UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE SCIENCES FONDAMENTALES ET APPLIQUÉES

Inria

THÈSE DE DOCTORAT

Inria Sophia-Antipolis Méditerranée

Isogeometric Discontinuous Galerkin method with time-dependent domains

---

Méthode de Galerkin Discontinue Isogéométrique avec domaines dépendants du temps

---

Stefano Pezzano

Présentée en vue de l'obtention du grade de docteur
en Mathématiques d'Université Côte d'Azur

Dirigée par: Régis Duvigneau
Soutenue le: 13 Septembre 2021

Devant le jury, composé de:

Angelo Iollo	Université de Bordeaux	Président du jury
Christophe Corre	École Centrale de Lyon	Rapporteur
Michael Dumbser	Universitá di Trento	Rapporteur
Nathalie Bartoli	ONERA	Examinatrice
Marianna Braza	CNRS, IMFT	Examinatrice
Matthias Möller	TU Delft	Examinateur
Régis Duvigneau	Inria, ACUMES	Directeur de Thèse

Abstract

Time-dependent domains are encountered in a vast category of fluid mechanics applications. Such problems are often characterised by complex geometries and physical phenomena. The aim of the present dissertation is the development of an accurate and reliable methodology to investigate compressible flows with time-dependent geometries. To this end, we combine ideas from Isogeometric Analysis (IGA) and high-order numerical schemes for fluid dynamics, specifically Discontinuous Galerkin (DG) methods.

We start by discussing the implementation details of the Isogeometric DG method. To this end, we introduce the DG scheme for conservation laws and the fundamentals of IGA. The mathematical representation of Computer Aided Design (CAD) is explained and some important geometry manipulation algorithms are presented. We then show how a DG-compatible description can be extracted starting from a CAD object, allowing us to employ a geometrically exact DG discretisation to solve the equations of fluid mechanics. Finally, we conduct two simple numerical experiments to illustrate the impact of the boundary representation on flow simulations.

We then proceed to extend the Isogeometric DG methodology to deformable domains using the Arbitrary Lagrangian-Eulerian (ALE) formalism. After an analysis of the existing ALE-DG schemes, we propose an ALE formulation using the Isogeometric framework. The CAD basis functions are also adopted to deform the mesh, leading to a fully unified description of the simulation variables. Besides, we show how the proposed grid velocity algorithm can be seamlessly coupled with Adaptive Mesh Refinement (AMR). The presented approach is firstly validated with two analytical problems, showing optimal convergence rates. Then, we simulate the flows past an oscillating cylinder and a pitching airfoil to prove the robustness of the methodology. We conclude by demonstrating the capabilities of the developed ALE-AMR coupling algorithm using the pitching airfoil benchmark.

Secondly, we propose to employ sliding grids for flows with rotating components, as an alternative to the deformation-based ALE technique. Using the CAD basis functions, circular interfaces can be exactly represented. Therefore, it is possible to develop a high-order and fully conservative sliding mesh formulation. We begin by detailing the treatment of the sliding interface geometry and the computation of the numerical fluxes. Then, we show the optimal convergence of the implemented approach using an analytical problem. We further characterise the behaviour of the sliding interface considering the flow past a pitching ellipse. We show that the sliding mesh technique and the deformation-based ALE methodology are equivalent in terms of accuracy. Finally, a vertical axis wind turbine is simulated to present

a more topologically complex test case.

Lastly, we apply the proposed ALE formulation to a flow control problem. In particular, we consider a dynamic trailing edge morphing actuation to control the separation of the laminar boundary layer around a NACA 6-series airfoil. Using the proposed mesh deformation algorithm, we develop a high-order accurate trailing edge morphing model. In order to find the optimal set of control parameters, the flow solver is coupled with a Bayesian optimisation algorithm. Thanks to the strong coupling between the geometry description and the numerical schemes, the resulting design chain is fully automated. The capabilities of the proposed framework are explored considering both single-objective and multi-objective optimisation problems.

Keywords: *Computational Fluid Mechanics, Discontinuous Galerkin, Isogeometric Analysis, Arbitrary Lagrangian-Eulerian Formulation, Sliding Meshes*

Contents

List of Figures

List of Tables

Introduction

In the last 20 years, Computational Fluid Dynamics (CFD) has become a standard tool for aerodynamic analysis and design. In this context, the issue of a better integration of CFD solvers in complex multidisciplinary analysis chains has emerged. In particular, the lack of integration between Computer Aided Design (CAD) and CFD software regarding the treatment of the geometry has been reported. This difficulty is mainly due to the fact that, in most industrial cases, the grid generation process is not fully automated yet and requires several geometrical transformations, such as defeaturing and geometry repair, that are still carried out by hand. In the context of multidisciplinary design chain, it is vital to have a high degree of automation in order to efficiently search for the optimal design. In order to facilitate the integration between CAD and simulation, Hughes et al. (1) introduced the concept of Iso-Geometric Analysis (IGA), whose formulation relies on Non-Uniform Rational B-Splines (NURBS), the standard mathematical representation of modern CAD (2). Essentially, IGA consists in a continuous finite element method in which both the geometry and the solution are described by NURBS functions. It is well known, however, that Continuous Galerkin (CG) schemes need to be stabilised in order to deal with convection dominated problems (3; 4). Furthermore, additional stabilisation is required to properly capture discontinuities (5). Due to the mentioned difficulties, the application of IGA to compressible fluid mechanics has been so far very limited. An investigation of Lagrangian hydrodynamics has been conducted by Bazilevs et al. (6) using the IGA framework. Möller et al. (7) recently presented a stabilised Isogeometric CG scheme for the compressible Euler equations. On the other hand, IGA has been successfully employed to simulate incompressible flows with complex geometries (8; 9; 10). Nevertheless, some limitations are still encountered in generating NURBS-based grids for complex topologies. However, new approaches are emerging to construct NURBS-based parameterizations of computational domains, inspired by both structured (11) and unstructured (12; 13; 14) mesh generation techniques.

Meanwhile, the CFD community has developed a growing interest in high-order schemes for conservation laws (15). The majority of the production-level CFD simulations is carried out with finite volume discretisations based on linear reconstruction, hence second-order schemes. When high mesh resolutions are required, such solvers can become inefficient with

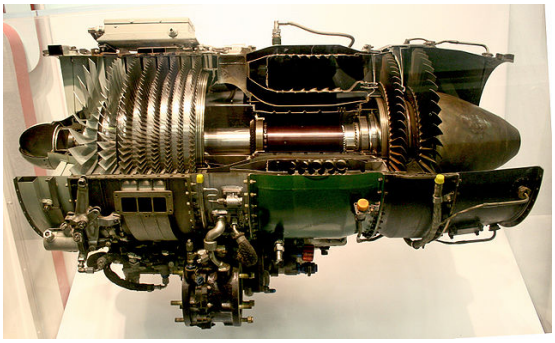
respect to more accurate schemes. In order to improve finite volume discretisations, Harten et al. (16) introduced Essentially Non-Oscillatory (ENO) reconstructions, which were later extended by Liu et al. (17) to construct the family of Weighted ENO (WENO) schemes. However, the implementation and parallelisation of WENO reconstructions on unstructured grids is not trivial, due to the non-compact stencil. Such issues led to the study and development of compact high-order discretisations, such as the Discontinuous Galerkin (DG) scheme. Oden et al. (18) presented the first application of discontinuous finite elements to inviscid compressible flows and the extension to Navier-Stokes equations was first proposed by Bassi et al. (19). Thanks to the combination of stability and high-order accuracy, DG schemes have been employed in a variety of complex CFD applications. For example, the simulation of a flapping wing was performed in (20). Luo et al. (21) illustrated the capabilities of the DG schemes for flows with shocks and performed a simulation of a full aircraft geometry in transonic flight conditions. A DG discretisation of the Reynolds Averaged Navier-Stokes (RANS) equations has been employed for the simulation of a turbine cascade by Bassi et al. (22) and an aircraft wing-body configuration by Hartmann et al. (23). Moreover, the low numerical dissipation and dispersion of DG schemes are suitable characteristics to efficiently simulate turbulent flows (24; 25) and aeroacoustic problems (26).

However, as shown by several authors, the use of piecewise-linear geometry descriptions in high-order CFD solvers may lead to non-physical phenomena in numerical solutions (19; 27; 28) and generally limits the accuracy of such schemes to second order for problems with curvilinear geometries. This observation justified the development of a new generation of high-order solvers, capable of handling CAD geometries natively. One can refer for instance to the NURBS-Enhanced Finite Element Method (NEFEM) (29) and its Discontinuous Galerkin counterpart (27), in which the NURBS representations are only employed for the geometry of the boundaries. Similarly, an extension of the residual distribution scheme for NURBS geometries has been proposed (30). More recently, two DG schemes inspired by IGA have been developed: the Blended-Isogeometric Discontinuous Galerkin (BIDG) method (31) which relies on nodal elements whose geometry is defined by NURBS in the whole computational domain, and its fully isogeometric counterpart which uses NURBS for both the geometry and the solution (32). All these approaches have demonstrated the interest of using high-order boundary representations in terms of accuracy for some selected test-cases governed by compressible or incompressible flow models. Besides, the DG framework may alleviate the mentioned difficulties of generating NURBS-based grids, thanks to the higher flexibility of the DG method in handling non-conformities and local refinement (33).

At the same time, complex flows often involve time-dependent geometries or moving interfaces. We illustrate in Fig. 1 some sample applications in which the computational domain is time-dependent. As a first example, we consider the aerodynamics of modern civil air-

crafts, which are characterised by extremely flexible and lightweight structures, thanks to the widespread use of composite materials. As a consequence, it is critical to take into account the deflection of the wings in order to accurately predict the airflow and correctly design the lifting surfaces. Another key application is represented by turbomachinery, in which it is necessary to compute the flow about rotating geometries. Besides, rotors are usually followed by stators, meaning that the numerical solver has to deal with a rapidly rotating component in close proximity of a fixed geometry. Similar challenges are encountered in the numerical simulation of wind turbines. Furthermore, other examples of flows with time-dependent geometries can be found in civil engineering, where the effects of wind plays a fundamental role in the design of slender structures, like skyscrapers and long-span bridges. Being able to predict the interaction between the structure and the wind is essential in order to prevent catastrophic failures, such as the infamous collapse of the Tacoma Narrows bridge. Another important application of Fluid-Structure Interaction (FSI) is haemodynamics, in which it is crucial to consider the deformation of vessels to estimate the bloodstream. Lastly, time-dependent domains are also used to simulate multi-material flows, which are characterised by multiple immiscible fluids separated by a certain number of moving interfaces. Multi-material flow formulations are often used to model explosion problems. Although not exhaustive, the provided list of examples covers a wide range of significantly different disciplines.

(a) Flexible aircraft structures



(b) Jet engines

(c) Wind turbines

(d) Skyscrapers

(e) Long-span bridges

Figure 1: Example applications of time-dependent domains in engineering. Authors of the pictures: (a) Andrew W. Sieber, (b) Sanjay Acharya, (c) David Clarke, (d) Fred Hsu, (e) Fred Romero

Given the number and the impact of the possible applications, numerical algorithms for flows with time-dependent domains play a major role in CFD. Furthermore, such applications are often characterised by complex geometries and highly non-linear physical phenomena, as testified by the mentioned examples. In this context, the adoption of high-order numerical schemes can provide significant advantages with respect to traditional second-order solvers. Firstly, the faster convergence allows to reduce the number of degrees of freedom required to properly resolve complex flow fields. Secondly, improving the description of the geometry is certainly beneficial, in particular because the displacement of the boundary may amplify the spurious phenomena generated by a low-order geometry representation. Therefore, simulating flows while accounting exactly for CAD geometries should be an advantage over traditional approaches where the moving boundaries are piecewise linear. This has been illustrated for FSI problems in the context of classic IGA (34; 35). The aim of the present work is therefore the development of a reliable and accurate methodology for the solution of compressible flows with time-dependent domains. The proposed framework is based on the Isogeometric Discontinuous Galerkin (32), thanks to which a tight coupling between geometry and simulation is achieved.

In order to analyse their properties, we classify the existing methodologies for the solution of conservation laws on time-dependent domains into three categories: Eulerian, Lagrangian and Arbitrary Lagrangian-Eulerian (ALE) algorithms. In the first family, the Eulerian form of the equation of fluid mechanics is solved and the mesh is fixed. Such an approach is natural when the boundary of the computational domain is not moving, as the mesh can be boundary-fitted. In order to extend the Eulerian viewpoint with time-dependent domains, the Immersed Boundary method was proposed by Peskin (36). In this framework, the boundary of the fluid domain is immersed within the cells of the computational mesh. The main advantage of such a methodology is the possibility of using Cartesian grids (37; 38), allowing to drastically reduce the cost of mesh generation. Moreover, immersed boundary techniques can be easily used to simulate large domain deformations, since the mesh is not boundary-fitted. On the other hand, the position of immersed interface with respect to the grid has to be re-estimated at each time iteration. Besides, spurious phenomena can be induced by an inaccurate treatment of the immersed boundary, causing non-physical oscillations in the predicted aerodynamic forces (39). In the context of DG schemes, the cut-cell immersed boundary method has been successfully employed to simulate both compressible (40; 41) and incompressible flows (42). A DG discretisation of the penalised Navier-Stokes equations has been adopted for the investigation of a screw compressor in (43). Lastly, Kamensky et al. (44) proposed to adopt a CAD-consistent representation in the immersed boundary framework. The developed approach, named Immersogeometric Analysis, relies on a second-order stabilised CG solver for the incompressible flow and on a weak imposition of the boundary conditions on the immersed

CAD surface. The extension of the Immersogeometric methodology to compressible flow problems has been presented and applied to the simulation of a rotorcraft by Xu et al. (45).

The antithetical approach consists in the use of the Lagrangian viewpoint, in which the computational grid exactly follows the movement of the fluid. Since the Lagrangian form of the conservation laws is employed, the convective term is absent. As a consequence, the discretisation is less complex with respect to Eulerian schemes and, ideally, the resulting algorithm is not affected by numerical dissipation. Such characteristics make Lagrangian methods extremely effective for tracking moving boundaries, interfaces and discontinuities. In an effort to develop high-order Lagrangian schemes, several DG formulations were proposed and studied for hydrodynamics problems (46; 47; 48). Moreover, since the mesh is forced to move at the same speed as the fluid particles, the use of piecewise-linear grids limits Lagrangian schemes to second-order accuracy. This justified the development of curvilinear finite elements in Lagrangian algorithms (49). Bazilevs et al. (6) demonstrated the benefits of IGA for Lagrangian hydrodynamics, and, in the same field, a Finite Volume method based on conical cells has been presented in (50). As a drawback, the Lagrangian description can lead to highly distorted grids. For this reason, mesh-based Lagrangian methods are not usually employed for the computation of vortical flows.

In order to combine the best features of the two classical viewpoints, the ALE framework was developed. The ALE methodology represents a generalisation of the Eulerian and Lagrangian descriptions and allows to perform simulations with meshes that deform in an arbitrary way. Considering the case of a moving boundary, ALE algorithms permit the adoption of a Lagrangian description for the boundary region and an Eulerian viewpoint for the far-field, with a smoothly deformed transition zone. Thanks to this capability, ALE methods are commonly used for CFD applications that present time-dependent domains. For this reason, various ALE-DG schemes have been proposed for the solution of the Navier-Stokes equations with application to external flows (20; 51; 52). Developing a robust methodology to deform curvilinear grids is not straightforward, therefore, this first generation of ALE-DG solvers still relied on piecewise-linear grids. More recently, however, a family of ALE algorithms with high-order mesh deformations has been proposed for both WENO and DG schemes (53; 54). Such approaches have demonstrated the interest in using moving curvilinear grids for flows with strong discontinuities. Similarly, Anderson et al. (55) showed the advantages of curvilinear ALE to track multi-material flow interfaces. Although more flexible with respect to Lagrangian methods, ALE algorithms based purely on mesh deformation still have difficulties in coping with large displacements of the boundary. In order to guarantee the best possible grid quality over time, a remeshing step can be added (56; 57; 58). On the other hand, whenever the grid topology is updated, the numerical solution has to be remapped from the old to the new mesh, with a possible loss of conservativity. An alternative to the ALE-remeshing

coupling is the use of space-time DG formulations (59; 60; 61), in which the topology change can be performed via a proper construction of the space-time mesh.

Another alternative to remeshing is the use of overset and sliding grids. Both approaches have been successfully employed for problems characterised by large displacements of the boundaries (62; 63; 64; 65). In particular, overset grids are especially effective for CFD simulations of multi-body systems, since each component can be meshed independently. The main technical difficulties are represented by the computation of the overlap regions between the independent grids and the evaluation of the fluxes in the overlap band. The first DG method with overset meshes has been proposed by Galbraith et al. (66; 67) for fixed boundaries. Subsequently, DG solvers with overlapping grids and time-dependent domains have been presented in (68; 69). Lastly, sliding meshes can be seen as the limit case of the overset approach, with an overlap region constituted exclusively by the sliding interface. The main applications of sliding techniques are flows with rotating components, in which it is natural to subdivide the computational domain into a fixed and a rotating region separated by a circular interface. Bazilevs et al. (70) exploited the properties of IGA to develop an efficient sliding mesh algorithm. In the DG framework, Ferrer et al. (71) developed the first sliding grid solver for incompressible Navier-Stokes equations, whereas an algorithm for compressible flow was illustrated in (72).

In this work we focus on the ALE methodology. In order to allow the investigation a wide range of problems, we develop two complementary approaches, the first one is based on mesh deformation, whereas the second relies on sliding grids. The primary goal is to demonstrate the potential of IGA and the benefits of using a high-order accurate boundary representations for applications with time-dependent domains. The second aim is to propose an efficient aerodynamic design methodology exploiting the tight coupling between geometry and simulation provided by the Isogeometric paradigm. The manuscript is organised as follows. In chapter 1 we present the fundamentals of the Isogeometric DG schemes for Navier-Stokes equations. Chapter 2 is dedicated to the development and validation of the ALE formulation and the mesh deformation algorithm. In chapter 3 we propose a high-order accurate sliding grid technique. In chapter 4 we apply the developed ALE methodology to a flow control problem and we carry out an optimisation study. Lastly, the main outcomes and perspectives are discussed in the conclusion.

Open source code

All the methodologies presented in this dissertation are implemented in the **Igloo** software suite, developed at Inria. The source code is available, under the GNU General Public Licence v3, at the following repository: `https://gitlab.inria.fr/igloo/igloo/-/wikis/home`.

Chapter 1

Isogeometric DG scheme

In this chapter, we illustrate the main ideas behind the Isogeometric Discontinuous Galerkin scheme. We briefly introduce the well-known DG method for a generic system of conservation laws. We then present some of the mathematical concepts of the CAD representation along with some useful geometry manipulation algorithms. Next, the Isogeometric DG method is detailed and two test cases are discussed.

1.1 Discontinuous Galerkin method

We consider, for the sake of generality, a system of conservation laws of the following form:

$$\frac{\partial \mathbf{W}}{\partial t} + \nabla \cdot \mathbf{F} = 0, \tag{1.1}$$

with $\mathbf{W}$ being the vector of conservative variables and $\mathbf{F}$ the physical flux. The computational domain Ω is discretized using a set of elements Ω_j, $j = \{1, 2, ..., N_{el}\}$. In order to discretise eq. (1.1) using a DG scheme, the numerical solution $\mathbf{w}_h$ is expressed in each element of index j as:

$$\mathbf{w}_h^j = \sum_{i=1}^{N_p} \varphi_i^j(\mathbf{x})\, \mathbf{w}_i^j\,, \tag{1.2}$$

with φ_i^j being a generic set of basis functions and $\mathbf{w}_i^j$ the N_p degrees of freedom, defined on the j-th element. The weak formulation is obtained multiplying eq. (1.1) by a basis function φ_k and integrating over the elemental domain Ω_j:

$$\int_{\Omega_j} \varphi_k \frac{\partial \mathbf{w}_h}{\partial t}\, \mathrm{d}\Omega + \int_{\Omega_j} \varphi_k \nabla \cdot \mathbf{F}\, \mathrm{d}\Omega = 0. \tag{1.3}$$

Note that the index j on the basis functions and the discrete solution has been dropped to ease the notation. Then, integrating by parts, the classic DG discretisation is found:

$$\int_{\Omega_j} \varphi_k \frac{\partial \mathbf{w}_h}{\partial t}\, \mathrm{d}\Omega = \int_{\Omega_j} \nabla \varphi_k \cdot \mathbf{F}\, \mathrm{d}\Omega - \oint_{\partial\Omega_j} \varphi_k \mathbf{F}^*\, \mathrm{d}\Gamma. \tag{1.4}$$

Since the discrete solution admits multiple values across elements boundaries, the physical flux in the surface integral is replaced by a numerical flux $\mathbf{F}^*(\mathbf{w}_h^+, \mathbf{w}_h^-, \mathbf{n})$, which is computed with a consistent Riemann solver, according to the values of the solution $\mathbf{w}_h^+$ and $\mathbf{w}_h^-$ that prevail at each side of the interface and the unitary normal vector $\mathbf{n}$, as in finite volume methods. Depending on the choice the basis functions, different DG methods can be developed. In particular, the family of *nodal* DG schemes (73) relies on Lagrange polynomials, whereas, *modal* DG approaches are based on a set of orthogonal basis functions (25). In this work, we employ a DG discretisation with a CAD-consistent representation.

1.2 Fundamentals of CAD representations

Before detailing the implementation of the Isogeometric DG, a brief introduction to the mathematical foundations of CAD is needed. The present section is thus dedicated to illustrate the main features of the CAD representation.

1.2.1 Bézier curves

The construction of a CAD-consistent scheme necessarily relies on bases used in CAD. The first approach to interactive geometric design consisted in the use of Bézier curves, defined as (2):

$$\mathbf{C}(\xi) = \sum_{i=1}^{p+1} B_i^p(\xi)\ \mathbf{x}_i, \tag{1.5}$$

where $\mathbf{x}_i$ are the control points, and B_i^p are the Bernstein polynomials of degree p:

$$B_i^p(\xi) = \binom{p}{i-1} \xi^{i-1}(1-\xi)^{p-i-1}, \tag{1.6}$$

with $\xi \in [0, 1]$. An example of cubic Bézier curve is represented in Fig. 1.1, where the black dots represents the control points of the curve. The polygon defined by the control points, represented by the dashed black lines, is known as *control polygon*.

Polynomial curves are not able to exactly represent conic sections, therefore, in order to overcome this limitation, rational Bernstein functions were introduced (74):

$$R_i^p(\xi) = \frac{B_i^p(\xi)\,\omega_i}{\sum_{j=1}^{p+1} B_j^p(\xi)\,\omega_j}, \tag{1.7}$$

with the coefficients ω_i being positive real numbers called weights. Since Bernstein polynomials are a partition of unity, $R_i^p(\xi)$ coincides with $B_i^p(\xi)$ when the weights are uniform. Rational Bézier surfaces are defined as:

$$\mathbf{C}(\xi) = \sum_{i=1}^{p+1} R_i^p(\xi)\ \mathbf{x}_i. \tag{1.8}$$

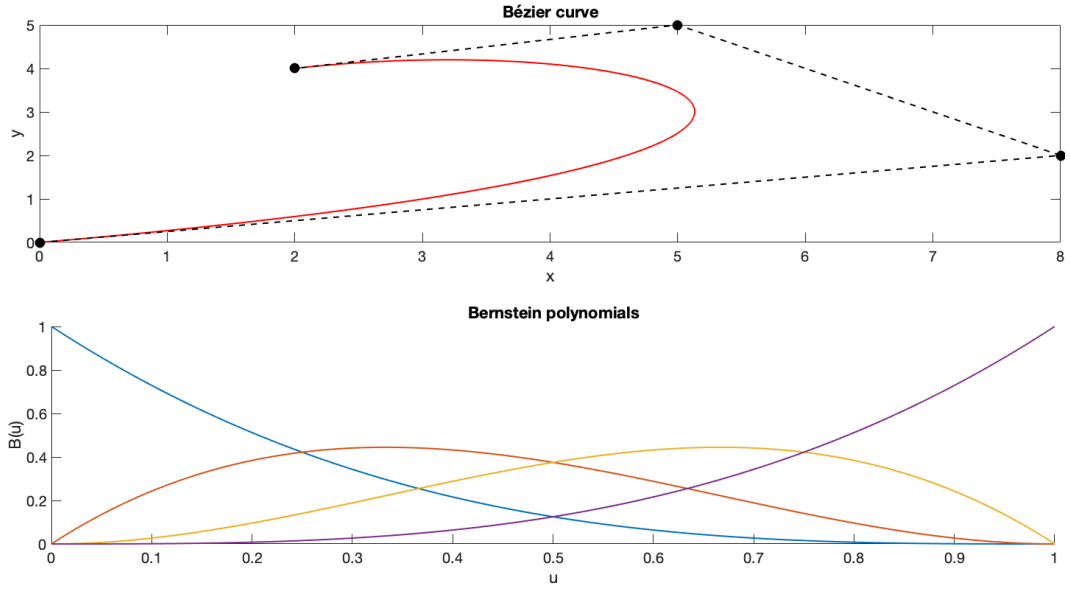


Figure 1.1: Example of cubic Bézier curve with associated basis functions

The Bézier representation possesses several interesting mathematical properties (74), among which we cite:

- non-negativity: $R_i^p(\xi) \geq 0$, $\forall i, p$ and $\forall \xi \in [0, 1]$;
- convex hull property: the curve $\mathbf{C}$ is always contained in the convex hull defined by the control points $\mathbf{x}_i$;
- affine invariance: applying an affine transformation (rotation, translation and scaling, etc.) to the control polygon is equivalent to applying an affine transformation to the entire curve.

1.2.2 B-Splines and NURBS curves

Complex geometries require a high-degree basis when represented using functions defined on a single support, leading to several drawbacks from the numerical point of view. Therefore, as explained in (1), CAD software representations are commonly based on B-Splines, rather than Bézier curves. To introduce B-Splines, we consider the parametric domain $= [\xi_1, \xi_l]$, discretized by the knot vector $\Xi = \{\xi_1, \ldots, \xi_i, \ldots, \xi_l\}$. B-Spline functions of degree p are evaluated recursively:

$$N_i^0(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

$$N_i^p(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_i^{p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1}^{p-1}(\xi). \tag{1.10}$$

The number of basis functions is determined by both the number of knots l and the degree:

$$n = l - p - 1. \tag{1.11}$$

Conventionally, the i-th knot span is defined as the half-open interval $[\xi_i, \xi_{i+1})$. One of the main characteristic of the B-Spline basis is the local support property: each basis function N_i^p is non-zero exclusively on the interval $[\xi_i, \xi_{i+p+1})$, implying that at most $p+1$ basis functions are non-zero on each knot span. It is important to note that knots can be repeated and zero length knot spans are possible. The multiplicity of a knot plays an essential role in the regularity of B-Spline functions. In fact, if we denote s as the multiplicity of a generic knot ξ_i, the basis functions N_i^p are $\mathcal{C}^{p-s}$ regular at ξ_i. Commonly, in CAD representations, *open* knot vectors are employed, i.e., the multiplicity of the first and last knots is equal to $p+1$. In this work we will exclusively employ open knot vectors.

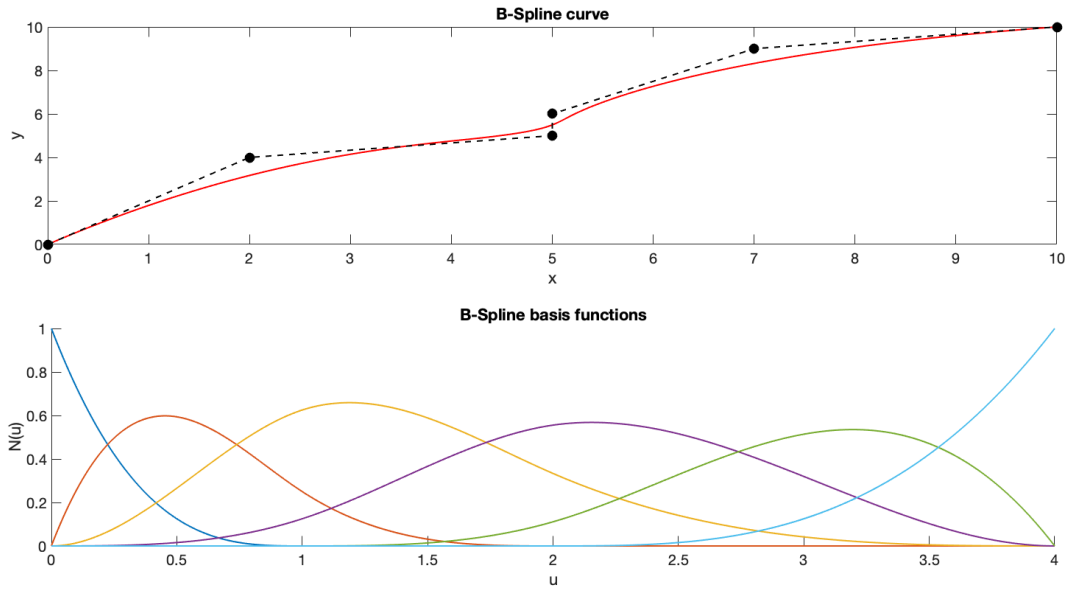


Figure 1.2: Example of cubic B-Spline curve with associated basis functions

Non-Uniform Rational B-Splines (NURBS) functions are defined as the rational extension of B-Splines:

$$R_i^p(\xi) = \frac{N_i^p(\xi)\,\omega_i}{\sum_{j=1}^n N_j^p(\xi)\,\omega_j}\,. \tag{1.12}$$

As for Bézier geometries, NURBS curves are represented by means of control points:

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} R_i^p(\xi)\mathbf{x}_i. \tag{1.13}$$

An example of cubic NURBS curve with knot vector $\{0,0,0,0,1,2,4,4,4,4\}$ is illustrated in Fig. 1.2. The previously cited properties of Bézier curves are valid for NURBS representations as well.

1.2.3 CAD surfaces

The description of multidimensional CAD objects usually relies on tensor product representations (2). Polynomial tensor product surfaces based on the Bernstein basis are defined as follows:

$$\mathbf{S}(\xi,\eta) = \sum_{i_1=1}^{p+1}\sum_{i_2=1}^{p+1} B_{i_1}^p(\xi)\, B_{i_2}^p(\eta)\, \mathbf{x}_{i_1 i_2}, \tag{1.14}$$

where the control points $\mathbf{x}_{i_1 i_2}$ form a bidirectional net, as in the example provided in fig. 1.3. In order to describe rational patches, we first define the bivariate rational Bernstein functions:

$$R_{i_1 i_2}^p(\xi,\eta) = \frac{B_{i_1}^p(\xi)\, B_{i_2}^p(\eta)\, \omega_{i_1 i_2}}{\sum_{j_1=1}^{p+1}\sum_{j_2=1}^{p+1} B_{j_1}^p(\xi)\, B_{j_2}^p(\eta)\, \omega_{j_1 j_2}}. \tag{1.15}$$

The expression of a rational Bézier surface is then:

$$\mathbf{S}(\xi,\eta) = \sum_{i_1=1}^{p+1}\sum_{i_2=1}^{p+1} R_{i_1 i_2}^p(\xi,\eta)\, \mathbf{x}_{i_1 i_2}\,. \tag{1.16}$$

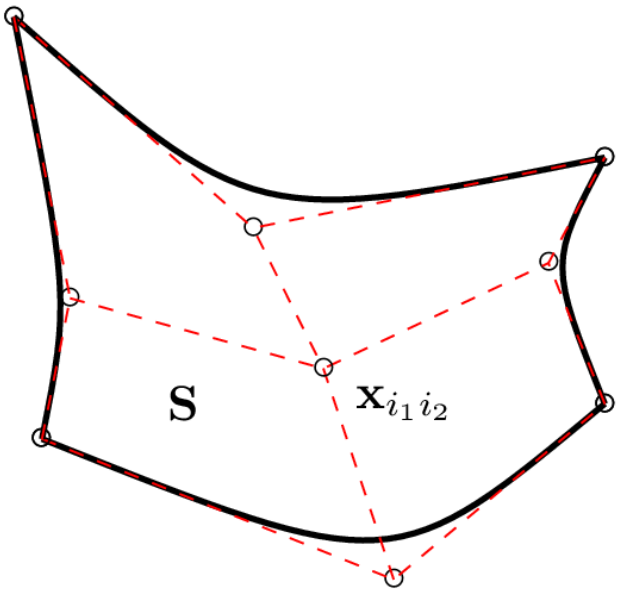


Figure 1.3: Example of quadratic Bézier surface

The construction of NURBS surfaces is based on the same principles we illustrated for Bézier representations. Two separate knot vectors can be employed to build the B-Spline basis in the ξ and η directions, providing a lot of flexibility. Both Bézier and NURBS surfaces inherit the mathematical properties mentioned in section 1.2.1.

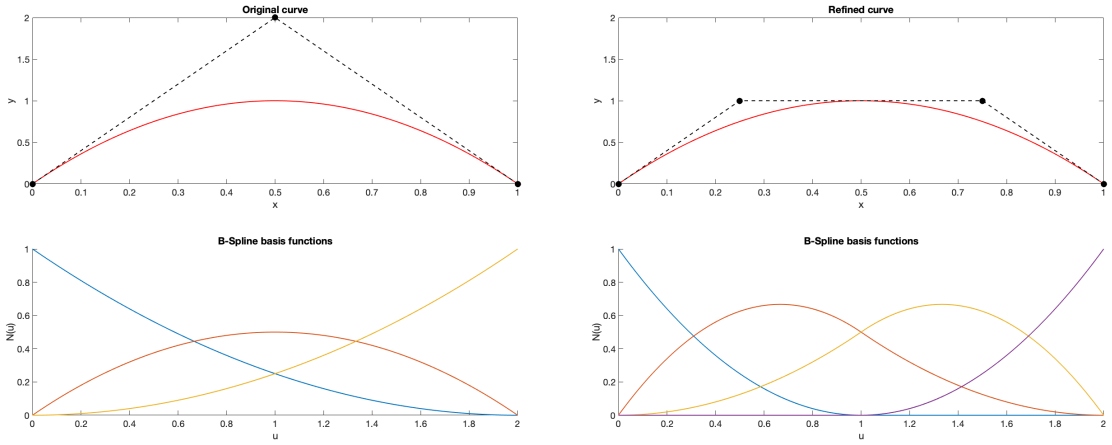
1.2.4 Knot insertion

An important geometry manipulation technique is the knot insertion algorithm. Let us consider a B-Spline curve $\mathbf{C}$ with knot vector $\Xi = \{\xi_1, \dots, \xi_l\}$ and control points $\mathbf{x}$. Thanks to the properties of the B-Spline basis, a new knot $\bar{\xi}$ can be added to Ξ without altering the geometry of the curve (74). As a consequence of the additional knot, the spline space is enriched and the curve is represented with one more control point. The updated knot vector can be written as $\bar{\Xi} = \{\xi_1, \dots, \xi_q, \bar{\xi}, \xi_{q+1}, \dots, \xi_l\}$ and the curve can be represented as:

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} N_i^p(\xi)\,\mathbf{x}_i = \sum_{i=1}^{n+1} \bar{N}_i^p(\xi)\,\bar{\mathbf{x}}_i, \tag{1.17}$$

where $\bar{N}_i^p$ is the set of B-Spline functions defined by the knot vector $\bar{\Xi}$. The control points of the new representation can be explicitly computed with the following formula:

$$\bar{\mathbf{x}}_i = (1-\alpha_i)\mathbf{x}_{i-1} + \alpha_i \mathbf{x}_i, \quad \alpha_i = \begin{cases} 1 & \text{if } i \le q-p \\ \dfrac{\bar{\xi} - \xi_i}{\xi_{i+p} + \xi_i} & \text{if } q-p+1 \le i \le q \\ 0 & \text{if } i \ge q+1 \end{cases} \tag{1.18}$$



(a) Original curve (b) Refined curve

Figure 1.4: Example of knot insertion

When a NURBS curve is considered, the knot insertion formula (1.18) cannot be immediately applied. First, the d-dimensional rational curve must be represented as a (d+1)-dimensional polynomial curve using homogenous coordinates (74). Defining ω as:

$$\omega = \sum_{i=1}^{n} N_i^p(\xi)\,\omega_i, \tag{1.19}$$

then, in the case of a bidimensional curve, we can write:

$$\mathbf{C}^{\omega}(\xi) = \begin{pmatrix} x\omega \\ y\omega \\ \omega \end{pmatrix} = \sum_{i=1}^{n} N_i^p(\xi) \begin{pmatrix} x_i\omega_i \\ y_i\omega_i \\ \omega_i \end{pmatrix} = \sum_{i=1}^{n} N_i^p(\xi)\,\mathbf{x}_i^{\omega} \tag{1.20}$$

The updated coordinates and weights are obtained by applying eq. (1.18) to $\mathbf{x}_i^{\omega}$:

$$\bar{\mathbf{x}}_i^{\omega} = (1-\alpha_i)\mathbf{x}_{i-1} + \alpha_i \mathbf{x}_i^{\omega}, \tag{1.21}$$

using the previously defined α_i coefficients. Then, the enriched (d+1)-dimensional polynomial curve is transformed back to a d-dimensional rational curve to obtain the updated control points. The knot insertion algorithm is widely used in Isogeometric Analysis as it allows to enrich the approximation space without altering the geometry of the domain. Thus, it can be interpreted as the equivalent of the h-refinement for NURBS representations (1).

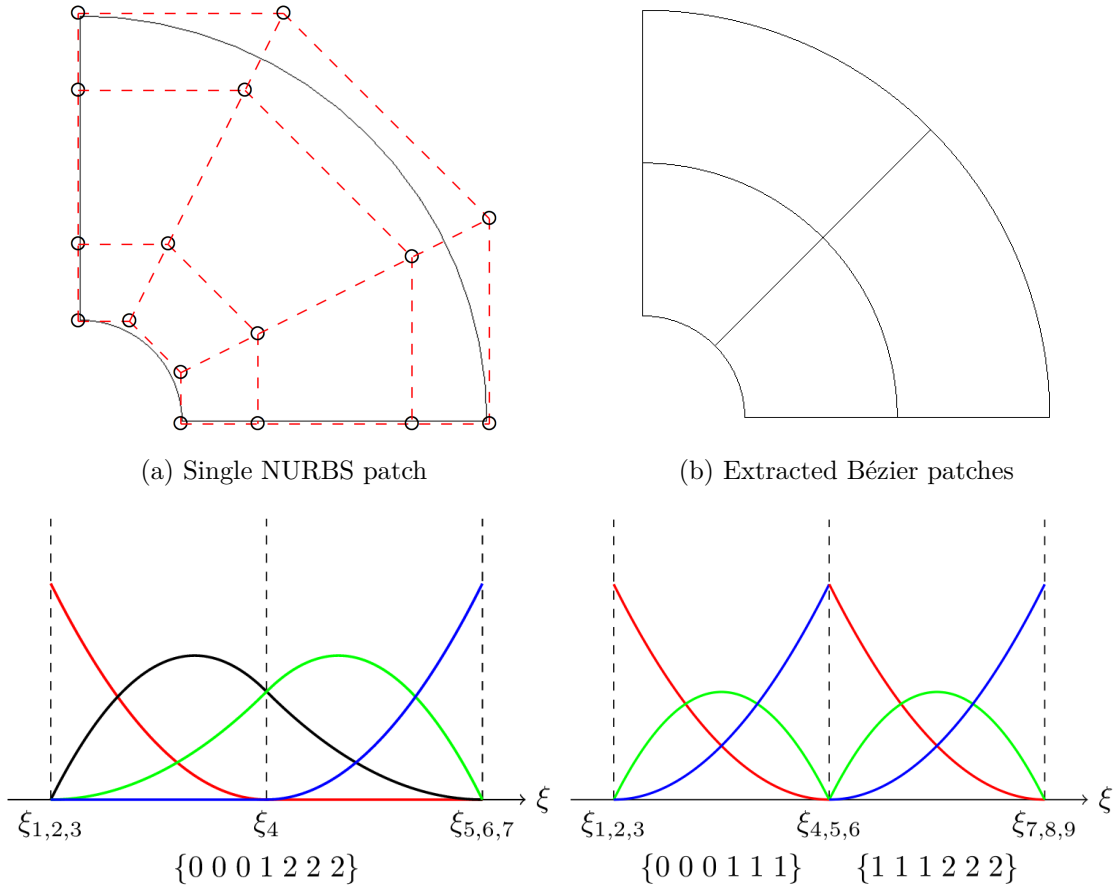
1.2.5 Bézier extraction

Bézier curves can be seen as a special case of NURBS curves defined over a single knot interval. Indeed, it can be shown that Bernstein polynomials of degree p can be expressed as B-Spline basis functions computed using the following knot vector:

$$\Xi = \{\underbrace{0, \ldots, 0}_{p+1}, \underbrace{1, \ldots, 1}_{p+1}\}. \tag{1.22}$$

As a consequence, it is possible to convert a CAD object described using NURBS functions to a set of independent Bézier patches, without altering the geometry. The procedure is known as Bézier extraction.

If we consider a single NURBS curve of degree p, built using a knot vector Ψ containing m non-zero knot spans, a set of m independent Bézier curves is obtained by applying the knot insertion algorithm until the multiplicity of every internal knot of Ψ is equal to p+1. The procedure can be extended to NURBS surfaces, by applying the extraction algorithm to each NURBS curve of the tensor product representation. We report in Fig. 1.5 the extraction of 4 independent Bézier surfaces starting from a single quadratic NURBS patch with knot vector $\{0,0,0,1,2,2,2\}$ in both directions ξ and η.



(a) Single NURBS patch

(b) Extracted Bézier patches

(c) B-Spline basis

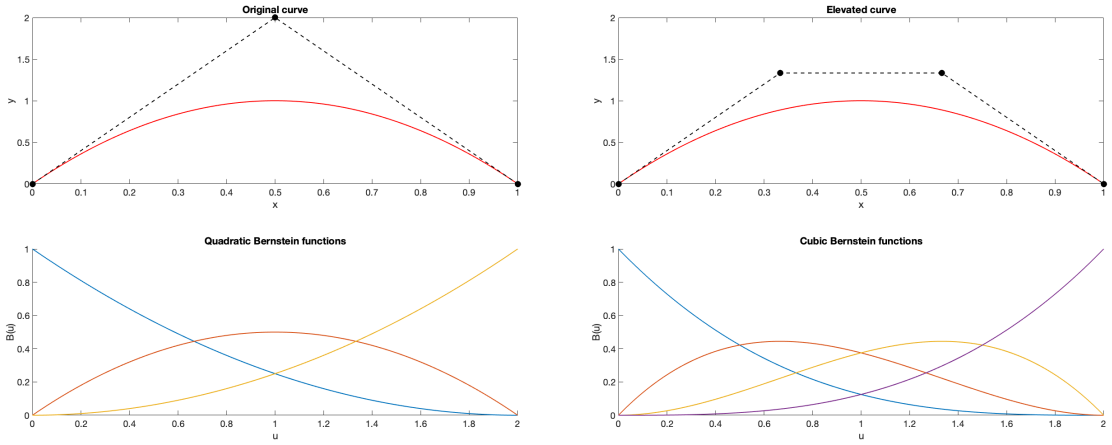
(d) Bernstein basis

Figure 1.5: Bézier extraction

The Bézier extraction algorithm is fundamental as it allows to construct a CAD-consistent DG representation (32). Indeed, traditional Isogeometric Analysis approaches adopt B-Spline and NURBS as a basis for the Galerkin method (1). Since the spline basis is continuous, inside each NURBS patch a CG scheme has to be employed. On the contrary, polynomial and rational Bernstein functions can be employed as a basis for a DG discretization, thanks to the capability of generating discontinuous solutions on the extracted set of Bézier patches.

1.2.6 Degree elevation

Another useful geometry manipulation is the degree elevation, thanks to which it is possible to implement p-refinement techniques for finite element discretizations. Since our numerical scheme relies on Bézier patches, we limit our description to the degree elevation of a Bézier representation. The algorithm for NURBS patches is considerably more complex, due to the spline nature of the basis functions. The reader may refer to Piegl et al. (74) for more details.



(a) Original quadratic curve

(b) Elevated curve

Figure 1.6: Example of degree elevation

Let us consider a rational Bézier curve $\mathbf{C}$ of degree p. Using the homogeneous coordinate notation, the degree elevation problem can be expressed as:

$$\mathbf{C}^{\omega}(\xi) = \sum_{i=1}^{p+1} B_i^p(\xi)\, \mathbf{x}_i^{\omega} = \sum_{i=1}^{p+2} B_i^{p+1}(\xi)\, \bar{\mathbf{x}}_i^{\omega}, \tag{1.23}$$

where $\bar{\mathbf{x}}_i^{\omega}$ represents the control points of the $(d+1)$-dimensional polynomial curve. It could be possible to write the problem as a linear system of equations and solve to obtain the new set of control points $\bar{\mathbf{x}}_i^{\omega}$. However, exploiting the definition of Bernstein polynomials an explicit solution can be obtained (74), and the updated control points are computed thanks to the following formula:

$$\bar{\mathbf{x}}_i^{\omega} = (1-\alpha_i)\mathbf{x}_i^{\omega} + \alpha_i \mathbf{x}_{i-1}^{\omega}, \tag{1.24}$$

with:

$$\alpha_i = \frac{i-1}{p+1}, \quad i = 1, \ldots, p+2. \tag{1.25}$$

Again, the elevated polynomial curve is reverted to a rational representation to obtain the updated set of control points in the physical space.

1.3 Isogeometric DG formulation

As the fundamentals of the CAD representation have been illustrated, we focus in this section on the Isogeometric DG method. We assume that, in general, the physical flux of the considered system of conservation laws can be expressed as:

$$\mathbf{F} = \mathbf{F}_c(\mathbf{W}) - \mathbf{F}_v(\mathbf{W}, \nabla \mathbf{W}) \tag{1.26}$$

where $\mathbf{F}_c$ is the convective flux and $\mathbf{F}_v$ is the viscous flux. $\mathbf{G} = \nabla\mathbf{W}$ is the gradient of the conservative variables. The second order derivatives are discretized with the Local Discontinuous Galerkin (LDG) approach (75). We thus rewrite eq. (1.1) as a system of first order equations:

$$\begin{cases} \dfrac{\partial \mathbf{W}}{\partial t} + \nabla \cdot \mathbf{F}_c(\mathbf{W}) - \nabla \cdot \mathbf{F}_v(\mathbf{W}, \mathbf{G}), \\ \mathbf{G} - \nabla\mathbf{W} = 0. \end{cases} \tag{1.27}$$

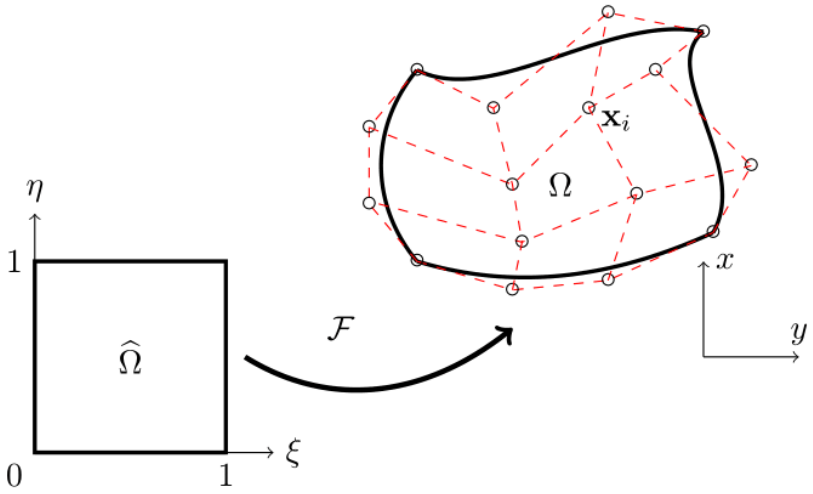


Figure 1.7: representation of a bidimensional Bézier element

In the Isogeometric DG framework, each element is a rational Bézier patch, as represented in figure 1.7. The geometry $\mathbf{x}$ and the local solution fields $\mathbf{w}_h$ and $\mathbf{g}_h$ are expressed using the same basis functions:

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{w}_h \\ \mathbf{g}_h \end{pmatrix} = \sum_{i=1}^{(p+1)^2} R_i(\xi, \eta) \begin{pmatrix} \mathbf{x}_i \\ \mathbf{w}_i \\ \mathbf{g}_i \end{pmatrix}, \tag{1.28}$$

where $R_i(\xi, \eta)$ are the rational Bernstein functions $R^p_{i_1,i_2}(\xi, \eta)$ with a trivial index change and omission of degree p. The isogeometric paradigm is adopted to extend the classic DG scheme (1.4) and discretize eq. (1.27). Using the map defined by the rational Bézier functions on each element Ω_j, the integrals are transposed from the physical space to the parametric unit square $\widehat{\Omega} = [0, 1]^2$, obtaining the Isogeometric DG formulation:

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{w}_i}{\mathrm{d}t} \displaystyle\int_{\widehat{\Omega}} R_k R_i \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} = \int_{\widehat{\Omega}} \nabla R_k \cdot (\mathbf{F}_c - \mathbf{F}_v) \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} - \oint_{\partial\widehat{\Omega}} R_k (\mathbf{F}_c^* - \mathbf{F}_v^*) \left|\mathrm{J}_{\Gamma_j}\right| \mathrm{d}\widehat{\Gamma}, & \text{(1.29a)} \\ \mathbf{g}_i \displaystyle\int_{\widehat{\Omega}} R_k R_i \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} = \int_{\widehat{\Omega}} \nabla R_k \mathbf{w}_h \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} - \oint_{\partial\widehat{\Omega}} R_k \mathbf{W}^* \left|\mathrm{J}_{\Gamma_j}\right| \mathrm{d}\widehat{\Gamma}. & \text{(1.29b)} \end{cases}$$

With respect to the original method, the integrals in (1.29) contain the additional metric terms $|\mathrm{J}_{\Omega_j}|$ and $|\mathrm{J}_{\Gamma_j}|$. In particular, J_{Ω_j} is the Jacobian matrix of the coordinate transformation defined by equation (1.28):

$$\mathrm{J}_{\Omega_j} = \begin{pmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial x}{\partial \eta} \\ \dfrac{\partial y}{\partial \xi} & \dfrac{\partial y}{\partial \eta} \end{pmatrix} = \sum_{i=1}^{(p+1)^2} \begin{pmatrix} \dfrac{\partial R_i}{\partial \xi}(\xi,\eta)\, x_i & \dfrac{\partial R_i}{\partial \eta}(\xi,\eta)\, x_i \\ \dfrac{\partial R_i}{\partial \xi}(\xi,\eta)\, y_i & \dfrac{\partial R_i}{\partial \eta}(\xi,\eta)\, y_i \end{pmatrix}, \tag{1.30}$$

whereas $|\mathrm{J}_{\Gamma_j}|$ is the Jacobian of the map between the reference interval $\widehat{\Gamma} = [0,1]$ and each edge of the elements:

$$|\mathrm{J}_{\Gamma_j}| = \sqrt{\left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2} = \sqrt{\left(\sum_{i=1}^{p+1} \frac{\partial R_i}{\partial \xi}(\xi)\, x_i\right)^2 + \left(\sum_{i=1}^{p+1} \frac{\partial R_i}{\partial \xi}(\xi)\, y_i\right)^2}. \tag{1.31}$$

Furthermore, the gradient of the basis functions on each element Ω_j, in the physical space, is computed by means of J_{Ω_j}:

$$\nabla R_k = \mathrm{J}_{\Omega_j}^{-T} \widehat{\nabla} R_k, \tag{1.32}$$

where $\widehat{\nabla} R_k$ is the gradient of the basis function in the parametric domain. The Gauss-Legendre quadrature rule is employed to approximate the integrals. The convective numerical flux $\mathbf{F}_c^*$ is computed with a consistent Riemann solver. The viscous numerical fluxes $\mathbf{F}_v^*$ and $\mathbf{W}^*$ are computed with the LDG approach (75):

$$\begin{cases} \mathbf{F}_v^* = \mathbf{F}_v(\mathbf{w}_h^+, \mathbf{g}_h^+), \\ \mathbf{W}^* = \mathbf{w}_h^-. \end{cases} \tag{1.33}$$

Equation (1.29b) does not contain a time derivative, therefore it can be solved separately from eq. (1.29a) within each time iteration. The system of equations (1.29) can be thus rewritten in a compact form:

$$\mathcal{M} \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \mathcal{R}(\mathbf{w}_h), \tag{1.34}$$

where the residual $\mathcal{R}$ is the right-hand side of eq. (1.29a) and $\mathcal{M}$ is the block-diagonal mass matrix.

1.4 Equations of fluid mechanics

The applications presented in this work mainly concern flows governed by the two-dimensional Navier-Stokes equations. Using the formalism of equation (1.27), the conservative variables

and the physical fluxes are:

$$\mathbf{W} = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho e \end{pmatrix}, \quad \mathbf{F}_{c,i} = \begin{pmatrix} \rho u_i \\ \rho u_1 u_i + p\delta_{1i} \\ \rho u_2 u_i + p\delta_{2i} \\ \rho u_i (e + \frac{p}{\rho}) \end{pmatrix}, \quad \mathbf{F}_{v,i} = \begin{pmatrix} 0 \\ \tau_{1i} \\ \tau_{2i} \\ u_k \tau_{ki} - q_i \end{pmatrix}, \tag{1.35}$$

where u_i are the components of the fluid velocity vector, ρ is flow density and e is the total energy. The relation between the thermodynamic pressure p and the conservative variables is given by the ideal gas model:

$$\rho e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho(u_1^2 + u_2^2) \tag{1.36}$$

with $\gamma = 1.4$. The viscous stress tensor τ_{ij} and the thermal conduction flux q_i are:

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{ij}, \tag{1.37}$$

$$q_i = -\gamma\frac{\mu}{Pr}\frac{\partial e}{\partial x_i}, \tag{1.38}$$

where $Pr = 0.72$ and μ is the dynamic viscosity coefficient. When Euler equations are considered, the viscous flux $\mathbf{F}_v$ is set equal to 0 and the numerical algorithm just solves eq. (1.29a).

1.5 Shock capturing

Compressible flows are characterized by the presence of shocks. It is therefore critical for a robust numerical algorithm to be able to accurately capture discontinuities. Two approaches are commonly used for DG schemes. The first method consists in introducing an artificial viscosity term in the conservation law (76; 77), whereas the second technique is inspired by Finite Volume slope limiting (78; 79). The approach employed here adapts the original idea presented in (76) to rational elements, and it was tested in the context of the proposed method in (32), on a simple problem with a fixed geometry. A constant by element artificial viscosity ε is added to μ in eq. (1.37), when a shock is detected in the element. Its value is computed with the following smooth function:

$$\varepsilon = \begin{cases} 0 & \text{if } s_j \leq s_0 - \kappa, \\ \dfrac{\varepsilon_0}{2}\left(1 + \sin\dfrac{\pi(s_j - s_0)}{2\kappa}\right) & \text{if } s_0 - \kappa < s_j < s_0 + \kappa, \\ \varepsilon_0 & \text{if } s_j \geq s_0 + \kappa, \end{cases} \tag{1.39}$$

where $\varepsilon_0 = d/(p+1)$, $\kappa = s_0$ and $s_0 = \tilde{s}_0/(p+1)^2$. The shock sensor s_j gives a measure of the oscillatory nature of the solution in each element, and it is based on the total variation of the control points of the density field in the two directions (32):

$$s_\xi^{i_2} = \left(\sum_{i_1=1}^{p} | \rho_{i_1+1,\, i_2} - \rho_{i_1,\, i_2} | \right) - | \rho_{p+1,\, i_2} - \rho_{1,\, i_2} | , \tag{1.40}$$

$$s_\eta^{i_1} = \left(\sum_{i_2=1}^{p} |\rho_{i_1,\, i_2+1} - \rho_{i_1,\, i_2}| \right) - |\rho_{i_1,\, p+1} - \rho_{i_1,\, 1}| . \tag{1.41}$$

The sensor for a given element is finally defined as:

$$s_j = \frac{1}{2(p+1)} \left(\sum_{i_2=1}^{p+1} s_\xi^{i_2} + \sum_{i_1=1}^{p+1} s_\eta^{i_1} \right) \frac{1}{\bar{\rho}} , \tag{1.42}$$

where $\bar{\rho}$ is the average density and $\tilde{s}_0$ is a user-defined parameter to adjust the sensitivity of the shock capturing algorithm.

1.6 Boundary conditions

The treatment of boundary conditions has to be examined in order to conclude the analysis of the space discretisation. The imposition of the boundary conditions in DG schemes is performed by means of the numerical fluxes $\mathbf{F}_c^*$ and $\mathbf{F}_v^*$. Following the terminology adopted by Mengaldo et al. (80), two approaches can be distinguished: *weak Riemann* and *weak prescribed*. In the first technique a numerical flux is computed using the inner conservative variables $\mathbf{w}_{in}$ and a properly computed mirror state $\mathbf{w}_{out}$. In this work, we adopt the weak Riemann approach in three different cases: analytical problems, far field boundaries and slip walls (or flow tangency condition). In the first situation, $\mathbf{w}_{out}$ is computed using the exact solution. When far field conditions are considered, the mirror state $\mathbf{w}_{out}$ is computed by combining the Riemann invariants of the Euler equations evaluated with the inner variables $\mathbf{w}_{in}$ and the known freestream state $\mathbf{w}_\infty$. Lastly, the mirror state for the slip wall condition is defined as:

$$\mathbf{w}_{out} = \begin{pmatrix} \rho_{in} \\ \rho_{in}\, u_{1,m} \\ \rho_{in}\, u_{2,m} \\ \rho_{in}\, e_{in} \end{pmatrix} , \tag{1.43}$$

where $\mathbf{V}_m = (u_{1,m},\, u_{2,m})$ is computed by means of the inner fluid velocity $\mathbf{V}_{in} = (u_{1,in},\, u_{2,in})$ and the boundary velocity $\mathbf{V}_b = (u_{1,b},\, u_{2,b})$:

$$\mathbf{V}_m = \mathbf{V}_{in} - 2\big[(\mathbf{V}_{in} - \mathbf{V}_b) \cdot \mathbf{n}\big]\mathbf{n}. \tag{1.44}$$

In this chapter, the boundary is fixed, therefore $\mathbf{V}_b = 0$. Using eq. (1.44), we force the average between $\mathbf{V}_m$ and $\mathbf{V}_{in}$ to be tangential to the wall. When viscous problems are considered, the weak Riemann approach requires to compute the LDG flux at the boundary. This is done by means of the inner auxiliary state $\mathbf{g}_{in}$:

$$\begin{cases} \mathbf{F}_v^* = \mathbf{F}_v(\mathbf{w}_{in}, \mathbf{g}_{in}), \\ \mathbf{W}^* = \mathbf{w}_{out}. \end{cases} \tag{1.45}$$

Conversely, in the weak prescribed approach, the value of the boundary flux is directly imposed by means of the physics of the boundary condition. We adopt this technique for adiabatic no-slip wall conditions. The convective numerical flux $\mathbf{F}_c^*$ is computed using the convective physical flux $\mathbf{F}_c$ of eq. 1.35:

$$\mathbf{F}_c^* = \mathbf{F}_c(\mathbf{w}_{wall}), \tag{1.46}$$

with the following boundary state:

$$\mathbf{w}_{wall} = \begin{pmatrix} \rho_{in} \\ \rho_{in}\, u_{1,b} \\ \rho_{in}\, u_{2,b} \\ \rho_{in}\, e_{in} \end{pmatrix}. \tag{1.47}$$

Concerning the computation of the LDG numerical flux, $\mathbf{W}^* = \mathbf{w}_{wall}$, whereas, for the evaluation of $\mathbf{F}_v^*$, the viscous stress tensor has to be estimated by means of $\mathbf{g}_{in}$ and the adiabaticity condition is imposed by ignoring the conduction flux vector $\mathbf{q}$.

1.7 Time integration

In order to solve the system of ordinary differential equations (1.34) resulting from the space discretisation, a suitable time integration method has to be adopted. In this manuscript we consider explicit Runge-Kutta (RK) schemes. Considering an initial value problem of the form:

$$\begin{cases} \dfrac{\mathrm{d}z}{\mathrm{d}t} = h(t, z), \\ z(0) = z_0, \end{cases} \tag{1.48}$$

the formulation of a generic explicit RK scheme is:

$$z_{n+1} = z_n + \Delta t \sum_{i=1}^{s} b_i k_i, \tag{1.49}$$

where Δt is the time step, s is the number of stages and k_i:

$$k_i = h\Big(t_n + c_i \Delta t,\, y_n + \Delta t \sum_{j=1}^{i-1} a_{ij} k_j\Big). \tag{1.50}$$

The coefficients a_{ij}, b_i and c_i allow to define RK integrators with different characteristics, especially in terms of numerical stability. It is possible to summarise the coefficients of each RK method using the Butcher tableau:

$$\begin{array}{c|ccccc} 0 & & & & & \\ c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & & \ddots & & \\ c_s & a_{s1} & a_{s2} & & a_{s,s-1} & \\ \hline & b_1 & b_2 & \cdots & b_{s-1} & b_s \end{array} \tag{1.51}$$

In order to achieve high accuracy in time, we employ the classic 4th-order explicit RK method (81), with coefficients:

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array} \tag{1.52}$$

Alternatively, when flows with shocks are considered, we employ the Strong Stability Preserving (SSP) 3rd-order RK method of Gottlieb et al. (82), defined by the following Butcher tableau:

$$\begin{array}{c|ccc} 0 & & & \\ 1 & 1 & & \\ 1/2 & 1/4 & 1/4 & \\ \hline & 1/6 & 1/6 & 2/3 \end{array} \tag{1.53}$$

Lastly, the time step Δt has to be determined. In this work we adopt the following empirical rule:

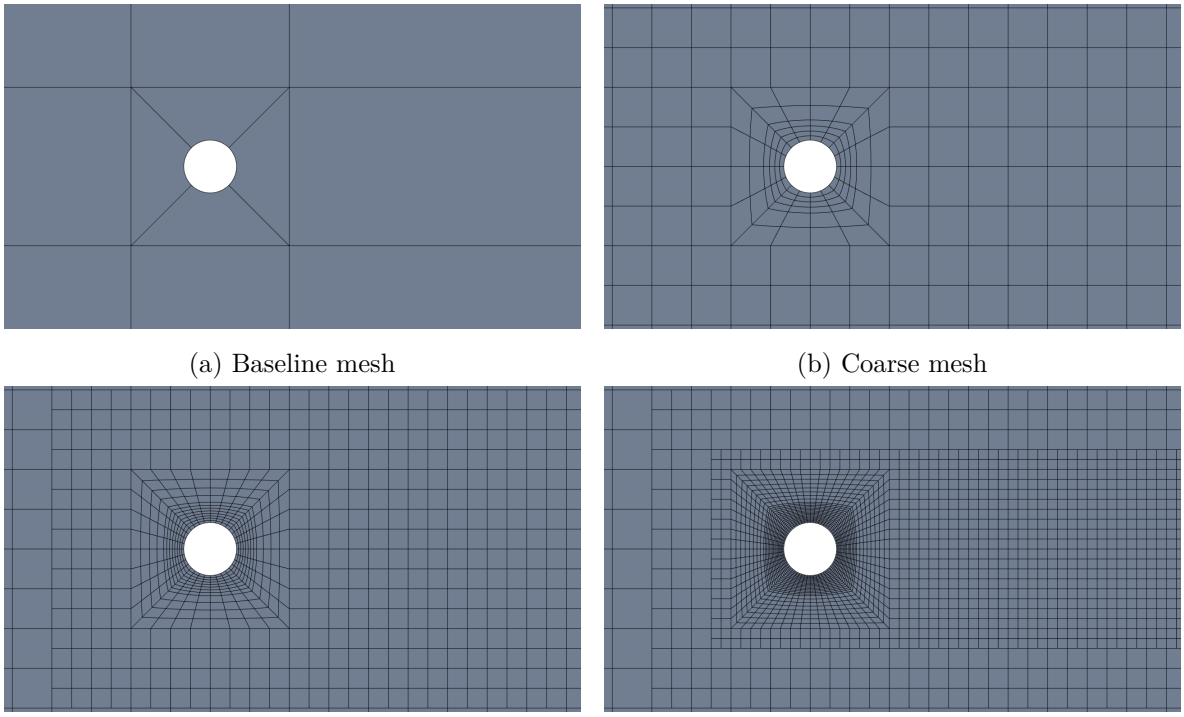
$$\Delta t = \min_i \frac{C}{\frac{\widehat{\lambda}_i}{d_i} + \frac{2\nu}{d_i^2}}, \tag{1.54}$$

where C is a user defined constant, $\widehat{\lambda}_i$ is the maximum wavespeed in the i-th element, and $d_i = 2h_i/(1+2p)$, as proposed by Cockburn et al. (83), with h_i being the characteristic size of the i-th element.

1.8 Cylinder flow

In order to characterise the Isogeometric DG method, we consider a couple of problems with fixed geometries before treating the extension to time-dependent domains. As a first test case, we investigate the laminar flow around a circular cylinder. Thanks to the proposed

formulation, it is possible to exactly represent the computational domain, in fact, the circular boundary can be perfectly described by means of 4 rational Bézier arcs, from which a very coarse baseline grid is generated. Local refinement is then applied to the initial patches to obtain the computational mesh without the necessity of refitting the curved boundary. The cylinder of diameter D is centered at the origin (0,0) and the domain is delimited by the rectangle $[-25D, 100D] \times [-25D, 25D]$, the freestream Mach number is 0.2 and the Reynolds number is equal to 500.



(a) Baseline mesh

(b) Coarse mesh

(c) Intermediate mesh

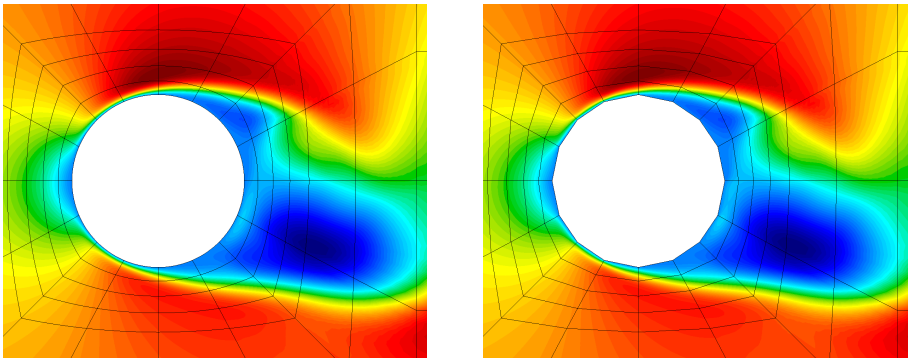
(d) Fine mesh

Figure 1.8: Different refinement levels for the cylinder simulation

Rational shape functions based on polynomials of degrees from 3 to 5 are tested, and, for each degree, the simulations are carried out with 3 different refinement levels:

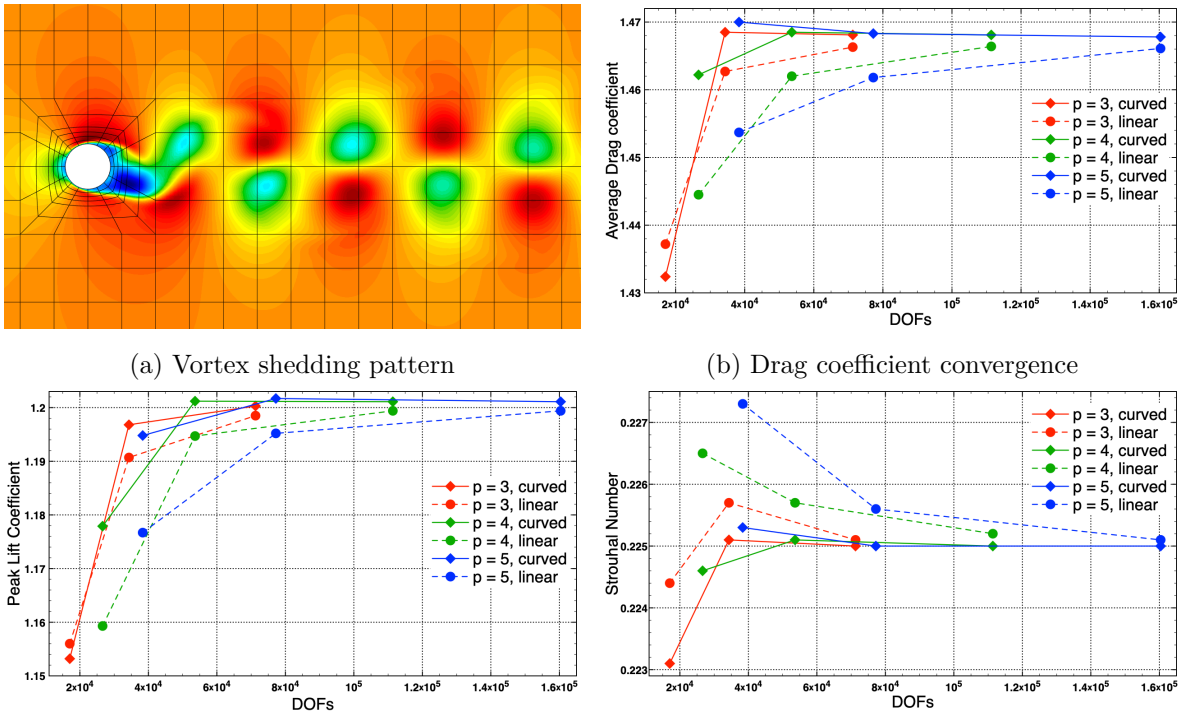
- coarse: 1065 elements, with 16 Bézier arcs on the boundary,
- intermediate: 2145 elements, with 32 Bézier arcs on the boundary,
- fine: 4455 elements, with 64 Bézier arcs on the boundary.

The different mesh levels are represented in Fig. 1.8. For each combination of degree and grid refinement, we compare the results obtained with curved and linear boundaries, as it can be observed in Fig. 1.9, in order to assess the gain given by the NURBS-based representation.



(a) Curved mesh (b) Linear mesh

Figure 1.9: Close-up view of streamwise velocity field, coarse mesh, p = 5



(a) Vortex shedding pattern (b) Drag coefficient convergence

(c) Lift coefficient convergence (d) Strouhal number convergence

Figure 1.10: Cylinder flow, numerical results. The continuous lines represent results obtained with curved boundaries, whereas dashed lines are used for those obtained with linear boundaries.

As we can observe in Fig. 1.10a, the solution is characterized by the well known vortex shedding pattern. The average drag coefficient $\bar{C}_d$, the peak lift coefficient $\widehat{C}_l$ and the Strouhal number St are calculated, and, for each basis degree, we study their evolution over the different refinement levels. In Fig. 1.10, it is possible to observe that, for all the tested degrees, the

same converged solution is found, and, as expected, the convergence is faster for higher order basis functions, but only when curved boundaries are used, indeed, the approximated geometry acts as a source of error, lowering the convergence rate even if the solution field is represented with a high-order basis. In table 1.1 we report the comparison of the Strouhal number obtained with the present methodology with reference data from the literature.

	St
Present work	0.225
Blackburn et al. (84)	0.228
Lu et al. (85)	0.222
Nguyen (52)	0.218
Roshko (Experimental) (86)	0.20 - 0.22

Table 1.1: Comparison of Strouhal number with reference data

1.9 NACA airfoil flow

The inviscid flow around a NACA 0012 airfoil at zero incidence is studied as a second application. A freestream Mach number equal to 0.15 is considered. The shape of the airfoil is defined by:

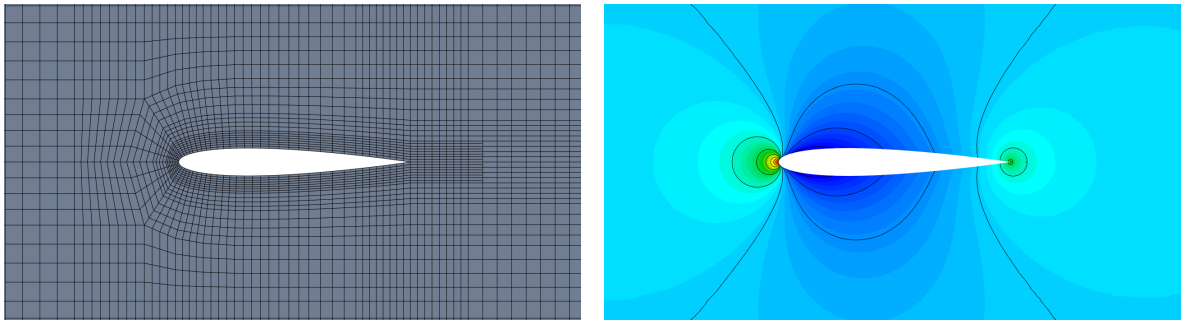
$$y = 5t(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1036x^4), \tag{1.55}$$

with $t = 0.12$ and $x \in [0, 1]$. Using Bernstein polynomials it is possible to exactly represent the boundary by means of the following parameterisation:

$$\begin{cases} x = \xi^2, \\ y = 5t(0.2969\xi - 0.1260\xi^2 - 0.3516\xi^4 + 0.2843\xi^6 - 0.1036\xi^8), \end{cases} \tag{1.56}$$

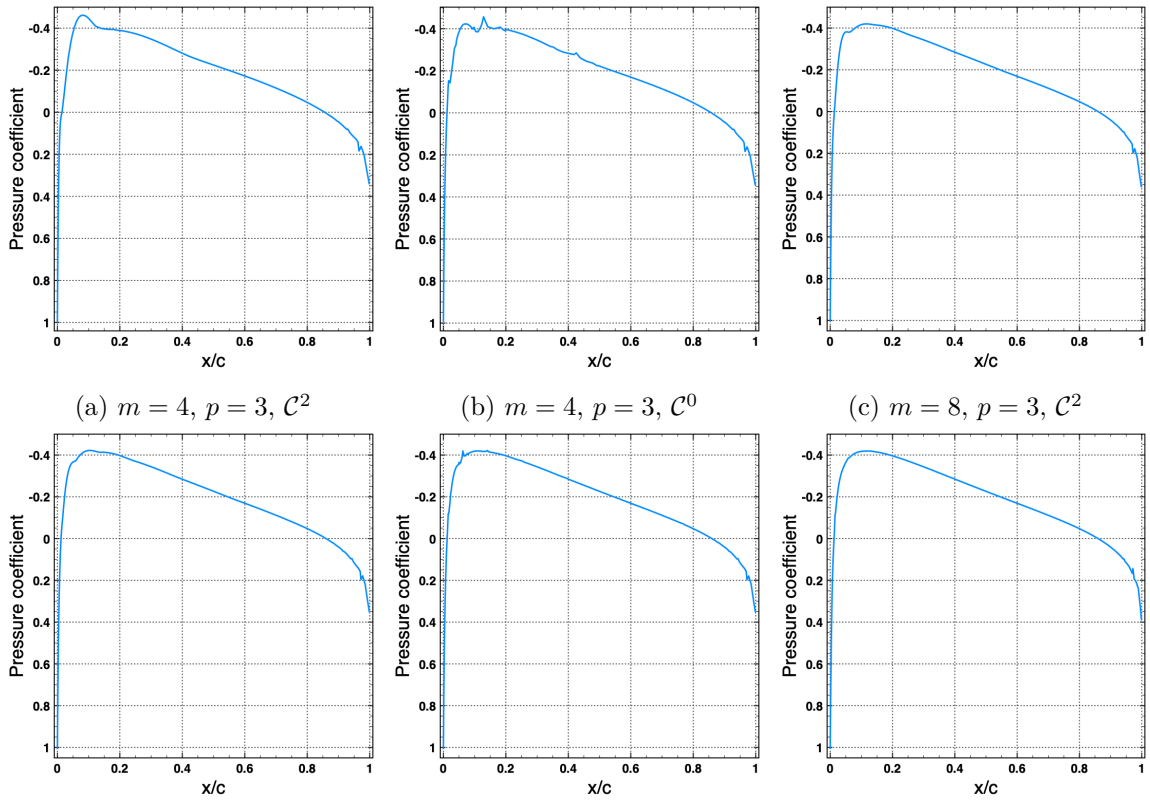
with $\xi \in [0, 1]$. However, adopting a Bézier representation of degree 8 for the numerical scheme would not be practical. As a consequence, we recur to least square B-Spline fitting to represent the boundary. Depending on how the knot vector is chosen, different B-Spline basis can be generated. The goal of the present test case is to better understand the nature of the numerical error introduced by the fitting of the boundary. To this end, we generate a coarse baseline mesh starting from the fitted boundary and we do not refit the geometry during the mesh refinement procedure. The mesh for the simulation is reported in Fig. 1.11 together with the converged density field. As the steady state solution is reached, the curve of the pressure coefficient over the airfoil is computed.

The fitting algorithm is controlled by means of three parameters: the basis degree p, the regularity of the curve and the number of knot spans m, which corresponds to the amount



(a) Mesh (b) Density field

Figure 1.11: Mesh and solution, inviscid NACA airfoil flow



(a) $m = 4$, $p = 3$, $\mathcal{C}^2$ (b) $m = 4$, $p = 3$, $\mathcal{C}^0$ (c) $m = 8$, $p = 3$, $\mathcal{C}^2$

(d) $m = 8$, $p = 3$, $\mathcal{C}^1$ (e) $m = 8$, $p = 3$, $\mathcal{C}^0$ (f) $m = 8$, $p = 4$, $\mathcal{C}^0$

Figure 1.12: Comparison of C_p curve for different choices of fitting

of Bézier elements generated after the extraction procedure. As previously mentioned, the regularity is determined by the multiplicity of the internal knots of the knot vector. The effect of the regularity of the boundary is twofold. On one hand, a smooth representation implies a better regularity of the boundary conditions. On the other hand, for a given value of m, each continuity constraint reduces the degrees of freedom of the curve, leading to a less

accurate fitting. This phenomenon can be observed in Fig. 1.12: C^2 boundary representations present a higher pressure errors with respect to the corresponding C^0 approximation, which are conversely affected by strong oscillations due to the discontinuous boundary derivatives between the elements. It is also possible to observe that using a cubic basis with 4 knot spans is not enough to obtain satisfying results. A significant improvement is noticed using 8 knot spans. Even better results are observed with a quartic basis and 8 knot spans. In this case, the error induced by the geometric approximation is negligible.

1.10 Conclusion

In this chapter we presented the Isogeometric DG framework. We showed how the CAD representation can be manipulated to obtain a DG compatible description. The proposed numerical algorithm has been employed to solve the compressible Navier-Stokes equations and two benchmark problems were analysed. In particular, we investigated the effects of the boundary representation on the flow solution. The results obtained in the proposed test cases allowed to characterise the behaviour of the Isogeometric DG scheme with fixed boundaries. In the following chapter we will thus focus on the extension the Isogeometric DG methodology to problems with time-dependent domains by means of an ALE formulation.

Chapter 2

ALE formulation

ALE algorithms can be classified into two categories: direct methods (54; 20), in which the ALE form of the conservation laws is solved, and indirect methods (55; 58), which are characterised by a pure Lagrangian computation, followed by a rezoning and a remapping phase. The present work is focused on direct ALE approaches. In particular, several ALE formulations have been proposed in the context of DG methods. A review of the possible direct ALE techniques is therefore carried out in section 2.1 and two categories of approaches are identified. In the first family of methods, the conservation laws are effectively solved in the moving domain (20; 52), whereas, in the second class of schemes, the equations are solved in a fixed reference domain thanks to the introduction of a map (51; 47). The main characteristics of the various approaches are discussed and, in particular, the respect or the violation of the Discrete Geometric Conservation Law (DGCL) is examined in each case. For the sake of simplicity, the proposed analysis is limited to method of lines DG discretisations. It is however worth mentioning that interesting results have also been obtained with space-time ALE-DG schemes, either in fully implicit form (59) or, more recently, in the ADER (Arbitrary high-order schemes using DERivatives) framework (54; 61).

The most suitable ALE formulation for NURBS representation is selected, and its extension to rational Bézier elements is detailed in section 2.2. Secondly, we explain how the regularity and hierarchy properties of NURBS can be exploited to define a very smooth mesh deformation algorithm, even in presence of large boundary displacements. Next, we present an effective ALE-AMR coupling technique to perform adaptive simulations with deforming meshes. In section 2.5, a rigorous verification of the methods is presented using two problems with analytic solutions governed by linear advection and compressible Euler equations. Then, three more complex test-cases are considered to assess the capabilities of the proposed methodologies. The compressible Navier-Stokes equations are first solved for an oscillating cylinder and a systematic grid refinement study is carried out to assess the accuracy of the computations and compare the results obtained with reference data from the literature. As

a second problem, we consider the subsonic flow around a pitching airfoil. Here, we focus on quantifying the impact of using curved grids, in comparison with classical piecewise-linear meshes, in order to highlight the gain obtained by improving the geometry description in the context of deformable domains. We then investigate the pitching airfoil flow in transonic conditions, with the goal of assessing the robustness of the developed scheme in presence of shocks. The chapter ends with a demonstration of the ALE-AMR coupling algorithm using the pitching airfoil benchmark.

2.1 ALE formulations for Discontinuous Galerkin

In continuum mechanics, motion is usually described using either the Lagrangian or the Eulerian viewpoint. In the first approach, equations of motion describe the trajectory of every material particle $\boldsymbol{X}$, whose ensemble represents the material domain. In Lagrangian algorithms, each node of the computational grid corresponds to one single material particle, hence, the mesh moves following the same law as the material points. As a consequence, Lagrangian formulations do not present convective terms to discretise, but, as a drawback, they can lead to highly distorted meshes. On the other hand, the Eulerian viewpoint uses a fixed reference frame and the equations of motion describe the evolution of the physical system in each point $\boldsymbol{x}$ of the spatial domain. Since the mesh is fixed, complex motion can be easily simulated with the Eulerian description. The mathematical relation between the two formulations is given by the material derivative:

$$\left.\frac{\partial f}{\partial t}\right|_X = \left.\frac{\partial f}{\partial t}\right|_x + \mathbf{V} \cdot \nabla_x f \tag{2.1}$$

One of the main shortcoming of the Eulerian framework is the difficulty in dealing with moving interfaces and boundaries. Arbitrary Lagrangian-Eulerian methods have been developed to overcome the limitations of the classical viewpoints, introducing a referential domain $\boldsymbol{\chi}$, that moves with an arbitrary velocity $\mathbf{V}_g$, in which the equations of motion are solved. Material, spatial and referential domain are related by the fundamental ALE equation:

$$\left.\frac{\partial f}{\partial t}\right|_X = \left.\frac{\partial f}{\partial t}\right|_\chi + (\mathbf{V} - \mathbf{V}_g) \cdot \nabla_x f \tag{2.2}$$

As shown by Donea et al. (87) and by Venkatasubban (88), using relation (2.2), it is possible to write conservation laws, such as (1.1), into their respective ALE form:

$$\frac{\partial \mathbf{W}}{\partial t} + \nabla \cdot \mathbf{F} - \mathbf{V}_g \cdot \nabla \mathbf{W} = 0, \tag{2.3}$$

where $\partial/\partial t$ is the time derivative in the referential domain.

In a first family of ALE-DG approaches, the weak form of the conservation law is solved in the referential domain. The weak formulation is obtained multiplying eq. (2.3) by a basis function φ_k and integrating over the elemental domain Ω_j:

$$\int_{\Omega_j} \varphi_k \frac{\partial \mathbf{w}_h}{\partial t} \,\mathrm{d}\Omega + \int_{\Omega_j} \varphi_k \nabla \cdot \mathbf{F} \,\mathrm{d}\Omega - \int_{\Omega_j} \varphi_k \mathbf{V}_g \cdot \nabla \mathbf{w}_h \,\mathrm{d}\Omega = 0. \tag{2.4}$$

After integration by parts, one obtains:

$$\int_{\Omega_j} \varphi_k \frac{\partial \mathbf{w}_h}{\partial t} \,\mathrm{d}\Omega + \int_{\Omega_j} \nabla \varphi_k \cdot \mathbf{F} \,\mathrm{d}\Omega - \int_{\Omega_j} \nabla \cdot (\varphi_k \mathbf{V}_g) \mathbf{w}_h \,\mathrm{d}\Omega + \oint_{\partial\Omega_j} \varphi_k (\mathbf{F}^* - \mathbf{V}_g \mathbf{w}^*) \cdot \mathbf{n} \,\mathrm{d}\Gamma = 0, \tag{2.5}$$

where $\mathbf{F}^*(\mathbf{w}_h^+, \mathbf{w}_h^-, \mathbf{n})$ is a consistent Riemann solver and $\mathbf{w}^*(\mathbf{w}_h^+, \mathbf{w}_h^-)$ is the solution of the associated Riemann problem at the element interface. Manipulating equation (2.5) it is possible to derive several ALE-DG schemes, that are equivalent from a continuous point of view, but not at a discrete level. Indeed, constant solutions may not be preserved by the discrete scheme due to the mesh movement, as discussed in (20). In order to ensure an exact preservation of constant solutions, an ALE method has to satisfy the so-called Discrete Geometric Conservation Law (DGCL), which is specific to each numerical scheme, as explained by Guillard et al. (89).

Applying again integration by parts to the second and third terms of eq. (2.5), the ALE-DG scheme of Lomtev et al. (20) is found:

$$\int_{\Omega_j} \varphi_k \frac{\partial \mathbf{w}_h}{\partial t} \,\mathrm{d}\Omega + \int_{\Omega_j} \varphi_k (\nabla \cdot \mathbf{F} - \mathbf{V}_g \cdot \nabla \mathbf{w}_h) \,\mathrm{d}\Omega + \oint_{\partial\Omega_j} \varphi_k \left[\mathbf{F}^* - \mathbf{F} - \mathbf{V}_g (\mathbf{w}^* - \mathbf{w}_h)\right] \cdot \mathbf{n} \,\mathrm{d}\Gamma = 0. \tag{2.6}$$

This formulation can be interpreted as the ALE extension of the strong form DG, presented in (73). Provided that the numerical fluxes $\mathbf{F}^*$ and $\mathbf{w}^*$ are consistent, eq. (2.6) always preserves constant solutions exactly. However, the implementation is substantially different with respect to the more commonly used DG method in weak form. In order to find an alternative formulation, we rewrite eq. (2.5) as:

$$\int_{\Omega_j} \varphi_k \frac{\partial \mathbf{w}_h}{\partial t} \,\mathrm{d}\Omega + \int_{\Omega_j} \varphi_k \mathbf{w}_h \nabla \cdot \mathbf{V}_g \,\mathrm{d}\Omega - \int_{\Omega_j} \nabla \varphi_k \cdot (\mathbf{F} - \mathbf{V}_g \mathbf{w}_h) \,\mathrm{d}\Omega + \oint_{\partial\Omega_j} \varphi_k (\mathbf{F}^* - \mathbf{V}_g \mathbf{w}^*) \cdot \mathbf{n} \,\mathrm{d}\Gamma = 0. \tag{2.7}$$

Introducing $\mathbf{F}^*_{ale} = \mathbf{F}^* - \mathbf{V}_g \mathbf{w}^*$ and applying the Reynolds transport theorem for moving control volumes, the following ALE-DG scheme is found:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_j} \varphi_k \mathbf{w}_h \,\mathrm{d}\Omega - \int_{\Omega_j} \nabla \varphi_k \cdot (\mathbf{F} - \mathbf{V}_g \mathbf{w}_h) \,\mathrm{d}\Omega + \oint_{\partial\Omega_j} \varphi_k \mathbf{F}^*_{ale} \,\mathrm{d}\Gamma = 0. \tag{2.8}$$

From a practical point of view, $\mathbf{F}^*_{ale}(w_h^+, w_h^-, \mathbf{V}_g, \mathbf{n})$ is computed with a numerical flux function that satisfies the modified consistency property:

$$\mathbf{F}^*_{ale}(\mathbf{w}_0, \mathbf{w}_0, \mathbf{V}_g, \mathbf{n}) = \mathbf{F}(\mathbf{w}_0) \cdot \mathbf{n} \; - (\mathbf{V}_g \cdot \mathbf{n})\mathbf{w}_0. \tag{2.9}$$

The formulation (2.8) was first introduced by Nguyen (52) and represents an elegant extension to moving domains of the classic DG method in weak form. One can show that this scheme exactly preserves constant solutions if the following identity is verified at the discrete level:

$$\oint_{\partial\Omega_j} \varphi_k \mathbf{V}_g \cdot \mathbf{n} \, \mathrm{d}\Gamma - \int_{\Omega_j} \nabla\varphi_k \cdot \mathbf{V}_g \, \mathrm{d}\Omega = 0, \tag{2.10}$$

which means that the numerical quadrature has to be sufficiently accurate to exactly compute the terms of the identity. As a consequence, constant solutions may not be exactly preserved for elements of general shape, such as curvilinear elements, and for arbitrary grid velocity fields.

In a second family of ALE approaches the conservation law (1.1) is mapped to a fixed reference domain by means of a function $\mathcal{G}$ that transforms a point $\hat{\mathbf{x}}$ in the reference domain into the corresponding point $\mathbf{x}$ in the physical space: $\mathbf{x}(t) = \mathcal{G}(\hat{\mathbf{x}}, t)$. It has been shown in (90; 51) that the equivalent of equations (1.1) in the reference domain can be written as:

$$\frac{\partial \widehat{\mathbf{W}}}{\partial t} + \widehat{\nabla} \cdot \widehat{\mathbf{F}} = 0, \tag{2.11}$$

where $\widehat{\nabla}$ is the vector of partial derivatives with respect to the reference coordinate frame, and the transformed conservative variables $\widehat{\mathbf{W}}$ and corresponding flux vector $\widehat{\mathbf{F}}$ are defined as:

$$\widehat{\mathbf{W}} = |\mathrm{J}_G| \, \mathbf{W}, \quad \widehat{\mathbf{F}} = |\mathrm{J}_G| \, \mathrm{J}_G^{-1}(\mathbf{F} - \mathbf{V}_M \mathbf{W}), \tag{2.12}$$

where J_G is the Jacobian matrix of the map and $\mathbf{V}_M$ is the map velocity:

$$\mathbf{V}_M = \frac{\partial \mathcal{G}}{\partial t}. \tag{2.13}$$

The mapped ALE technique consists in solving the transformed system of conservation laws (2.11) in the reference domain. The mapped ALE-DG method has been introduced by Persson et al. (51) in the context of compressible fluid mechanics. The reference domain is discretised and the basis functions are defined in the reference space on each element $\widehat{\Omega}_j$. Then, a DG discretization is applied to (2.11), obtaining:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \int_{\widehat{\Omega}_j} \varphi_k \mathbf{w}_h \, |\mathrm{J}_G| \, \mathrm{d}\widehat{\Omega} - \int_{\widehat{\Omega}_j} \widehat{\nabla}\varphi_k \cdot \mathrm{J}_G^{-1}(\mathbf{F} - \mathbf{V}_M \mathbf{w}_h) \, |\mathrm{J}_G| \, \mathrm{d}\widehat{\Omega} \\ + \oint_{\partial\widehat{\Omega}_j} \varphi_k \mathrm{J}_G^{-1} \mathbf{F}^*_{ale} \, |\mathrm{J}_G| \, \mathrm{d}\widehat{\Gamma} = 0, \end{aligned} \tag{2.14}$$

where the map $\mathcal{G}$ is assumed to be $\mathcal{C}^1$. It is worthwhile noting that, even though the mapped ALE approach is significantly different, many similarities can be observed comparing equations (2.14) and (2.8). Due to the presence of the metric terms of the map within the discrete equations, this method violates the DGCL in general. However, as explained in (51), the preservation of constant solutions can be enforced by slightly modifying eq. (2.11) and adding a scalar conservation equation that has to be solved with the same DG scheme, see (51) for more details. Since arbitrary maps can be employed, the mapped ALE approach yields an extremely flexible method and allows high-order domain deformations. On the other hand, computing the map function may not be straightforward when the movement law is not known *a priori*. After this synthesis on ALE methods for DG, we describe in the following section how to adapt one of the analysed schemes to CAD-consistent representations.

2.2 Isogeometric ALE-DG formulation

The rational Bernstein functions (1.15) are defined on a parametric domain, therefore, as in Isogeometric Analysis, the elements are mapped from the physical space to the parametric domain. In this context, a mapped ALE approach would require an additional map function to simulate the moving domain, resulting in a complex and cumbersome implementation. For this reason we opt to solve the equations in the moving referential domain. In particular, the formulation (2.8) is chosen, as the implementation of the strong form ALE-DG scheme (2.6) would represent a major redesign of the framework we illustrated in chapter 1.

In order to adopt the LDG approach, we rewrite eq. (2.3) as a system of first order equations:

$$\begin{cases} \dfrac{\partial \mathbf{W}}{\partial t} + \nabla \cdot \mathbf{F}_c(\mathbf{W}) - \nabla \cdot \mathbf{F}_v(\mathbf{W}, \mathbf{G}) - \mathbf{V}_g \cdot \nabla \mathbf{W} = 0, \\ \mathbf{G} - \nabla \mathbf{W} = 0. \end{cases} \tag{2.15}$$

The isogeometric paradigm is adopted to extend the scheme (2.8) and discretise eq. (2.15). Using the map defined by the rational Bézier functions on each element Ω_j, the integrals are transposed from the physical space to the parametric unit square $\widehat{\Omega} = [0,1]^2$. The isogeometric ALE-DG formulation is thus obtained:

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t}\left(\mathbf{w}_i \displaystyle\int_{\widehat{\Omega}} R_k R_i \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega}\right) = \displaystyle\int_{\widehat{\Omega}} \nabla R_k \cdot (\mathbf{F}_c - \mathbf{F}_v - \mathbf{V}_g \mathbf{w}_h) \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} \\ \qquad\qquad - \displaystyle\oint_{\partial\widehat{\Omega}} R_k (\mathbf{F}^*_{ale} - \mathbf{F}^*_v) \left|\mathrm{J}_{\Gamma_j}\right| \mathrm{d}\widehat{\Gamma}, & \text{(2.16a)} \\ \mathbf{g}_i \displaystyle\int_{\widehat{\Omega}} R_k R_i \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} = \displaystyle\int_{\widehat{\Omega}} \nabla R_k \mathbf{w}_h \left|\mathrm{J}_{\Omega_j}\right| \mathrm{d}\widehat{\Omega} - \displaystyle\oint_{\partial\widehat{\Omega}} R_k \mathbf{W}^* \left|\mathrm{J}_{\Gamma_j}\right| \mathrm{d}\widehat{\Gamma}. & \text{(2.16b)} \end{cases}$$

For the evaluation of the numerical flux function $\mathbf{F}^*_{ale}$, an already existing Riemann solver can be modified to comply with the consistency condition (2.9). It is indeed possible to write

the Jacobian matrix of the convective physical flux in the direction $\boldsymbol{\tau}$ as:

$$\mathbf{J}_{ale} \cdot \boldsymbol{\tau} = \mathbf{J}_F \cdot \boldsymbol{\tau} - (\mathbf{V}_g \cdot \boldsymbol{\tau})\mathbf{I}, \tag{2.17}$$

with $\mathbf{J}_F$ being the Jacobian of the convective flux on the fixed mesh and $\mathbf{I}$ the identity matrix. It is thus trivial to show that the eigenvectors of $\mathbf{J}_{ale}$ are invariant with respect to the grid velocity, whereas for the eigenvalues the following relation holds:

$$S_{ale} = S_0 - \mathbf{V}_g \cdot \boldsymbol{\tau}, \tag{2.18}$$

where S_0 are the eigenvalues computed without mesh movement. It is thus possible to obtain a consistent numerical flux for moving meshes using the modified wave speeds computed with eq. (2.18) and adding the flux contribution generated by mesh movement to the fixed grid Riemann solver. Two examples will be provided in section 2.5. It is important to remark that, since the grid velocity $\mathbf{V}_g$ alters the wave speeds of the equations, the stability condition for explicit time integration depends on $\mathbf{V}_g$. As a consequence, the time step used for the RK integration is a function of the mesh movement.

Equation (2.16b) is identical to its fixed mesh counterpart. The system of equations (2.16) can be rewritten in a compact form:

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathcal{M}\mathbf{w}) = \mathcal{R}(\mathbf{w}_h, \mathbf{V}_g), \tag{2.19}$$

where the residual $\mathcal{R}$ is the right-hand side of eq. (2.16a). The system of ODEs (2.19) is integrated with explicit RK methods. Since the elements are deforming, the mass matrix $\mathcal{M}$ is not constant in time. Therefore, $\mathcal{M}$ has to be updated in each iteration of the RK algorithm. It is however possible to simplify the time integration scheme when the mesh movement is rigid. Indeed, it is trivial to show that the mass matrix is constant in time when the elements are rotating and translating. We thus obtain:

$$\mathcal{M}\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \mathcal{R}(\mathbf{w}_h, \mathbf{V}_g). \tag{2.20}$$

In this case, the mesh movement contributions are limited to the flux terms due to the grid velocity $\mathbf{V}_g$. Integrating equation (2.20) is significantly less expensive from the computational standpoint, as the inverse of the mass matrix is computed in the pre-processing phase.

One can observe that, due to the introduction of the isogeometric map, the mathematical structure of the proposed formulation (2.16) is nearly identical to that of the mapped ALE-DG scheme (2.14). However, in our approach the map is local to each element and it is only used to transform the integrals of the weak formulation. On the other hand, in the technique proposed by Persson et al. (51), the map is global and it is applied to modify the conservation laws at the continuous level. Similarly to eq. (2.14), the proposed scheme (2.16) violates the

DGCL in the general case. Indeed, the equivalent of the identity (2.10) for equation (2.16a) is:

$$\oint_{\partial\widehat{\Omega}} R_k \mathbf{V}_g \cdot \mathbf{n} \, |\mathrm{J}_{\Gamma_j}| \, \mathrm{d}\widehat{\Gamma} - \int_{\widehat{\Omega}} \mathrm{J}_{\Omega_j}^{-T} \widehat{\nabla} R_k \cdot \mathbf{V}_g \, |\mathrm{J}_{\Omega_j}| \, \mathrm{d}\widehat{\Omega} = 0. \tag{2.21}$$

When the basis functions R_k are rational, an exact integration is not achievable by means of Gauss-Legendre quadrature. In the case of uniform weights, R_k is a polynomial, and the products $|\mathrm{J}_{\Gamma_j}|\mathbf{n}$ and $|\mathrm{J}_{\Omega_j}|\mathrm{J}_{\Omega_j}^{-T}$ are polynomials as well. Therefore, the integrals in (2.21) can be exactly computed by means of numerical quadrature. As a consequence the DGCL is satisfied when the weights are uniform. We decide to not enforce the DGCL for rational functions as, in the majority of applications, uniform weights are employed. When it is not the case, the use of rational functions is usually limited to very small regions around the boundary, and polynomial Bézier elements are used in the rest of the computational domain. Moreover, as demonstrated in (51), the preservation of constant solutions is not a necessary condition for stability and high-order accuracy.

2.3 Mesh movement technique

The second key ingredient of an ALE method is the mesh movement algorithm. Traditional ALE approaches for CFD applications are based on rectilinear grids. Therefore, the grid velocity in each element is defined by the velocities of its vertices, usually computed via elastic analogy (91) or graph theory based methods (20). On the contrary, our formulation is based on a high-order geometric representation, enabling more control over the shape of the elements. Therefore, a wider range of mesh velocities can be employed with respect to traditional approaches. We decide to adopt the isogeometric paradigm to define the grid velocity field $\mathbf{V}_g$ in each element:

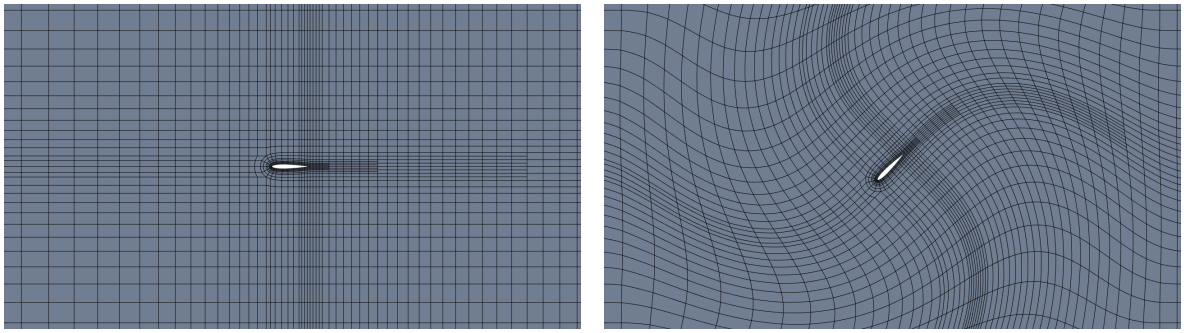
$$\mathbf{V}_g = \sum_{i=1}^{(p+1)^2} R_i(\xi, \eta) \mathbf{v}_{g,i} \, , \tag{2.22}$$

with $\mathbf{v}_{g,i}$ being the velocity of the control point $\mathbf{x}_i$. Obviously, we constrain the velocities to exactly match at the interface between two elements in order to avoid discontinuous movements. This choice leads to a completely unified description of all the variables appearing in the ALE formulation: the solution fields, the geometry and the mesh velocity. Thus, the time evolution of the control point net is determined by:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{v}_{g,i}. \tag{2.23}$$

Given the distribution of control point velocities, the eq. (2.23) is integrated with the same Runge-Kutta method employed for the DG formulation. In the present work, we analyse cases where the mesh movement is imposed in the whole computational domain. We also assume

that no topology change occurs during the movement. The control point velocities are thus explicitly assigned in each time step.



(a) Initial (b) Deformed

Figure 2.1: Mesh movement example

The proposed technique is capable of generating arbitrarily high-order mesh deformations. Moreover, we can take advantage of regularity properties of Bézier surfaces, such as the convex hull property, to preserve the mesh quality over time and avoid tangling. The velocity field $\mathbf{V}_g$ inside each element is always $\mathcal{C}^\infty$ thanks to the definition (2.22), whereas, across the element boundaries, the derivatives can be discontinuous. In order to enhance the regularity at element interfaces, the hierarchical construction of the computational domain can be exploited. As previously explained, the elements are obtained by recursive splitting of a few baseline NURBS patches. When internal knots multiplicities are equal to 1, the inner geometric regularity of these patches is $\mathcal{C}^{p-1}$. The control points velocities $\mathbf{v}_g^0$ of the baseline patches are computed by means of an explicit function of space and time. Thus, the resulting baseline velocity field exhibits also a $\mathcal{C}^{p-1}$ regularity. The control point velocities $\mathbf{v}_g$ of the final DG elements can be obtained by applying the recursive splitting to $\mathbf{v}_g^0$:

$$\mathbf{v}_g = \mathbf{B}\,\mathbf{v}_g^0\,, \tag{2.24}$$

where $\mathbf{B}$ is the matrix that contains the weights of the Bézier extraction and splitting operators. As result, the mesh velocity field $\mathbf{V}_g$ computed with this algorithm possess a $\mathcal{C}^{p-1}$ regularity. Moreover, the hierarchical splitting of the geometrical elements allows to easily deform non-conformal meshes. Indeed, a naive application of any non-linear deformations would generate gaps between elements in the proximity of hanging nodes. The splitting ensures the continuity of the mesh deformation in the proximity of hanging nodes, by imposing the same velocity for the refined and the coarse elements.

Furthermore, the hierarchical definition of the mesh velocity facilitates the avoidance of tangling: thanks to the convex hull property (74), an element remains admissible as long as its control points are not overlaid. By applying the velocity function at the coarsest level,

this admissibility criterion can be easily imposed because the distance between the control points at this level is usually far larger than the displacement. Fig. 2.1 illustrates the mesh deformation for a large amplitude movement. The potential and the flexibility of the proposed mesh movement algorithm are also demonstrated in sections 2.5, 2.6, and 2.7.

2.4 ALE-AMR coupling

Adaptive Mesh Refinement (AMR) techniques allow to optimise the grid at runtime by choosing the appropriate resolution for each region of the computational domain. It is thus interesting to use adaptive meshes for flows around moving and deforming obstacles, where highly unsteady physical phenomena occurs, such as moving shocks or complex vortical wakes. The generation of an optimal grid in these cases heavily relies on user experience and knowledge of the flow physics. Using adaptive meshes is therefore a crucial step towards user-independent unsteady CFD simulations.

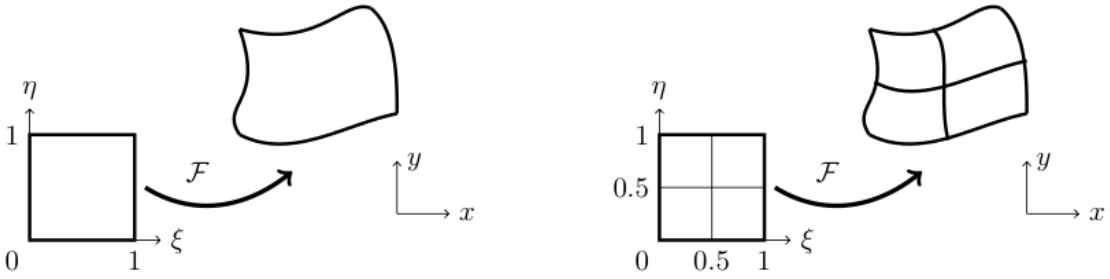


Figure 2.2: Quadtree-like splitting of a Bézier patch

The AMR methodology for the Isogeometric DG has been presented and validated in (33). We summarise here the main features and we then focus on the coupling with the ALE formulation. A quadtree-like approach is adopted to refine the rational Bézier patches. Whenever an element of Level-k is marked for refinement, it is split into 4 child elements of Level-(k+1) by repeatedly inserting knots at $\xi = 0.5$ and $\eta = 0.5$, as in Fig. 2.2. The parent element is stored and it can be recovered if its child elements are tagged for coarsening. Thanks to the isogeometric paradigm, the numerical solution are split using the same knot refinement algorithm used to split the geometry.

In order to select which elements should be refined, an error estimator has to be defined. The natural choice is to exploit the discontinuities of the solution fields due to the DG formulation. Indeed, regions of under-resolution are characterized by strong inter-element jumps. For each element Ω_j, the total jump at the interface between Ω_j and the neighbouring elements is computed:

$$\varepsilon_j = \sum_{k \in \mathcal{N}_j} \int_{\Gamma_{jk}} \| \mathbf{w}_h^+ - \mathbf{w}_h^- \| \, \mathrm{d}\Gamma, \tag{2.25}$$

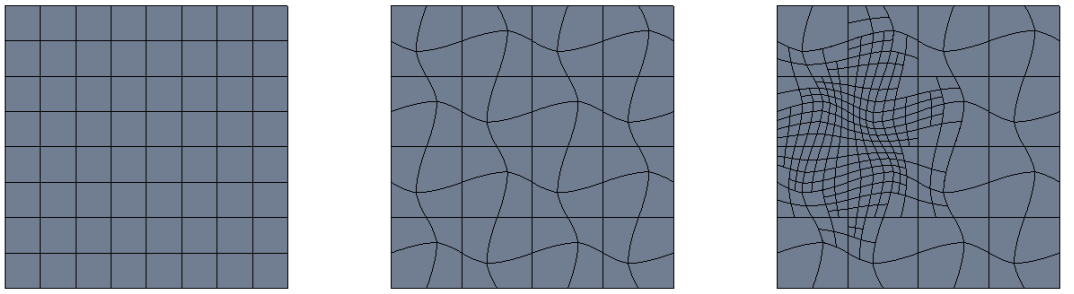
where $\mathcal{N}_j$ represents the set of elements around Ω_j and Γ_{jk} the interface between Ω_j and Ω_k. Then, the element Ω_j is flagged for refinement if:

$$\varepsilon_j > \varepsilon_r \, \bar{\varepsilon}, \tag{2.26}$$

or, similarly, the element is flagged for coarsening if:

$$\varepsilon_j < \varepsilon_c \, \bar{\varepsilon}, \tag{2.27}$$

where ε_c and ε_r are user-defined thresholds and $\bar{\varepsilon}$ is the average value of the indicator ε_j. When the mesh is coarsened, the solution field in the parent element is reconstructed from the data located in the four sons elements. This is achieved by using an average preserving least-squares projection, as detailed in (33).



(a) Level-0 mesh (b) Deformed Level-0 mesh (c) Refined deformed mesh

Figure 2.3: Example of adaptive refinement with mesh deformation

When an adapted mesh is subject to non-linear deformations, the refinement operation might become irreversible. In fact, parent elements are not capable of exactly representing their deformed child elements, yielding possible holes within the mesh. The issue can be avoided by applying the mesh movement to the Level-0 elements and then propagate the velocity field and the deformation to the higher levels of the tree by means of the knot insertion procedure, as illustrated in Fig. 2.3. This can be accomplished thanks to the mesh movement technique presented in section 2.3. The main difference is that, since the refinement is dynamic, the matrix $\mathbf{B}$ of eq. (2.24) is time dependent and has to be recomputed after each step of the AMR algorithm.

2.5 Verification

2.5.1 Advection equation

The proposed approach is firstly tested on the scalar advection equation in ALE form:

$$\frac{\partial w}{\partial t} + (\mathbf{V} - \mathbf{V}_g) \cdot \nabla w = 0, \tag{2.28}$$

where w is the scalar variable and $\mathbf{V}$ is the advection velocity, that we assume constant in space. As numerical flux function we adopt a modified Local Lax-Friedrichs flux that suits the Arbitrary Lagrangian-Eulerian formulation:

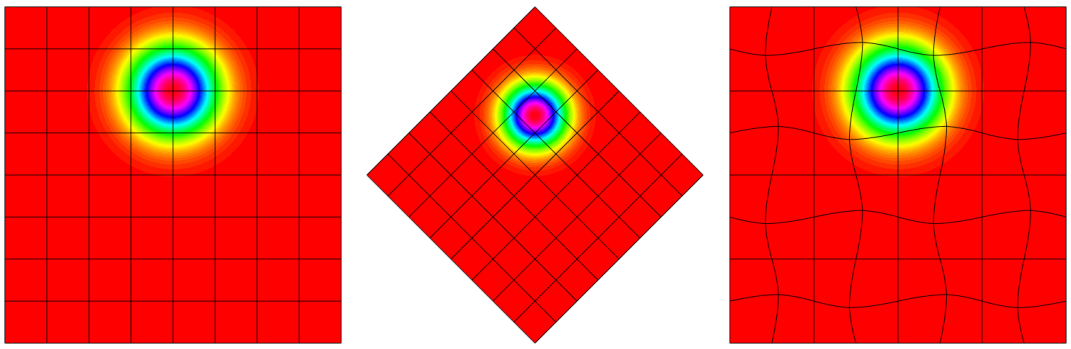
$$\mathbf{F}^*_{ale}(w^+, w^-, \mathbf{V}_g, \mathbf{n}) = \frac{1}{2}(w^+ + w^-)(\mathbf{V} - \mathbf{V}_g)\cdot\mathbf{n} + \frac{1}{2}|(\mathbf{V} - \mathbf{V}_g)\cdot\mathbf{n}|(w^+ - w^-). \tag{2.29}$$

It is straightforward to verify that the proposed numerical flux complies with the consistency condition (2.9). A convergence analysis is performed on the following test case:

$$\begin{cases} w(\mathbf{x}, 0) = \exp\left(-(x-2)^2 - y^2\right), \\ u_1(t) = -4\pi\sin(2\pi t), \\ u_2(t) = 4\pi\cos(2\pi t). \end{cases} \tag{2.30}$$

The initial solution is advected along an anticlockwise circle of radius 2, thus, the analytic solution is:

$$w(\mathbf{x}, t) = \exp\Big(-[x - 2\cos(2\pi t)]^2 - [y - 2\sin(2\pi t)]^2\Big). \tag{2.31}$$

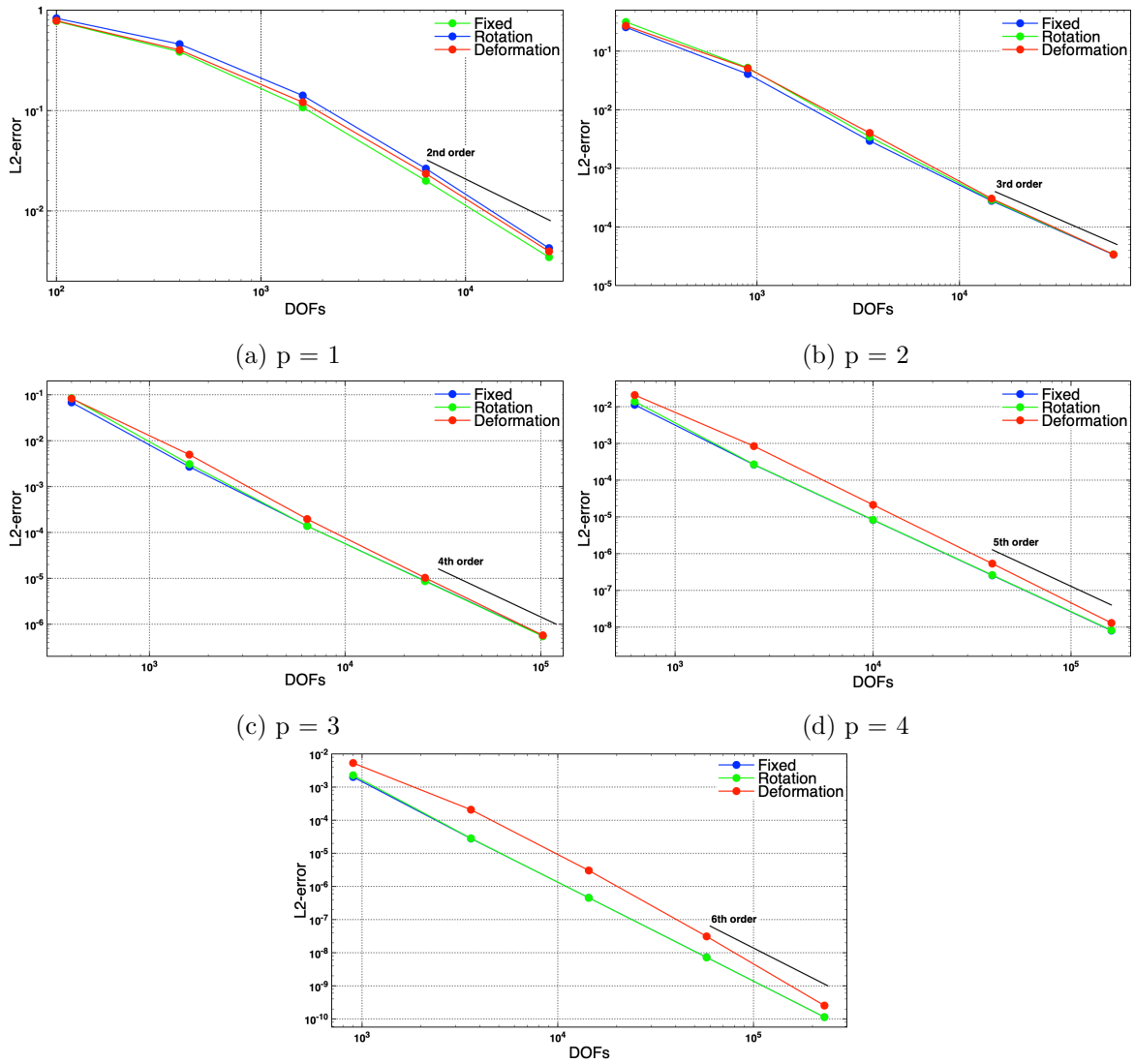


(a) Fixed mesh (b) Rotating mesh (c) Deforming mesh

Figure 2.4: Advection test case, solution at t=0.25, p=5

The computational domain is $[-4, 4] \times [-4, 4]$, discretised with a Cartesian grid, and the boundary numerical fluxes are computed by means of the exact solution. The error in L^2-norm is evaluated using numerical quadrature after one rotation period. In order to assess the accuracy of the numerical scheme, two mesh velocity laws are tested, as illustrated in Fig. 2.4, a rigid clockwise rotation about the origin and a sinusoidal deformation where the control points move with the following velocity:

$$u_g(\mathbf{x}, t) = v_g(\mathbf{x}, t) = \sin\left(\frac{N_x\pi}{L_x}x\right)\sin\left(\frac{N_y\pi}{L_y}y\right)\sin(2\pi t). \tag{2.32}$$



(a) p = 1

(b) p = 2

(c) p = 3

(d) p = 4

(e) p = 5

Figure 2.5: Convergence analysis, 2D advection equation

We compare the results obtained with the 3 configurations. Polynomials up to fifth degree are tested and optimal convergence rates are verified for each degree, as shown in figure 2.5. It is also possible to observe that results on rigidly moving mesh do not substantially differ from the fixed domain simulation, whereas the error increases when the mesh deforms over time, especially for high polynomial degrees. Defining r_d as the ratio between the errors computed respectively on the deforming and the fixed mesh:

$$r_d = \frac{\varepsilon_{deforming}}{\varepsilon_{fixed}}, \tag{2.33}$$

the maximum value of r_d for cubic functions is around 1.8, whereas for quartic polynomials

a factor 3.2 is found in the worst case and for the quintic basis a value of 7.4 is reached. This steep increase is caused by the loss of accuracy of the Bézier approximation on distorted elements. Indeed, as proved by Bazilevs et al. (8), the norm of the truncation error depends on the Jacobian matrix of the isogeometric map.

2.5.2 Euler equations

In order to validate the proposed approach for non-linear problems, we consider the compressible Euler equations for the second test case. The numerical flux is computed using the HLL Riemann solver (92), which has been adapted to take into account the mesh movement flux:

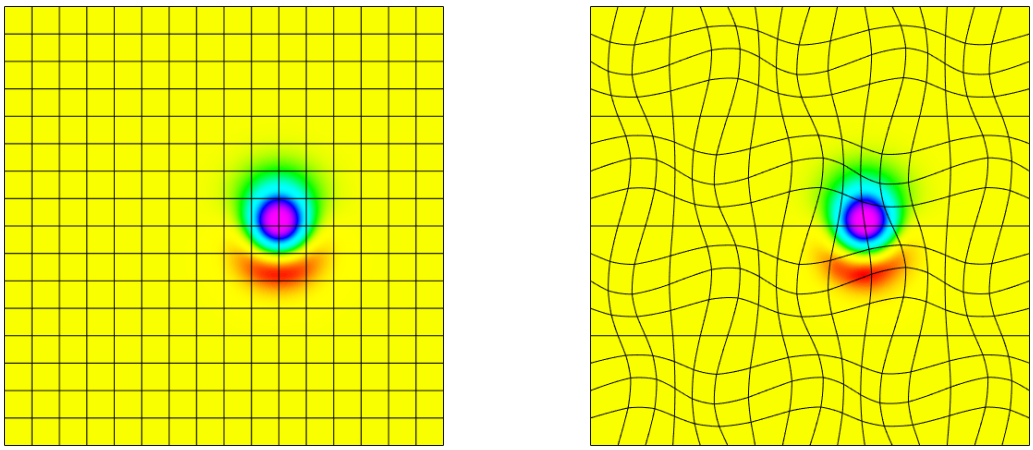
$$\mathbf{F}^*_{ale} = \begin{cases} \mathbf{F}^-_{ale} = \mathbf{F}(\mathbf{w}^-_h) - (\mathbf{V}_g \cdot \mathbf{n})\mathbf{w}^-_h & \text{if } S^-_{ale} \geq 0 \\ \dfrac{S^+_{ale}\mathbf{F}^-_{ale} - S^-_{ale}\mathbf{F}^+_{ale} + S^+_{ale}S^-_{ale}(\mathbf{w}^+_h - \mathbf{w}^-_h)}{S^+_{ale} - S^-_{ale}} & \text{if } S^-_{ale} \leq 0 \leq S^+_{ale} \\ \mathbf{F}^+_{ale} = \mathbf{F}(\mathbf{w}^+_h) - (\mathbf{V}_g \cdot \mathbf{n})\mathbf{w}^+_h & \text{if } S^+_{ale} \leq 0 \end{cases} \tag{2.34}$$

where S^-_{ale} and S^+_{ale} are respectively the minimum and the maximum wave speeds computed with the two states $\mathbf{w}^-_h$ and $\mathbf{w}^+_h$. The proposed numerical flux function (2.34) respects the consistency condition (2.9). The advection of an isentropic vortex (73) is considered, with the following analytic solution:

$$\begin{cases} \rho = \left(1 - \dfrac{\gamma - 1}{16\gamma\pi^2}\beta^2 e^{2(1-r^2)}\right)^{\frac{1}{\gamma-1}} \\ u_1 = 1 - \beta\dfrac{y - y_0}{2\pi}e^{1-r^2} \\ u_2 = \beta\dfrac{x - t - x_0}{2\pi}e^{1-r^2} \\ p = \rho^\gamma \end{cases} \tag{2.35}$$

where $r = \sqrt{(x - t - x_0)^2 + (y - y_0)^2}$, with $x_0 = 5$, $y_0 = 0$ and $\beta = 5$. The computational domain is $[0, 10] \times [-5, 5]$ and the boundary conditions are weakly imposed using the analytic solution. The simulation is run on a fixed Cartesian grid and on a deforming mesh, whose control points follow the sinusoidal law (2.32) used for the advection test case.

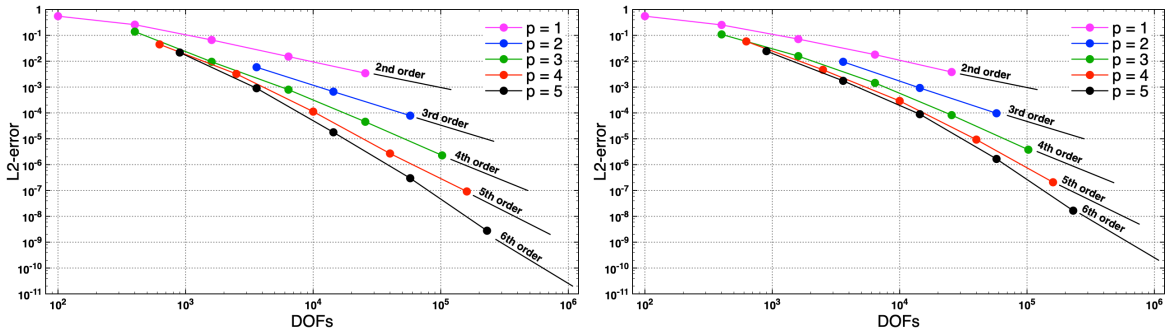
We compute the L^2-norm of the error of the total energy at time $t = 2$. Polynomials up to fifth degree are tested. Optimal convergence rates are verified in both cases for each degree as it is shown in Fig. 2.7, where it is also possible to compare the error levels between the two setups. The evolution of the ratio r_d (2.33) confirms the results obtained for the advection equation: for cubic polynomials the maximum ratio is around 1.8, whereas for the quartic basis a factor 3.5 is found in the worst case and for quintic functions a value of 6 is reached. The sensitivity with respect to mesh deformation is a natural consequence of the



(a) Fixed mesh (b) Deforming mesh

Figure 2.6: Isentropic vortex, energy, t=1.25, p=3

truncation error, as already evidenced for the advection problem. The proposed numerical scheme is hence capable of accurately solving non-linear conservation equations on moving meshes, preserving the high-order convergence rate of the DG discretization.



(a) Fixed mesh (b) Deforming mesh

Figure 2.7: Convergence analysis, Euler equations, L^2-error of total energy

2.6 Oscillating cylinder

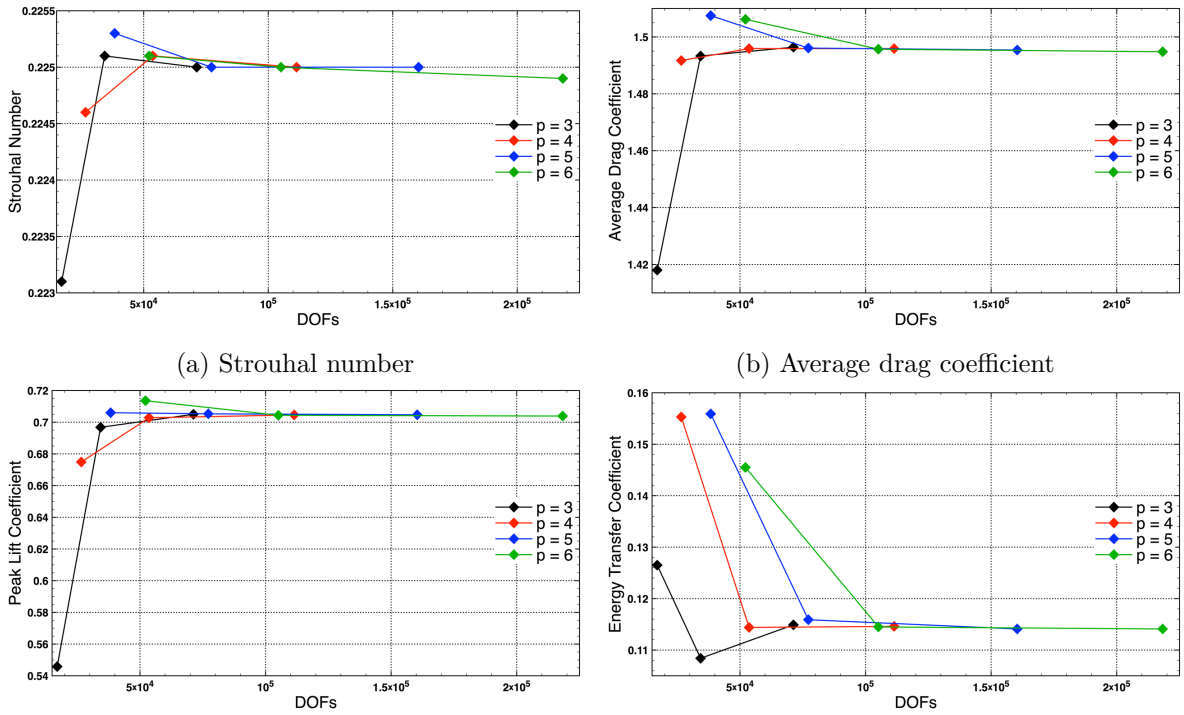
In the next example we simulate the bidimensional viscous flow around a circular cylinder oscillating in the crossflow direction. This case study has been widely investigated in the literature, using both experimental and numerical techniques (52; 93; 84). It is characterised by complex non-linear physics, such as the lock-in phenomenon, that consists in the synchronisation of the vortex shedding with the oscillation of the body. Due to its non-linear nature, the problem is extremely sensitive to acoustic perturbations, therefore, it is important to avoid

the interference between the acoustic waves possibly reflected at the boundaries and the flow in the vicinity of the cylinder. This is achieved by choosing a sufficiently large computational domain and a layer of coarse cells in the proximity of the external boundaries. Moreover, the implemented far-field boundary conditions reduce spurious reflection effects thanks to the use of Riemann invariants. The cylinder surface is modelled as an adiabatic no-slip wall and a weakly prescribed boundary condition is employed. The initial state corresponds to uniform fields, whose values are determined according to the freestream Mach number.

We perform a convergence analysis with the grids presented in chapter 1, using rational functions built from polynomials of degrees from 3 up to 6. We consider a freestream Mach number equal to 0.2, in order to avoid strong compressibility effects, and a Reynolds number of 500 with respect to the diameter. The cylinder motion is given by:

$$y = A\sin(2\pi f t), \tag{2.36}$$

with $A = 0.25D$ and $f = 0.875 f_{sh}$, with f_{sh} being the vortex shedding frequency of the fixed cylinder at equal Mach and Reynolds numbers. The chosen configuration of A and f lies in the lock-in range. For each combination of degree and refinement level the oscillation frequency f is determined from the value of f_{sh} obtained with the corresponding fixed mesh simulation.



(a) Strouhal number

(b) Average drag coefficient

(c) Peak lift coefficient

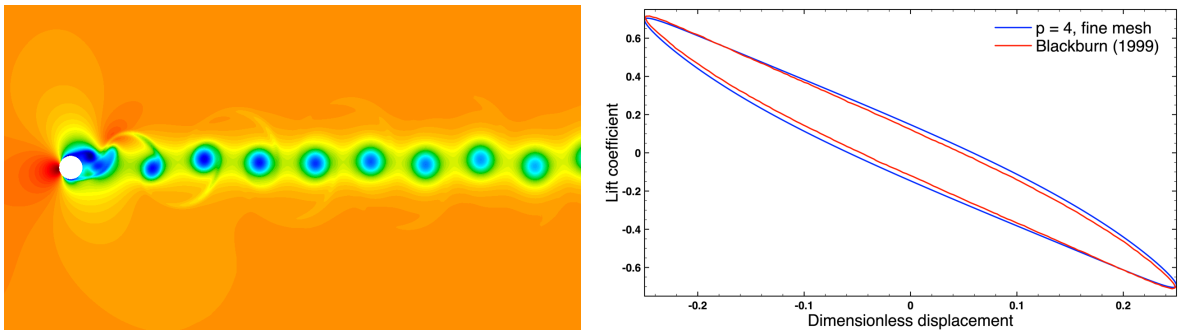
(d) Energy transfer coefficient

Figure 2.8: Oscillating cylinder flow, convergence study

A first set of results, shown in Fig. 2.8, is obtained by running the test case with a rigid mesh movement law, in which the entire grid moves at the velocity of the cylinder. Once a periodic solution is established, we examine more precisely the peak lift coefficient $\widehat{C}_l$, the average drag coefficient $\bar{C}_d$, and the energy transfer coefficient E, which quantifies the mechanical work done by the fluid on the cylinder over a lock-in cycle:

$$E = \frac{1}{\frac{1}{2}\rho_\infty u_\infty^2 D^2} \oint \mathbf{f} \cdot \mathrm{d}\mathbf{x} = \oint \frac{L}{\frac{1}{2}\rho_\infty u_\infty^2 D} \mathrm{d}\Big(\frac{y}{D}\Big) = \oint C_l \, \mathrm{d}\alpha, \tag{2.37}$$

where $\mathbf{f}$ is the force vector, $\mathbf{x}$ the displacement, L the lift force and α the dimensionless displacement in the y direction. The convergence of the Strouhal number of the fixed cylinder is illustrated for the sake of completeness. The lock-in limit cycle is well reproduced with all the basis degrees and a faster convergence is observed with functions of degree 4,5 and 6. Note that using high-order bases on the coarsest mesh does not lead to significant improvements. In order to estimate E with sufficient precision, it is necessary to use at least the intermediate refinement level. Hence, increasing the basis degree on coarse grids becomes inefficient, indeed, the increased number of quadrature points and the severe stability restrictions imposed by the explicit time stepping significantly raise the computational costs without benefits in terms of accuracy. Therefore, a combination of h and p refinements has to be used in order to optimise the computational cost, for a given level of accuracy. The numerical solution of the lock-in limit cycle is compared to the results provided by Blackburn et al. (84). The comparison is presented in Fig. 2.9b, the lift-displacement curve obtained with the proposed scheme is in close comparison with the reference cycle.



(a) Density, p=4, fine mesh

(b) Lock-in limit cycle

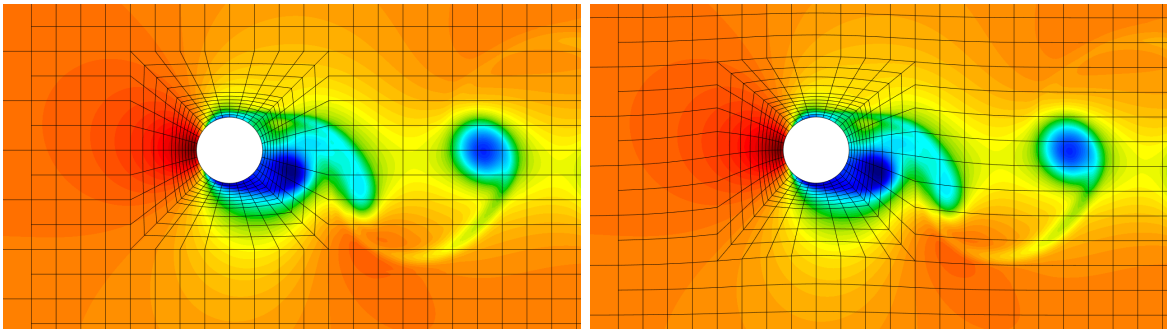
Figure 2.9: Numerical solution of the oscillating cylinder test case

The proposed test case is also used to evaluate the influence of the grid movement law in the presence of moving walls. Therefore, the results obtained with rigid mesh displacements are compared to the simulations performed with a smooth deformation velocity field, where the control point net moves with the following law:

$$v_g(\mathbf{x},t) = v_c(t)\, e^{-\frac{\psi^2(\mathbf{x},t)}{d^2}} \tag{2.38}$$

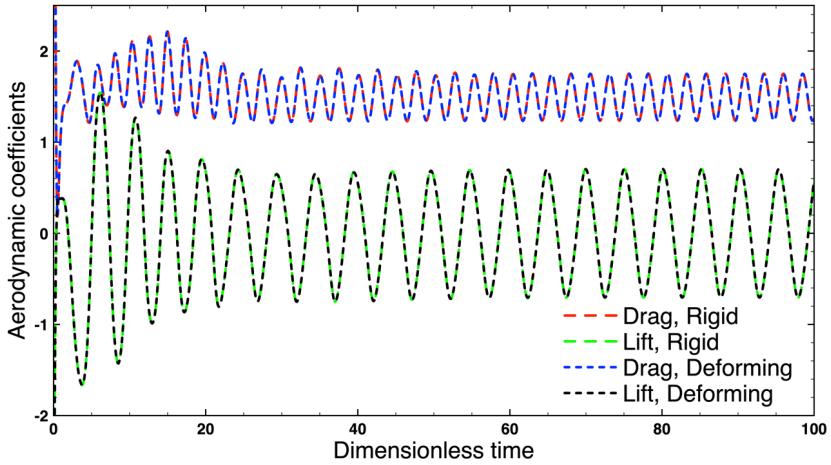
where $v_c(t)$ is the velocity of the cylinder, $\psi(\mathbf{x},t)$ is the distance of the point $\mathbf{x}$ from the cylinder surface at time t and d is a characteristic length, that controls the propagation of the boundary displacement within the domain. As shown in Fig. 2.10b, this profile allows the elements close to the wall to move almost rigidly, preserving the initial mesh quality. At the same time, thanks to the exponential decay far from the cylinder, only a small portion of the domain is deformed.

In Fig. 2.10a and 2.10b the comparison of the density field in the periodic regime for the rigid and deforming mesh movement laws is presented for the intermediate mesh level and quintic basis. The two solutions are nearly identical and, for the deforming mesh, no spurious effects generated by the lack of freestream preservation can be observed, even if the numerical quadrature of rational functions is less accurate. The same conclusion is obtained by comparing the evolution of the aerodynamic coefficients, reported in Fig. 2.10c for the same mesh and basis configuration.



(a) Density field, rigid mesh

(b) Density field, deforming mesh



(c) Force coefficients

Figure 2.10: Comparison of movement laws, p=5, intermediate mesh

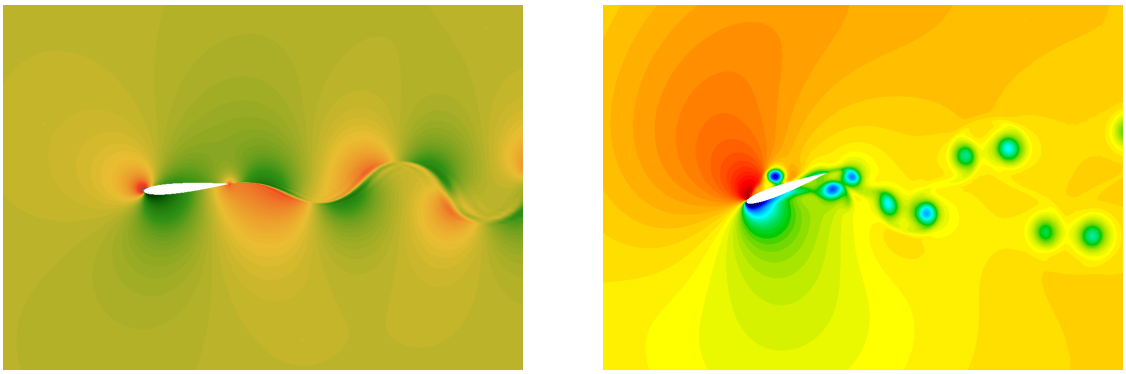
An extensive comparison of the two mesh movement techniques has been carried out, for all the combination of meshes and basis degrees. We report in table 2.1 the energy transfer coefficient, as this quantity is more sensitive than the force coefficients. For a given basis degree, a discrepancy in the third significant figure can be observed for coarse meshes. However, both values are significantly far from the converged result. Refining the grid, the difference between the two movement techniques becomes smaller and the predicted values are equally accurate.

	Rigid			Deformation		
degree	coarse	intermediate	fine	coarse	intermediate	fine
3	0.1265	0.1084	0.1149	0.1298	0.1086	0.1151
4	0.1553	0.1144	0.1146	0.1566	0.1150	0.1148
5	0.1559	0.1159	0.1141	0.1590	0.1154	0.1141
6	0.1455	0.1145	0.1141	0.1422	0.1145	0.1141

Table 2.1: Energy transfer coefficient for different mesh movement laws

2.7 Pitching airfoil in subsonic flow

In the second case study, the subsonic flow around a pitching NACA 0012 airfoil is investigated. We employ a cubic Bernstein basis for both the geometry and the solution fields. The goal of the proposed test case is to assess the impact of using curved grids. This is achieved via a double convergence study: the first is performed using the high-order boundary representation, whereas a piecewise linear approximation is used for the second. For each refinement level, the low-order grid is obtained by linearizing the respective curved mesh. Both inviscid and viscous flow models are considered, as shown in Fig. 2.11.



(a) Inviscid flow, streamwise momentum

(b) Laminar flow, density

Figure 2.11: Pitching airfoil, solution fields

The airfoil is subject to a pure pitching motion about mid-chord. The angle of attack oscillates in time with a sinusoidal law:

$$\alpha(t) = -A\sin(2\pi f t), \tag{2.39}$$

with A and f being respectively the pitch amplitude and frequency. The rotation of the airfoil is transferred to the mesh through a piecewise defined velocity field:

$$\begin{cases} u_g(\mathbf{x},t) = -\tilde{\omega}\,(y - y_c) \\ v_g(\mathbf{x},t) = \tilde{\omega}\,(x - x_c) \end{cases} \tag{2.40}$$

where (x_c, y_c) is the mid-chord position and $\tilde{\omega} = \dot{\alpha}\,\sigma(\mathbf{x},t)$, with σ being a blending function of the following form:

$$\sigma(\mathbf{x},t) = \begin{cases} 1, & \text{if } R \le R_{int} \\ \dfrac{1}{2}\left[1 + \cos\left(\pi \dfrac{R - R_{int}}{R_{ext} - R_{int}}\right)\right], & \text{if } R_{int} < R < R_{ext} \\ 0, & \text{if } R \ge R_{ext} \end{cases} \tag{2.41}$$

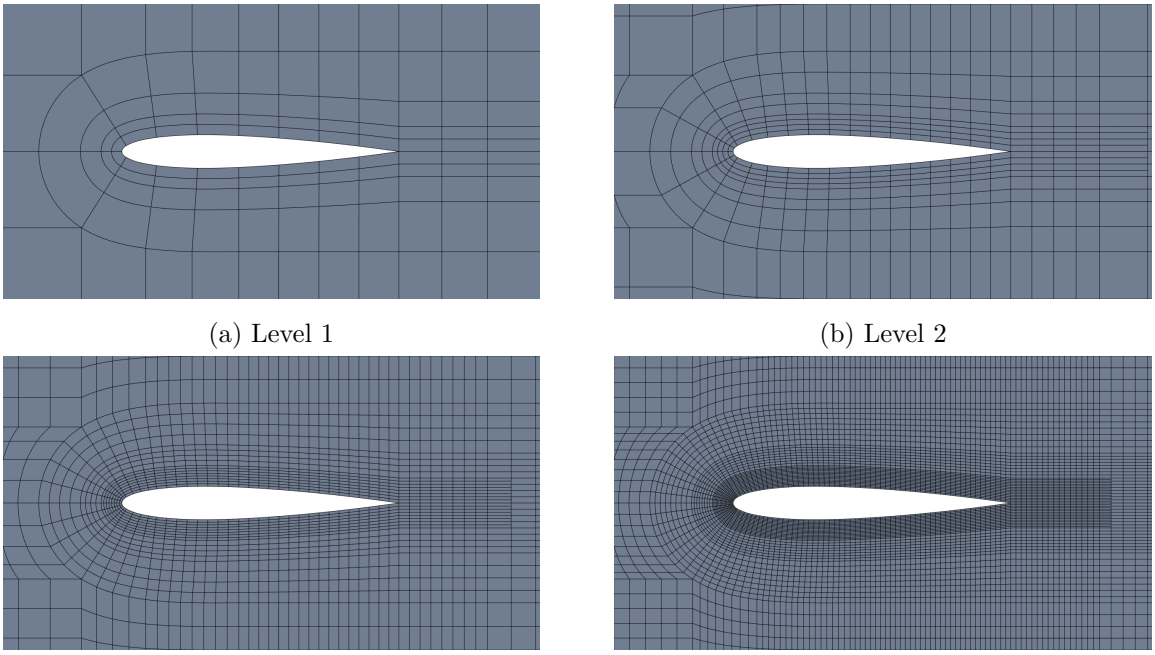
with R the distance with respect to the mid-chord position. The blending function divides the domain into 3 regions, the internal one, delimited by a circle of radius R_{int}, moves rigidly with the airfoil, while the area outside the circle of radius R_{ext} is fixed. The 2 regions are connected by a deforming transition ring. The proposed function (2.41) is C^1, in order to have a smoother deformation profile, however a C^0 class function could be employed as well.

The proposed problem is first studied using an inviscid flow model, based on the compressible Euler equations, and then with a viscous model, based on compressible Navier-Stokes equations. In both cases, the initial state corresponds to uniform fields, defined according to the freestream Mach number. For the inviscid example, the slip wall boundary condition is adopted, whereas, the adiabatic no-slip wall is employed in the viscous case.

2.7.1 Inviscid flow

We first consider an inviscid flow configuration, the freestream Mach number M_∞ is set to 0.2, a pitch amplitude of 5° is considered, with a reduced frequency $k = \pi f c/U_\infty$ of 0.25, where c is the chord length and U_∞ the freestream flow velocity. The streamwise momentum field ρu is illustrated in Fig. 2.11a. The main physical feature is the slip line that develops from the trailing edge of the airfoil due to the pitching motion. Once the transitory effects have vanished, the solution converges to a periodic flow regime, it is hence possible to compute an average drag coefficient $\bar{C}_d$ and a peak lift coefficient $\widehat{C}_l$. The meshes adopted for the inviscid test case are reported in Fig. 2.12.

In Fig. 2.13 we compare the results obtained using the high-order and piecewise linear grids. In particular, the density fields are presented in Fig. 2.13a and 2.13b for the coarsest



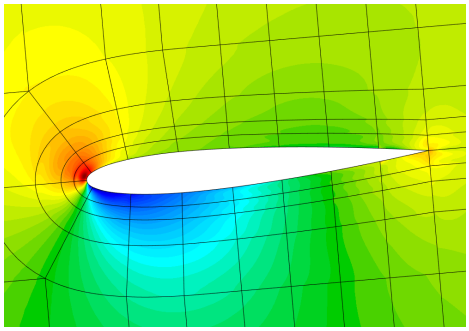
(a) Level 1

(b) Level 2

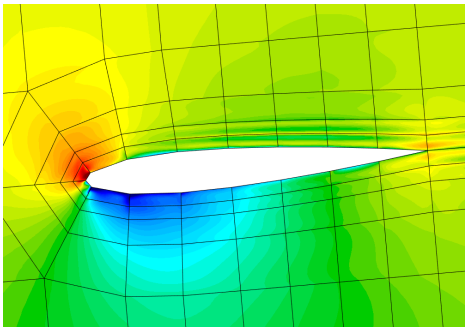
(c) Level 3

(d) Level 4

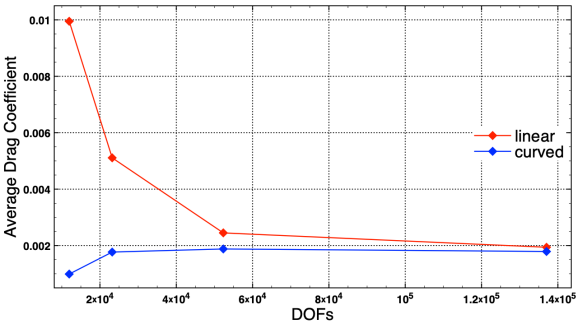
Figure 2.12: Meshes for the inviscid test case



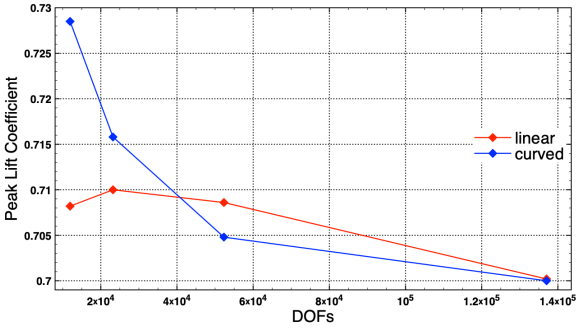
(a) Density field, curved mesh



(b) Density field, linear mesh



(c) Convergence of average drag coefficient



(d) Convergence of peak lift coefficient

Figure 2.13: Influence of the geometry, inviscid flow

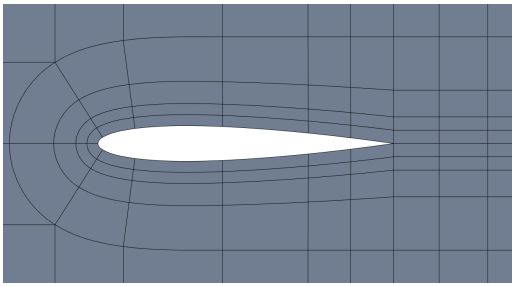
mesh. The spurious effects introduced by the piecewise linear mesh can be easily noticed, non-physical expansions develop at each boundary vertex, generating further numerical oscillations. On the contrary, the high-order mesh guarantees a physically coherent solution even when the mesh is very coarse. The strong influence of the geometry description is caused by the flow-tangency boundary condition, which depends on the normal vector. For piecewise linear approximations, the normal vector is piecewise constant, therefore, across each element the boundary condition presents a discontinuity that creates the non-physical oscillations. This behaviour has already been evidenced in (27; 32). The observations are confirmed by the convergence study, illustrated in Fig. 2.13c and 2.13d for $\bar{C}_d$ and $\widehat{C}_l$. The aerodynamic coefficients converge considerably faster when the curved boundary representation is adopted. Moreover, the error decreases monotonically for the high-order mesh.

2.7.2 Laminar flow

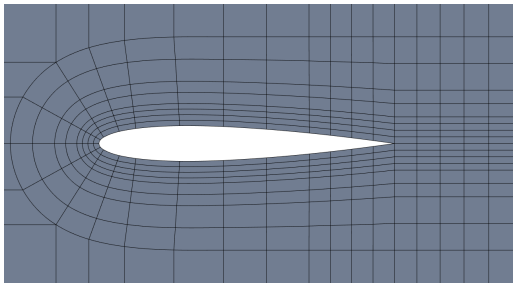
In the second part of the case study, we repeat the convergence analysis for a viscous fluid. The Reynolds number with respect to the chord length is equal to 1000. The freestream Mach number and the pitching frequency are unchanged, whereas the pitch amplitude is increased to 20^o, in order to generate a massively separated flow. Indeed, the considered setup leads to a dynamic stall of the airfoil with a periodic shedding of vortex pairs, as shown in Fig. 2.11b, where the density field is represented. The meshes adopted for the laminar test case are reported in Fig. 2.14.

In Fig. 2.15a and 2.15b it is possible to compare the density fields computed with the high-order and the linearized mesh. Despite the low grid resolution, only very small discrepancies between the two solutions can be spotted, on the suction side of the airfoil close to the leading edge, but, overall, the two fields are very similar. The aerodynamic forces obtained with the linearized geometry are comparable to those computed using the curved boundary too. Even for the coarsest mesh, the values of the aerodynamic coefficients are nearly identical, as reported in Fig. 2.15c and 2.15d.

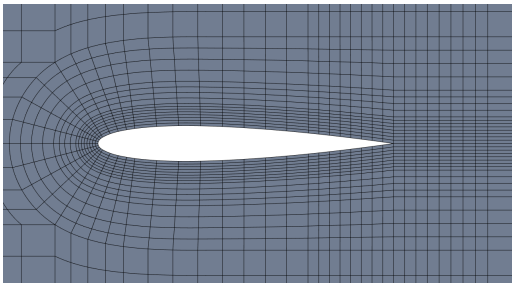
With respect to the inviscid configuration, the high-order boundary representation seems to play a smaller role. This effect is caused by the different nature of the no-slip boundary condition, which is solely a function of the position. In contrast to the flow tangency condition used for Euler equations, the no-slip boundary condition is continuous across each element, therefore it introduces less numerical error. Furthermore, the physical viscosity regularizes the solution field, damping the artificial oscillations. It is thus possible that the influence of the boundary representation increases for higher Reynolds numbers, also due to the development of thin boundary layers and turbulent phenomena. However, such an investigation is beyond the scope of the present manuscript.



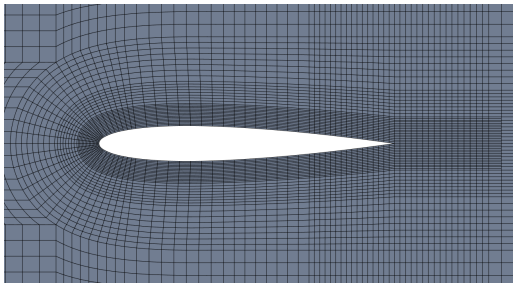
(a) Level 1



(b) Level 2

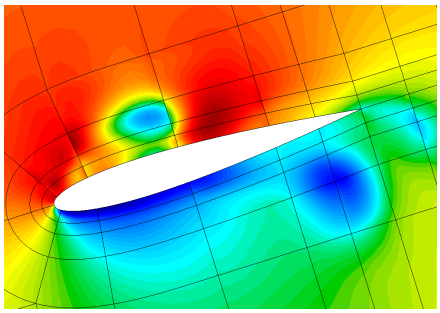


(c) Level 3

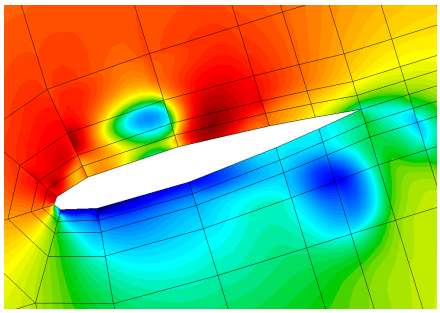


(d) Level 4

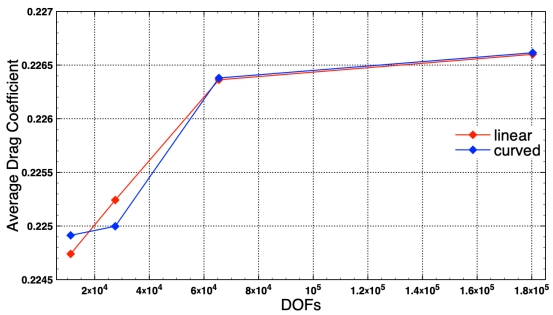
Figure 2.14: Meshes for the laminar test case



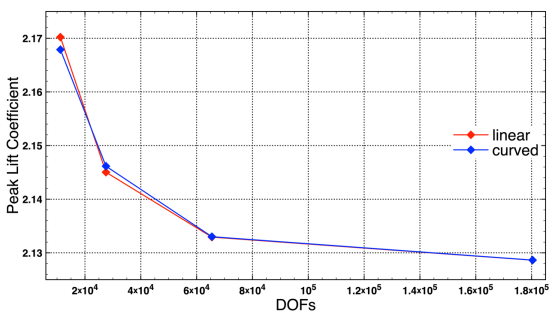
(a) Density field, curved mesh



(b) Density field, linear mesh



(c) Convergence of average drag coefficient



(d) Convergence of peak lift coefficient

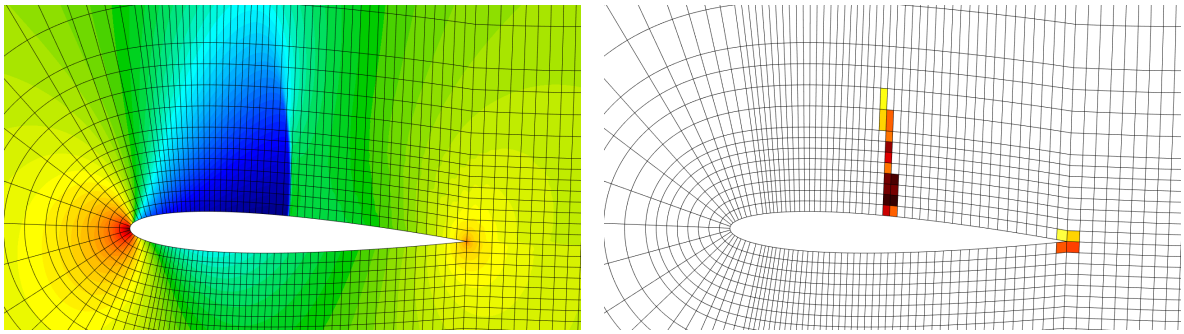
Figure 2.15: Influence of the geometry, laminar flow

2.8 Pitching airfoil in transonic flow

In the third case study, we validate the shock capturing capability of the proposed scheme by investigating the transonic flow around a pitching NACA 0012 airfoil. The angle of attack evolves with the following law:

$$\alpha(t) = \alpha_0 + \Delta\alpha \sin(2\pi f t), \tag{2.42}$$

where $\alpha_0 = 0.016°$, $\Delta\alpha = 2.51°$ and the reduced frequency $k = 0.0814$. The airfoil oscillates about the quarter-chord and the mesh movement technique is the same used in the previous case study. The freestream Mach number is equal to 0.755, in order to create a transonic flow over the airfoil. An inviscid fluid is considered, therefore $\mu = 0$. However, due to the presence of the artificial viscosity, the Navier-Stokes equations are solved. A uniform flow field is used as initial condition and a slip wall model is used for the airfoil surface. The results presented here are obtained using a quartic polynomial approximation with 3790 elements.



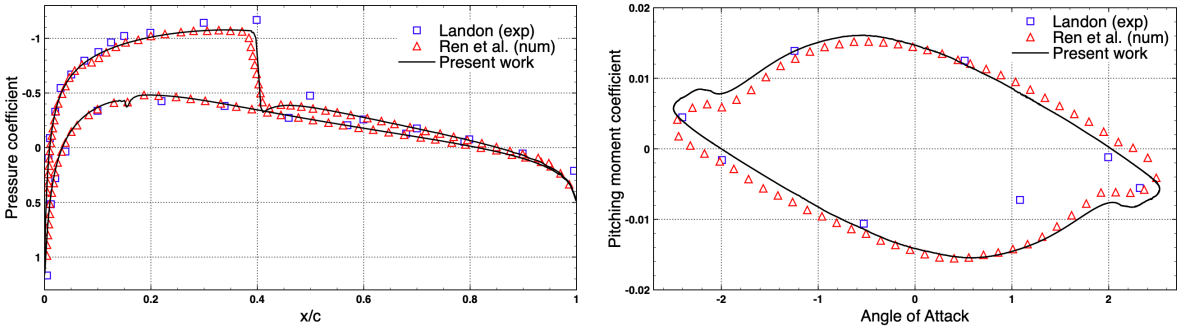
(a) Density field

(b) Artificial viscosity

Figure 2.16: Transonic pitching airfoil, solution at $\alpha = 2.25°$, descending phase

Transonic flows around pitching airfoils are a challenging benchmark for shock capturing algorithms, as the discontinuity position and intensity vary with the angle of attack. The density field and the artificial viscosity at $\alpha = 2.25°$ are reported in Fig. 2.16. The shock is well detected by the sensor, thus, the artificial viscosity is added only in the cells that require stabilization. It can also be observed that, due to a slight under-resolution, some artificial viscosity is added around the trailing edge.

In Fig. 2.17 we propose a comparison with the numerical results of Ren et al. (94) and the experimental data of Landon (95). The curve of the pressure coefficient C_p, presented in Fig. 2.17a, correlates very well with both the experimental and numerical references. Moreover, it is possible to appreciate the precision of the developed shock capturing technique. In Fig. 2.17b the evolution of the pitching moment coefficient over an oscillation cycle is presented. Even though a small discrepancy can be observed, the obtained curve is comparable to the numerical reference and it is in line with the majority of the experimental measurements.



(a) C_p curve at $\alpha = 2.34°$, ascending phase

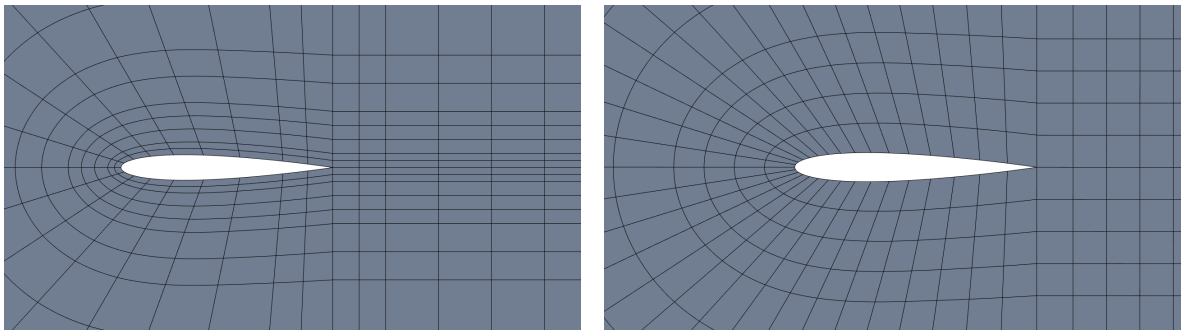
(b) C_m curve over a limit cycle

Figure 2.17: Transonic pitching airfoil, comparison with reference data

The deviations with respect to the wind tunnel data can be imputed to the impossibility to exactly reproduce the experimental conditions, characterized by a very high Reynolds number $(5.5 \cdot 10^6)$.

2.9 Pitching airfoil with adaptive mesh

The potential of developed ALE-AMR coupling technique is demonstrated by repeating the experience of the pitching airfoil, both in the laminar condition and in the transonic configuration. We first construct a different baseline mesh for each flow condition, as shown in Fig. 2.18. In the baseline grid for the laminar flow, each side of the airfoil is described with 9 Bézier curves and the elements close to the wall are anisotropic for an optimal boundary layer resolution. On the other hand, the baseline mesh for the transonic case is characterized by isotropic elements and 16 Bézier arcs are employed for a smooth approximation of the airfoil. A polynomial representation of degree 3 is employed for both configurations.



(a) Laminar flow

(b) Transonic flow

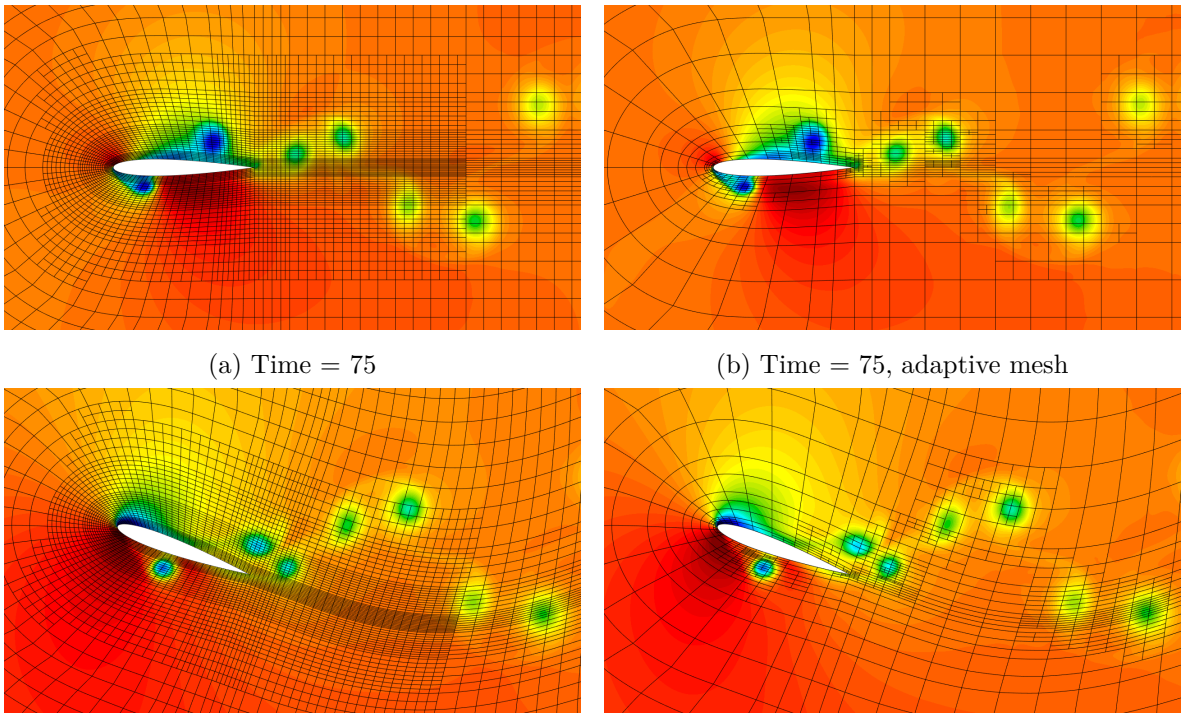
Figure 2.18: Baseline meshes for adaptive pitching airfoil simulation

A comparison between non-adaptive and adaptive grids is carried out. In order to provide

a fair evaluation, we adopt the same refinement levels for both grids. The adaptive simulation is performed using 2 levels of dynamic quadtree refinement. Similarly, the non-adaptive mesh is obtained by twice refining the baseline using rectangular boxes to select the regions where a higher resolution is desired. This ensures that the size of the smallest elements is the identical for both simulations.

2.9.1 Laminar flow

The laminar configuration allows to test the AMR capabilities for massively separated flows. As shown in Fig. 2.19, the dynamic mesh adaptation procedure is capable of capturing the formation of the vortices at the leading edge of the airfoil and following the complex wake pattern induced by the pitching motion. The advantage in terms of the number of elements is represented in Fig. 2.20a, one can observe that after an initial peak, the adaptive mesh has nearly three times less elements than the non-adaptive grid. In Fig. 2.20b we report the time evolution of the aerodynamic force coefficients C_d and C_l. The curves obtained with the two meshes are superposed, meaning that the reduction in the number of elements has very little impact on the accuracy of the simulation. As shown in Fig. 2.19, the solution fields are in close agreement as well.



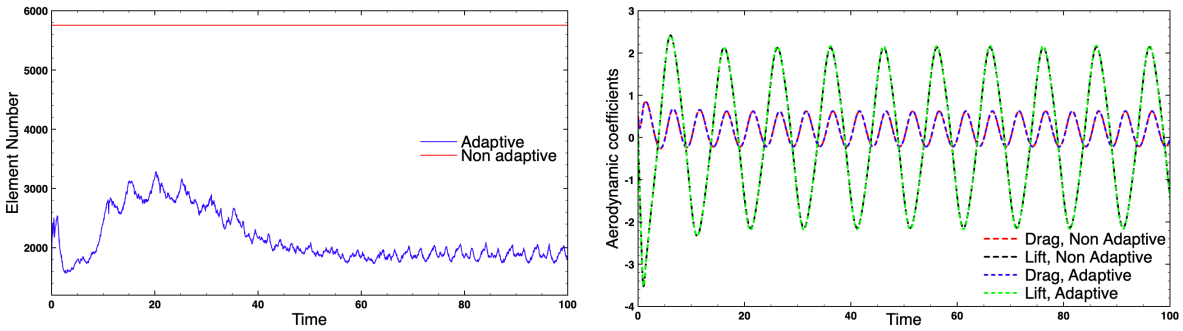
(a) Time = 75

(b) Time = 75, adaptive mesh

(c) Time = 87.5

(d) Time = 87.5, adaptive mesh

Figure 2.19: Laminar pitching NACA airfoil, comparison of the density field



(a) Evolution of the number of elements

(b) Evolution of the aerodynamic coefficients

Figure 2.20: Laminar pitching NACA airfoil, results comparison

2.9.2 Transonic flow

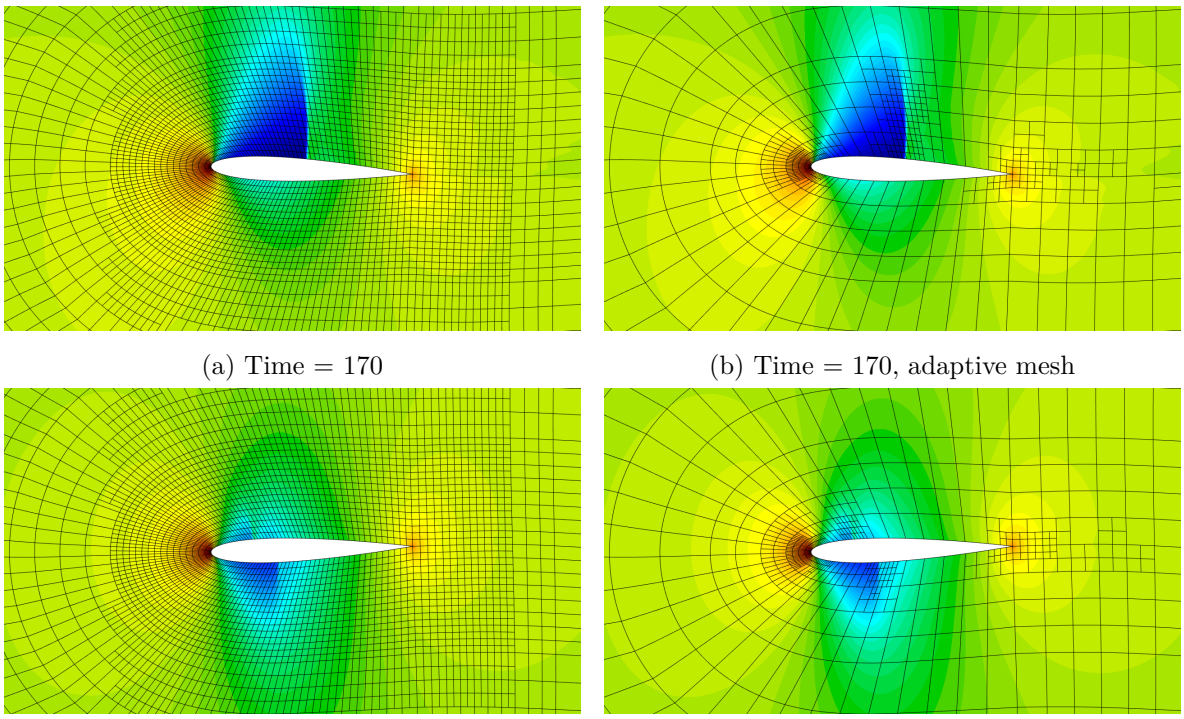
Simulating unsteady transonic flows with high-order solvers can be challenging, indeed, the presence of moving shocks requires a very fine grid around the airfoil and a robust shock capturing algorithm. In this context, the use of AMR allows to have the necessary mesh resolution around the shock while keeping a very coarse grid in the other regions of the airfoil, as shown in Fig. 2.21. This is obviously possible thanks to the high order representation of the boundary. The time evolution of the pitching moment coefficient C_m is reported in Fig. 2.22b for both meshes, the two curves are comparable, but a slight discrepancy can be observed. Indeed, the mesh adaptation interacts with the shock sensor, thus altering the quantity of artificial viscosity introduced to capture the discontinuity. The density fields obtained with the two approaches are comparable as well, as shown in Fig. 2.21, and the gain in terms of number of elements, illustrated in Fig. 2.22a, is considerable.

2.10 Conclusion

In this chapter we investigated the use of NURBS-based grids in the context of moving computational domains. Different possible approaches to account for moving meshes in a DG formulation were analysed, in the perspective of implementing a scheme for moving rational Bézier elements. The proposed formulation combines an isogeometric approach for DG with the ALE description, leading to a unified representation of the geometry, the grid velocity and the solution variables.

Firstly, the accuracy of the method was verified on two problems with analytical solutions, exhibiting optimal convergence rates for rigidly moving or deforming grids. As expected, large grid deformations cause an increase of the error, due to the loss of the accuracy of the approximation on distorted elements. The discrete geometry conservation law is satisfied when polynomial Bézier patches are considered. On the contrary, constant solutions are not

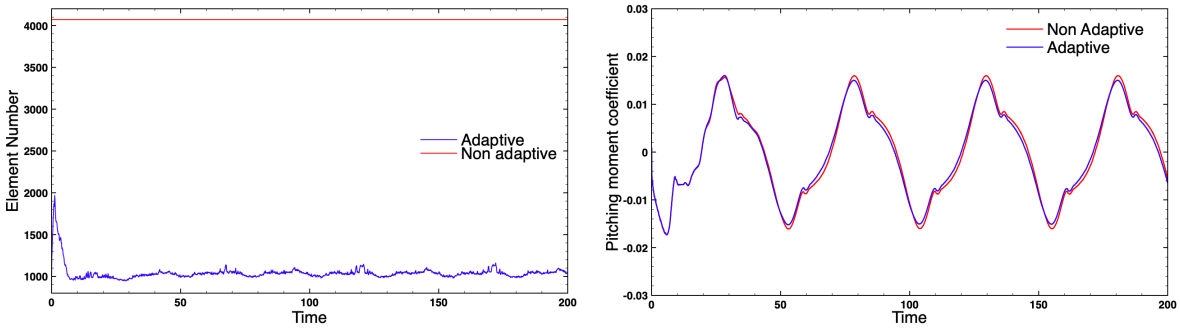
exactly preserved when rational elements are used, due to the approximation introduced by the numerical quadrature.



(a) Time = 170 (b) Time = 170, adaptive mesh

(c) Time = 185 (d) Time = 185, adaptive mesh

Figure 2.21: Transonic pitching NACA airfoil, comparison of the density field



(a) Evolution of the number of elements (b) Evolution of the pitching moment coefficient

Figure 2.22: Transonic pitching NACA airfoil, results comparison

We then considered three more demanding test-cases, involving inviscid and viscous flows around an oscillating cylinder and a pitching airfoil. The results confirmed the robustness of the proposed scheme in presence of high-order mesh deformations. Moreover, a faster convergence of the flow characteristics thanks to the use of curved grids was clearly established,

in the context of moving bodies. It was also noted that inviscid flows are particularly sensitive to the lack of regularity in the geometry, due to the slip boundary condition at wall. Finally, we showed the robustness of the present methodology in presence of shocks, using a transonic benchmark problem.

Furthermore, we proposed a simple yet effective strategy to couple AMR and ALE using the hierarchical properties of NURBS. The dynamic mesh deformation-adaptation was tested on the pitching airfoil benchmark in two different flow configurations. The obtained results show the interest of the ALE-AMR coupling. A significant reduction in the number of elements can be obtained without losing accuracy, thanks to the dynamic mesh refinement, which is capable of capturing the most prominent features of unsteady flows, such as moving shocks and complex wakes.

Chapter 3

Sliding meshes

One of the principal limitations of ALE approaches is the necessity of remeshing when confronted with large displacements of the boundary. For this reason, the simulation of flows involving rotating objects is often carried out using sliding grids, in which the computational domains is subdivided into a fixed and a rotating region. The discontinuous mesh movement between the fixed and the moving subdomain generates hanging nodes along the sliding interface. In this context, DG schemes are natively capable of handling geometric non-conformities. Ferrer et al. (71) proposed the first DG method with sliding grids for incompressible flows. Similarly, a DG discretization can be employed in IGA, when non-matching parameterisation are encountered (96). Moreover, thanks to the rational representation, NURBS are capable of exactly representing circular interfaces. In particular, Bazilevs et al. (70) combined a quadratic NURBS representation with a DG discretization of the sliding interface to solve the incompressible Navier-Stokes equations. An alternative strategy is to employ mortars to deal with the hanging nodes. The mortar approach is based on projections and it was first introduced in the context of spectral element methods to treat non-conforming meshes (97; 98). More recently, the mortar method has been adopted to simulate compressible flows with sliding meshes, combined with compact high-order discretizations (99; 72). Mortar based methods, however, suffer from loss of conservativity when dealing with curvilinear sliding interfaces, due to the nature of the projections (72). Moreover, as evidenced in (100), interpolation based DG methods are significantly less expensive than mortar approaches.

In this chapter we develop a fully conservative DG method for compressible flows with sliding meshes. The proposed algorithm is explained in details in section 3.1. The properties of the NURBS representation are employed to subdivide the sliding interface and compute the flux integrals. The developed methodology is then validated using a classic benchmark for compressible flows and a thorough error analysis is conducted. Section 3.3 is dedicated to the investigation of the viscous flow around a pitching ellipse, with the aim of testing the sliding mesh algorithm on a more demanding test case. In particular, a mesh sensitivity analysis

is performed, with respect to the grid refinement level, the basis degree and the position of the sliding interface. Moreover, a comparison with the previously developed ALE approach is presented. We also simulate the pitching ellipse in a supersonic flow condition to test the robustness of the sliding mesh algorithm in presence of strong shocks. Lastly, in order to prove the potential of the Isogeometric DG framework for more complex geometries, a 3-bladed vertical axis wind turbine configuration is considered in section 3.4. The generation of a suitable computational grid is discussed and a flow simulation is carried out.

3.1 Sliding mesh algorithm

In this section we explain in details the treatment of sliding meshes. The algorithm is characterized by three phases. First, the geometry is updated and the control points of the mesh are moved. As the movement is performed, the connectivity between the interface elements is checked and updated. Once the grid connectivity is established, the interface is subdivided into smaller arcs shared by only two elements. Lastly, a special treatment of the sliding faces is required in order to accurately compute the flux integrals.

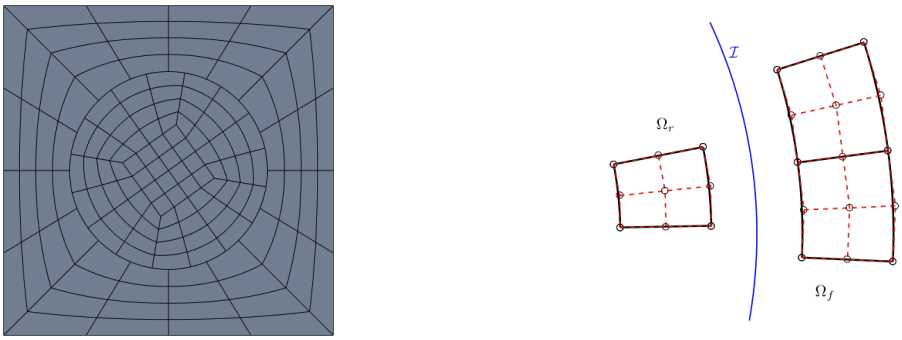
3.1.1 Mesh movement and connectivity update

For the sake of simplicity, we analyse the case of a single rotating subdomain Ω_r surrounded by a fixed region Ω_f. The sliding interface is thus defined as $\mathcal{I} = \partial\Omega_r \cap \partial\Omega_f$. Using the ALE framework we proposed in chapter 2, the mesh velocity is described as a DG field. As a consequence, it is naturally possible to represent discontinuous displacements, which is particularly useful in the context of sliding meshes. Indeed, the sliding interface is characterized by a discontinuity of the tangential velocity $\mathbf{V}_g \cdot \mathbf{t}$. On the other hand, the normal component $\mathbf{V}_g \cdot \mathbf{n}$ must be continuous, in order to keep the mesh watertight. For perfectly circular interfaces, the control point velocities $(u_{g,i},\, v_{g,i})$ can be easily computed thanks to the affine invariance property of Bézier patches (74). Defining the angular velocity ω and the axis of rotation $O = (x_0, y_0)$, we have, for each control point $(x_i,\, y_i)$ in the rotating subdomain Ω_r:

$$\begin{cases} u_{g,i} = -\omega(y_i - y_0), \\ v_{g,i} = \omega(x_i - x_0). \end{cases} \tag{3.1}$$

Since the mesh movement is piecewise rigid for perfectly circular interfaces, we can adopt eq. (2.20) to perform time integration and the mass matrix does not need to be recomputed for each time step.

Assuming that the elements are initially aligned, the discontinuity in the mesh movement generates hanging nodes along the sliding interface, as it can be observed in Fig. 3.1. Consequently, the degrees of freedom of the elements adjacent to the interface are not aligned.



(a) Domain subdivision (b) Degrees of freedom along the interface

Figure 3.1: Generation of hanging nodes due to the sliding movement

As the inner domain rotates, the position of the hanging nodes changes as well. In order to update the mesh connectivity, we rely on the computation of two angles, as represented in Fig. 3.2. $\Delta\theta_0$ is the angular amplitude of the element edge between $\mathbf{x}_1$ and $\mathbf{x}_2$, whereas $\Delta\theta_s$ is the angular amplitude of the circular arc between $\mathbf{x}_1$ and the hanging node $\mathbf{x}_s$. We then define the ratio s as:

$$s = \frac{\Delta\theta_s}{\Delta\theta_0}\,. \tag{3.2}$$

The connectivity is valid when $0 < s < 1$, whereas it must be updated when the angular ratio s becomes negative or greater than the unity.

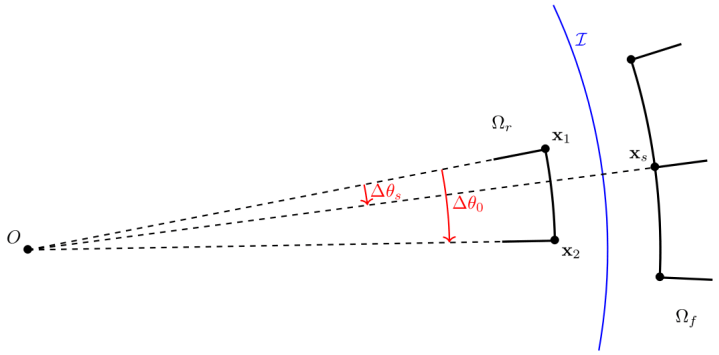


Figure 3.2: Definition of the angles for connectivity update

3.1.2 Interface splitting

The second step of the sliding algorithm is the subdivision of the interface into the elemental faces, which are shared uniquely by two elements. Let us name Ω_0 the rotating element, Ω_1 and Ω_2 the corresponding fixed elements, as illustrated in Fig. 3.3. The elemental sliding faces are therefore $\Gamma_1 = \partial\Omega_0 \cap \partial\Omega_1$ and $\Gamma_2 = \partial\Omega_0 \cap \partial\Omega_2$. We also define $\Gamma_0 = \partial\Omega_0 \cap \mathcal{I} = \Gamma_1 \cup \Gamma_2$.

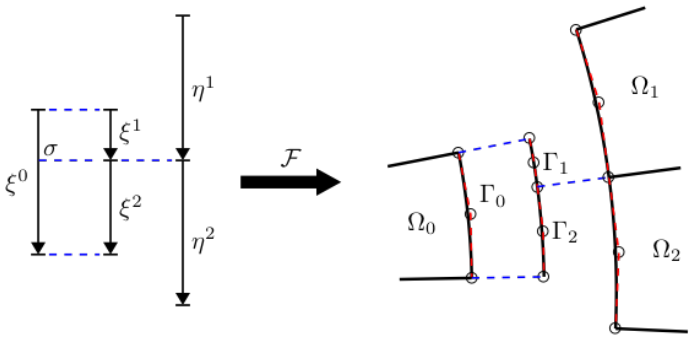


Figure 3.3: Splitting procedure in the parametric and physical spaces

The splitting procedure relies on the properties of the NURBS representation. Indeed, we recall that a Bézier curve of degree p defined on the parametric interval $[0, 1]$ can be interpreted as a NURBS curve with a the following knot vector:

$$\Xi = \{\underbrace{0, \ldots, 0}_{p+1}, \underbrace{1, \ldots, 1}_{p+1}\}. \tag{3.3}$$

Let us define the splitting parameter σ as the value of ξ^0 that corresponds to the physical position of the hanging node $\mathbf{x}_s$ on the face Γ_0. It is then possible to divide Γ_0 into the two elemental faces, Γ_1 and Γ_2, by applying $(p+1)$-times the knot insertion algorithm at the parametric coordinate $\xi^0 = \sigma$. We thus obtain two Bézier curves with the following knot vectors:

$$\Xi^1 = \{\underbrace{0, \ldots, 0}_{p+1}, \underbrace{\sigma, \ldots, \sigma}_{p+1}\}, \quad \Xi^2 = \{\underbrace{\sigma, \ldots, \sigma}_{p+1}, \underbrace{1, \ldots, 1}_{p+1}\}. \tag{3.4}$$

The control points of the new curves are obtained by recursively using p times the knot insertion formula, specialized here for Bézier representations:

$$\mathbf{x}_i^{r+1} = (1 - \alpha_i^r)\mathbf{x}_{i-1}^r + \alpha_i^r \mathbf{x}_i^r, \quad \alpha_i^r = \begin{cases} 1 & \text{if } i \leq r+1 \\ \sigma & \text{if } r+2 \leq i \leq p+1 \\ 0 & \text{if } i \geq p+2 \end{cases} \tag{3.5}$$

with $r = 0, \ldots, (p-1)$. The result of the splitting procedure is illustrated in Fig. 3.3 for a quadratic representation.

In order to adopt the described splitting algorithm, the value of σ must be known. Due to the rational nature of Bézier representation, computing σ is not trivial. The procedure that allows to determine the parameter corresponding to a given point of a Bézier curve is the point inversion algorithm (74). If we denote $\mathbf{C}$ as the Bézier representation of the Γ_0 curve,

we can write:

$$\mathbf{C}(\sigma) = \frac{\sum_{i=1}^{p+1} B_i^p(\sigma)\,\omega_i\,\mathbf{x}_i}{\sum_{j=1}^{p+1} B_j^p(\sigma)\,\omega_j} = \mathbf{x}_s, \tag{3.6}$$

where ω_i and $\mathbf{x}_i$ are respectively the weights and the control points of the considered edge. The problem of determining σ is solved by minimizing the square of the distance between the curve $\mathbf{C}$ and the hanging node:

$$g(\xi^0) = ||\mathbf{C}(\xi^0) - \mathbf{x}_s||^2. \tag{3.7}$$

Imposing $g'(\sigma) = 0$, one obtains:

$$f(\sigma) = \mathbf{C}'(\sigma) \cdot (\mathbf{C}(\sigma) - \mathbf{x}_s) = 0. \tag{3.8}$$

The solution of eq. (3.8) is achieved by means of an iterative algorithm. Piegl et al. (74) adopted the Newton algorithm to solve the point inversion problem. Such an approach requires to analytically compute the first derivative of f, which contains the second derivative of the Bézier curve $\mathbf{C}$. In order to avoid the costly computation of $\mathbf{C}''$, we employ the secant algorithm, which results in the following recursive formula:

$$\sigma^i = \frac{\sigma^{i-2} f(\sigma^{i-1}) - \sigma^{i-1} f(\sigma^{i-2})}{f(\sigma^{i-1}) - f(\sigma^{i-2})}, \tag{3.9}$$

and we initialise the algorithm taking σ_1 equal to the angular ratio s used to update the connectivity, and σ_0 is equal to the splitting parameter computed for the previous time step. The convergence conditions to stop the secant iterations are:

$$\begin{cases} f(\sigma^i) \leq 10^{-15}, \\ |\sigma^i - \sigma^{i-1}| \leq 10^{-14}. \end{cases} \tag{3.10}$$

Since the values used to initialise the algorithm already provide a close estimate of the actual splitting parameter, few iterations are required. In practice, we observed that the convergence conditions are always satisfied in less than 5 iterations.

3.1.3 Flux computation

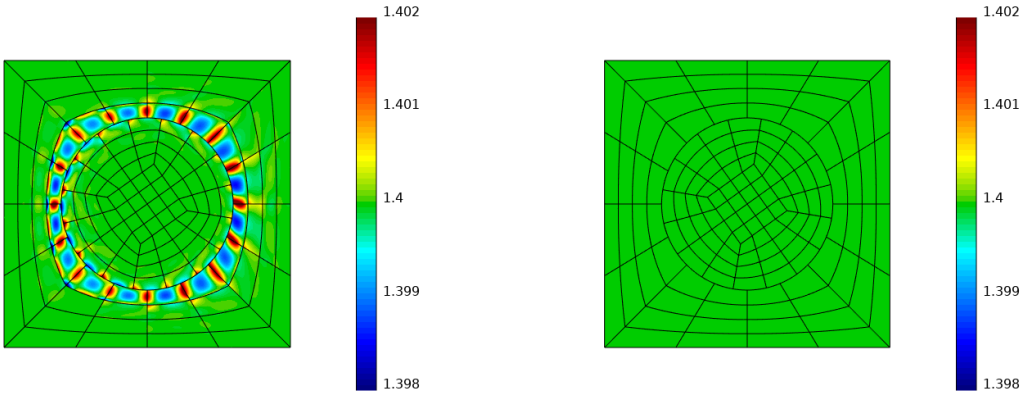
Once the interface is split into the elemental faces, it is possible to compute the flux integrals of the DG scheme. For each elemental segment of the interface, two integrals have to be computed, one for the rotating element and one for the fixed element. The difficulty of computing the flux integrals is due to the non-matching parameterisation on the two sides of the interface. Indeed, due to the lack of alignment of the elements between Ω_r and Ω_f, the face Γ_1 is described using two different parameterisations and the same is true for Γ_2, as

presented in Fig. 3.3. If we focus on Γ_1, the flux integrals for a numerical flux function $\mathbf{F}^*$ are:

$$\mathbf{f}_1^- = \int_{\Gamma_1} R_k(\xi^1)\, \mathbf{F}^*(w_h^-(\xi^1), w_h^+(\eta^1), \mathbf{n})\, \mathrm{d}\Gamma_1, \tag{3.11}$$

$$\mathbf{f}_1^+ = \int_{\Gamma_1} R_k(\eta^1)\, \mathbf{F}^*(w_h^-(\xi^1), w_h^+(\eta^1), \mathbf{n})\, \mathrm{d}\Gamma_1. \tag{3.12}$$

One could naively try to use as quadrature points the Gauss-Legendre nodes on the interval $\xi^1 \in [0, \sigma]$ for $\mathbf{f}_1^-$ and on the interval $\eta^1 \in [1-\sigma, 1]$ for $\mathbf{f}_1^+$. However, the positions of the integration points in the physical space do not match, due to the non-linearity of the coordinate transformation, resulting in a locally non-conservative scheme. The effect of the loss of conservativity is presented in Fig. 3.4a for a freestream preservation problem. We can clearly observe the spurious oscillations around the sliding interface.



(a) Non-conservative interface

(b) Fully conservative interface

Figure 3.4: Freestream preservation with non-conservative and conservative sliding interfaces

We can instead obtain a fully conservative scheme by computing both flux integrals on the parametric interval $\xi_1 \in [0, \sigma]$. We therefore rewrite $\mathbf{f}_1^-$ and $\mathbf{f}_1^+$ as:

$$\mathbf{f}_1^- = \int_0^{\sigma} R_k(\xi^1)\, \mathbf{F}^*(w_h^-(\xi^1), w_h^+(\eta^1(\xi^1)), \mathbf{n})\, |\mathrm{J}_{\Gamma_1}|\, \mathrm{d}\xi^1, \tag{3.13}$$

$$\mathbf{f}_1^+ = \int_0^{\sigma} R_k(\eta^1(\xi^1))\, \mathbf{F}^*(w_h^-(\xi^1), w_h^+(\eta^1(\xi^1)), \mathbf{n})\, |\mathrm{J}_{\Gamma_1}|\, \mathrm{d}\xi^1, \tag{3.14}$$

where $|\mathrm{J}_{\Gamma_1}|$ is the Jacobian of the map between ξ_1 and Γ_1. In order to approximate expressions (3.14) and (3.13) with numerical quadrature, the reparameterisation $\eta^1(\xi^1)$ has to be computed for the quadrature points. If we denote ξ^1_{gp} as the parametric coordinate of a generic Gauss-Legendre point, then its physical coordinate is equal to:

$$\mathbf{x}_{gp} = \frac{\sum_{i=1}^{p+1} B_i^p(\xi^1_{gp})\, \omega_i\, \mathbf{x}_i}{\sum_{j=1}^{p+1} B_j^p(\xi^1_{gp})\, \omega_j} = \frac{\sum_{i=1}^{p+1} B_i^p(\eta^1_{gp})\, \upsilon_i\, \mathbf{y}_i}{\sum_{j=1}^{p+1} B_j^p(\eta^1_{gp})\, \upsilon_j}, \tag{3.15}$$

where $\mathbf{x}_i$ and ω_i are the control points and weights of the elemental face, whereas $\mathbf{y}_i$ and υ_i are the control points and weights of the edge of Ω_1. The reparameterisation η^1_{gp} is found by solving eq. (3.15) with the point inversion algorithm described in the previous section. The same treatment is then repeated for the integrals over the face Γ_2:

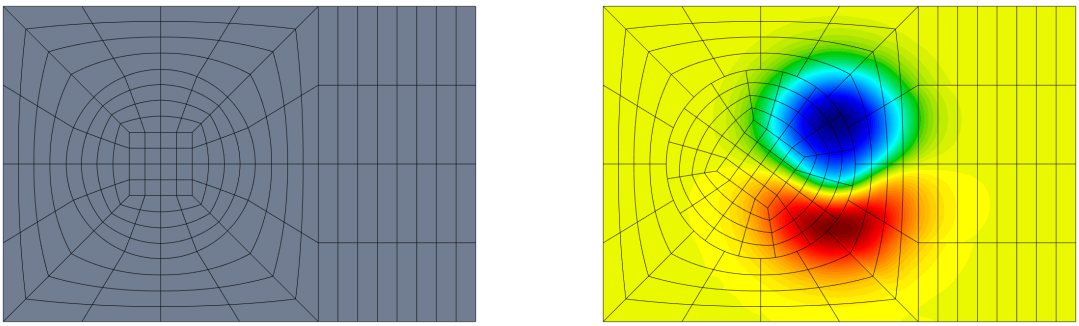
$$\mathbf{f}_2^- = \int_\sigma^1 R_k(\xi^2)\, \mathbf{F}^*(w_h^-(\xi^2), w_h^+(\eta^2(\xi^2)), \mathbf{n})\, |\mathrm{J}_{\Gamma_2}|\, \mathrm{d}\xi^2, \tag{3.16}$$

$$\mathbf{f}_2^+ = \int_\sigma^1 R_k(\eta^2(\xi^2))\, \mathbf{F}^*(w_h^-(\xi^2), w_h^+(\eta^2(\xi^2)), \mathbf{n})\, |\mathrm{J}_{\Gamma_2}|\, \mathrm{d}\xi^2. \tag{3.17}$$

In order to show the effectiveness of the proposed strategy, we report in Fig. 3.4b the result of the freestream preservation test and compare it with non-conservative integral computation. We can observe that the spurious oscillation around the interface are absent and that the constant solution is well preserved.

3.2 Verification: isentropic vortex

The developed sliding mesh technique is firstly tested on the isentropic vortex test case already used to validate the ALE formulation in chapter 2, section 2.5. The computational domain is $[2.5, 10] \times [-2.5, 2.5]$ and the boundary fluxes are computed using the exact solution 2.35.



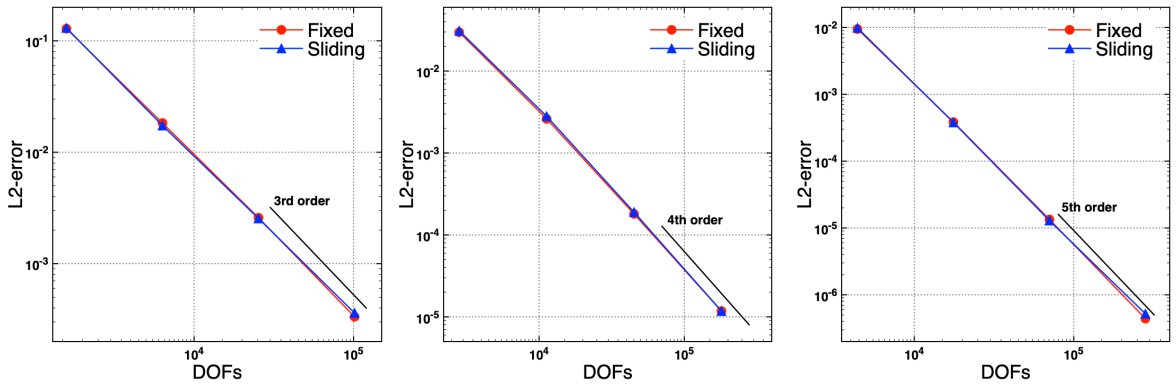
(a) Baseline mesh

(b) Mesh and solution field at t=1.2

Figure 3.5: Isentropic vortex, setup

The computational domain is divided into two regions, a rotating one and a fixed one, as illustrated in Fig. 3.5. The radius of the sliding interface is equal to 1.5, the centre of rotation of the inner region corresponds to the axis of the vortex, and a unitary rotational frequency is adopted. The core of the vortex is therefore inside the rotating subdomain in the initial phase of the simulation. As the vortex is advected downstream, it crosses the sliding interface to enter the fixed subdomain, as it can be observed in Fig. 3.5b. The proposed test case allows to investigate how well the sliding interface is capable of preserving the flow characteristics.

In order to perform a rigorous error analysis, a convergence study is carried out. Starting from the baseline mesh presented in Fig. 3.5a, isotropic refinement is applied to obtain four different levels of mesh resolution. Quadratic, cubic and quartic basis functions are tested. For each combination of degree and mesh level, a simulation is performed considering a fixed inner subdomain as well. By comparing the results obtained with the two approaches, it is possible to quantify the error introduced by the computation of the flux integrals on the sliding interface. The L^2-norm of the error for the total energy field is evaluated at $t = 2$ using numerical quadrature. The results of the convergence analysis are presented in Fig. 3.6. It can be observed that optimal convergence rates are obtained for all basis degrees for both fixed and sliding meshes. Furthermore, roughly the same errors are obtained with the two techniques, meaning that the error generated by the sliding interface is negligible and flow structures are perfectly preserved.



(a) p = 2 (b) p = 3 (c) p = 4

Figure 3.6: Isentropic vortex, error analysis

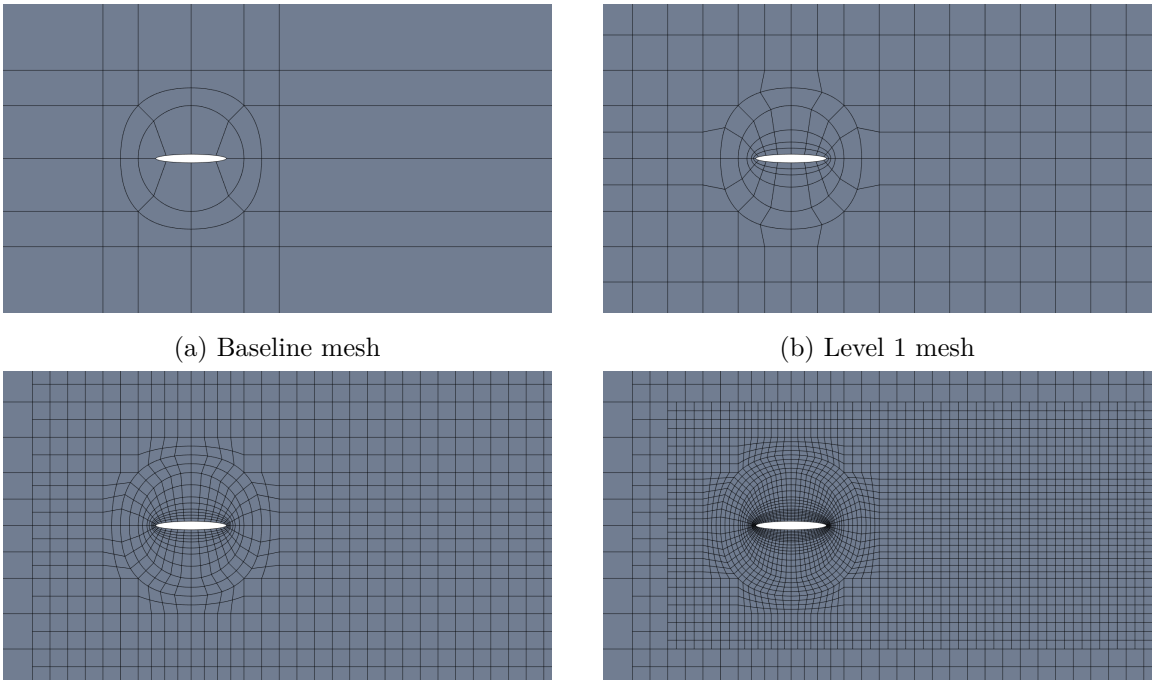
3.3 Case study: pitching ellipse flow

The second test case concerns the bidimensional flow around a pitching ellipse. Thanks to the use of rational Bézier functions, the ellipse can be exactly represented. Thus, there is no additional error introduced by the discretization of the computational domain. The aspect ratio of the ellipse is equal to 12 and its angle of attack evolves in time with the following law:

$$\alpha(t) = -A \sin(2\pi f t), \tag{3.18}$$

with reduced frequency $k = \pi f c / U_\infty = 0.5\pi$ and amplitude $A = 30°$. The ellipse is pitching about its center. We consider a laminar subsonic flow condition with freestream Mach number equal to 0.2 and a Reynolds number of 500. The flow configuration is characterized by a periodic vortical wake generated by the dynamic separation of the boundary layer around

the ellipse. Therefore, the aim of the proposed test case is to study the behaviour of the developed sliding mesh technique for complex unsteady flows.



(a) Baseline mesh

(b) Level 1 mesh

(c) Level 2 mesh

(d) Level 3 mesh

Figure 3.7: Different refinement levels for the ellipse simulation

3.3.1 Mesh convergence

Firstly, a mesh convergence study is conducted. We consider a large computational domain, delimited by the rectangle $[-30c, 60c] \times [-30c, 30c]$, with c being the major axis of the ellipse. The far-field boundary conditions are weakly imposed using Riemann invariants. On the ellipse surface, a weakly prescribed no-slip adiabatic wall boundary condition is adopted. Since the isogeometric approach allows an exact representation of the boundary, an extremely coarse baseline mesh can be generated. As represented in Fig. 3.7, the meshes adopted for the actual computations are obtained applying a hierarchical refinement to the baseline configuration. The convergence analysis is carried out using 3 mesh refinement levels:

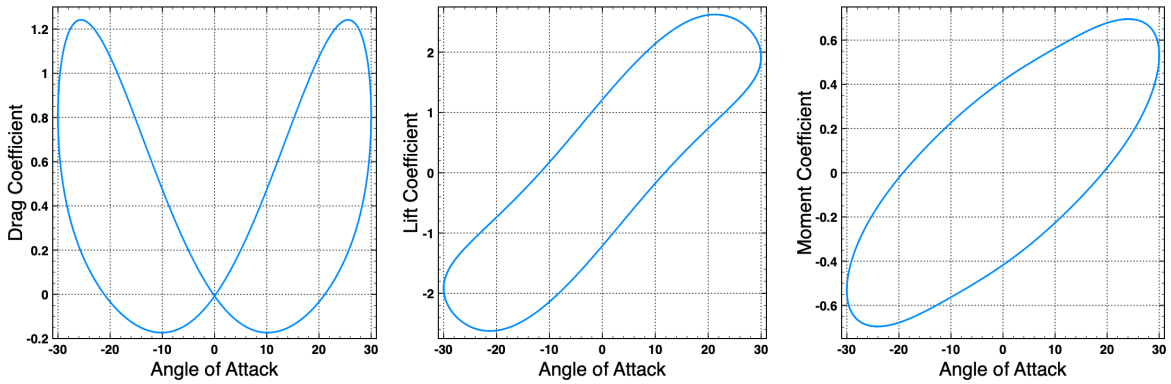
- level 1: 1464 total elements, 16 of which on the ellipse boundary,
- level 2: 2406 total elements, 32 of which on the ellipse boundary,
- level 3: 5046 total elements, 64 of which on the ellipse boundary.

The study is performed using rational basis defined using quadratic, cubic and quartic polynomials. The radius of the sliding interface is equal to $0.75c$, meaning that the minimum distance between the ellipse and the interface is of a quarter of the major axis length c.

The initial condition of the simulations corresponds to the uniform freestream flow state. After a numerical transient, the solution converges to a stable limit cycle in which the flow fluctuations are synchronized with the pitching movement of the ellipse. We present in Fig. 3.8 the evolution of the aerodynamic coefficients over one pitching cycle. We compute, for each cycle, the average drag coefficient $\bar{C}_d$, the peak lift coefficient $\widehat{C}_l$ and the energy transfer coefficient E. The latter being defined as the area delimited by the $C_m - \alpha$ curve:

$$E = \frac{1}{\frac{1}{2}\rho_\infty u_\infty^2 c^2} \oint M_z \, \mathrm{d}\alpha = \oint C_m \, \mathrm{d}\alpha, \tag{3.19}$$

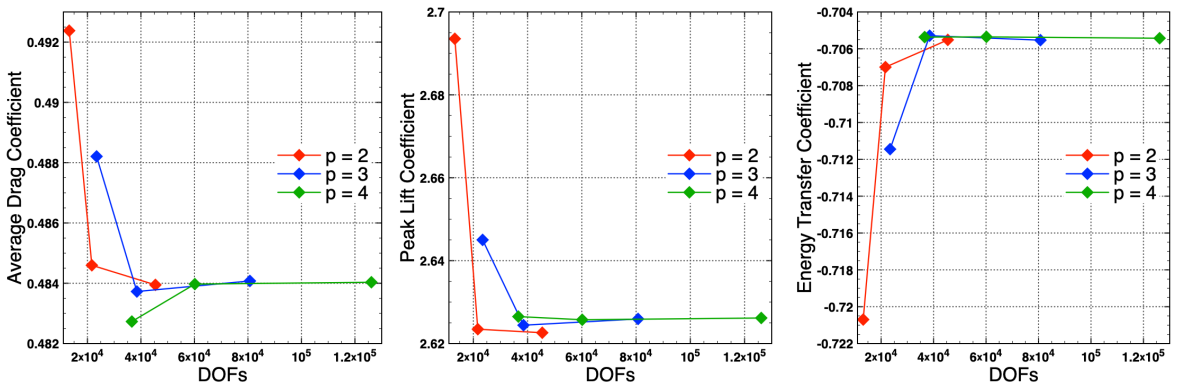
where M_z is the moment of the aerodynamic forces with respect to the pitching axis. The simulations are stopped when the considered coefficients are sufficiently converged in time.



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

Figure 3.8: Pitching ellipse, limit cycle solution, cubic level 2 mesh

The results of the mesh convergence analysis are reported in Fig. 3.9. All the tested configurations of degrees and refinement levels are able to reproduce the synchronization between the flow and the pitching movement. As expected, a significantly faster convergence is observed when the degree of the basis is increased. At the same time, the coarsest mesh is not capable of predicting with accuracy all the coefficients, even with quartic polynomials. Indeed, the level 2 grid with a cubic representation provides a better estimation of $\bar{C}_d$, with a similar number of degrees of freedom. Therefore, we conclude that the cubic level 2 mesh represents the best compromise between accuracy and computational cost. In general, we can say that the best trade-off is found by combining h and p refinement, as we already observed in chapter 2. We can also remark that the value of E is negative, which means that, from a physical point of view, the energy is transferred from the ellipse to the fluid.

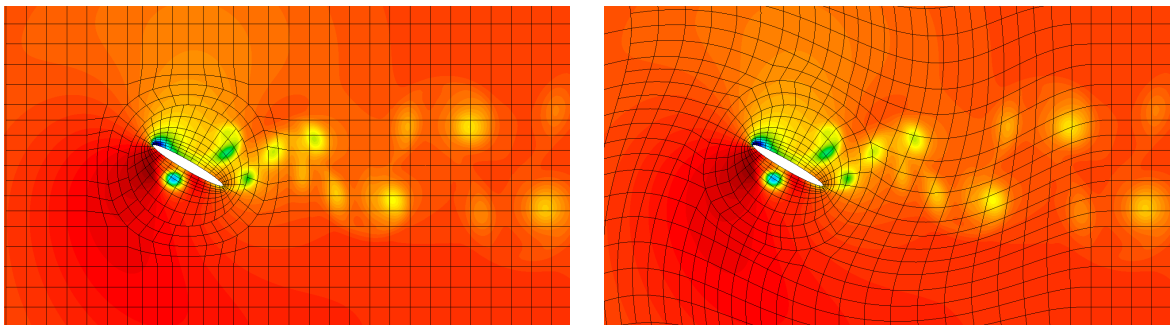


(a) Average drag (b) Peak lift (c) Energy transfer

Figure 3.9: Pitching ellipse, convergence of the aerodynamic coefficients

3.3.2 Comparison with deforming mesh

In the second part of the ellipse case study, we compare the sliding mesh approach with the ALE technique that we developed and validated in chapter 2. Since the pitching amplitude is limited to 30°, it is still possible to continuously deform the computational domain without incurring into an excessive distortion of the mesh, which could potentially decrease the accuracy of the solution. In order to compare the two techniques, we repeat the convergence study with the grids of Fig. 3.7 using the ALE methodology. The mesh velocity field $\mathbf{V}_g$ for the ALE computations is defined via the approach proposed in chapter 2, section 2.7. To compute the blending function (2.41), the distance R_{int} is equal to the radius of the sliding interface and $R_{ext} = 4c$. Intuitively, the sliding mesh movement can be interpreted as the limit case where $R_{ext} = R_{int}$. We illustrate in Fig. 3.10 the differences between the deforming and the sliding meshes. Besides, we can observe that, despite the significant differences between the two grids, the density fields obtained with the two approaches are visually identical.



(a) Sliding mesh (b) Deforming mesh

Figure 3.10: Comparison of sliding and deforming meshes, density field, quartic level 2 mesh

In order to better compare the two approaches, $\bar{C}_d$, $\widehat{C}_l$ and E are computed for the deforming mesh as well. The numerical values of the three aerodynamic coefficients are reported in Tables 3.1, 3.2 and 3.3 respectively. We can observe that the same converged values are obtained with both the techniques. Therefore, we conclude that the proposed sliding mesh approach is correctly capable of simulating complex flows. Although some small discrepancies are observed between the values obtained using the level 1 meshes, there is no evidence to infer that one methodology is more accurate than the other. However, the sliding approach is less expensive from a computational point of view, because the mass matrix is constant in time.

	Sliding			Deformation		
Degree	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
2	0.4924	0.4846	0.4839	0.4916	0.4833	0.4838
3	0.4882	0.4837	0.4841	0.4835	0.4836	0.4841
4	0.4827	0.4840	0.4840	0.4820	0.4840	0.4840

Table 3.1: Average drag coefficient for sliding and deforming meshes

	Sliding			Deformation		
Degree	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
2	2.694	2.623	2.623	2.691	2.619	2.622
3	2.645	2.624	2.626	2.630	2.624	2.626
4	2.627	2.626	2.626	2.624	2.626	2.626

Table 3.2: Peak lift coefficient for sliding and deforming meshes

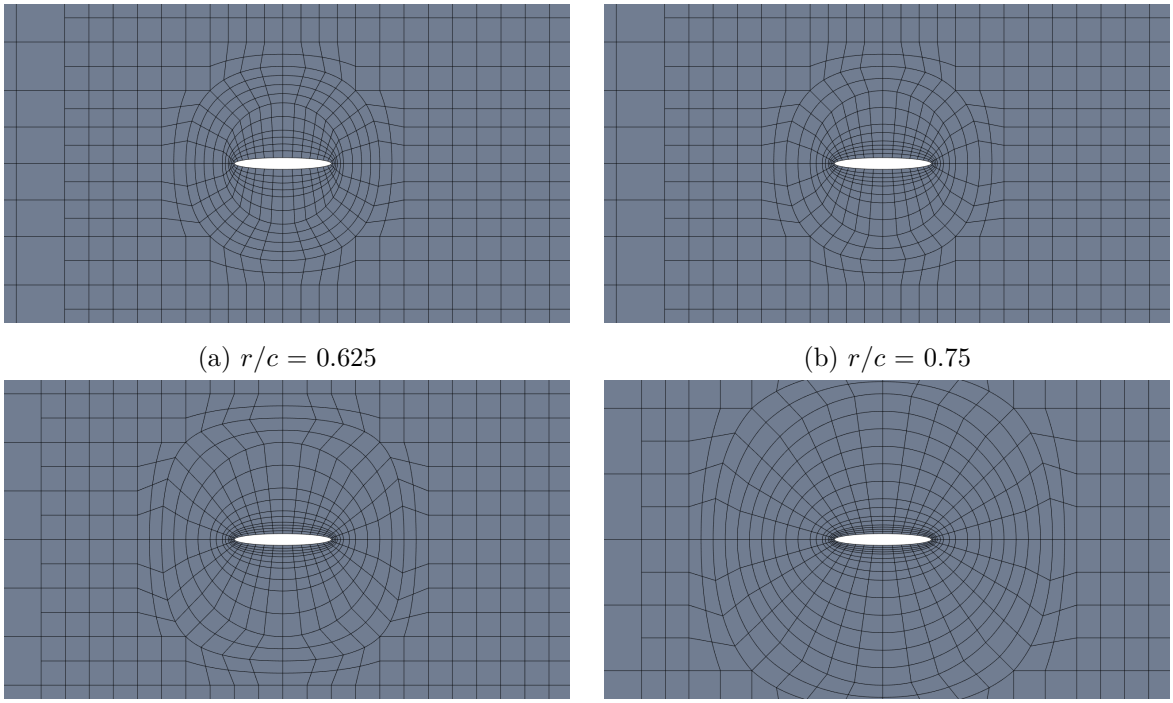
	Sliding			Deformation		
Degree	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
2	-0.7207	-0.7070	-0.7055	-0.7204	-0.7059	-0.7054
3	-0.7114	-0.7053	-0.7055	-0.7065	-0.7052	-0.7055
4	-0.7054	-0.7054	-0.7054	-0.7047	-0.7054	-0.7054

Table 3.3: Energy transfer coefficient for sliding and deforming meshes

3.3.3 Influence of the sliding interface radius

We then employ the ellipse case study to investigate the effect of the placement of the sliding interface on the numerical results. Low-order sliding mesh approaches can introduce numerical

artifacts in the approximate solution, due to loss of conservativity and poor interpolation. Ideally, the sliding interface should not be in close proximity of the boundary in order to limit the effects of the spurious phenomena on the aerodynamic coefficients. On the other hand, a smaller rotating region allows to reduce the computational costs, because the smaller tangential velocity of the mesh allows to increase the time step. Therefore, the generation of meshes with sliding interfaces is the result of a trade-off and the process heavily relies on user experience. In this context, the use of a geometrically exact movement coupled with a high-order interpolation can significantly improve the accuracy of sliding grids and simplify the mesh generation task.



(a) $r/c = 0.625$ (b) $r/c = 0.75$

(c) $r/c = 1.00$ (d) $r/c = 1.50$

Figure 3.11: Meshes with different sliding interface placements

The previous convergence analysis was carried out considering a sliding interface radius equal to $0.75c$. By slightly modifying the baseline grid 3.7a, we were able to obtain three additional meshes, with different placements of the interface: one with a smaller radius of $0.625c$, and two with a larger rotating region, with r equal to c and $1.5c$, as illustrated in Fig. 3.11. In order to provide a fair comparison, the refinement process has been adapted to obtain a similar resolution in the wall area for all the considered grids. A cubic representation is employed and the overall mesh resolutions are comparable to the level 2 refinement adopted in the convergence study.

The values of $\bar{C}_d$, $\widehat{C}_l$ and E obtained for the different positions of the sliding interface are

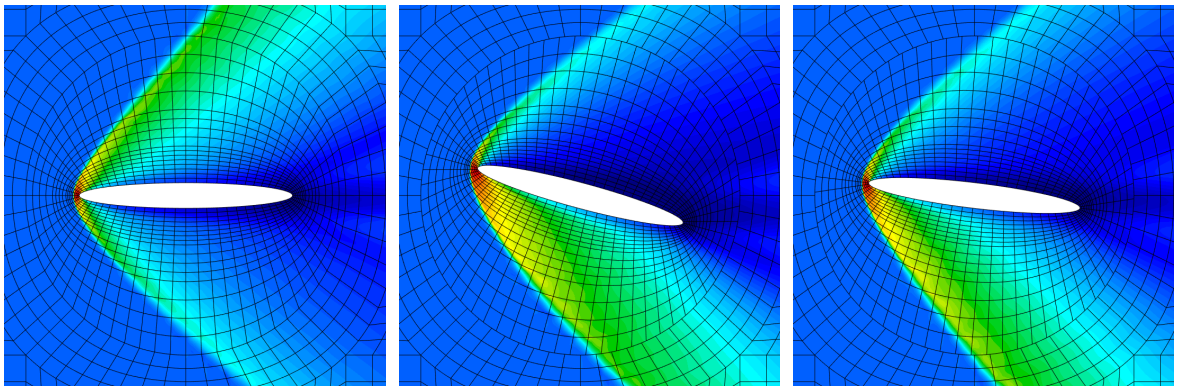
reported in Table 3.4. We do not observe a significant variation of the aerodynamic coefficients with respect to the radius of the interface. One can notice that the most sensitive quantity is the average drag coefficient. However, the relative variation between the maximum and the minimum value is less than 0.13%, which is completely negligible in a practical application. We can therefore state that, thanks to the accuracy of the developed sliding mesh technique, the results are nearly independent with respect to the position of the sliding interface. This simplifies the mesh generation problem, as the subdivision of the computational domain will be primarily dictated by the geometrical features of the boundary.

r/c	0.625	0.75	1.00	1.50
$\bar{C}_d$	0.4834	0.4837	0.4839	0.4840
$\widehat{C}_l$	2.6246	2.6244	2.6248	2.6247
E	-0.7052	-0.7053	-0.7054	-0.7054

Table 3.4: Variation of the aerodynamic coefficients with respect to r/c

3.3.4 Supersonic flow

We conclude the ellipse case study with a simulation in the supersonic regime. The amplitude of the pitching motion is reduced to $A = 15°$ and a freestream Mach number equal to 2 is considered. The computation is performed using quadratic rational basis functions and the level 3 mesh, illustrated in Fig. 3.7d. The simulation of supersonic flows requires robust numerical algorithms, because of the presence of strong shocks. Furthermore, non-conservative schemes can lead to incorrect estimations of the shock velocity.

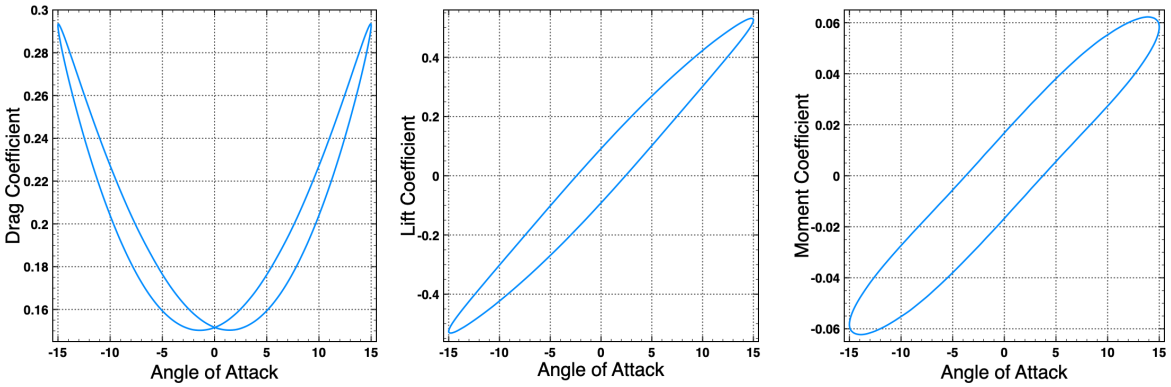


(a) $\alpha = 0°$, ascending phase (b) $\alpha = 15°$ (c) $\alpha = 7.5°$, descending phase

Figure 3.12: Supersonic pitching ellipse flow, density fields

The considered flow is characterised by a bow shock in front of the ellipse. As the angle of attack α oscillates, the shape and intensity of the bow shock evolve. The density fields for

three different values of α are illustrated in Fig. 3.12. As we can observe, the discontinuity is well resolved and the sliding interface does not interfere with the movement of the shock, since there is no loss of conservativity. As in the subsonic case, the solution converges to a stable limit cycle after the initial transient. In Fig. 3.13 we report the limit cycle curves for the aerodynamic coefficients C_l, C_d and C_m. We also estimate $\bar{C}_d$, $\widehat{C}_l$ and E for the supersonic flow condition. The results are: $\bar{C}_d = 0.223$, $\widehat{C}_l = 0.532$, and $E = -0.0146$.



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

Figure 3.13: Supersonic pitching ellipse flow, limit cycle solution

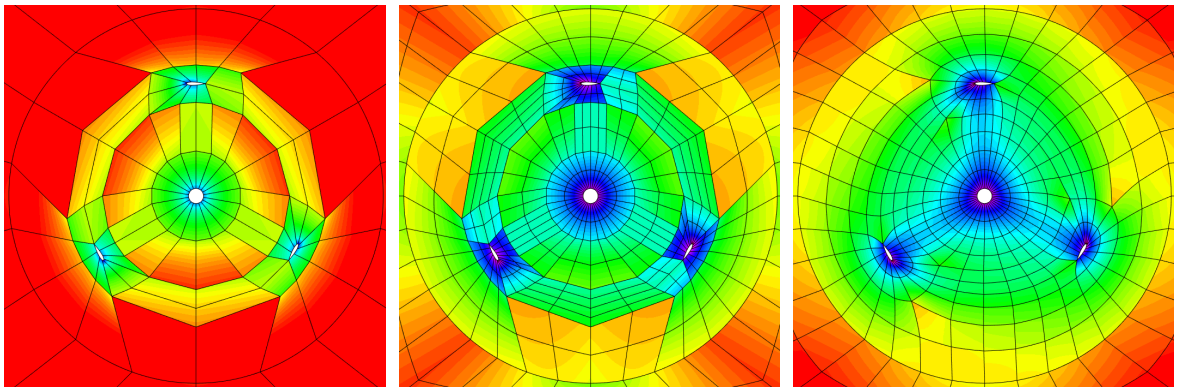
3.4 Application: vertical axis wind turbine

Sliding meshes are commonly employed to simulate rotating machinery such as helicopter rotors (62), axial compressors (65) and wind turbines (101). These kind of applications are characterized by complex geometries and topologies. For this reason the present section is dedicated to the investigation of a more challenging test case. In particular, we consider a Vertical Axis Wind Turbine (VAWT).

3.4.1 Construction of the computational domain

The first challenge of the proposed application is the generation of a suitable mesh to perform the flow simulation. We start by defining the geometry of the turbine. Among the several possible VAWT designs, we consider an H-Darrieus rotor type (102) with three NACA 0018 blades. Being N the number of blades, the radius of the turbine R and the chord of the blades airfoils c are defined by means of the solidity $\sigma = Nc/R$, which is equal to 0.5 for our application. A cylindrical mast of radius $r = 0.2c$ completes the geometry of the rotor. The outer boundaries are sufficiently far from the turbine in order to allow the development of the wake and to avoid unwanted acoustic perturbations due to spurious reflections. As for the

ellipse case study, the far-field boundary conditions are based on Riemann invariants and the rotor surfaces are considered to be adiabatic no-slip walls.



(a) Baseline mesh (b) First refinement step (c) Smoothed mesh

Figure 3.14: Initial VAWT mesh, refinement and smoothing. The colorscale represents the Jacobian of the Isogeometric map.

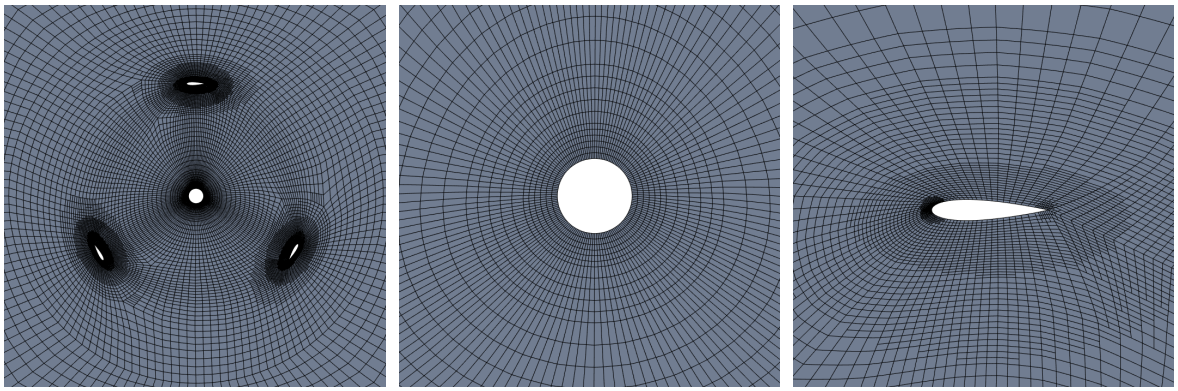
A rational quadratic representation is adopted for the rotor geometry and the sliding interface. The mast is therefore exactly described, whereas the NACA airfoil is approximated via least square fitting. In order to generate the mesh, we exploit the periodic pattern of the geometry. We first create a very coarse block-structured grid, presented in Fig. 3.14a. The baseline grid must respect the constraint of having a uniform subdivision of the sliding interface. Moreover, due the scale differences between the mast radius, the blade chord and turbine diameter, the initial patches are visibly irregular. We illustrate in Fig. 3.14b the mesh obtained after one isotropic refinement step and the Jacobian of the isogeometric map. It can be observed that strong discontinuities in the value of the Jacobian are present between the mesh blocks.

In order to improve the mesh regularity, we adopt a simple elliptic relaxation technique. The developed approach is an adaptation to high-order Bézier grids of the barycentric smoothing algorithm described by Falsafioon et al. (103). In each step of the relaxation procedure, the updated position of the j-th control point is equal to the barycentre of the neighbouring control points:

$$\mathbf{x}_j^{k+1} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i^k, \tag{3.20}$$

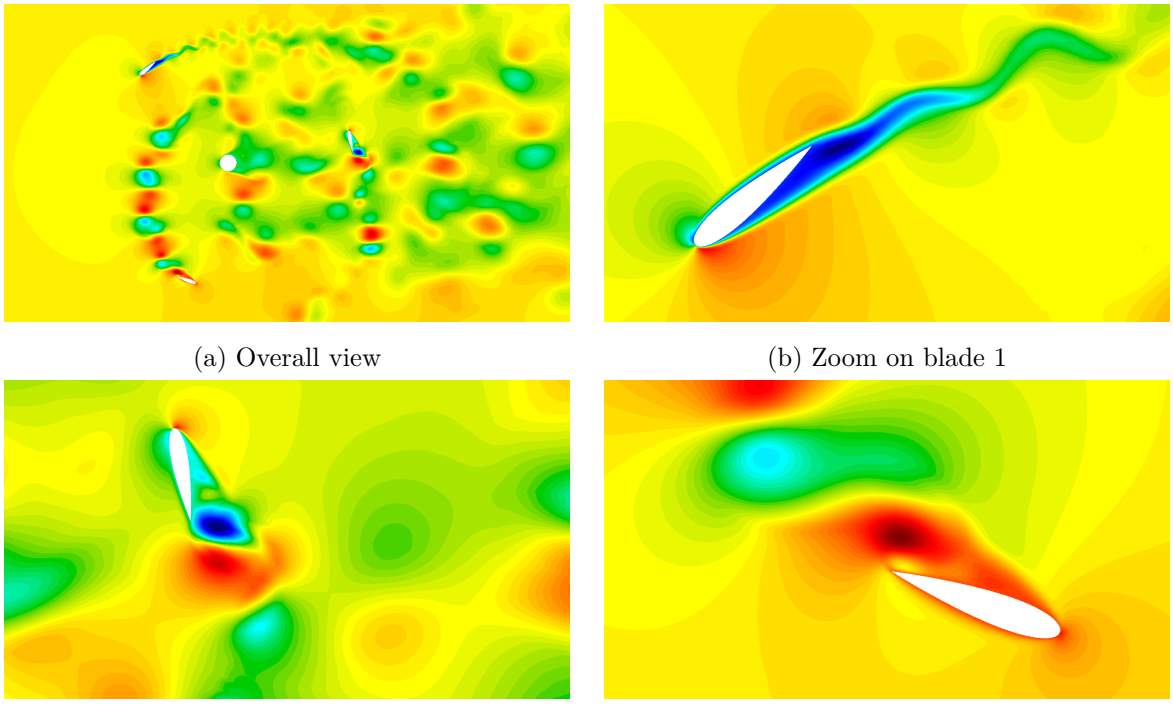
with n being the number of neighbours of $\mathbf{x}_j$. The sequence defined by eq. (3.20) converges towards the solution of the Laplace equation for the coordinate fields. The mesh illustrated in Fig. 3.14c is obtained applying the barycentric smoothing algorithm for 50 iterations starting from the grid of Fig. 3.14b. We notice that the regularity is significantly improved and that the gradient of the Jacobian is globally reduced. The mesh used for the actual computations

is obtained by ulteriorly refining the smoothed grid in the rotor region, in particular around the mast and the blades, as presented in Fig. 3.15. The resulting grid consists of 18545 elements in total.



(a) Final rotor mesh (b) Detail of the mast region (c) Detail of the blade region

Figure 3.15: Final VAWT mesh



(a) Overall view (b) Zoom on blade 1

(c) Zoom on blade 2 (d) Zoom on blade 3

Figure 3.16: VAWT simulation, velocity field

3.4.2 Flow simulation

The typical working conditions of VAWTs are characterized by turbulent flows. Indeed, the blades need to have a sufficiently high lift-to-drag ratio to generate power, meaning that the flow regime of the airfoil has to be at least transitional. Simulating such a flow configuration requires turbulence modelling, high mesh resolution and implicit time integration, in order to keep the computational costs reasonable. For these reasons, the present simulation is carried out in the laminar regime, instead of considering realistic flow conditions.

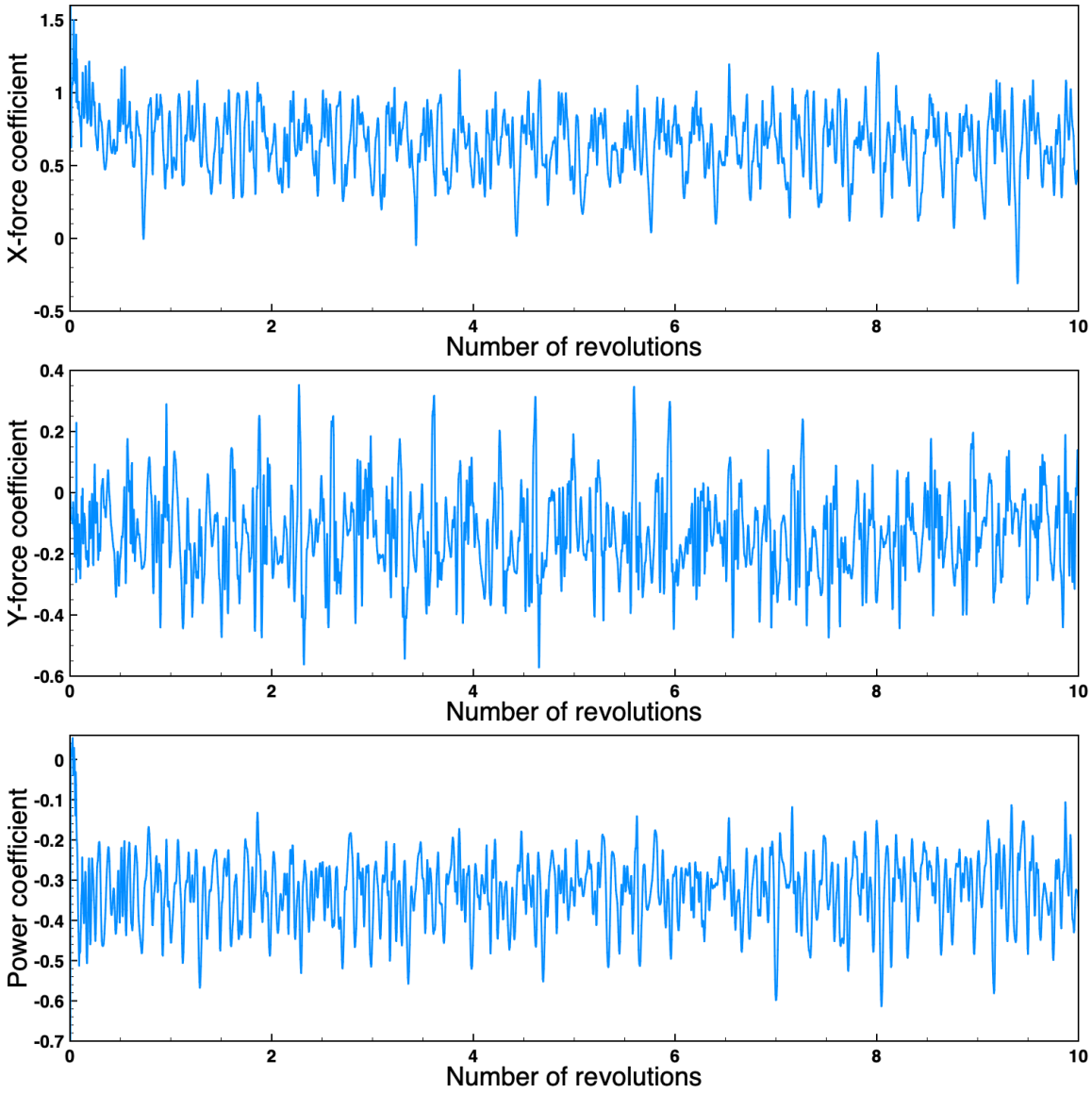


Figure 3.17: VAWT simulation, evolution of the aerodynamic coefficients

The freestream Mach number is equal to 0.1 to avoid unwanted compressibility effects.

The rotational speed ω of the turbine is defined using the tip-speed ratio $\lambda = \omega R/u_\infty$. For the present simulation $\lambda = 2$, meaning that the tangential velocity of the blades is twice the freestream speed. The Reynolds number Re_{rotor} computed using u_∞ and the turbine radius is equal to 3000. In order to estimate the flow regime around the airfoils, we can consider the maximum possible relative velocity of the blades with respect to the freestream condition and compute:

$$Re_{blade} = \frac{c\,(u_\infty + \omega R)}{\nu} = \frac{\sigma}{N}\,(\lambda + 1)\,Re_{rotor} = 1500. \tag{3.21}$$

The flow around the blades is therefore laminar. In a realistic condition $Re_{blade} \approx 10^5$ at least. The result of the laminar regime is the separation of the blades boundary layer very close to the leading edge and the consequent formation of large vortical structures, as it can be observed in Fig. 3.16. As the large structures are advected, interactions occur between the eddies and both the mast and the blades, resulting in a complex chaotic flow. A total of 10 revolutions are simulated. The evolution of the power coefficient C_p and of the aerodynamic force coefficients of the turbine are reported in Fig. 3.17. Due to the chaotic nature of the solution it is not possible to extract a periodic pattern for the C_p curve. Indeed, the interactions between the rotor and the vortical structures generates strong fluctuations in the power coefficient. Also, we can notice that the value of C_p is negative, meaning that the movement of the turbine is absorbing energy. This is a direct consequence of the poor aerodynamic performance of the blades in the laminar regime.

3.5 Conclusion

In this chapter we developed a high-order accurate sliding mesh technique for compressible flows. Then, starting from the general ALE framework developed in chapter 2, we detailed the treatment of the sliding interface. In particular, the properties of the NURBS representations were employed to implement an accurate and fully conservative sliding mesh algorithm.

The proposed approach has been firstly validated on a classic benchmark problem for inviscid flows and optimal convergence rates were observed for all the tested basis degrees. Furthermore, the comparison with an equivalent fixed grid showed that the error introduced by the sliding interface is negligible. As a second test case, we studied the viscous flow around a pitching ellipse, and the obtained results confirmed the precision and robustness of the sliding algorithm. We were able to compare the sliding mesh technique with a previously validated ALE formulation based on mesh deformation. We then showed that the outcome of the simulation is independent with respect to the position of the sliding interface. Lastly, we proved the robustness of the developed methodology considering a supersonic flow condition.

As last case study, we simulated the flow around a 3-blade VAWT configuration, allowing us to evaluate the potential of the proposed approach on a more complex geometry. We

especially focused on the construction of the computational grid, starting from a set of coarse Bézier patches. In order to obtain a better quality mesh, a smoothing algorithm had to be employed, showing that the use of ideas taken from unstructured grid generation can be beneficial in the context of isogeometric analysis as well.

Chapter 4

Aerodynamic optimisation of a morphing airfoil

Thanks to the inherent coupling between geometry and simulation, Isogeometric Analysis constitutes the ideal framework for shape optimisation problems. So far, the main applications field of Isogeometric optimisation has been structural mechanics (104; 105; 106). In the context of fluid dynamics, only a few examples of shape optimisation studies are available in the literature. Nørtoft et al. (107) presented the first results on simple steady laminar flows combining a spline discretization of the incompressible Navier-Stokes equations with a gradient descent algorithm. A low fidelity aeroelastic optimisation method has been proposed in (108), relying on a potential flow model solved using an Isogeometric Boundary Element Method. Lastly, Wang et al. (109) adopted the Isogeometric DG method to solve Euler equations and optimise airfoil shapes using an adjoint technique.

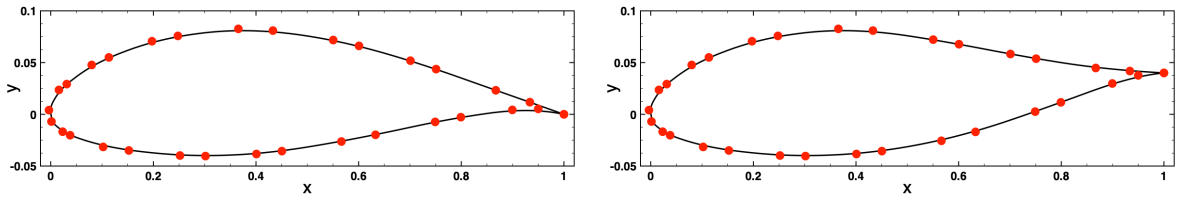
In the recent years, a revived interest in bio-inspired flight mechanisms has been observed. The fixed wing design is indeed well adapted to high-speed flying vehicles of large dimension, but, with the development of Micro Aerial Vehicles (MAV), different solutions are being explored to improve the aerodynamic performance in the low Reynolds regime. To this end, Platzer et al. (110) conducted an aerodynamic analysis of multiple flapping wing configurations. A bat-inspired drone design with flexible multi-body wings has been studied by Joshi et al. (111). An increasingly popular bio-inspired technique is the aerodynamic morphing, which can be defined as a smooth adaptation of the wing shape. The morphing concept takes inspiration from the birds' feathers, thanks to which birds can control the geometry of their wings. Thanks to morphing, it is possible to obtain optimal aerodynamic performance for multiple flight conditions. For this reason, several morphing prototypes have been studied over the years (112). A multi-point optimisation study of a morphing airfoil has been carried out by Fincham et al. (113). The benefits of trailing edge morphing for a transonic wing design have been proved in (114). A leading edge morphing actuation has been studied by

Kang et al. (115) for laminar flow conditions. Lastly, promising results have been obtained by using a vibrating trailing edge to control the wake of an A320 wing section (116; 117; 118).

In the present chapter we investigate a trailing edge morphing technique to control the boundary layer separation in the laminar regime. The first aim of the study is to prove the benefits of using the Isogeometric framework for flow problems with deforming geometries. We therefore exploit the properties of NURBS to develop an accurate morphing model and deform the computational mesh accordingly. The second goal of the investigation is to develop a proof-of-concept aerodynamic design chain entirely based on the CAD representation. The Isogeometric ALE-DG is thus combined with a Bayesian optimisation algorithm to find the best suitable morphing configuration with respect to a given criterion. Since a unique geometry description is adopted in all the stages of the design process, the proposed approach is particularly efficient and presents a high degree of automation. The chapter is organised in the following manner: section 4.1 is dedicated to the description of the morphing model. Then, the considered airfoil is characterised and a mesh sensitivity analysis is performed. In section 4.4 we describe the optimisation algorithm and the design process in its entirety. Two single-objective optimisation studies are then performed in sections 4.5 and 4.6. Lastly, a multi-objective example is illustrated in section 4.7.

4.1 Morphing model

The study presented in this chapter is carried out considering a NACA $63_1 - 412$ airfoil. The NACA 6-series of airfoils is widely employed for aircraft wings and wind turbine blades (119) and it was originally designed with the aim of maximizing the region of laminar flow around the airfoil. The considered geometry is characterized by a maximum thickness equal to 12% of the chord length. Using least-square fitting, we obtain a B-Spline description of the NACA $63_1 - 412$ airfoil, as reported in Fig. 4.1a. We adopt a knot vector defined by 15 non-zero knot spans and a cubic representation. The multiplicity of each internal knot vector is equal to 2, meaning that the obtained B-Spline curve is $\mathcal{C}^1$.



(a) Normal (b) Morphed

Figure 4.1: B-Spline representation of the NACA $63_1 - 412$ airfoil

Starting from the B-Spline representation, it is possible to define the modes of deformation

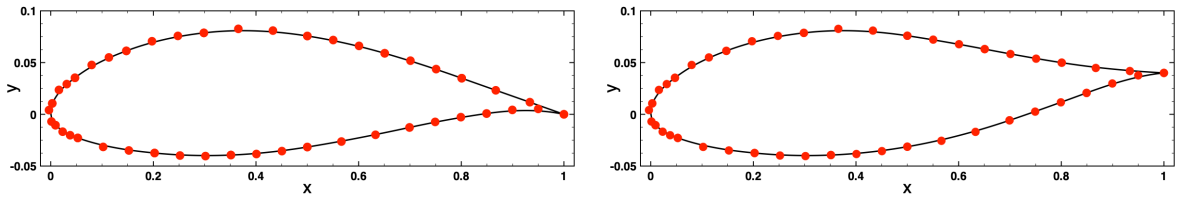
of the airfoil. A general technique would consist in choosing to move each control point independently without a global movement law. However, the number of degrees of freedom would be too large and the cost of the optimisation would be unreasonable with such approach. For this reason, we impose a predefined global deformation function. Similarly to the work of Simiriotis et al. (116), we adopt a quadratic morphing law. The movement of each control point of the B-Spline airfoil is given by:

$$\begin{cases} x_i^s(t) = x_{i,0}^s + a_{x,i}^s \sin(2\pi f t), \\ y_i^s(t) = y_{i,0}^s + a_{y,i}^s \sin(2\pi f t), \end{cases} \tag{4.1}$$

where $\mathbf{x}_{i,0}^s = (x_{i,0}^s, y_{i,0}^s)^T$ are the control points of the original B-Spline, f is the frequency, $a_{x,i}^s = 0$ and $a_{y,i}^s$ is determined by:

$$a_{y,i}^s = \begin{cases} 0, & \text{if } x_{i,0}^s < x_0 \\ A\left(\dfrac{x_{i,0}^s - x_0}{1 - x_0}\right)^2, & \text{if } x_{i,0}^s \geq x_0 \end{cases} \tag{4.2}$$

where A is the amplitude of the oscillation of the trailing edge and $x_0 = 1 - L$, with L being the controlled fraction of the airfoil surface. Using the proposed deformation law, the morphing is controlled by just 3 parameters: L, A and f. An example of morphed B-Spline airfoil is illustrated in Fig. 4.1b.



(a) Normal (b) Morphed

Figure 4.2: Bézier representation of the NACA $63_1 - 412$ airfoil

The Bézier extraction is then employed to obtain a DG compliant representation of the airfoil:

$$\mathbf{x}_0^b = \mathbf{B}\,\mathbf{x}_0^s, \tag{4.3}$$

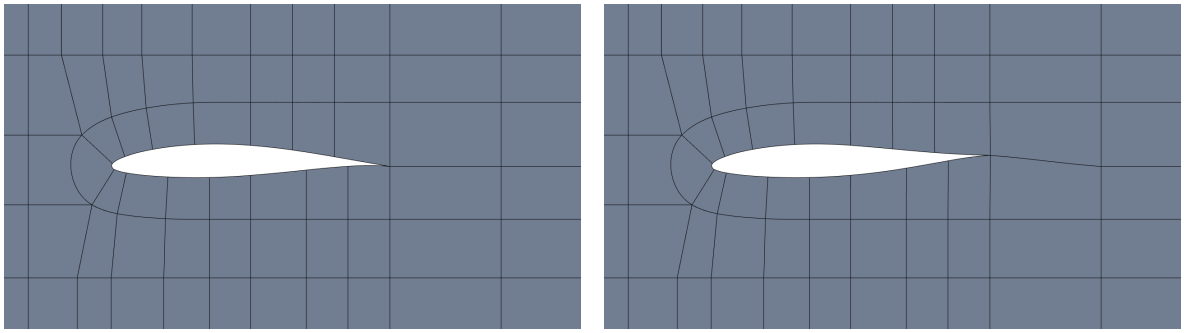
where $\mathbf{x}_{i,0}^b = (x_{i,0}^b, y_{i,0}^b)^T$ are the control points of the Bézier airfoil, presented in Fig. 4.2a and $\mathbf{B}$ is the extraction operator. The morphing of the Bézier geometry is thus given by:

$$\begin{cases} x_i^b(t) = x_{i,0}^b + a_{x,i}^b \sin(2\pi f t), \\ y_i^b(t) = y_{i,0}^b + a_{y,i}^b \sin(2\pi f t), \end{cases} \tag{4.4}$$

where the amplitudes are computed using the same extraction operator:

$$\mathbf{a}^b = \mathbf{B}\,\mathbf{a}^s. \tag{4.5}$$

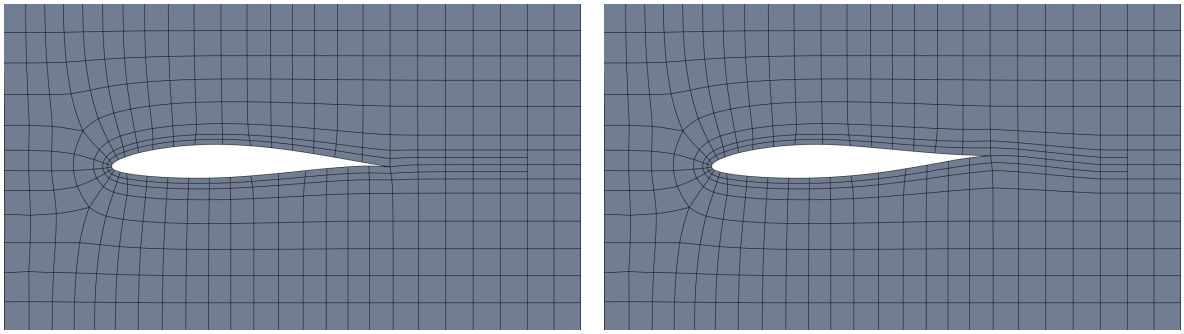
The morphed Bézier airfoil computed with this approach is reported in Fig. 4.2b. We remark that, since the extraction operation preserves the B-Spline geometry, the Bézier curve of the morphing airfoil is $\mathcal{C}^1$ at all times, as we explained in chapter 2.



(a) Normal (b) Morphed

Figure 4.3: Morphing of the baseline mesh

We construct a very coarse C-H type mesh using the extracted Bézier airfoil. The mesh is illustrated in Fig. 4.3a and contains 65 cubic elements. The morphing movement is transferred from the boundary to the mesh using cubic functions, defined such that only the first layer of patches around the airfoil is deformed, as shown in Fig. 4.3b. Since the baseline discretization is coarse, the elements are easily deformed, without incurring in inadmissible shapes.



(a) Normal (b) Morphed

Figure 4.4: Morphing of the refined mesh

In order to perform the flow simulations, the baseline mesh needs to be refined. The control points of the final grid are computed by recursive splitting of the baseline elements:

$$\mathbf{x}_0 = \mathbf{R}\,\mathbf{x}_0^b. \tag{4.6}$$

Therefore, the movement of the refined mesh is given by:

$$\mathbf{x}(t) = \mathbf{x}_0 + \mathbf{a}\sin\left(2\pi f t\right), \tag{4.7}$$

with:

$$\mathbf{a} = \mathbf{R}\,\mathbf{a}^b. \tag{4.8}$$

Hence, the control point velocities required for the ALE formulation are:

$$\mathbf{v}_g(t) = 2\pi f \mathbf{a}\,\sin\left(2\pi f t\right). \tag{4.9}$$

In Fig. 4.4 it is possible to observe an example of morphing for a sample refined mesh. We can compare Fig. 4.4b with Fig. 4.3b to notice that the geometry is perfectly preserved. We remark that, in the practical implementation, the operators $\mathbf{B}$ and $\mathbf{R}$ are never explicitly computed. The extraction and refinement procedures are carried out by recursive application of the knot insertion formula (1.18). The amplitude field $\mathbf{a}$ is thus computed during the mesh generation process and then stored as a DG field to be used by the flow solver.

4.2 Airfoil characterisation

We consider a subsonic laminar flow condition in which the freestream Mach number is equal to 0.2 and the Reynolds number with respect to the airfoil chord is equal to 5000. In order to choose a suitable flow configuration to control using the morphing technique, we first characterise the airfoil for different values of the angle of attack α. In particular, we trace the curves of the aerodynamic coefficients with respect to α and we observe the variations of the boundary layer and wake structures.

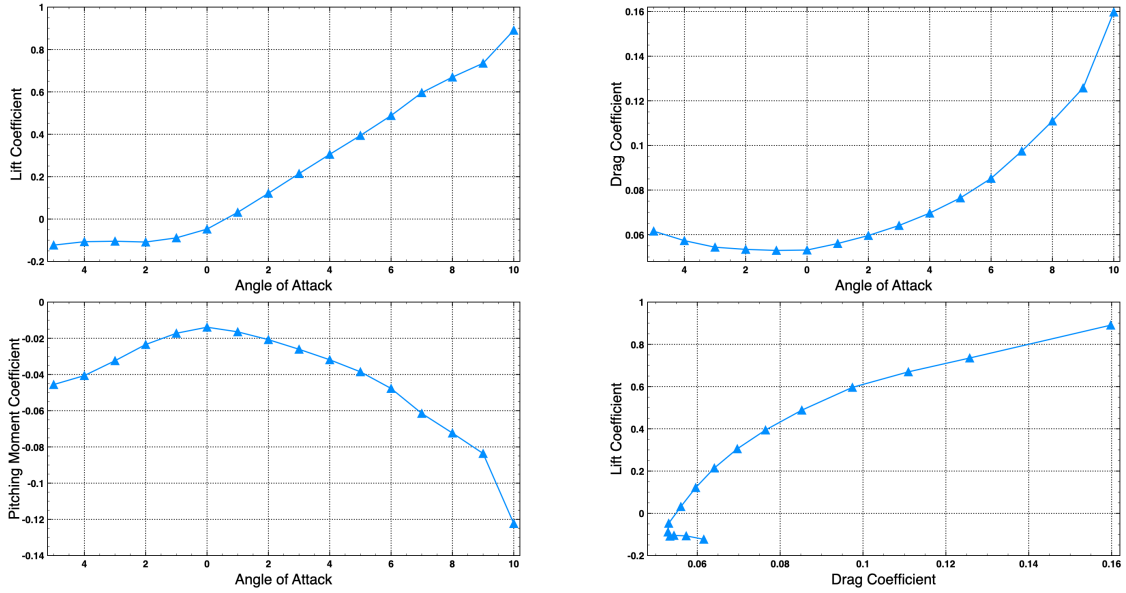
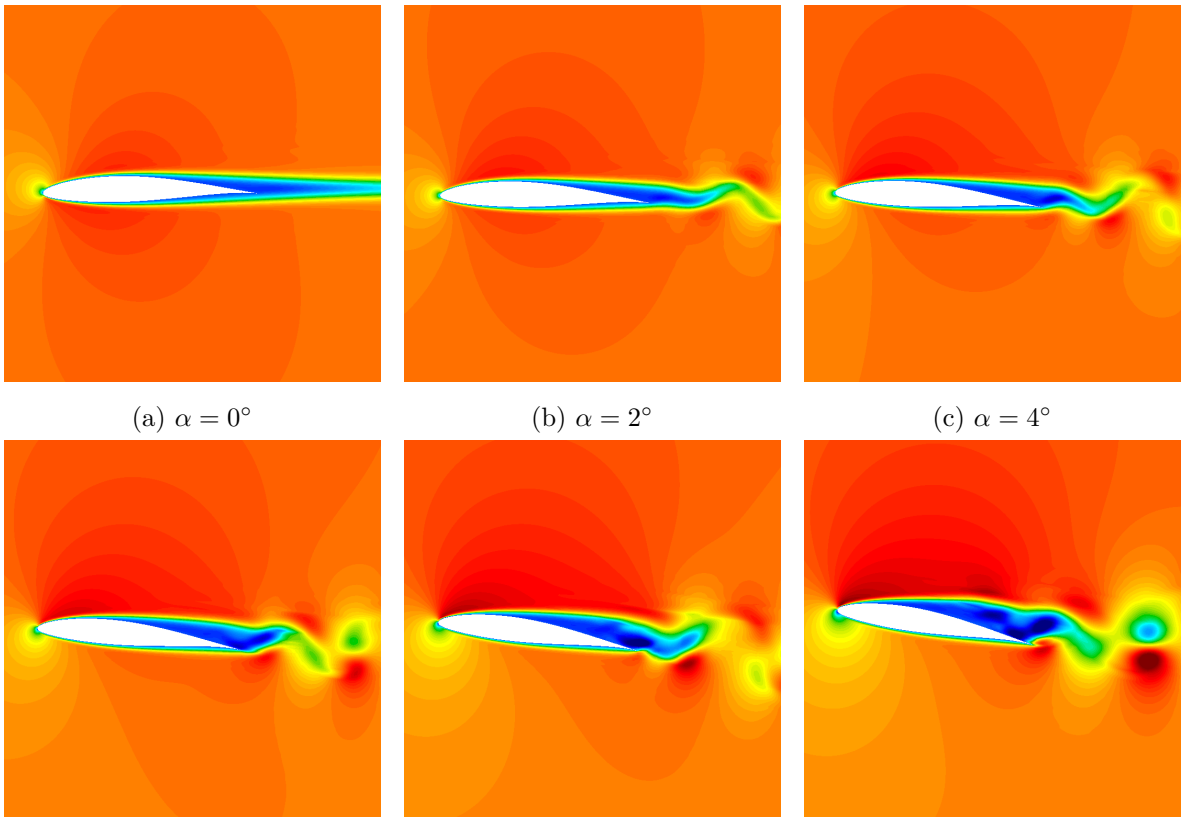


Figure 4.5: Airfoil characterization study, aerodynamic coefficient curves

Since the aim of the characterisation study is to have a rough estimate of the behaviour of the airfoil, we adopt a coarse grid to reduce the computational time. The mesh employed

for this study is illustrated in FIg. 4.4a. We consider values of α between $-5°$ to $15°$, with a step of $1°$. The curves of the aerodynamic coefficients are reported in Fig 4.5 together with the polar curve. The velocity fields for some sample values of α are illustrated in Fig. 4.6. For negative values of α the behaviour of the airfoil is totally non-linear. We can observe that at zero incidence the lift coefficient C_l is negative and that the flow is already separated. The recirculation bubble is stable and vortex shedding does not occur. For positive α the growth of C_l is linear even if the flow is separated. We can observe that the boundary layer separation point progressively moves towards the leading edge as α increases. The recirculation bubble is unstable and a periodic vortex shedding is observed. Non-linearities start to appear in the $C_l - \alpha$ curve for $\alpha \geq 8°$ as the recirculation region becomes larger. The C_d curve shows the characteristic quadratic growth for positive angles of attack. In terms of lit-to-drag ratio, the best performance is found for α between $2°$ and $4°$. For higher values of the angle-of-attack the growth of C_d is significantly faster than the linear increase of the lift coefficient.



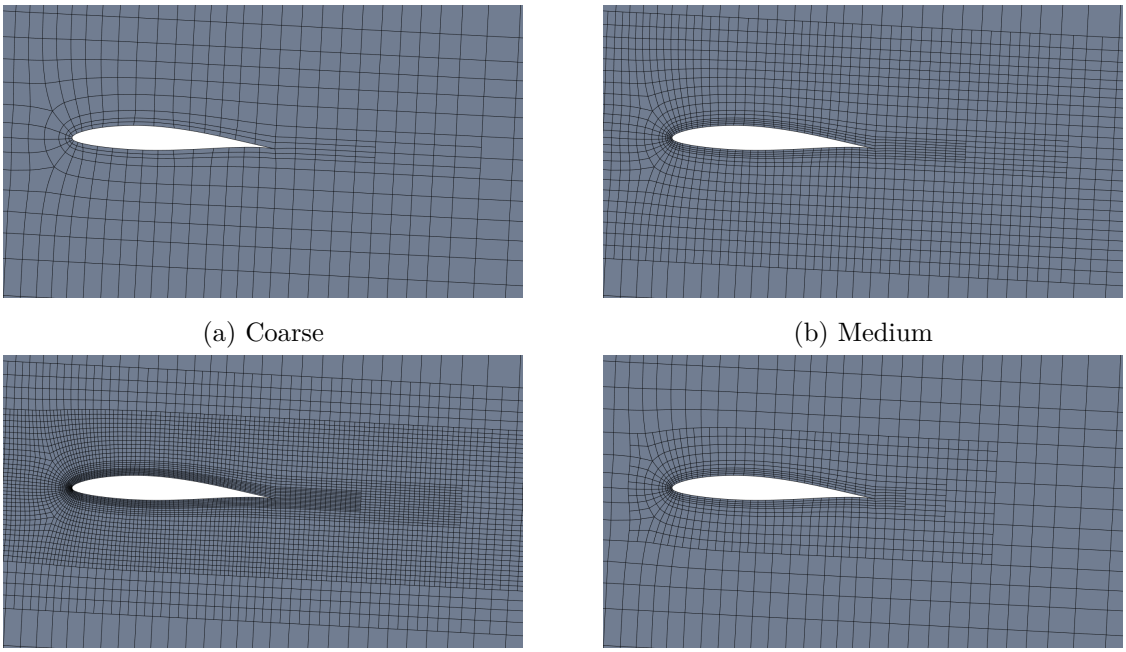
(a) $\alpha = 0°$ (b) $\alpha = 2°$ (c) $\alpha = 4°$

(d) $\alpha = 6°$ (e) $\alpha = 8°$ (f) $\alpha = 10°$

Figure 4.6: Influence of the angle of attack on the velocity field

4.3 Mesh sensitivity study

Since the flow is already separated for low value of α and the region of maximum aerodynamic performance is within 2° and 4°, we adopt an angle of attack of 3° to investigate the morphing technique. Before running the optimisation, we carry out a mesh sensitivity study. Starting from the baseline mesh illustrated in Fig. 4.3a, we obtain four meshes by refinement of the initial Bézier patches. In order to increase the regularity of the refined grid, we employ the barycentric smoothing algorithm presented in chapter 3.



(a) Coarse (b) Medium

(c) Fine (d) Optimised

Figure 4.7: Refinement levels for mesh sensitivity study

N_{el}	$\bar{C}_l$	$\bar{C}_d$	$\bar{C}_m$	St
1810	0.2145	0.0641	-0.0261	1.9366
3512	0.2212	0.0644	-0.0266	1.9843
7864	0.2225	0.0645	-0.0267	2.0009
2414	0.2215	0.0645	-0.0267	1.9858

Table 4.1: Rigid airfoil, convergence of the aerodynamic coefficients

The four meshes are presented in Fig. 4.7. The coarse, medium and fine grids are hierarchically computed and contain 1810, 3512 and 7864 elements respectively. The optimised mesh is characterised by the same resolution level of the medium mesh around the airfoil with a much coarser discretization of the far wake area. The optimised mesh contains 2414

elements. We first conduct the analysis considering the rigid airfoil, without morphing. We compute for each grid the average values of the lift, drag and pitching moment coefficients and the Strouhal number. The results are reported in table 4.1. The considered quantities are already well estimated using the coarse mesh. The results obtained with the optimised grid are in line with those predicted with the medium refinement level.

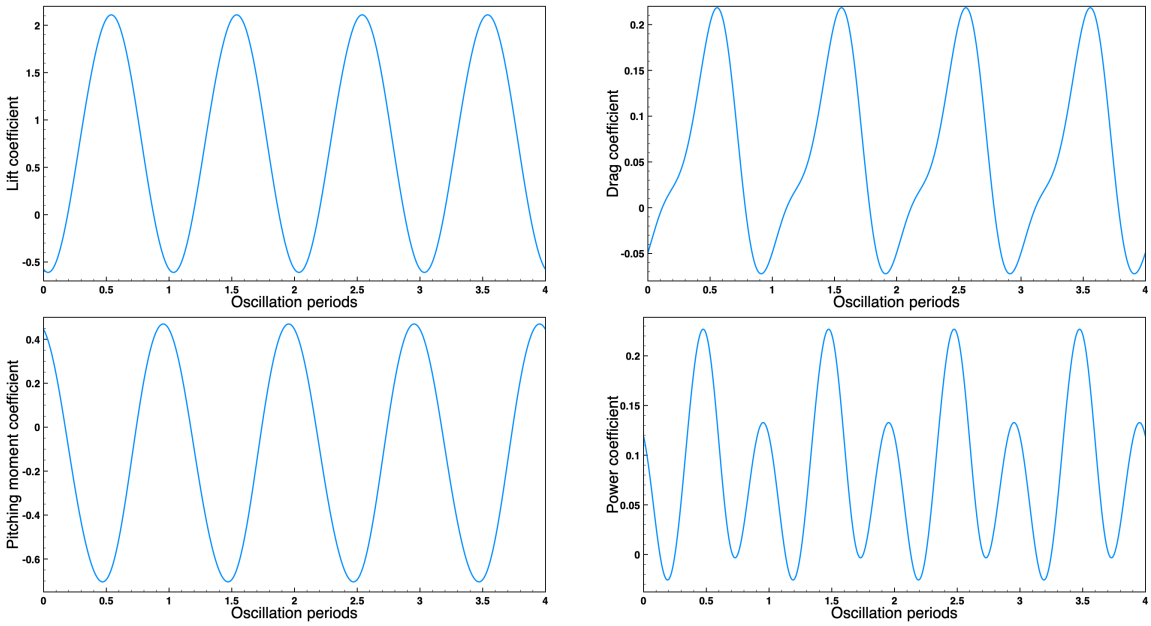


Figure 4.8: Morphing airfoil, evolution of the aerodynamic coefficients

The mesh sensitivity analysis is then repeated considering the trailing edge morphing. For this analysis, the controlled fraction of the airfoil surface is equal to 50%, the morphing frequency is equal to the shedding frequency and the amplitude is equal to a third of the airfoil thickness. The control parameters are thus $L = 0.5$, $A = 0.04$, $f = 0.4$, considering a unitary chord length. After the initial transient, a periodic flow condition is found. We report in Fig. 4.8 the evolution of the aerodynamic coefficients for the last 4 computed oscillation periods. The power coefficient C_P quantifies the amount of power necessary actuate the morphing, and it is defined as:

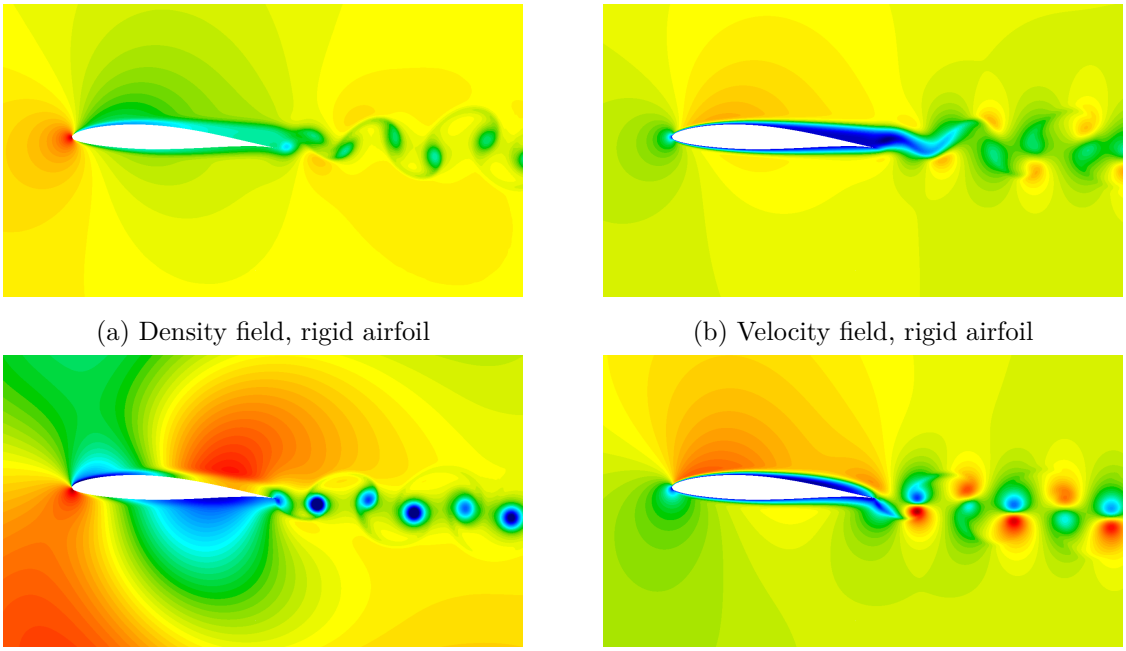
$$C_P = \frac{1}{\frac{1}{2}\rho_\infty U_\infty^3 c} \oint_{\partial\Omega_a} (p\,\mathbf{n} - \tau \cdot \mathbf{n}) \cdot \mathbf{V}_g \, \mathrm{d}\Gamma, \tag{4.10}$$

where p is the pressure, τ is the viscous stress tensor and $\partial\Omega_a$ is airfoil surface. Using the 4 last converged oscillation periods we compute the average values of the aerodynamic coefficients, which are reported in table 4.2 for each mesh. As for the rigid airfoil, the flow solver is already capable of predicting the correct flow physics using the coarse mesh and a sufficiently accurate estimate of the quantities of interest. The results computed with the optimised mesh

are in line with the medium mesh, except for the average lift coefficient $\bar{C}_l$, which is slightly underestimated. This effect may be caused by the reduced resolution in the wake region.

N_{el}	$\bar{C}_l$	$\bar{C}_d$	$\bar{C}_m$	$\bar{C}_P$	T_c
1810	0.7471	0.0564	-0.1225	0.0859	197
3512	0.7420	0.0572	-0.1204	0.0855	872
7864	0.7438	0.0577	-0.1205	0.0844	4301
2414	0.7393	0.0572	-0.1200	0.0855	656

Table 4.2: Morphing airfoil, mesh sensitivity study



(a) Density field, rigid airfoil

(b) Velocity field, rigid airfoil

(c) Density field, morphing airfoil

(d) Velocity field, morphing airfoil

Figure 4.9: Comparison of rigid and morphing airfoil flows, fine grid

Comparing the aerodynamic coefficients for the rigid and the morphing airfoil, we can notice that the average lift and pitching moment are significantly increased, whereas the drag is slightly reduced. However, strong oscillations are observed in the instantaneous lift and the minimum value of C_l is negative. In Fig. 4.9, a comparison of the density and velocity fields for the rigid and morphing airfoil is illustrated. A significant difference in the separation of the boundary layer is found. For the morphing airfoil the recirculation bubble is concentrated in a thin region close to the wall, and a reattachment of the boundary layer is observed at the trailing edge. As a consequence of the behaviour of the boundary layer, the vortical structures in the wake are significantly different.

As we previously mentioned, the goal of the mesh sensitivity analysis is the determination of a suitable mesh to compute the objective functions for the optimisation. From the results of table 4.2, we could assume that the coarse mesh provides a sufficiently reliable estimate of the aerodynamic coefficients. However, for some combinations of the morphing parameters, the simulation performed with the coarse mesh does not converge, due to a lack of resolution of the boundary layer close to the leading edge. The medium grid has the adequate resolution close to the wall, but the flow computation would be too expensive. The computational costs are reported in table 4.2 in terms of total CPU time T_c, expressed in hours. It can be observed that, with the optimised mesh, the computational time is reduced by 25% compared to the medium refinement level, while maintaining the same wall resolution. Since the relative difference between the C_l computed with the two grids is less than 1%, we conclude that the optimised mesh provides a reasonable compromise between accuracy, reliability and computational cost.

4.4 Optimisation algorithm

In this chapter, we deal with optimisation problems of the following form:

$$\begin{aligned} \min_{\mathbf{z}\in\mathcal{D}} \quad & g(\mathbf{z}), \\ \text{subject to} \quad & h(\mathbf{z}) \leq 0, \end{aligned} \tag{4.11}$$

where g is the cost, or objective, function and h is the constraint function. The design variables $\mathbf{z}$ constitute a d-dimensional vector contained in the design space $\mathcal{D} \subset \mathbb{R}^d$.

In aerodynamic optimisation problems, the relation between the design parameters and the objective function cannot be expressed in closed-form. Using gradient-based minimisation algorithms is therefore not straightforward. Adjoint techniques are commonly employed to compute the flow sensitivities with respect to the design variables (120; 121; 122). The drawbacks of the adjoint framework are represented by the complexity of the implementation and the heavy memory usage due to the backward time integration of the adjoint equations for time-dependent flows (123). Moreover, flow control problems often present non-linear behaviours, which may lead to multi-modal cost functions. In this context, it is desirable to adopt a global optimisation technique. A possible approach would be the use of Genetic Algorithms (GA) (124). Although very accurate, GAs require a high number of evaluations to converge. For this reason, the present work relies on the Efficient Global Optimisation (EGO) framework of Jones et al. (125; 126). In the recent literature, it is common to refer to EGO as Bayesian optimisation (127).

The EGO method belongs to the family of metamodel-based optimisation techniques (128), in which a surrogate model is used to approximate the flow response as a function of the design

variables. In particular, the EGO is built upon a probabilistic description and the associated metamodel is computed using a Gaussian process regression, also known as Kriging. Thanks to the probabilistic representation, Kriging also provides an estimate of the uncertainty of the surrogate model. In order to select a set of parameters to carry out the successive evaluations, a merit function is employed. The metamodel is then recalibrated using the new evaluation and the process is repeated until convergence is reached. Thanks to their computational efficiency and ability of finding global optima, Kriging-based algorithms have been successfully employed in flow control applications (129; 130) and aerodynamic design (131). As the present manuscript is not dedicated to the development of EGO algorithms, we explain in the present section only the main features. For a more detailed discussion, the reader may refer to (126; 127).

4.4.1 Kriging model

Kriging is a powerful non-linear regression method with a probabilistic formulation. Supposing that a set of observation $G_n = (g_1, g_2, \ldots, g_n)$ are available at the corresponding points $Z_n = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n)$, with $\mathbf{z}_i \in \mathcal{D}$. We assume that the response G_n is the realisation of a Gaussian process with the following likelihood function:

$$p(G_n) = \frac{\exp\left(-\frac{1}{2} G_n^T \mathbf{C}_n^{-1} G_n\right)}{\sqrt{(2\pi)^n \, |\mathbf{C}_n|}}, \tag{4.12}$$

where $\mathbf{C}_n \in \mathbb{R}^{n \times n}$ is the correlation matrix, whose terms are computed by means of a covariance kernel, which we express here as:

$$C_{i,j}(\mathbf{h}, \Theta) = \sigma^2 \prod_{k=1}^{d} c(h_k, \theta_k), \tag{4.13}$$

where $c(h, \theta)$ is the one-dimensional correlation function, $\mathbf{h} = \mathbf{z}_i - \mathbf{z}_j$, σ is the standard deviation of the process, and $\Theta = (\theta_1, \theta_2, \ldots, \theta_d)$ is the set of hyperparameters. Several functions can be employed for the one-dimensional covariance kernel (132). In this work, we adopt the Matérn class function with $\nu = 5/2$:

$$c(h, \theta) = \left(1 + \frac{\sqrt{5}|h|}{\theta} + \frac{5h^2}{3\theta^2}\right) \exp\left(-\frac{\sqrt{5}|h|}{\theta}\right). \tag{4.14}$$

The hyperparameters are computed by maximising the probability density $p(G_n)$, which is equivalent to minimising the log-likelihood function (126):

$$\mathcal{L} = G_n^T \mathbf{C}_n^{-1} G_n + \log |\mathbf{C}_n|, \tag{4.15}$$

ignoring the constant terms in eq. (4.12). For the construction of Kriging models, we employ the **DiceKriging** library[1]. For more details on the algorithms, the reader can refer to Roustant et al. (132).

[1] `https://cran.r-project.org/web/packages/DiceKriging`

The Kriging model is then used as a predictor to estimate the value of g for points of the domain $\mathcal{D}$ that are not contained in the set of observations. Considering a point x_{n+1}, the vector $G_{n+1} = (g_1, g_2, \ldots, g_n, g_{n+1})$ is still assumed to be the realisation of a Gaussian process with likelihood function equal to:

$$p(G_{n+1}) = \frac{\exp\left(-\frac{1}{2} G_{n+1}^T \mathbf{C}_{n+1}^{-1} G_{n+1}\right)}{\sqrt{(2\pi)^{n+1} \left|\mathbf{C}_{n+1}\right|}}, \tag{4.16}$$

where the new correlation matrix can be written as:

$$\mathbf{C}_{n+1} = \begin{bmatrix} \mathbf{C}_n & k \\ k^T & \kappa \end{bmatrix}, \tag{4.17}$$

where $\kappa = C_{n+1,n+1}$ and k is a column vector with components $k_j = C_{j,n+1}$. Using the conditional probability rule, the likelihood of g_{n+1}, knowing the observations G_n, is equal to:

$$p(g_{n+1}|G_n) = \frac{p(G_{n+1})}{p(G_n)}. \tag{4.18}$$

It is thus possible to show that the resulting probability density function is Gaussian (126; 130):

$$p(g_{n+1}|G_n) = \frac{1}{\sqrt{2\pi}\,\widehat{\sigma}_{n+1}} \exp\left[-\frac{(g_{n+1} - \widehat{g}_{n+1})^2}{2\,\widehat{\sigma}_{n+1}^2}\right] \tag{4.19}$$

with mean value $\widehat{g}_{n+1}$:

$$\widehat{g}_{n+1} = k^T C_n^{-1} G_n, \tag{4.20}$$

and standard deviation $\widehat{\sigma}_{n+1}$:

$$\widehat{\sigma}_{n+1} = \kappa - k^T C_n^{-1} k. \tag{4.21}$$

The Kriging predictor thus provides an estimate of the value $g(\mathbf{z}_{n+1})$, which corresponds to the mean value $\widehat{g}_{n+1}$, and a measure of the uncertainty, given by the standard deviation $\widehat{\sigma}_{n+1}$. Under suitable hypothesis, the real value of $g(\mathbf{z}_{n+1})$ is contained in a range of $3\widehat{\sigma}_{n+1}$ from $\widehat{g}_{n+1}$ with a 99.7% probability. We hence designate $[\widehat{g} - 3\widehat{\sigma}, \widehat{g} + 3\widehat{\sigma}]$ as the confidence interval.

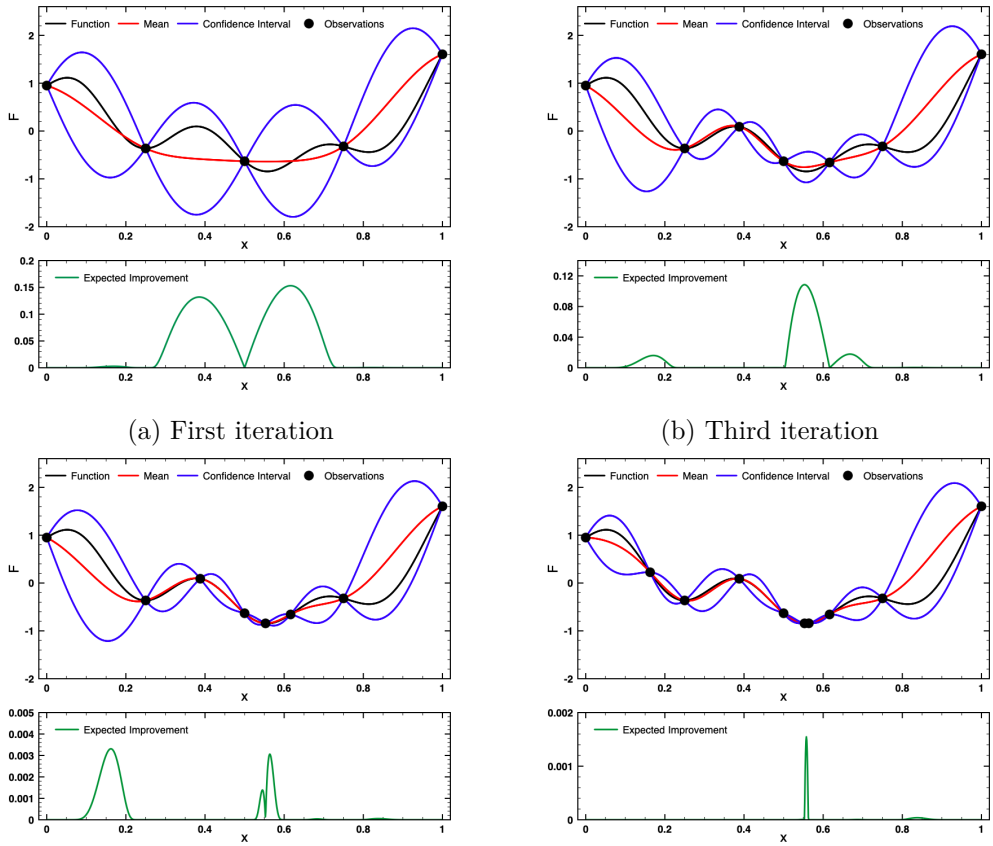
4.4.2 Merit functions

The constructed stochastic metamodel is then used to steer the optimisation process. It can be shown that just using the information about the mean value $\widehat{g}(\mathbf{z})$ does not guarantee the convergence towards the global optimum (126). Indeed, Torn et al. (133) demonstrated that the sequence of evaluations must be dense in order to achieve global convergence. This can be achieved by taking into account the uncertainty of the model to drive the search of the

optimal design. Therefore, both the mean and the standard deviation are used to compute a merit function, which is either maximised or minimised to select the next evaluation point. The most immediate approach consists in minimising a statistical lower bound, defined as:

$$LB(\mathbf{z}) = \widehat{g}(\mathbf{z}) - \beta\widehat{\sigma}(\mathbf{z}), \tag{4.22}$$

with β being a positive constant. The lower bound has been successfully employed in fluid dynamics applications(134; 129). However, Jones (126) provided a counterexample in which the lower bound technique fails.



(a) First iteration

(b) Third iteration

(c) Fourth iteration

(d) Sixth iteration

Figure 4.10: Example of EGO on a 1D function

A more robust methodology consists in maximising the Expected Improvement (126), as shown in the example in Fig. 4.10. Considering a random variable $G(\mathbf{z})$, the improvement is defined as:

$$I = \max(g_{min} - G(\mathbf{z}), 0), \tag{4.23}$$

where g_{min} is the minimum observed value of the cost function. Since we are considering a

Gaussian process model, the probability density function of the improvement is:

$$p(I) = \frac{1}{\sqrt{2\pi}\,\widehat{\sigma}(\mathbf{z})} \exp\left\{ -\frac{[g_{min} - I - \widehat{g}(\mathbf{z})]^2}{2\,\widehat{\sigma}^2(\mathbf{z})} \right\}. \tag{4.24}$$

Thus, the Expected Improvement is:

$$EI = \int_0^\infty I\,p(I)\,\mathrm{d}I = \int_0^\infty \frac{I}{\sqrt{2\pi}\,\widehat{\sigma}(\mathbf{z})} \exp\left\{ -\frac{[g_{min} - I - \widehat{g}(\mathbf{z})]^2}{2\,\widehat{\sigma}^2(\mathbf{z})} \right\} \mathrm{d}I. \tag{4.25}$$

After integration by parts, eq. (4.25) reduces to:

$$EI = \widehat{\sigma}(\mathbf{z})\,\big[u\Phi(u) + \Psi(u)\big], \tag{4.26}$$

with:

$$\Phi(u) = \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{u}{\sqrt{2}}\right)\right], \quad \Psi(u) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right), \quad u = \frac{g_{min} - \widehat{g}(\mathbf{z})}{\widehat{\sigma}(\mathbf{z})}. \tag{4.27}$$

In the example of Fig. 4.10 it is possible to observe that the EI effectively takes into account both the mean value and the standard deviation, leading to a dense sequence of evaluations. Under some mild assumptions on the covariance kernel, Vazquez et al. (135) effectively proved that the EI approach converges towards the global optimum.

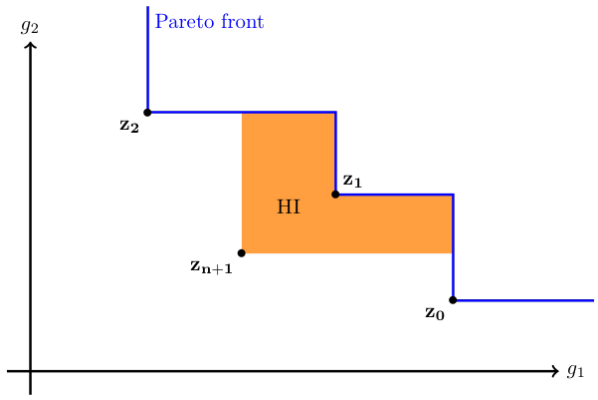


Figure 4.11: Hypervolume improvement criterion

The Expected Improvement function does not take into account the constraints. However, the approach can be easily generalized for constrained optimisation (125). The cost and constraint responses are treated as independent stochastic processes. Therefore, two separated Kriging models are computed. The merit function for the constrained problem is the product of the EI and the probability of respecting the constraint:

$$EFI(\mathbf{z}) = EI(\mathbf{z})\,P\big(h(\mathbf{z}) \leq 0\big), \tag{4.28}$$

where EFI stands for Expected Feasible Improvement. The computation and maximisation of the EFI function is carried out with the **DiceOptim** package[2] of Roustant et al. (132).

The methodology can be extended to multi-objective optimisation as well. We consider, for the sake of simplicity, an unconstrained multi-objective problem of the form:

$$\min_{\mathbf{z}\in\mathcal{D}} \left(g_1(\mathbf{z}),\, g_2(\mathbf{z}),\, \ldots,\, g_m(\mathbf{z})\right). \tag{4.29}$$

Similarly to the constrained optimisation algorithm, the m objective responses are considered as uncorrelated processes, therefore m independent Kriging models are constructed. Since it is usually not possible to minimise all the objectives at the same time, the goal of multi-objective optimisation is to find the Pareto front, defined as the set of non-dominated points. Starting from the available observations, an initial Pareto front is computed. As represented in Fig. 4.11, when a new non-dominated point is added, the feasible objective space is enlarged. It is then possible to evaluate the hypervolume improvement, defined as the measure of the increase in the feasible region, illustrated in orange in Fig. 4.11. Similarly to the single-objective EGO, an Expected Hypervolume Improvement EHI can be computed using the stochastic representation (136). The EHI is then maximised to determine the design point for the successive evaluation. For the multi-objective EGO, we rely on the **GPareto** package[3] of Binois et al. (137).

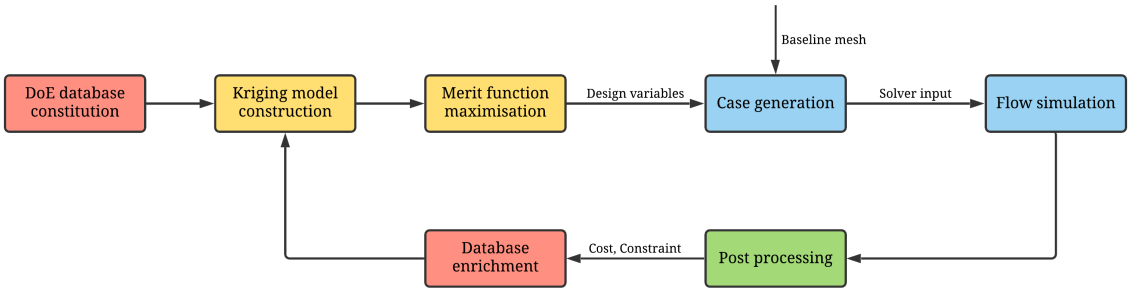


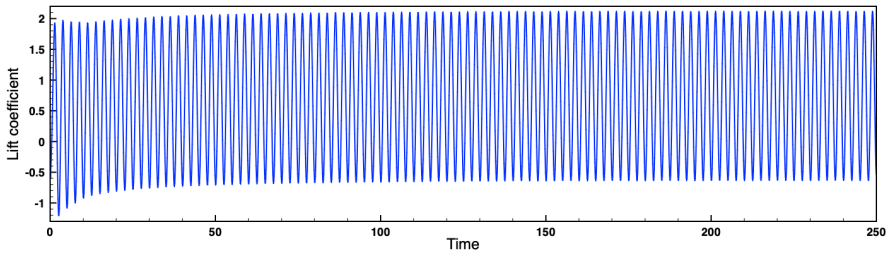
Figure 4.12: Optimisation workflow

4.4.3 Optimisation chain

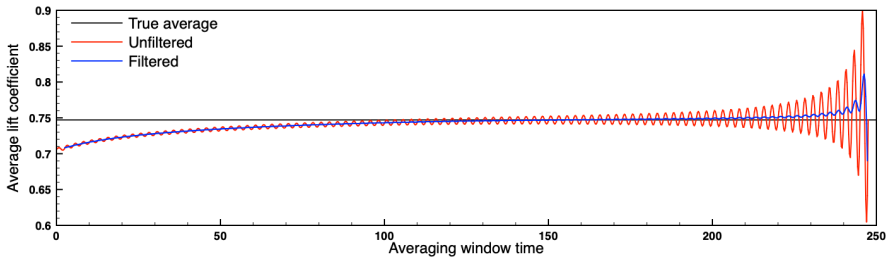
The optimisation algorithm is completed by coupling the EGO technique with the flow solver. An initial set of evaluations is needed to initialise the Kriging models. Therefore some preliminary flow computations have to be realised to produce the initial database. This preparatory phase is known as Design of Experiment (DOE). The distribution of the DOE points in the

[2] `https://cran.r-project.org/web/packages/DiceOptim`
[3] `https://cran.r-project.org/web/packages/GPareto`

design space is computed using the Latin Hypercube Sampling algorithm (138), unless otherwise specified. Once the DOE database is complete, the EGO iterations can start. The Gaussian process surrogate is constructed and the merit function is used to select the design variables. In order to generate the input for the CFD solver, the chosen set of parameters is processed in the case generator. This pre-processing step is necessary to compute the amplitudes **a** of the morphing deformation. The flow simulation is then performed and the aerodynamic forces are post-processed to evaluate the cost and constraint functions. Lastly, the database is enriched with the data from the new evaluation and the updated Kriging models can be computed for the next iteration. The entire workflow is presented in Fig. 4.12. Due to the dense search pattern and the non-monotonic convergence of the EGO algorithm, it is not trivial to determine an effective stopping criterion. The decrease of the maximum value of EI can be used as to estimate the convergence. However, since unsteady simulations are computationally expensive, the total number of evaluations is mainly determined by fixing a reasonable computational budget. In any case, if the convergence of the optimum is not satisfactory, the process can be easily restarted using the database from the last iteration.



(a) Instantaneous lift coefficient



(b) Average lift coefficient

Figure 4.13: Time analysis of the lift coefficient

We underline that a particular attention should be given to the post-processing step. Indeed, the computation of the average aerodynamic coefficients can be an important source of noise. The results contained in section 4.3 are computed considering only the last oscillation period. Since the simulation is sufficiently converged in time and the flow is synchronised with the morphing movement, the values of table 4.2 are time-accurate. However, we cannot assume the flow to be locked-in for all the configurations contained in the design space. Therefore,

we have to compute the average coefficients over several morphing cycles. Considering, for example, the lift coefficient, we evaluate:

$$\bar{C_l} = \frac{1}{T_f - T_w} \int_{T_w}^{T_f} C_l(t)\,\mathrm{d}t, \tag{4.30}$$

where T_f is the final simulation time and T_w is the initial time of the averaging window. In Fig. 4.13 we illustrate the time history of the lift coefficient for the example configuration used for the mesh sensitivity study. The red curve in Fig. 4.13b represents the value of $\bar{C_l}$ computed with eq. (4.30) as a function of T_w. The average is underestimated when $T_w < 130$ because the window contains part of the initial transient. On the other hand, as T_w tends toward T_f, the oscillations of the value of $\bar{C_l}$ become stronger. Values of $T_w \approx 150$ provide the best estimate of the average with the least noise. A further reduction of the oscillation would be desirable to make the evaluation of $\bar{C_l}$ more robust. However, reaching statistical convergence would be too costly. Therefore, we filter the oscillation of $\bar{C_l}$ with an additional averaging step. We consider M equally spaced values of T_w contained in the interval $\Delta = [\bar{T_w} - \delta t, \bar{T_w} - \delta t]$ and we compute M values of $\bar{C_l}$. The final average lift coefficient is the arithmetic average:

$$\bar{C_l} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{T_f - T_{w,i}} \int_{T_{w,i}}^{T_f} C_l(t)\,\mathrm{d}t. \tag{4.31}$$

The blue curve in fig. 4.13b is computed using eq. (4.31) with $M = 20$ and $\delta t = 2.5$ for different values of $\bar{T_w}$. We can observe that the filtered $\bar{C_l}$ provides a more reliable estimate of the true average, provided that $\bar{T_w}$ is sufficiently far from T_f and that the initial transient is discarded.

4.5 Single-objective optimisation: first run

We carry out a single-objective optimisation, considering the following design space:

$$\mathcal{D} = \Big\{ (L, A, f) : L \in [0.3,\, 0.9],\, A \in [0.01,\, 0.06],\, f \in [0.2,\, 1.0] \Big\}. \tag{4.32}$$

The chosen design space permits a large variation of the morphing parameters. In particular, the maximum amplitude is equal to 50% of the airfoil thickness and the maximum oscillation frequency is twice the natural vortex shedding frequency $f_s = 0.4$. The aim of the optimisation is to find the configuration that maximise the average lift. Since the employed EGO algorithm solves minimisation problems, the objective function is the opposite of the average lift coefficient:

$$g(L, A, f) = -\bar{C_l}. \tag{4.33}$$

The drag of the morphing configuration used for the mesh sensitivity study is inferior with respect to the rigid airfoil. We thus choose the following constraint function:

$$h(L, A, f) = \bar{C_d} - C_{d,ref}, \tag{4.34}$$

where $C_{d,ref} = 0.0645$ is the average drag coefficient of the rigid airfoil. The design of experiment contains 8 points and 32 EGO iterations are performed, for a total of 40 evaluations. Each flow simulation is carried out using 80 parallel cores for an average wall-clock time of 8 hours.

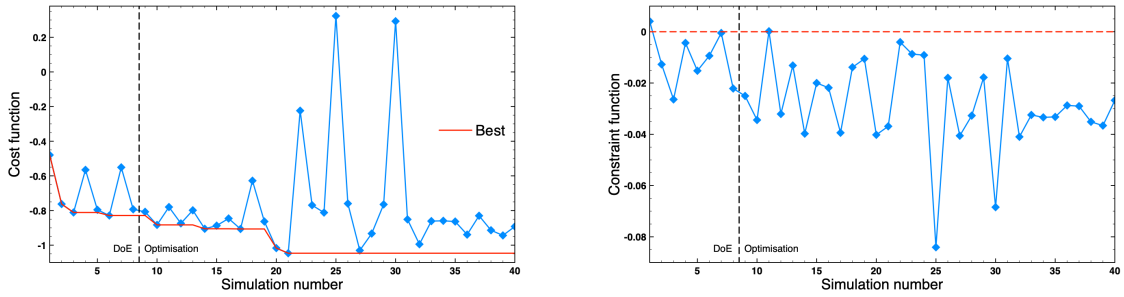


Figure 4.14: First run, DOE and EGO iterations

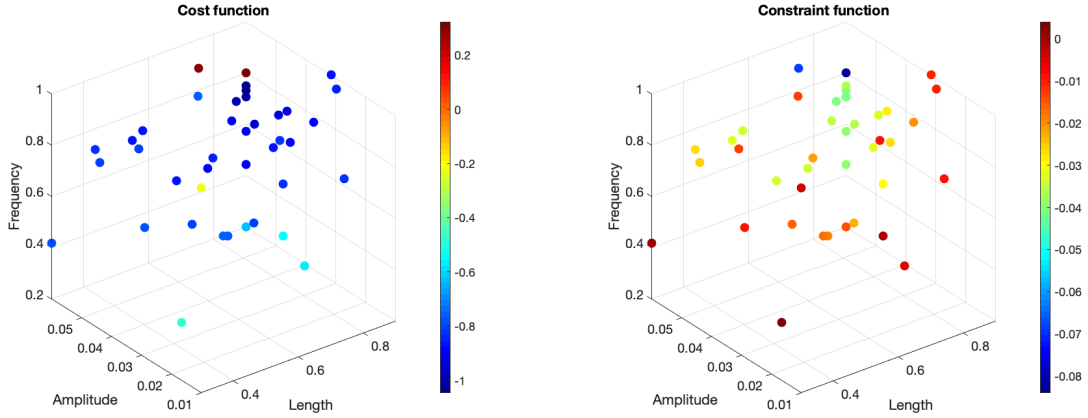


Figure 4.15: First run, scatter plots of cost and constraint functions

In Fig. 4.14 we report the evolution of the objective and constraint functions for all the iterations. We can observe that for nearly all the explored configurations the constraint is satisfied and the average lift is significantly improved with respect to the rigid airfoil. Some promising configurations are already found in the DOE. The optimisation algorithm is thus able to quickly determine the region where the improvement is maximised and, after 13 EGO iterations, the optimal configuration is found, with $\bar{C}_l = 1.0465$ and design parameters equal to (0.9, 0.06, 0.7511). As illustrated in Fig. 4.15, the best values of $\bar{C}_l$ are associated with high morphing amplitudes and lengths. Interesting results are found for frequencies higher than f_s and the optimal frequency range lies in the interval [0.71, 0.76]. The best improvements in terms of average drag are observed in the region of optimal $\bar{C}_l$. For higher frequencies, two configurations characterised by negative values of $\bar{C}_d$ and $\bar{C}_l$ are found. In particular,

$\bar{C}_l = -0.2382$ is obtained for the set of parameters (0.9, 0.06, 0.8019), representing the worst design point.

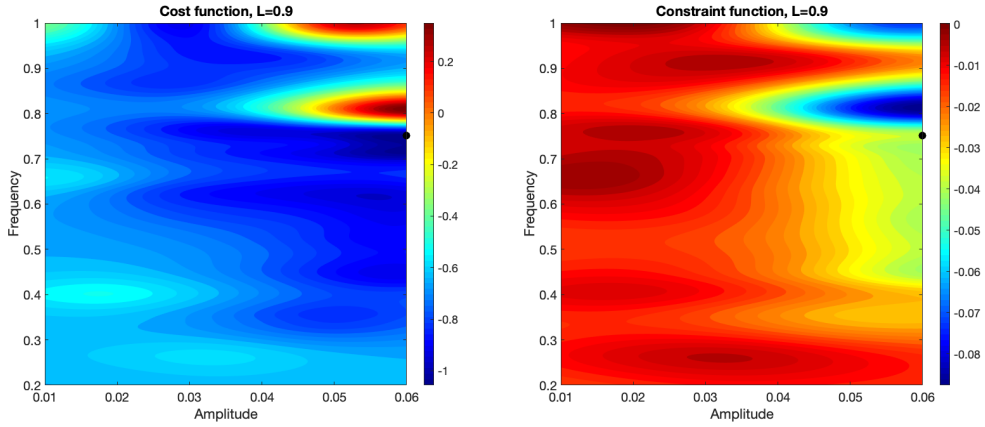


Figure 4.16: First run, contour plots of the mean values of the final Kriging models. The black dot represents the optimal design.

We can remark that the optimal configuration is in close proximity of the worst one. This phenomenon is probably caused by a flow instability. Due to the sudden variation of the objective and constraint functions, the Kriging algorithm struggles to find a correlation between the design points and the hyperparameters θ_i of the covariance kernel tend towards 0. We can observe this behaviour in Fig. 4.16, where the mean value of the final Kriging models are illustrated for $L = 0.9$. Each point in the design space has a small region of influence. As a consequence, the obtained meta-model is not capable of predicting the behaviour of the cost and constraint functions with sufficient fidelity.

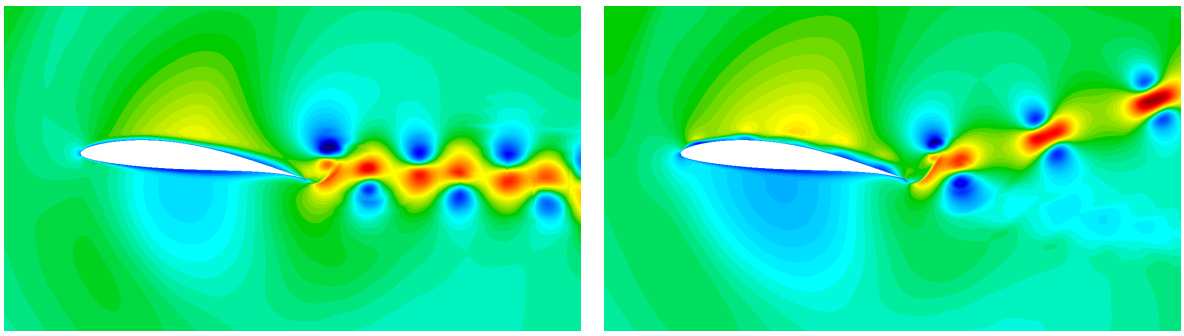
				Optimised mesh			Fine mesh		
N_s	L	A	f	$\bar{C}_d$	$\bar{C}_l$	$\bar{C}_P$	$\bar{C}_d$	$\bar{C}_l$	$\bar{C}_P$
17	0.9	0.06	0.4428	0.0250	0.9066	0.8560	0.0261	0.9214	0.8506
21	0.9	0.06	0.7511	0.0275	1.0465	3.5135	-0.0144	1.9887	3.4720
25	0.9	0.06	0.8019	-0.0197	-0.2382	4.1810	-0.0088	-0.2382	4.1754

Table 4.3: First run, comparison with fine mesh

In order to validate the obtained results, we repeat, for some sample configurations, the simulation using the fine mesh of Fig. 4.7. The chosen configurations are:

- $N_s = 17$: high $\bar{C}_l$, with f close to shedding frequency,
- $N_s = 21$: optimal design,
- $N_s = 25$: worst design.

We then compare the numerical values obtained with the two meshes. The results are summarized in table 4.3. For configuration 17 and 25, the aerodynamic coefficients obtained with the optimised and fine meshes are comparable. On the contrary, a completely different flow field is found for the optimal design when using the fine mesh, as illustrated in Fig. 4.17, resulting in a much higher $\bar{C}_l$ with respect to the coarser mesh used for the optimisation. Since $L = 0.9$, almost the entire surface of the airfoil is deformed, causing several laminar separation bubbles, which are only resolved using the fine mesh. As a consequence of the different boundary layer separation mechanism, the wake structure is significantly altered as well. Lastly, we analyse the time evolution of the lift coefficient for the selected design points. In all three cases, the vortex shedding is synchronised with the morphing oscillation. We report in Fig. 4.18 the curves of C_l for the rigid airfoil and the three configurations of interest. We can observe that the morphing actuation induces strong oscillations of lift and that the minimum value of the lift coefficient is strongly negative.



(a) Optimised mesh (b) Fine mesh

Figure 4.17: First run, optimal configuration, velocity field

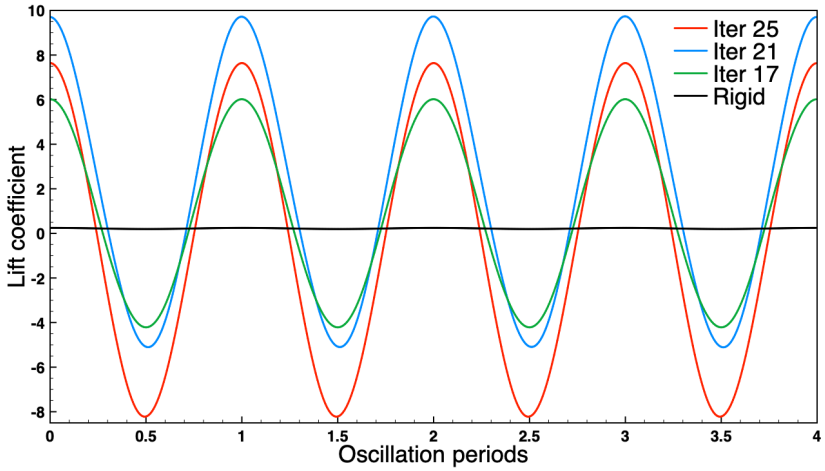


Figure 4.18: First run, comparison with rigid airfoil

To conclude, configurations with high amplitudes and lengths are researched by the EGO algorithm with the aim of finding the best possible average lift value. However, the mentioned region of the design space is characterised by highly non-linear and unstable physical phenomena and the resulting Kriging model is not reliable. Furthermore, the computed optimal design is not robust, since a variation of the flight conditions could induce an abrupt change in the flow physics.

4.6 Single-objective optimisation: second run

In order to avoid the region of chaotic phenomena, we restrict the design space. Particularly, we limit the upper bound of the amplitude to a third of the airfoil thickness and the maximum morphing length to 60% of the chord:

$$\mathcal{D} = \Big\{(L, A, f) : L \in [0.2,\, 0.6],\, A \in [0.005,\, 0.040],\, f \in [0.2,\, 1.0]\Big\}. \tag{4.35}$$

The frequency range is unchanged as well as the objective function:

$$g(L, A, f) = -\bar{C}_l. \tag{4.36}$$

As illustrated in Fig. 4.18, morphing can induce strong oscillations of lift, which are undesirable. Therefore, we force the lift to remain positive during the whole morphing oscillation by employing the following constraint function:

$$h(L, A, f) = -\check{C}_l, \tag{4.37}$$

where $\check{C}_l$ is the minimum value of C_l contained in the averaging window. We perform 40 total simulations, with 8 points for the DOE and 32 EGO iterations.

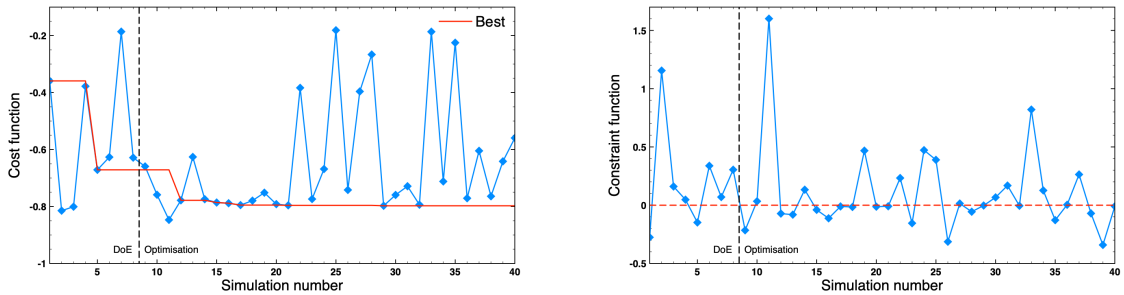


Figure 4.19: Second run, DOE and EGO iterations

The evolution of the cost and constraint functions is illustrated in Fig. 4.19. We can remark that the constraint plays a significant role in the EGO iterations. Among the 8 DOE points, only 2 respect the constraint. For nearly half of the 40 simulations a negative value of $\check{C}_l$ is found. Moreover, the first EGO iterations that present an improvement of the

average lift coefficient do not satisfy the constraint and are therefore discarded. Indeed, the maximum lift configuration is characterised by $\bar{C}_l = 0.8469$ and $\check{C}_l = -1.6007$, meaning that strong lift oscillations occur. As the Kriging models are enriched, the EGO algorithm is able to identify the region where the constraint is satisfied and the cost function is minimised. A first significant improvement is found for the fourth EGO step, then some slightly improved configurations are obtained, until the optimum is found after 21 EGO iterations. The optimal set of morphing parameters is (0.2312, 0.0394, 0.7197), with $\bar{C}_l = 0.7975$ and $\check{C}_l = 0.0027$.

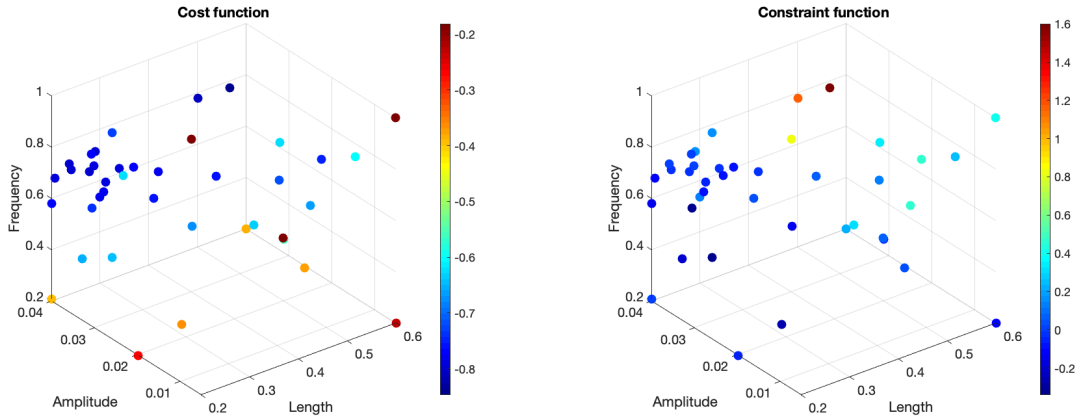


Figure 4.20: Second run, scatter plots of cost and constraint functions

In Fig. 4.20 we can observe the distribution of the investigated configurations in the design space. The maximum value of average lift is found at high values of amplitude and length, where the constraint is not satisfied due to the strong fluctuations in the flow field. The design points added by the EGO algorithm are condensed towards the lower bound of the length interval and the maximum value of the morphing amplitude. Indeed, the best values of $\check{C}_l$ are found for $L = 0.2$. Very few points are present towards the lower bounds of the amplitude and frequency intervals, as the region is characterised by low values of average lift.

In order to better appreciate the effect of the parameters on the objective and constraint functions, we report in Fig. 4.21 the final Kriging models computed with the EGO procedure. We can observe that increasing the amplitude A clearly leads to a decrease of the cost function. The impact of the length L on $\bar{C}_l$ is more subtle, especially for low amplitudes and frequencies. For $A > 0.025$ and $f \in [0.6, 0.9]$, increasing L enhances the average lift. However, the morphing length has a significant influence on the constraint function. Indeed, the parameter L allows to deeply alter the camber of the airfoil. Therefore, large oscillations of lift are induced for high values of L. We can thus observe that, for a given frequency, the maximum $\bar{C}_l$ is found on the corner of maximum values of A and L, where the constraint is violated.

The optimal condition is found decreasing L until $\check{C}_l \approx 0$. For a wide range of frequencies, the lift is improved with respect to the rigid airfoil. The optimal value of f is roughly 1.8 times the shedding frequency. The best values of the constraint functions are localized at the lower bound of L, around $A \approx 0.025$ for frequencies contained in $[0.5, 0.7]$.

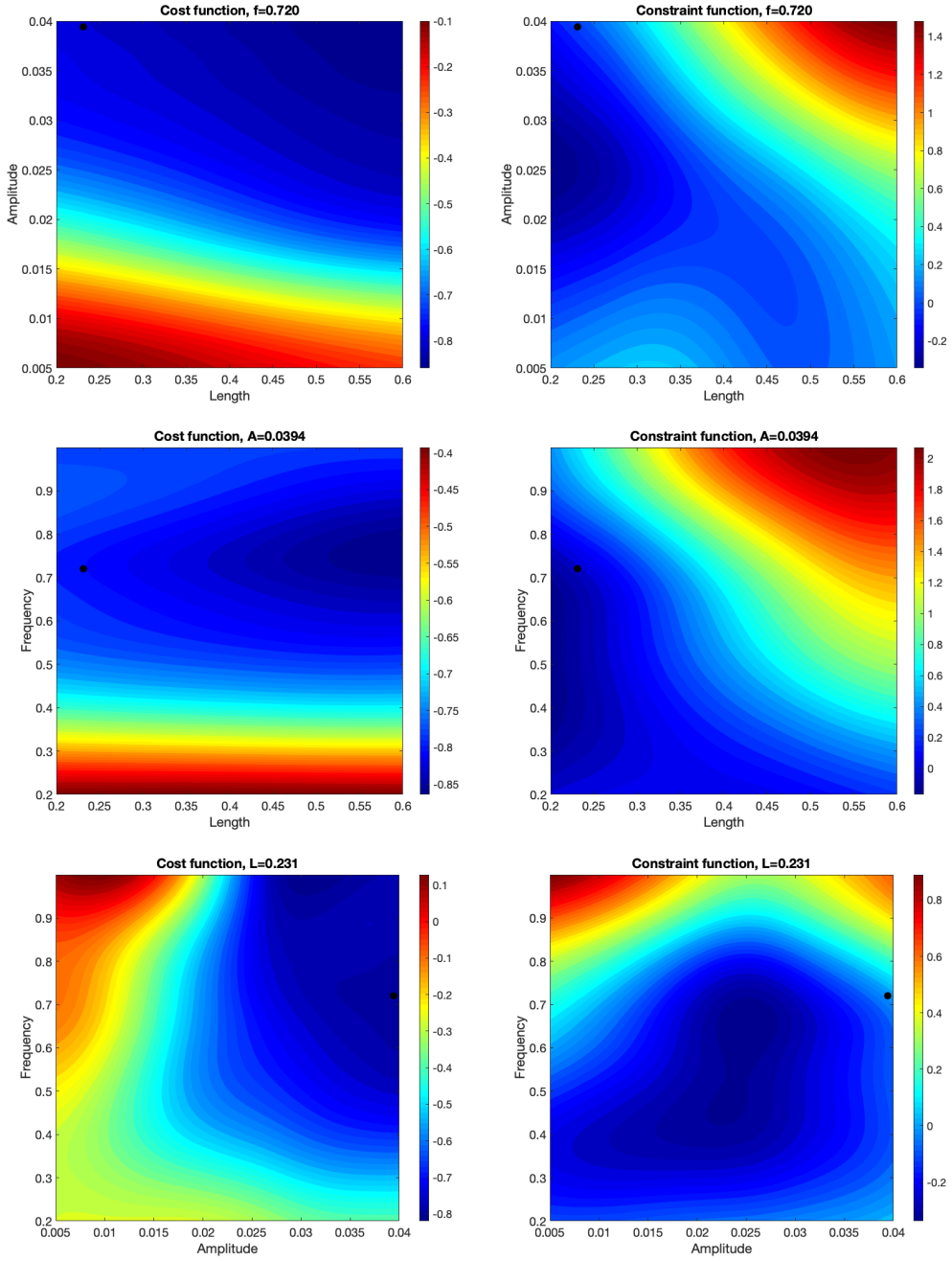


Figure 4.21: Second run, contour plots of the mean values of the final Kriging models. The black dot represents the optimal design.

We then validate the results of the optimisation by performing a comparison with the fine mesh. To this end, we select three relevant configurations:

- $N_s = 26$: best trade-off between $\bar{C}_l$ and $\check{C}_l$,
- $N_s = 29$: optimal design,
- $N_s = 39$: best value of $\check{C}_l$.

We report in table 4.4 the results of the validation test. Some small variations of the aerodynamic coefficients are registered between the two grids. However, for all the selected design points the results obtained with the fine mesh are in line with the predictions of the optimisation loop and, more importantly, the flow physics is not altered when the mesh is refined.

				Optimised mesh				Fine mesh			
N_s	L	A	f	$\bar{C}_d$	$\bar{C}_l$	$\check{C}_l$	$\bar{C}_P$	$\bar{C}_d$	$\bar{C}_l$	$\check{C}_l$	$\bar{C}_P$
26	0.200	0.031	0.662	0.060	0.742	0.313	0.056	0.060	0.741	0.324	0.054
29	0.231	0.039	0.720	0.056	0.798	0.002	0.178	0.057	0.801	0.020	0.174
39	0.200	0.026	0.518	0.066	0.642	0.341	0.014	0.067	0.633	0.342	0.014

Table 4.4: Second run, comparison with fine mesh

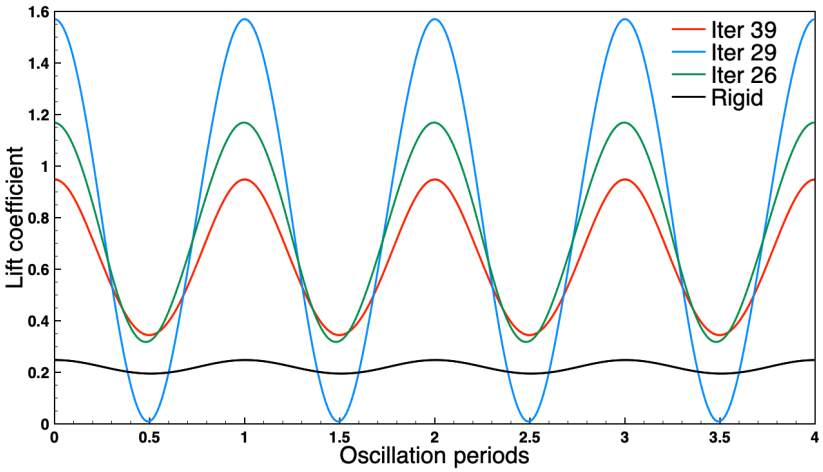
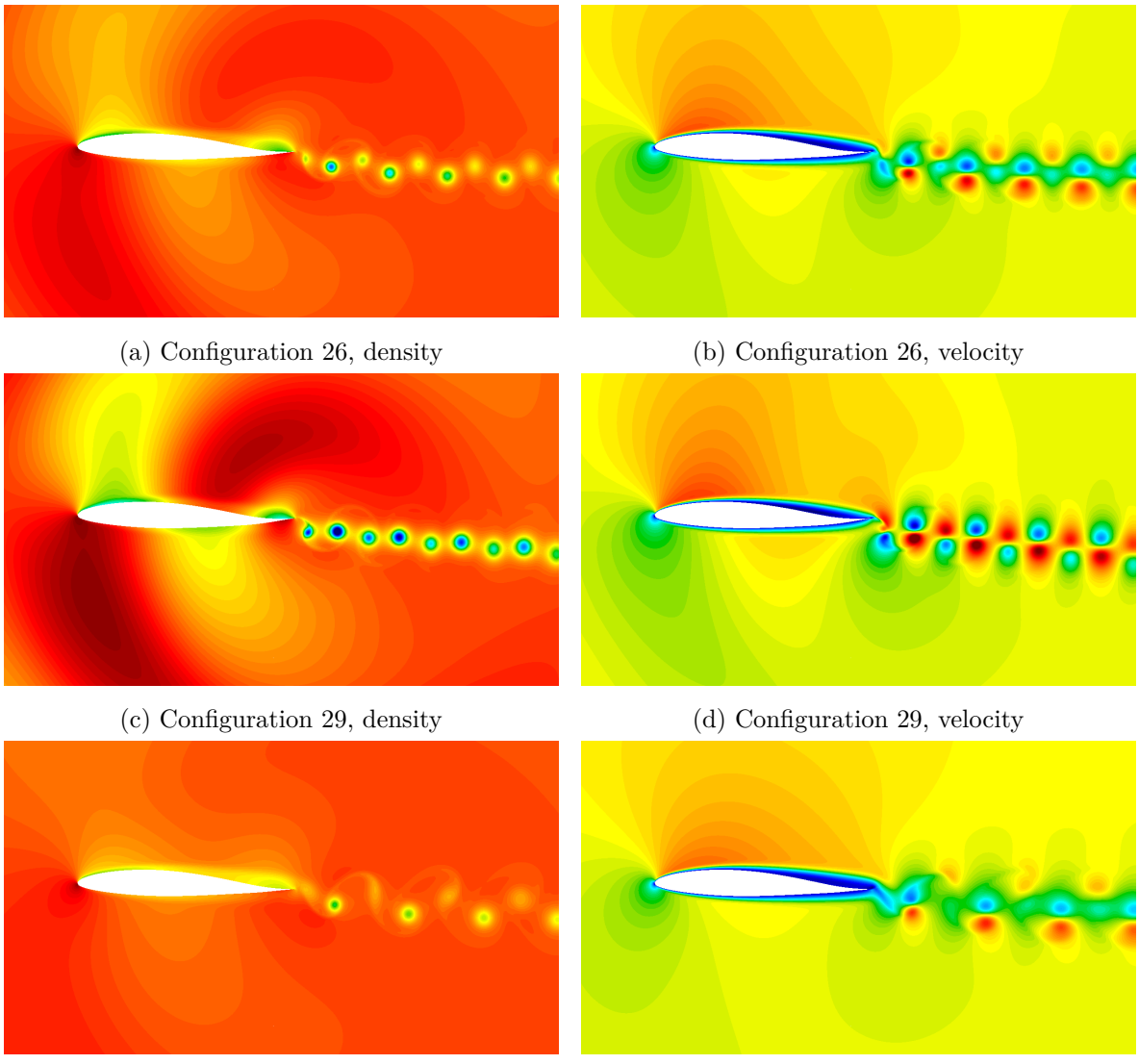


Figure 4.22: Second run, comparison with rigid airfoil

In Fig. 4.22 we illustrate the evolution of the lift coefficient for the selected configurations in comparison with the rigid airfoil. The lock-in phenomenon occurs for the three design points, therefore the lift coefficient oscillates at the same frequency of the morphing actuation. We can observe that the optimal design induces the strongest oscillations of lift, even though the constraint is respected. If we instead consider the configuration 39, characterised by

the best value of $\check{C}_l$ found by the EGO procedure, the value of lift of the morphing airfoil is always greater than the lift of the rigid airfoil. The best trade-off is found for configuration 26. Moreover, we can observe in table 4.4 that the average power required to actuate the morphing decreases for higher $\check{C}_l$. The flow fields of the three selected design points are illustrated in Fig. 4.23. For the sake of comparison, all the snapshots are taken at the peak of the morphing movement and the same colour scale is employed. The wake is formed by the sequence of vortex pairs shed during each morphing oscillation. We can remark that configurations with higher values of $\check{C}_l$ appear to be correlated with stronger disparities between the intensities of the two paired vortices. Among the three configuration, the optimal design is characterised by the most intense density fluctuations.



(a) Configuration 26, density

(b) Configuration 26, velocity

(c) Configuration 29, density

(d) Configuration 29, velocity

(e) Configuration 39, density

(f) Configuration 29, velocity

Figure 4.23: Second run, solution fields

To summarise, the optimisation process is able to identify the region of interest where the $\bar{C}_l$ is improved and determine an optimal design. Moreover, thanks to the dense exploration pattern of the EGO algorithm, some sub-optimal configurations with desirable properties are found. In particular, it is possible to enhance both $\bar{C}_l$ and $\check{C}_l$ with a careful choice of the morphing parameters, as shown by configuration 26 and 39. However, we underline that the best found value of $\check{C}_l$ may not be optimal, since the cost function only considers $\bar{C}_l$. Therefore, we recur to multi-objective optimisation to further explore the design space.

4.7 Multi-objective optimisation

Using the data collected in the second single-objective study as a Design of Experiment, we perform 40 more iterations considering a multi-objective criterion. The first objective is still the maximisation of the average lift coefficient. In order to further control the time evolution of the lift, we maximise $\check{C}_l$ as well. In contrast, high values of $\bar{C}_l$ appears to be characterised by elevated power consumptions of the morphing actuation. It is thus interesting to minimise the average power coefficient $\bar{C}_P$ to find the most energetically efficient morphing design. Therefore, we adopt the following set of cost functions:

$$\begin{cases} g_1(L, A, f) = -\bar{C}_l, \\ g_2(L, A, f) = -\check{C}_l, \\ g_3(L, A, f) = \bar{C}_P. \end{cases} \tag{4.38}$$

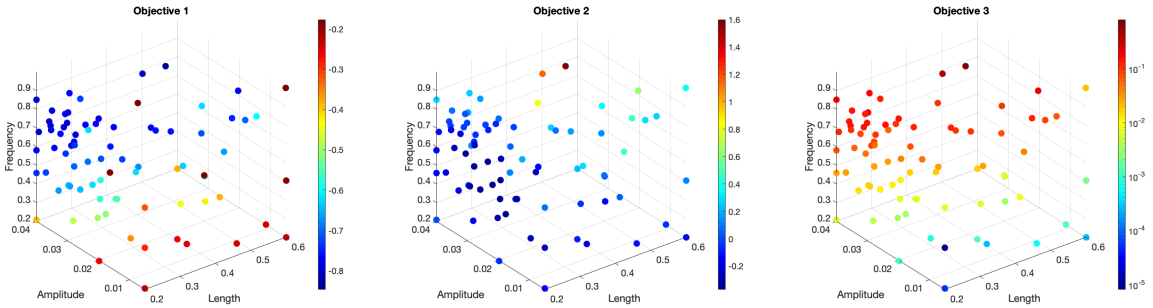


Figure 4.24: Multi-objective optimisation, scatter plots of the objective functions

We report in Fig. 4.24 the scatter plots of the three objective functions for all the morphing configurations investigated by the multi-objective EGO algorithm, together with the points from the DOE. Comparing the distribution of the evaluations in the design space with Fig. 4.20, we can observe the influence of the additional objectives on the optimisation iterations. A cluster of points is added on the $L = 0.2$ plane because the second cost function improves for low values of the morphing length. Similarly, many evaluations with low amplitude and

frequency are present since, intuitively, $\bar{C}_P$ is minimal for such configurations. As a result, no region of the design domain is left totally unexplored and a more complete overview of the physical behaviour of the system is given. We can also observe that the most energetically efficient design is found when the morphing frequency matches the natural vortex shedding frequency.

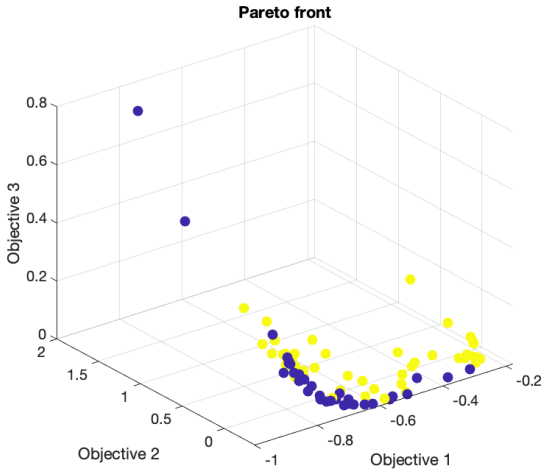


Figure 4.25: Visualisation of the Pareto front

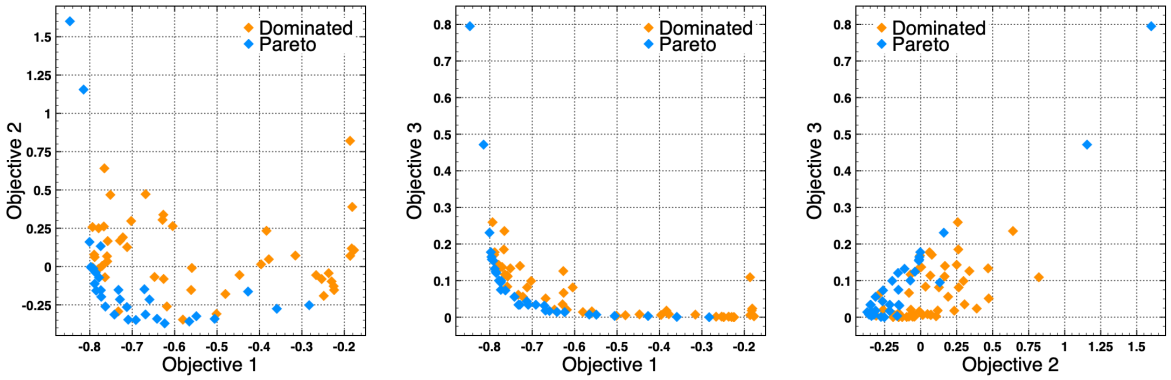


Figure 4.26: Bi-objective scatter plots

The distribution of the evaluations in the objective space is shown in Fig. 4.25. The Pareto set is coloured in violet, whereas the dominated points are represented in yellow. Among the 80 observations, 32 are on the Pareto front. Interestingly, all of the three configurations analysed in Table 4.4 are Pareto optimal. For a better interpretation of the shape of the Pareto front, we compare each pair of objectives in Fig. 4.26. We can clearly observe that two extreme configurations stand out from the rest, with high average lift, but very poor performance in terms of the other two objectives. In the design space, the two said observations are close

to the upper amplitude, length and frequency bound, meaning that the fluid receives a large amount of energy from the deformation of the airfoil, causing large oscillations of lift. We can also observe that the infeasible region constitutes a large portion of the objective space: configurations with low g_2 and high g_3 are not observed and the minimums of g_2 and g_3 almost coincide. It is thus possible to prioritize objectives g_1 and g_2 (or g_1 and g_3) to find an initial compromise and only then use the remaining objective to further refine the design.

				Optimised mesh			Fine mesh		
N_s	L	A	f	$\bar{C}_l$	$\check{C}_l$	$\bar{C}_P$	$\bar{C}_l$	$\check{C}_l$	$\bar{C}_P$
1	0.2134	0.0111	0.4013	0.3591	0.2754	0.00077	0.3539	0.2701	0.00078
11	0.5472	0.0379	0.8108	0.8469	-1.601	0.7950	0.8878	-1.535	0.7844
58	0.2000	0.0219	0.5850	0.6235	0.3710	0.0139	0.6117	0.3672	0.0136
76	0.2000	0.0178	0.4721	0.5057	0.3413	0.0035	0.4947	0.3351	0.0034

Table 4.5: Multi-objective optimisation, comparison with fine mesh

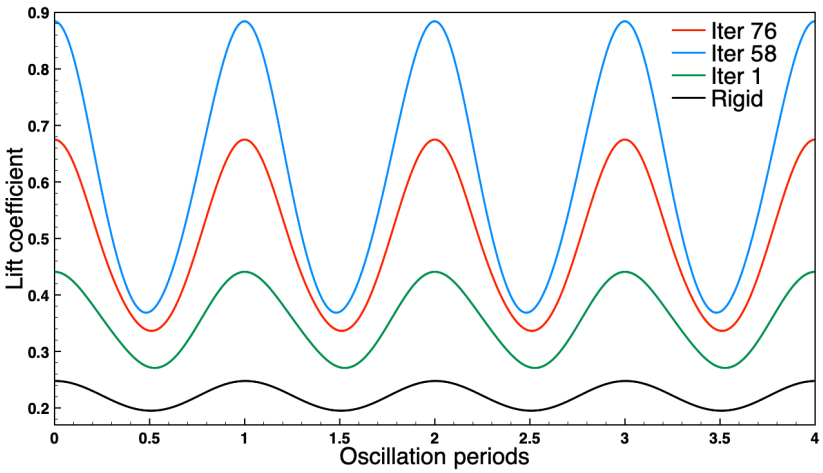


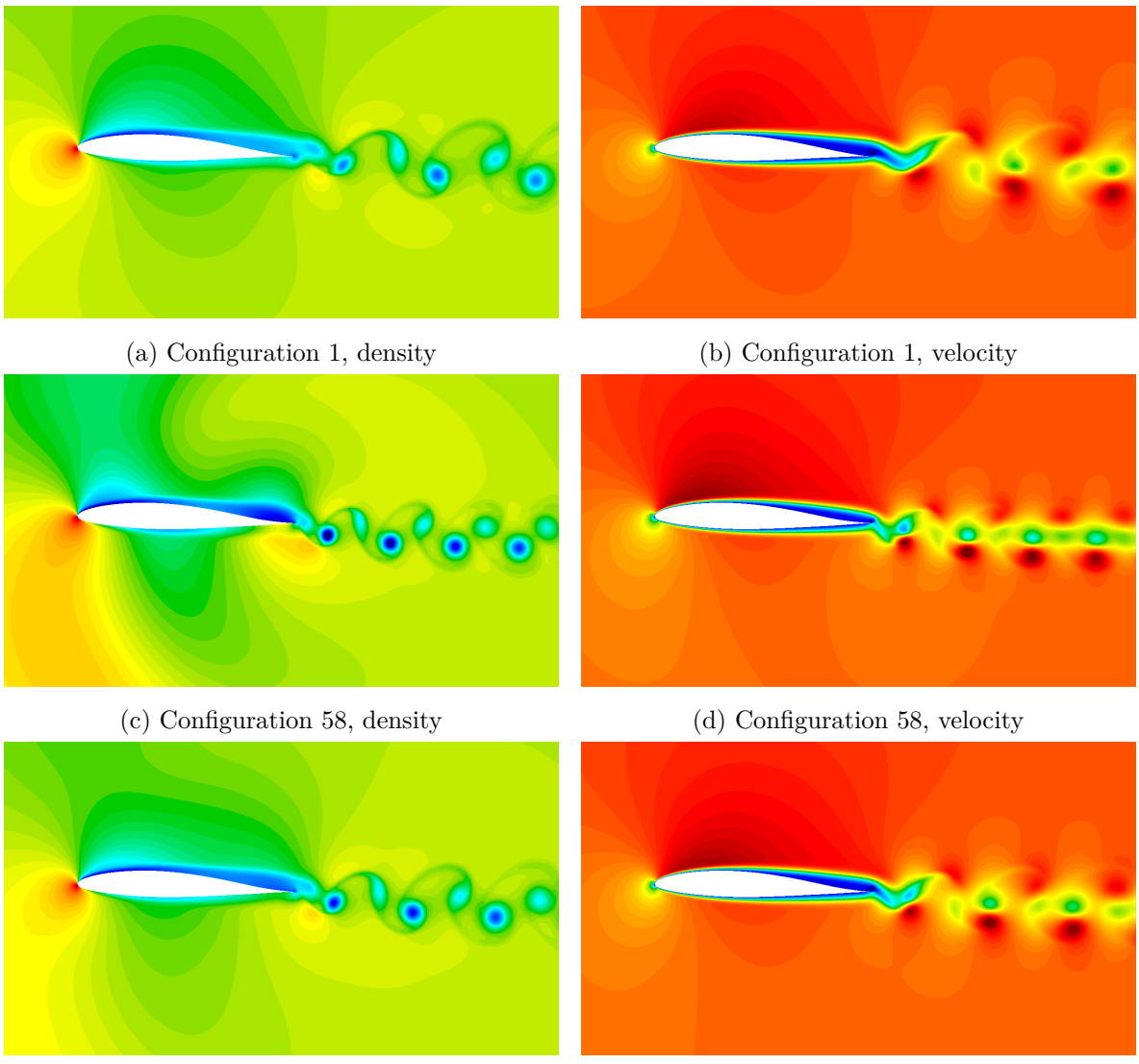
Figure 4.27: Multi-objective optimisation, comparison with rigid airfoil

We then select the four following points from the Pareto set:

- $N_s = 1$: best $\bar{C}_P$ for which $\bar{C}_l$ is improved w.r.t. the rigid airfoil;
- $N_s = 11$: best $\bar{C}_l$, worst $\check{C}_l$ and $\bar{C}_P$;
- $N_s = 58$: best $\check{C}_l$;
- $N_s = 76$: balanced trade-off between the three objectives.

The flow computations for the considered observations are validated using the fine mesh. We can remark in table 4.5 that no significant discrepancies are found when the mesh is refined. In

Fig. 4.27 we present the comparison between the instantaneous lift coefficients of evaluation 1, 58, 76 and the rigid airfoil. We can see that both $\check{C}_l$ and $\bar{C}_l$ are enhanced with respect to the rigid airfoil for all the analysed morphing designs. The smallest improvement is found for $N_S = 1$, which, on the other hand, is extremely efficient in terms of power consumption. Configuration 76 performs well in all the three objectives, requiring 4 times less actuation power than $N_S = 58$, for only 1.25 times less average lift.



(a) Configuration 1, density

(b) Configuration 1, velocity

(c) Configuration 58, density

(d) Configuration 58, velocity

(e) Configuration 76, density

(f) Configuration 76, velocity

Figure 4.28: Multi-objective optimisation, solution fields

Finally, we illustrate the flow fields of the three analysed configurations in Fig. 4.28. It is clearly possible to observe the effect of the morphing frequency on the spacing of the vortex pairs forming the wake. Moreover, a significantly larger recirculation region is found for

$N_S = 1$, probably due to the smaller amplitude and frequency. Once more, the configuration with the strongest oscillations of lift ($N_S = 58$) is associated with larger density and velocity gradients.

4.8 Conclusion

In this chapter we demonstrated the capabilities of the implemented ALE methodology in a flow control application and we exploited the Isogeometric framework to conceive an efficient, reliable and fully automated aerodynamic optimisation chain. To this end, the flow solver has been coupled with an EGO library, which allows to easily treat unconstrained, constrained and multi-objective optimisation problems.

Using the mesh deformation paradigm we proposed in chapter 2, we developed a high-order morphing model, which has been employed to control the characteristics of the laminar boundary layer separation around the NACA 63_1-412 airfoil. Then, a rigorous mesh sensitivity study has been performed to choose the suitable mesh for the aerodynamic optimisation. The selected grid is the result of a compromise between accuracy, robustness and computational cost.

We then proceeded to perform a first single-objective optimisation run, with the aim of improving the average lift of the morphing airfoil without increasing the drag with respect to the rigid configuration. We found that the optimal solution features large lift oscillations and is located in a region of unstable flow phenomena. In order to avoid such undesirable characteristics, we reduced the design space and we imposed a positivity constraint on the instantaneous lift and we carried out another single-objective optimisation run. Thanks to such adjustments, we were able to find a more robust optimal design. To prove the reliability of the optimisation process, some sample configurations were validated using a finer mesh. We observed a significant improvement of the aerodynamic performance with respect to the rigid airfoil and we were able to find some sub-optimal designs with interesting properties.

Lastly, we employed multi-objective optimisation to better explore the design possibilities. In particular, we added an energy minimisation criterion to find the most efficient morphing configuration. We were able to estimate the Pareto front and some trade-off solutions were found and analysed. Overall, the developed methodology proved to be an effective tool for complex aerodynamic design problems.

Conclusion

In this dissertation, we investigated the use of IGA in the context of compressible flows with time-dependent domains. The proposed approach relies on a DG scheme with a CAD-consistent rational Bézier representation. In order to take into account the movement of the computational domain, an ALE formulation with high-order mesh deformations was developed. In particular, the proposed grid velocity algorithm exploits the hierarchical properties of NURBS to achieve smooth curvilinear mesh movements, which can be seamlessly coupled with dynamic adaptive refinement. The implemented ALE methodology has been verified and tested on a selection of benchmark problems in compressible fluid mechanics and proved to be both accurate and robust. The results presented in the present work are limited to bi-dimensional problems. However, the extension of the formulation to tri-dimensional application is currently in development.

As an alternative to the mesh deformation strategy, we developed a sliding grid algorithm that allows to simulate flows with rotating components. Thanks to the rational representation and its geometry manipulation algorithms, a fully conservative high-order discretisation of the sliding interface can be obtained. We showed on an analytical problem that the error introduced by the computation of the numerical fluxes on the sliding interface is negligible. As a consequence, the sliding mesh and the deformation-based ALE solvers are equivalent in terms of accuracy, as proved by the pitching ellipse test case. Finally, a vertical axis wind turbine geometry was considered to present an application with a more complex topology.

Lastly, we demonstrated the potential of IGA for aerodynamic design with an optimisation study of a flow control problem. In particular, we employed a trailing edge morphing actuation to control the laminar boundary layer separation on a reference airfoil. To this end, the previously developed ALE formulation has been coupled with a Bayesian optimisation algorithm. In this context, the Isogeometric framework allows the implementation of a fully automated design chain, whose building blocks share the same representation of the geometry. We employed both single-objective and multi-objective criteria to find the most efficient morphing configuration and we were able to significantly improve the aerodynamic performance with respect to the baseline rigid airfoil.

Starting from the results illustrated in the present manuscript, several research axes can

be explored. The first possibility is the extension of the proposed methodology to multidisciplinary problems, such as aeroelasticity. Indeed, the tight geometry-simulation coupling provided by the Isogeometric framework would allow to easily model the interaction between the structure and fluid. To this end, the ALE-DG flow solver could be combined with a NURBS-based CG scheme for the equations of elasticity. Since one of the limitations of the proposed formulation is the use of explicit time stepping, another perspective is represented by the implementation of an efficient time integration technique. Even though high-order explicit RK were employed, the numerical stability condition imposes very small time steps, especially for high basis degree. In order to be able to simulate turbulent flow problems, implicit time integration is probably required, since the size of boundary layer mesh would lead to prohibitively small time steps. Furthermore, reducing the number of time iterations would have a huge impact in the context of ALE simulations, since the mesh metrics have to be updated for each time iteration. Lastly, the ongoing extension of the numerical scheme to three-dimensional demands the development of a general algorithm to automatically build volume meshes starting from NURBS surfaces. So far, IGA has suffered from the lack of a general methodology to build computational domains for complex topologies. It is thus critical to further enhance the available geometry manipulation tools. An interesting perspective arises from the recent development of triangular Bézier grids (13), thanks to which IGA can be combined with unstructured grid generation techniques. Therefore, it would be possible to generate hybrid grids that would be extremely efficient for CFD applications.

Bibliography

[1] T. Hughes, J. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Computer Methods in Applied Mechanics and Engineering 194 (2005) 4135 – 4195.

[2] G. Farin, Curves and Surfaces for CAGD, 5th ed., Morgan Kaufmann, 2002.

[3] T. Tezduyar, Stabilized finite element formulations for incompressible flow computations, volume 28 of *Advances in Applied Mechanics*, Elsevier, 1991, pp. 1–44.

[4] T. J. R. Hughes, G. Scovazzi, L. P. Franca, Multiscale and Stabilized Methods, American Cancer Society, 2017, pp. 1–64.

[5] T. E. Tezduyar, M. Senga, Stabilization and shock-capturing parameters in SUPG formulation of compressible flows, Computer Methods in Applied Mechanics and Engineering 195 (2006) 1621–1632.

[6] Y. Bazilevs, I. Akkerman, D. Benson, G. Scovazzi, M. Shashkov, Isogeometric analysis of Lagrangian hydrodynamics, Journal of Computational Physics 243 (2013) 224 – 243.

[7] M. Möller, A. Jaeschke, High-Order Isogeometric Methods for Compressible Flows, Springer, 2020, pp. 31–39.

[8] Y. Bazilevs, L. Beirão Da Veiga, J. A. Cottrell, T. J. R. Hughes, G. Sangalli, Isogeometric Analysis: approximation, stability and error estimates for h-refined meshes, Mathematical Models and Methods in Applied Sciences 16 (2006) 1031–1090.

[9] Y. Bazilevs, M.-C. Hsu, I. Akkerman, S. Wright, K. Takizawa, B. Henicke, T. Spielman, T. E. Tezduyar, 3D simulation of wind turbine rotors at full scale. Part i: Geometry modeling and aerodynamics, International Journal for Numerical Methods in Fluids 65 (2011) 207–235.

[10] Y. Otoguro, K. Takizawa, T. E. Tezduyar, Space–time VMS computational flow analysis with isogeometric discretization and a general-purpose NURBS mesh generation method, Computers & Fluids 158 (2017) 189–200.

[11] G. Xu, B. Mourrain, A. Galligo, R. Duvigneau, Constructing analysis-suitable parameterization of computational domain from CAD boundary by variational harmonic method, J. Comput. Physics 252 (2013).

[12] N. Jaxon, X. Qian, Isogeometric analysis on triangulations, Computer Aided Design (2014) 45–57.

[13] L. Engvall, J. Evans, Isogeometric triangular Bernstein-Bézier discretizations: automatic mesh generation and geometrically exact finite-element analysis, Computer Methods in Applied Mechanics and Engineering (2016) 378–407.

[14] S. Xia, X. Qian, Isogeometric analysis with Bézier tetrahedra, Computer Methods in Applied Mechanics and Engineering (2017) 782–816.

[15] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, N. Kroll, G. May, P.-O. Persson, B. van Leer, M. Visbal, High-order CFD methods: current status and perspective, International Journal for Numerical Methods in Fluids 72 (2013) 811–845.

[16] A. Harten, B. Engquist, S. Osher, S. R. Chakravarthy, Uniformly high order accurate essentially non-oscillatory schemes, iii, Journal of Computational Physics 71 (1987) 231–303.

[17] X.-D. Liu, S. Osher, T. Chan, Weighted essentially non-oscillatory schemes, Journal of Computational Physics 115 (1994) 200–212.

[18] K. Bey, J. Oden, A Runge-Kutta discontinuous finite element method for high speed flows, 1991. doi:10.2514/6.1991-1575.

[19] F. Bassi, S. Rebay, A High-Order Accurate Discontinuous Finite Element Method for the Numerical solution of the compressible navier–stokes equations, Journal of Computational Physics 131 (1997) 267 – 279.

[20] I. Lomtev, R. Kirby, G. Karniadakis, A Discontinuous Galerkin ALE Method for Compressible Viscous Flows in Moving Domains, Journal of Computational Physics 155 (1999) 128 – 159.

[21] H. Luo, J. D. Baum, R. Löhner, A discontinuous Galerkin method based on a Taylor basis for the compressible flows on arbitrary grids, Journal of Computational Physics 227 (2008) 8875–8893.

[22] F. Bassi, A. Crivellini, S. Rebay, M. Savini, Discontinuous Galerkin solution of the Reynolds-averaged Navier–Stokes and k–ω turbulence model equations, Computers & Fluids 34 (2005) 507–540.

[23] R. Hartmann, J. Held, T. Leicht, Adjoint-based error estimation and adaptive mesh refinement for the RANS and k–ω turbulence model equations, Journal of Computational Physics 230 (2011) 4268–4284.

[24] A. D. Beck, T. Bolemann, D. Flad, H. Frank, G. J. Gassner, F. Hindenlang, C.-D. Munz, High-order discontinuous Galerkin spectral element methods for transitional and turbulent flow simulations, International Journal for Numerical Methods in Fluids 76 (2014) 522–548.

[25] J.-B. Chapelier, M. de la Llave Plata, F. Renac, E. Lamballais, Evaluation of a high-order discontinuous Galerkin method for the DNS of turbulent flows, Computers & Fluids 95 (2014) 210 – 226.

[26] D. Flad, A. D. Beck, G. Gassner, C. dieter Munz, A Discontinuous Galerkin Spectral Element Method for the direct numerical simulation of aeroacoustics, 2014. doi:10.2514/6.2014-2740.

[27] A. S. Silveira, R. C. Moura, A. F. C. Silva, M. A. Ortega, Higher-order surface treatment for discontinuous Galerkin methods with applications to aerodynamics, International Journal for Numerical Methods in Fluids 79 (2015) 323–342.

[28] R. Costa, S. Clain, R. Loubère, G. J. Machado, High-order accurate finite volume scheme on curved boundaries for the two-dimensional steady-state convection-diffusion equation with Dirichlet condition, Applied Mathematical Modelling 54 (2018).

[29] R. Sevilla, S. Fernandez-Mendez, A. Huerta, NURBS-Enhanced Finite Element Method for Euler equations, International Journal for Numerical Methods in Fluids 57 (2008).

[30] R. Abgrall, C. Dobrzynski, A. Froehly, An example of high order residual distribution scheme using non Lagrange elements : example of Bézier and NURBS., in: WCCM 2010 - the 9th World Congress on Computational Mechanics and 4th Asian Pacific Congress on Computational Mechanics, Sydney, Australia, 2010.

[31] C. Michoski, J. Chan, L. Engvall, J. Evans, Foundations of the blended isogeometric discontinuous Galerkin (BIDG) method, Computer Methods in Applied Mechanics and Engineering 305 (2016) 658–681.

[32] R. Duvigneau, Isogeometric analysis for compressible flows using a Discontinuous Galerkin method, Computer Methods in Applied Mechanics and Engineering 333 (2018) 443 – 461.

[33] R. Duvigneau, CAD-consistent adaptive refinement using a NURBS-based discontinuous Galerkin method, Int. J. for Numerical Methods in Fluids (2020).

[34] Y. Bazilevs, V. Calo, T. Hughes, Y. Zhang, Isogeometric fluid-structure interaction: theory, algorithms, and computations, Comput. Mech. 43 (2008) 3 – 37.

[35] Y. Bazilevs, K. Takizawa, T. Tezduyar, Computational Fluid–Structure Interaction: Methods and Applications, John Wiley & Sons, 2013.

[36] C. S. Peskin, The immersed boundary method, Acta Numerica 11 (2002) 479–517.

[37] M. Bergmann, J. Hovnanian, A. Iollo, An accurate cartesian method for incompressible flows with moving boundaries, Communications in Computational Physics 15 (2014) 1266–1290.

[38] J. Fernández-Fidalgo, S. Clain, L. Ramírez, I. Colominas, X. Nogueira, Very high-order method on immersed curved domains for finite difference schemes with regular Cartesian grids, Computer Methods in Applied Mechanics and Engineering 360 (2020) 112782.

[39] J. Lee, J. Kim, H. Choi, K.-S. Yang, Sources of spurious force oscillations from an immersed boundary method for moving-body problems, Journal of Computational Physics 230 (2011) 2677–2695.

[40] K. Fidkowski, D. Darmofal, An Adaptive Simplex Cut-Cell Method for Discontinuous Galerkin Discretizations of the Navier-Stokes Equations, 2007. doi:10.2514/6.2007-3941.

[41] B. Müller, S. Krämer-Eis, F. Kummer, M. Oberlack, A high-order discontinuous galerkin method for compressible flows with immersed boundaries, International Journal for Numerical Methods in Engineering 110 (2017) 3–30.

[42] D. Krause, F. Kummer, An incompressible immersed boundary solver for moving body flows using a cut cell discontinuous galerkin method, Computers & Fluids 153 (2017) 118–129.

[43] E. A. Napoli, Innovative numerical methods to investigate internal flow in screw compressors, 2019.

[44] D. Kamensky, M.-C. Hsu, D. Schillinger, J. A. Evans, A. Aggarwal, Y. Bazilevs, M. S. Sacks, T. J. Hughes, An immersogeometric variational framework for fluid–structure interaction: application to bioprosthetic heart valves, Computer Methods in Applied Mechanics and Engineering 284 (2015) 1005–1053.

[45] F. Xu, Y. Bazilevs, M.-C. Hsu, Immersogeometric analysis of compressible flows with application to aerodynamic simulation of rotorcraft, Mathematical Models and Methods in Applied Sciences 29 (2019) 905–938.

[46] R. Loubère, J. Ovadia, R. Abgrall, A Lagrangian discontinuous Galerkin-type method on unstructured meshes to solve hydrodynamics problems, International Journal for Numerical Methods in Fluids 44 (2004) 645–663.

[47] F. Vilar, P.-H. Maire, R. Abgrall, A discontinuous Galerkin discretization for solving the two-dimensional gas dynamics equations written under total Lagrangian formulation on general unstructured grids, Journal of Computational Physics 276 (2014) 188 – 234.

[48] X. Liu, N. R. Morgan, D. E. Burton, A Lagrangian discontinuous Galerkin hydrodynamic method, Computers & Fluids 163 (2018) 68–85.

[49] V. A. Dobrev, T. E. Ellis, T. V. Kolev, R. N. Rieben, Curvilinear finite elements for lagrangian hydrodynamics, International Journal for Numerical Methods in Fluids 65 (2011) 1295–1310.

[50] B. Boutin, E. Deriaz, P. Hoch, P. Navaro, Extension of ALE methodology to unstructured conical meshes, ESAIM Proceedings 32 (2011).

[51] P.-O. Persson, J. Bonet, J. Peraire, Discontinuous Galerkin solution of the Navier–Stokes equations on deformable domains, Computer Methods in Applied Mechanics and Engineering 198 (2009) 1585 – 1595.

[52] V.-T. Nguyen, An arbitrary Lagrangian-Eulerian discontinuous Galerkin method for simulations of flows over variable geometries, Journal of Fluids and Structures 26 (2010) 312, 329.

[53] W. Boscheri, M. Dumbser, High order accurate direct Arbitrary-Lagrangian-Eulerian ADER-WENO finite volume schemes on moving curvilinear unstructured meshes, Computers & Fluids 136 (2016) 48–66.

[54] W. Boscheri, M. Dumbser, Arbitrary-Lagrangian–Eulerian Discontinuous Galerkin schemes with a posteriori subcell finite volume limiting on moving unstructured meshes, Journal of Computational Physics 346 (2017) 449 – 479.

[55] W. Anderson, V. Dobrev, T. Kolev, R. Rieben, V. Tomov, High-order multi-material ALE hydrodynamics, SIAM J. Sci. Comput. 40 (2018).

[56] R. Löhner, Adaptive remeshing for transient problems, Computer Methods in Applied Mechanics and Engineering 75 (1989) 195–214.

[57] P. H. Saksono, W. G. Dettmer, D. Perić, An adaptive remeshing strategy for flows with moving boundaries and fluid–structure interaction, International Journal for Numerical Methods in Engineering 71 (2007) 1009–1050.

[58] R. Loubère, P.-H. Maire, M. Shashkov, J. Breil, S. Galera, ReALE: A reconnection-based arbitrary-Lagrangian–Eulerian method, Journal of Computational Physics 229 (2010) 4724 – 4761.

[59] J. van der Vegt, H. van der Ven, Space–Time Discontinuous Galerkin Finite Element Method with Dynamic Grid Motion for Inviscid Compressible Flows: I. General Formulation, Journal of Computational Physics 182 (2002) 546 – 585.

[60] L. Wang, P.-O. Persson, A high-order discontinuous Galerkin method with unstructured space–time meshes for two-dimensional compressible flows on domains with large deformations, Computers & Fluids 118 (2015) 53–68.

[61] E. Gaburro, W. Boscheri, S. Chiocchetti, C. Klingenberg, V. Springel, M. Dumbser, High order direct Arbitrary-Lagrangian-Eulerian schemes on moving Voronoi meshes with topology changes, Journal of Computational Physics 407 (2020) 109 – 167.

[62] R. Steijl, G. Barakos, Sliding mesh algorithm for CFD analysis of helicopter rotor–fuselage aerodynamics, International Journal for Numerical Methods in Fluids 58 (2008) 527–549.

[63] W. M. Chan, Overset grid technology development at NASA Ames research center, Computers & Fluids 38 (2009) 496–503.

[64] G. Wang, F. Duchaine, D. Papadogiannis, I. Duran, S. Moreau, L. Y. Gicquel, An overset grid method for large eddy simulation of turbomachinery stages, Journal of Computational Physics 274 (2014) 333–355.

[65] N. Gourdain, Prediction of the unsteady turbulent flow in an axial compressor stage. Part 1: Comparison of unsteady RANS and LES with experiments, Computers & Fluids 106 (2015) 119–129.

[66] M. C. Galbraith, J. A. Benek, P. D. Orkwis, M. G. Turner, A discontinuous Galerkin Chimera scheme, Computers & Fluids 98 (2014) 27–53.

[67] M. C. Galbraith, J. A. Benek, P. D. Orkwis, M. G. Turner, A discontinuous Galerkin scheme for Chimera overset viscous meshes on curved geometries, Computers & Fluids 119 (2015) 176–196.

[68] M. J. Brazell, J. Sitaraman, D. J. Mavriplis, An overset mesh approach for 3D mixed element high-order discretizations, Journal of Computational Physics 322 (2016) 33–51.

[69] J. Crabill, F. Witherden, A. Jameson, A parallel direct cut algorithm for high-order overset methods with application to a spinning golf ball, Journal of Computational Physics 374 (2018) 692–723.

[70] Y. Bazilevs, T. J. R. Hughes, Nurbs-based isogeometric analysis for the computation of flows about rotating components, Computational Mechanics 43 (2008) 143–150.

[71] E. Ferrer, R. H. Willden, A high order Discontinuous Galerkin – Fourier incompressible 3D Navier–Stokes solver with rotating sliding meshes, Journal of Computational Physics 231 (2012) 7037–7056.

[72] J. Dürrwächter, M. Kurz, P. Kopper, D. Kempf, C.-D. Munz, A. Beck, An efficient sliding mesh interface method for high-order discontinuous Galerkin schemes, Computers & Fluids 217 (2021) 104825.

[73] J. S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, 1st ed., Springer, 2007.

[74] L. Piegl, W. Tiller, The NURBS Book, second ed., Springer-Verlag, New York, NY, USA, 1996.

[75] B. Cockburn, C.-W. Shu, The Local Discontinuous Galerkin Method for Time-Dependent Convection-Diffusion Systems, SIAM Journal on Numerical Analysis 35 (1998) 2440–2463.

[76] P.-O. Persson, J. Peraire, Sub-Cell Shock Capturing for Discontinuous Galerkin Methods, 2006.

[77] G. E. Barter, D. L. Darmofal, Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation, Journal of Computational Physics 229 (2010) 1810 – 1827.

[78] J. Qiu, C.-W. Shu, Runge-Kutta Discontinuous Galerkin Method Using WENO Limiters, SIAM Journal on Scientific Computing 26 (2005) 907–929.

[79] M. Dumbser, R. Loubère, A simple robust and accurate a posteriori sub-cell finite volume limiter for the discontinuous Galerkin method on unstructured meshes, Journal of Computational Physics 319 (2016) 163 – 199.

[80] G. Mengaldo, D. D. Grazia, F. Witherden, A. Farrington, P. Vincent, S. Sherwin, J. Peiro, A Guide to the Implementation of Boundary Conditions in Compact High-Order Methods for Compressible Aerodynamics, 2014.

[81] J. Butcher, A history of Runge-Kutta methods, Applied Numerical Mathematics 20 (1996) 247–260.

[82] S. Gottlieb, C.-W. Shu, E. Tadmor, Strong Stability-Preserving High-Order Time Discretization Methods, SIAM Review 43 (2001) 89, 112.

[83] B. Cockburn, C.-W. Shu, Runge–kutta discontinuous galerkin methods for convection-dominated problems, Journal of Scientific Computing 16 (2001) 173–261.

[84] H. M. Blackburn, R. D. Henderson, A study of two-dimensional flow past an oscillating cylinder, Journal of Fluid Mechanics 385 (1999) 255–286.

[85] X.-Y. Lu, C. Dalton, Calculation of the timing of vortex formation from an oscillating cylinder, Journal of Fluids and Structures 10 (1996) 527 – 541.

[86] A. Roshko, On the Development of Turbulent Wakes from Vortex Streets, National Advisory Committee for Aeronautics (NACA), 1954.

[87] J. Donea, A. Huerta, J.-P. Ponthot, A. Rodríguez-Ferran, Arbitrary Lagrangian–Eulerian Methods, 2004.

[88] C. S. Venkatasubban, A new finite element formulation for ALE (Arbitrary Lagrangian Eulerian) compressible fluid mechanics, International Journal of Engineering Science 33 (1995) 1743 – 1762.

[89] H. Guillard, C. Farhat, On the significance of the geometric conservation law for flow computations on moving meshes, Computer Methods in Applied Mechanics and Engineering 190 (2000) 1467 – 1482.

[90] M. R. Visbal, D. V. Gaitonde, On the Use of Higher-Order Finite-Difference Schemes on Curvilinear and Deforming Meshes, Journal of Computational Physics 181 (2002) 155 – 185.

[91] R. Löhner, C. Yang, Improved ALE mesh velocities for moving bodies, Communications in Numerical Methods in Engineering 12 (1996) 599–608.

[92] A. Harten, P. D. Lax, B. van Leer, On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws, SIAM Review 25 (1983) 35–61.

[93] C. Williamson, A. Roshko, Vortex formation in the wake of an oscillating cylinder, Journal of Fluids and Structures 2 (1988) 355 – 381.

[94] X. Ren, K. Xu, W. Shyy, A multi-dimensional high-order DG-ALE method based on gas-kinetic theory with application to oscillating bodies, Journal of Computational Physics 316 (2016) 700 – 720.

[95] R. H. Landon, NACA 0012 oscillatory and transient pitching, Advisory Group for Aerospace Research and Development (AGARD), 1982.

[96] A. Seiler, B. Jüttler, Reparameterization and Adaptive Quadrature for the Isogeometric Discontinuous Galerkin Method, in: M. Floater, T. Lyche, M.-L. Mazure, K. Mørken, L. L. Schumaker (Eds.), Mathematical Methods for Curves and Surfaces, Springer International Publishing, 2017, pp. 251–269.

[97] C. Mavripilis, Nonconforming discretizations and a posteriori error estimators for adaptive spectral element techniques, Ph.D. thesis, Massachusetts Institute of Technology, 1989.

[98] D. A. Kopriva, A conservative staggered-grid Chebyshev multidomain method for compressible flows. ii. A semi-structured method, Journal of Computational Physics 128 (1996) 475–488.

[99] B. Zhang, C. Liang, A simple, efficient, and high-order accurate curved sliding-mesh interface approach to spectral difference method on coupled rotating and stationary domains, Journal of Computational Physics 295 (2015) 147–160.

[100] E. Laughton, G. Tabor, D. Moxey, A comparison of interpolation techniques for non-conformal high-order discontinuous galerkin methods, Computer Methods in Applied Mechanics and Engineering 381 (2021) 113820.

[101] R. Patil, L. Daróczy, G. Janiga, D. Thévenin, Large eddy simulation of an H-Darrieus rotor, Energy 160 (2018) 388–398.

[102] L. Daróczy, G. Janiga, K. Petrasch, M. Webner, D. Thévenin, Comparative analysis of turbulence models for the aerodynamic simulation of H-Darrieus rotors, Energy 90 (2015) 680–690.

[103] M. Falsafioon, S. Arabi, R. Camarero, F. Guibault, Comparison of Two Mesh Smoothing Techniques for Unstructured Grids, 2014.

[104] W. A. Wall, M. A. Frenzel, C. Cyron, Isogeometric structural shape optimization, Computer Methods in Applied Mechanics and Engineering 197 (2008) 2976–2988.

[105] N. D. Manh, A. Evgrafov, A. R. Gersborg, J. Gravesen, Isogeometric shape optimization of vibrating membranes, Computer Methods in Applied Mechanics and Engineering 200 (2011) 1343–1353.

[106] T. Hirschler, R. Bouclier, A. Duval, T. Elguedj, J. Morlier, Isogeometric sizing and shape optimization of thin structures with a solid-shell approach, Structural and Multidisciplinary Optimization 59 (2019) 767–785.

[107] P. Nørtoft, J. Gravesen, Isogeometric shape optimization in fluid mechanics, Structural and Multidisciplinary Optimization 48 (2013) 909–925.

[108] E. Gillebaart, R. De Breuker, Low-fidelity 2D isogeometric aeroelastic analysis and optimization method with application to a morphing airfoil, Computer Methods in Applied Mechanics and Engineering 305 (2016) 512–536.

[109] K. Wang, S. Yu, Z. Wang, R. Feng, T. Liu, Adjoint-based airfoil optimization with adaptive isogeometric discontinuous Galerkin method, Computer Methods in Applied Mechanics and Engineering 344 (2019) 602–625.

[110] M. F. Platzer, K. D. Jones, J. Young, J. C. S. Lai, Flapping wing aerodynamics: Progress and challenges, AIAA Journal 46 (2008) 2136–2149.

[111] V. Joshi, R. K. Jaiman, C. Ollivier-Gooch, A variational flexible multibody formulation for partitioned fluid–structure interaction: Application to bat-inspired drones and unmanned air-vehicles, Computers & Mathematics with Applications 80 (2020) 2707–2737.

[112] S. Barbarino, O. Bilgen, R. M. Ajaj, M. I. Friswell, D. J. Inman, A review of morphing aircraft, Journal of Intelligent Material Systems and Structures 22 (2011) 823–877.

[113] J. Fincham, M. Friswell, Aerodynamic optimisation of a camber morphing aerofoil, Aerospace Science and Technology 43 (2015) 245–255.

[114] D. A. Burdette, J. R. Martins, Design of a transonic wing with an adaptive morphing trailing edge via aerostructural optimization, Aerospace Science and Technology 81 (2018) 192–203.

[115] W. Kang, M. Xu, W. Yao, J. Zhang, Lock-in mechanism of flow over a low-reynolds-number airfoil with morphing surface, Aerospace Science and Technology 97 (2020) 105647.

[116] N. Simiriotis, K. Diakakis, G. Jodin, F. Kramer, A. Marouf, Y. Hoarau, J.-F. Rouchon, G. Tzabiras, M. Braza, Synthesis on High-Fidelity Numerical simulation of a morphing A320 wing in subsonic speeds and sensitivity evaluation, 2019. doi:10.2514/6.2019-2911.

[117] N. Simiriotis, G. Jodin, A. Marouf, P. Elyakime, Y. Hoarau, J. Hunt, J. Rouchon, M. Braza, Morphing of a supercritical wing by means of trailing edge deformation and vibration at high Reynolds numbers: Experimental and numerical investigation, Journal of Fluids and Structures 91 (2019) 102676.

[118] J.-B. Tô, N. Simiriotis, A. Marouf, D. Szubert, I. Asproulias, D. Zilli, Y. Hoarau, J. Hunt, M. Braza, Effects of vibrating and deformed trailing edge of a morphing supercritical airfoil in transonic regime by numerical simulation at high reynolds number, Journal of Fluids and Structures 91 (2019) 102595.

[119] W. Timmer, An Overview of NACA 6-Digit Airfoil Series Characteristics with Reference to Airfoils for Large Wind Turbine Blades, 2008. doi:10.2514/6.2009-268.

[120] O. Pironneau, On optimum design in fluid mechanics, Journal of Fluid Mechanics 64 (1974) 97–110.

[121] S. Nadarajah, A. Jameson, A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization, 2000. doi:10.2514/6.2000-667.

[122] R. Lohner, O. Soto, C. Yang, An Adjoint-Based Design Methodology for CFD Optimization Problems, 2003. doi:10.2514/6.2003-299.

[123] A. Belme, Unsteady aerodynamics and adjoint method, Ph.D. thesis, Université Nice Sophia Antipolis, 2011.

[124] C. Corre, T. Renaud, A. Lerat, Transonic flow control using a navier-stokes solver and a multi-objective genetic algorithm, in: IUTAM Symposium Transsonicum IV, Springer, 2003, pp. 297–302.

[125] D. R. Jones, M. Schonlau, W. J. Welch, Efficient Global Optimization of expensive black-box functions, Journal of Global Optimization 13 (1998) 455–492.

[126] D. R. Jones, A taxonomy of global optimization methods based on response surfaces, Journal of Global Optimization 21 (2001) 345–383.

[127] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, N. de Freitas, Taking the human out of the loop: A review of Bayesian optimization, Proceedings of the IEEE 104 (2016) 148–175.

[128] S. Skinner, H. Zare-Behtash, State-of-the-art in aerodynamic shape optimisation methods, Applied Soft Computing 62 (2018) 933–962.

[129] R. Duvigneau, P. Chandrashekar, Kriging-based optimization applied to flow control, International Journal for Numerical Methods in Fluids 69 (2012) 1701–1714.

[130] J. Labroquère, Optimization of active control devices for separated turbulent flows, Ph.D. thesis, Université Nice Sophia Antipolis, 2014.

[131] N. Bartoli, T. Lefebvre, S. Dubreuil, R. Olivanti, R. Priem, N. Bons, J. Martins, J. Morlier, Adaptive modeling strategy for constrained global optimization with application to aerodynamic wing design, Aerospace Science and Technology 90 (2019) 85–102.

[132] O. Roustant, D. Ginsbourger, Y. Deville, DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by Kriging-based metamodeling and optimization, Journal of Statistical Software, Articles 51 (2012) 1–55.

[133] A. T orn, A. Zilinskas, Global optimization, 1st ed., Springer, 1989.

[134] F. Duchaine, T. Morel, L. Y. M. Gicquel, Computational-Fluid-Dynamics-based Kriging optimization tool for aeronautical combustion chambers, AIAA Journal 47 (2009) 631–645.

[135] E. Vazquez, J. Bect, Convergence properties of the expected improvement algorithm with fixed mean and covariance functions, Journal of Statistical Planning and Inference 140 (2010) 3088–3095.

[136] M. T. M. Emmerich, A. H. Deutz, J. W. Klinkenberg, Hypervolume-based expected improvement: Monotonicity properties and exact computation, in: 2011 IEEE Congress of Evolutionary Computation (CEC), 2011, pp. 2147–2154. doi:10.1109/CEC.2011.5949880.

[137] M. Binois, V. Picheny, GPareto: An R package for Gaussian-process-based multi-objective optimization and analysis, Journal of Statistical Software, Articles 89 (2019) 1–30.

[138] M. D. McKay, R. J. Beckman, W. J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (1979) 239–245.