In: *ACM Trans. Graph.* 4 (1985), pp. 276–290.

[43] Francesco Bassi and Stefano Rebay. "A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier–Stokes Equations". In: *Journal of Computational Physics* 131.2 (1997), pp. 267–279. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.1996.5572.

[44] Steven Allmaras, Forrester Johnson, and Philippe Spalart. "Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model". In: *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)* (2012), pp. 1–11.

[45] Jeffery White et al. "Geometrically Flexible and Efficient Flow Analysis of High Speed Vehicles Via Domain Decomposition, Part 1, Unstructured-grid Solver for High Speed Flows". In: 2017.

[46] NASA Langley Research Center. *SST-Vm Expected Results - 2D Zero Pressure Gradient Flat Plate*.

[47] Ngoc Nguyen, Per-Olof Persson, and Jaime Peraire. "RANS Solutions Using High Order Discontinuous Galerkin Methods". In: *45th AIAA Aerospace Sciences Meeting and Exhibit*. DOI: 10.2514/6.2007-914.

[48] Todd Oliver and David Darmofal. "An Unsteady Adaptation Algorithm for Discontinuous Galerkin Discretizations of the RANS Equations". In: 1 (2007). DOI: 10.2514/6.2007-3940.

[49] Andrea Crivellini, Valerio D'Alessandro, and Francesco Bassi. "A Spalart-Allmaras Turbulence Model Implementation in a Discontinuous Galerkin Solver for Incompressible Flows". In: *J. Comput. Phys.* 241 (2013), pp. 388–415. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2012.12.038.

[50] Phlip L. Roe. "Approximate Riemann solvers, parameter vectors, and difference schemes". In: *Journal of Computational Physics* 43.2 (1981), pp. 357–372. ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(81)90128-5.

[51] Régis Duvigneau. "CAD-consistent adaptive refinement using a NURBS-based Discontinuous Galerkin method". In: *International Journal for Numerical Methods in Fluids* 92 (2020). DOI: 10.1002/fld.4819.

[52] Zhi J. Wang. *Adaptive High-Order Methods in Computational Fluid Dynamics*. World Scientific, 2011. DOI: 10.1142/7792.

[53] Friedrich Ringleb. "Exakte Lösungen der Differentialgleichungen einer adiabatischen Gasströmung". In: *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 20.4 (1940), pp. 185–198. DOI: https://doi.org/10.1002/zamm.19400200402.

[54] Katate Masatsuka. *I do like CFD, VOL.1, Second Edition*. Lulu, 2013, p. 303.

[55] Karl Wieghardt and Walter Tillmann. "On the turbulent friction layer for rising pressure". In: 1951.

[56] Antony Jameson. "Solution of the Euler equations for two dimensional transonic flow by a multigrid method". In: *Applied Mathematics and Computation - AMC* 13 (1983), pp. 327–355. DOI: 10.1016/0096-3003(83)90019-X.

[57] Charles L. Ladson, Acquilla S. Hill, and William G. Jr. Johnson. "Pressure distributions from high Reynolds number transonic tests of an NACA 0012 airfoil in the Langley 0.3-meter transonic cryogenic tunnel". In: (1987).

[58] N. Gregory and C. L. O'Reilly. "Low-Speed Aerodynamic Characteristics of NACA 0012 Aerofoi] Section , including the Effects of Upper-Surface Roughness Simulating Hoar Frost". In: 1970.

[59] Raphael Finkel and Jon Bentley. "Quad Trees: A Data Structure for Retrieval on Composite Keys." In: *Acta Inf.* 4 (1974), pp. 1–9. DOI: 10.1007/BF00288933.

[60] Timothy A. Rose, Peter D. Smith, and Shaun A. Forth. "Development of an adaptive mesh CFD code for high explosive blast simulation". In: (2005).

[61] Göktürk Kuru et al. "An Adaptive Variational Multiscale Discontinuous Galerkin Method For Large Eddy Simulation". In: *54th AIAA Aerospace Sciences Meeting*. DOI: 10.2514/6.2016-0584.

[62] Per-Olof Persson and Jaime Peraire. "Sub-Cell Shock Capturing for Discontinuous Galerkin Methods". In: *44th AIAA Aerospace Sciences Meeting and Exhibit*. DOI: 10.2514/6.2006-112.

[63] Lilia Krivodonova and Joseph E. Flaherty. "Error estimation for discontinuous Galerkin solutions of two-dimensional hyperbolic problems". In: *Advances in Computational Mathematics* 19.1-3 (2003), pp. 57–71. DOI: 10.1023/A:1022894504834.

[64] Fabio Naddei et al. "A comparison of refinement indicators for p-adaptive simulations of steady and unsteady flows using discontinuous Galerkin methods". In: *Journal of Computational Physics* 376 (2019), pp. 508–533. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2018.09.045.

[65] Jeremy Ims and Zhi J. Wang. "A comparison of three error indicators for adaptive high-order large eddy simulation". In: *Journal of Computational Physics* 490 (2023), p. 112312. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2023.112312.

[66] J E.A. John. "Gas dynamics. Second edition". In: (1984).

[67] Hans W. Liepmann, Anatol Roshko, and Robert Bruce Lindsay. "Elements of gasdynamics". In: *Physics Today* 10 (1957), pp. 41–42.

[68] John B. McDevitt and Arthur F. Okuno. "Static and dynamic pressure measurements on a NACA 0012 airfoil in the Ames High Reynolds Number Facility". In: (1985).

[69] H. Hoheisel. *Entwicklung neuer Entwurfskonzepte fur zwei Turbinengitter, Teil III, Ergebnisse T106*. Technical Report. Institut fur Entwurfsaerodynamik, Braunschweig, 1981.

[70] Jacques E.V. Peter and Richard P. Dwight. "Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches". In: *Computers & Fluids* 39.3 (2010), pp. 373–391. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2009.09.013.

[71] Antony Jameson. "Aerodynamic Shape Optimization Using the Adjoint Method". In: (2003).

[72] James Reuther et al. "Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation". In: *34th Aerospace Sciences Meeting and Exhibit*. DOI: 10.2514/6.1996-94.

[73] Siva Nadarajah and Antony Jameson. "The discrete adjoint approach to aerodynamic shape optimization". In: (2003).

[74] David A. Venditti and David L. Darmofal. "Adjoint Error Estimation and Grid Adaptation for Functional Outputs: Application to Quasi-One-Dimensional Flow". In: *Journal of Computational Physics* 164.1 (2000), pp. 204–227. ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.2000.6600.

[75] Lei Shi and Zhi J. Wang. "Adjoint-based error estimation and mesh adaptation for the correction procedure via reconstruction method". In: *Journal of Computational Physics* 295 (2015), pp. 261–284. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2015.04.011.

[76] Krzysztof J. Fidkowski. "Output-based error estimation and mesh adaptation for unsteady turbulent flow simulations". In: *Computer Methods in Applied Mechanics and Engineering* 399 (2022), p. 115322. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2022.115322.

[77] Siva Nadarajah and Antony Jameson. "A Comparison of the Continuous and Discrete Adjoint Approach to Automatic Aerodynamic Optimization". In: (2014). DOI: 10.2514/6.2000-667.

[78] Harold W. Kuhn and Albert W. Tucker. "Nonlinear programming". In: *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*. Univ. California Press, Berkeley-Los Angeles, Calif., 1951, pp. 481–492.

[79] Robert B. Wilson. "A Simplicial Algorithm for Concave Programming". In: *Ph.D. Thesis* (1963).

[80] Shih-Ping Han. "Superlinearly convergent variable metric algorithms for general nonlinear programming problems". In: *Mathematical Programming* 11.1 (1976), pp. 263–282. ISSN: 1436-4646. DOI: 10.1007/BF01580395.

[81] Michael J. D. Powell. "Algorithms for nonlinear constraints that use lagrangian functions". In: *Mathematical Programming* 14.1 (1978), pp. 224–248. ISSN: 1436-4646. DOI: 10.1007/BF01588967.

[82] Charles G. Broyden. "The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations". In: *IMA Journal of Applied Mathematics* 6.1 (1970), pp. 76–90. ISSN: 0272-4960. DOI: 10.1093/imamat/6.1.76.

[83] Roger Fletcher. "A new approach to variable metric algorithms". In: *The Computer Journal* 13.3 (1970), pp. 317–322. ISSN: 0010-4620. DOI: 10.1093/comjnl/13.3.317.

[84] Donald Goldfarb. "A Family of Variable-Metric Methods Derived by Variational Means". In: *Mathematics of Computation* 24.109 (1970), pp. 23–26. ISSN: 00255718, 10886842.

[85] David F. Shanno. "Conditioning of Quasi-Newton Methods for Function Minimization". In: *Mathematics of Computation* 24.111 (1970), pp. 647–656. ISSN: 00255718, 10886842.

[86] Ding-Yu Peng and Donald B. Robinson. "A New Two-Constant Equation of State". In: *Industrial & Engineering Chemistry Fundamentals* 15.1 (1976), pp. 59–64. DOI: 10.1021/i160057a011.

[87] Johannes D. Van der Waals. "The equation of state for gases and liquids". In: *Nobel Lectures, Physics* 1 (1910), pp. 254–265.

[88] Roland Span and Wolfgang Wagner. "A New Equation of State for Carbon Dioxide Covering the Fluid Region from the Triple-Point Temperature to 1100 K at Pressures up to 800 MPa". In: *Journal of Physical and Chemical Reference Data* 25.6 (1996), pp. 1509–1596. ISSN: 0047-2689. DOI: 10.1063/1.555991.

[89] Joseph J. Martin. "Cubic Equations of State-Which?" In: *Industrial & Engineering Chemistry Fundamentals* 18.2 (1979), pp. 81–97. DOI: 10.1021/i160070a001.

[90] Herbert B. Callen. "Thermodynamics and an Introduction to Thermostatistics". In: (1985).

[91] Roland Span. *Multiparameter equations of state : an accurate source of thermodynamic property data : with 151 figures and tables*. 2000. ISBN: 978-3-642-08671-7. DOI: 10.1007/978-3-662-04092-8.

[92] Fouad A. Aly and Lloyd L. Lee. "Self-consistent equations for calculating the ideal gas heat capacity, enthalpy, and entropy". In: *Fluid Phase Equilibria* 6.3 (1981), pp. 169–179. ISSN: 0378-3812. DOI: https://doi.org/10.1016/0378-3812(81)85002-9.

[93] Marcel Vinokur and Jean-Louis Montagné. "Generalized flux-vector splitting and Roe average for an equilibrium real gas". In: *Journal of Computational Physics* 89.2 (1990), pp. 276–300. ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(90)90145-Q.

[94] Paul Glaister. "An approximate linearised riemann solver for the Euler equations for real gases". In: *Journal of Computational Physics* 74.2 (1988), pp. 382–408. ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(88)90084-8.

[95] Ami Harten. "High resolution schemes for hyperbolic conservation laws". In: *Journal of Computational Physics* 49.3 (1983), pp. 357–393. ISSN: 0021-9991. DOI: https://doi.org/10.1016/0021-9991(83)90136-5.

[96] Ting Horng Chung et al. "Generalized multiparameter correlation for nonpolar and polar fluid transport properties". In: *Industrial & Engineering Chemistry Research* 27.4 (1988), pp. 671–679. DOI: 10.1021/ie00076a024.

[97] Turboden S.p.A. *ORC Turbine Nozzle*.

[98] Piero Colonna et al. "Real-Gas Effects in Organic Rankine Cycle Turbine Nozzles". In: *Journal of Propulsion and Power* 24.2 (2008), pp. 282–294. DOI: 10.2514/1.29718.

[99] Howser L. M. Smith R. E. Jr. Price J. M. *A smoothing algorithm using cubic spline functions*. Technical Report. NASA Langley Research Center Hampton, VA, United States, 1974.

[100] Steven A. Coons. "Surfaces for Computer-Aided Design of space forms". In: 1967.

[101] Bruce E. Poling, John M. Prausnitz, and John P. O'Connell. *The Properties of Gases and Liquids 5E*. McGraw Hill professional. McGraw Hill LLC, 2000. ISBN: 9780071499996.

[102] Mark S. Cramer. "Negative nonlinearity in selected fluorocarbons". In: *Physics of Fluids A: Fluid Dynamics* 1.11 (1989), pp. 1894–1897. ISSN: 0899-8213. DOI: 10.1063/1.857514.